SIDDHARTH INSTITUTE OF ENGINEERING AND TECHNOLOGY::PUTTUR

(AUTONOMOUS)

Siddharth Nagar, Narayanavanam Road–517583

Subject with Code: Data Warehousing and Data Mining(20CS0517) **Course & Branch:** CSE

Year & Sem: III B.Tech & I Sem **Regulation:** R20

UNIT–I
INTRODUCTION TO DATA MINING AND DATA PREPROCESSING

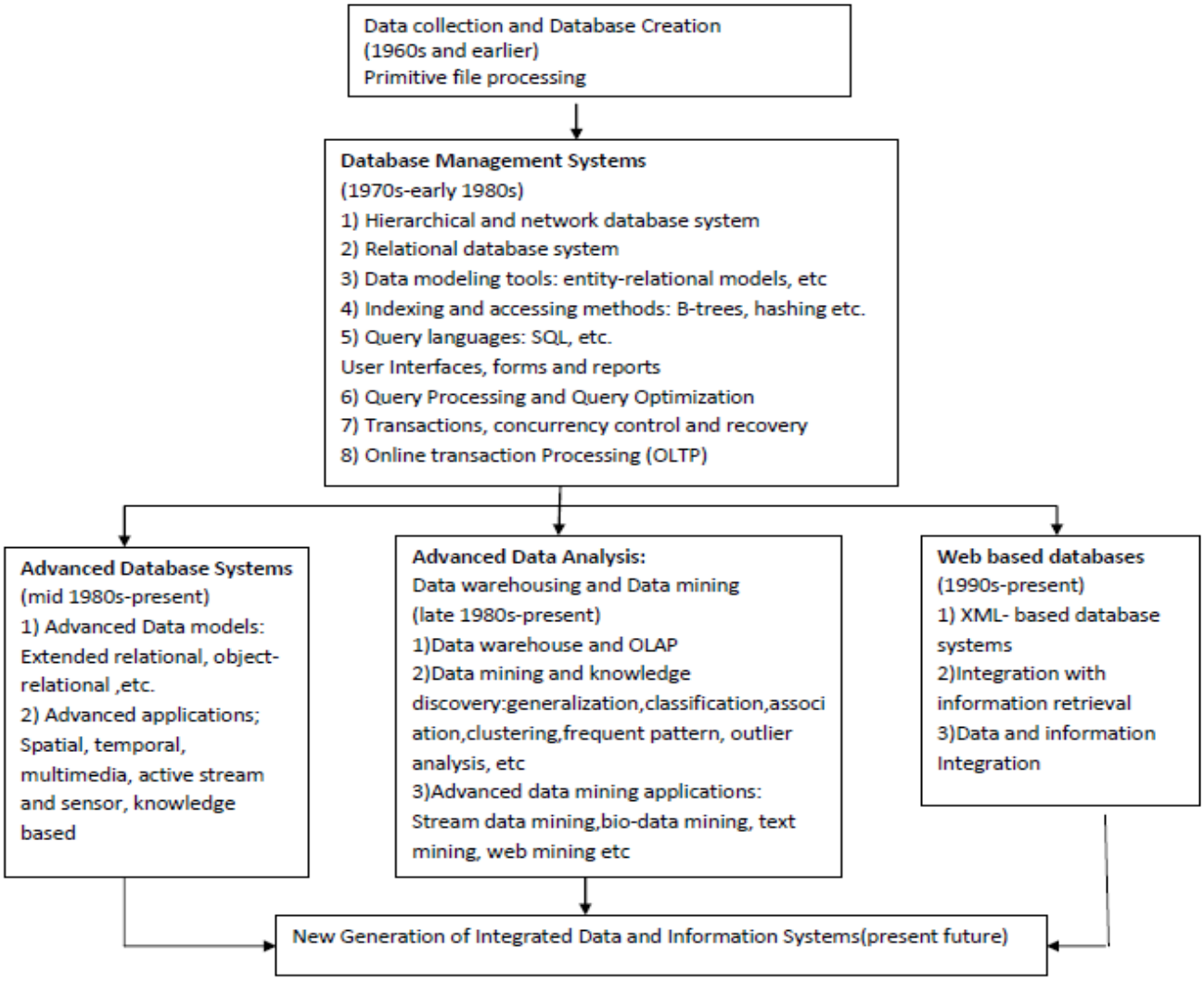
1. a. Define Data mining? What are all points to be discussed to motivated datamining? [6M]

What is data mining?

- Data mining refers to extracting or mining" knowledge from large amounts of data.
- There are many other terms related to data mining, such as knowledge mining, knowledge extraction, data/pattern analysis, data archaeology, and data dredging.
- Many people treat data mining as a synonym for another popularly used term, "Knowledge Discovery in Databases", or KDD.

What are all points to be discussed to motivated datamining?

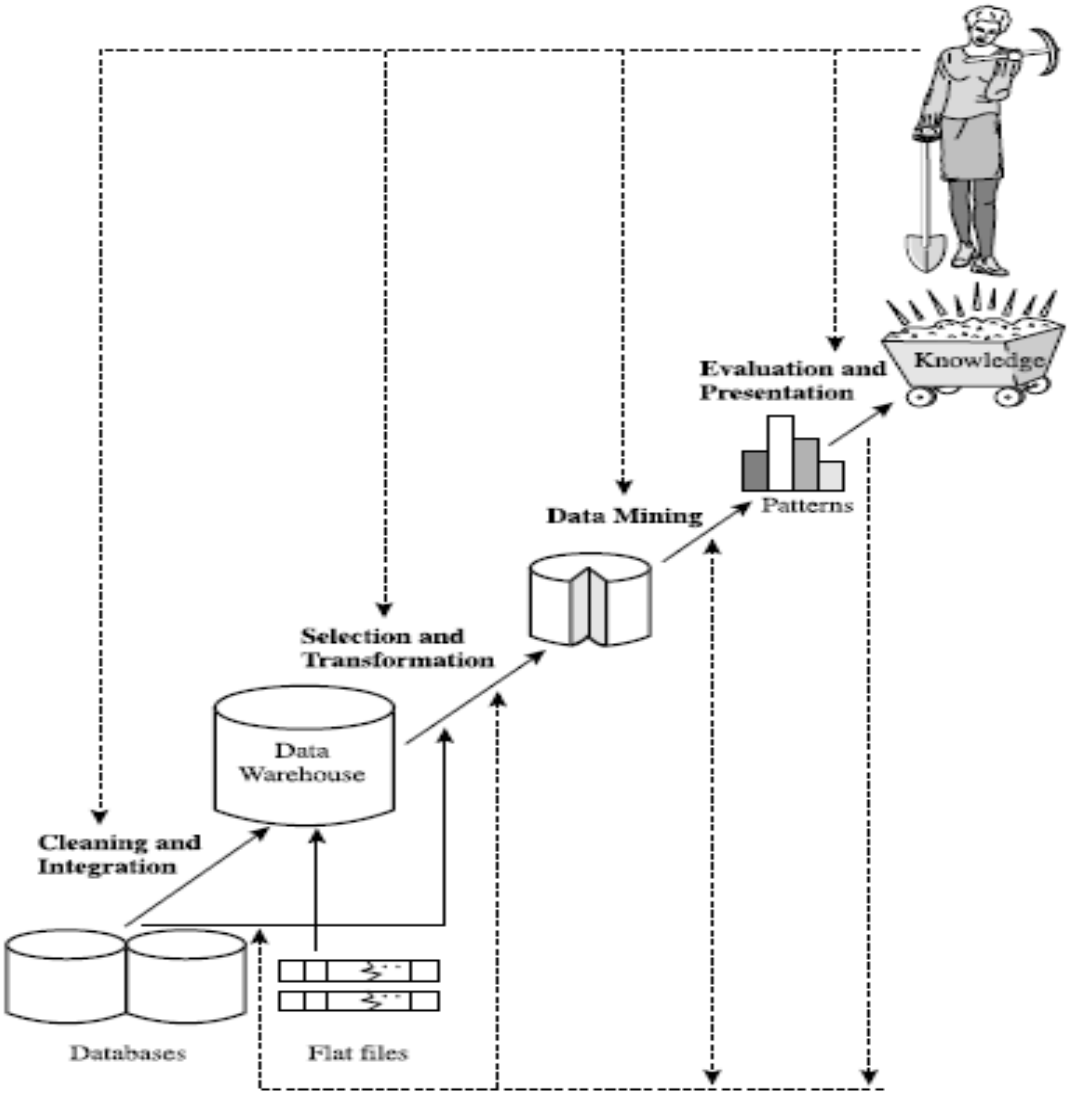
- The major reason that data mining has attracted a great deal of attention in information industry in recent years is due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge.
- The information and knowledge gained can be used for applications ranging from business management, production control, and market analysis, to engineering design and science exploration.



1. b. Explain Data mining as a step in the process of knowledge discovery. [6M]

Knowledge discovery as a process consists of an iterative sequence of the following steps:

1. **Data cleaning** (to remove noise and inconsistent data)
2. **Data integration** (where multiple data sources may be combined)
3. **Data selection** (where data relevant to the analysis task are retrieved from the database)
4. **Data transformation** (where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations)
5. **Data mining** (an essential process where intelligent methods are applied in order to extract data patterns)
6. **Pattern evaluation** (to identify the truly interesting patterns representing knowledge based on some interestingness measures)
7. **Knowledge presentation** (where visualization and knowledge representation techniques are used to present the mined knowledge to the user)



Data mining as a step in the process of knowledge discovery.

- Steps 1 to 4 are different forms of data preprocessing, where the data are prepared for mining. The data mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user and may be stored as new knowledge in the knowledge base.

2. a. Explain about data mining on what kind of data. [6M]

- In principle, data mining should be applicable to any kind of information repository. This includes relational databases, data warehouses, transactional databases, advanced database systems, flat files, and the World-Wide Web.
- Advanced database systems include object-oriented and object-relational databases, and special c application-oriented databases, such as spatial databases, time-series databases, text databases, and multimedia databases.

Flat files:

- Flat files are actually the most common data source for data mining algorithms, especially at the research level.
- Flat files are simple data files in text or binary format with a structure known by the data mining algorithm to be applied.
- The data in these files can be transactions, time-series data, scientific measurements, etc.

Relational Databases:

- A relational database consists of a set of tables containing either values of entity attributes, or values of attributes from entity relationships.
- Tables have columns and rows, where columns represent attributes and rows represent tuples.
- A tuple in a relational table corresponds to either an object or a relationship between objects and is identified by a set of attribute values representing a unique key.
- The most commonly used query language for relational database is SQL, which allows retrieval and manipulation of the data stored in the tables, as well as the calculation of aggregate functions such as average, sum, min, max and count.

customer

cust_ID	*name*	*address*	*age*	*income*	*credit_info*	*category*	...
C1	Smith, Sandy	1223 Lake Ave., Chicago, IL	31	$78000	1	3	...
...	...	...	...	...	...	...	...

item

item_ID	*name*	*brand*	*category*	*type*	*price*	*place_made*	*supplier*	*cost*
I3	hi-res-TV	Toshiba	high resolution	TV	$988.00	Japan	NikoX	$600.00
I8	Laptop	Dell	laptop	computer	$1369.00	USA	Dell	$983.00
...	...	...	...	...	...	...	...	...

employee

empl_ID	*name*	*category*	*group*	*salary*	*commission*
E55	Jones, Jane	home entertainment	manager	$118,000	2%
...	...	...	...	...	...

branch

branch_ID	*name*	*address*
B1	City Square	396 Michigan Ave., Chicago, IL
...	...	...

purchases

trans_ID	*cust_ID*	*empl_ID*	*date*	*time*	*method_paid*	*amount*
T100	C1	E55	03/21/2005	15:45	Visa	$1357.00
...	...	...	...	...	...	...

items_sold

trans_ID	*item_ID*	*qty*
T100	I3	1
T100	I8	2
...	...	...

works_at

empl_ID	*branch_ID*
E55	B1
...	...

- For instance, an SQL query to select the videos grouped by category would be:

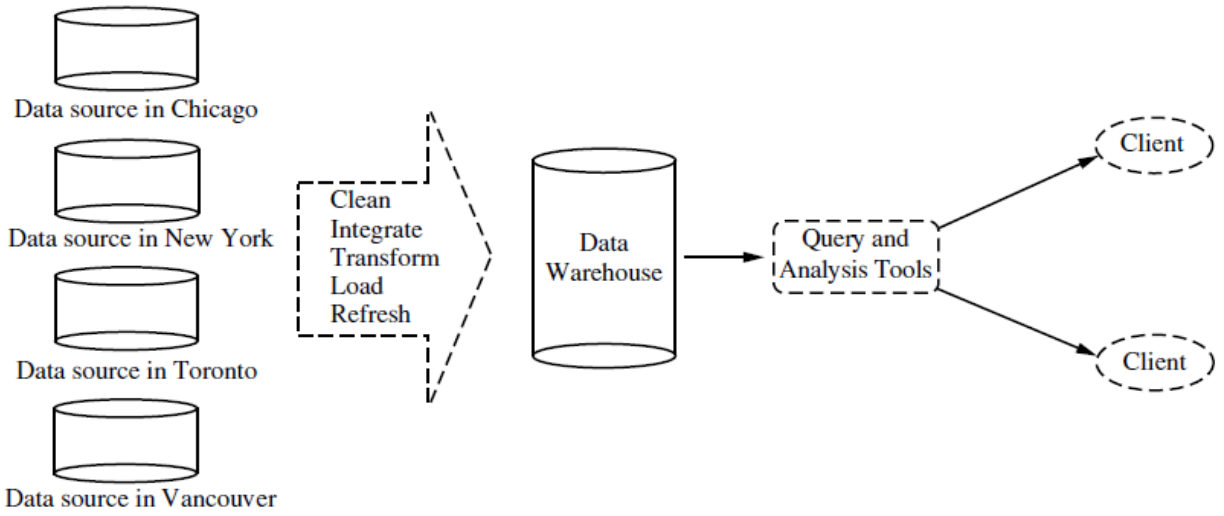
  SELECT count (*) FROM Items WHERE type=video GROUP BY category.

- Data mining algorithms using relational databases can be more versatile than data mining algorithms specifically written for flat files, since they can take advantage of the structure inherent to relational databases.
- While data mining can benefit from SQL for data selection, transformation and consolidation, it goes beyond what SQL could provide, such as predicting, comparing, detecting deviations, etc.

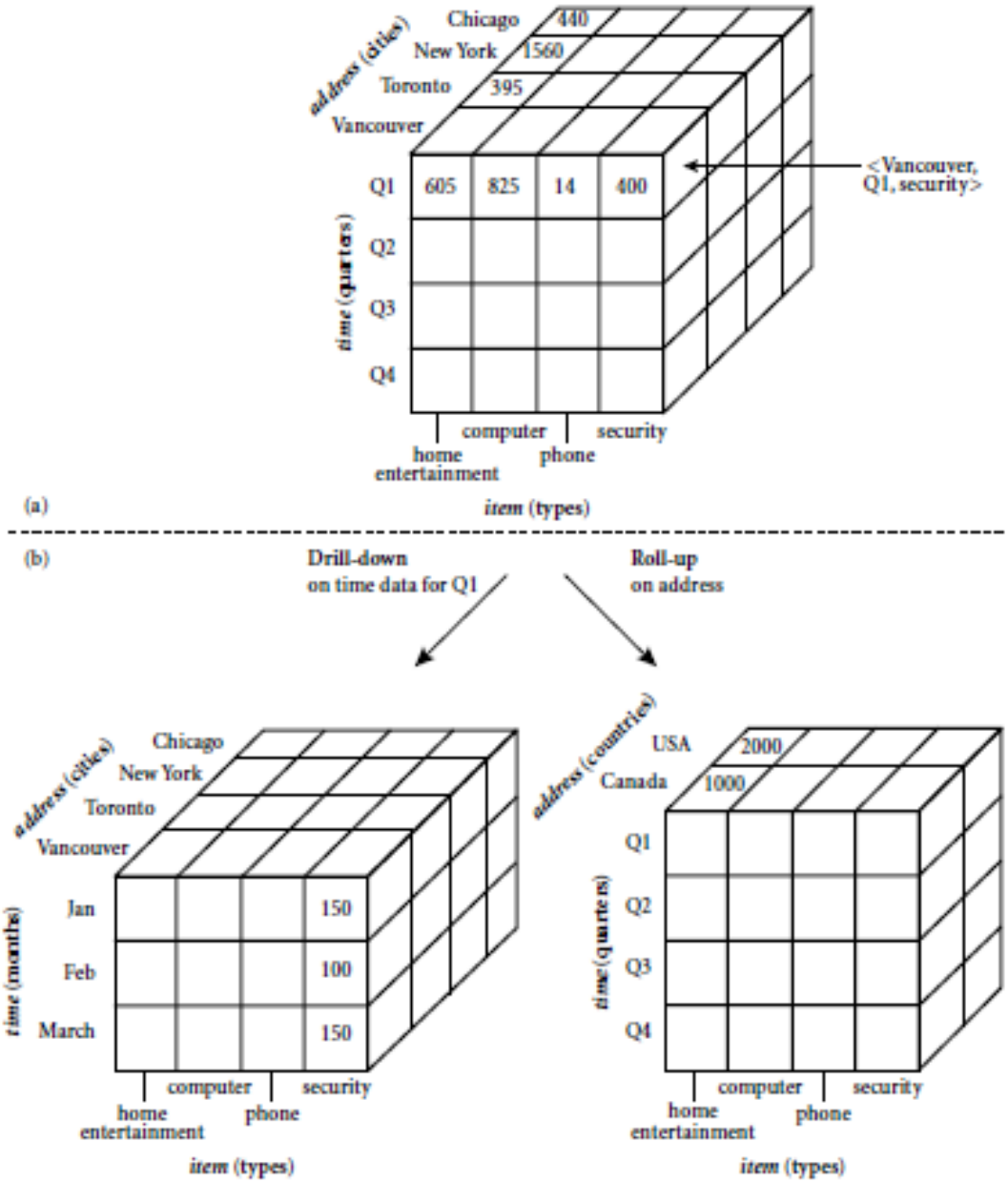
Data warehouses:

- A data warehouse is a repository of information collected from multiple sources, stored under a unified schema, and which usually resides at a single site.

- Data warehouses are constructed via a process of data cleansing, data transformation, data integration, data loading, and periodic data refreshing.



- In order to facilitate decision making, the data in a data warehouse are organized around major subjects, such as customer, item, supplier, and activity.
- The data are stored to provide information from a historical perspective and are typically summarized.
- A data warehouse is usually modeled by a multidimensional database structure, where each dimension corresponds to an attribute or a set of attributes in the schema, and each cell stores the value of some aggregate measure, such as count or sales amount.
- The actual physical structure of a data warehouse may be a relational data store or a multidimensional data cube.
- It provides a multidimensional view of data and allows the pre-computation and fast accessing of summarized data.



Transactional databases:

- In general, a transactional database consists of a flat file where each record represents a transaction.
- A transaction typically includes a unique transaction identity number (trans ID), and a list of the items making up the transaction (such as items purchased in a store) as shown below:

trans_ID	*list of item_IDs*
T100	I1, I3, I8, I16
T200	I2, I8
...	...

2. b. Compare Data Warehousing and Data Mining. **[6M]**

Comparison Parameters	Data Mining	Data Warehousing
Definition	Data mining refers to the process of extracting relevant data from a compiled set of stored data. It is used for the analysis and improvisation strategies chosen by the organization.	Data Warehousing is the process of compiling, sequencing, and organizing groups of data in a commonly accessible database. A data warehouse is used to help management make and implement decisions.
Functionality	Data mining tools use Artificial Intelligence, statistics, databases, and machine learning systems.	Data warehouses are topic-oriented, integrated, time-varying, and non-volatile.
Tasks	Data mining includes the use of pattern recognition logic to identify patterns.	Data warehousing includes data extraction and storage for easier reporting.
Process	Data is regularly analyzed.	Data is periodically stored.
Application	Can be carried out by entrepreneurs and business owners with the assistance of data technicians.	This is a crucial process performed by the organization's data scientists and technical data collection teams.
Advantages	Facilitates the analysis of information and data.	Makes data mining easier and convenient. Helps to sort and upload important data into databases.
Disadvantages	Data mining is not always 100% accurate and can lead to data breaches and hacking if not done correctly.	A high possibility of accumulation of irrelevant and useless data can occur. Data loss and erasure can also be a problem.
Frequency of Update	Data is regularly analyzed in small phases, although it may differ during crisis communication.	Data is loaded periodically, and stacking is a common practice for easy access during extraction.

3. a. Define KDD? **[2M]**

- The term KDD stands for Knowledge Discovery in Databases. It refers to the broad procedure of discovering knowledge in data and emphasizes the high-level applications of specific Data Mining techniques.
- It is a field of interest to researchers in various fields, including artificial intelligence, machine learning, pattern recognition, databases, statistics, knowledge acquisition for expert systems, and data visualization.

3. b. Explain about data mining as a step in the process of Knowledge discovery. **[10M]**

- KDD represents Knowledge Discovery in Databases. It defines the broad process of discovering knowledge in data and emphasizes the high-level applications of definite data mining techniques.
- It is an area of interest to researchers in several fields, such as artificial intelligence, machine learning, pattern recognition, databases, statistics, knowledge acquisition for professional systems, and data visualization.
- The main objective of the KDD process is to extract data from information in the context of huge databases. It does this by utilizing Data Mining algorithms to recognize what is considered knowledge.
- The Knowledge Discovery in Databases is treated as a programmed, exploratory analysis and modeling of huge data repositories. KDD is the organized process of identifying valid, helpful, and understandable designs from large and difficult data sets.
- Data Mining is the root of the KDD procedure, such as the inferring of algorithms that investigate the records, develop the model, and discover previously unknown patterns. The model is used for extracting the knowledge from the information, analyzing the information, and predicting the information.
- Data mining is a step in the KDD process that includes applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, make a specific enumeration of patterns (or models) over the data.
- The KDD process contains using the database along with some required selection, preprocessing, sub sampling, and transformations of it; using data-mining methods (algorithms) to enumerate patterns from it; and computing the products of data mining to recognize the subset of the enumerated patterns deemed knowledge.
- The steps involved in the knowledge discovery process are as follows:

Selection − The data needed for the data mining process is collected from various sources. Therefore, the first step is choosing a dataset or focusing on a subset of variables or data samples on which discovery is to be implemented.

Data cleaning and preprocessing − The data to be used by the process can contain missing or incorrect values so as the basic operations include removing noise, collecting the necessary information to model or account for noise, deciding on techniques for handling missing data fields, and accounting for time-sequence information, is completed in the second step of KDD process.

Data transformation − This step includes finding useful features to represent the data depending on the goal of the task. With dimensionality reduction or transformation approaches, the efficient number of the variable under consideration can be reduced, or invariant representation for the data can be discovered.

Data mining − It is based on the data mining task being performed, this step applies an algorithm to the transformed data, searches for patterns of interest in a particular representational form or a set of specific representations, including classification rules or trees, regression, and clustering.

Interpreting mined patterns − This step can also involve visualization of the extracted patterns and models or visualization of the data given in the extracted models.

4. a. How to classify data mining systems? Discuss in detail. [6M]

- There are many data mining systems available or being developed. Some are specialized systems dedicated to a given data source or are confined to limited data mining functionalities, other are more versatile and comprehensive.
- Data mining systems can be categorized according to various criteria among other classification are the following:

Classification according to the type of data source mined:

- This classification categorizes data mining systems according to the type of data handled such as spatial data, multimedia data, time-series data, text data, World Wide Web, etc.

Classification according to the data model drawn on:

- This classification categorizes data mining systems based on the data model involved such as relational database, object-oriented database, data warehouse, transactional, etc.

Classification according to the king of knowledge discovered:

- This classification categorizes data mining systems based on the kind of knowledge discovered or data mining functionalities, such as characterization, discrimination, association, classification, clustering, etc. Some systems tend to be comprehensive systems offering several data mining functionalities together.

Classification according to mining techniques used:

- Data mining systems employ and provide different techniques. This classification categorizes data mining systems according to the data analysis approach used such as machine learning, neural networks, genetic algorithms, statistics, visualization, database oriented or data warehouse-oriented, etc.
- The classification can also take into account the degree of user interaction involved in the data mining process such as query-driven systems, interactive exploratory systems, or autonomous systems.

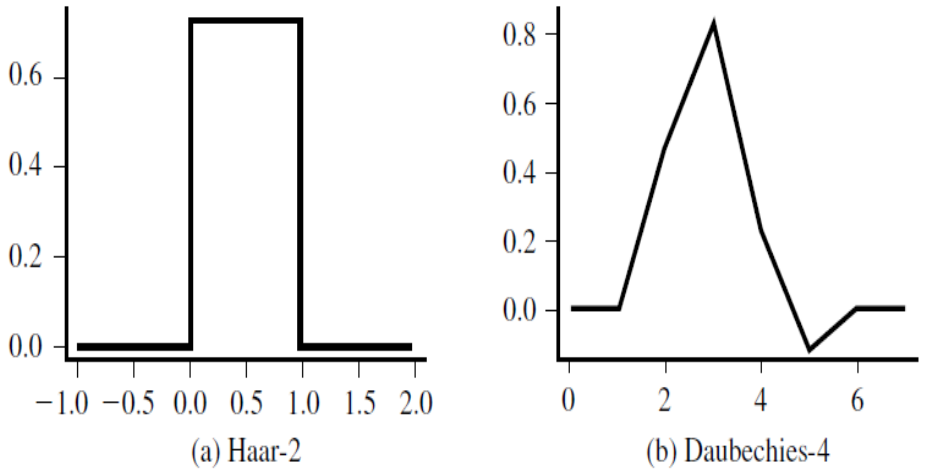
- A comprehensive system would provide a wide variety of data mining techniques to fit different situations and options, and offer different degrees of user interaction.

4. b. Explain about Dimensionality reduction methods? [6M]

- In dimensionality reduction, data encoding or transformations are applied so as to obtain a reduced or "compressed" representation of the original data.
- If the original data can be reconstructed from the compressed data without any loss of information, the data reduction is called **lossless**.
- If, instead, we can reconstruct only an approximation of the original data, then the data reduction is called **lossy**.
- Effective methods of **lossy dimensionality reduction**:
  - **Wavelet transforms and**
  - **Principal components analysis.**

Wavelet transforms:

- The **discrete wavelet transform (DWT)** is a linear signal processing technique that, when applied to a data vector ***X,*** transforms it to a numerically different vector, X', of wavelet coefficients. The two vectors are of the same length.



Examples of wavelet families. The number next to a wavelet name is the number of *vanishing moments* of the wavelet. This is a set of mathematical relationships that the coefficients must satisfy and is related to the number of coefficients.

- The general procedure for applying a discrete wavelet transform uses a hierarchical pyramid algorithm that halves the data at each iteration, resulting in fast computational speed. The method is as follows:
  - Length, L, must be an integer power of 2 (padding with 0's, when necessary)
  - Each transform has 2 functions: **smoothing, difference**
  - Applies to pairs of data, resulting in two set of data of length L/2
  - Applies two functions recursively, until reaches the desired length

Principal Components Analysis:

- Given N data vectors from n-dimensions, find k ≤ n orthogonal vectors (principal components) that can be best used to represent data
    - Normalize input data: Each attribute falls within the same range
    - Compute k orthonormal (unit) vectors, i.e., principal components
    - Each input data (vector) is a linear combination of the k principal component vectors
    - The principal components are sorted in order of decreasing "significance" or strength
    - Since the components are sorted, the size of the data can be reduced by eliminating the weak components, i.e., those with low variance (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data)
- Works for numeric data only.

5. Explain in detail about Data Mining Functionalities with example. [12M]

- Data mining functionalities are used to specify the kind of patterns to be found in data mining tasks. In general, data mining tasks can be classified into two categories: **descriptive and predictive.**
    - **Descriptive mining tasks** characterize the general properties of the data in the database.
    - **Predictive mining tasks** perform inference on the current data in order to make predictions.

Concept/Class Description: Characterization and Discrimination

- Data can be associated with classes or concepts. For example, in the AllElectronics store, **classes of items** for sale include **computers and printers**, and **concepts of customers** include **bigSpenders and budgetSpenders.**
- Such descriptions of a class or a concept are called class/concept descriptions. These descriptions can be derived via

  (1) **data characterization**, by summarizing the data of the class under study (often called the target class) in general terms, or

  (2) **data discrimination**, by comparison of the target class with one or a set of comparative classes (often called the contrasting classes), or

  (3) **both data characterization and discrimination**.
- **Data characterization** is a summarization of the general characteristics or features of a target class of data. The data corresponding to the user-specified class are typically collected by a database query.
- For example, to study the characteristics of software products whose sales increased by 10% in the last year, the data related to such products can be collected by executing an SQL query.

- The output of data characterization can be presented in various forms. Examples include **pie charts, bar charts, curves, multidimensional data cubes, and multidimensional tables, including crosstabs.**
- The resulting descriptions can also be presented as generalized relations or in rule form (called **characteristic rules**).
- **Data discrimination** is a comparison of the general features of target class data objects with the general features of objects from one or a set of contrasting classes.
- The **target and contrasting classes** can be specified by the user, and the corresponding data objects retrieved through database queries.
- For example, the user may like to compare the general features of software products whose sales increased by 10% in the last year with those whose sales decreased by at least 30% during the same period.
- The methods used for data discrimination are similar to those used for data characterization.
- Discrimination descriptions expressed in rule form are referred to as **discriminant rules.**

Mining Frequent Patterns, Associations, and Correlations

- **Frequent patterns**, as the name suggests, are patterns that occur frequently in data. There are many kinds of frequent patterns, including **itemsets, subsequences, and substructures.**
- A **frequent itemset** typically refers to a set of items that frequently appear together in a transactional data set, such as milk and bread.
- A frequently **occurring subsequence**, such as the pattern that customers tend to purchase first a PC, followed by a digital camera, and then a memory card, is a (frequent) sequential pattern.
- A **substructure** can refer to different structural forms, such as graphs, trees, or lattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a (frequent) structured pattern.
- Mining frequent patterns leads to the discovery of interesting associations and correlations within data.
- An example of such a rule, mined from the AllElectronics transactional database, is

$$buys(X, \text{“}computer\text{”}) \Rightarrow buys(X, \text{“}software\text{”}) \ [support = 1\%, confidence = 50\%]$$

  where *X* is a variable representing a customer. A confidence, or certainty, of 50% means that if a customer buys a computer, there is a 50% chance that she will buy software as well.
- A 1% support means that 1% of all of the transactions under analysis showed that computer and software were purchased together.
- This association rule involves a **single attribute or predicate** (i.e., *buys*) that repeats. Association rules that contain a single predicate are referred to as **single-dimensional association rules**. Dropping the predicate notation, the above rule can be written simply as “computer) software [1%, 50%]”.

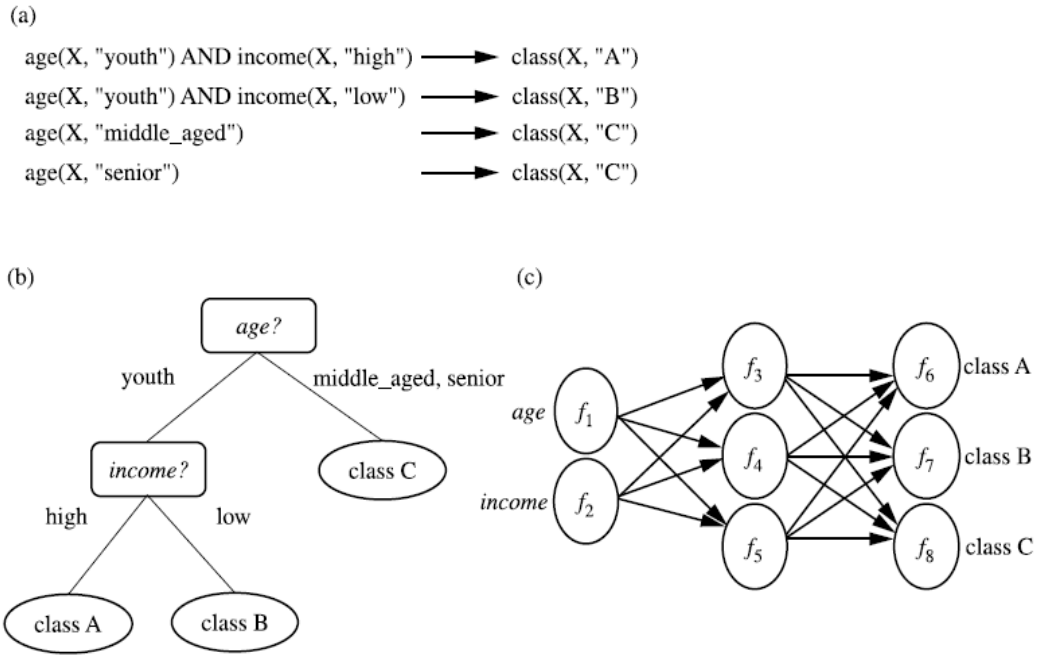
- Suppose, instead, that we are given the AllElectronics relational database relating to purchases. A data mining system may find association rules like

$$age(X, \text{“20...29”}) \wedge income(X, \text{“20K...29K”}) \Rightarrow buys(X, \text{“CD player”})$$
$$[support = 2\%, confidence = 60\%]$$

- An association between **more than one attribute, or predicate** (i.e., age, income, and buys). Adopting the terminology used in **multidimensional databases**, where each attribute is referred to as a dimension, the above rule can be referred to as a multidimensional association rule.
- Typically, association rules are discarded as uninteresting if they do not satisfy both a **minimum support threshold and a minimum confidence threshold**.

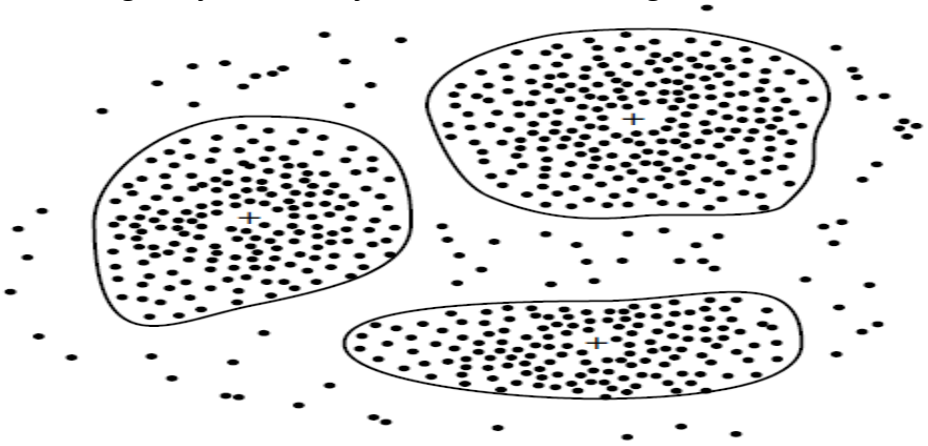
Classification and Prediction:

- Classification is the process of finding a model (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown.
- The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known).
- Decision tree is a flow-chart-like tree structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distributions. Decision trees can easily be converted to classification rules.
- A neural network, when used for classification, is typically a collection of neuron-like processing units with weighted connections between the units.
- There are many other methods for constructing classification models, such as naive Bayesian classification, support vector machines, and k-nearest neighbor classification.



Cluster Analysis:

- Clustering analyzes data objects without consulting a known class label.



- The class labels are not present in the training data simply because they are not known to begin with. Clustering can be used to generate such labels. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the interclass similarity.

Outlier Analysis:

- A database may contain data objects that do not comply with the general behavior or model of the data. These data objects are outliers. Most data mining methods discard outliers as noise or exceptions.
- However, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring ones. The analysis of outlier data is referred to as outlier mining.

Evolution Analysis:

- Data evolution analysis describes and models regularities or trends for objects whose behavior changes over time.
- Although this may include characterization, discrimination, association and correlation analysis, classification, prediction, or clustering of time related data, distinct features of such an analysis include time-series data analysis, sequence or periodicity pattern matching, and similarity-based data analysis.

6. Discuss about Data Mining Task primitives with examples. [12M]

- Each user will have a data mining task in mind, that is, some form of data analysis that he or she would like to have performed.
- A data mining task can be specified in the form of a data mining query, which is input to the data mining system.
- A data mining query is defined in terms of data mining task primitives. These primitives allow the user to interactively communicate with the data mining system during discovery in order to direct the mining process, or examine the findings from different angles or depths.

The set of task-relevant data to be mined:

- This specifies the portions of the database or the set of data in which the user is interested. This includes the database attributes or data warehouse dimensions of interest (referred to as the relevant attributes or dimensions).
- The kind of knowledge to be mined: This specifies the data mining functions to be performed, such as characterization, discrimination, association or correlation analysis, classification, prediction, clustering, outlier analysis, or evolution analysis.

The background knowledge to be used in the discovery process:

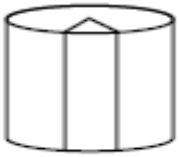
- This knowledge about the domain to be mined is useful for guiding the knowledge discovery process and for evaluating the patterns found. Concept hierarchies are a popular form of background knowledge, which allow data to be mined at multiple levels of abstraction.

