UNIT - I

Introduction: Why Data Mining? What Is Data Mining? What Kinds of Data Can Be Mined? What Kinds of Patterns Can Be Mined? Which Technologies Are Used? Which Kinds of Applications Are Targeted? Major Issues in Data Mining. Data Objects and Attribute Types, Basic Statistical Descriptions of Data, Data Visualization, Measuring Data Similarity and Dissimilarity

1. Why Data Mining?

We live in a world where vast amounts of data are collected daily. Analyzing such data is an important need. The Explosive Growth of Data: from terabytes to petabytes

- Data collection and data availability
  - Automated data collection tools, database systems, Web, computerized society
- Major sources of abundant data
  - Business: Web, e-commerce, transactions, stocks,
  - Science: Remote sensing, bioinformatics, scientific simulation,
  - Society and everyone: news, digital cameras, YouTube
- We are drowning in data, but starving for knowledge!

Data Mining as the Evolution of Information Technology

Data mining can be viewed as a result of the natural evolution of information technology. The database and data management industry evolved in the development of several critical functionalities

Before 1600, empirical science

1600-1950s, theoretical science:

- Each discipline has grown a theoretical component. Theoretical models often motivate experiments and generalize our understanding.

1950s-1990s, computational science:

- Over the last 50 years, most disciplines have grown a third, computational branch (e.g. empirical, theoretical, and computational ecology, or physics, or linguistics.)
- Computational Science traditionally meant simulation. It grew out of our inability to find closed-form solutions for complex mathematical models.

1990-now, data science

- The flood of data from new scientific instruments and simulations
- The ability to economically store and manage petabytes of data online
- The Internet and computing Grid that makes all these archives universally accessible
- Scientific info. Management, acquisition, organization, query, and visualization tasks scale almost linearly with data volumes. Data mining is a major new challenge!

1960s:

- Data collection, database creation, IMS and network DBMS

1970s:

- Relational data model, relational DBMS implementation

1980s:

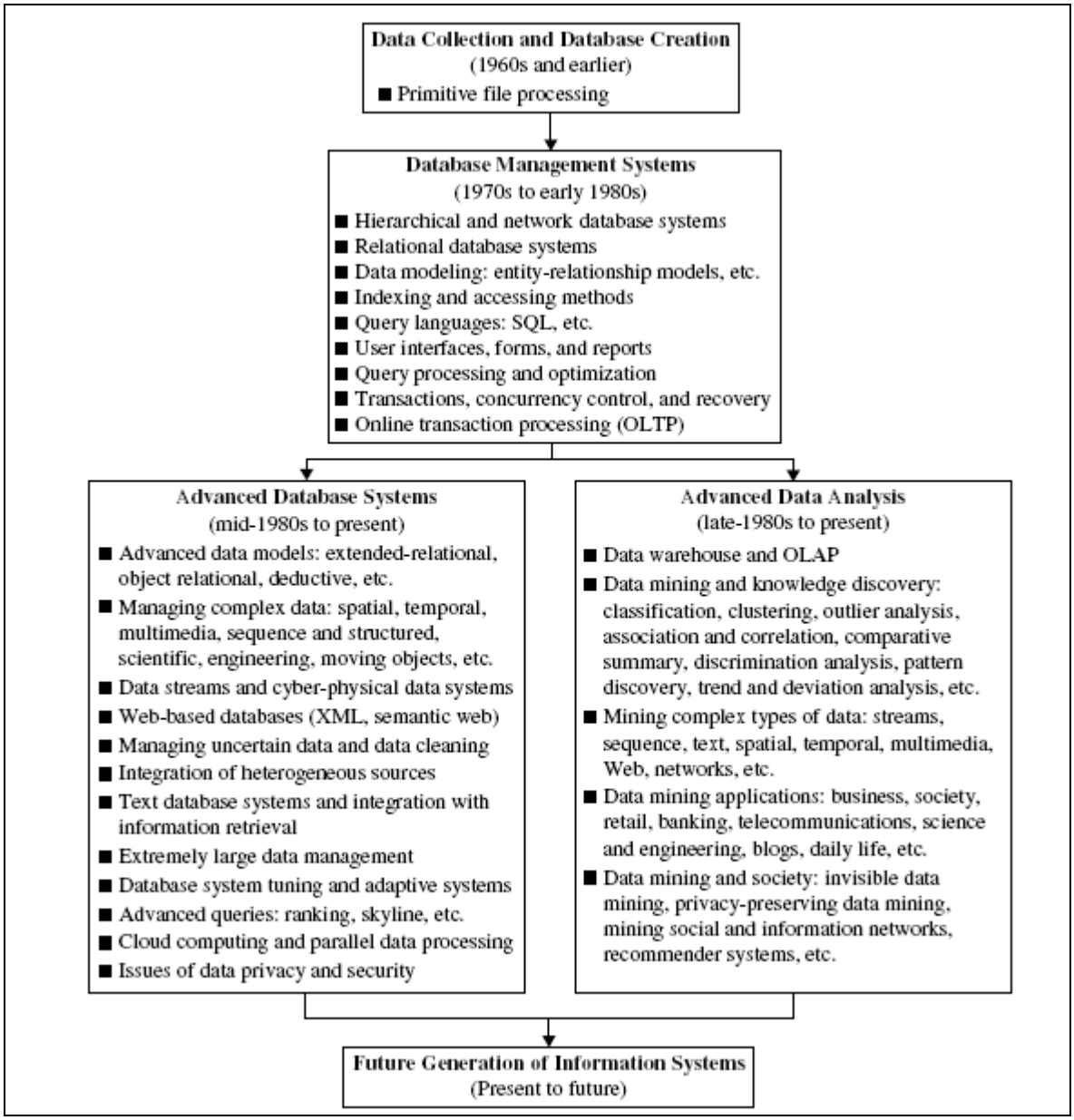
- RDBMS, advanced data models (extended-relational, OO, deductive, etc.) Application-oriented DBMS (spatial, scientific, engineering, etc.)

1990s:

- Data mining, data warehousing, multimedia databases, and Web databases

2000s:

- Stream data management and mining, Data mining and its applications, Web technology (XML, data integration) and global information systems

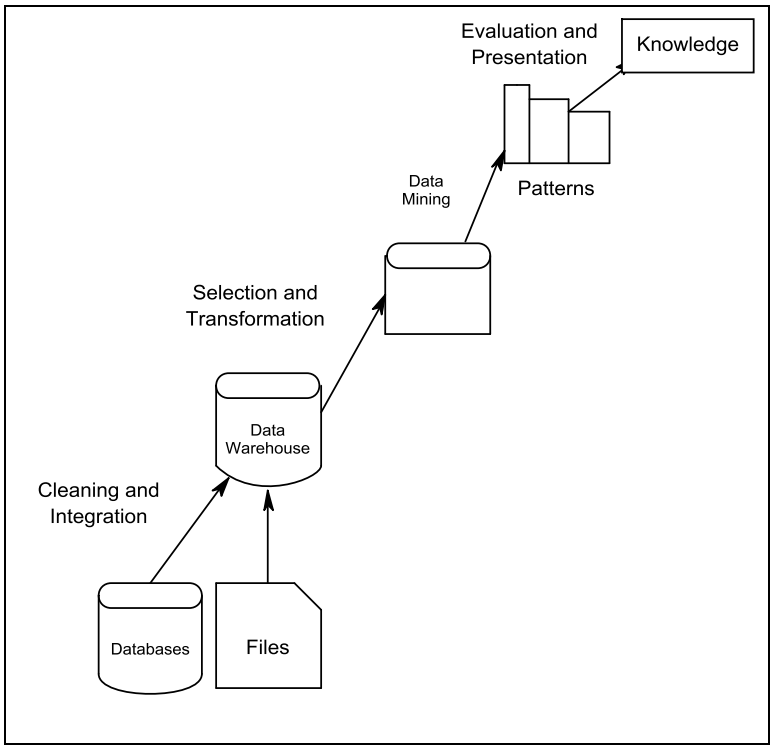


2. **What Is Data Mining?** Data mining (knowledge discovery from data)

Data mining is the process of discovering interesting patterns and knowledge from large amounts of data. The data sources can include databases, data warehouses, the Web, other information repositories, or data that are streamed into the system dynamically.

- Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data
- Alternative names: Knowledge discovery (mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.