The interestingness measures and thresholds for pattern evaluation:

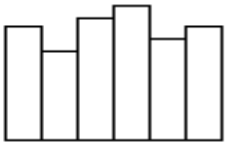
- They may be used to guide the mining process or, after discovery, to evaluate the discovered patterns.
- Different kinds of knowledge may have different interestingness measures. For example, interestingness measures for association rules include support and confidence.
- Rules whose support and confidence values are below user-specified thresholds are considered uninteresting.

The expected representation for visualizing the discovered patterns:

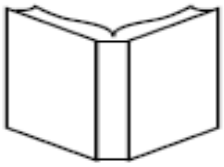
- This refers to the form in which discovered patterns are to be displayed, which may include rules, tables, charts, graphs, decision trees, and cubes.
- A data mining query language can be designed to incorporate these primitives, allowing users to flexibly interact with data mining systems.
- Having a data mining query language provides a foundation on which user-friendly graphical interfaces can be built.



Task-relevant data
Database or data warehouse name
Database tables or data warehouse cubes
Conditions for data selection
Relevant attributes or dimensions
Data grouping criteria



Knowledge type to be mined
Characterization
Discrimination
Association/correlation
Classification/prediction
Clustering



Background knowledge
Concept hierarchies
User beliefs about relationships in the data

Pattern interestingness measures
Simplicity
Certainty (e.g., confidence)
Utility (e.g., support)
Novelty

Visualization of discovered patterns
Rules, tables, reports, charts, graphs, decision trees, and cubes
Drill-down and roll-up

7. a. Discuss the Major issues in Data mining. **[6M]**

Mining methodology and user interaction issues:

- These reflect the kinds of knowledge mined, the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad hoc mining, and knowledge visualization.

Mining different kinds of knowledge in databases:

- Because different users can be interested in different kinds of knowledge, data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including **data characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis** (which includes trend and similarity analysis).

Interactive mining of knowledge at multiple levels of abstraction:

- Because it is difficult to know exactly what can be discovered within a database, the data mining process should be interactive.
- For databases containing a huge amount of data, appropriate sampling techniques can first be applied to facilitate interactive data exploration.
- Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results.

Incorporation of background knowledge:

- Background knowledge, or information regarding the domain under study, may be used to guide the discovery process and allow discovered patterns to be expressed in concise terms and at different levels of abstraction.
- Domain knowledge related to databases, such as integrity constraints and deduction rules, can help focus and speed up a data mining process, or judge the interestingness of discovered patterns.

Data mining query languages and ad hoc data mining:

- Relational query languages (such as SQL) allow users to pose ad hoc queries for data retrieval.
- In a similar vein, high-level data mining query languages need to be developed to allow users to describe ad hoc data mining tasks by facilitating the specification of the relevant sets of data for analysis, the domain knowledge, the kinds of knowledge to be mined, and the conditions and constraints to be enforced on the discovered patterns.

Presentation and visualization of data mining results:

- Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that the knowledge can be easily understood and directly usable by humans.

Handling noisy or incomplete data:

- The data stored in a database may reflect noise, exceptional cases, or incomplete data objects.
- When mining data regularities, these objects may confuse the process, causing the knowledge model constructed to over fit the data. As a result, the accuracy of the discovered patterns can be poor.
- Data cleaning methods and data analysis methods that can handle noise are required, as well as outlier mining methods for the discovery and analysis of exceptional cases.

Pattern evaluation — the interestingness problem:

- A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, either because they represent common knowledge or lack novelty.
- Several challenges remain regarding the development of techniques to assess the interestingness of discovered patterns, particularly with regard to subjective measures that estimate the value of patterns with respect to a given user class, based on user beliefs or expectations.

Performance issues:

- These include **efficiency, scalability, and parallelization** of data mining algorithms.

Efficiency and scalability of data mining algorithms:

- To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable.
- In other words, the running time of a data mining algorithm must be predictable and acceptable in large databases.
- From a database perspective on knowledge discovery, efficiency and scalability are key issues in the implementation of data mining systems.
- Many of the issues discussed above under mining methodology and user interaction must also consider efficiency and scalability.

Parallel, distributed, and incremental mining algorithms:

- The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of parallel and distributed data mining algorithms.
- Such algorithms divide the data into partitions, which are processed in parallel. The results from the partitions are then merged.
- Moreover, the high cost of some data mining processes promotes the need for incremental data mining algorithms that incorporate database updates without having to mine the entire data again "from scratch."

Issues relating to the diversity of database types:

Handling of relational and complex types of data:

- Because relational databases and data warehouses are widely used, the development of efficient and effective data mining systems for such data is important.
- However, other databases may contain complex data objects, hypertext and multimedia data, spatial data, temporal data, or transaction data. It is unrealistic to expect one system to mine all kinds of data, given the diversity of data types and different goals of data mining.

Mining information from heterogeneous databases and global information systems:

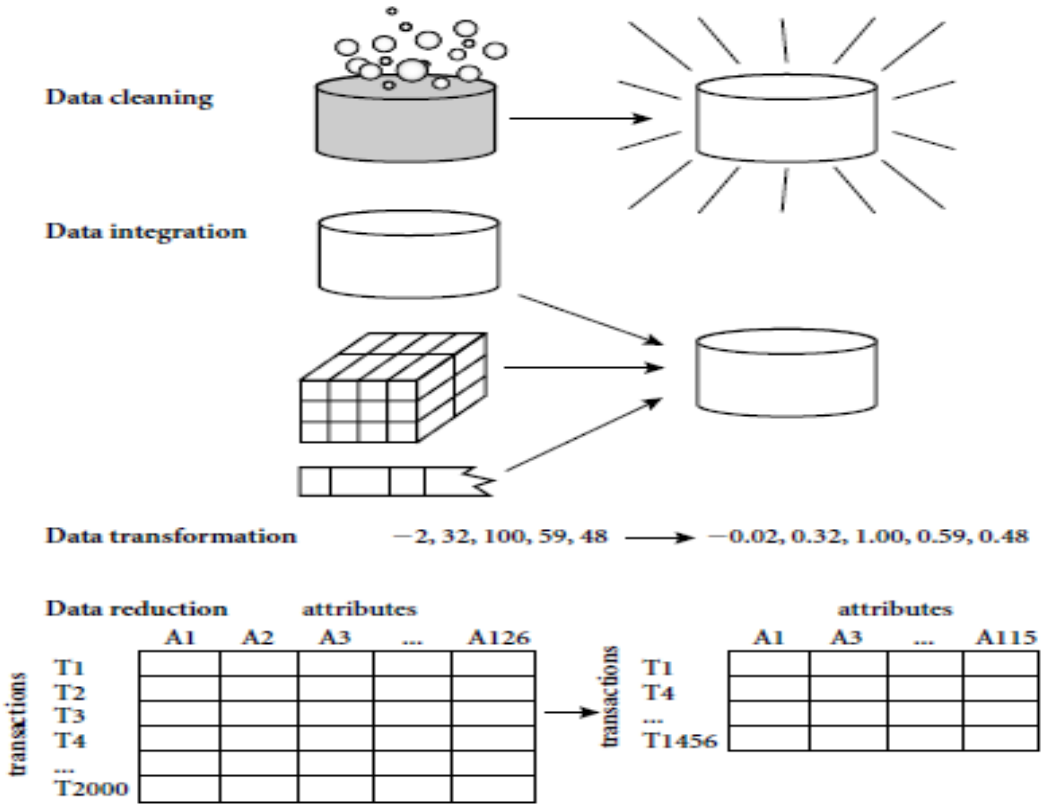
- Local- and wide-area computer networks (such as the Internet) connect many sources of data, forming huge, distributed, and heterogeneous databases. The discovery of knowledge from different sources of structured, semi structured, or unstructured data with diverse data semantics poses great challenges to data mining.

7. b. Why do we pre-process the data? Discuss? [6M]

Why Preprocess the Data?

- Incomplete, noisy, and inconsistent data are commonplace properties of large real world databases and data warehouses.
- Incomplete data can occur for a number of reasons. Attributes of interest may not always be available, such as customer information for sales transaction data.
- Other data may not be included simply because it was not considered important at the time of entry.

- Relevant data may not be recorded due to a misunderstanding, or because of equipment malfunctions.
- Data that were inconsistent with other recorded data may have been deleted.
- Missing data, particularly for tuples with missing values for some attributes, may need to be inferred.
- There are many possible reasons for noisy data (having incorrect attribute values). The data collection instruments used may be faulty.
- There may have been human or computer errors occurring at data entry. Errors in data transmission can also occur.
- Incorrect data may also result from inconsistencies in naming conventions or data codes used, or inconsistent formats for input fields, such as date.
- Duplicate tuples also require data cleaning. Data cleaning routines work to “clean” the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.
- If users believe the data are dirty, they are unlikely to trust the results of any data mining that has been applied to it.
- Furthermore, dirty data can cause confusion for the mining procedure, resulting in unreliable output.
- Although most mining routines have some procedures for dealing with incomplete or noisy data, they are not always robust. Instead, they may concentrate on avoiding over fitting the data to the function being modeled.
- Getting back to your task at AllElectronics, suppose that you would like to include data from multiple sources in your analysis. This would involve integrating multiple databases, data cubes, or files, that is, data integration.
- Yet some attributes representing a given concept may have different names in different databases, causing inconsistencies and redundancies.
- For example, the attribute for customer identification may be referred to as customer id in one data store and cust_id in another.
- Naming inconsistencies may also occur for attribute values. For example, the same first name could be registered as “Bill” in one database, but “William” in another, and “B.” in the third.

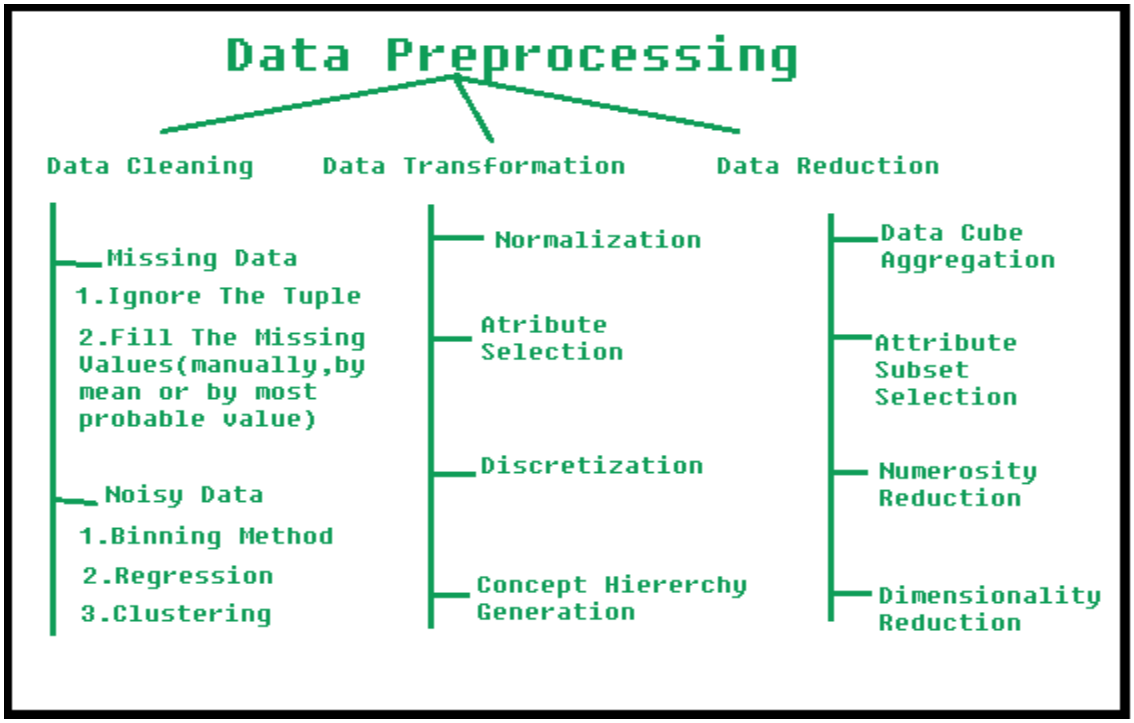


- Having a large amount of redundant data may slow down or confuse the knowledge discovery process.
- Clearly, in addition to data cleaning, steps must be taken to help avoid redundancies during data integration.
- Typically, data cleaning and data integration are performed as a preprocessing step when preparing the data for a data warehouse.
- Additional data cleaning can be performed to detect and remove redundancies that may have resulted from data integration.
- Real-world data tend to be dirty, incomplete, and inconsistent. Data preprocessing techniques can improve the quality of the data, thereby helping to improve the accuracy and efficiency of the subsequent mining process.
- Data preprocessing is an important step in the knowledge discovery process, because quality decisions must be based on quality data.
- Detecting data anomalies, rectifying them early, and reducing the data to be analyzed can lead to huge payoffs for decision making.

8. a. Classify different data pre-processing techniques used to improve the Overall quality of the mined data. [6M]

- Users of your database system have reported errors, unusual values, and inconsistencies in the data recorded for some transactions.
- The data you wish to analyze by data mining techniques are **incomplete** (lacking attribute values or certain attributes of interest, or containing only aggregate data), **noisy**

(containing errors, or outlier values that deviate from the expected), and **inconsistent** (e.g., containing discrepancies in the department codes used to categorize items).



1. Data Cleaning:

- The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

a. Missing Data:

- This situation arises when some data is missing in the data. It can be handled in various ways. Some of them are:
    - **Ignore the tuples:**
      This approach is suitable only when the dataset we have is quite large and multiple values are missing within a tuple.
    - **Fill the Missing values:**
      There are various ways to do this task. You can choose to fill the missing values manually, by attribute mean or the most probable value.

b. Noisy Data:

- Noisy data is a meaningless data that can't be interpreted by machines. It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways:
    - **Binning Method:**
      This method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.
    - **Regression:**
      Here data can be made smooth by fitting it to a regression function. The regression used may be linear (having one independent variable) or multiple (having multiple independent variables).

- **Clustering:**
  This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters.

2. Data Transformation:

- This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:
  - **Normalization:**
    It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)
  - **Attribute Selection:**
    In this strategy, new attributes are constructed from the given set of attributes to help the mining process.
  - **Discretization:**
    This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.
  - **Concept Hierarchy Generation:**
    Here attributes are converted from lower level to higher level in hierarchy. For Example-The attribute "city" can be converted to "country".

3. Data Reduction:

- Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases.
- In order to get rid of this, we uses data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.
- The various steps to data reduction are:
  - **Data Cube Aggregation:**
    Aggregation operation is applied to data for the construction of the data cube.
  - **Attribute Subset Selection:**
    The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute. The attribute having p-value greater than significance level can be discarded.
  - **Numerosity Reduction:**
    This enables to store the model of data instead of whole data, for example: Regression Models.
  - **Dimensionality Reduction:**
    This reduces the size of data by encoding mechanisms. It can be lossy or lossless. If after reconstruction from compressed data, original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction. The two effective methods of dimensionality reduction are: **Wavelet transforms and PCA (Principal Component Analysis).**

8. b. What is data cleaning? Describe in detail the different methods for data cleaning. [6M]

- Real-world data tend to be incomplete, noisy, and inconsistent. Data cleaning (or data cleansing) routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data.

Missing Values:

1. Ignore the tuple:

- This is usually done when the class label is missing (assuming the mining task involves classification).
- This method is not very effective, unless the tuple contains several attributes with missing values. It is especially poor when the percentage of missing values per attribute varies considerably.

2. Fill in the missing value manually:

- In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.

3. Use a global constant to fill in the missing value:

- Replace all missing attribute values by the same constant, such as a label like "Unknown" or - ∞.
- If missing values are replaced by, say, "Unknown," then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of "Unknown."

4. Use the attribute mean to fill in the missing value:

- For example, suppose that the average income of AllElectronics customers is $56,000. Use this value to replace the missing value for income.

5. Use the attribute mean for all samples belonging to the same class as the given tuple:

- For example, if classifying customers according to credit risk, replace the missing value with the average income value for customers in the same credit risk category as that of the given tuple.

6. Use the most probable value to fill in the missing value:

- This may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction.
- For example, using the other customer attributes in your data set, you may construct a decision tree to predict the missing values for income.

Noisy Data:

- Noise is a random error or variance in a measured variable. Let's look at the following data smoothing techniques:

1. Binning:

- Binning methods smooth a sorted data value by consulting its "neighborhood," that is, the values around it.
- The sorted values are distributed into a number of "buckets," or bins. Because binning methods consult the neighborhood of values, they perform local smoothing.
- In this example, the data for price are first sorted and then partitioned into equal-frequency bins of size 3 (i.e., each bin contains three values).
- In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is 9. Therefore, each original value in this bin is replaced by the value 9.
- Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median.

- In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries.
- Each bin value is then replaced by the closest boundary value. In general, the larger the width, the greater the effect of the smoothing.

Sorted data for price (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34
Partition into (equal-frequency) bins:
Bin 1: 4, 8, 15
Bin 2: 21, 21, 24
Bin 3: 25, 28, 34
Smoothing by bin means:
Bin 1: 9, 9, 9
Bin 2: 22, 22, 22
Bin 3: 29, 29, 29
Smoothing by bin boundaries:
Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

2. Regression:

- Data can be smoothed by fitting the data to a function, such as with regression.
- Linear regression involves finding the "best" line to fit two attributes (or variables), so that one attribute can be used to predict the other.
- Multiple linear regressions is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

3. Clustering:

- Outliers may be detected by clustering, where similar values are organized into groups, or "clusters."
- Many methods for data smoothing are also methods for data reduction involving discretization.
- Concept hierarchies are a form of data Discretization that can also be used for data smoothing.
- A concept hierarchy for price, for example, may map real price values into inexpensive, moderately priced, and expensive, thereby reducing the number of data values to be handled by the mining process.

Data Cleaning as a Process:

- The first step in data cleaning as a process is discrepancy detection.
- Discrepancies can be caused by several factors, including
  - poorly designed data entry forms that have many optional fields,
  - Human error in data entry, deliberate errors and
  - data decay
- There are a number of different commercial tools that can aid in the step of discrepancy detection.

- Data scrubbing tools use simple domain knowledge (e.g., knowledge of postal addresses, and spell-checking) to detect errors and make corrections in the data.
- Data auditing tools find discrepancies by analyzing the data to discover rules and relationships, and detecting data that violate such conditions.

- Commercial tools can assist in the data transformation step.
  - Data migration tools allow simple transformations to be specified, such as to replace the string "gender" by "sex".
  - ETL (extraction/transformation/loading) tools allow users to specify transforms through a graphical user interface (GUI).

9. a. Explain about Data Transformation in data mining. [6M]

- Data transformation can involve the following:

Smoothing:

- This works to remove noise from the data.

Aggregation:

- Where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute weekly and annual total scores.

Generalization of the data:

- Where low-level or "primitive" (raw) data are replaced by higher-level concepts through the use of concept hierarchies.
- For example, categorical attributes, like street, can be generalized to higher-level concepts, like city or country.

Normalization:

- Where the attribute data are scaled so as to fall within a small specified range, such as −1.0 to 1.0, or 0.0 to 1.0.

Attribute construction (feature construction):

- This is where new attributes are constructed and added from the given set of attributes to help the mining process.

Normalization:

- In which data are scaled to fall within a small, specified range, useful for classification algorithms involving neural networks, distance measurements such as nearest neighbor classification and clustering.
- There are 3 methods for data normalization. They are:
  1) min-max normalization
  2) z-score normalization
  3) normalization by decimal scaling

Min-max normalization:

- Performs linear transformation on the original data values. It can be defined as,

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

is the value to be normalized, min_A, max_A are minimum and maximum values of an attribute A new_ max_A, new_ min_A are the normalization range.

Z-score normalization / zero-mean normalization:

- In which values of an attribute A are normalized based on the mean and standard deviation of A.
- It can be defined as,

$$v' = \frac{v - mean_A}{stand\_dev_A}$$

- This method is useful when min and max value of attribute A are unknown or when outliers that are dominate min-max normalization.

Normalization by decimal scaling:

- Normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A.
- A value v of A is normalized to v' by computing,

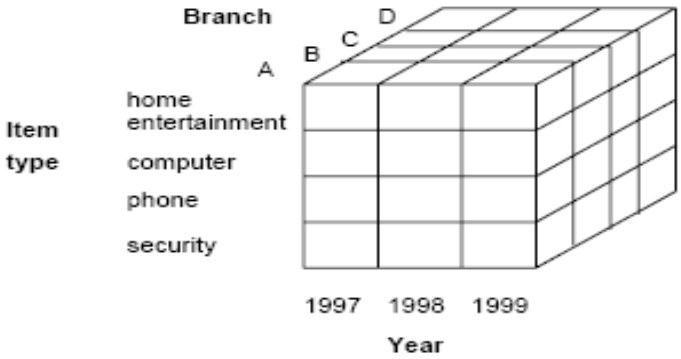
$$v' = \frac{v}{10^j} \quad \text{Where } j \text{ is the smallest integer such that } \text{Max}(|v'|)<1$$

9. b. What is Data Reduction? Discuss in brief. [6M]

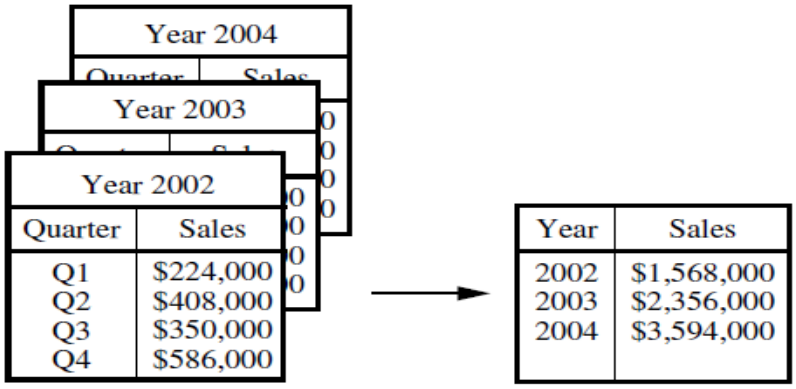
- Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data.
- Strategies for data reduction include the following:
    - **Data cube aggregation**
    - **Attribute subset selection**
    - **Dimensionality reduction**
    - **Numerosity reduction**
    - **Discretization and concept hierarchy generation**

Data Cube Aggregation:

- Reduce the data to the concept level needed in the analysis. Queries regarding aggregated information should be answered using data cube when possible. Data cubes store multidimensional aggregated information.
- The following figure shows a data cube for multidimensional analysis of sales data with respect to annual sales per item type for each branch.



- Each cells holds an aggregate data value, corresponding to the data point in multidimensional space.
- Data cubes provide fast access to pre computed, summarized data, thereby benefiting on-line analytical processing as well as data mining.
- The cube created at the lowest level of abstraction is referred to as the base cuboid. A cube for the highest level of abstraction is the apex cuboid. The lowest level of a data cube (base cuboid).
- Data cubes created for varying levels of abstraction are sometimes referred to as cuboids, so that a "data cube" may instead refer to a lattice of cuboids. Each higher level of abstraction further reduces the resulting data size.
- The following database consists of sales per quarter for the years 1997-1999.
- Suppose, the analyzer interested in the annual sales rather than sales per quarter, the above data can be aggregated so that the resulting data summarizes the total sales per year instead of per quarter.
- The resulting data in smaller in volume, without loss of information necessary for the analysis task.



Attribute Subset Selection:

- Attribute subset selection reduces the data set size by removing irrelevant or redundant attributes (or dimensions).

- Basic heuristic methods of attribute subset selection include the following techniques,

  - **Stepwise forward selection:** The procedure starts with an empty set of attributes as the reduced set. The best of the original attributes is determined and added to the reduced set. At each subsequent iteration or step, the best of the remaining original attributes is added to the set.

  - **Stepwise backward elimination:** The procedure starts with the full set of attributes. At each step, it removes the worst attribute remaining in the set.

  - **Combination of forward selection and backward elimination:** The stepwise forward selection and backward elimination methods can be combined so that, at

each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.

- **Decision tree induction:** Decision tree algorithms, such as **ID3, C4.5, and CART,** were originally intended for classification.
- Decision tree induction constructs a flow chart like structure where each internal (nonleaf) node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each external (leaf) node denotes a class prediction.
- At each node, the algorithm chooses the "best" attribute to partition the data into individual classes.

Forward Selection

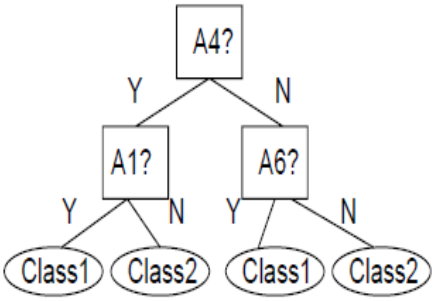
Initial attribute set:
{A1, A2, A3, A4, A5, A6}
Initial reduced set:
{}
-> {A1}
--> {A1, A4}
---> Reduced attribute set:
{A1, A4, A6}

Backward Elimination

Initial attribute set:
{A1, A2, A3, A4, A5, A6}
-> {A1, A3, A4, A5, A6}
--> {A1, A4, A5, A6}
---> Reduced attribute set:
{A1, A4, A6}

Decision Tree Induction

Initial attribute set:
{A1, A2, A3, A4, A5, A6}



---> Reduced attribute set:
{A1, A4, A6}

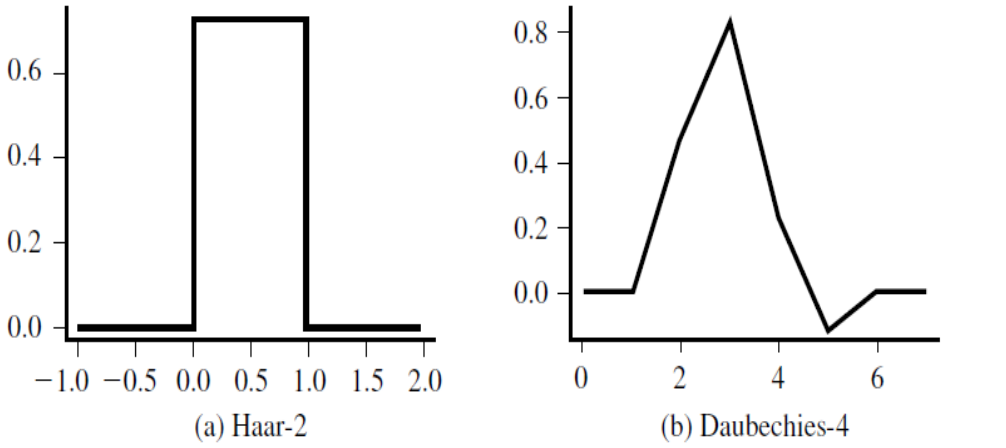
Greedy (heuristic) methods for attribute subset selection.

Dimensionality Reduction:

- In dimensionality reduction, data encoding or transformations are applied so as to obtain a **reduced or "compressed" representation** of the original data.
- If the original data can be reconstructed from the compressed data **without any loss of information**, the data reduction is called **lossless**.
- If, instead, we can reconstruct only an approximation of the original data, then the data reduction is called **lossy**.
- Effective methods of lossy dimensionality reduction:
  - **Wavelet transforms and**
  - **Principal components analysis.**

Wavelet transforms:

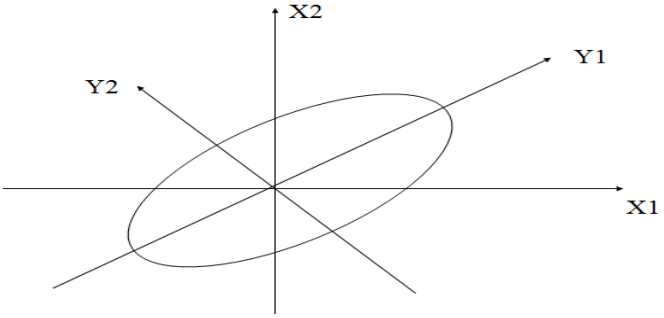
- The **discrete wavelet transform (DWT)** is a linear signal processing technique that, when applied to a data vector ***X,*** transforms it to a numerically different vector, X', of wavelet coefficients. The two vectors are of the same length.
- The general procedure for applying a discrete wavelet transform uses a hierarchical pyramid algorithm that halves the data at each iteration, resulting in fast computational speed. The method is as follows:
  - Length, L, must be an integer power of 2 (padding with 0's, when necessary)
  - Each transform has 2 functions: **smoothing, difference**
  - Applies to pairs of data, resulting in two set of data of length L/2
  - Applies two functions recursively, until reaches the desired length



Examples of wavelet families. The number next to a wavelet name is the number of *vanishing moments* of the wavelet. This is a set of mathematical relationships that the coefficients must satisfy and is related to the number of coefficients.

Principal Components Analysis:

- Given N data vectors from n-dimensions, find k ≤ n orthogonal vectors (principal components) that can be best used to represent data.
  - Normalize input data: Each attribute falls within the same range.
  - Compute k orthonormal (unit) vectors, i.e., principal components.
  - Each input data (vector) is a linear combination of the k principal component vectors.
  - The principal components are sorted in order of decreasing "significance" or strength.
  - Since the components are sorted, the size of the data can be reduced by eliminating the weak components, i.e., those with low variance (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data)
  - Works for numeric data only



Numerosity Reduction:

- Regression and Log-Linear Models
- Histograms
- Clustering
- Sampling

- Data volume can be reduced by choosing alternative smaller forms of data.
  - Parametric method
  - Non parametric method

Parametric:

- Assume the data fits some model, then estimate model parameters, and store only the parameters, instead of actual data.

Non parametric:

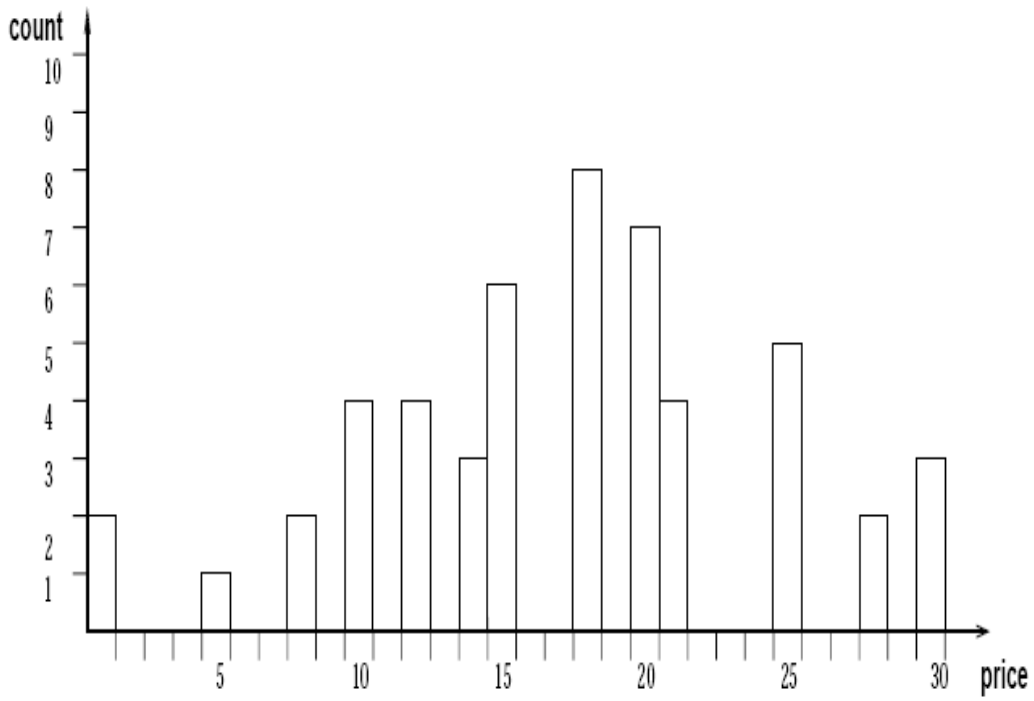
- In which histogram, clustering and sampling is used to store reduced form of data.

1. Regression and log linear models:

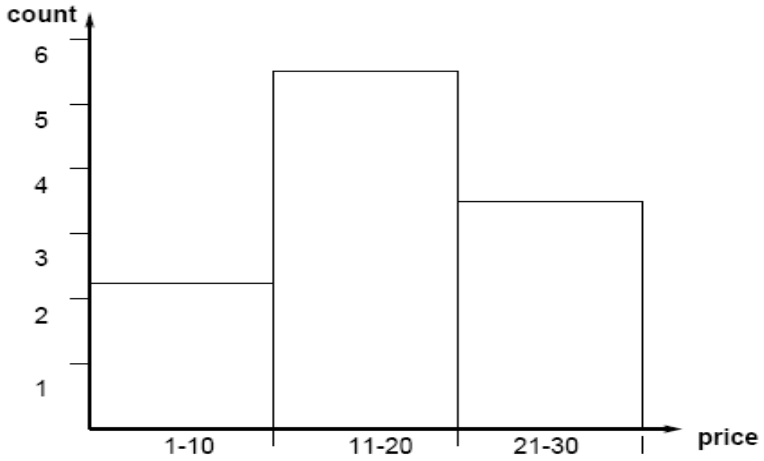
- Can be used to approximate the given data.
- In linear regression, the data are modeled to fit a straight line using **Y = α + β X,** where α, β are coefficients.
- Multiple regression: $Y = b_0 + b_1 X_1 + b_2 X_2$.
- Many nonlinear functions can be transformed into the above.
- Log-linear model: The multi-way table of joint probabilities is approximated by a product of lower-order tables.

2. Histogram:

- Divide data into buckets and store average (sum) for each bucket.
- A bucket represents an attribute-value/frequency pair.
- It can be constructed optimally in one dimension using dynamic programming.
- It divides up the range of possible values in a data set into classes or groups. For each group, a rectangle (bucket) is constructed with a base length equal to the range of values in that specific group, and an area proportional to the number of observations falling into that group.
- The buckets are displayed in a horizontal axis while height of a bucket represents the average frequency of the values.



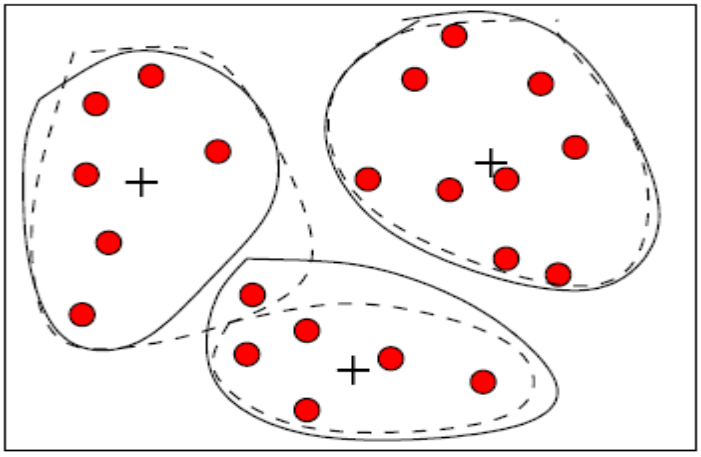
Example:
The following data are a list of prices of commonly sold items. The numbers have been sorted. 1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30. Draw histogram plot for price where each bucket should have equi-width of 10.



- The buckets can be determined based on the following partitioning rules, including the following.
    1. **Equi-width:** histogram with bars having the same width
    2. **Equi-depth:** histogram with bars having the same height
    3. **V-Optimal:** histogram with least variance (countb*valueb)
    4. **MaxDiff:** bucket boundaries defined by user specified threshold

Clustering techniques:

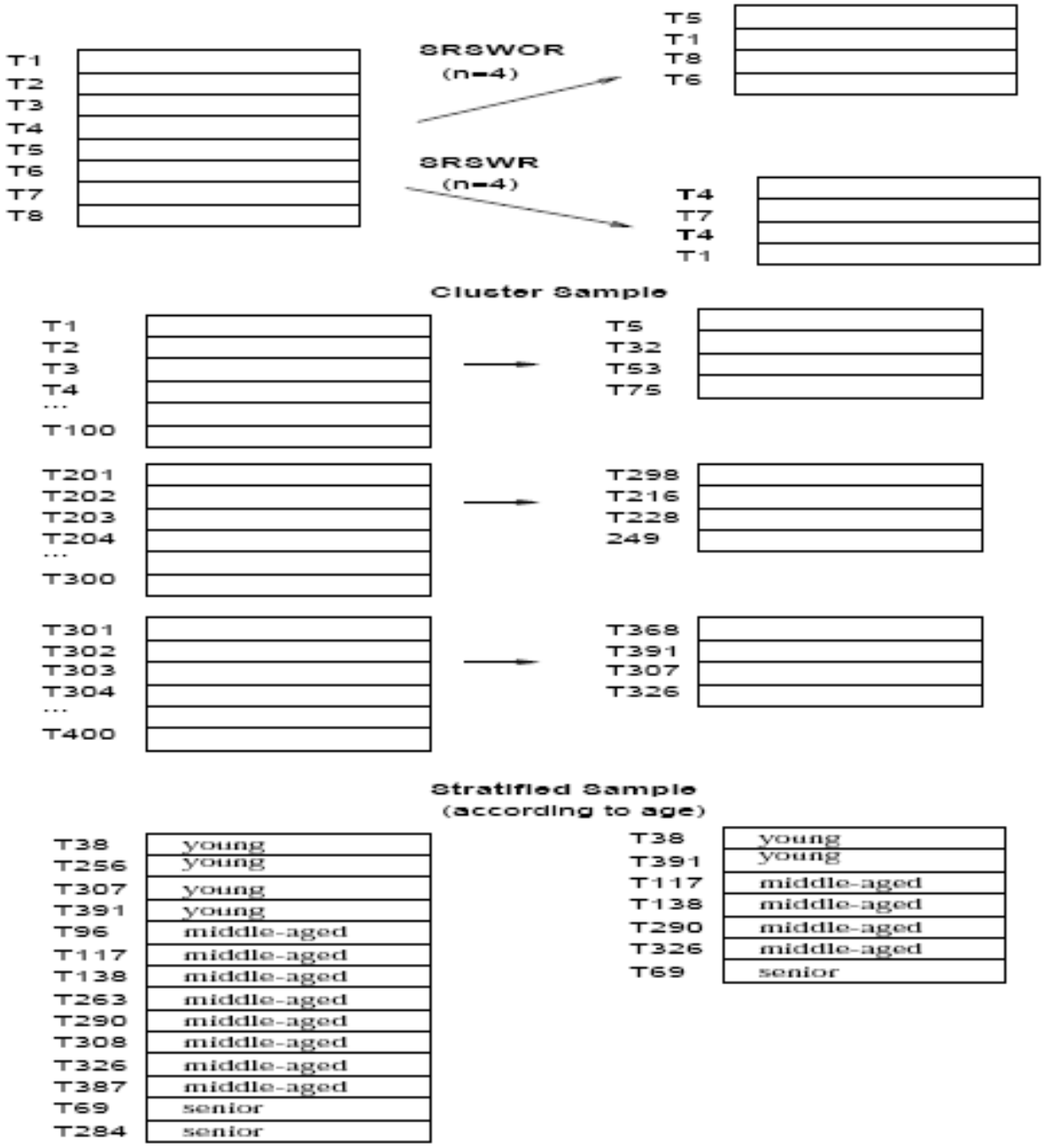
- Consider data tuples as objects. They partition the objects into groups or clusters, so that objects within a cluster are “similar" to one another and “dissimilar" to objects in other clusters. Similarity is commonly defined in terms of how “close" the objects are in space, based on a distance function.



- Quality of clusters measured by their diameter (max distance between any two objects in the cluster) or centroid distance (avg. distance of each cluster object from its centroid)

Sampling:

- Sampling can be used as a data reduction technique since it allows a large data set to be represented by a much smaller random sample (or subset) of the data. Suppose that a large data set, D, contains N tuples. Let's have a look at some possible samples for D.



1. Simple random sample without replacement (SRSWOR) of size n: This is created by drawing n of the N tuples from D (n < N), where the probably of drawing any tuple in D is 1=N, i.e., all tuples are equally likely.

2. Simple random sample with replacement (SRSWR) of size n: This is similar to SRSWOR, except that each time a tuple is drawn from D, it is recorded and then replaced. That is, after a tuple is drawn, it is placed back in D so that it may be drawn again.

3. Cluster sample: If the tuples in D are grouped into M mutually disjoint “clusters", then a SRS of m clusters can be obtained, where m < M.

4. Stratified sample: If D is divided into mutually disjoint parts called “strata", a stratified sample of D is generated by obtaining a SRS at each stratum. This helps to ensure a representative sample, especially when the data are skewed.

10. a. Illustrate the concept of Data discretization. [6M]

- Data discretization techniques can be used to reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals.
- Interval labels can then be used to replace actual data values. Replacing numerous values of a continuous attribute by a small number of interval labels thereby reduces and simplifies the original data. This leads to a concise, easy-to-use, knowledge-level representation of mining results.
- Discretization techniques can be categorized based on how the discretization is performed, such as whether it uses class information or which direction it proceeds
    - Top-down
    - Bottom-up.
- If the discretization process uses class information, then we say it is **supervised discretization**. Otherwise, it is **unsupervised**.
- If the process starts by first finding one or a few points (called split points or cut points) to split the entire attribute range, and then repeats this recursively on the resulting intervals, it is called **top-down discretization or splitting.** This contrasts with **bottom-up Discretization or merging,** which starts by considering all of the continuous values as potential split-points, removes some by merging neighborhood values to form intervals, and then recursively applies this process to the resulting intervals.

Binning:

- Binning refers to a data smoothing technique that helps to group a huge number of continuous values into smaller values. For data discretization and the development of idea hierarchy, this technique can also be used.

Histogram analysis:

- Histogram refers to a plot used to represent the underlying frequency distribution of a continuous data set. Histogram assists the data inspection for data distribution. For example, Outliers, skewness representation, normal distribution representation, etc.

Entropy-Based Discretization:

- Entropy-based discretization is a supervised, top-down splitting technique.
- It explores class distribution information in its calculation and determination of split points.
- Let D consist of data instances defined by a set of attributes and a class-label attribute.

$$Info_A(D) = \frac{|D_1|}{|D|} Entropy(D_1) + \frac{|D_2|}{|D|} Entropy(D_2),$$

$$Entropy(D_1) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

- The class-label attribute provides the class information per instance.
- In this, the interval boundaries or split-points defined may help to improve classification accuracy.

- The entropy and information gain measures are used for decision tree induction.

Interval Merge by $\chi 2$ Analysis:

- It is a bottom-up method.
- Find the best neighboring intervals and merge them to form larger intervals recursively.
- The method is supervised in that it uses class information.
- ChiMerge treats intervals as discrete categories.
- The basic notion is that for accurate discretization, the relative class frequencies should be fairly consistent within an interval.
- Therefore, if two adjacent intervals have a very similar distribution of classes, then the intervals can be merged.
- Otherwise, they should remain separate.

10. b. Determine the concept hierarchy generation for categorical data. [6M]

- Categorical data are discrete data. Categorical attributes have a finite (but possibly large) number of distinct values, with no ordering among the values. Examples include **geographic location, job category, and itemtype**.
- There are several methods for the generation of concept hierarchies for categorical data.

Specification of a partial ordering of attributes explicitly at the schema level by users or experts:

- Concept hierarchies for categorical attributes or dimensions typically involve a group of attributes.
- A user or expert can easily define a concept hierarchy by specifying a partial or total ordering of the attributes at the schema level.
- For example, a relational database or a dimension location of a data warehouse may contain the following group of attributes: **street, city, province or state, and country**.
- A hierarchy can be defined by specifying the total ordering among these attributes at the schema level, such as **street < city < province or state < country**.

Specification of a portion of a hierarchy by explicit data grouping:

- This is essentially the manual definition of a portion of a concept hierarchy. In a large database, it is unrealistic to define an entire concept hierarchy by explicit value enumeration.
- We can easily specify explicit groupings for a small portion of intermediate-level data. For example, after specifying that province and country form a hierarchy at the schema level, a user could define some intermediate levels manually.

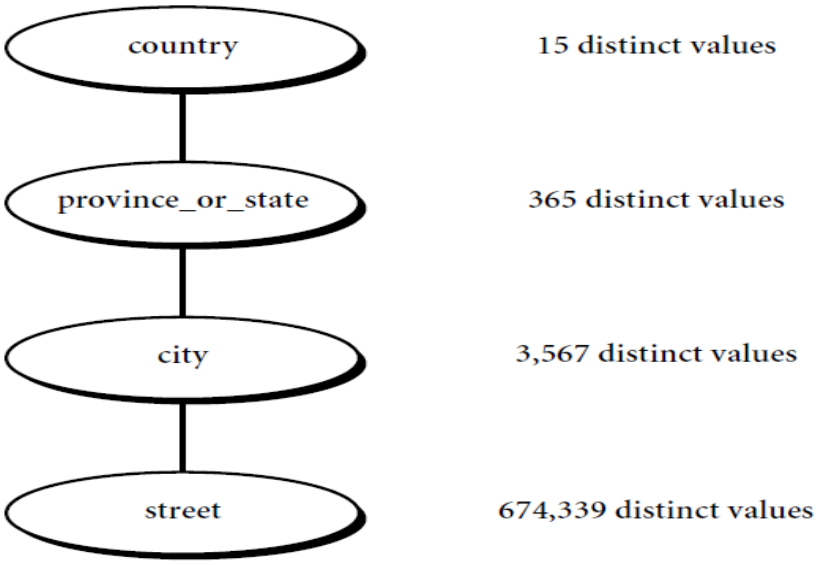
Specification of a set of attributes, but not of their partial ordering:

- A user may specify a set of attributes forming a concept hierarchy, but omit to explicitly state their partial ordering.
- The system can then try to automatically generate the attribute ordering so as to construct a meaningful concept hierarchy.
- Consider the following observation that since higher-level concepts generally cover several subordinate lower-level concepts, an attribute defining a high concept level (e.g., country) will usually contain a smaller number of distinct values than an attribute defining a lower concept level (e.g., street).
- Based on this observation, a concept hierarchy can be automatically generated based on the number of distinct values per attribute in the given attribute set.

- The attribute with the most distinct values is placed at the lowest level of the hierarchy. The lower the number of distinct values an attribute has, the higher it is in the generated concept hierarchy.

Specification of only a partial set of attributes:

- Sometimes a user can be sloppy when defining a hierarchy, or have only a vague idea about what should be included in a hierarchy.
- Consequently, the user may have included only a small subset of the relevant attributes in the hierarchy specification.
- For example, instead of including all of the hierarchically relevant attributes for location, the user may have specified only street and city.
- To handle such partially specified hierarchies, it is important to embed data semantics in the database schema so that attributes with tight semantic connections can be pinned together.
- In this way, the specification of one attribute may trigger a whole group of semantically tightly linked attributes to be "dragged in" to form a complete hierarchy.



UNIT–II

DATA WAREHOUSE AND OLAP TECHNOLOGY

1. a. Discuss in detail about Data Warehouse Implementation. [6M]

- Data warehouse implemented by using these methods.

    1. Efficient computation of data cubes.
    2. Indexing OLAP data.
    3. Efficient processing of OLAP queries.

1. Efficient Data Cube Computation: An Overview

- At the core of multidimensional data analysis is the efficient computation of aggregations across many sets of dimensions.
- In SQL terms, these aggregations are referred to as group-by's.
- Each group-by can be represented by a cuboid, where the set of group-by's forms a lattice of cuboids defining a data cube.

The compute cube Operator and the Curse of Dimensionality

- One approach to cube computation extends SQL so as to include a compute cube operator.
- The compute cube operator computes aggregates over all subsets of the dimensions specified in the operation.

Example:

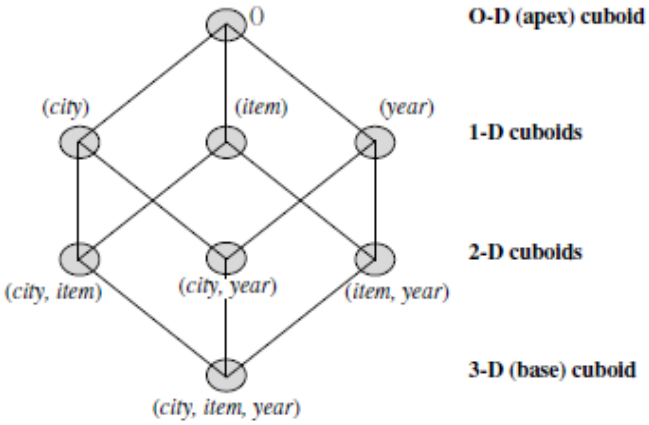
- A data cube is a lattice of cuboids. Suppose that you want to create a data cube for AllElectronics sales that contains the following: city, item, year, and sales in dollars. You want to be able to analyze the data, with queries such as the following:

  "Compute the sum of sales, grouping by city and item."

  "Compute the sum of sales, grouping by city."

  "Compute the sum of sales, grouping by item."

  What is the total number of cuboids, or group-by's that can be computed for this data cube?



- The **base cuboid** contains all three dimensions, ***city, item, and year***. It can return the total sales for any combination of the three dimensions.
- The **apex cuboid, or 0-D** cuboid, refers to the case where the group-by is empty. It contains the total sum of all sales.
- The **base cuboid** is the **least generalized** (most specific) of the cuboids.
- The **apex cuboid** is the **most generalized** (least specific) of the cuboids.
- Similar to the SQL syntax, the data cube could be defined as

  define cube sales cube [city, item, year]: sum (sales in dollars)
- For a cube with n dimensions, there are a total of 2^n cuboids, including the base cuboid.
- A statement such as

  compute cube sales cube

Partial Materialization: Selected Computation of Cuboids

- There are three choices for data cube materialization given a base cuboid:

1. No materialization: Do not precompute any of the "nonbase" cuboids.

2. Full materialization: Precompute all of the cuboids. The resulting lattice of computed cuboids is referred to as the *full cube.* This choice typically requires huge amounts of memory space in order to store all of the precomputed cuboids.

3. Partial materialization: Selectively compute a proper subset of the whole set of possible cuboids. The partial materialization of cuboids or subcubes should consider three factors:

(1) identify the subset of cuboids or subcubes to materialize;

(2) exploit the materialized cuboids or subcubes during query processing; and

(3) efficiently update the materialized cuboids or subcubes during load and refresh.

2. Indexing OLAP Data: Bitmap Index and Join Index

- In the bitmap index for a given attribute, there is a distinct bit vector, **Bv,** for each value v in the attribute's domain. If a given attribute's domain consists of n values, then n bits are needed for each entry in the bitmap index.
- If the attribute has the value v for a given row in the data table, then the bit representing that value is set to 1 in the corresponding row of the bitmap index. All other bits for that row are set to 0.

Example: Bitmap indexing.

- In the AllElectronics data warehouse, suppose the dimension item at the top level has four values: **"home entertainment", "computer", "phone", and "security".** Each value (e.g., "computer") is represented by a bit vector in the item bitmap index table.

Base table

RID	*item*	*city*
R1	H	V
R2	C	V
R3	P	V
R4	S	V
R5	H	T
R6	C	T
R7	P	T
R8	S	T

item bitmap index table

RID	H	C	P	S
R1	1	0	0	0
R2	0	1	0	0
R3	0	0	1	0
R4	0	0	0	1
R5	1	0	0	0
R6	0	1	0	0
R7	0	0	1	0
R8	0	0	0	1

city bitmap index table

RID	V	T
R1	1	0
R2	1	0
R3	1	0
R4	1	0
R5	0	1
R6	0	1
R7	0	1
R8	0	1

Note: H for "home entertainment," C for "computer," P for "phone," S for "security," V for "Vancouver," T for "Toronto."

- **Join indexing** registers the joinable rows of two relations from a relational database.
- For example, if two relations R(RID,A) and S(B, SID) join on the attributes A and B, then the join index record contains the pair (RID, SID), where RID and SID are record identifiers from the R and S relations.
- Hence, the join index records can identify joinable tuples without performing costly join operations

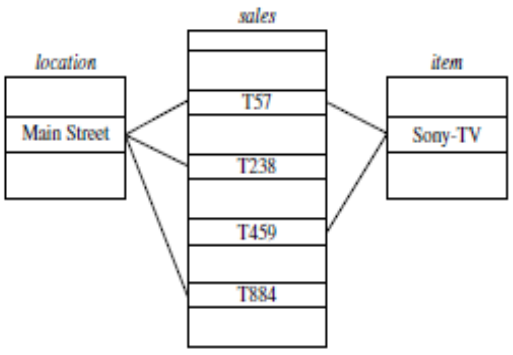


Figure 4.16 Linkages between a *sales* fact table and *location* and *item* dimension tables.

Join index table for *location/sales*

location	*sales_key*
...	...
Main Street	T57
Main Street	T238
Main Street	T884
...	...

Join index table for *item/sales*

item	*sales_key*
...	...
Sony-TV	T57
Sony-TV	T459
...	...

Join index table linking *location* and *item* to *sales*

location	*item*	*sales_key*
...	...	...
Main Street	Sony-TV	T57
...	...	...

Figure 4.17 Join index tables based on the linkages between the *sales* fact table and the *location* and *item* dimension tables shown in Figure 4.16.

3. Efficient Processing of OLAP Queries:

- **Determine which operations should be performed on the available cuboids:** This involves transforming any selection, projection, roll-up (group-by), and drill-down operations specified in the query into corresponding SQL and/or OLAP operations. For example, slicing and dicing a data cube may correspond to selection and/or projection operations on a materialized cuboid.
- **Determine to which materialized cuboid(s) the relevant operations should be applied:** This involves identifying all of the materialized cuboids that may potentially be used to answer the query, pruning the above set using knowledge of "dominance" relationships among the cuboids, estimating the costs of using the remaining materialized cuboids, and selecting the cuboid with the least cost.

1. b. Explain in detail about the key feature of Data Warehousing. [6M]

- **Subject-oriented:** A data warehouse is organized around major subjects, such as customer, supplier, product, and sales. Rather than concentrating on the day-to-day operations and transaction processing of an organization, a data warehouse focuses on the modeling and analysis of data for decision makers. Hence, data warehouses typically provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.
- **Integrated:** A data warehouse is usually constructed by integrating multiple heterogeneous sources, such as relational databases, flat files, and on-line transaction

records. Data cleaning and data integration techniques are applied to ensure consistency in naming conventions, encoding structures, attribute measures, and so on.

- **Time-variant:** Data are stored to provide information from a historical perspective (e.g., the past 5–10 years). Every key structure in the data warehouse contains, either implicitly or explicitly, an element of time.
- **Nonvolatile:** A data warehouse is always a physically separate store of data transformed from the application data found in the operational environment. Due to this separation, a data warehouse does not require transaction processing, recovery, and concurrency control mechanisms. It usually requires only two operations in data accessing: initial loading of data and access of data.

2. a. Discuss in details about different types of Data ware Housing. [6M]

- There are three main types of data warehouse.

Enterprise Data Warehouse (EDW)

- This type of warehouse serves as a key or central database that facilitates decision-support services throughout the enterprise.
- The advantage to this type of warehouse is that it provides access to cross-organizational information, offers a unified approach to data representation, and allows running complex queries.

Operational Data Store (ODS)

- This type of data warehouse refreshes in real-time. It is often preferred for routine activities like storing employee records.
- It is required when data warehouse systems do not support reporting needs of the business.

Data Mart

- A data mart is a subset of a data warehouse built to maintain a particular department, region, or business unit.
- Every department of a business has a central repository or data mart to store data. The data from the data mart is stored in the ODS periodically.
- The ODS then sends the data to the EDW, where it is stored and used.

2. b. Distinguish between OLTP and OLAP. [6M]

Feature	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)

Function	day-to-day operations	long-term informational requirements decision support
DB design	ER-based, application-oriented	star/snowflake, subject-oriented
Data	current, guaranteed up-to-date	historic, accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
Number of records accessed	tens	millions
Number of users	thousands	hundreds
DB size	GB to high-order GB	>=TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

3. a. Discuss in brief about Multi-dimensional data model. [6M]

- Data warehouses and OLAP tools are based on a multidimensional data model. This model views data in the form of a data cube.

From Tables and Spreadsheets to Data Cubes

- A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.
- Dimensions are the perspectives or entities with respect to which an organization wants to keep records. For example, AllElectronics may create a sales data warehouse in order to keep records of the store's sales with respect to the dimensions time, item, branch, and location.
- These dimensions allow the store to keep track of things like monthly sales of items and the branches and locations at which the items were sold. Each dimension may have a table associated with it, called a dimension table.

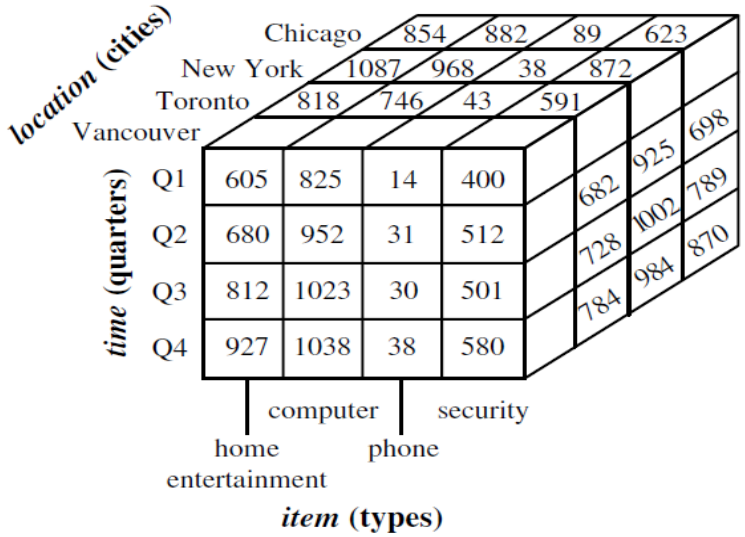
- For example, a dimension table for item may contain the attributes item name, brand, and type.
- Dimension tables can be specified by users or experts, or automatically generated and adjusted based on data distributions.
- A multidimensional data model is typically organized around a central theme, like sales, for instance. This theme is represented by a fact table. Facts are numerical measures.
- Examples of facts for a sales data warehouse include dollars sold (sales amount in dollars), units sold (number of units sold), and amount budgeted.
- The fact table contains the names of the facts, or measures, as well as keys to each of the related dimension tables.

A 2-D view of sales data for *AllElectronics* according to the dimensions *time* and *item*, where the sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).

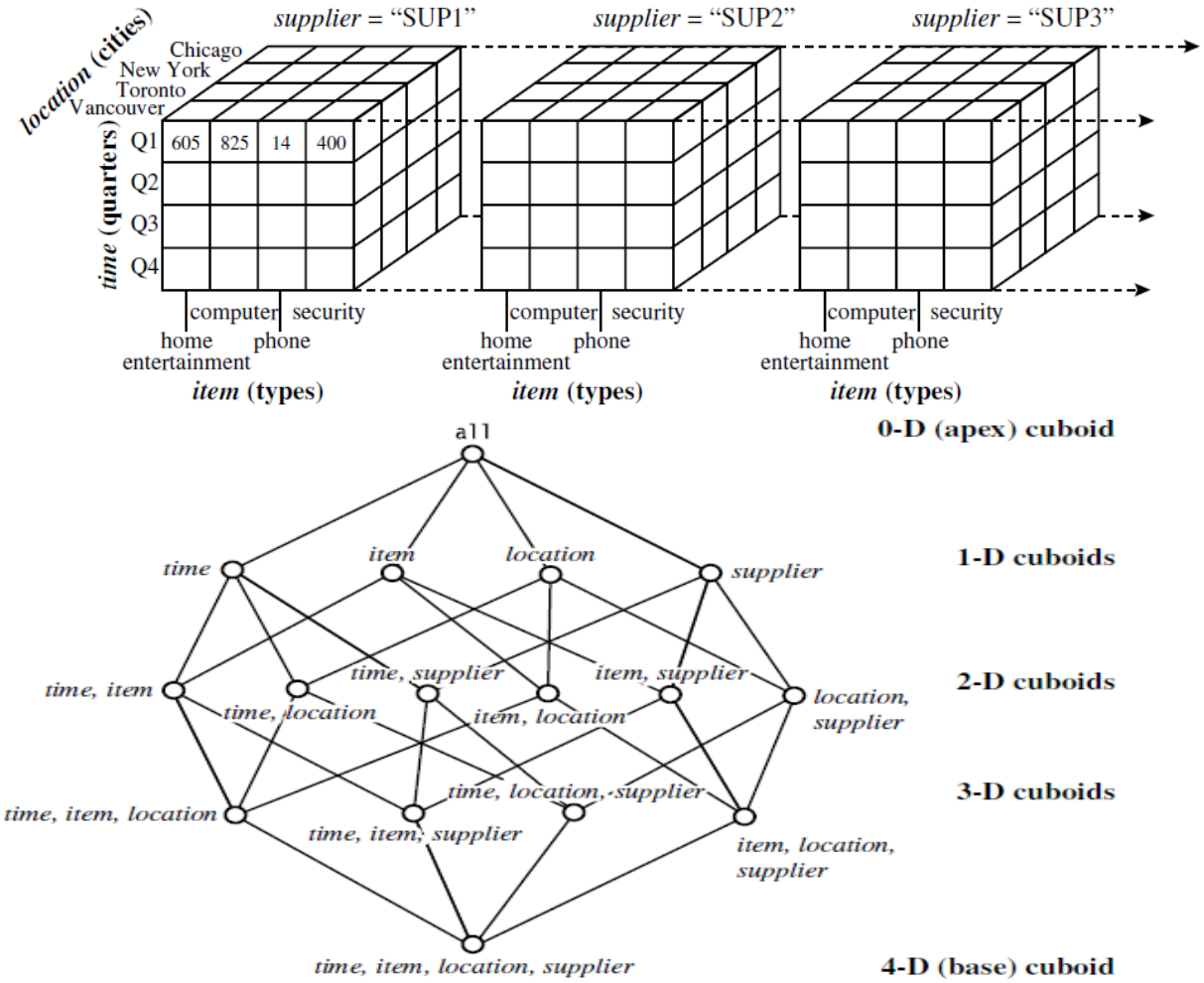
	location = "Vancouver"			
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

A 3-D view of sales data for *AllElectronics*, according to the dimensions *time*, *item*, and *location*. The measure displayed is *dollars_sold* (in thousands).

	location = "Chicago"				**location** = "New York"				**location** = "Toronto"				**location** = "Vancouver"			
	item				**item**				**item**				**item**			
time	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580



- Suppose that we would now like to view our sales data with an additional fourth dimension, such as **supplier.**



Lattice of cuboids, making up a 4-D data cube for the dimensions *time*, *item*, *location*, and *supplier*. Each cuboid represents a different degree of summarization.

3. b. Why Have a Separate Data Warehouse? [6M]

- A major reason for such a separation is to help promote the high performance of both systems.
- An operational database is designed and tuned from known tasks and workloads, such as indexing and hashing using primary keys, searching for particular records, and optimizing "canned" queries.
- On the other hand, data warehouse queries are often complex. They involve the computation of large groups of data at summarized levels, and may require the use of special data organization, access, and implementation methods based on multidimensional views.
- Processing OLAP queries in operational databases would substantially degrade the performance of operational tasks.
- An operational database supports the concurrent processing of multiple transactions. Concurrency control and recovery mechanisms, such as locking and logging, are required to ensure the consistency and robustness of transactions.
- An OLAP query often needs read-only access of data records for summarization and aggregation. Concurrency control and recovery mechanisms, if applied for such OLAP operations, may jeopardize the execution of concurrent transactions and thus substantially reduce the throughput of an OLTP system.
- Finally, the separation of operational databases from data warehouses is based on the different structures, contents, and uses of the data in these two systems. Decision support requires historical data, whereas operational databases do not typically maintain historical data.
- Decision support requires consolidation (such as aggregation and summarization) of data from heterogeneous sources, resulting in high-quality, clean, and integrated data.
- In contrast, operational databases contain only detailed raw data, such as transactions, which need to be consolidated before analysis.

4. Discuss the following data warehouse Model: [12M]

i) Enterprise Warehouse

ii) Data Mart

iii) Virtual Warehouse

i) Enterprise Warehouse

- An enterprise warehouse collects all of the information about subjects spanning the entire organization.
- It provides corporate-wide data integration, usually from one or more operational systems or external information providers, and is cross-functional in scope.
- It typically contains detailed data as well as summarized data, and can range in size from a few gigabytes to hundreds of gigabytes, terabytes, or beyond.
- An enterprise data warehouse may be implemented on traditional mainframes, computer super servers, or parallel architecture platforms.
- It requires extensive business modeling and may take years to design and build.

ii) Data Mart

- A data mart contains a subset of corporate-wide data that is of value to a specific group of users. The scope is confined to specific selected subjects. For example, a marketing data mart may confine its subjects to customer, item, and sales.
- The data contained in data marts tend to be summarized. Data marts are usually implemented on low-cost departmental servers that are UNIX/LINUX- or Windows-based.
- Depending on the source of data, data marts can be categorized as **independent or dependent.**
- **Independent data marts** are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area.
- **Dependent data marts** are sourced directly from enterprise data warehouses.

iii) Virtual Warehouse

- A virtual warehouse is a set of views over operational databases. For efficient query processing, only some of the possible summary views may be materialized.
- A virtual warehouse is easy to build but requires excess capacity on operational database servers.

5. a. What is data Cube? How will data cube allows data to be modeled and viewed in multiple dimensions. [6M]

- Data warehouses and OLAP tools are based on a multidimensional data model. This model views data in the form of a data cube.

From Tables and Spreadsheets to Data Cubes

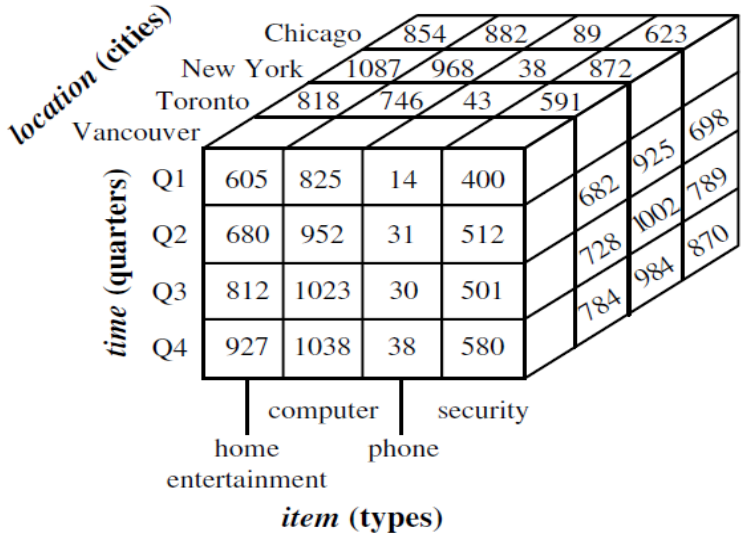
- A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.
- Dimensions are the perspectives or entities with respect to which an organization wants to keep records. For example, AllElectronics may create a sales data warehouse in order to keep records of the store's sales with respect to the dimensions time, item, branch, and location.
- These dimensions allow the store to keep track of things like monthly sales of items and the branches and locations at which the items were sold. Each dimension may have a table associated with it, called a dimension table.
- For example, a dimension table for item may contain the attributes item name, brand, and type.
- Dimension tables can be specified by users or experts, or automatically generated and adjusted based on data distributions.
- A multidimensional data model is typically organized around a central theme, like sales, for instance. This theme is represented by a fact table. Facts are numerical measures.
- Examples of facts for a sales data warehouse include dollars sold (sales amount in dollars), units sold (number of units sold), and amount budgeted.
- The fact table contains the names of the facts, or measures, as well as keys to each of the related dimension tables.

A 2-D view of sales data for *AllElectronics* according to the dimensions *time* and *item*, where the sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).

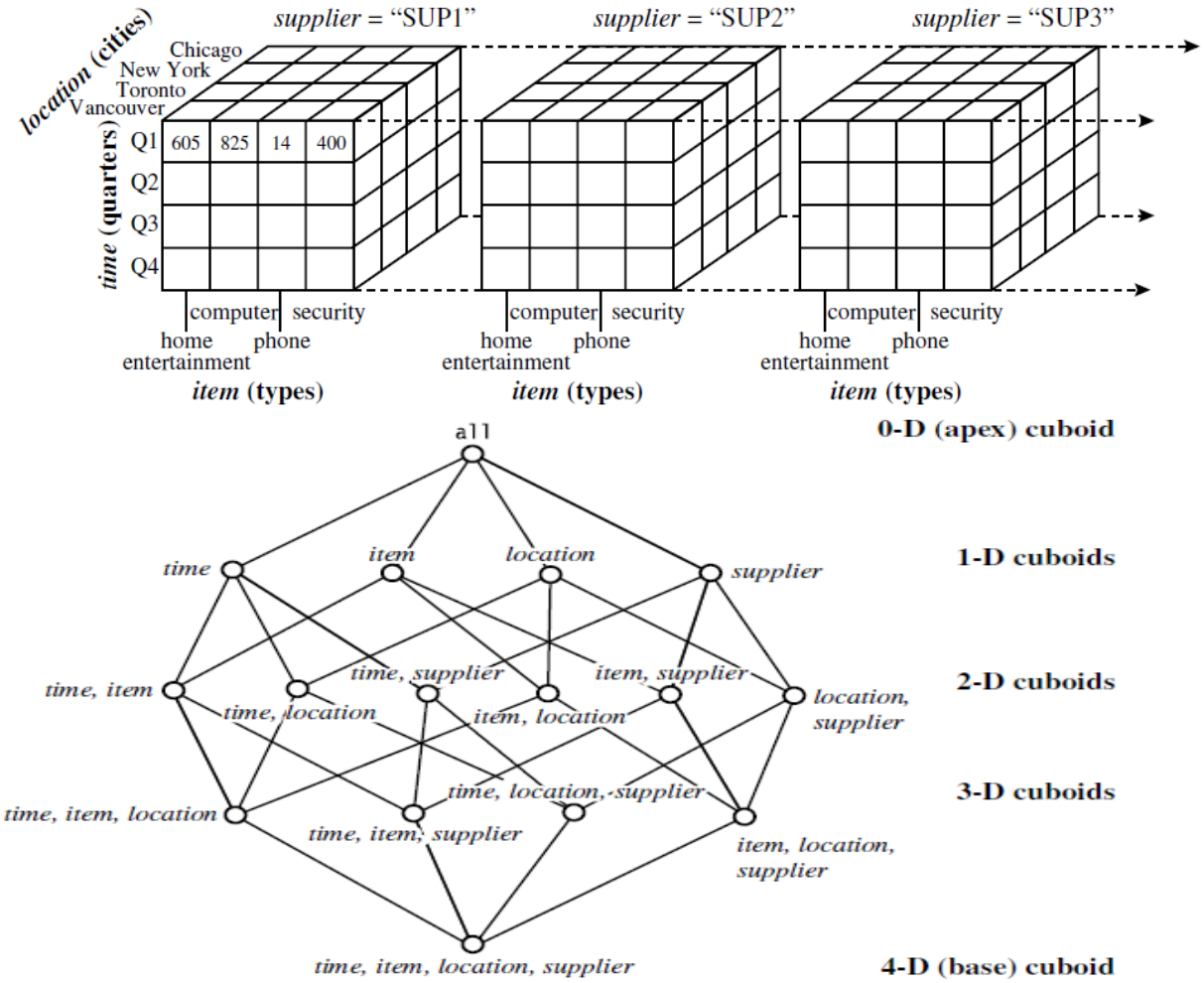
location = "Vancouver"				
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

A 3-D view of sales data for *AllElectronics*, according to the dimensions *time*, *item*, and *location*. The measure displayed is *dollars_sold* (in thousands).

	location = "Chicago"				**location** = "New York"				**location** = "Toronto"				**location** = "Vancouver"			
	item				**item**				**item**				**item**			
time	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580



- Suppose that we would now like to view our sales data with an additional fourth dimension, such as **supplier.**

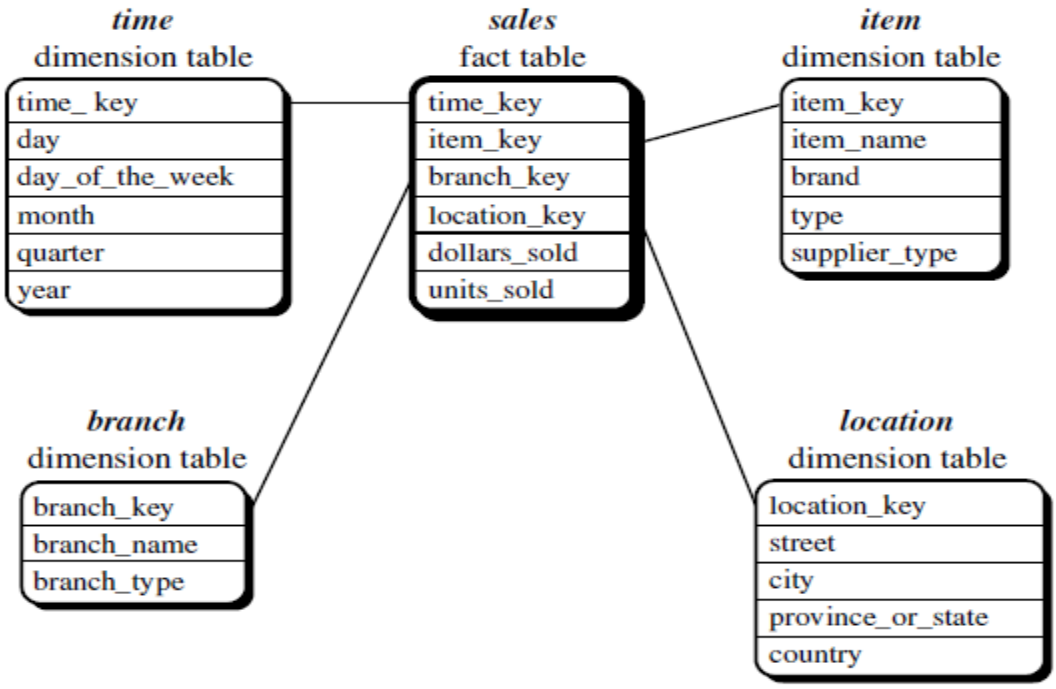


Lattice of cuboids, making up a 4-D data cube for the dimensions *time*, *item*, *location*, and *supplier*. Each cuboid represents a different degree of summarization.