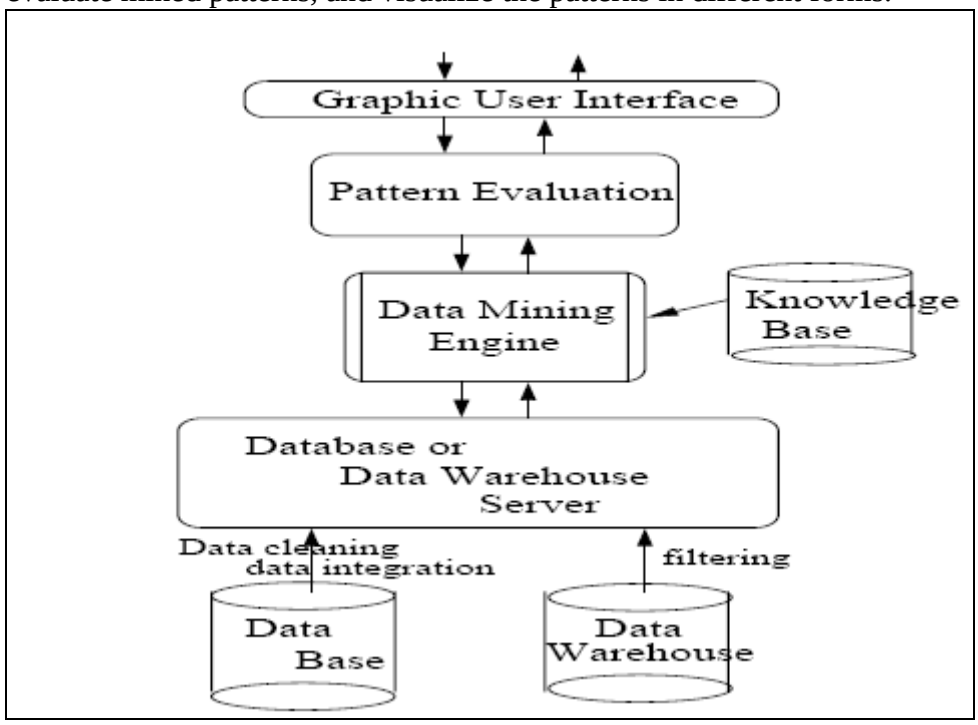
The knowledge discovery process is shown in Following Figure as an iterative sequence of the following steps:
1. Data cleaning (to remove noise and inconsistent data)
2. Data integration (where multiple data sources may be combined)
3. Data selection (where data relevant to the analysis task are retrieved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



Steps 1 through 4 are different forms of data preprocessing, where 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.

Architecture of Data Mining System: - The architecture of a data mining system may have the following components
1. Data Sources or Repositories: - This component represents multiple data sources such as database, data warehouse, or any other information repository. Data cleaning and data integration techniques may be performed on the data.
2. Database server or data warehouse server: - The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.
3. Knowledge base: - . It is the area of knowledge that is used to guide the search, or to perform analysis of the resulting patterns.

4. Data mining engine: - This is core component to the data mining system and consists of a set of functional modules for tasks such as characterization, association analysis, classification, and evolution and deviation analysis.
5. Pattern evaluation module: - This component typically employs interestingness measures and interacts with the data mining modules so as to focus the search towards interesting patterns.
6. Graphical user interface: - This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task. This component allows the user to browse database and data warehouse schemas or data structures, evaluate mined patterns, and visualize the patterns in different forms.



3. What Kinds of Data Can Be Mined?

Data mining can be applied to any kind of data as long as the data are meaningful for a target application.

I. **Database Data:** A database system, also called a database management system (DBMS), consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data. A relational database is a collection of tables, each of which is assigned a unique name. Each table consists of a set of attributes (columns or fields) and usually stores a large number of tuples (records or rows). Each tuple in a relational table represents a record identified by a unique key and described by a set of attribute values.

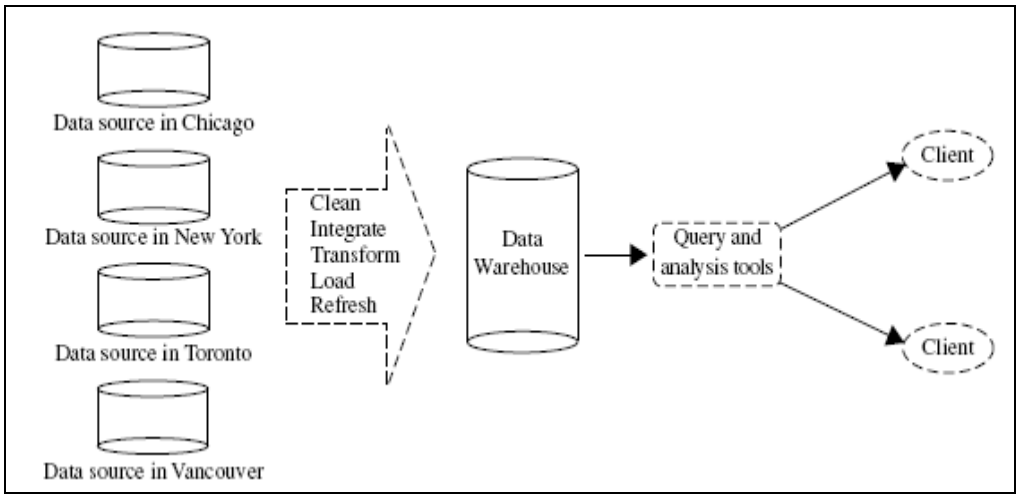
Ex:-

St_ID	St_Name	Address
101	Sai	Ramireddypeta
102	Meghana	Kukatpally

St_ID	Marks
101	90
102	85

II. 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. 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.



III. **Transactional databases:** - A transactional database consists of a 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. The Transactional database may have additional tables associated with it, which contain other information regarding the sale, such as the date of the transaction, the customer ID number, the ID number of the sales person, and of the branch at which the sale occurred, and so on

trans_ID	*list_of_item_IDs*
T100	I1, I3, I8, I16
T200	I2, I8
…	…

IV. Advanced data sets and advanced applications
- Data streams and sensor data
  - Many applications involve the generation and analysis of a new kind of data, called stream data, where data flow in and out of an observation platform (or window) dynamically.
  - **Example:** video surveillance and sensor data, which are continuously transmitted)
- Time-series data, temporal data, sequence data
  - A temporal database typically stores relational data that include time-related attributes. These attributes may involve several timestamps, each having different semantics.

        - A sequence database stores sequences of ordered events, with or without a concrete notion of time. Examples include customer shopping sequences, Web click streams, and biological sequences.
        - A time-series database stores sequences of values or events obtained over repeated measurements of time (e.g., hourly, daily, weekly).
        - **Example:** historical records, stock exchange data, and time-series and biological sequence data
    - Structure data, graphs, social networks and multi-linked data
    - Object-relational databases
        - Object-relational databases are constructed based on an object-relational data model. This model extends the relational model by providing a rich data type for handling complex objects and object orientation.
        - Conceptually, the object-relational data model inherits the essential concepts of object-oriented databases, where, in general terms, each entity is considered as an object.
        -
    - Spatial data and spatiotemporal data
        - Spatial databases contain spatial-related information. Examples include geographic (Map) databases, very large-scale integration (VLSI) or computer-aided design databases, and medical and satellite image databases
        - A spatial database that stores spatial objects that change with time is called a spatiotemporal database, from which interesting information can be mined. **For example**, we may be able to group the trends of moving objects and identify some strangely moving vehicles
    - Multimedia database
        - Multimedia databases store image, audio, and video data. They are used in applications such as picture content-based retrieval, voice-mail systems, video-on-demand systems, the World Wide Web, and speech-based user interfaces that recognize spoken commands.
    - Text databases
        - Text databases are databases that contain word descriptions for objects. Text databases may be highly unstructured (such as some Web pages on the World Wide Web). Some text databases may be somewhat structured, that is, semi structured such as e-mail messages and many HTML/XML Web pages), whereas others are relatively well structured (such as library catalogue databases).
        -
    - The World-Wide Web
        - (a huge, widely distributed information repository made available by the Internet)
    - Heterogeneous databases and legacy databases
        - A heterogeneous database consists of a set of interconnected, autonomous component databases. The components communicate in order to exchange information and answer queries.
        - A legacy database is a group of heterogeneous databases that combines different kinds of data systems, such as relational or object-oriented databases, hierarchical databases, network databases, spreadsheets, multimedia databases, or file systems. The heterogeneous databases in a legacy database may be connected by intra or inter-computer networks.

4. What Kinds of Patterns Can Be Mined?

There are a number of data mining functionalities. Data mining functionalities are used to specify the kinds of patterns to be found in data mining tasks. In general, such tasks can be classified into two categories: descriptive and predictive. Descriptive mining tasks characterize properties of the data in a target data set. Predictive mining tasks perform induction on the current data in order to make predictions.

Class/Concept Description: Characterization and Discrimination

Data entries can be associated with classes or concepts. These descriptions can be derived using data characterization or data discrimination or 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 query. **For example**, to study the characteristics of software products with sales that increased by 10% in the previous year, the data related to such products can be collected by executing an SQL query on the sales database. Output formats include pie charts, bar charts, curves, multidimensional data cubes, and multidimensional tables, including crosstabs.

Data discrimination is a comparison of the general features of the target class data objects against the general features of objects from one or multiple contrasting classes. The target and contrasting classes can be specified by a user, and the corresponding data objects can be retrieved through database queries. **For example**, a user may want to compare the general features of software products with sales that increased by 10% last year against those with sales that decreased by at least 30% during the same period. The methods used for data discrimination are similar to those used for data characterization.