5. b. Discuss in brief about schemas in multidimensional data model. [6M]

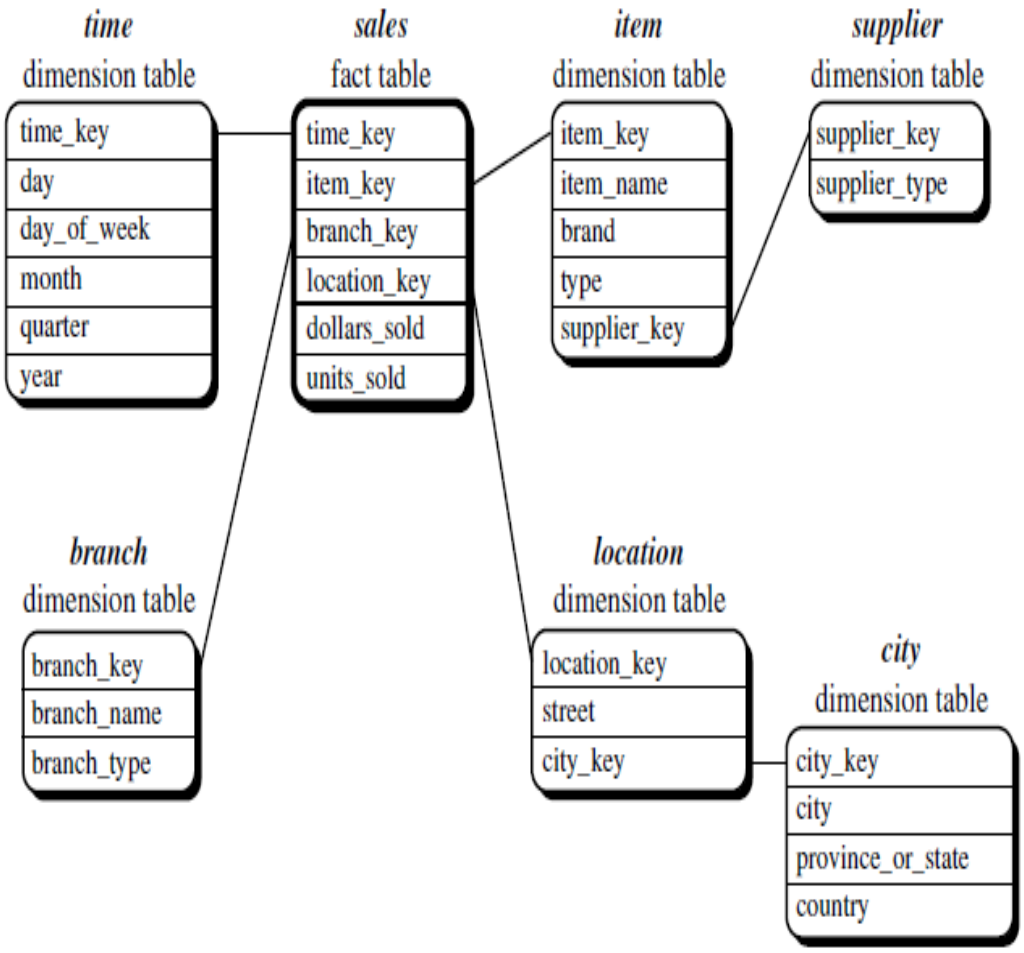
Star schema:

- The most common modeling paradigm is the star schema, in which the data warehouse contains
  (1) a large central table (fact table) containing the bulk of the data, with no redundancy, and
  (2) a set of smaller attendant tables (dimension tables), one for each dimension. The schema graph resembles a starburst, with the dimension tables displayed in a radial pattern around the central fact table.



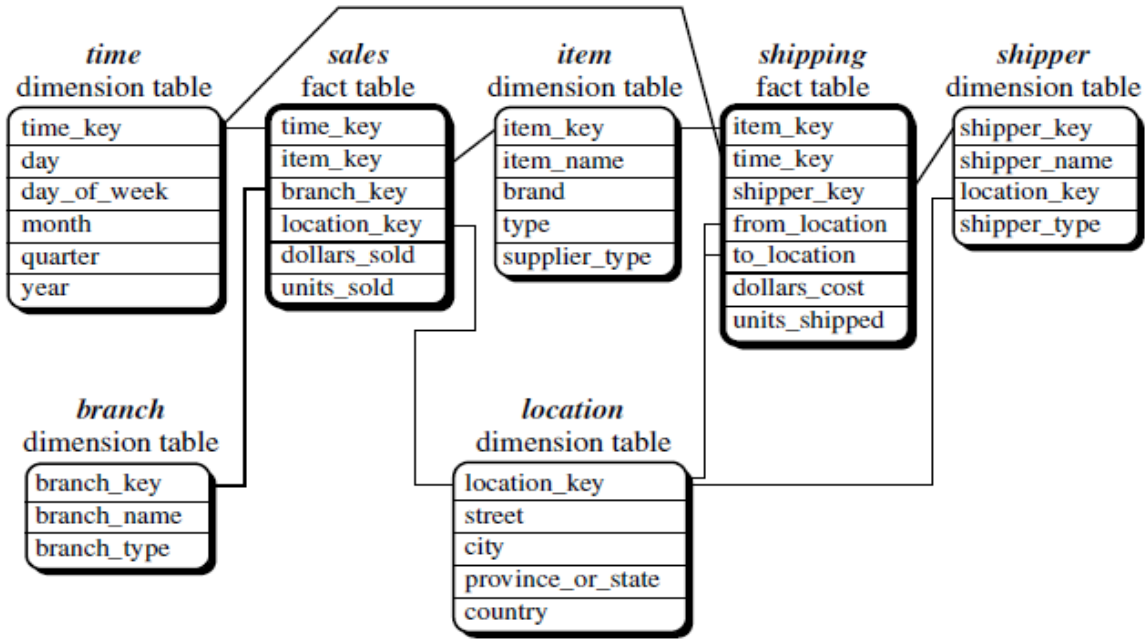
Snowflake schema:

- The snowflake schema is a variant of the star schema model, where some dimension tables are *normalized*, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.
- The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalized form to reduce redundancies. Such a table is easy to maintain and saves storage space.



Fact constellation:

- Sophisticated applications may require multiple fact tables to *share* dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.



Examples for Defining Star, Snowflake, and Fact Constellation Schemas

- Data warehouses and data marts can be defined using two language primitives, one for *cube definition* and one for *dimension definition*.
- The **cube definition** statement has the following syntax:

  define cube <cube name> [<dimension list>]: <measure list>
- The **dimension definition** statement has the following syntax:

  define dimension <dimension name> as (<attribute or dimension list>)

Star schema definition.

define cube sales star [time, item, branch, location]: dollars sold = sum(sales in dollars), units sold = count(*)
define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, country)

Snowflake schema definition.

define cube sales snowflake [time, item, branch, location]: dollars sold = sum(sales in dollars), units sold = count(*)
define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier (supplier key, supplier type))
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city (city key, city, province or state, country))

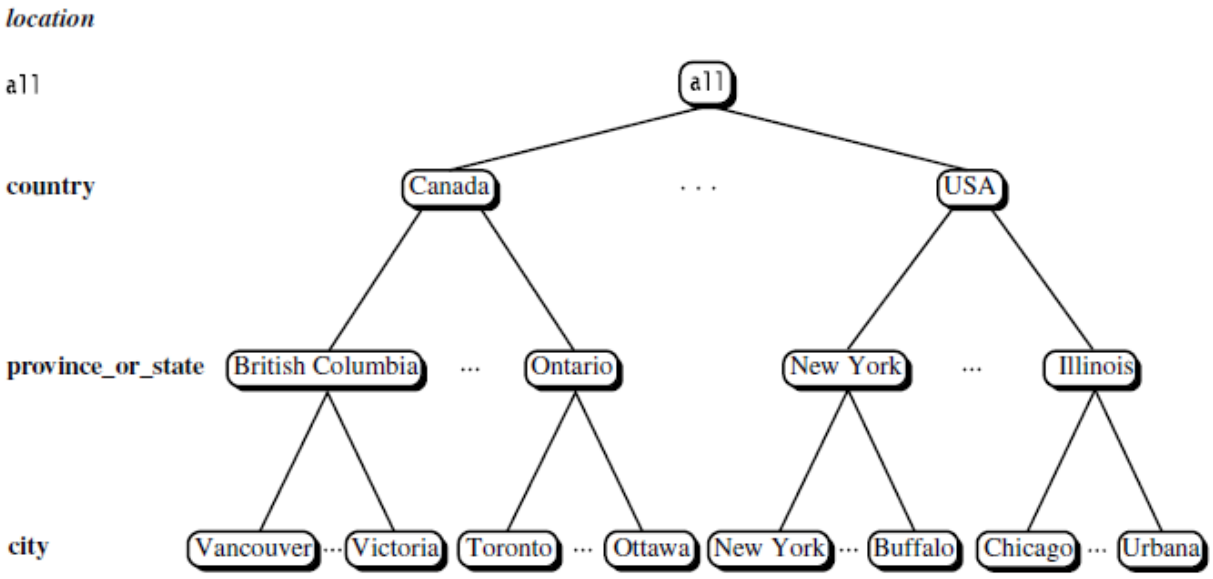
Fact constellation schema definition.

define cube sales [time, item, branch, location]: dollars sold = sum(sales in dollars), units sold = count(*)
define dimension time as (time key, day, day of week, month, quarter, year)

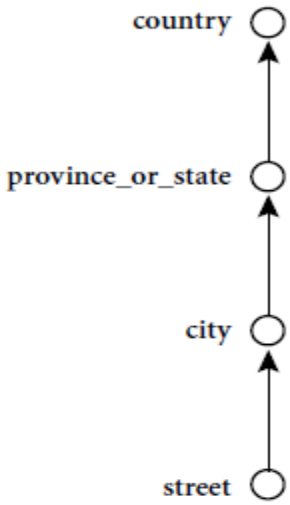
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, country)
define cube shipping [time, item, shipper, from location, to location]: dollars cost = sum(cost in dollars), units shipped = count(*)
define dimension time as time in cube sales
define dimension item as item in cube sales
define dimension shipper as (shipper key, shipper name, location as location in cube sales, shipper type)
define dimension from location as location in cube sales
define dimension to location as location in cube sales

6. a. Explain the Role of Concept Hierarchies in dimension. [6M]

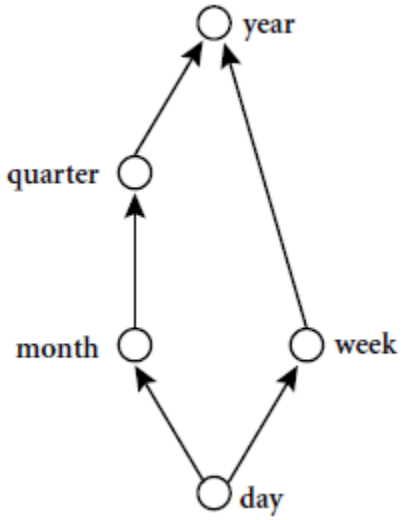
- A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts.
- Consider a concept hierarchy for the dimension location. City values for location include Vancouver, Toronto, NewYork, and Chicago. Each city, however, can be mapped to the province or state to which it belongs.
- For example, Vancouver can be mapped to British Columbia, and Chicago to Illinois. The provinces and states can in turn be mapped to the country to which they belong, such as Canada or the USA.
- These mappings forma concept hierarchy for the dimension location, mapping a set of low-level concepts (i.e., cities) to higher-level, more general concepts (i.e., countries).



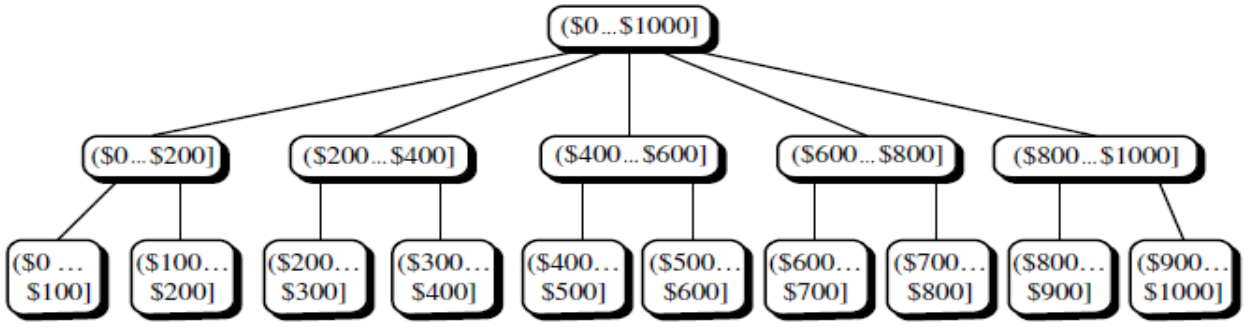
- Many concept hierarchies are implicit within the database schema. For example, suppose that the dimension location is described by the attributes number, street, city, province or state, zipcode, and country.
- These attributes are related by a total order, forming a concept hierarchy such as **"street < city < province or state < country".**



- Alternatively, the attributes of a dimension may be organized in a partial order, forming a lattice. An example of a partial order for the *time* dimension based on the attributes day, week, month, quarter, and year is “day < {month <quarter; week} < year”.



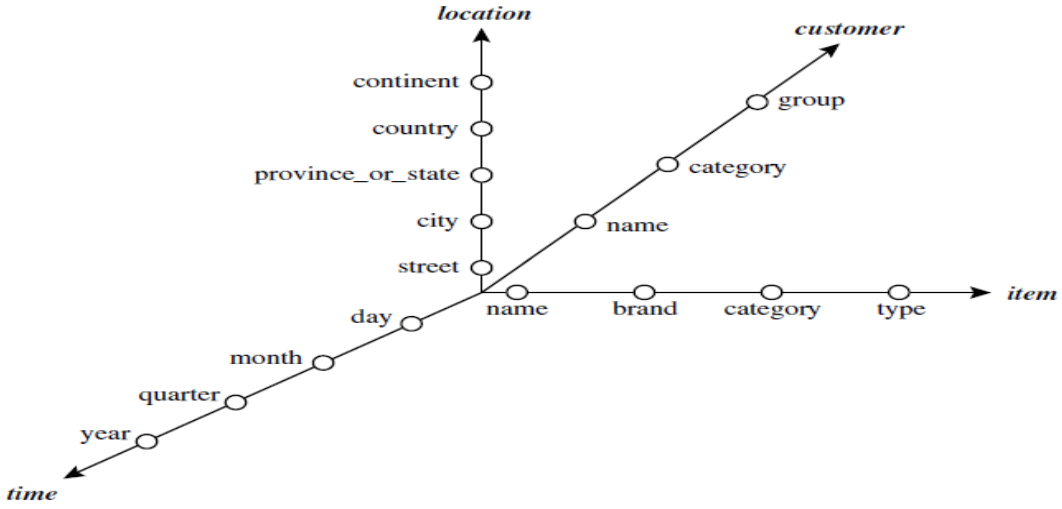
- A concept hierarchy that is a total or partial order among attributes in a database schema is called a schema hierarchy. Concept hierarchies that are common to many applications may be predefined in the data mining system, such as the concept hierarchy for time.
- Concept hierarchies may also be defined by discretizing or grouping values for a given dimension or attribute, resulting in a set-grouping hierarchy. A total or partial order can be defined among groups of values.
- There may be more than one concept hierarchy for a given attribute or dimension, based on different user viewpoints. For instance, a user may prefer to organize pric*e* by defining ranges for inexpensive, moderately priced, and expensive.



- Concept hierarchies may be provided manually by system users, domain experts, or knowledge engineers, or may be automatically generated based on statistical analysis of the data distribution.

6. b. Classify a Starnet Query Model how will involve MultidimensionalDatabases. [6M]

- The querying of multidimensional databases can be based on a starnet model. A starnet model consists of radial lines emanating from a central point, where each line represents a concept hierarchy for a dimension.
- Each abstraction level in the hierarchy is called a footprint. These represent the granularities available for use by OLAP operations such as drill-down and roll-up for the dimensions location, customer, item, and time, respectively.
- Each line consists of footprints representing abstraction levels of the dimension. For example, the time line has four footprints: “day,” “month,” “quarter,” and “year.”
- A concept hierarchy may involve a single attribute (like date for the time hierarchy) or several attributes (e.g., the concept hierarchy for location involves the attributes street, city, province or state, and country).
- Concept hierarchies can be used to generalize data by replacing low-level values (such as “day” for the time dimension) by higher-level abstractions (such as “year”), or to specialize data by replacing higher-level abstractions with lower-level values.

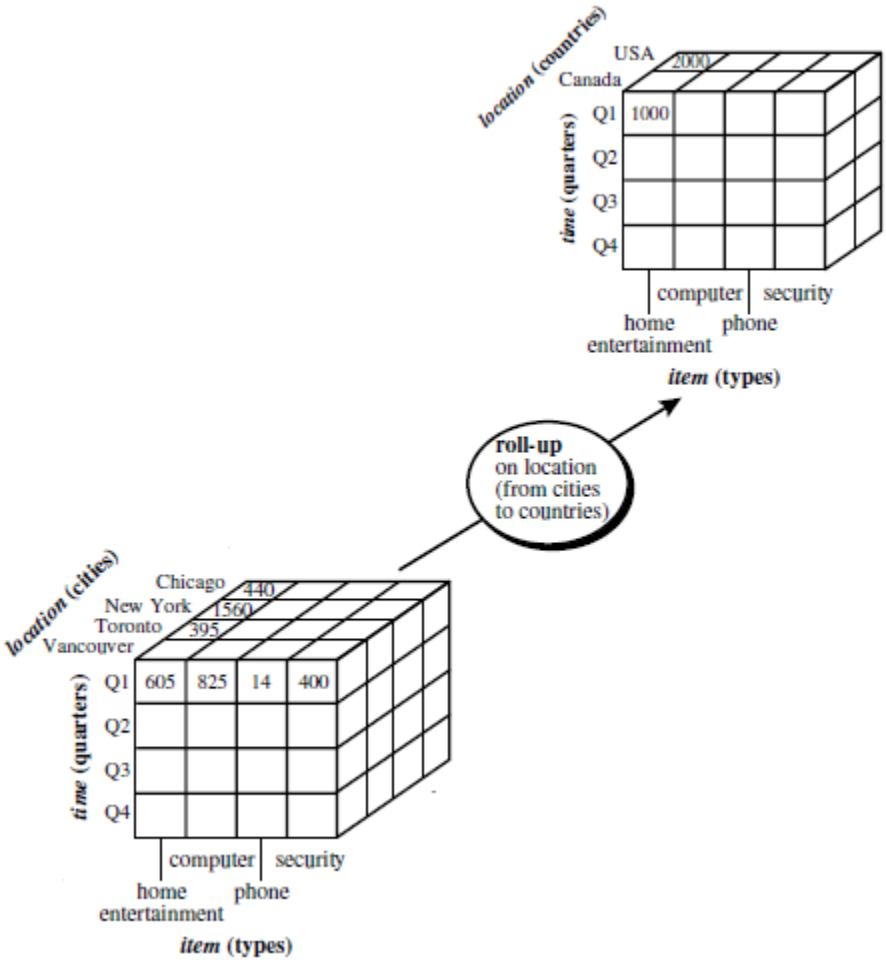


Modeling business queries: a starnet model.

7. Analyze the OLAP operation in multidimensional data. **[12M]**

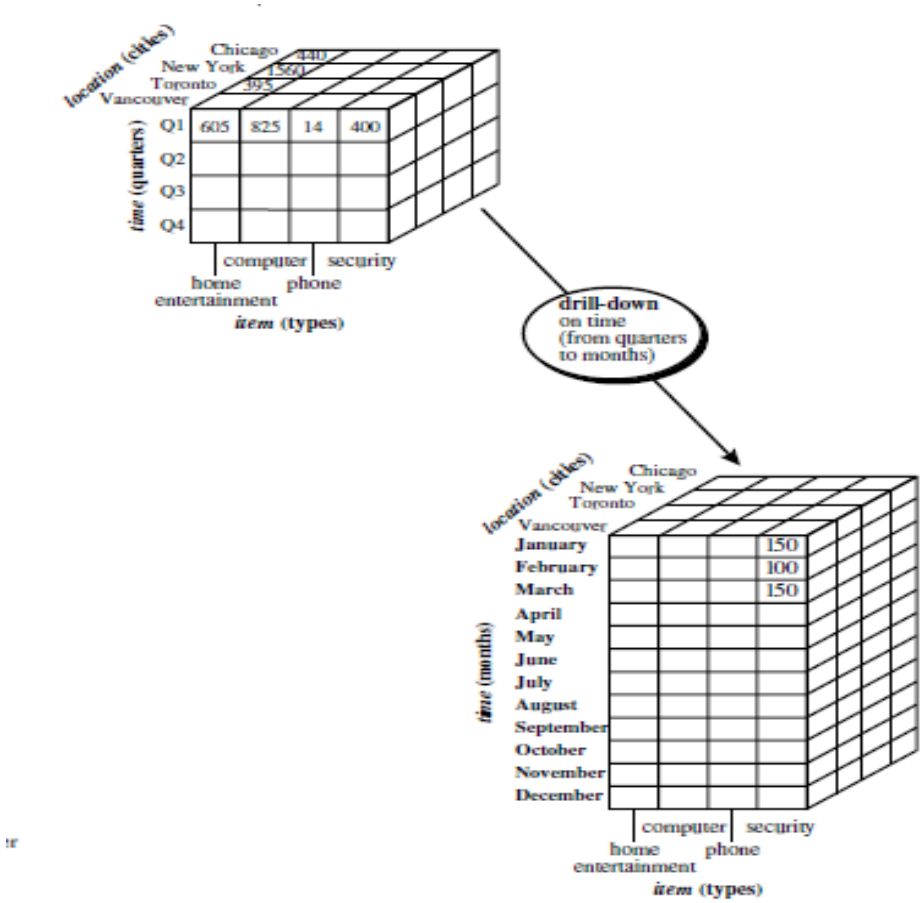
Roll-up:

- The roll-up operation performs aggregation on a data cube, either by climbing-up a concept hierarchy for a dimension or by dimension reduction.
- Figure shows the result of a roll-up operation performed on the central cube by climbing up the concept hierarchy for location.
- This hierarchy was defined as the total order street < city < province or state <country.



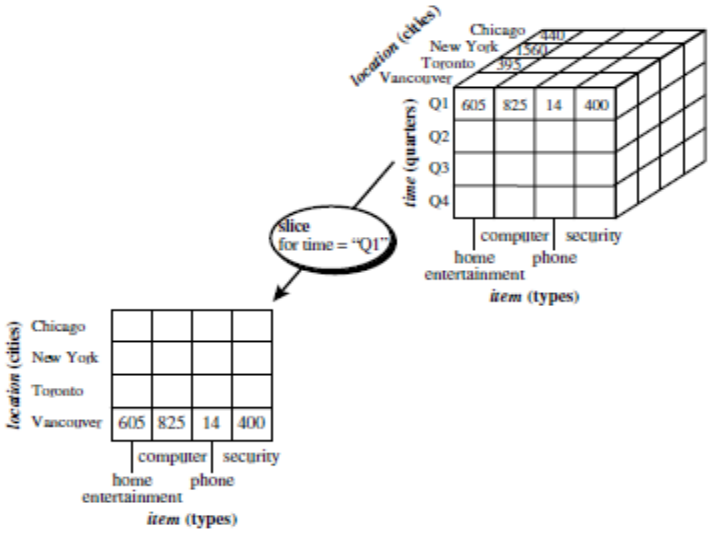
Drill-down:

- Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data.
- Drill-down can be realized by either stepping-down a concept hierarchy for a dimension or introducing additional dimensions.
- Figure shows the result of a drill-down operation performed on the central cube by stepping down a concept hierarchy for time defined as day < month < quarter < year.
- Drill-down occurs by descending the time hierarchy from the level of quarter to the more detailed level of month.



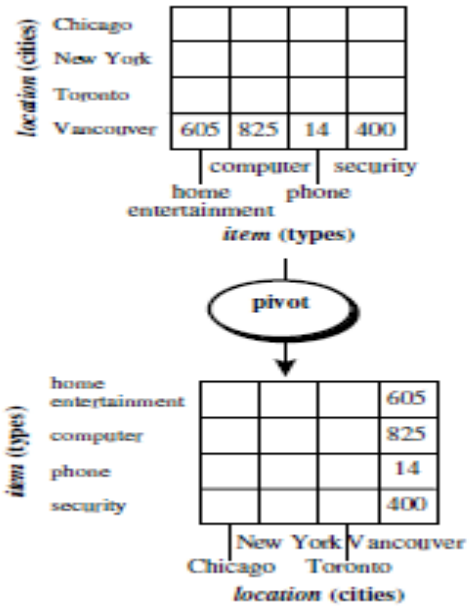
Slice and dice:

- The slice operation performs a selection on one dimension of the given cube, resulting in a sub cube.
- Figure shows a slice operation where the sales data are selected from the central cube for the dimension time using the criteria time=”Q2".
- The dice operation defines a sub cube by performing a selection on two or more dimensions.



Pivot (rotate):

- Pivot is a visualization operation which rotates the data axes in view in order to provide an alternative presentation of the data.
- Figure shows a pivot operation where the item and location axes in a 2-D slice are rotated.



8. Explain about the Three-tier data warehouse architecture with a neat Diagram. [12M]

- The **bottom tier** is a warehouse database server that is almost always a relational database system.
- **Back-end tools and utilities** are used to feed data into the bottom tier from operational databases or other external sources (such as customer profile information provided by external consultants).
- These tools and utilities perform data extraction, cleaning, and transformation (e.g., to merge similar data from different sources into a unified format), as well as load and refresh functions to update the data warehouse.
- The data are extracted using application program interfaces known as gateways. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server.
- Examples of gateways include ODBC (Open Database Connection) and OLEDB (Open Linking and Embedding for Databases) by Microsoft and JDBC (Java Database Connection). This tier also contains a metadata repository, which stores information about the data warehouse and its contents.
- The **middle tier** is an OLAP server that is typically implemented using either

  (1) a relational OLAP (ROLAP) model, that is, an extended relational DBMS that maps operations on multidimensional data to standard relational operations; or

(2) a multidimensional OLAP (MOLAP) model, that is, a special-purpose server that directly implements multidimensional data and operations.

- The top tier is a front-end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).
- From the architecture point of view, there are three data warehouse models: **the enterprise warehouse, the data mart, and the virtual warehouse.**

Enterprise warehouse:

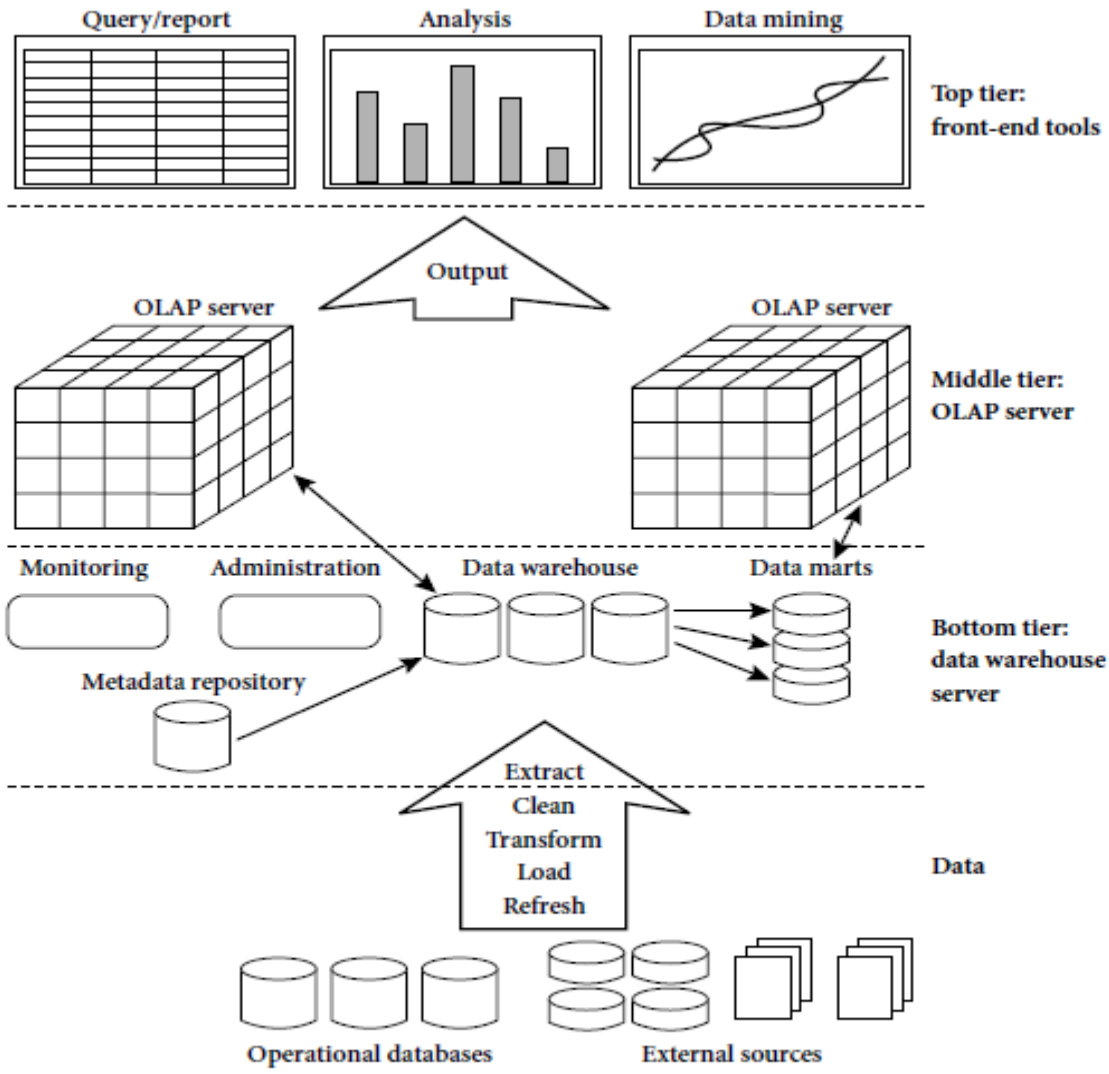
- An enterprise warehouse collects all of the information about subjects spanning the entire organization.
- It provides corporate-wide data integration, usually from one or more operational systems or external information providers, and is cross-functional in scope.
- It typically contains detailed data as well as summarized data, and can range in size from a few gigabytes to hundreds of gigabytes, terabytes, or beyond.
- An enterprise data warehouse may be implemented on traditional mainframes, computer super servers, or parallel architecture platforms. It requires extensive business modeling and may take years to design and build.

Data mart:

- A data mart contains a subset of corporate-wide data that is of value to a specific group of users. The scope is confined to specific selected subjects. For example, a marketing data mart may confine its subjects to customer, item, and sales. The data contained in data marts tend to be summarized.
- Data marts are usually implemented on low-cost departmental servers that are UNIX/LINUX- or Windows-based.
- The implementation cycle of a data mart is more likely to be measured in weeks rather than months or years.
- Depending on the source of data, data marts can be categorized as independent or dependent. Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area.
- Dependent data marts are sourced directly from enterprise data warehouses.

Virtual warehouse:

- A virtual warehouse is a set of views over operational databases. For efficient query processing, only some of the possible summary views may be materialized.
- A virtual warehouse is easy to build but requires excess capacity on operational database servers.



A three-tier data warehousing architecture.

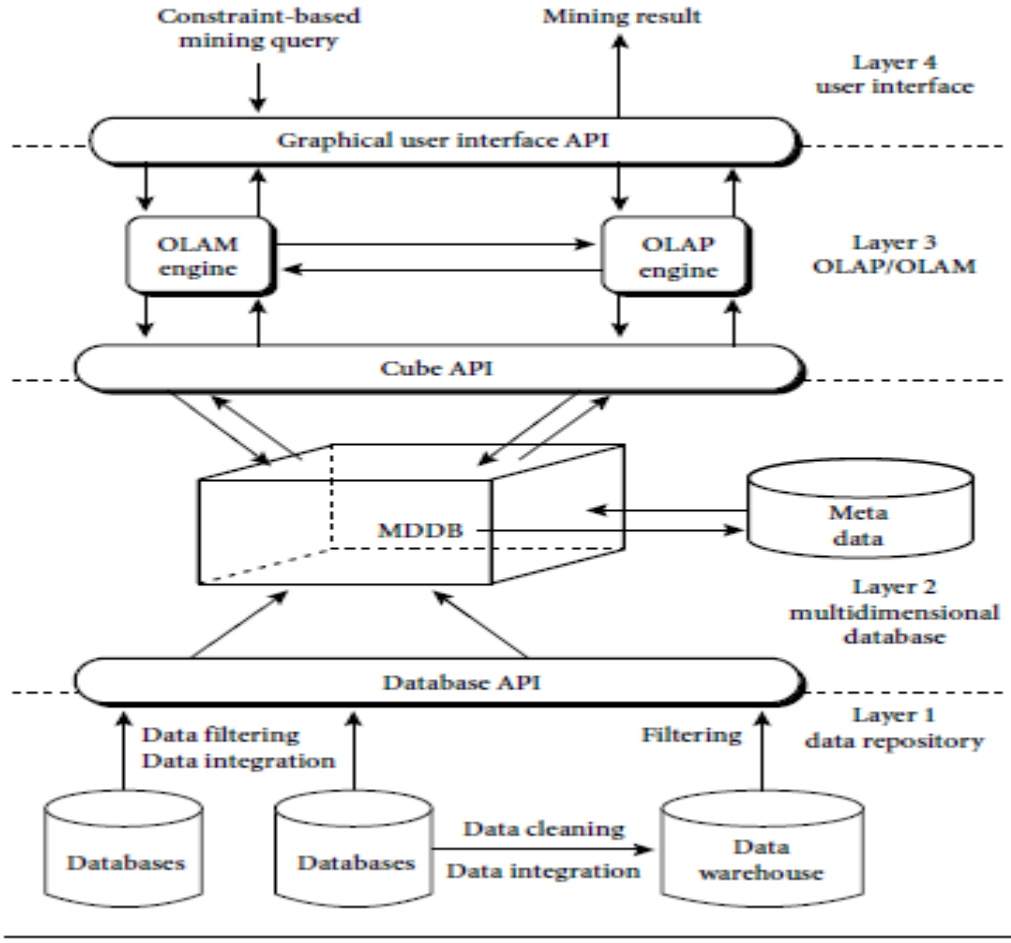
9. Define OLAM? Construct the architecture of OLAM. [12M]

- On-line analytical mining (OLAM) (also called OLAP mining) integrates on-line analytical processing (OLAP) with data mining and mining knowledge in multidimensional databases.
- Among the many different paradigms and architectures of data mining systems, OLAM is particularly important for the following reasons:
    - High quality of data in data warehouses
    - Available information processing infrastructure surrounding data warehouses
    - OLAP-based exploratory data analysis
    - On-line selection of data mining functions

Architecture for On-Line Analytical Mining

- An OLAM server performs analytical mining in data cubes in a similar manner as an OLAP server performs on-line analytical processing.

- An integrated OLAM and OLAP architecture is shown in Figure, where the OLAM and OLAP servers both accept user on-line queries (or commands) via a graphical user interface API and work with the data cube in the data analysis via a cube API.



An integrated OLAM and OLAP architecture.

- A metadata directory is used to guide the access of the data cube. The data cube can be constructed by accessing and/or integrating multiple databases via an MDDB API and/or by filtering a data warehouse via a database API that may support OLE DB or ODBC connections.
- Since an OLAM server may perform multiple data mining tasks, such as concept description, association, classification, prediction, clustering, time-series analysis, and so on, it usually consists of multiple integrated data mining modules and is more sophisticated than an OLAP server.
- Data warehousing provides users with large amounts of clean, organized, and summarized data, which greatly facilitates data mining.
- For example, rather than storing the details of each sales transaction, a data warehouse may store a summary of the transactions per item type for each branch or, summarized to a higher level, for each country.
- The capability of OLAP to provide multiple and dynamic views of summarized data in a data warehouse sets a solid foundation for successful data mining.

- Moreover, we also believe that data mining should be a human-centered process. Rather than asking a data mining system to generate patterns and knowledge automatically, a user will often need to interact with the system to perform exploratory data analysis.

10. a. Explain in brief about ROLAP, MOLAP and HOLAP servers. [6M]

Relational OLAP (ROLAP) servers:

- These are the intermediate servers that stand in between a relational back-end server and client front-end tools.
- They use a relational or extended-relational DBMS to store and manage warehouse data, and OLAP middleware to support missing pieces.
- ROLAP servers include optimization for each DBMS back end, implementation of aggregation navigation logic, and additional tools and services.
- ROLAP technology tends to have greater scalability than MOLAP technology. The DSS server of Micro strategy, for example, adopts the ROLAP approach.

Multidimensional OLAP (MOLAP) servers:

- These servers support multidimensional views of data through array-based multidimensional storage engines.
- They map multi dimensional views directly to data cube array structures. The advantage of using a data cube is that it allows fast indexing to pre computed summarized data.
- Many MOLAP servers adopt a two-level storage representation to handle dense and sparse data sets: denser sub cubes are identified and stored as array structures, whereas sparse sub cubes employ compression technology for efficient storage utilization.

Hybrid OLAP (HOLAP) servers:

- The hybrid OLAP approach combines ROLAP and MOLAP technology, benefiting from the greater scalability of ROLAP and the faster computation of MOLAP.
- For example, a HOLAP server may allow large volumes of detail data to be stored in a relational database, while aggregations are kept in a separate MOLAP store.
- The Microsoft SQL Server 2000 supports a hybrid OLAP server.

10. b. Elaborate about Attribute Oriented Induction with example. [6M]

- Data generalization summarizes data by replacing relatively low-level values (e.g., numeric values for an attribute age) with higher-level concepts (e.g., young, middle-aged, and senior), or by reducing the number of dimensions to summarize data in concept space involving fewer dimensions (e.g., removing birth date and telephone number when summarizing the behavior of a group of students).

Attribute-Oriented Induction for Data Characterization:

- First **collect the task-relevant data** using a **database query** and then **perform generalization** based on the examination of the number of each attribute's distinct values in the relevant data set.

- The **generalization** is performed by either **attribute removal or attribute generalization.**
- Aggregation is performed by merging identical generalized tuples and accumulating their respective counts.
- This reduces the size of the generalized data set. The resulting generalized relation can be mapped into different forms (e.g., charts or rules) for presentation to the user.

Example: A data mining query for characterization.

- Suppose that a user wants to describe the general characteristics of graduate students in the Big University database, given the attributes name, gender, major, birth place, birth date, residence, phone# (telephone number), and gpa (grade point average).
- A data mining query for this characterization can be expressed in the data mining query language, DMQL, as follows:

  use Big University DB

  mine characteristics as "Science Students"

  in relevance to name, gender, major, birth place, birth date, residence, phone#, gpa

  from student

  where status in "graduate" 27, 22

Table 4.12 Initial working relation: a collection of task-relevant data.

name	*gender*	*major*	*birth_place*	*birth_date*	*residence*	*phone#*	*gpa*
Jim Woodman	M	CS	Vancouver, BC, Canada	8-12-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	physics	Seattle, WA, USA	25-8-70	125 Austin Ave., Burnaby	420-5232	3.83
...	...	...	...	...	...	...	...

- The data mining query presented above is transformed into the following relational query for the collection of the task-relevant set of data:

  use Big University DB

  select name, gender, major, birth place, birth date, residence, phone#, gpa

  from student

  where status in {"M.Sc.", "M.A.", "M.B.A.", "Ph.D."}

- **Attribute removal** is based on the following rule: If there is a large set of distinct values for an attribute of the initial working relation, but either

  (case 1) there is no generalization operator on the attribute(e.g., there is no concept hierarchy defined for the attribute), or

  (case 2) its higher-level concepts are expressed in terms of other attributes, then the attribute should be removed from the working relation.

- **Attribute generalization** is based on the following rule:

  – If there is a large set of distinct values for an attribute in the initial working relation, and there exists a set of generalization operators on the attribute, then a generalization operator should be selected and applied to the attribute.

Example: Attribute-oriented induction.

- Here we show how attribute-oriented induction is performed on the initial working relation of Table 4.5. For each attribute of the relation, the generalization proceeds as follows:

Table 4.5 Initial Working Relation: A Collection of Task-Relevant Data

name	*gender*	*major*	*birth_place*	*birth_date*	*residence*	*phone#*	*gpa*
Jim Woodman	M	CS	Vancouver, BC, Canada	12-8-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	7-28-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	Physics	Seattle, WA, USA	8-25-70	125 Austin Ave., Burnaby	420-5232	3.83
…	…	…	…	…	…	…	…

1. name: large number of distinct values and no generalization operation defined on it (removed).
2. gender: only two distinct values (retained) and no generalization is performed on it.
3. major: generalized to the values {arts & sciences, engineering, business}. (retained)
4. birth place: has a large number of distinct values; therefore, we would like to generalize it and concept hierarchy can be defined as **"city < province or state < country."** The number of distinct values for **birth place > threshold** but if we take country it is < **threshold** so birth place should be generalized as country. (retained)
5. birth date: concept hierarchy existed as birth date < age < age range. So, it is generalized as age range (retained)
6. **residence:** (street < city < state < country). Generalized as state. (retained)
7. phone#: phone# contains too many distinct values and no generalization. (removed).
8. gpa: numeric intervals like {3.75 – 4.0, 3.5–3.75, . . . }, which in turn are grouped into descriptive values such as {"excellent", "very good", . . . }. and generalized. (retained)

Algorithm: Attribute - oriented induction.

- Mining generalized characteristics in a relational database given a user's data mining request.

Input:

- DB, a relational database;
- DMQuery, a data mining query;
- a list, a list of attributes (containing attributes, ai);
- Gen(ai), a set of concept hierarchies or generalization operators on attributes, ai ;
- a gen thresh(ai), attribute generalization thresholds for each ai .

Output:

- P, a Prime generalized relation.

Method:

1. $W \leftarrow$ get_task_relevant_data (*DMQuery, DB*); // Let W, the working relation, hold the task-relevant data.
2. prepare_for_generalization (W); // This is implemented as follows.
   - **(a)** Scan W and collect the distinct values for each attribute, a_i. (*Note:* If W is very large, this may be done by examining a sample of W.)
   - **(b)** For each attribute a_i, determine whether a_i should be removed. If not, compute its minimum desired level L_i based on its given or default attribute threshold, and determine the mapping pairs (v, v'), where v is a distinct value of a_i in W, and v' is its corresponding generalized value at level L_i.
3. $P \leftarrow$ generalization (W),

   The *Prime_generalized_relation*, P, is derived by replacing each value v in W by its corresponding v' in the mapping while accumulating `count` and computing any other aggregate values.

Example: Mining a class comparison.

- Suppose that you would like to compare the general properties of the graduate and undergraduate students at Big University, given the attributes name, gender, major, birth place, birth date, residence, phone#, and gpa. This data mining task can be expressed in DMQL as follows:

```
use Big University DB
mine comparison as "grad vs undergrad students"
in relevance to name, gender, major, birth place, birth date, residence,
                phone#, gpa
for "graduate students"
where status in "graduate"
versus "undergraduate students"
where status in "undergraduate"
analyze count%
from student
```

Table 4.8 Initial Working Relations: The Target Class (Graduate Students)

name	*gender*	*major*	*birth_place*	*birth_date*	*residence*	*phone#*	*gpa*
Jim Woodman	M	CS	Vancouver, BC, Canada	12-8-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	7-28-75	345 1st Ave., Vancouver	253-9106	3.70
Laura Lee	F	Physics	Seattle, WA, USA	8-25-70	125 Austin Ave., Burnaby	420-5232	3.83
...	...	...	...	...	...	...	...

Table 4.9 Initial Working Relations: The Contrasting Class (Undergraduate Students)

name	*gender*	*major*	*birth_place*	*birth_date*	*residence*	*phone#*	*gpa*
Bob Schumann	M	Chemistry	Calgary, Alt, Canada	1-10-78	2642 Halifax St., Burnaby	294-4291	2.96
Amy Eau	F	Biology	Golden, BC, Canada	3-30-76	463 Sunset Cres., Vancouver	681-5417	3.52
...	...	...	...	...	...	...	...

Table 4.10 Prime Generalized Relation for the Target Class (Graduate Students)

major	*age_range*	*gpa*	count%
Science	21...25	good	5.53
Science	26...30	good	5.02
Science	over_30	very good	5.86
...	...	...	...
Business	over_30	excellent	4.68

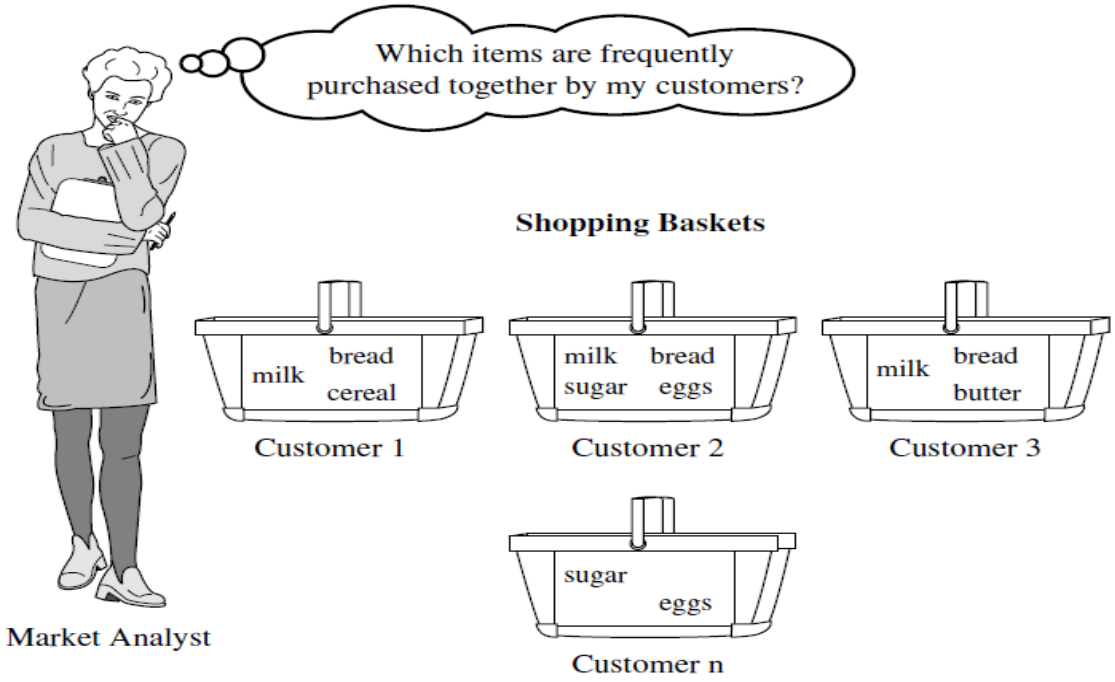
Table 4.11 Prime Generalized Relation for the Contrasting Class (Undergraduate Students)

major	*age_range*	*gpa*	count%
Science	16...20	fair	5.53
Science	16...20	good	4.53
...	...	...	...
Science	26...30	good	2.32
...	...	...	...
Business	over_30	excellent	0.68

UNIT–III
MINING FREQUENT PATTERNS, ASSOCIATIONS AND CORRELATIONS

1. a. Explain about market basket Association Mining. [6M]

- The discovery of **interesting correlation relationships** among huge amounts of business transaction records can help in many business decision-making processes, such as
  - catalog design,
  - cross-marketing, and
  - customer shopping behavior analysis.
- This process analyzes customer buying habits by finding associations between the different items that customers place in their "shopping baskets".
- The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers.
- For instance, if customers are buying milk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket?
- Such information can lead to increased sales by helping retailers do selective marketing and plan their shelf space.



- Frequent pattern mining searches for recurring relationships in a given data set.
- Frequent patterns are patterns
  - Itemsets
  - Subsequences
  - substructures
- A set of items that appear frequently together in a transaction data set is a **frequent itemset**.

- A subsequence 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**.
- These patterns can be represented in the form of **association rules**.

  computer => antivirus software [support = 2%; confidence = 60%]
- Rule **support and confidence** are two measures of rule interestingness.
- Association rules are considered interesting if they satisfy both a **minimum support threshold and a minimum confidence threshold**.
- Find all the rules X & Y => z with minimum confidence and support.
  - **support**, **s**, probability that a transaction contains {X & Y & Z}
  - **confidence**, **c**, conditional probability that a transaction having {X & Y} also contains Z
- **Frequent Itemset:** A frequent itemset is an itemset whose support is greater than some user-specified minimum support.
- **Closed Frequent Itemset:** An itemset is closed if none of its immediate supersets has the same support as that of the itemset.
- **Maximal Frequent Itemset:** An itemset is maximal frequent if none of its immediate supersets is frequent.

Downward closure property of frequent patters.

- All subset of any frequent itemsets must also be frequent.

Example:

- If **Milk, Bread, Butter** is a frequent itemset, then the following itemsets are frequent:
  - Milk
  - Bread
  - Butter
  - Milk, Bread
  - Milk, Butter
  - Bread, Butter
- In general, association rule mining can be viewed as a two-step process:
  - **Find all frequent itemsets:** By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, *min sup.*
  - **Generate strong association rules from the frequent itemsets:** By definition, these rules must satisfy minimum support and minimum confidence.

1. b. Explain support, confidence and lift measure with respect to Association Mining. [6M]

Support:

- This says how popular an itemset is, as measured by the proportion of transactions in which an itemset appears. In Table 1 below, the support of {apple} is 4 out of 8, or 50%. Itemsets can also contain multiple items. For instance, the support of {apple, beer, rice} is 2 out of 8, or 25%.

$$\text{Support } \{🍎\} = \frac{4}{8}$$

Transaction 1	🍎 🍺 ⚪ 🍗
Transaction 2	🍎 🍺 ⚪
Transaction 3	🍎 🍺
Transaction 4	🍎 🍐
Transaction 5	💉 🍺 ⚪ 🍗
Transaction 6	💉 🍺 ⚪
Transaction 7	💉 🍺
Transaction 8	💉 🍐

Table 1. Example Transactions

Confidence:

- This says how likely item Y is purchased when item X is purchased, expressed as {X -> Y}. This is measured by the proportion of transactions with item X, in which item Y also appears. In Table 1, the confidence of {apple -> beer} is 3 out of 4, or 75%.

$$\text{Confidence } \{🍎 \rightarrow 🍺\} = \frac{\text{Support } \{🍎, 🍺\}}{\text{Support } \{🍎\}}$$

Lift:

- This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is. In Table 1, the lift of {apple -> beer} is 1, which implies no association between items.
- A lift value greater than 1 means that item Y is likely to be bought if item X is bought, while a value less than 1 means that item Y is unlikely to be bought if item X is bought.

$$\text{Lift}\{🍎 \rightarrow 🍺\} = \frac{\text{Support}\{🍎, 🍺\}}{\text{Support}\{🍎\} \times \text{Support}\{🍺\}}$$

2. a. Discuss about Basic Concepts of Frequent Item set mining. [6M]

- Frequent pattern mining searches for recurring relationships in a given data set.
- Frequent patterns are patterns
  - Itemsets
  - Subsequences
  - substructures
- A set of items that appear frequently together in a transaction data set is a **frequent itemset**.
- A subsequence 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**.
- These patterns can be represented in the form of **association rules**.

  computer => antivirus software [support = 2%; confidence = 60%]
- Rule **support and confidence** are two measures of rule interestingness.
- Association rules are considered interesting if they satisfy both a **minimum support threshold and a minimum confidence threshold**.
- Find all the rules X & Y => z with minimum confidence and support.
  - **support**, **s**, probability that a transaction contains {X & Y & Z}
  - **confidence**, **c**, conditional probability that a transaction having {X & Y} also contains Z
- **Frequent Itemset:** A frequent itemset is an itemset whose support is greater than some user-specified minimum support.
- **Closed Frequent Itemset:** An itemset is closed if none of its immediate supersets has the same support as that of the itemset.
- **Maximal Frequent Itemset:** An itemset is maximal frequent if none of its immediate supersets is frequent.

Downward closure property of frequent patters.

- All subset of any frequent itemsets must also be frequent.

Example:

- If **Milk, Bread, Butter** is a frequent itemset, then the following itemsets are frequent:
  - Milk
  - Bread
  - Butter
  - Milk, Bread

  - Milk, Butter
  - Bread, Butter
- In general, association rule mining can be viewed as a two-step process:
  - **Find all frequent itemsets:** By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, min sup.
  - **Generate strong association rules from the frequent itemsets:** By definition, these rules must satisfy minimum support and minimum confidence.
- Frequent pattern mining can be classified in various ways, based on the following criteria:

Based on the completeness of patterns to be mined:

- We can mine the complete set of frequent itemsets, the **closed frequent itemsets**, and **the maximal frequent itemsets**, **constrained frequent itemsets**, **approximate frequent itemsets**, **near-match frequent itemsets**, **top-*k* frequent itemsets.**

Based on the levels of abstraction involved in the rule set:

- Some methods for association rule mining can find rules at differing **levels of abstraction**.

Based on the number of data dimensions involved in the rule:

- If the items or attributes in an association rule reference only one dimension, then it is a **single-dimensional association rule**.
- If a rule references two or more dimensions, such as the dimensions age, income, and buys, then it is a **multidimensional association rule**.

Based on the types of values handled in the rule:

- If a rule involves associations between the **presence or absence of items**, it is a Boolean association rule.

Based on the kinds of rules to be mined:

- Frequent pattern analysis can generate various kinds of rules and other interesting relationships.
- **Association rules** are the most popular kind of rules generated from frequent patterns.

Based on the kinds of patterns to be mined:

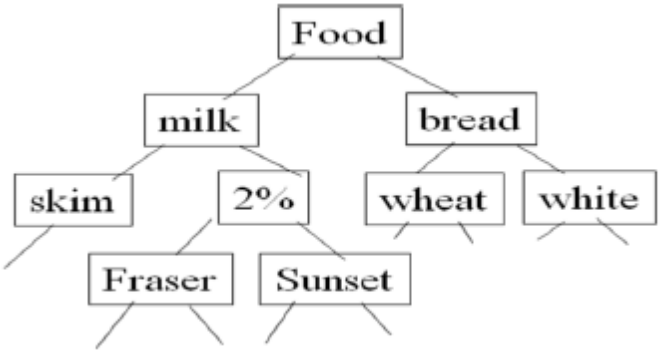
- Many kinds of frequent patterns can be mined from different kinds of data sets.
- Such as **Sequential pattern mining** searches for frequent subsequences in a sequence data set, where a sequence records an ordering of events.
- **Structured pattern mining** searches for frequent substructures in a structured data set.

2. b. What are the advantages of FP-Growth algorithm? [6M]

1. This algorithm needs to scan the database twice when compared to Apriori, which scans the transactions for each iteration.
2. The pairing of items is not done in this algorithm, making it faster.
3. The database is stored in a compact version in memory.
4. It is efficient and scalable for mining both long and short frequent patterns.

3. a. Explain Multilevel Association rules for mining data. [6M]

- Multilevel association rules can be defined as applying association rules over different levels of data abstraction.



Ex:

Steps to perform multilevel association rules from transactional database are:

Step 1: Consider frequent item sets.

Step 2: Arrange the items in hierarchy form.

Step 3: Find the Items at the lower level (expected to have lower support).

Step 4: Apply association rules on frequent item sets.

Step 5: Use some methods and identify frequent itemsets.

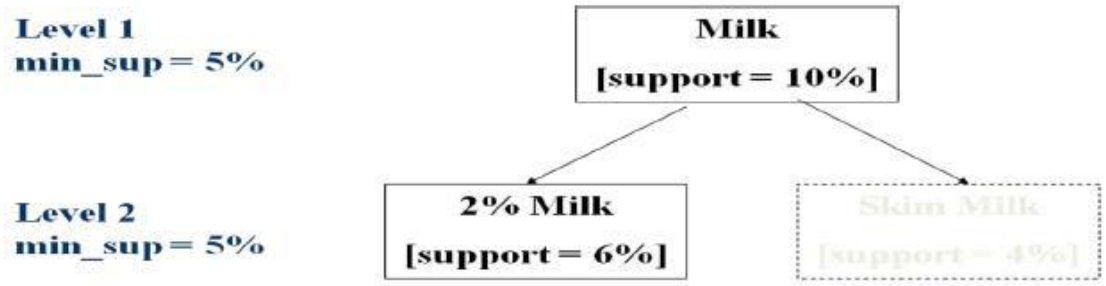
1. Uniform Support: The same minimum support for all levels.

2. Reduced Support: Reduced minimum support at lower levels.

Multilevel association rules can be applied on both the supports.

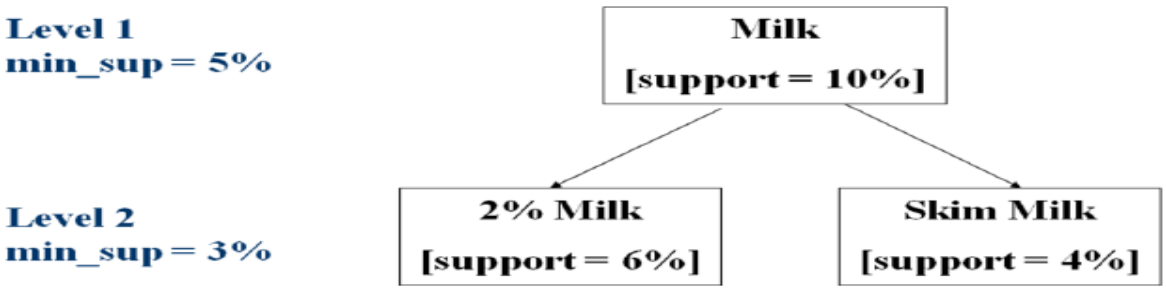
Example figure for Uniform Support:

Multi-level mining with uniform support.



Reduced Support:

Multi-level mining with reduced support.



3. b. Discuss in details Multidimensional association rules for mining data. [6M]

- Multi dimensional association rule can be defined as the statement which contains only two (or) more predicates/dimensions.
- Multi dimensional association rule also called as Inter dimensional association rule.
- We can perform the following association rules on relational database and data warehouse.

    1) Boolean dimensional association rule: Boolean dimensional association rule can be defined as comparing existing predicates/dimensions with non existing predicates/dimensions.

    2) Single dimensional association rule: Single dimensional association rule can be defined as the statement which contains only single predicate/dimension.

- Single dimensional association rule also called as Intra dimensional association rule as usually multi dimensional association rule.
- Multi dimensional association rule can be applied on different types of attributes, the attributes are

    1. Categorical Attributes.

        – finite number of possible values, no ordering among values.

    2. Quantitative Attributes.

        – numeric, implicit ordering among values.

- In relational database we are using the concept hierarchy that means generalization in order to find out the frequent item sets.
- Generalization: replacing low level attributes with high level attributes called generalization.

4. Describe the steps involved in improving the efficiency of the Apriori algorithm. [12M]

- Many variations of the Apriori algorithm have been proposed that focus on improving the efficiency of the original algorithm.
- Several of these variations are summarized as follows:

1**. Hash-based technique (hashing itemsets into corresponding buckets):**

- A hash-based technique can be used to reduce the size of the candidate *k*-itemsets, *Ck*, 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.

Create hash table H_2 using hash function $h(x, y) = ((order\ of\ x) \times 10 + (order\ of\ y))\ mod\ 7$ →

H_2

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}

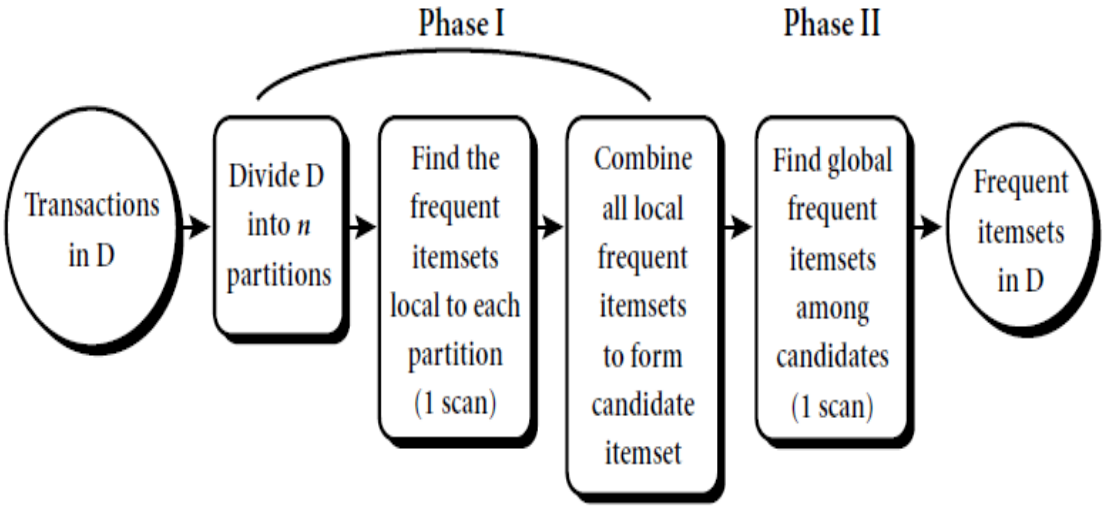
- A 2-itemset whose corresponding bucket count in the hash table is below the support threshold cannot be frequent and thus should be removed from the candidate set.

2. Transaction reduction (reducing the number of transactions scanned in future iterations):

- 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 because subsequent scans of the database for j-itemsets, where j > k, will not require it.

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

- A partitioning technique can be used that requires just two database scans to mine the frequent itemsets.
- It consists of two phases. In Phase I, the algorithm subdivides the transactions of *D* into n non overlapping partitions.
- If the minimum support threshold for transactions in *D* is min sup, then the minimum support count for a partition is min sup the number of transactions in that partition.
- For each partition, all frequent itemsets within the partition are found. These are referred to as local frequent itemsets.
- Therefore, all local frequent itemsets are candidate itemsets with respect to *D*. The collection of frequent itemsets from all partitions forms the global candidate itemsets with respect to *D*.
- In Phase II, a second scan of *D* is conducted in which the actual support of each candidate is assessed in order to determine the global frequent itemsets.
- Partition size and the number of partitions are set so that each partition can fit into main memory and therefore be read only once in each phase.



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 frequent itemsets in *S* instead of *D*.

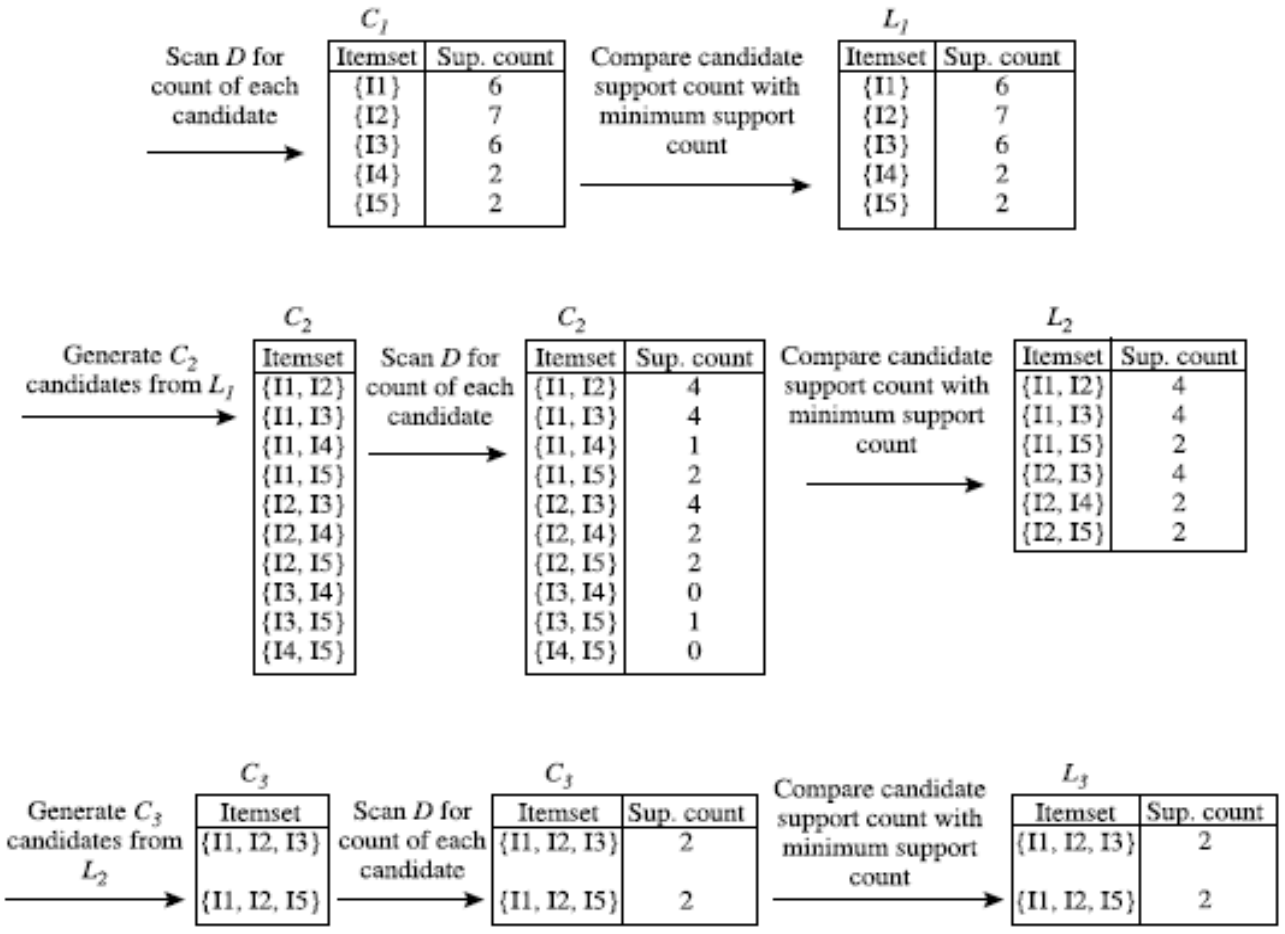
5. Dynamic itemset counting (adding candidate itemsets at different points during a scan):

- 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 determines new candidate itemsets only immediately before each complete database scan.
- The technique is dynamic in that it estimates the support of all of the itemsets that have been counted so far, adding new candidate itemsets if all of their subsets are estimated to be frequent.
- The resulting algorithm requires fewer database scans than Apriori.

5. Explain about the Apriori algorithm for finding frequent item sets with an example.

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

Generate the list of frequent item-set ordered by their corresponding Suffixes, where the minimum support count is 2 and Minimum Confidenceis 60%. [12M]



Generation of candidate itemsets and frequent itemsets, where the minimum support count is 2.

Generating Association Rules from Frequent Itemsets

- Once the frequent itemsets from transactions in a database *D* have been found, it is straightforward to generate strong association rules from them (where *strong* association rules satisfy both minimum support and minimum confidence).

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

Suppose the data contain the frequent itemset l = {I1, I2, I5}. What are the association rules that can be generated from l?

The nonempty subsets of l are **{I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}.** The resulting association rules are as shown below, each listed with its confidence:

*I*1^*I*2 => *I*5, *confidence* = 2=4 = 50%

*I*1^*I*5 => *I*2, *confidence* = 2=2 = 100%

*I*2^*I*5 => *I*1, *confidence* = 2=2 = 100%

*I*1 => *I*2^*I*5, *confidence* = 2=6 = 33%

*I*2 => *I*1^*I*5, *confidence* = 2=7 = 29%
*I*5 => *I*1^*I*2, *confidence* = 2=2 = 100%

- If the minimum confidence threshold is, say, 70%, then only the second, third, and last rules above are output, because these are the only ones generated that are strong.

6. a. What are the Drawbacks of Apriori Algorithm? [6M]

- One of the biggest limitations of the Apriori Algorithm is that it is slow. This is so because of the bare decided by the:
    1. A large number of itemsets in the Apriori algorithm dataset.
    2. Low minimum support in the data set for the Apriori algorithm.
    3. The time needed to hold a large number of candidate sets with many frequent itemsets.
    4. Thus it is inefficient when used with large volumes of datasets.

6. b. Explain about FP Growth Concept in Detail? [6M]

FP-tree construction:

Input: A transaction database DB and a minimum support threshold

Output: FP-tree, the frequent-pattern tree of DB.

Method: The FP-tree is constructed as follows.

1. The first step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the first step of Apriori. The count of 1-itemsets in the database is called support count or frequency of 1-itemset.
2. The second step is to construct the FP tree. For this, create the root of the tree. The root is represented by null.
3. The next step is to scan the database again and examine the transactions. Examine the first transaction and find out the itemset in it. The itemset with the max count is taken at the top, and then the next itemset with the lower count. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.
4. The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch, then this transaction branch would share a common prefix to the root. This means that the common itemset is linked to the new node of another itemset in this transaction.
5. Also, the count of the itemset is incremented as it occurs in the transactions. The common node and new node count are increased by 1 as they are created and linked according to transactions.

6. The next step is to mine the created FP Tree. For this, the lowest node is examined first, along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1. From this, traverse the path in the FP Tree. This path or paths is called a conditional pattern base. A conditional pattern base is a sub-database consisting of prefix paths in the FP tree occurring with the lowest node (suffix).
7. Construct a Conditional FP Tree, formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP Tree.
8. Frequent Patterns are generated from the Conditional FP Tree.

7. Make use of the database which has five transactions. Let minimum Support = 60% and minimum confidence = 80%.

Transaction	Items
T10	M, O, N, K, E, Y
T20	D, O, N, K, E, Y
T30	M, A, K, E
T40	M, U, C, K, Y
T50	C, O, O, K, I, E

Find all frequent item sets using FP-growth. **[12M]**

FP Tree :-

Lets Min. support = 3.

C1 Table :-

Item	sup-count
M	3
O	3
N	2
K	5
E	4
Y	3
D	1
A	1
U	1
C	2
I	1

L1 Table :-

Item	sup-count
M	3
O	3
K	5
E	4
Y	3

Sorted L1 Table :-

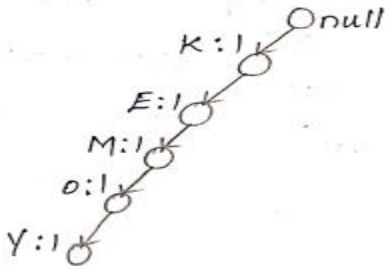
Item	sup-count
K	5
E	4
M	3
O	3
Y	3

→ Rewrite transaction table as per the order of sorted L1 table:

Tid	Items
T100	{K, E, M, O, Y}
T200	{K, E, O, Y}
T300	{K, E, M}
T400	{K, M, Y}
T500	{K, E, O}

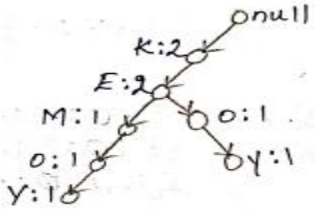
Step 1 :-

T10 :. {K, E, M, O, Y}



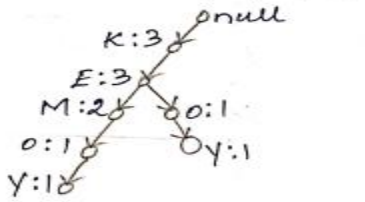
Step 2 :-

T20 : {K, E, O, Y}



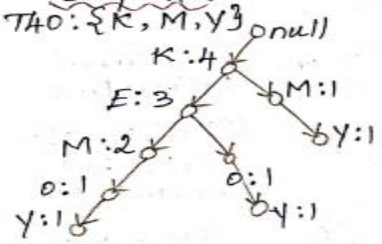
Step 3 :-

T30 : {K, E, M}



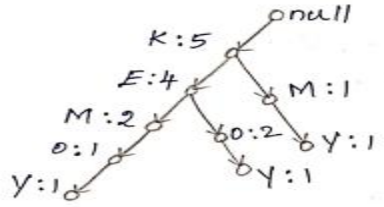
Step 4 :-

T40 : {K, M, Y}

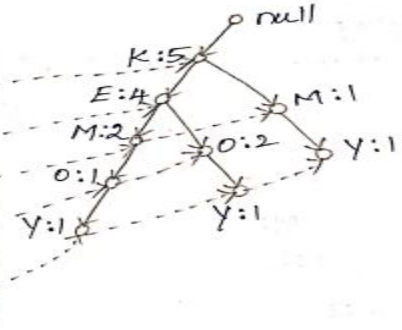


Step 5 :-

T50 :- {K, E, O}



Item	sup-count	
K	5	
E	4	
M	3	
O	3	
Y	3	



Item	conditional Pattern Base	conditional FP Tree	Frequent Itemset 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 : 3>, <E : 3>	{K, O : 3}, {E, O : 3}, {K, E, O : 3}
M	{K, E : 2}, {K : 1}	{K : 3}	{K, M} : 3
E	{K : 4}	{K : 4}	{K, E} : 4

8. Explain about Apriori Algorithm with an example. [12M]

- Apriori algorithm was the first algorithm that was proposed for frequent itemset mining. It was later improved by R Agarwal and R Srikant and came to be known as Apriori.
- This algorithm uses two steps "**join**" and "**prune**" to reduce the search space. It is an iterative approach to discover the most frequent itemsets.

Apriori says:

- The probability that item I is not frequent is if:
  - P(I) < minimum support threshold, then I is not frequent.
  - P (I+A) < minimum support threshold, then I+A is not frequent, where A also belongs to itemset.
  - If an itemset set has value less than minimum support then all of its supersets will also fall below min support, and thus can be ignored. This property is called the Anti monotone property.