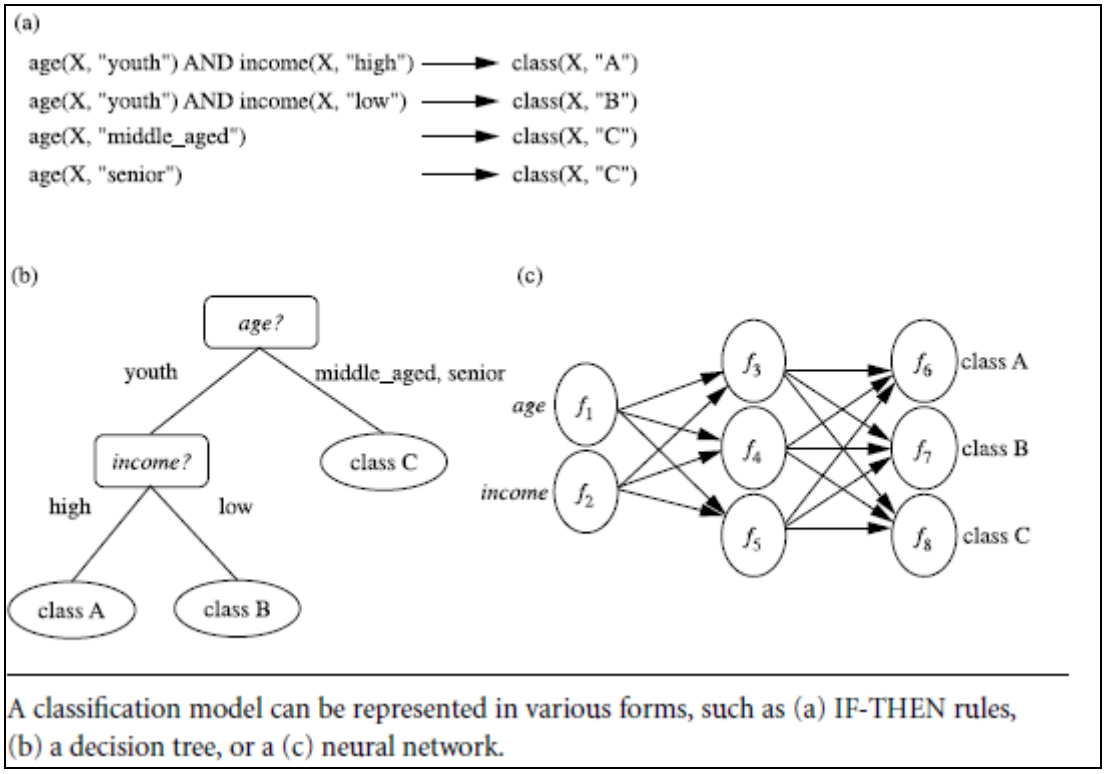
Mining Frequent Patterns, Associations, and Correlations: - Association analysis is the discovery of association rules showing attribute-value conditions that occur frequently together in a given set of data. Association analysis is widely used for market basket or transaction data analysis.

Ex: - The association rules may be specified as

Age(X; “20…29") ^ income(X; “20K…30K") => buys(X, “CD player")

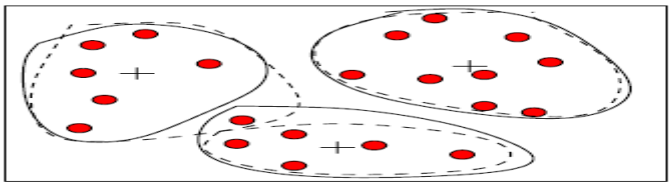
[Support = 2%; confidence = 60%]

Classification and prediction: - Classification is the processing of finding a set of models (or functions) which describe and distinguish data classes or concepts 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). Classification can be used for predicting the class label of data objects. Prediction may refer to both data value prediction and class label prediction; it is usually confined to data value prediction and thus is distinct from classification.



A classification model can be represented in various forms, such as (a) IF-THEN rules, (b) a decision tree, or a (c) neural network.

Clustering analysis: - Clustering analyzes data objects without consulting a known class label. Clustering can be used to generate class labels. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the interclass similarity. That is, clusters of objects are formed so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Each cluster that is formed can be viewed as a class of objects, from which rules can be derived.



Outlier Analysis

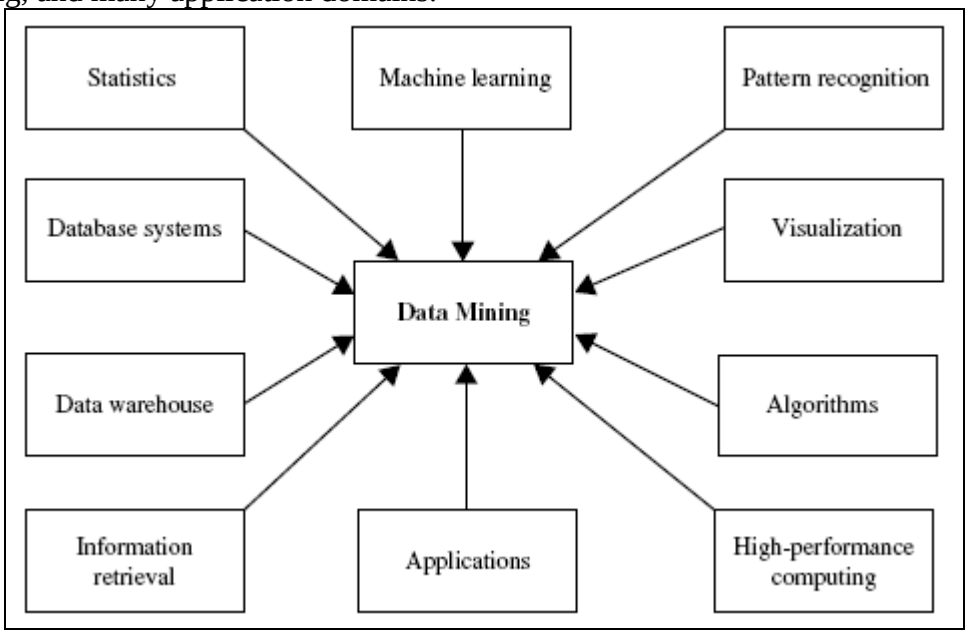
A database may contain data objects that do not comply with the general behavior or Model of the data. These data objects are outliers. Most data mining methods discard outliers as noise or exceptions. However, in some applications such as fraud detection, the rare events can be more interesting than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

Outlier analysis may uncover fraudulent usage of credit cards by detecting purchases of extremely large amounts for a given account number in comparison to regular charges incurred by the same account.

Evolution and deviation analysis: - Data evolution analysis describes and models regularities or trends for objects whose behavior changes over time. Although this may include characterization, discrimination, association, classification, 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.

5. Which Technologies Are Used?

As a highly application-driven domain, data mining has incorporated many techniques from other domains such as statistics, machine learning, pattern recognition, database and data warehouse systems, information retrieval, visualization, algorithms, high performance computing, and many application domains.



Statistics

Statistics studies the collection, analysis, interpretation or explanation, and presentation of data. Data mining has an inherent connection with statistics.

A statistical model is a set of mathematical functions that describe the behavior of the objects in a target class in terms of random variables and their associated probability distributions.

Machine learning investigates how computers can learn (or improve their performance) based on data. A main research area is for computer programs to automatically learn to recognize complex patterns and make intelligent decisions based on data. For example, a typical machine learning problem is to program a computer so that it can automatically recognize handwritten postal codes on mail after learning from a set of examples. Machine learning is a fast-growing discipline. Here, we illustrate classic problems in machine learning that are highly related to data mining.

- **Supervised learning** is basically a synonym for classification. The supervision in the learning comes from the labeled examples in the training data set. **For example**, in the postal code recognition problem, a set of handwritten postal code images and their corresponding machine-readable translations are used as the training examples, which supervise the learning of the classification model.

- **Unsupervised learning** is essentially a synonym for clustering. The learning process is unsupervised since the input examples are not class labeled. Typically, we may use clustering to discover classes within the data. **For example**, an unsupervised learning method can take, as input, a set of images of handwritten digits. Suppose that it finds 10 clusters of data. These clusters may correspond to the 10 distinct digits of 0 to 9,

respectively. However, since the training data are not labeled, the learned model cannot tell us the semantic meaning of the clusters found.

- **Semi-supervised learning** is a class of machine learning techniques that make use of both labeled and unlabeled examples when learning a model. In one approach, labeled examples are used to learn class models and unlabeled examples are used to refine the boundaries between classes. For a two-class problem, we can think of the set of examples belonging to one class as the positive examples and those belonging to the other class as the negative examples. In Figure 1.12, if we do not consider the unlabeled examples, the dashed line is the decision boundary that best partitions the positive examples from the negative examples. Using the unlabeled examples, we can refine the decision boundary to the solid line. Moreover, we can detect that the two positive examples at the top right corner, though labeled, are likely noise or outliers.

- **Active learning** is a machine learning approach that lets users play an active role in the learning process. An active learning approach can ask a user (e.g., a domain expert) to label an example, which may be from a set of unlabeled examples or synthesized by the learning program. The goal is to optimize the model quality by actively acquiring knowledge from human users, given a constraint on how many examples they can be asked to label.

Database Systems and Data Warehouses

Database systems research focuses on the creation, maintenance, and use of databases for organizations and end-users. Particularly, database systems researchers have established highly recognized principles in data models, query languages, query processing and optimization methods, data storage, and indexing and accessing methods.

A **data warehouse** integrates data originating from multiple sources and various timeframes. It consolidates data in multidimensional space to form partially materialized data cubes. The data cube model not only facilitates OLAP in multidimensional databases but also promotes multidimensional data mining

Information Retrieval

Information retrieval (IR) is the science of searching for documents or information in documents. Documents can be text or multimedia, and may reside on the Web. The differences between traditional information retrieval and database systems are twofold: Information retrieval assumes that (1) the data under search are unstructured; and (2) the queries are formed mainly by keywords, which do not have complex structures (unlike SQL queries in database systems).

6. Which Kinds of Applications Are Targeted?

As a highly application-driven discipline, data mining has seen great successes in many applications.

Business Intelligence

It is critical for businesses to acquire a better understanding of the commercial context of their organization, such as their customers, the market, supply and resources, and competitors. Business intelligence (BI) technologies provide historical, current, and predictive views of business operations. Examples include reporting, online analytical processing, business performance management, competitive intelligence, benchmarking, and predictive analytics.