The steps followed in the Apriori Algorithm of data mining are:

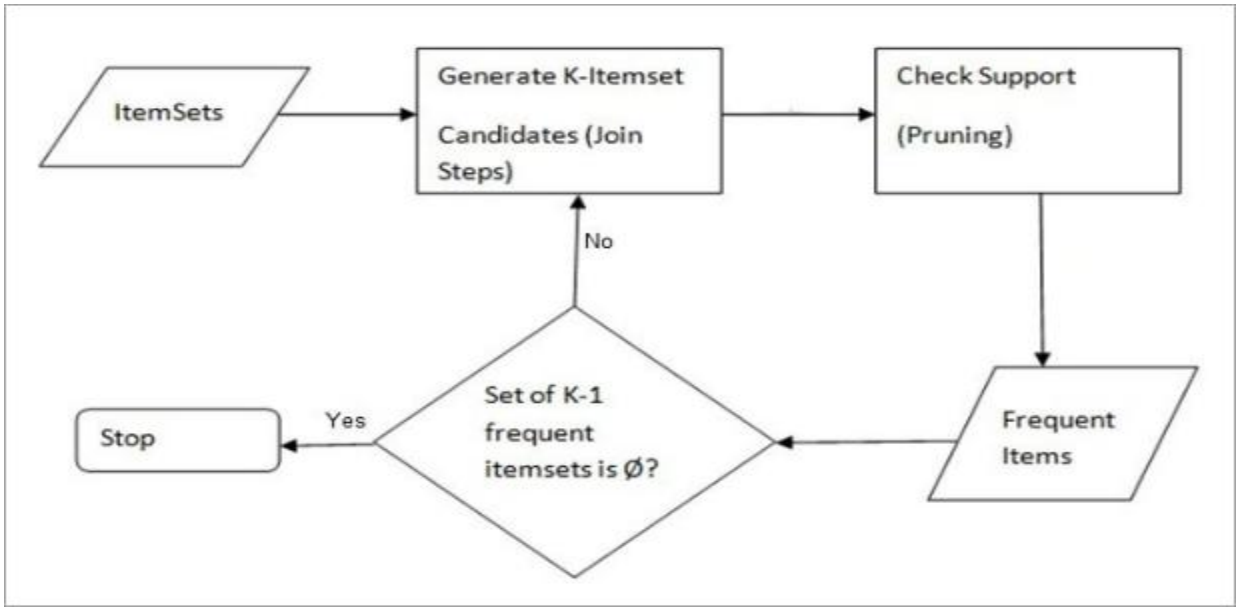
1. **Join Step**: This step generates (K+1) itemset from K-itemsets by joining each item with itself.
2. **Prune Step**: This step scans the count of each item in the database. If the candidate item does not meet minimum support, then it is regarded as infrequent and thus it is removed. This step is performed to reduce the size of the candidate itemsets.

1) In the first iteration of the algorithm, each item is taken as a 1-itemsets candidate. The algorithm will count the occurrences of each item.

2) Let there be some minimum support, min_sup (eg. 2). The set of 1 – itemsets whose occurrence is satisfying the min sup are determined. Only those candidates which count more than or equal to min_sup, are taken ahead for the next iteration and the others are pruned.

3) Next, 2-itemset frequent items with min_sup are discovered. For this in the join step, the 2-itemset is generated by forming a group of 2 by combining items with itself.

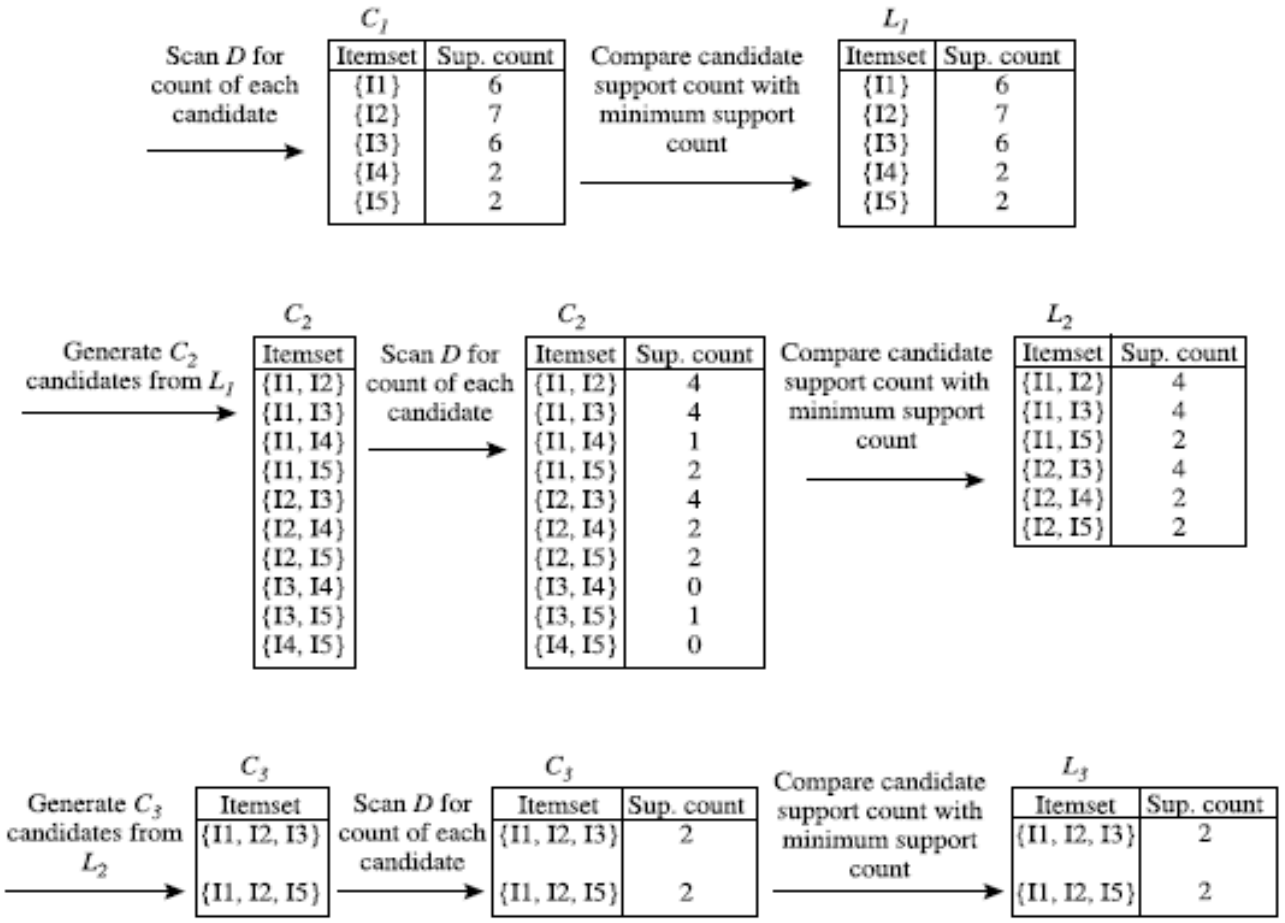
4) The 2-itemset candidates are pruned using min-sup threshold value. Now the table will have 2 –itemsets with min-sup only.
5) The next iteration will form 3 –itemsets using join and prune step. This iteration will follow antimonotone property where the subsets of 3-itemsets, that is the 2 –itemset subsets of each group fall in min_sup. If all 2-itemset subsets are frequent then the superset will be frequent otherwise it is pruned.
6) Next step will follow making 4-itemset by joining 3-itemset with itself and pruning if its subset does not meet the min_sup criteria. The algorithm is stopped when the most frequent itemset is achieved.



Example:

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

Generate the list of frequent item-set ordered by their corresponding Suffixes, where the minimum support count is 2 and Minimum Confidenceis 60%.



Generation of candidate itemsets and frequent itemsets, where the minimum support count is 2.

Generating Association Rules from Frequent Itemsets

- Once the frequent itemsets from transactions in a database *D* have been found, it is straightforward to generate strong association rules from them (where *strong* association rules satisfy both minimum support and minimum confidence).

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

Suppose the data contain the frequent itemset l = {I1, I2, I5}. What are the association rules that can be generated from l?

The nonempty subsets of l are **{I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}.** The resulting association rules are as shown below, each listed with its confidence:

*I*1^*I*2 => *I*5, *confidence* = 2=4 = 50%

*I*1^*I*5 => *I*2, *confidence* = 2=2 = 100%

*I*2^*I*5 => *I*1, *confidence* = 2=2 = 100%

*I*1 => *I*2^*I*5, *confidence* = 2=6 = 33%

*I*2 => *I*1^*I*5, *confidence* = 2=7 = 29%
*I*5 => *I*1^*I*2, *confidence* = 2=2 = 100%

- If the minimum confidence threshold is, say, 70%, then only the second, third, and last rules above are output, because these are the only ones generated that are strong.

9. a. Outline FP growth algorithm with an example. [6M]

FP-tree construction:

Input: A transaction database DB and a minimum support threshold

Output: FP-tree, the frequent-pattern tree of DB.

Method: The FP-tree is constructed as follows.

1. The first step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the first step of Apriori. The count of 1-itemsets in the database is called support count or frequency of 1-itemset.
2. The second step is to construct the FP tree. For this, create the root of the tree. The root is represented by null.
3. The next step is to scan the database again and examine the transactions. Examine the first transaction and find out the itemset in it. The itemset with the max count is taken at the top, and then the next itemset with the lower count. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.
4. The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch, then this transaction branch would share a common prefix to the root. This means that the common itemset is linked to the new node of another itemset in this transaction.
5. Also, the count of the itemset is incremented as it occurs in the transactions. The common node and new node count are increased by 1 as they are created and linked according to transactions.
6. The next step is to mine the created FP Tree. For this, the lowest node is examined first, along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1. From this, traverse the path in the FP Tree. This path or paths is called a conditional pattern base. A conditional pattern base is a sub-database consisting of prefix paths in the FP tree occurring with the lowest node (suffix).
7. Construct a Conditional FP Tree, formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP Tree.
8. Frequent Patterns are generated from the Conditional FP Tree.

Example:

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:
Support threshold=50% => 0.5*6= 3 => min_sup=3

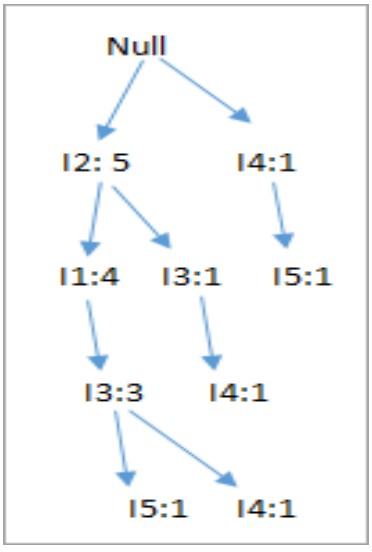
1. Count of each item
Table 2

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

2. Sort the itemset in descending order.
Table 3

Item	Count
I2	5
I1	4
I3	4
I4	4

3. Build FP Tree



Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I4	{I2,I1,I3:1},{I2,I3:1}	{I2:2, I3:2}	{I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
I3	{I2,I1:3},{I2:1}	{I2:4, I1:3}	{I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}
I1	{I2:4}	{I2:4}	{I2,I1:4}

9. b. How will measure from Association Analysis to Correlation Analysis. [6M]

- Correlation measure can be used to augment the support-confidence framework for association rules. This leads to correlation rules of the form

$$A \Rightarrow B\ [support, confidence. correlation].$$

- That is, a correlation rule is measured not only by its support and confidence but also by the correlation between itemsets *A* and *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* and *B* are dependent and correlated as events.
- This definition can easily be extended to more than two itemsets. The lift between the occurrence of *A* and *B* can be measured by computing

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

Correlation analysis using lift.

A 2 × 2 contingency table summarizing the transactions with respect to game and video purchases.

	game	$\overline{game}$	Σ_{row}
video	4,000	3,500	7,500
$\overline{video}$	2,000	500	2,500
Σ_{col}	6,000	4,000	10,000

The above contingency table, now shown with the expected values.

	game	$\overline{game}$	Σ_{row}
video	4,000 (4,500)	3,500 (3,000)	7,500
$\overline{video}$	2,000 (1,500)	500 (1,000)	2,500
Σ_{col}	6,000	4,000	10,000

Correlation analysis using χ2. To compute the correlation using χ2 analysis, we need the observed value and expected value (displayed in parenthesis) for each slot of the contingency table.

$$\chi^2 = \Sigma \frac{(observed - expected)^2}{expected} = \frac{(4{,}000 - 4{,}500)^2}{4{,}500} + \frac{(3{,}500 - 3{,}000)^2}{3{,}000} + \frac{(2{,}000 - 1{,}500)^2}{1{,}500} + \frac{(500 - 1{,}000)^2}{1{,}000} = 555.6.$$

10. Explain about Constraint based Association mining. [12M]

- Constraint based association mining can be defined as applying different types of constraints on different types of knowledge.
- So the kinds of constraints used in the mining are
    1. Knowledge type constraint.
    2. Data constraint.
    3. Dimension/level constraints.
    4. Interestingness constraints.
    5. Rule constraints.

Knowledge type constraints:

- These specify the type of knowledge to be mined, such as association or correlation.

Data constraints:

- These specify the set of task-relevant data.

Dimension/level constraints:

- These specify the desired dimensions (or attributes) of the data, or levels of the concept hierarchies, to be used in mining.

Interestingness constraints:

- These specify thresholds on statistical measures of rule interestingness, such as support, confidence, and correlation.

Rule constraints:

- These specify the form of rules to be mined. Such constraints may be expressed as metarules (rule templates), as the maximum or minimum number of predicates that can occur in the rule antecedent or consequent, or as relationships among attributes, attribute values, and/or aggregates.

Metarule - Guided Mining of Association Rule:

- Metarules allow users to specify the syntactic form of rules that they are interested in mining.
- The rule forms can be used as constraints to help improve the efficiency of the mining process.
- Metarules may be based on the analyst's experience, expectations, or intuition regarding the data or may be automatically generated based on the database schema.

Constraint Pushing: Mining Guided by Rule Constraints:

- Rule constraints specify expected set/subset relationships of the variables in the mined rules, constant initiation of variables, and aggregate functions.
- Users typically employ their knowledge of the application or data to specify rule constraints for the mining task.
- These rule constraints may be used together with, or as an alternative to, meta rule-guided mining.

A closer look at mining guided by rule constraints. Suppose that AllElectronics has a sales multidimensional database with the following interrelated relations:

- sales(customer name, item name, TID)

- lives in(customer name, region, city)
- item(item name, group, price)
- transaction(TID, day, month, year)

where lives in, item, and transaction are three dimension tables, linked to the fact table sales via three keys, customer name, item name, and TID (transaction id), respectively.
Our association mining query is to "Find the sales of which cheap items (where the sum of the prices is less than $100) may promote the sales of which expensive items (where the minimum price is $500) of the same group for Chicago customers in 2004." This can be expressed in the DMQL data mining query language as follows.

(1) mine associations as
(2) $lives\_in(C, \_, \text{“}Chicago\text{”}) \wedge sales^{+}(C, ?\{I\}, \{S\}) \Rightarrow sales^{+}(C, ?\{J\}, \{T\})$
(3) from sales
(4) where S.year = 2004 and T.year = 2004 and I.group = J.group
(5) group by C, I.group
(6) having sum(I.price) < 100 and min(J.price) ≥ 500
(7) with support threshold = 1%
(8) with confidence threshold = 50%

Line 1 is a knowledge type constraint, where association patterns are to be discovered.
Line 2 specifies a metarule. This is an abbreviated form for the following metarule for hybrid-dimensional association rules (multidimensional association rules where the repeated predicate here is *sales*):

$$lives\_in(C, \_, \text{“}Chicago\text{”}) \wedge sales(C, ?I_1, S_1) \wedge \ldots \wedge sales(C, ?I_k, S_k) \wedge I = \{I_1, \ldots, I_k\} \wedge S = \{S_1, \ldots, S_k\} \Rightarrow sales(C, ?J_1, T_1) \wedge \ldots \wedge sales(C, ?J_m, T_m) \wedge J = \{J_1, \ldots, J_m\} \wedge T = \{T_1, \ldots, T_m\}$$

The metarule may allow the generation of association rules like the following:

$$lives\_in(C, \_, \text{“}Chicago\text{”}) \wedge sales(C, \text{“}Census\_CD\text{”}, \_) \wedge sales(C, \text{“}MS/Office\text{”}, \_) \Rightarrow sales(C, \text{“}MS/SQLServer\text{”}, \_) \quad [1.5\%, 68\%],$$

Rule constraints can be classified into the following five categories with respect to frequent itemset mining:
(1) antimonotonic - if an itemset does not satisfy this rule constraint, none of its supersets can satisfy the constraint.
(2) monotonic - if an itemset satisfies this rule constraint, so do all of its supersets.
(3) succinct - if a rule constraint is succinct, we can directly generate precisely the sets that satisfy it, even before support counting begins.
(4) convertible - Some constraints belong to none of the above three categories. However, if the items in the itemset are arranged in a particular order, the constraint may be come monotonic or antimonotonic with regard to the frequent itemset mining process.
(5) inconvertible - there still exist some tough constraints that are not convertible.

UNIT–IV

CLASSIFICATION AND PREDICTION

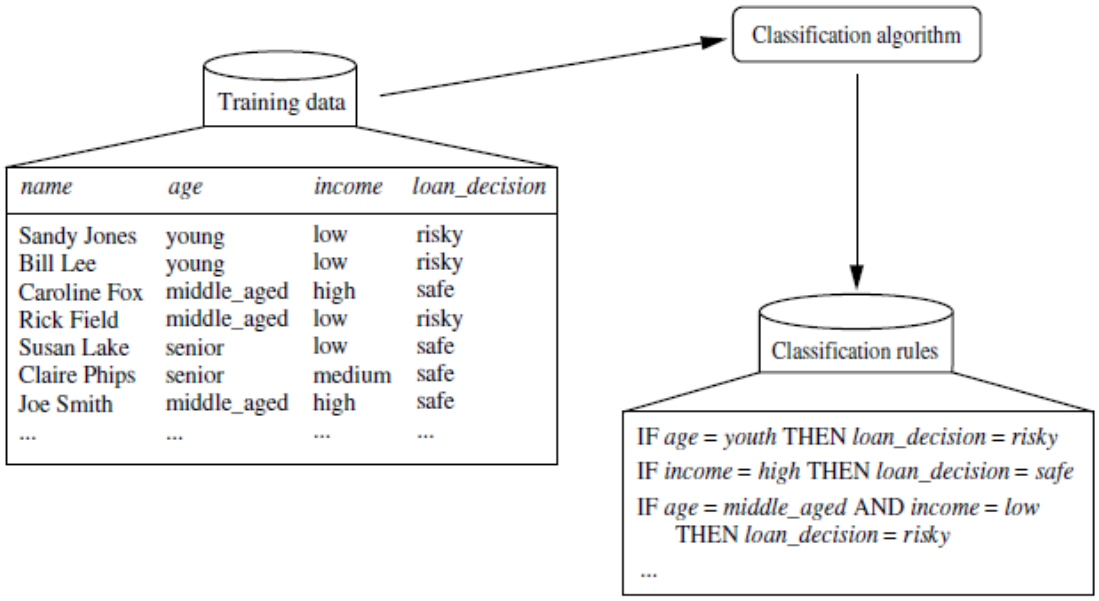
1. Explain the following terms **[12M]**

i) Classification

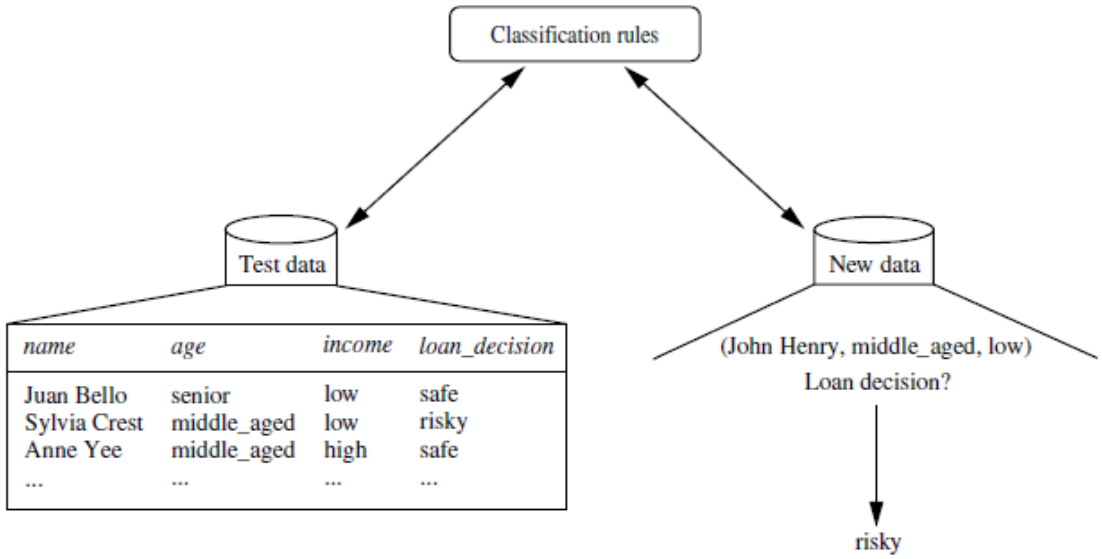
ii) Prediction

i) Classification:

- Data classification is a two-step process. In the first step, a classifier is built describing a predetermined set of data classes or concepts. This is the learning step (or training phase), where a classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels.
- A tuple, ***X***, is represented by an *n*-dimensional attribute vector, $X = (x_1, x_2, \ldots, x_n)$, depicting *n* measurements made on the tuple from *n* database attributes, respectively, A_1, $A_2, \ldots, A_n$.
- Each tuple, *X*, is assumed to belong to a predefined class as determined by another database attribute called the class label attribute.
- The class label attribute is discrete-valued and unordered. It is categorical in that each value serves as a category or class.
- The individual tuples making up the training set are referred to as training tuples and are selected from the database under analysis.
- In the context of classification, data tuples can be referred to as samples, examples, instances, data points, or objects.
- Because the class label of each training tuple is provided, this step is also known as supervised learning.
- It contrasts with unsupervised learning, in which the class label of each training tuple is not known, and the number or set of classes to be learned may not be known in advance.
- This first step of the classification process can also be viewed as the learning of a mapping or function, $y = f(\boldsymbol{X})$, that can predict the associated class label *y* of a given tuple ***X***.
- In this view, we wish to learn a mapping or function that separates the data classes. Typically, this mapping is represented in the form of classification rules, decision trees, or mathematical formulae.
- The rules can be used to categorize future data tuples, as well as provide deeper insight into the database contents. They also provide a compressed representation of the data.



- In the second step, the model is used for classification. First, the predictive accuracy of the classifier is estimated. If we were to use the training set to measure the accuracy of the classifier, this estimate would likely be optimistic, because the classifier tends to overfit the data.
- Therefore, a test set is used, made up of test tuples and their associated class labels. These tuples are randomly selected from the general data set.
- They are independent of the training tuples, meaning that they are not used to construct the classifier. The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.
- The associated class label of each test tuple is compared with the learned classifier's class prediction for that tuple.
- If the accuracy of the classifier is considered acceptable, the classifier can be used to classify future data tuples for which the class label is not known.



ii) Prediction:

- Data prediction is a two step process. However, for prediction, we lose the terminology of "class label attribute" because the attribute for which values are being predicted is continuous-valued (ordered) rather than categorical (discrete-valued and unordered).
- The attribute can be referred to simply as the predicted attribute. The data mining task becomes prediction, rather than classification.
- We would replace the categorical attribute, loan decision, with the continuous-valued loan amount as the predicted attribute, and build a predictor for our task.
- Prediction can also be viewed as a mapping or function, y= f (**X**), where **X** is the input, and the output y is a continuous or ordered value.
- That is, we wish to learn a mapping or function that models the relationship between ***X*** and *y*. Prediction and classification also differ in the methods that are used to build their respective models.
- As with classification, the training set used to build a predictor should not be used to assess its accuracy. An independent test set should be used instead.
- The accuracy of a predictor is estimated by computing an error based on the difference between the predicted value and the actual known value of *y* for each of the test tuples, ***X***.

2. a. Distinguish between supervised and unsupervised learning. [6M]

Parameters	Supervised machine learning technique	Unsupervised machine learning technique
Process	In a supervised learning model, input and output variables will be given.	In unsupervised learning model, only input data will be given
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data which is not labeled
Algorithms Used	Support vector machine, Neural network, Linear and logistics regression, random forest, and Classification trees.	Unsupervised algorithms can be divided into different categories: like Cluster algorithms, K-means, Hierarchical clustering, etc.
Computational Complexity	Supervised learning is a simpler method.	Unsupervised learning is computationally complex
Use of Data	Supervised learning model uses training data to learn a link between the input and the outputs.	Unsupervised learning does not use output data.
Accuracy of Results	Highly accurate and trustworthy method.	Less accurate and trustworthy method.
Real Time Learning	Learning method takes place offline.	Learning method takes place in real time.

Number of Classes	Number of classes is known.	Number of classes is not known.
Main Drawback	Classifying big data can be a real challenge in Supervised Learning.	You cannot get precise information regarding data sorting, and the output as data used in unsupervised learning is labeled and not known.

2. b. What are the Issues regarding Classification and Prediction? Explain. [6M]

1. Preparing the Data for Classification and Prediction:

- The following preprocessing steps may be applied to the data to help improve the accuracy, efficiency, and scalability of the classification or prediction process.

(i) Data cleaning:

- This refers to the preprocessing of data in order to remove or reduce noise (by applying smoothing techniques) and the treatment of missing values (e.g., by replacing a missing value with the most commonly occurring value for that attribute).
- Although most classification algorithms have some mechanisms for handling noisy or missing data, this step can help reduce confusion during learning.

(ii) Relevance analysis:

- Many of the attributes in the data may be redundant. Correlation analysis can be used to identify whether any two given attributes are statistically related.
- For example, a strong correlation between attributes A1 and A2 would suggest that one of the two could be removed from further analysis.
- A database may also contain irrelevant attributes. Attribute subset selection can be used in these cases to find a reduced set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.
- Hence, relevance analysis, in the form of correlation analysis and attribute subset selection, can be used to detect attributes that do not contribute to the classification or prediction task.
- Such analysis can help improve classification efficiency and scalability.

(iii) Data Transformation and Reduction:

- The data may be transformed by normalization, particularly when neural networks or methods involving distance measurements are used in the learning step.
- Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1 to +1 or 0 to 1.
- The data can also be transformed by generalizing it to higher-level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous valued attributes.
- For example, numeric values for the attribute income can be generalized to discrete ranges, such as low, medium, and high.
- Similarly, categorical attributes, like street, can be generalized to higher-level concepts, like city. Data can also be reduced by applying many other methods, ranging from

wavelet transformation and principle components analysis to discretization techniques, such as binning, histogram analysis, and clustering.

2. Comparing Classification and Prediction Methods:

Accuracy:

- The accuracy of a classifier refers to the ability of a given classifier to correctly predict the class label of new or previously unseen data (i.e., tuples without class label information).
- The accuracy of a predictor refers to how well a given predictor can guess the value of the predicted attribute for new or previously unseen data.

Speed:

- This refers to the computational costs involved in generating and using the given classifier or predictor.

Robustness:

- This is the ability of the classifier or predictor to make correct predictions given noisy data or data with missing values.

Scalability:

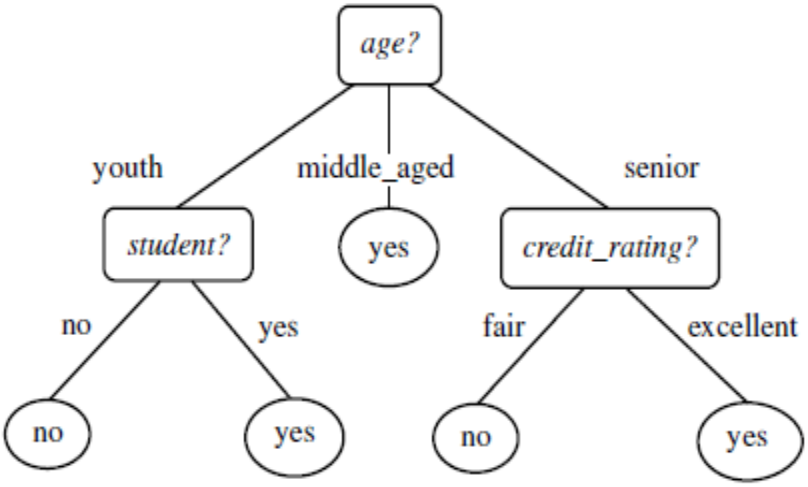
- This refers to the ability to construct the classifier or predictor efficiently given large amounts of data.

Interpretability:

- This refers to the level of understanding and insight that is provided by the classifier or predictor. Interpretability is subjective and therefore more difficult to assess.

3. a. Define Decision Tree. Why are decision tree classifiers so popular? [6M]

- A decision tree is a flowchart-like tree structure, where each internal node (non leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node.



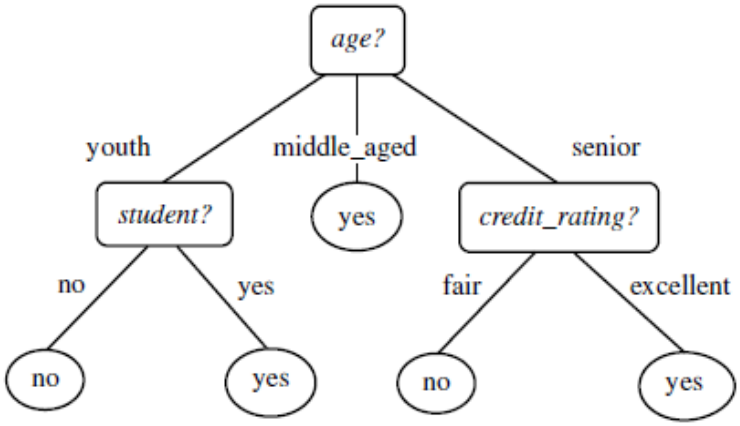
- It represents the concept buys computer, that is, it predicts whether a customer at *AllElectronics* is likely to purchase a computer.
- Internal nodes are denoted by rectangles, and leaf nodes are denoted by ovals. Some decision tree algorithms produce only *binary* trees (where each internal node branches to exactly two other nodes), whereas others can produce non binary trees.

Why are decision tree classifiers so popular?

- The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.
- Decision trees can handle high dimensional data. Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
- The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy.
- Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology. Decision trees are the basis of several commercial rule induction systems.

3. b. Outline the concept of Classification by Decision Tree Induction. [6M]

- Decision tree induction is the learning of decision trees from class-labeled training tuples.
- A decision tree is a flowchart-like tree structure, where
  - Each internal node denotes a test on an attribute.
  - Each branch represents an outcome of the test.
  - Each leaf node holds a class label.
  - The topmost node in a tree is the root node.



- The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore appropriate for exploratory knowledge discovery.
- Decision trees can handle high dimensional data. Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
- The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy.
- Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology.

Algorithm for Decision Tree Induction:

Algorithm: Generate decision tree. Generate a decision tree from the training tuples of data partition *D*.

Input:

- **Data partition**, *D*, which is a set of training tuples and their associated class labels;
- ***attribute list***, the set of candidate attributes;

- ***Attribute selection method***, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting attribute* and, possibly, either a *split point* or *splitting subset*.

Output: A decision tree.

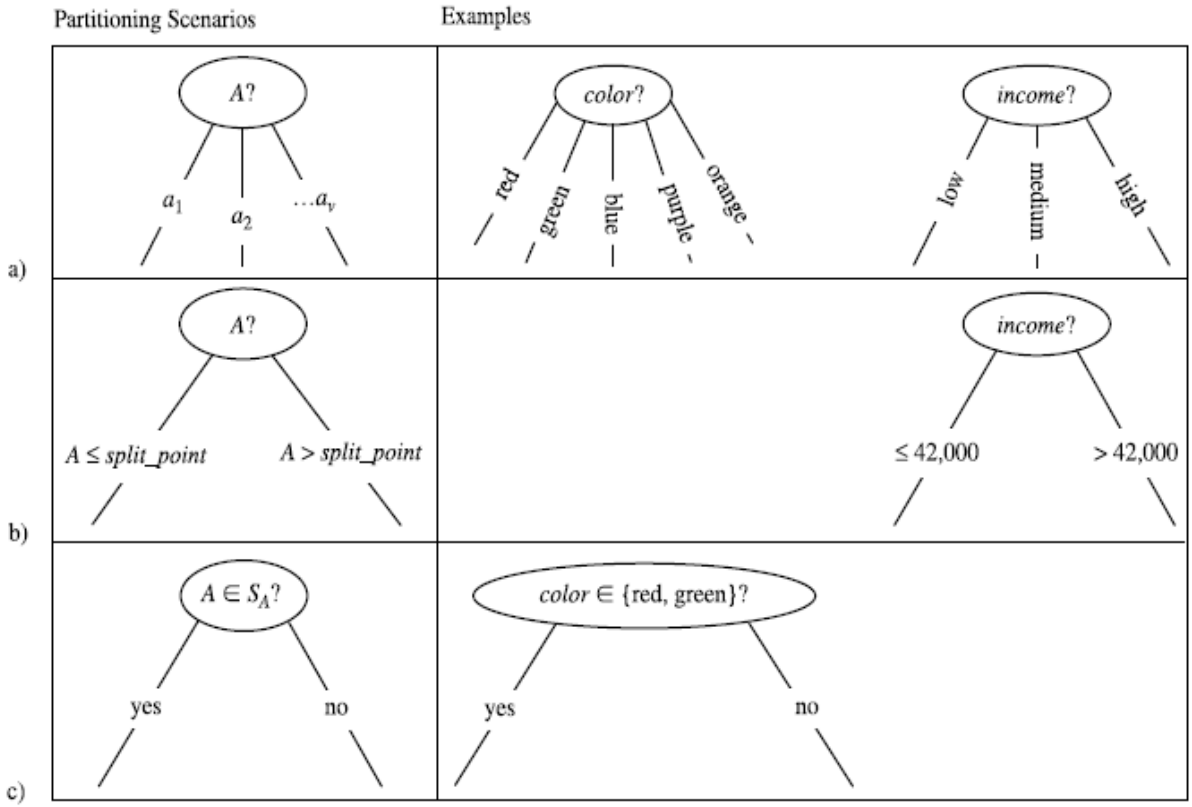
Method:

(1) create a node N;
(2) if tuples in D are all of the same class, C then
(3) return N as a leaf node labeled with the class C;
(4) if attribute_list is empty then
(5) return N as a leaf node labeled with the majority class in D;
(6) apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7) label node N with splitting_criterion;
(8) if splitting_attribute is discrete-valued and
multiway splits allowed then
(9) attribute_list ←attribute_list - splitting attribute;
(10) for each outcome j of splitting_criterion
(11) let D_j be the set of data tuples in D satisfying outcome j;
(12) if D_j is empty then
(13) attach a leaf labeled with the majority class in D to node N;
(14) else attach the node returned by Generate decision tree(Dj, attribute list) to node N;
endfor
(15) return *N*;

- The algorithm is called with three parameters:
  - Data partition
  - Attribute list
  - Attribute selection method
- The parameter attribute list is a list of attributes describing the tuples.
- Attribute selection method specifies a heuristic procedure for selecting the attribute that ″best″ discriminates the given tuples according to class.
- The tree starts as a single node, N, representing the training tuples in D.
- If the tuples in D are all of the same class, then node N becomes a leaf and is labeled with that class.
- All of the terminating conditions are explained at the end of the algorithm. Otherwise, the algorithm calls Attribute selection method to determine the splitting criterion.
- The splitting criterion tells us which attribute to test at node N by determining the ″best″ way to separate or partition the tuples in D into individual classes.
- There are three possible scenarios. Let A be the splitting attribute. A has v distinct values, $\{a_1, a_2, \ldots, a_v\}$, based on the training data.
  - **A is discrete-valued:** In this case, the outcomes of the test at node N correspond directly to the known values of A. A branch is created for each

known value, a_j, of A and labeled with that value. A need not be considered in any future partitioning of the tuples.

- **A is continuous-valued:** In this case, the test at node N has two possible outcomes, corresponding to the conditions A <=split point and A >split point, respectively where split point is the split-point returned by Attribute selection method as part of the splitting criterion.
- **A is discrete-valued and a binary tree must be produced:** The test at node N is of the form "A∈SA?". SA is the splitting subset for A, returned by Attribute selection method as part of the splitting criterion. It is a subset of the known values of A.



4. Discuss the following terms **[12M]**
i) Gini Index
ii) Gain ratio
iii) Information Gain

i) Gini Index:

- The Gini index is used in CART. The Gini index measures the impurity of *D*, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

where pi is the probability that a tuple in D belongs to class Ci.

- The Gini index considers a binary split for each attribute. Let's first consider the case where A is a discrete-valued attribute having v distinct values, $\{a_1, a_2, \ldots, a_v\}$, occurring in D.
- To determine the best binary split on A, we examine all of the possible subsets that can be formed using known values of A. Each subset, SA, can be considered as a binary test for attribute A of the form "$A \in S_A$?".
- Given a tuple, this test is satisfied if the value of A for the tuple is among the values listed in S_A.
- If A has v possible values, then there are 2^v possible subsets. For example, if income has three possible values, namely {low, medium, high}, then the possible subsets are {low, medium, high}, {low, medium}, {low, high}, {medium, high}, {low}, {medium}, {high}, and {}.
- For example, if a binary split on *A* partitions *D* into D_1 and D_2, the gini index of *D* given that partitioning is

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2).$$

- D_1 is the set of tuples in *D* satisfying $A \leq$ *split point*, and D_2 is the set of tuples in *D* satisfying $A >$ *split point*.
- The reduction in impurity that would be incurred by a binary split on a discrete- or continuous-valued attribute *A* is

$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

ii) Gain ratio:

- The information gain measure is biased toward tests with many outcomes. That is, it prefers to select attributes having a large number of values.
- It applies a kind of normalization to information gain using a "split information" value defined analogously with *Info*(*D*) as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

- This value represents the potential information generated by splitting the training data set, *D*, into *v* partitions, corresponding to the *v* outcomes of a test on attribute *A*.