"How important is business intelligence?" Without data mining, many businesses may not be able to perform effective market analysis, compare customer feedback on similar products, discover the strengths and weaknesses of their competitors, retain highly valuable customers, and make smart business decisions.

Web Search Engines

A Web search engine is a specialized computer server that searches for information on the Web. The search results of a user query are often returned as a list (sometimes called hits). The hits may consist of web pages, images, and other types of files. Some search engines also search and return data available in public databases or open directories. Search engines differ from web directories in that web directories are maintained by human editors whereas search engines operate algorithmically or by a mixture of algorithmic and human input.

Web search engines are essentially very large data mining applications. Various data mining techniques are used in all aspects of search engines, ranging from crawling 5 (e.g., deciding which pages should be crawled and the crawling frequencies), indexing (e.g., selecting pages to be indexed and deciding to which extent the index should be constructed), and searching (e.g., deciding how pages should be ranked, which advertisements should be added, and how the search results can be personalized or made "context aware").

7. Major Issues in Data Mining

Data mining is a dynamic and fast-expanding field with great strengths. In this section, we briefly outline the major issues in data mining research, partitioning them into five groups: mining methodology, user interaction, efficiency and scalability, diversity of data types, and data mining and society.

- **Mining Methodology**
  - Mining various and new kinds of knowledge
    - Data mining covers a wide spectrum of data analysis and knowledge discovery tasks, from data characterization and discrimination to association and correlation analysis, classification, regression, clustering, outlier analysis, sequence analysis, and trend and evolution analysis.
  - Mining knowledge in multi-dimensional space
    - When searching for knowledge in large data sets, we can explore the data in multidimensional space. That is, we can search for interesting patterns among combinations of dimensions (attributes) at varying levels of abstraction.
  - Data mining: An interdisciplinary effort
    - The power of data mining can be substantially enhanced by integrating new methods from multiple disciplines. For example, to mine data with natural language text, it makes sense to fuse data mining methods with methods of information retrieval and natural language processing.
  - Boosting the power of discovery in a networked environment
    - Most data objects reside in a linked or interconnected environment, whether it is the Web, database relations, files, or documents. Semantic links across multiple data objects can be used to advantage in data mining.
  - Handling noise, uncertainty, and incompleteness of data
    - Data often contain noise, errors, exceptions, or uncertainty, or are incomplete. Errors and noise may confuse the data mining process, leading

to the derivation of erroneous patterns. Data cleaning, data preprocessing, outlier detection and removal, and uncertainty reasoning are examples of techniques that need to be integrated with the data mining process.

- Pattern evaluation and pattern- or constraint-guided mining
  - Not all the patterns generated by data mining processes are interesting. What makes a pattern interesting may vary from user to user. Therefore, techniques are needed to assess the interestingness of discovered patterns based on subjective measures.

- **User Interaction**
  - Interactive mining
    - The data mining process should be highly interactive. Thus, it is important to build flexible user interfaces and an exploratory mining environment, facilitating the user's interaction with the system. A user may like to first sample a set of data, explore general characteristics of the data, and estimate potential mining results.
  - Incorporation of background knowledge
    - Background knowledge, constraints, rules, and other information regarding the domain under study should be incorporated into the knowledge discovery process.
  - Ad hoc data mining and data mining query languages:
    - Query languages (e.g., SQL) have played an important role in flexible searching because they allow users to pose ad hoc queries. Similarly, high-level data mining query languages or other high-level flexible user interfaces will give users the freedom to define ad hoc data mining tasks.
  - Presentation and visualization of data mining results
    - How can a data mining system present data mining results, vividly and flexibly, so that the discovered knowledge can be easily understood and directly usable by humans? This is especially crucial if the data mining process is interactive.
- Efficiency and Scalability
  - Efficiency and scalability of data mining algorithms
    - Data mining algorithms must be efficient and scalable in order to effectively extract information from huge amounts of data in many data repositories or in dynamic data streams.
  - Parallel, distributed, stream, and incremental mining methods
    - The humongous size of many data sets, the wide distribution of data, and the computational complexity of some data mining methods are factors that motivate the development of parallel and distributed data-intensive mining algorithms.
  - Cloud computing and cluster computing,
    - Which use computers in a distributed and collaborative way to tackle very large-scale computational tasks, are also active research themes in parallel data mining.
- Diversity of data types
  - Handling complex types of data
    - Diverse applications generate a wide spectrum of new data types, from structured data such as relational and data warehouse data to semi-structured and unstructured data; from stable data repositories to dynamic data streams; from simple data objects to temporal data, biological

sequences, sensor data, spatial data, hypertext data, multimedia data, software program code, Web data, and social network data.

- Mining dynamic, networked, and global data repositories
  - Multiple sources of data are connected by the Internet and various kinds of networks, forming gigantic, distributed, and heterogeneous global information systems and networks. The discovery of knowledge from different sources of structured, semi-structured, or unstructured yet interconnected data with diverse data semantics poses great challenges to data mining.

- Data mining and society
  - Social impacts of data mining
    - With data mining penetrating our everyday lives, it is important to study the impact of data mining on society.
  - Privacy-preserving data mining
    - Data mining will help scientific discovery, business management, economy recovery, and security protection (e.g., the real-time discovery of intruders and cyber attacks).
  - Invisible data mining
    - We cannot expect everyone in society to learn and master data mining techniques. More and more systems should have data mining functions built within so that people can perform data mining or use data mining results simply by mouse clicking, without any knowledge of data mining algorithms.

8. Data Objects and Attribute Types

Data sets are made up of data objects. A data object represents an entity—in a sales database, the objects may be customers, store items, and sales; in a medical database, the objects may be patients; in a university database, the objects may be students, professors, and courses. Data objects are typically described by attributes. Data objects can also be referred to as samples, examples, instances, data points, or objects. If the data objects are stored in a database, they are data tuples. That is, the rows of a database correspond to the data objects, and the columns correspond to the attributes. In this section, we define attributes and look at the various attribute types.

What Is an Attribute: An attribute is a data field, representing a characteristic or feature of a data object. The nouns attribute, dimension, feature, and variable are often used interchangeably in the literature. **E.g: customer _ID, name, address.** The type of an attribute is determined by the set of possible values—nominal, binary, ordinal, or numeric—the attribute can have.

i. Nominal Attributes: Nominal means "relating to names." The values of a nominal attribute are symbols or names of things. Each value represents some kind of category, code, or state, and so nominal attributes are also referred to as categorical. The values do not have any meaningful order. In computer science, the values are also known as enumerations.

Example: Hair_color = {auburn, black, blond, brown, grey, red, white}
Marital status, occupation, ID numbers, zip codes

ii. Binary Attributes: A binary attribute is a nominal attribute with only two categories or states: 0 or 1, where 0 typically means that the attribute is absent, and 1 means that it is present. Binary attributes are referred to as Boolean if the two states correspond to true and false.

Example:

- Nominal attribute with only 2 states (0 and 1)
- Symmetric binary: both outcomes equally important e.g., gender
- Asymmetric binary: outcomes not equally important. e.g., medical test (positive vs. Negative) Convention: assign 1 to most important outcome (e.g., HIV positive)

iii. Ordinal Attributes: An ordinal attribute is an attribute with possible values that have a meaningful order or ranking among them, but the magnitude between successive values is not known.

Example: Values have a meaningful order (ranking) but magnitude between successive values is not known.
Size = {small, medium, large}, grades, army rankings

iv. Numeric Attributes: A numeric attribute is quantitative; that is, it is a measurable quantity, represented in integer or real values. Numeric attributes can be interval-scaled or ratio-scaled.

a. **Interval-Scaled Attributes**: Interval-scaled attributes are measured on a scale of equal-size units. The values of interval-scaled attributes have order and can be positive, 0, or negative. Thus, in addition to providing a ranking of values, such attributes allow us to compare and quantify the difference between values.

Example: Interval

- Measured on a scale of equal-sized units
- Values have order. E.g., temperature in C˚or F˚, calendar dates
- No true zero-point

b. **Ratio-Scaled Attributes**: A ratio-scaled attribute is a numeric attribute with an inherent zero-point. That is, if a measurement is ratio-scaled, we can speak of a value as being a multiple (or ratio) of another value. In addition, the values are ordered, and we can also compute the difference between values, as well as the mean, median, and mode.

Example: Ratio

- Inherent zero-point
- We can speak of values as being an order of magnitude larger than the unit of measurement (10 K˚ is twice as high as 5 K˚).
    - e.g., temperature in Kelvin, length, counts, monetary quantities

v. Discrete versus Continuous Attributes: A discrete attribute has a finite or countably infinite set of values, which may or may not be represented as integers. The attributes hair color, smoker, medical test, and drink size each have a finite number of values, and so are discrete. Note that discrete attributes may have numeric values, such as 0 and 1 for binary attributes or, the values 0 to 110 for the attribute age. If an attribute is not discrete, it is continuous. The terms numeric attribute and continuous attribute are often used interchangeably in the literature

Example: Discrete Attribute

- Has only a finite or countable infinite set of values
    - E.g., zip codes, profession, or the set of words in a collection of documents

- Sometimes, represented as integer variables
- Note: Binary attributes are a special case of discrete attributes

Example: Continuous Attribute

- Has real numbers as attribute values
  - E.g., temperature, height, or weight
- Practically, real values can only be measured and represented using a finite number of digits
- Continuous attributes are typically represented as floating-point variables

9. Basic Statistical Descriptions of Data

For data preprocessing to be successful, it is essential to have an overall picture of your data. Basic statistical descriptions can be used to identify properties of the data and highlight which data values should be treated as noise or outliers. We start with measures of central tendency, which measure the location of the middle or center of a data distribution.