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}.$$

iii) Information Gain:

- Let node N represent or hold the tuples of partition D. The attribute with the highest information gain is chosen as the splitting attribute for node *N*.
- This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or "impurity" in these partitions.
- The expected information needed to classify a tuple in *D* is given by

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

where p_i is the probability that an arbitrary tuple in *D* belongs to class C_i.

- *Info(D)* is just the average amount of information needed to identify the class label of a tuple in *D*. Note that, at this point, the information we have is based solely on the proportions of tuples of each class. *Info(D)* is also known as the entropy of *D*.

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

- The term |*Dj* | / |Dj| acts as the weight of the *j*th partition. *InfoA*(*D*) is the expected information required to classify a tuple from *D* based on the partitioning by *A*.
- Information gain is defined as the difference between the original information requirement and the new requirement.
- That is,

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

5. a. Define Bayes theorem. **[6M]**

- Let X be a data tuple. In Bayesian terms, X is considered ″evidence″ and it is described by measurements made on a set of n attributes.
- Let H be some hypothesis, such as that the data tuple X belongs to a specified class C. For classification problems, we want to determine P(H|X), the probability that the hypothesis H holds given the ″evidence″ or observed data tuple X.
- P(H|X) is the posterior probability, or a posteriori probability, of H conditioned on X.
- Bayes' theorem is useful in that it provides a way of calculating the posterior probability, P(H|X), from P(H), P(X|H), and P(X).

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}.$$

5. b. Explain the Naive Bayesian Classification with an example. [6M]

- The naive Bayesian classifier, or simple Bayesian classifier, works as follows:

1. Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X = (x_1, x_2, …,x_n), depicting n measurements made on the tuple from n attributes, respectively, A_1, A_2, …, A_n.

2. Suppose that there are m classes, C_1, C_2, …, C_m. Given a tuple, X, the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X.
That is, the naive Bayesian classifier predicts that tuple X belongs to the class C_i if and only if

$$P(C_i|X) > P(C_j|X) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

Thus we maximize $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is called the maximum posteriori hypothesis. By Bayes' theorem

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}.$$

3. As P(X) is constant for all classes, only $P(X|C_i)P(C_i)$ need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, $P(C_1) = P(C_2) = \ldots = P(C_m)$, and we would therefore maximize $P(X|C_i)$. Otherwise, we maximize $P(X|C_i)P(C_i)$.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute $P(X|C_i)$. In order to reduce computation in evaluating $P(X|C_i)$, the naive assumption of class conditional independence is made.
This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple.
Thus,

$$\begin{aligned} P(X|C_i) &= \prod_{k=1}^{n} P(x_k|C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i). \end{aligned}$$

Example:

Predicting a class label using naive Bayesian classification. We wish to predict the class label of a tuple using naive Bayesian classification, The data tuples are described by the attributes *age, income, student,* and *credit rating*.

Class-labeled training tuples from the *AllElectronics* customer database.

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

The class label attribute, *buys computer*, has two distinct values (namely, {*yes, no*}). Let *C*1 correspond to the class *buys computer* = *yes* and *C*2 correspond to *buys computer* = *no*. The tuple we wish to classify is

X = (*age* = *youth, income* = *medium, student* = *yes, credit rating* = *fair*)

We need to maximize *P*(*X*|*Ci*)*P*(*Ci*), for *i* = 1, 2. *P*(*Ci*), the prior probability of each class, can be computed based on the training tuples:

P(buys computer = yes) = 9/14 = 0.643

P(buys computer = no) = 5/14 = 0.357

To compute P(**X**|Ci), for i = 1, 2, we compute the following conditional probabilities:

P(age = youth | buys computer = yes) = 2/9 = 0.222

P(age = youth | buys computer = no) = 3/5 = 0.600

P(income = medium | buys computer = yes) = 4/9 = 0.444

P(income = medium | buys computer = no) = 2/5 = 0.400

P(student = yes | buys computer = yes) = 6/9 = 0.667

P(student = yes | buys computer = no) = 1/5 = 0.200

P(credit rating = fair | buys computer = yes) = 6/9 = 0.667

P(credit rating = fair | buys computer = no) = 2/5 = 0.400

Using the above probabilities, we obtain

P(**X** | buys computer = yes) = P(age = youth | buys computer = yes) X

P(income = medium | buys computer = yes) X

P(student = yes | buys computer = yes) X

P(credit rating = fair | buys computer = yes)

= 0.222 X 0.444 X 0.667 X 0.667 = 0.044.

Similarly,

P(**X** | buys computer = no) = 0.600 X 0.400 X 0.200 X 0.400 = 0.019

To find the class, Ci, that maximizes P(**X**|Ci)P(Ci), we compute

P(**X**|buys computer = yes)P(buys computer = yes) = 0.044 X 0.643 = 0.028

P(**X**|buys computer = no)P(buys computer = no) = 0.019 X 0.357 = 0.007

Therefore, the naive Bayesian classifier predicts buys computer = yes for tuple **X**.

6. Summarize about attribute selection measures. **[12M]**

- An attribute selection measure is a heuristic for selecting the splitting criterion that "best" separates a given data partition, *D*, of class-labeled training tuples into individual classes.
- Attribute selection measures are also known as splitting rules.
- Three popular attribute selection measures
  - Information gain,
  - gain ratio, and
  - gini index.

i) Information Gain:

- Let node N represent or hold the tuples of partition D. The attribute with the highest information gain is chosen as the splitting attribute for node *N*.
- This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or "impurity" in these partitions.
- The expected information needed to classify a tuple in *D* is given by

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

where p_i is the probability that an arbitrary tuple in *D* belongs to class C_i.

- *Info(D)* is just the average amount of information needed to identify the class label of a tuple in *D*. Note that, at this point, the information we have is based solely on the proportions of tuples of each class. *Info(D)* is also known as the entropy of *D*.

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

- The term |*Dj* | / |*D*j| acts as the weight of the *j*th partition. *InfoA*(*D*) is the expected information required to classify a tuple from *D* based on the partitioning by *A*.
- Information gain is defined as the difference between the original information requirement and the new requirement.
- That is,

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

ii) Gain ratio:

- The information gain measure is biased toward tests with many outcomes. That is, it prefers to select attributes having a large number of values.
- It applies a kind of normalization to information gain using a "split information" value defined analogously with *Info*(*D*) as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

- This value represents the potential information generated by splitting the training data set, *D*, into *v* partitions, corresponding to the *v* outcomes of a test on attribute *A*.

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}.$$

iii) Gini Index:

- The Gini index is used in CART. The Gini index measures the impurity of *D*, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

where pi is the probability that a tuple in D belongs to class Ci.

- The Gini index considers a binary split for each attribute. Let's first consider the case where A is a discrete-valued attribute having v distinct values, $\{a_1, a_2, \ldots, a_v\}$, occurring in D.
- To determine the best binary split on A, we examine all of the possible subsets that can be formed using known values of A. Each subset, SA, can be considered as a binary test for attribute A of the form "$A \in S_A$?".
- Given a tuple, this test is satisfied if the value of A for the tuple is among the values

listed in S_A.

- If A has v possible values, then there are 2^v possible subsets. For example, if income has three possible values, namely {low, medium, high}, then the possible subsets are {low, medium, high}, {low, medium}, {low, high}, {medium, high}, {low}, {medium}, {high}, and {}.
- For example, if a binary split on *A* partitions *D* into D_1 and D_2, the gini index of *D* given that partitioning is

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2).$$

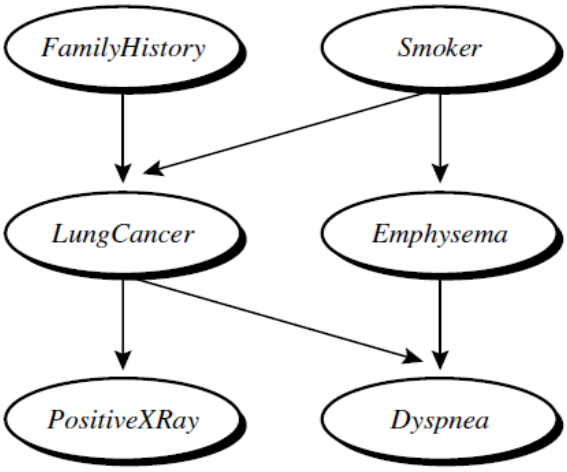
- D_1 is the set of tuples in *D* satisfying $A \leq$ *split point*, and D_2 is the set of tuples in *D* satisfying $A >$ *split point*.
- The reduction in impurity that would be incurred by a binary split on a discrete- or continuous-valued attribute *A* is

$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

7. Explain about Bayesian belief networks with an example. [12M]

- Bayesian belief networks specify joint conditional probability distributions. They allow class conditional independencies to be defined between subsets of variables.
- They provide a graphical model of causal relationships, on which learning can be performed.
- Trained Bayesian belief networks can be used for classification. Bayesian belief networks are also known as belief networks, Bayesian networks, and probabilistic networks.
- A belief network is defined by two components — a **directed acyclic graph** and a set of **conditional probability table**.
- Each node in the directed acyclic graph represents a random variable. The variables may be discrete or continuous-valued.
- They may correspond to actual attributes given in the data or to "hidden variables" believed to form a relationship.
- Each arc represents a probabilistic dependence. If an arc is drawn from a node *Y* to a node *Z*, then *Y* is a parent or immediate predecessor of *Z*, and *Z* is a descendant of *Y*. Each variable is conditionally independent of its non descendants in the graph, given its parents.

(a)



(b)

	FH, S	*FH, ~S*	*~FH, S*	*~FH, ~S*
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

- A belief network has one conditional probability table (CPT) for each variable. The CPT for a variable *Y* specifies the conditional distribution *P*(*Y* | *Parents*(*Y*)), where *Parents*(*Y*) are the parents of *Y*.
- Figure shows a CPT for the variable LungCancer.
- The conditional probability for each known value of LungCancer is given for each possible combination of values of its parents. For instance, from the upper leftmost and bottom rightmost entries, respectively.

  P(LungCancer = yes | FamilyHistory = yes, Smoker = yes) = 0.8

  P(LungCancer = no | FamilyHistory = no, Smoker = no) = 0.9

8. a. Discuss about Rule based Classification method. [6M]

- A rule-based classifier uses a set of IF-THEN rules for classification. An IF-THEN rule is an expression of the form

  IF *condition* THEN *conclusion*.
- An example is rule *R*1,

  R1: IF *age* = *youth* AND *student* = *yes* THEN *buys computer* = *yes*.
- The "IF"-part (or left-hand side) of a rule is known as the rule antecedent or precondition. The "THEN"-part (or right-hand side) is the rule consequent. In the rule antecedent, the condition consists of one or more *attribute tests* (such as *age* = *youth*, and *student* = *yes*) that are logically ANDed. The rule's consequent contains a class prediction.
- *R*1 can also be written as

  R1: (*age* = *youth*) ^ (*student* = *yes*))(*buys computer* = *yes*).
- If the condition (that is, all of the attribute tests) in a rule antecedent holds true for a given tuple, we say that the rule antecedent is satisfied (or simply, that the rule is

satisfied) and that the rule covers the tuple.

- A rule *R* can be assessed by its coverage and accuracy. Given a tuple, ***X***, from a class labeled data set, *D*, let n_{covers} be the number of tuples covered by *R*; $n_{correct}$ be the number of tuples correctly classified by *R*; and |*D*| be the number of tuples in *D*.
- We can define the coverage and accuracy of *R* as

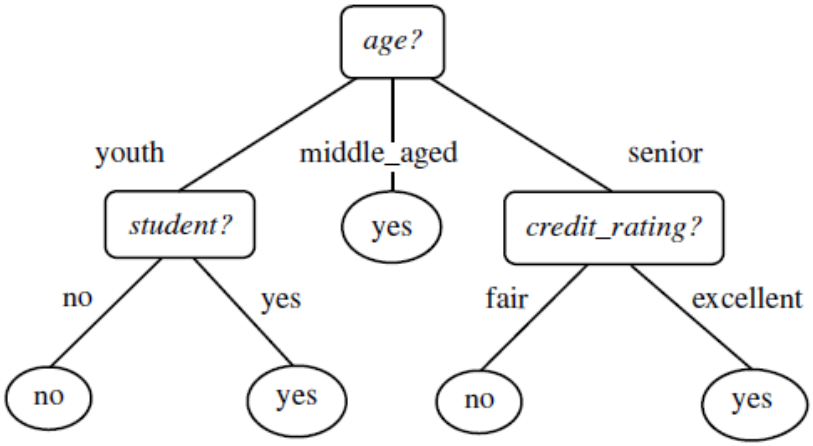
$$coverage(R) = \frac{n_{covers}}{|D|}$$

$$accuracy(R) = \frac{n_{correct}}{n_{covers}}.$$

Rule Extraction from a Decision Tree:

- To extract rules from a decision tree, one rule is created for each path from the root to a leaf node.
- Each splitting criterion along a given path is logically ANDed to form the rule antecedent ("IF" part). The leaf node holds the class prediction, forming the rule consequent ("THEN" part).

Example:



R1: IF age = youth AND student = no THEN buys computer = no

R2: IF age = youth AND student = yes THEN buys computer = yes

R3: IF age = middle aged THEN buys computer = yes

R4: IF age = senior AND credit rating = excellent THEN buys computer = yes

R5: IF age = senior AND credit rating = fair THEN buys computer = no

Rule Induction Using a Sequential Covering Algorithm:

- IF-THEN rules can be extracted directly from the training data using a sequential covering algorithm.
- The name comes from the notion that the rules are learned *sequentially* (one at a time), where each rule for a given class will ideally *cover* many of the tuples of that class.

Algorithm: Sequential covering. Learn a set of IF-THEN rules for classification.

Input:

- D, a data set class-labeled tuples;
- Att_vals, the set of all attributes and their possible values.

Output: A set of IF-THEN rules.

Method:

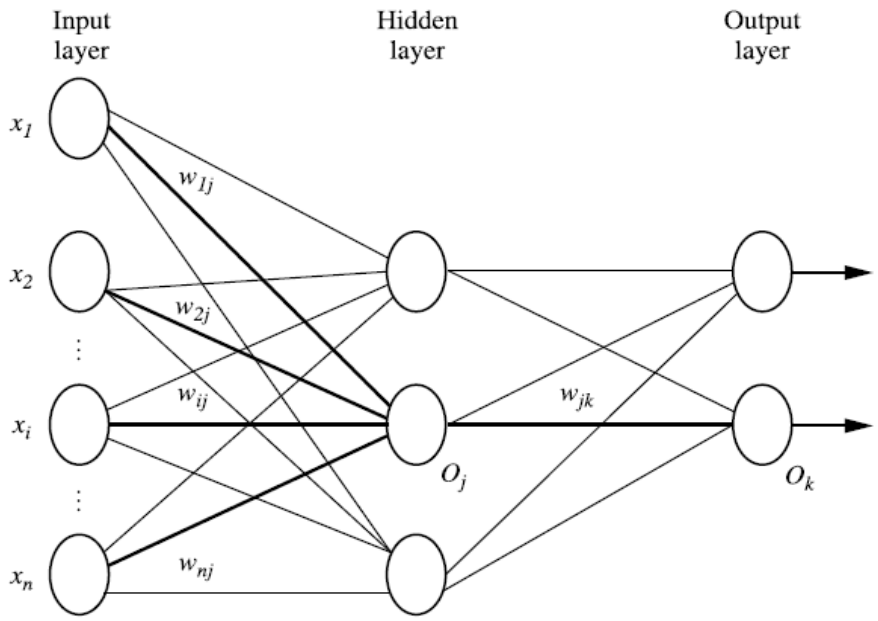
```
(1)  Rule_set = {}; // initial set of rules learned is empty
(2)  for each class c do
(3)      repeat
(4)          Rule = Learn_One_Rule(D, Att_vals, c);
(5)          remove tuples covered by Rule from D;
(6)      until terminating condition;
(7)      Rule_set = Rule_set + Rule; // add new rule to rule set
(8)  endfor
(9)  return Rule_Set;
```

8. b. Define Neural Network. Explain the Classification by Back Propagation. [6M]

- A neural network is a set of connected input/output units in which each connection has a weight associated with it.
- During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples.
- Neural network learning is also referred to as connectionist learning due to the connections between units.
- Back propagation is a neural network learning algorithm.

A Multilayer Feed-Forward Neural Network:

- The backpropagation algorithm performs learning on a multilayer feed-forward neural network. It iteratively learns a set of weights for prediction of the class label of tuples.
- A multilayer feed-forward neural network consists of an input layer, one or more hidden layers, and an output layer.



- Each layer is made up of units. The inputs to the network correspond to the attributes measured for each training tuple.
- The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer of "neuronlike" units, known as a hidden layer.
- The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary, although in practice, usually only one is used.
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction for given tuples.
- The units in the input layer are called input units. The units in the hidden layers and output layer are sometimes referred to as neurodes, due to their symbolic biological basis, or as output units.

Backpropagation:

- Before training can begin, the user must decide on the network topology by specifying the number of units in the input layer, the number of hidden layers (if more than one), the number of units in each hidden layer, and the number of units in the output layer.

Algorithm: Backpropagation. Neural network learning for classification or prediction, using the backpropagation algorithm.

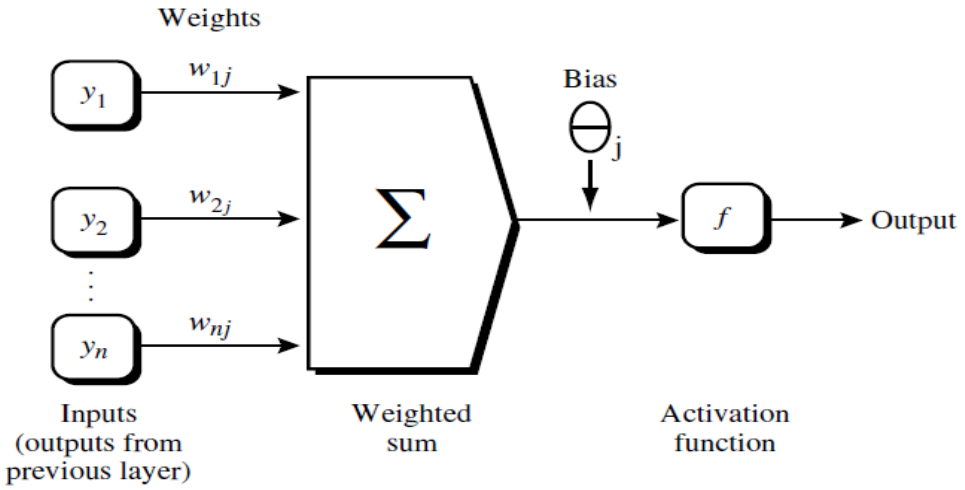
Input:

- D, a data set consisting of the training tuples and their associated target values;
- l, the learning rate;
- *network*, a multilayer feed-forward network.

Output: A trained neural network.

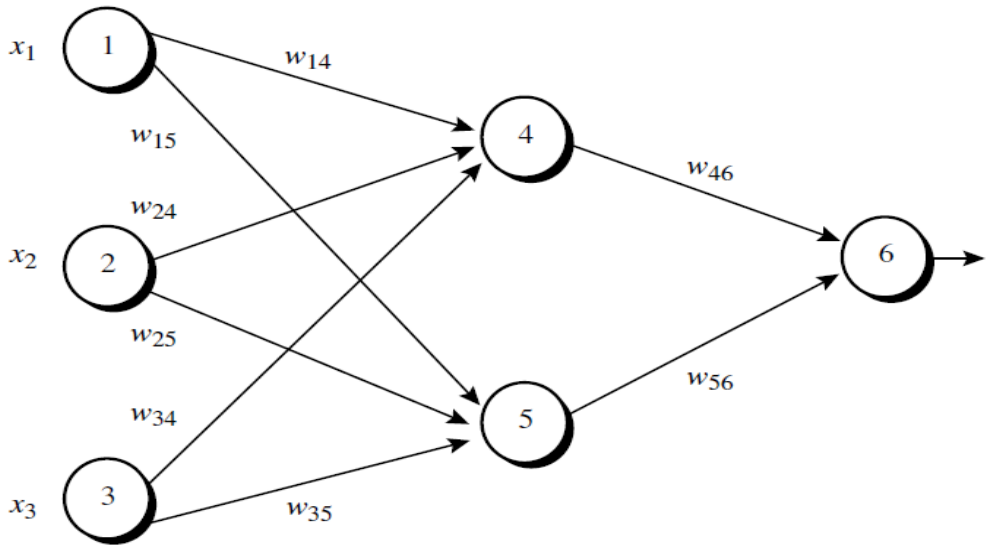
Method:

(1) Initialize all weights and biases in *network*;
(2) **while** terminating condition is not satisfied {
(3) **for** each training tuple $\boldsymbol{X}$ in D {
(4) // Propagate the inputs forward:
(5) **for** each input layer unit j {
(6) $O_j = I_j$; // output of an input unit is its actual input value
(7) **for** each hidden or output layer unit j {
(8) $I_j = \sum_i w_{ij} O_i + \theta_j$; //compute the net input of unit j with respect to the previous layer, i
(9) $O_j = \frac{1}{1+e^{-I_j}}$; } // compute the output of each unit j
(10) // Backpropagate the errors:
(11) **for** each unit j in the output layer
(12) $Err_j = O_j(1-O_j)(T_j - O_j)$; // compute the error
(13) **for** each unit j in the hidden layers, from the last to the first hidden layer
(14) $Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$; // compute the error with respect to the next higher layer, k
(15) **for** each weight w_{ij} in *network* {
(16) $\Delta w_{ij} = (l) Err_j O_i$; // weight increment
(17) $w_{ij} = w_{ij} + \Delta w_{ij}$; } // weight update
(18) **for** each bias θ_j in *network* {
(19) $\Delta\theta_j = (l) Err_j$; // bias increment
(20) $\theta_j = \theta_j + \Delta\theta_j$; } // bias update
(21) } }



Example:

Sample calculations for learning by the back propagation algorithm. Figure 6.18 shows a multilayer feed-forward neural network. Let the learning rate be 0.9. The initial weight and bias values of the network are given in Table 6.3, along with the first training tuple, $\boldsymbol{X}$ = (1, 0, 1), whose class label is 1. This example shows the calculations for back propagation, given the first training tuple, $\boldsymbol{X}$. The tuple is fed into the network, and the net input and output of each unit are computed. These values are shown in Table 6.4. The error of each unit is computed



Initial input, weight, and bias values.

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
1	0	1	0.2	−0.3	0.4	0.1	−0.5	0.2	−0.3	−0.2	−0.4	0.2	0.1

The net input and output calculations.

Unit j	*Net input, I_j*	*Output, O_j*
4	$0.2+0-0.5-0.4=-0.7$	$1/(1+e^{0.7})=0.332$
5	$-0.3+0+0.2+0.2=0.1$	$1/(1+e^{-0.1})=0.525$
6	$(-0.3)(0.332)-(0.2)(0.525)+0.1=-0.105$	$1/(1+e^{0.105})=0.474$

Calculation of the error at each node.

Unit j	*Err$_j$*
6	$(0.474)(1-0.474)(1-0.474)=0.1311$
5	$(0.525)(1-0.525)(0.1311)(-0.2)=-0.0065$
4	$(0.332)(1-0.332)(0.1311)(-0.3)=-0.0087$

Calculations for weight and bias updating.

Weight or bias	*New value*
w_{46}	$-0.3 + (0.9)(0.1311)(0.332) = -0.261$
w_{56}	$-0.2 + (0.9)(0.1311)(0.525) = -0.138$
w_{14}	$0.2 + (0.9)(-0.0087)(1) = 0.192$
w_{15}	$-0.3 + (0.9)(-0.0065)(1) = -0.306$
w_{24}	$0.4 + (0.9)(-0.0087)(0) = 0.4$
w_{25}	$0.1 + (0.9)(-0.0065)(0) = 0.1$
w_{34}	$-0.5 + (0.9)(-0.0087)(1) = -0.508$
w_{35}	$0.2 + (0.9)(-0.0065)(1) = 0.194$
θ_6	$0.1 + (0.9)(0.1311) = 0.218$
θ_5	$0.2 + (0.9)(-0.0065) = 0.194$
θ_4	$-0.4 + (0.9)(-0.0087) = -0.408$

9. What is prediction? Explain about Linear regression method. [12M]

- Numeric prediction is the task of predicting continuous (or ordered) values for given input.
- The most widely used approach for numeric prediction is regression, a statistical methodology that was developed by Sir Frances Galton (1822– 1911).
- Regression analysis can be used to model the relationship between one or more independent or predictor variables and a dependent or response variable
- The **predictor variables** are the attributes of interest describing the tuple. The values of the predictor variables are known.
- The **response variable** is what we want to predict

Linear Regression:

- **Straight-line regression analysis** involves a response variable, **y,** and a single predictor variable, **x.** It is the simplest form of regression, and models y as a linear function of x.
- That is,

$$y = b + wx;$$

where the variance of **y** is assumed to be constant, and **b** and **w** are regression coefficients.

- The regression coefficients, w and b, can also be thought of as weights, so that we can equivalently write,

$$y = w_0 + w_1 x$$

- The training set contains |D| data points of the form $(x_1, y_1), (x_2, y_2), \ldots, (x_{|D|}, y_{|D|})$.

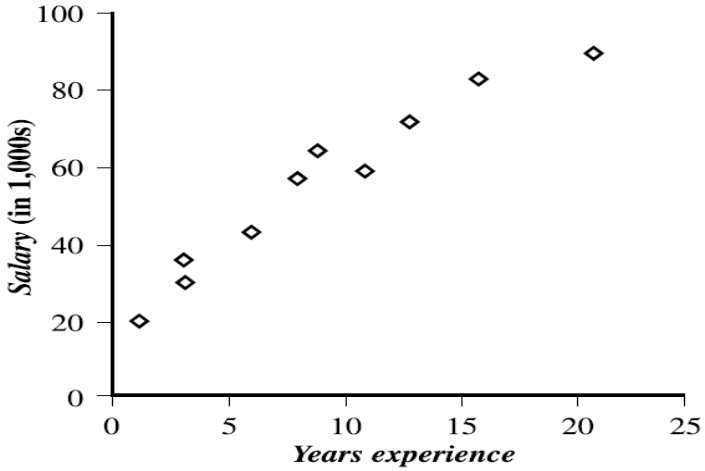
- The regression coefficients can be estimated using this method with the following equations:

$$w_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2}$$

$$w_0 = \bar{y} - w_1\bar{x}$$

Salary data.

x years experience	*y salary (in $1000s)*
3	30
8	57
9	64
13	72
3	36
6	43
11	59
21	90
1	20
16	83



Multiple linear regression:

- **Multiple linear regression** is an extension of straight-line regression so as to involve more than one predictor variable.
- It allows response variable y to be modeled as a linear function of, say, n predictor variables or attributes, A_1, A_2, : : : , A_n, describing a tuple, **X.**
- An example of a multiple linear regression model based on two predictor attributes or

variables, A_1 and A_2, is

$$y = w_0 + w_1 x_1 + w_2 x_2$$

where x_1 and x_2 are the values of attributes A_1 and A_2, respectively, in **X.**

Nonlinear Regression:

- Polynomial regression is, when there is just one predictor variable.
- It can be modeled by adding polynomial terms to the basic linear model.
- By applying transformations to the variables, we can convert the nonlinear model into a linear one that can then be solved by the method of least squares.
- Transformation of a polynomial regression model to a linear regression model.
- Consider a cubic polynomial relationship given by

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

- To convert this equation to linear form, we define new variables:

$$x_1 = x; \quad x_2 = x^2; \quad x_3 = x^3$$

- Equation (6.53) can then be converted to linear form by applying the above assignments, resulting in the equation

$$y = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$$

10. a. Discuss about Accuracy and Error measures. [6M]

Classifier Accuracy Measures:

- The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.
- The error rate or misclassification rate of a classifier, M, which is simply 1 - Acc(M), where Acc(M) is the accuracy of M.
- If we were to use the training set to estimate the error rate of a model, this quantity is known as the resubstitution error**.**
- The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes.
- Given m classes, a confusion matrix is a table of at least size m by m.
- Given two classes, we can talk in terms of positive tuples (buys computer = yes) versus negative tuples (buys computer = no).
- **True positives** refer to the positive tuples that were correctly labeled by the classifier, while **true negatives** are the negative tuples that were correctly labeled by the classifier.
- **False positives** are the negative tuples that were incorrectly labeled (e.g., tuples of class buys computer = no for which the classifier predicted buys computer = yes).

- Similarly, **false negatives** are the positive tuples that were incorrectly labeled (e.g., tuples of class buys computer = yes for which the classifier predicted buys computer = no).

		Predicted class	
		C_1	C_2
Actual class	C_1	true positives	false negatives
	C_2	false positives	true negatives

- The **sensitivity and specificity** measures can be used.
- **Sensitivity** is also referred to as the **true positive** rate (that is, the proportion of positive tuples that are correctly identified).
- **Specificity** is the **true negative** rate (that is, the proportion of negative tuples that are correctly identified).
- We may use **precision** to access the percentage of tuples labeled as "cancer" that actually are "cancer" tuples.
- These measures are defined as

$$sensitivity = \frac{t\_pos}{pos}$$

$$specificity = \frac{t\_neg}{neg}$$

$$precision = \frac{t\_pos}{(t\_pos + f\_pos)}$$

- It can be shown that accuracy is a function of sensitivity and specificity:

$$accuracy = sensitivity\frac{pos}{(pos+neg)} + specificity\frac{neg}{(pos+neg)}.$$

Predictor Error Measures:

- Let D^T be a test set of the form $(\mathbf{X_1, y_1}), (\mathbf{X_2, y_2}), \ldots, (\mathbf{X_d, y_d})$, where the Xi are the n-dimensional test tuples with associated known values, y_i, for a response variable, y, and d is the number of tuples in D^T .
- Since predictors return a continuous value rather than a categorical label, it is difficult to say exactly whether the predicted value, y`i, for $\mathbf{X_i}$ **is correct.**
- Loss functions measure the error between yi and the predicted value, $y`_i$.
- The most common loss functions are:

$$\textbf{Absolute error}: \quad |y_i - y_i'|$$

$$\textbf{Squared error}: \quad (y_i - y_i')^2$$

- The test error (rate), or generalization error, is the average loss over the test set.
- Thus, we get the following error rates.

$$\text{Mean absolute error :} \quad \frac{\sum_{i=1}^{d} |y_i - y_i'|}{d}$$

$$\text{Mean squared error :} \quad \frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$$

- Relative measures of error include:

$$\text{Relative absolute error :} \quad \frac{\sum_{i=1}^{d} |y_i| - y_i'|}{\sum_{i=1}^{d} |y_i - \bar{y}|}$$

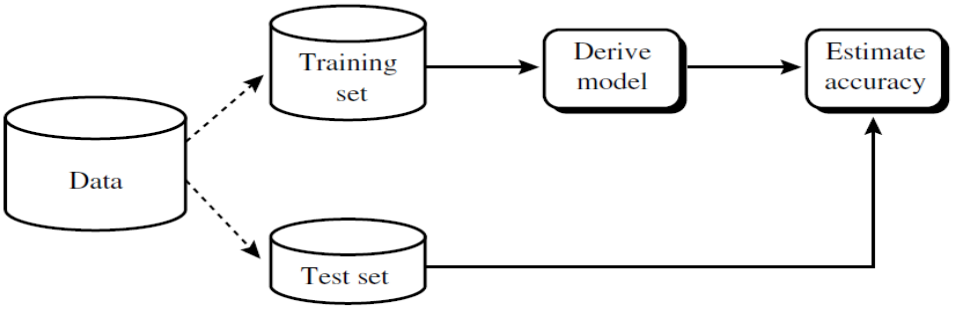
$$\text{Relative squared error :} \quad \frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$$

10. b. Evaluating the accuracy of a classifier in data mining. [6M]

- Holdout, random sub sampling, cross validation, and the bootstrap are common techniques for assessing accuracy based on randomly sampled partitions of the given data.
- The use of such techniques to estimate accuracy increases the overall computation time, yet is useful for model selection.

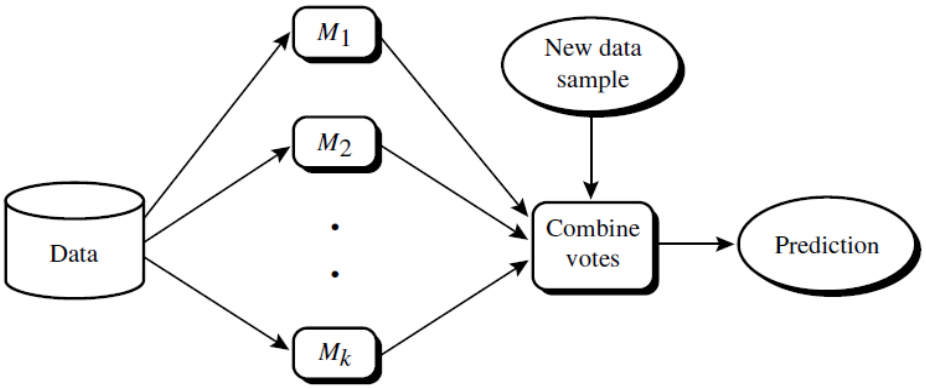
Holdout Method and Random Sub sampling:

- In this method, the given data are randomly partitioned into two independent sets, a training set and a test set.
- Typically, two-thirds of the data are allocated to the training set, and the remaining one-third is allocated to the test set.
- The training set is used to derive the model, whose accuracy is estimated with the test set.
- The estimate is pessimistic because only a portion of the initial data is used to derive the model.
- Random sub sampling is a variation of the holdout method in which the holdout method is repeated k times.
- The overall accuracy estimate is taken as the average of the accuracies obtained from each iteration.



Cross-validation:

- In **k-fold cross-validation,** the initial data are randomly partitioned into k mutually exclusive subsets or "folds," D_1, D_2, : : : , D_k, each of approximately equal size.
- Training and testing is performed k times. In iteration i, partition D_i is reserved as the test set, and the remaining partitions are collectively used to train the model.
- That is, in the first iteration, subsets D_2, … , D_k collectively serve as the training set in order to obtain a first model, which is tested on D_1; the second iteration is trained on subsets D_1, D_3, … , D_k and tested on D_2; and so on.
- The bootstrap method samples the given training tuples uniformly with replacement.
- That is, each time a tuple is selected, it is equally likely to be selected again and readded to the training set.



UNIT – V
CLUSTER ANALYSIS

1. a. Define Clustering. List basic requirements of cluster analysis. [6M]

Cluster Analysis:

- The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering.
- A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters.
- A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression.
- Clustering is also called data segmentation in some applications because clustering partitions large data sets into groups according to their similarity.

Typical Requirements of Clustering In Data Mining:

Scalability:

- Many clustering algorithms work well on small data sets containing fewer than several hundred data objects; however, a large database may contain millions of objects. Clustering on a sample of a given large data set may lead to biased results.
- Highly scalable clustering algorithms are needed.

Ability to deal with different types of attributes:

- Many algorithms are designed to cluster interval-based (numerical) data. However, applications may require clustering other types of data, such as binary, categorical (nominal), and ordinal data, or mixtures of these data types.

Discovery of clusters with arbitrary shape:

- Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures.
- Algorithms based on such distance measures tend to find spherical clusters with similar size and density.
- However, a cluster could be of any shape. It is important to develop algorithms that can detect clusters of arbitrary shape.

Minimal requirements for domain knowledge to determine input parameters:

- Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters).

- The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects.
- This not only burdens users, but it also makes the quality of clustering difficult to control.

Ability to deal with noisy data:

- Most real-world databases contain outliers or missing, unknown, or erroneous data. Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

Incremental clustering and insensitivity to the order of input records:

- Some clustering algorithms cannot incorporate newly inserted data (i.e., database updates) into existing clustering structures and, instead, must determine a new clustering from scratch. Some clustering algorithms are sensitive to the order of input data.
- It is important to develop incremental clustering algorithms and algorithms that are insensitive to the order of input.

High dimensionality:

- A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions.
- Finding clusters of data objects in high dimensional space is challenging, especially considering that such data can be sparse and highly skewed.

Constraint-based clustering:

- Real-world applications may need to perform clustering under various kinds of constraints.
- A challenging task is to find groups of data with good clustering behavior that satisfy specified constraints.

Interpretability and usability:

- Users expect clustering results to be interpretable, comprehensible, and usable. That is, clustering may need to be tied to specific semantic interpretations and applications.
- It is important to study how an application goal may influence the selection of clustering features and methods.

1. b. Describe the working of PAM algorithm. [6M]

- PAM (Partitioning Around Medoids) was one of the first *k*-medoids algorithms introduced.

- It attempts to determine *k* partitions for *n* objects. After an initial random selection of *k* representative objects, the algorithm repeatedly tries to make a better choice of cluster representatives.
- All of the possible pairs of objects are analyzed, where one object in each pair is considered a representative object and the other is not.
- The quality of the resulting clustering is calculated for each such combination. An object, ***oj***, is replaced with the object causing the greatest reduction in error.
- The set of best objects for each cluster in one iteration forms the representative objects for the next iteration.
- The final set of representative objects are the respective medoids of the clusters.