(i) Measuring the Central Tendency: Mean, Median, and Mode

Mean: The most common and effective numeric measure of the "center" of a set of data is the (arithmetic) mean. Let x1, x2, : : : ,xN be a set of N values or observations, such as for some numeric attribute X, like salary. The mean of this set of values is

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}.$$

Example: Suppose we have the following values for salary (in thousands of dollars), shown in increasing order: 30, 36, 47, 50, 52, 52, 56, 60, 63, 70, 70, and 110. Using above equation we have

$\bar{x}$ = (30+36+47+50+52+52+56+60+63+70+70+110)/ 12 = 696/12 = 58
Thus, the mean salary is $58,000.

Sometimes, each value xi in a set may be associated with a weight wi for i D 1… N. The weights reflect the significance, importance, or occurrence frequency attached to their respective values. In this case, we can compute

$$\bar{x} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i} = \frac{w_1 x_1 + w_2 x_2 + \cdots + w_N x_N}{w_1 + w_2 + \cdots + w_N}.$$

This is called the weighted arithmetic mean or the weighted average. This corresponds to the built-in aggregate function, average (avg () in SQL), provided in relational database systems.

Median: Middle value if odd number of values or average of the middle two values otherwise
The median is expensive to compute when we have a large number of observations. Estimated by interpolation (for grouped data):

$$median = L_1 + \left(\frac{N/2 - (\sum freq)_l}{freq_{median}} \right) width,$$

Where $\mathbf{L_1}$ is the lower boundary of the median interval, **N** is the number of values in the entire data set, $\mathbf{(\Sigma\ Freq)_l}$ is the sum of the frequencies of all of the intervals that are lower than the median interval, $\mathbf{Freq_{median}}$ is the frequency of the median interval, and **width** is the width of the median interval.

Example: Calculate median for the given salary values 30, 36, 47, 50, 52, 52, 56, 60, 63, 70, 70, and 110 (in thousands of dollars). By convention, we assign the average of the two middlemost values as the median; that is, (52 + 56)/ 2= 54 thus, the median is $54,000.

Mode: The mode is another measure of central tendency. The mode for a set of data is the value that occurs most frequently in the set. Therefore, it can be determined for qualitative and quantitative attributes. Data sets with one, two, or three modes are respectively called unimodal, bimodal, and trimodal. In general, a data set with two or more modes is multimodal

Example: The data shown in increasing order: 30K, 36K, 47K, 50K, 52K, 52K, 56K, 60K, 63K, 70K, 70K, and 110K. The two modes are $52,000 and $70,000.

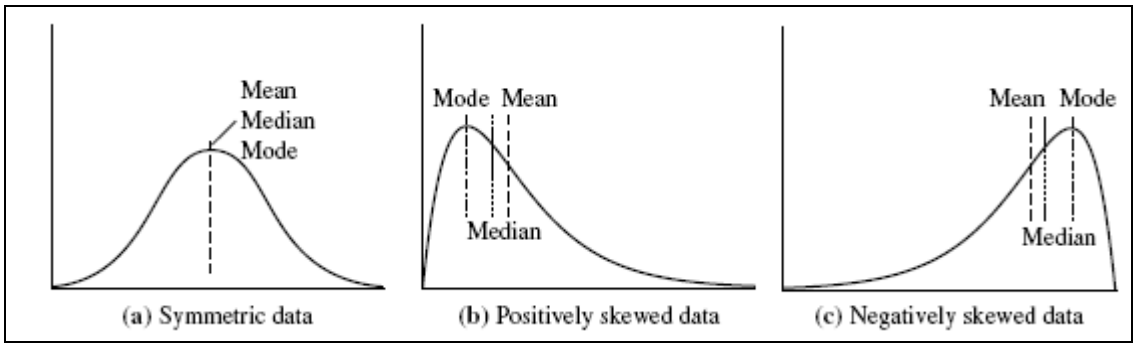
For unimodal numeric data that are moderately skewed (asymmetrical), we have the following empirical relation:

$$\text{Mean} - \text{mode} \approx 3 \times (\text{mean} - \text{median})$$

Midrange: The midrange can also be used to assess the central tendency of a numeric data set. It is the average of the largest and smallest values in the set. This measure is easy to compute using the SQL aggregate functions, max () and min ().

Example: The midrange of the data 30K, 36K, 47K, 50K, 52K, 52K, 56K, 60K, 63K, 70K, 70K, and 110K is (30,000+110,000)/2 = $70,000.

In a unimodal frequency curve with perfect symmetric data distribution, the mean, median, and mode are all at the same center value, as shown in below Figure (a). Data in most real applications are not symmetric. They may instead be either positively skewed, where the mode occurs at a value that is smaller than the median (Figure b), or negatively skewed, where the mode occurs at a value greater than the median (Figure c).



(ii) <u>Measuring the Dispersion of Data: Range, Quartiles, Variance, Standard Deviation and IQR (Inter quartile Range)</u>

Range: The range of the set is the difference between the largest and smallest values

Quartiles: The quartiles give an indication of a distribution's center, spread, and shape. The first Quartile, denoted by Q1, is the 25th percentile. It cuts off the lowest 25% of the data. The third quartile, denoted by Q3, is the 75th percentile—it cuts off the lowest 75% (or highest 25%) of the data. The second quartile is the 50th percentile. As the median, it gives the center of the data distribution.

The distance between the first and third quartiles is a simple measure of spread that gives the range covered by the middle half of the data. This distance is called the interquartile range (IQR) and is defined as

$$IQR = Q3 - Q1.$$

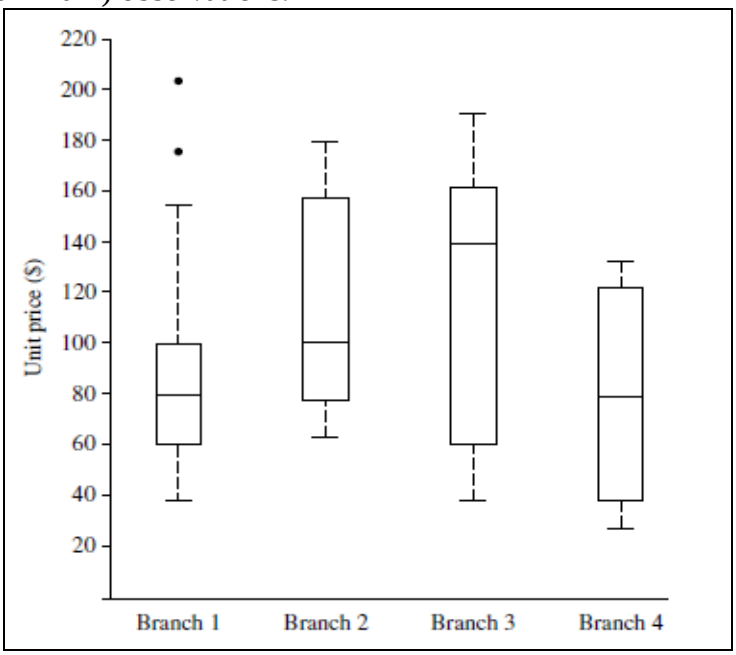
Example: The data of Example 2.6 contain 12 observations, already sorted in increasing order. Thus, the quartiles for this data are the third, sixth, and ninth values, respectively, in the sorted list. Therefore, Q1 is \$47,000 and Q3 is \$63,000. Thus, the interquartile range is IQR = 63-47 = \$16,000.

Five-Number Summary, Box plots, and Outliers

The five-number summary of a distribution consists of the median (Q2), the quartiles Q1 and Q3, and the smallest and largest individual observations, written in the order of Minimum, Q1, Median, Q3, and Maximum.

Box plots: Box plots are a popular way of visualizing a distribution. A box plot incorporates the five-number summary as follows:

- Typically, the ends of the box are at the quartiles so that the box length is the inter quartile range.
- The median is marked by a line within the box.
- Two lines (called whiskers) outside the box extend to the smallest (Minimum) and largest (Maximum) observations.



Outliers: A common rule of thumb for identifying suspected outliers is to single out values falling at least 1.5_IQR above the third quartile or below the first quartile.

Variance and Standard Deviation: Variance and standard deviation are measures of data dispersion. They indicate how spread out a data distribution is. A low standard deviation means that the data observations tend to be very close to the mean, while a high standard deviation indicates that the data are spread out over a large range of values.

The variance of N observations, x1, x2,: : : ,xN, for a numeric attribute X is

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 = \left(\frac{1}{N}\sum_{i=1}^{N}x_i^2\right) - \bar{x}^2,$$

Example: For the given salaries information, we found mean=$58,000, Calculate the variance and Standard deviation of the given dataset of size N=12.

Variance, $\sigma^2 = 1/12(30^2 + 36^2+47^2 +50^2 + 52^2+ 52^2+ 56^2+ 60^2+ 63^2+70^2+70^2+110^2) - 58^2$
$= 379.17$
Standard deviation, $\sigma = 19.47$

- Quartiles: Q1 (25th percentile), Q3 (75th percentile)
- Inter-quartile range: IQR = Q3 – Q1
- Five number summary: min, Q1, median, Q3, max
- Box plot: ends of the box are the quartiles; median is marked; add whiskers, and plot outliers individually
- Outlier: usually, a value higher/lower than 1.5 x IQR

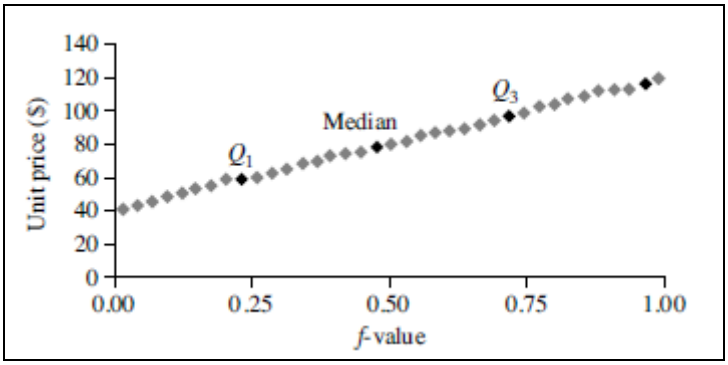
(iii) Graphic Displays of Basic Statistical Descriptions of Data

The basic graphic displays of basic statistical descriptions. These include quantile plots, quantile–quantile plots, histograms, and scatter plots. The first three of these show univariate distributions (i.e., data for one attribute), while scatter plots show bivariate distributions (i.e., involving two attributes).

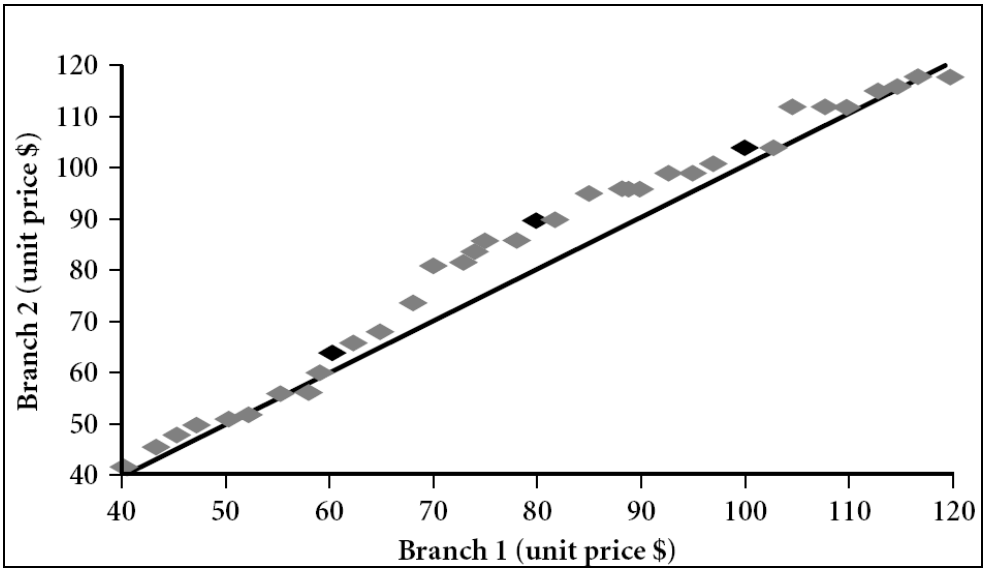
Quantile Plot: A quantile plot is a simple and effective way to have a first look at a univariate data distribution. First, it displays all of the data for the given attribute (allowing the user to assess both the overall behavior and unusual occurrences). Second, it plots quantile information**.**

- For a data x_i data sorted in increasing order, f_i indicates that approximately 100 f_i% of the data are below or equal to the value xi

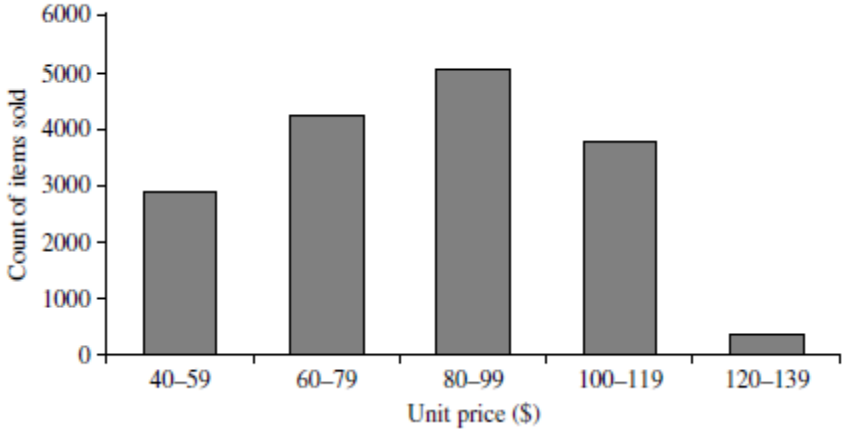
$$f_i = (i-0.5) / N$$



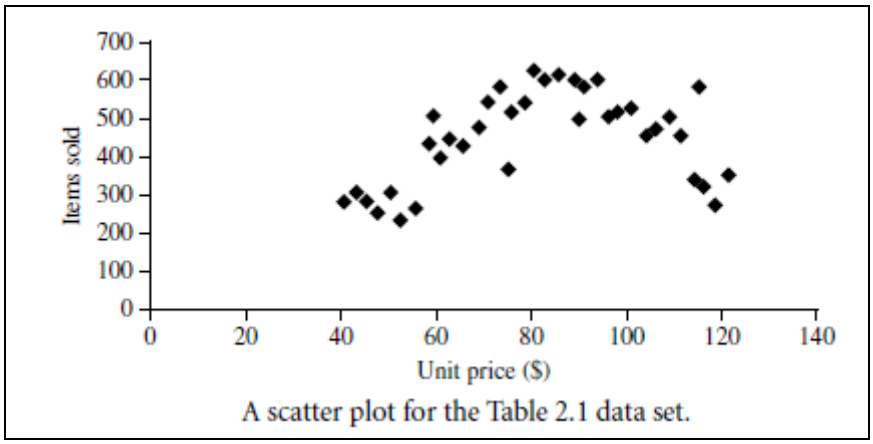
Quantile–Quantile Plot: A quantile–quantile plot, or q-q plot, graphs the quantiles of one univariate distribution against the corresponding quantiles of another. It is a powerful visualization tool in that it allows the user to view whether there is a shift in going from one distribution to another. **Example** shows unit price of items sold at Branch 1 vs. Branch 2 for each quantile. Unit prices of items sold at Branch 1 tend to be lower than those at Branch 2.



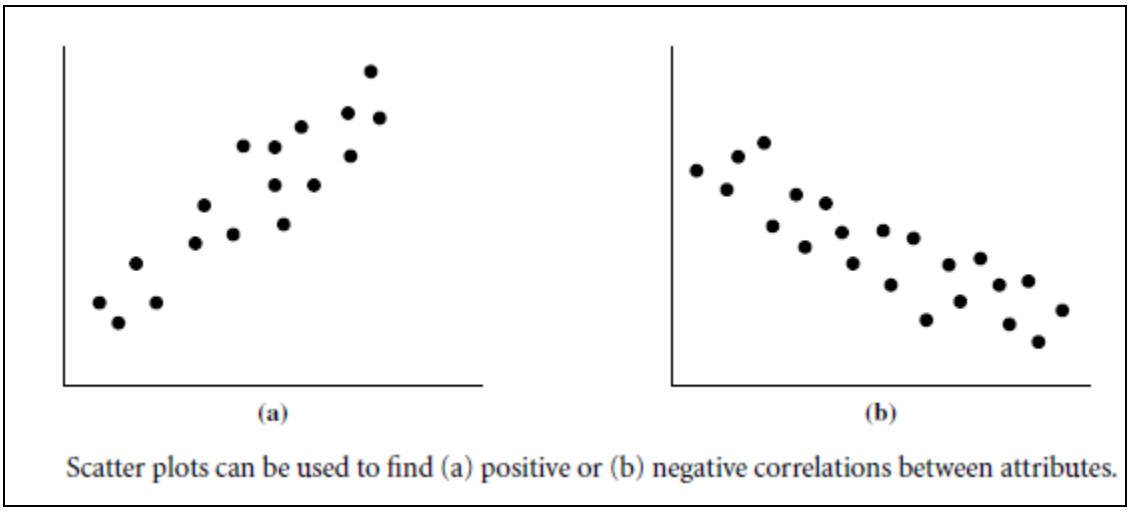
Histogram Analysis Graph display of tabulated frequencies, shown as bars If X is numeric, the term histogram is preferred. The range of values for X is partitioned into disjoint consecutive sub ranges. The sub ranges, referred to as buckets or bins, are disjoint subsets of the data distribution for X. The range of a bucket is known as the width.



Scatter Plots and Data Correlation: A scatter plot is one of the most effective graphical methods for determining if there appears to be a relationship, pattern, or trend between two numeric attributes. To construct a scatter plot, each pair of values is treated as a pair of coordinates in an algebraic sense and plotted as points in the plane.



A scatter plot for the Table 2.1 data set.