Algorithm: *k*-**medoids.** PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects in *D* as the initial representative objects or seeds;
(2) **repeat**
(3) assign each remaining object to the cluster with the nearest representative object;
(4) randomly select a nonrepresentative object, o_{random};
(5) compute the total cost, *S*, of swapping representative object, o_j, with o_{random};
(6) **if** $S < 0$ **then** swap o_j with o_{random} to form the new set of *k* representative objects;
(7) **until** no change;

2. a. Explain the various types of data in Cluster analysis. [6M]

- Main memory-based clustering algorithms typically operate on either of the following two data structures.

Data matrix (or object-by-variable structure):

- This represents n objects, such as persons, with p variables (also called measurements or attributes), such as age, height, weight, gender, and so on.
- The structure is in the form of a **relational table**, or **n-by-p matrix** (n objects p variables):

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

Dissimilarity matrix (or object-by-object structure):

- This stores a collection of proximities that are available for all pairs of n objects.
- It is often represented by an **n-by-n table**:

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

- The rows and columns of the data matrix represent **different entities**, while those of the dissimilarity matrix represent the **same entity**.
- Thus, the data matrix is often called a **two-mode matrix**, whereas the dissimilarity matrix is called a **one-mode matrix**.

Interval-Scaled Variables:

- These measures include the Euclidean, Manhattan, and Minkowski distances.
- Given measurements for a variable f, this can be performed as follows.

  1. Calculate the mean absolute deviation, s_f :

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \cdots + |x_{nf} - m_f|),$$

  where x_{1f} , … , x_{nf} are n measurements of f , and mf is the mean value of f

  2. Calculate the standardized measurement, or z-score:

$$z_{if} = \frac{x_{if} - m_f}{s_f}.$$

- The most popular distance measure is **Euclidean distance**, which is defined as

$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2},$$

where $i=(x_{i1}, x_{i2}, \ldots, x_{in})$ and $j=(x_{j1}, x_{j2}, \ldots, x_{jn})$ are two n-dimensional data objects.

- Another well-known metric is Manhattan (or city block) distance, defined as

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}|.$$

- Both the Euclidean distance and Manhattan distance satisfy the following mathematic requirements of a distance function:
  1. **d(i, j) ≥ 0:** Distance is a nonnegative number.
  2. **d(i, i) = 0:** The distance of an object to itself is 0.
  3. **d(i, j) = d(j, i):** Distance is a symmetric function.
  4. **d(i, j) ≤ d(i, h)+d(h, j):** Going directly from object i to object j in space is no more than making a detour over any other object h (triangular inequality).
- Minkowski distance is a generalization of both Euclidean distance and Manhattan distance.
- It is defined as

$$d(i, j) = (|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{in} - x_{jn}|^p)^{1/p},$$

where p is a positive integer. Such a distance is also called Lp norm, in some literature.

- It represents the Manhattan distance when $p = 1$ (i.e., $L1$ norm) and Euclidean distance when $p = 2$ (i.e., $L2$ norm).
- If each variable is assigned a weight according to its perceived importance, the weighted Euclidean distance can be computed as

$$d(i, j) = \sqrt{w_1|x_{i1} - x_{j1}|^2 + w_2|x_{i2} - x_{j2}|^2 + \cdots + w_m|x_{in} - x_{jn}|^2}.$$

- Weighting can also be applied to the Manhattan and Minkowski distances.

Binary Variables:

- A binary variable has only **two states: 0 or 1**, where 0 means that the variable is absent, and 1 means that it is present.

- A binary variable is **symmetric** if both of its states are equally valuable and carry the same weight; that is, there is no preference on which outcome should be coded as 0 or 1.
- Dissimilarity that is based on symmetric binary variables is called **symmetric binary dissimilarity**.

$$d(i, j) = \frac{r+s}{q+r+s+t}.$$

A contingency table for binary variables.

		object j		
		1	0	sum
	1	q	r	$q+r$
object i	0	s	t	$s+t$
	sum	$q+s$	$r+t$	p

- A binary variable is asymmetric if the outcomes of the states are not equally important. The dissimilarity based on such variables is called asymmetric binary dissimilarity.

$$d(i, j) = \frac{r+s}{q+r+s}.$$

$$sim(i, j) = \frac{q}{q+r+s} = 1 - d(i, j).$$

- The coefficient sim (i, j) is called the Jaccard coefficient

Categorical, Ordinal, and Ratio-Scaled Variables

Categorical Variables:

- A categorical variable is a generalization of the binary variable in that it can take on more than two states.
- Let the number of states of a categorical variable be *M.*
- The dissimilarity between two objects i and j can be computed based on the ratio of mismatches:

$$d(i, j) = \frac{p-m}{p},$$

where m is the number of matches, and p is the total number of variables.

Ordinal Variables

- A **discrete ordinal variable** resembles a categorical variable, except that the M states of the ordinal value are ordered in a **meaningful sequence.** For example, professional ranks, such as **assistant, associate, and professors.**
- A **continuous ordinal variable** looks like a set of continuous data of an unknown scale. For example, the relative ranking in a particular sport (e.g., **gold, silver, bronze**)
- The value of f for the i_{th} object is x_{if} , and f has Mf ordered states, representing the ranking 1, : : : , Mf .
- Replace each xif by its corresponding rank, $rif \in \{1, \dots , Mf\}$.
- Since each ordinal variable can have a different number of states, it is often necessary to map the range of each variable onto [0.0,1.0] so that each variable has equal weight. This can be achieved by replacing the rank r_{if} of the i_{th} object in the f th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}.$$

- Dissimilarity can then be computed using any of the distance measures.

Ratio-Scaled Variables:

- A ratio-scaled variable makes a positive measurement on a nonlinear scale, such as an exponential scale, approximately following the formula

  $\mathbf{Ae^{Bt}}$ **or** $\mathbf{Ae^{-Bt}}$

  where A and B are positive constants, and t typically represents time.

Variables of Mixed Types:

- Compute the dissimilarity between objects described by variables of the same type, where these types may be either interval-scaled, symmetric binary, asymmetric binary, categorical, ordinal, or ratio-scaled. However, in many real databases, objects are described by a mixture of variable types.

$$d(i, j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}},$$

2. b. Inference the working of k-means clustering. **[6M]**

Centroid-Based Technique:

- The K-Means Method: The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra cluster similarity is high but the inter cluster similarity is low.
- Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity. The k-means algorithm proceeds as follows.
- First, it randomly selects k of the objects, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster.
- This process iterates until the criterion function converges. Typically, the square-error criterion is used, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2,$$

Where E is the sum of the square error for all objects in the data set

p is the point in space representing a given object

m_i is the mean of cluster C_i

The k-means partitioning algorithm:

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster'sccenter is represented by the mean value of the objects in the cluster.

Input:

k: the number of clusters,

D: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

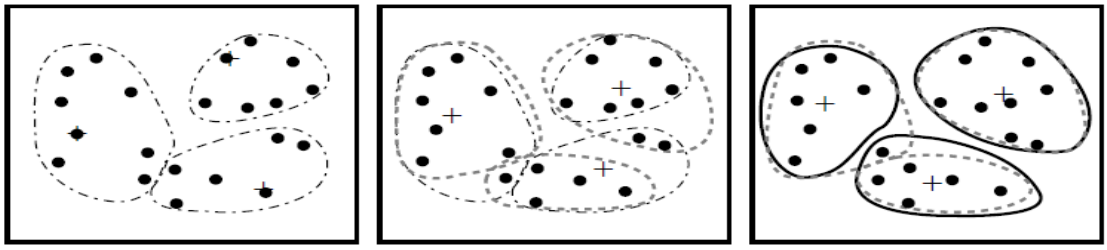
(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;

(2) repeat

(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;

(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;

(5) until no change;



(a) (b) (c)

Clustering of a set of objects based on the *k*-means method. (The mean of each cluster is marked by a "+".)

3. Explain K-Means and K-Medoids partitioning methods in detail. [12M]

Centroid-Based Technique:

- The K-Means Method: The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra cluster similarity is high but the inter cluster similarity is low.
- Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity. The k-means algorithm proceeds as follows.
- First, it randomly selects k of the objects, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster.
- This process iterates until the criterion function converges. Typically, the square-error criterion is used, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2,$$

Where E is the sum of the square error for all objects in the data set

p is the point in space representing a given object

m_i is the mean of cluster C_i

The k-means partitioning algorithm:

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster'sccenter is represented by the mean value of the objects in the cluster.

Input:

k: the number of clusters,

D: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

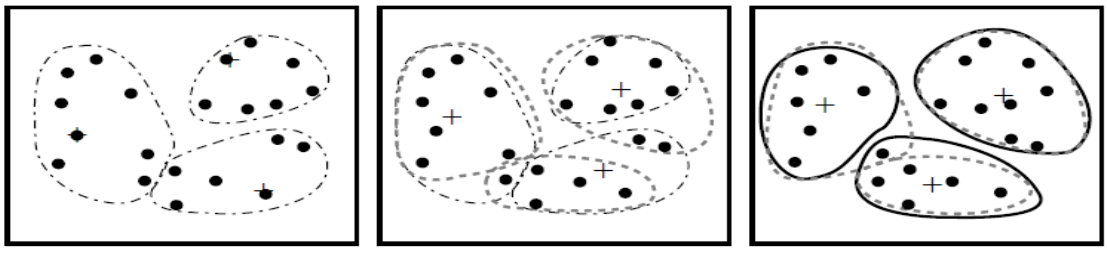
(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;

(2) repeat

(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;

(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;

(5) until no change;



(a) (b) (c)

Clustering of a set of objects based on the *k*-means method. (The mean of each cluster is marked by a "+".)

The k-Medoids Method:

- The k-means algorithm is sensitive to outliers because an object with an extremely large value may substantially distort the distribution of data.

- This effect is particularly exacerbated due to the use of the square-error function. Instead of taking the mean value of the objects in a cluster as a reference point, we can pick actual objects to represent the clusters, using one representative object per cluster.
- Each remaining object is clustered with the representative object to which it is the most similar. The partitioning method is then performed based on the principle of minimizing the sum of the dissimilarities between each object and its corresponding reference point. That is, an absolute-error criterion is used, defined as

$$E = \sum_{j=1}^{k} \sum_{p \in C_j} |p - o_j|,$$

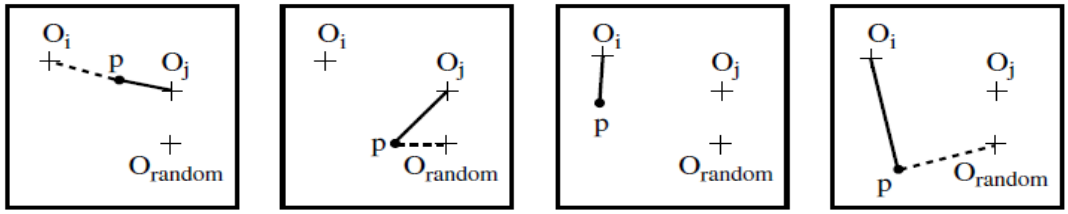
  Where E is the sum of the absolute error for all objects in the data set
  p is the point in space representing a given object in cluster Cj
  oj is the representative object of Cj
- The initial representative objects are chosen arbitrarily. The iterative process of replacing representative objects by non representative objects continues as long as the quality of the resulting clustering is improved.
- This quality is estimated using a cost function that measures the average dissimilarity between an object and the representative object of its cluster.
- To determine whether a non representative object, oj random, is a good replacement for a current representative object, oj, the following four cases are examined for each of the non representative objects.

Case 1: p currently belongs to representative object, o_j. If oj is replaced by o_{random} as a representative object and p is closest to one of the other representative objects, oi ,i≠j, then p is reassigned to oi .

Case 2: $p_{currently}$ belongs to representative object, oj. If oj is replaced by o_{random} as a representative object and p is closest to o_{random}, then p is reassigned to o_{random}.

Case 3: $p_{currently}$ belongs to representative object, oi , i≠j. If oj is replaced by o_{random} as a representative object and p is still closest to oi , then the assignment does not change.

Case 4: $p_{currently}$ belongs to representative object, oi, i≠j. If oj is replaced by o_{random} as a representative object and p is closest to o_{random}, then p is reassigned to o_{random}



The k-Medoids Algorithm:

- The k-medoids algorithm for partitioning based on medoid or central objects.

Algorithm: *k*-medoids. PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects in *D* as the initial representative objects or seeds;
(2) **repeat**
(3) assign each remaining object to the cluster with the nearest representative object;
(4) randomly select a nonrepresentative object, o_{random};
(5) compute the total cost, *S*, of swapping representative object, o_j, with o_{random};
(6) **if** $S < 0$ **then** swap o_j with o_{random} to form the new set of *k* representative objects;
(7) **until** no change;

4. Discuss in detail about Partitioning methods in clustering with Examples. [12M]

- Given *D*, a data set of *n* objects, and *k*, the number of clusters to form, a partitioning algorithm organizes the objects into *k* partitions ($k \leq n$), where each partition represents a cluster.
- The most well-known and commonly used partitioning methods are
  - The k-Means Method
  - k-Medoids Method

Centroid-Based Technique:

- The K-Means Method: The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra cluster similarity is high but the inter cluster similarity is low.
- Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity. The k-means algorithm proceeds as follows.
- First, it randomly selects k of the objects, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster.
- This process iterates until the criterion function converges. Typically, the square-error criterion is used, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2,$$

Where E is the sum of the square error for all objects in the data set

p is the point in space representing a given object

m_i is the mean of cluster C_i

The k-means partitioning algorithm:

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster'sccenter is represented by the mean value of the objects in the cluster.

Input:

k: the number of clusters,

D: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

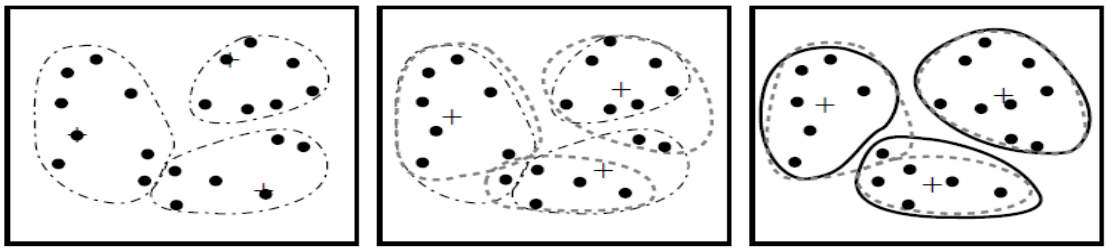
(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;

(2) repeat

(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;

(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;

(5) until no change;



(a) (b) (c)

Clustering of a set of objects based on the *k*-means method. (The mean of each cluster is marked by a "+".)

The k-Medoids Method:

- The k-means algorithm is sensitive to outliers because an object with an extremely large value may substantially distort the distribution of data.
- This effect is particularly exacerbated due to the use of the square-error function. Instead of taking the mean value of the objects in a cluster as a reference point, we can pick actual objects to represent the clusters, using one representative object per cluster.
- Each remaining object is clustered with the representative object to which it is the most similar. The partitioning method is then performed based on the principle of minimizing the sum of the dissimilarities between each object and its corresponding reference point. That is, an absolute-error criterion is used, defined as

$$E = \sum_{j=1}^{k} \sum_{p \in C_j} |p - o_j|,$$

  Where E is the sum of the absolute error for all objects in the data set p is the point in space representing a given object in cluster Cj. oj is the representative object of Cj
- The initial representative objects are chosen arbitrarily. The iterative process of replacing representative objects by non representative objects continues as long as the quality of the resulting clustering is improved.
- This quality is estimated using a cost function that measures the average dissimilarity between an object and the representative object of its cluster.

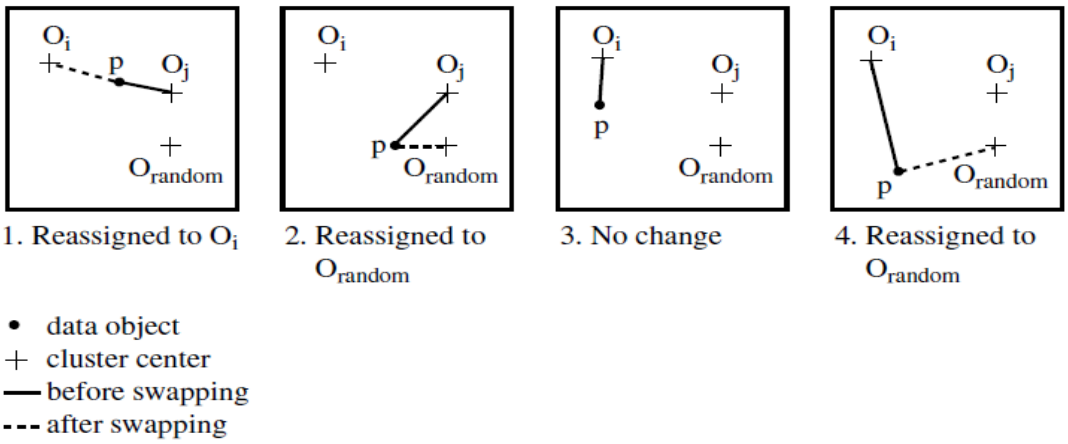
- To determine whether a non representative object, oj random, is a good replacement for a current representative object, oj, the following four cases are examined for each of the non representative objects.

Case 1: p currently belongs to representative object, o_j. If oj is replaced by o_{random} as a representative object and p is closest to one of the other representative objects, oi ,i≠j, then p is reassigned to oi .

Case 2: $p_{currently}$ belongs to representative object, oj. If oj is replaced by o_{random} as a representative object and p is closest to o_{random}, then p is reassigned to o_{random}.

Case 3: $p_{currently}$ belongs to representative object, oi , i≠j. If oj is replaced by o_{random} as a representative object and p is still closest to oi , then the assignment does not change.

Case 4: $p_{currently}$ belongs to representative object, oi, i≠j. If oj is replaced by o_{random} as a representative object and p is closest to o_{random}, then p is reassigned to o_{random}



The k-Medoids Algorithm:

- The k-medoids algorithm for partitioning based on medoid or central objects.

Algorithm: *k*-**medoids.** PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects in *D* as the initial representative objects or seeds;
(2) **repeat**
(3) assign each remaining object to the cluster with the nearest representative object;
(4) randomly select a nonrepresentative object, o_{random};
(5) compute the total cost, *S*, of swapping representative object, o_j, with o_{random};
(6) **if** $S < 0$ **then** swap o_j with o_{random} to form the new set of *k* representative objects;
(7) **until** no change;

5. Discuss the key issues in hierarchical clustering algorithm. [12M]

Key Issues in Hierarchical Clustering

1. Lack of a Global Objective Function:

- Agglomerative hierarchical clustering techniques perform clustering on a local level and as such there is no global objective function like in the K-Means algorithm.
- This is actually an advantage of this technique because the time and space complexity of global functions tends to be very expensive.

2. Ability to Handle Different cluster Sizes:

- We have to decide how to treat clusters of various sizes that are merged together.

3. Merging Decisions Are Final:

- One downside of this technique is that once two clusters have been merged they cannot be split up at a later time for a more favorable union.

6. a. Compare Agglomerative and Divisive hierarchical clustering. [6M]

Agglomerative hierarchical clustering:

- This bottom-up strategy starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are satisfied.
- Most hierarchical clustering methods belong to this category. They differ only in their definition of inter cluster similarity.

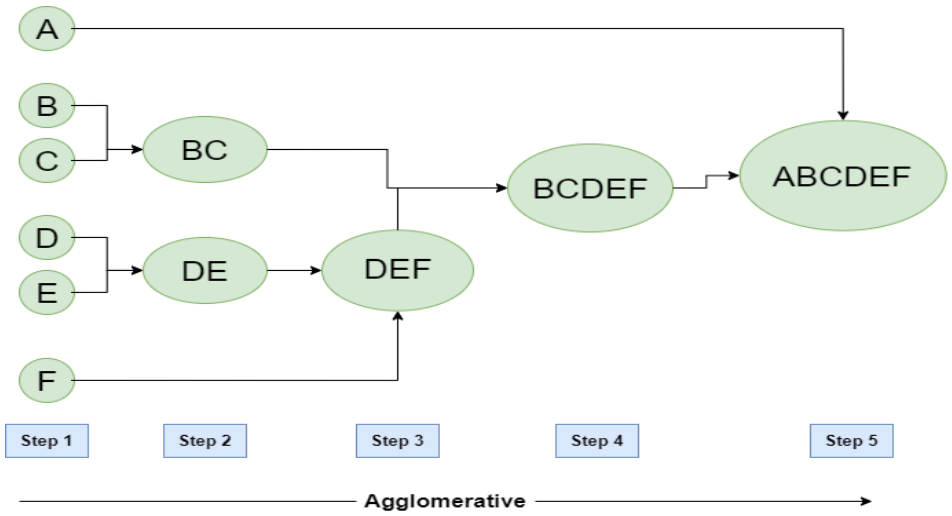
Divisive hierarchical clustering:

- This top-down strategy does the reverse of agglomerative hierarchical clustering by starting with all objects in one cluster.
- It sub divides the cluster into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfies certain termination conditions, such as a desired number of clusters is obtained or the diameter of each cluster is within a certain threshold.

6. b. What are the basic approaches for generating an agglomerative hierarchical clustering? Explain the algorithm. [6M]

Agglomerative:

- Initially consider every data point as an **individual** Cluster and at every step, **merge** the nearest pairs of the cluster. (It is a bottom-up method).
- At first, every dataset is considered an individual entity or cluster. At every iteration, the clusters merge with different clusters until one cluster is formed.
- The algorithm for Agglomerative Hierarchical Clustering is:
    - Calculate the similarity of one cluster with all the other clusters (calculate proximity matrix).
    - Consider every data point as an individual cluster.
    - Merge the clusters which are highly similar or close to each other.
    - Recalculate the proximity matrix for each cluster.
    - Repeat Steps 3 and 4 until only a single cluster remains.



Step-1: Consider each alphabet as a single cluster and calculate the distance of one cluster from all the other clusters.

Step-2: In the second step comparable clusters are merged together to form a single cluster. Let's say cluster (B) and cluster (C) are very similar to each other therefore we merge them in the second step similarly to cluster (D) and (E) and at last, we get the clusters [(A), (BC), (DE), (F)]

Step-3: We recalculate the proximity according to the algorithm and merge the two nearest clusters ([(DE), (F)]) together to form new clusters as [(A), (BC), (DEF)]

Step-4: Repeating the same process; The clusters DEF and BC are comparable and merged together to form a new cluster. We're now left with clusters [(A), (BCDEF)].

Step-5: At last the two remaining clusters are merged together to form a single cluster [(ABCDEF)].

7. a. Explain the following clustering methods in detail: [6M]

i) BIRCH

ii) CURE

i) BIRCH

- BIRCH is designed for clustering a large amount of numerical data by integration of hierarchical clustering (at the initial micro clustering stage) and other clustering methods such as iterative partitioning (at the later macro clustering stage).
- It overcomes the two difficulties of agglomerative clustering methods:

  (1) scalability and

(2) the inability to undo what was done in the previous step.

- BIRCH introduces two concepts, clustering feature and clustering feature tree (CF tree), which are used to summarize cluster representations.
- These structures help the clustering method achieve good speed and scalability in large databases and also make it effective for incremental and dynamic clustering of incoming objects.
    - Let's look closer at the above-mentioned structures. Given n d-dimensional data objects or points in a cluster, we can define the centroid X_0, radius R, and diameter D of the cluster as follows:

$$x_0 = \frac{\sum_{i=1}^{n} x_i}{n}$$

$$R = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_0)^2}{n}}$$

$$D = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j=1}^{n} (x_i - x_j)^2}{n(n-1)}}$$

where *R* is the average distance from member objects to the centroid, and *D* is the average pair wise distance within a cluster.

- Both *R* and *D* reflect the tightness of the cluster around the centroid. A clustering feature (CF) is a three-dimensional vector summarizing information about clusters of objects. Given *n d*-dimensional objects or points in a cluster, {***xi***}, then the CF of the cluster is defined as

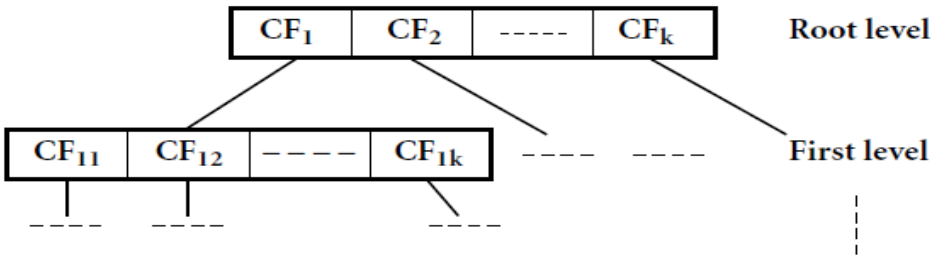
$$CF = \langle n, \boldsymbol{LS}, \boldsymbol{SS} \rangle,$$

where *n* is the number of points in the cluster, ***LS*** is the linear sum of the *n* points, and ***SS*** is the square sum of the data points.

- BIRCH applies a *multiphase* clustering technique: a single scan of the data set yields a basic good clustering, and one or more additional scans can (optionally) be used to further improve the quality.

- The primary phases are:

Phase 1: BIRCH scans the database to build an initial in-memory CF tree, which can be viewed as a multilevel compression of the data that tries to preserve the inherent clustering structure of the data.



Phase 2: BIRCH applies a (selected) clustering algorithm to cluster the leaf nodes of the CF tree, which removes sparse clusters as outliers and groups dense clusters into larger ones.

ii) ROCK

- ROCK stands for Robust Clustering using links. It is a hierarchical clustering algorithm that analyze the concept of links (the number of common neighbours among two objects) for data with categorical attributes.
- It display that such distance data cannot lead to high-quality clusters when clustering categorical information.
- Moreover, most clustering algorithms create only the similarity among points when clustering i.e., at each step, points that are combined into a single cluster. This "localized" method is prone to bugs.
- For instance, two distinct clusters can have a few points or outliers that are near; thus, relying on the similarity among points to create clustering decisions can generate the two clusters to be combined.
- ROCK takes a more global method to clustering by treating the neighborhoods of single pairs of points. If two similar points also have same neighborhoods, thus the two points likely belong to the similar cluster and so can be combined.
- There are two points, p_i and p_j, are neighbors if $sim(p_i, p_j) \geq \theta$, where sim is a similarity function and θ is a user-specified threshold. It can select sim to be a distance metric or even a nonmetric that is normalized so that its values fall among 0 and 1, with higher values denoting that the points are more same.

- The number of connection between p_i and p_j is represented as the number of common neighbors between p_i and p_j. If the number of links between two points is high, then it is more likely that they belong to the similar cluster.
- In the data for a transaction, the attribute corresponding to an item is correct if the transaction include the item; otherwise, it is false. There are several data sets with categorical attributes can be managed in a same manner. ROCK's terms of neighbors and links are the same between two "points" or transactions, T_i and T_j, is represented with the Jaccard coefficient as

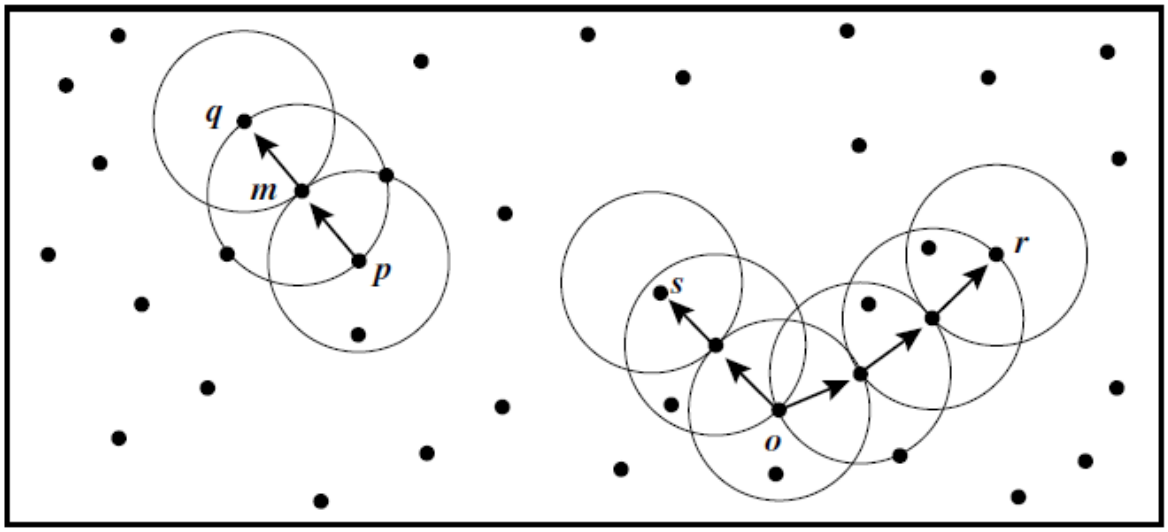
$$sim(T_i, T_j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|}.$$

- ROCK first produce a sparse graph from a given data similarity matrix utilizing a similarity threshold and the approach of shared neighbors. It can implements agglomerative hierarchical clustering on the sparse graph. A goodness measure can compute the clustering. Random sampling can be used for scaling up to high data sets.

7. b. How clusters are identified using DBSCAN algorithm? [6M]

- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density based clustering algorithm.
- The algorithm grows regions with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial databases with noise.
- It defines a cluster as a maximal set of *density-connected* points. The basic ideas of density-based clustering involve a number of new definitions.
- The neighborhood within a radius e of a given object is called the e-neighborhood of the object.
- If the e-neighborhood of an object contains at least a minimum number, *MinPts*, of objects, then the object is called a core object.
- Given a set of objects, *D*, we say that an object ***p*** is directly density-reachable from object ***q*** if ***p*** is within the e-neighborhood of ***q***, and ***q*** is a core object.
- An object ***p*** is density-reachable from object ***q*** with respect to e and *MinPts* in a set of objects, *D*, if there is a chain of objects $\boldsymbol{p_1}, \ldots, \boldsymbol{p_n}$, where $\boldsymbol{p_1} = \boldsymbol{q}$ and $\boldsymbol{p_n} = \boldsymbol{p}$ such that $\boldsymbol{p_{i+1}}$ is directly density-reachable from $\boldsymbol{p_i}$ with respect to e and *MinPts*, for $1 \leq i \leq n$, $\boldsymbol{p_i} \in D$.

- An object ***p*** is density-connected to object ***q*** with respect to e and *MinPts* in a set of objects, *D*, if there is an object **o** ∈ *D* such that both ***p*** and ***q*** are density-reachable from ***o*** with respect to e and *MinPts*.
- DBSCAN searches for clusters by checking the e-neighborhood of each point in the database.
- If the e-neighborhood of a point ***p*** contains more than *MinPts*, a new cluster with ***p*** as a core object is created.
- DBSCAN then iteratively collects directly density-reachable objects from these core objects, which may involve the merge of a few density-reachable clusters.
- The process terminates when no new point can be added to any cluster.



8. Influence the importance of Grid-based and Model-Based methods in detail. [12M]

Grid-Based Method:

- In the Grid-Based method a grid is formed using the object together, i.e, the object space is quantized into a finite number of cells that form a grid structure.
- One of the major advantages of the grid-based method is fast processing time and it is dependent only on the number of cells in each dimension in the quantized space.
- The processing time for this method is much faster so it can save time.

Model-Based Method:

- In the model-based method, all the clusters are hypothesized in order to find the data which is best suited for the model.
- The clustering of the density function is used to locate the clusters for a given model. It reflects the spatial distribution of data points and also provides a way to automatically

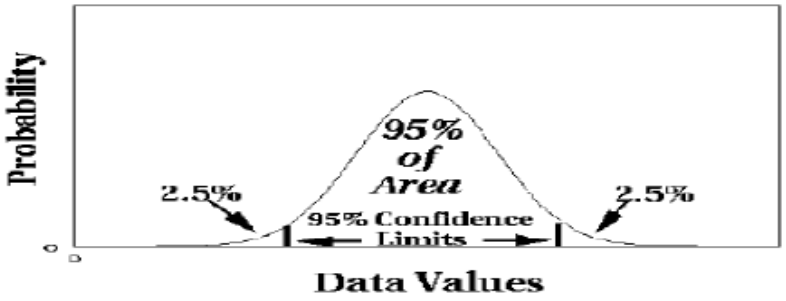
determine the number of clusters based on standard statistics, taking outlier or noise into account.

- Therefore it yields robust clustering methods.

9. What is outlier analysis? Name the methods for detecting outliers. [12M]

What Is Outlier Discovery?

- **What are outliers?**
  - The set of objects are considerably dissimilar from the remainder of the data.
  - Example: Sports: Michael Jordon, Wayne Gretzky.
- **Problem.**
  - Find top n outlier points.
- **Applications:**
  - Credit card fraud detection.
  - Telecom fraud detection.
  - Customer segmentation.
  - Medical analysis.
- **Outlier Discovery: Statistical Approaches.**



- Assume a model underlying distribution that generates data set (e.g. normal distribution).
  - Use discordance tests depending on.
    - data distribution.
    - distribution parameter (e.g., mean, variance).
    - number of expected outliers.
- Drawbacks.
  - most tests are for single attribute.
  - In many cases, data distribution may not be known.

Outlier Discovery: Distance-Based Approach.

- Introduced to counter the main limitations imposed by statistical methods.
    - We need multi-dimensional analysis without knowing data distribution.
- Distance-based outlier: A DB (p, D)-outlier is an object O in a dataset T such that at least a fraction p of the objects in T lies at a distance greater than D from O.
- Algorithms for mining distance-based outliers.
    - Index-based algorithm.
    - Nested-loop algorithm.
    - Cell-based algorithm.

Outlier Discovery: Deviation-Based Approach.

- Identifies outliers by examining the main characteristics of objects in a group.
- Objects that "deviate" from this description are considered outliers.
- Sequential exception technique.
    - simulates the way in which humans can distinguish unusual objects from among a series of supposedly like objects.
- OLAP data cube technique.
    - uses data cubes to identify regions of anomalies in large multidimensional data.

10. Discuss in detail about the Applications and trends in Data Mining. [12M]

- Data mining is a young discipline with wide and diverse applications.
    - There is still a nontrivial gap between general principles of data mining and domain-specific, effective data mining tools for particular applications.
- Some application domains.
    - Biomedical and DNA data analysis.
    - Financial data analysis.
    - Retail industry.
    - Telecommunication industry.

Biomedical Data Mining and DNA Analysis:

- DNA sequences: 4 basic building blocks (nucleotides): adenine (A), cytosine (C), guanine (G), and thymine (T).
- Gene: a sequence of hundreds of individual nucleotides arranged in a particular order.
- Humans have around 100,000 genes.
- Tremendous number of ways that the nucleotides can be ordered and sequenced to form distinct genes.

Data Mining for Financial Data Analysis:

- Financial data collected in banks and financial institutions are often relatively complete, reliable, and of high quality.
- Design and construction of data warehouses for multidimensional data analysis and data mining.
    - View the debt and revenue changes by month, by region, by sector, and by other factors.
    - Access statistical information such as max, min, total, average, trend, etc.
- Loan payment prediction/consumer credit policy analysis.
    - feature selection and attribute relevance ranking.
    - Loan payment performance.
    - Consumer credit rating.

Financial Data Mining:

- Classification and clustering of customers for targeted marketing.
    - multidimensional segmentation by nearest-neighbor, classification, decision trees, etc. to identify customer groups or associate a new customer to an appropriate customer group.
- Detection of money laundering and other financial crimes.
    - integration of from multiple DBs (e.g., bank transactions, federal/state crime history DBs).
    - Tools: data visualization, linkage analysis, classification, clustering tools, outlier analysis, and sequential pattern analysis tools (find unusual access sequences).

Data Mining for Retail Industry:

- Retail industry: huge amounts of data on sales, customer shopping history, etc.
- Applications of retail data mining.
    - Identify customer buying behaviors.
    - Discover customer shopping patterns and trends.
    - Improve the quality of customer service.
    - Achieve better customer retention and satisfaction.
    - Enhance goods consumption ratios.
    - Design more effective goods transportation and distribution policies.

Data Mining for Telecomm. Industry:

- A rapidly expanding and highly competitive industry and a great demand for data mining.

        - Understand the business involved.
        - Identify telecommunication patterns.
        - Catch fraudulent activities.
        - Make better use of resources.
        - Improve the quality of service.
    - Multidimensional analysis of telecommunication data.
        - Intrinsically multidimensional: calling-time, duration, location of caller, location of callee, type of call, etc.
    - Fraudulent pattern analysis and the identification of unusual patterns.
        - Identify potentially fraudulent users and their atypical usage patterns.
        - Detect attempts to gain fraudulent entry to customer accounts.
        - Discover unusual patterns which may need special attention.
    - Multidimensional association and sequential pattern analysis.
        - Find usage patterns for a set of communication services by customer group, by month, etc.
        - Promote the sales of specific services.
        - Improve the availability of particular services in a region.
- Use of visualization tools in telecommunication data analysis.