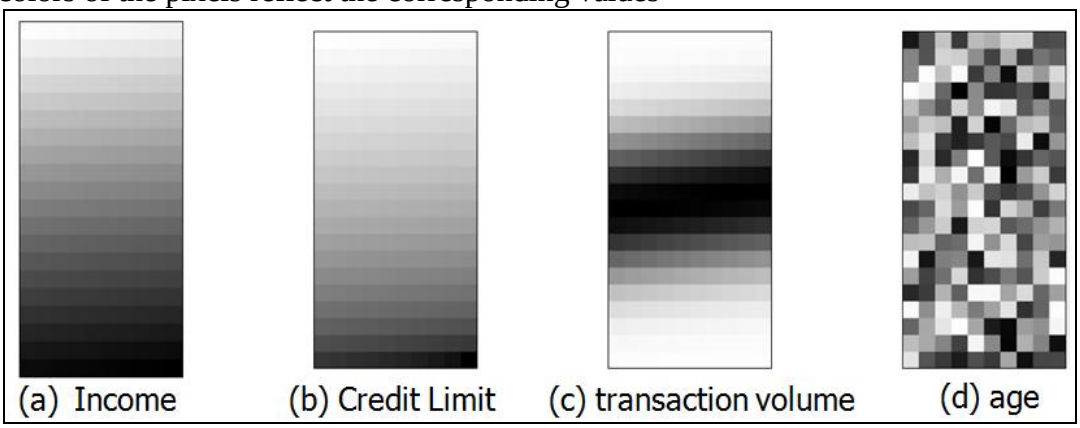
Scatter plots can be used to find (a) positive or (b) negative correlations between attributes.

10. Data Visualization

Why data visualization?

- Gain insight into an information space by mapping data onto graphical primitives
- Provide qualitative overview of large data sets
- Search for patterns, trends, structure, irregularities, relationships among data
- Help find interesting regions and suitable parameters for further quantitative analysis
- Provide a visual proof of computer representations derived

- Categorization of visualization methods:
  - Pixel-oriented visualization techniques
    - For a data set of m dimensions, create m windows on the screen, one for each dimension
    - The m dimension values of a record are mapped to m pixels at the corresponding positions in the windows
    - The colors of the pixels reflect the corresponding values



  - Geometric projection visualization techniques
    - Visualization of geometric transformations and projections of the data
    - Methods
      i. Direct visualization
      ii. Scatterplot and scatterplot matrices
      iii. Landscapes
      iv. Projection pursuit technique: Help users find meaningful projections of multidimensional data
      v. Prosection views
      vi. Hyperslice
      vii. Parallel coordinates
  - Icon-based visualization techniques
  - Hierarchical visualization techniques
  - Visualizing complex data and relations

11. Measuring Data Similarity and Dissimilarity

- **Similarity**
  - Numerical measure of how alike two data objects are
  - Value is higher when objects are more alike
  - Often falls in the range [0,1]
- **Dissimilarity** (e.g., distance)
  - Numerical measure of how different two data objects are
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies
- **Proximity** refers to a similarity or dissimilarity

Data Matrix and Dissimilarity Matrix

- Data matrix
  - n data points with p dimensions
  - Two modes

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

- Dissimilarity matrix
  - n data points, but registers only the distance
  - A triangular matrix
  - Single mode

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

- **Proximity Measure for Nominal Attributes**
  - Can take 2 or more states, e.g., red, yellow, blue, green (generalization of a binary attribute)
  - **Method 1**: Simple matching *m*: # of matches, *p*: total # of variables

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

  - **Method 2:** Use a large number of binary attributes
  - creating a new binary attribute for each of the *M* nominal states

- **Proximity Measure for Binary Attributes**
  - A contingency table for binary data Object *i*

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

  - Distance measure for symmetric binary variables:

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

  - Distance measure for asymmetric binary variables:

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

  - Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

$$sim_{Jaccard}(i, j) = \frac{q}{q + r + s}$$

  - Note: Jaccard coefficient is the same as "coherence":

$$coherence(i, j) = \frac{sup(i, j)}{sup(i) + sup(j) - sup(i, j)} = \frac{q}{(q + r) + (q + s) - q}$$

- **Dissimilarity between Binary Variables:** Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - **Gender is a symmetric attribute**
  - **The remaining attributes are asymmetric binary**
  - **Let the values Y and P be 1, and the value N 0**

$$d(jack, mary) = \frac{0 + 1}{2 + 0 + 1} = 0.33$$

$$d(jack, jim) = \frac{1 + 1}{1 + 1 + 1} = 0.67$$

$$d(jim, mary) = \frac{1 + 2}{1 + 1 + 2} = 0.75$$

Dissimilarity of Numeric Data:

1. Euclidean Distance

The most popular distance measure is Euclidean distance. The Euclidean distance between objects i and j is defined as

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

2. Manhattan distance

Another well-known measure is the Manhattan (or city block) distance, named so because it is the distance in blocks between any two points in a city (such as 2 blocks down and 3 blocks over for a total of 5 blocks). It is defined as

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

Both the Euclidean and the Manhattan distance satisfy the following mathematical properties: **Non-negativity, Identity of indiscernible, Symmetry, Triangle inequality**

Example: Euclidean distance and Manhattan distance. Let x1 = (1, 2) and x2 = (3, 5). The Euclidean distance between the two is $\sqrt{2^2 + 3^2}$ =3.61 The Manhattan distance between the two is 2+3 = 5.

3. Minkowski distance

Minkowski distance is a generalization of the Euclidean and Manhattan distances. It is defined as

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h},$$

where h is a real number such that h >= 1

Cosine Similarity

A document can be represented by thousands of attributes, each recording the frequency of a particular word (such as keywords) or phrase in the document.

Document	*team*	*coach*	*hockey*	*baseball*	*soccer*	*penalty*	*score*	*win*	*loss*	*season*
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

- Other vector objects: gene features in micro-arrays
- Applications: information retrieval, biologic taxonomy, gene feature mapping,
- Cosine measure: If d1 and d2 are two vectors (e.g., term-frequency vectors), then
  $\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||$,
  where $\bullet$ indicates vector dot product, ||d||: the length of vector d

Cosine similarity between two term-frequency vectors. Suppose that x and y are the first two term-frequency vectors in Table 2.5. That is, $x = (5,0,3,0,2,0,0,2,0,0)$ and $y = (3,0,2,0,1,1,0,1,0,1)$. How similar are x and y? Using Eq. (2.23) to compute the cosine similarity between the two vectors, we get:

$$x^t \cdot y = 5 \times 3 + 0 \times 0 + 3 \times 2 + 0 \times 0 + 2 \times 1 + 0 \times 1 + 0 \times 0 + 2 \times 1 + 0 \times 0 + 0 \times 1 = 25$$

$$||x|| = \sqrt{5^2 + 0^2 + 3^2 + 0^2 + 2^2 + 0^2 + 0^2 + 2^2 + 0^2 + 0^2} = 6.48$$

$$||y|| = \sqrt{3^2 + 0^2 + 2^2 + 0^2 + 1^2 + 1^2 + 0^2 + 1^2 + 0^2 + 1^2} = 4.12$$

$$sim(x, y) = 0.94$$

Frequently Asked Questions

Short Answer Questions

1. What is Data Mining and Why it is needed
2. Which Technologies Are Used?
3. Data Objects and Attribute Types,
4. Basic Statistical Descriptions of Data,

Long Answer Questions

1. Explain about Data Mining Functionalities—What Kinds of Patterns Can Be Mined? Are All of the Patterns Interesting?
2. Write short notes on Classification of Data Mining Systems
3. Major Issues in Data Mining.
4. What the various Data Visualization techniques are used in Data mining, explain in detail
5. Describe about Measuring Data Similarity and Dissimilarity of data object

Exercise Problems

1. Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.
   (a) What is the mean of the data? What is the median?
   (b) What is the mode of the data? Comment on the data's modality (i.e., bimodal, trimodal, etc.).
   (c) What is the midrange of the data?
   (d) Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?
   (e) Give the five-number summary of the data.
   (f) Show a boxplot of the data.
   (g) How is a quantile–quantile plot different from a quantile plot?

2. Given two objects represented by the tuples (22, 1, 42, 10) and (20, 0, 36, 8):
   (a) Compute the Euclidean distance between the two objects.
   (b) Compute the Manhattan distance between the two objects.
   (c) Compute the Minkowski distance between the two objects, using q D 3.
   (d) Compute the supremum distance between the two objects.

3. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

age	23	23	27	27	39	41	47	49	50
%fat	9.5	26.5	7.8	17.8	31.4	25.9	27.4	27.2	31.2
age	52	54	54	56	57	58	58	60	61
%fat	34.6	42.5	28.8	33.4	30.2	34.1	32.9	41.2	35.7

(a) Calculate the mean, median, and standard deviation of age and %fat.
(b) Draw the boxplots for age and %fat.
(c) Draw a scatter plot and a q-q plot based on these two variables.

UNIT –II:

Data Pre-processing: An Overview, Data Cleaning, Data Integration, Data Reduction, Data Transformation and Data Discretization

1. Data Pre-processing: An Overview

- **Data Quality: Why Preprocess the Data?**

There are many factors comprising data quality, including accuracy, completeness, consistency, timeliness, believability, and interpretability. Measures for data quality: A multidimensional view

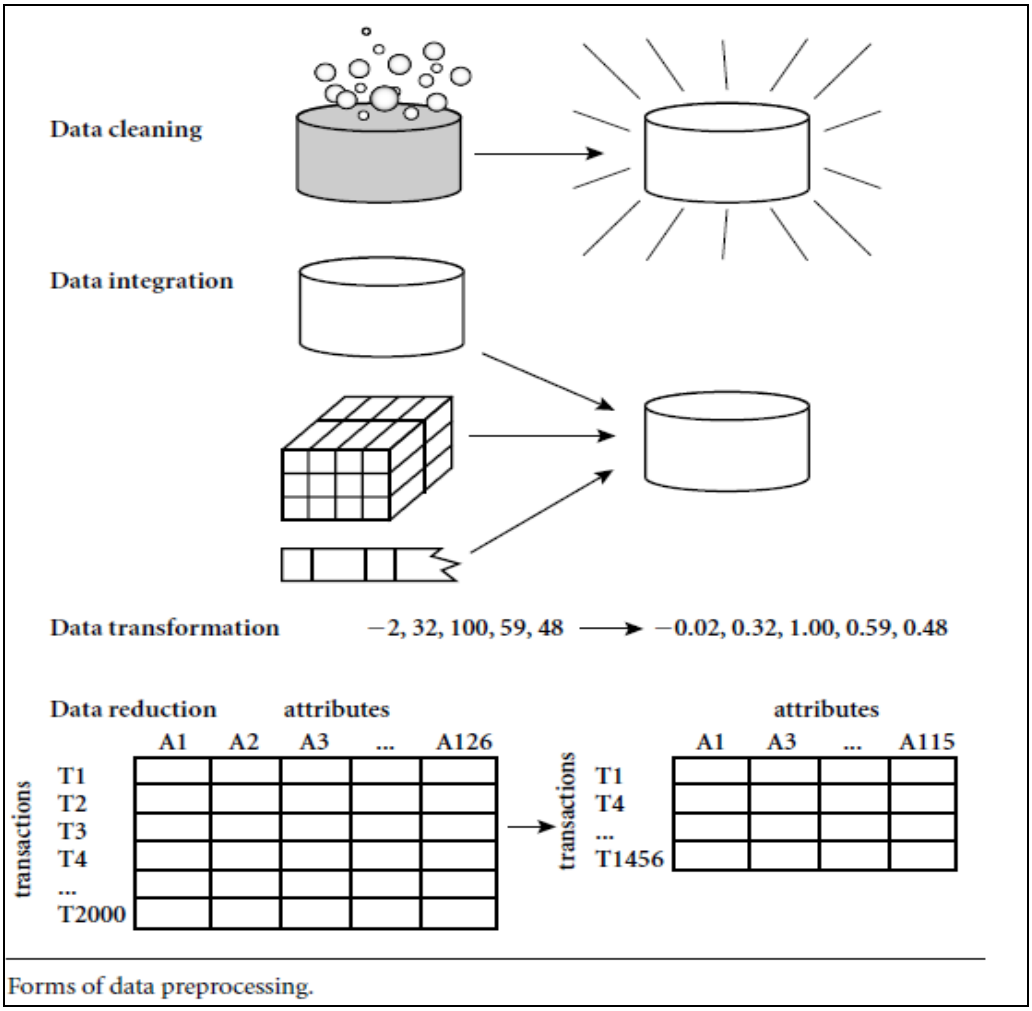
- Accuracy: correct or wrong, accurate or not
- Completeness: not recorded, unavailable,
- Consistency: some modified but some not, dangling,
- Timeliness: timely update?
- Believability: how trustable the data are correct?
- Interpretability: how easily the data can be understood?

- **Major Tasks in Data Preprocessing**

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
Data integration: Integration of multiple databases, data cubes, or files
Data reduction: Data reduction obtains a reduced representation of the data set that is much smaller in volume, yet produces the same analytical results. Data reduction strategies include dimensionality reduction and numerosity reduction.
Data transformation and data discretization: Normalization, Concept hierarchy generation

Why Pre-process the Data:-Today's real-world databases are highly susceptible to noise, and consist of missing, and inconsistent data due to their huge size. Data preprocessing is done to improve the quality of the data. Preprocessed data improve the efficiency and ease of the mining process. There are a number of data preprocessing techniques. They are

1. Data cleaning can be applied to remove noise and correct inconsistencies in the data.
2. Data integration merges data from multiple sources into a single data store, such as a data warehouse or a data cube.
3. Data transformations, such as normalization, may be applied. Normalization may improve the accuracy and efficiency of mining algorithms.
4. Data reduction can reduce the data size by aggregating, eliminating redundant features, or clustering.



Forms of data preprocessing.

2. Data Cleaning:

Data in the Real World Is Dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, transmission error

- Incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data.
  - **Ex: Occupation=“ ” (missing data)**
- Noisy: containing noise, errors, or outliers.
  - **Ex: e.g., Salary=“−10” (an error)**
- Inconsistent: containing discrepancies in codes or names, Example,
  - Age=“42”, Birthday=“03/07/2010”
  - Was rating “1, 2, 3”, now rating “A, B, C”
  - discrepancy between duplicate records
- Intentional (Ex: Disguised missing data)
  - Jan. 1 as everyone’s birthday?

How to Handle Missing Data?

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.
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 ∞.
4. Use a measure of central tendency for the attribute (e.g., the mean or median) to fill in the missing value.
5. **Use the attribute mean or median for all samples belonging to the same class as the given tuple:** For example, if classifying customers according to credit risk, we may replace the missing value with the mean 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:-

A data is said to be noisy if its attribute values are invalid and incorrect.

Noise is a random error or variance in a measured variable. "Smooth" out the data to remove noise. Some of the data smoothing techniques that are commonly used are.

1. Binning methods: - Binning methods smooth a sorted data value by consulting the neighborhood", that is values around it. In Binning method the sorted values are distributed into a number of 'buckets', or bins. Because binning methods consult the neighborhood of values, they perform local smoothing.

Commonly used binning methods are

a) *Smoothing by bin means:* - In smoothing by bin means, each value in a bin is replaced by the mean value of the bin.
b) *Smoothing by bin median:* - In smoothing by bin medians, each bin value is replaced by the bin median.
c) *Smoothing by bin means:* - 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.

Example: - Smooth out the following prices 21, 8, 28, 4, 34, 21, 15, 25, 24.

Data for price are first sorted and then partitioned into equi depth bins of depth 3.

Sorted data for price (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

Partition into (equi-width) 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 median:

Bin 1: 8, 8, 8
Bin 2: 21, 21, 21
Bin 3: 28, 28, 28

Smoothing by bin boundaries:

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 34

Figure 3.2: Binning methods for data smoothing.

2. Clustering: - Outliers may be detected by clustering. Similar values are organized into group's clusters. Values which fall outside all clusters may be considered outliers.

3. Combined computer and human inspection: - Outliers may be identified through a combination of computer and human inspection. This is much faster than having to manually search through the entire database. The garbage patterns can then be removed from the database.

4. Regression: - Data can be smoothed by using regression. Linear regression involves finding the best line to fit two variables, so that one variable can be used to predict the other. Multiple linear regressions are an extension of linear regression, where more than two variables are involved to predict the unknown value.

5. Inconsistent data: - Data inconsistencies may be corrected manually using external references. For example, errors made at data entry may be corrected by performing a

Paper trace. Knowledge engineering tools may also be used to detect the violation of known data constraints.

3. Data integration:-Data integration combines data from multiple sources into a single data store, such as large database or data warehouse. Major Issues that are to be considered during data integration are

Entity identification problem: *- Sometimes* customer_id in one database and cust_number in another refer to the same entity. Data analyst or computer decides whether they both refer to the same entity by examining the metadata of the data warehouse. Metadata is data about the data. Such metadata can be used to avoid errors in schema integration.

Redundancy: -

Redundancy is another important issue in data integration. An attribute (such as annual revenue, for instance) may be redundant if it can be "derived" from another attribute or set of attributes. Inconsistencies in attribute or dimension naming can also cause redundancies in the resulting data set.

Some redundancies can be detected by correlation analysis. Given two attributes, such analysis can measure how strongly one attribute implies the other, based on the available data. For nominal data, we use the $\chi 2$ (chi-square) test.

$\chi 2$ Correlation Test for Nominal Data

For nominal data, a correlation relationship between two attributes, A and B, can be discovered by a chi-square test. Suppose A has c distinct values, namely a1, a2…ac .

B has r distinct values, namely b1, b2…br. The data tuples described by A and B can be shown as a contingency table, with the c values of A making up the columns and the r values of B making up the rows. The $\chi 2$ value (also known as the Pearson 2 statistic) is computed as

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}},$$

Where Oij is the observed frequency (i.e., actual count) of the joint event (Ai, Bj) and eij is the expected frequency of (Ai, Bj), which can be computed as

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{n},$$

Where n is the number of data tuples, count (A = ai) is the number of tuples having value ai for A, and count (B = bj) is the number of tuples having value bj for B.

Example: Correlation analysis of nominal attributes using $\chi 2$:
Suppose that a group of 1500 people was surveyed. The gender of each person was noted. Each person was polled as to whether his or her preferred type of reading material was fiction or nonfiction. Thus, we have two attributes, gender and preferred reading. The observed frequency (or count) of each possible joint event is summarized in the contingency table shown in Table

Example 2.1's 2 × 2 Contingency Table Data

	male	*female*	*Total*
fiction	250 (90)	200 (360)	450
non_fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

Note: Are *gender* and *preferred_reading* correlated?

X2 (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

For this 2 X 2 table, the degrees of freedom are (2-1) X (2-1) = 1. For 1 degree of freedom, the 2 value needed to reject the hypothesis at the 0.001 significance level is 10.828. Since our computed value is above this, we can reject the hypothesis that gender and preferred reading are independent and conclude that the two attributes are (strongly) correlated for the given group of people.

For numeric attributes, we can evaluate the correlation between two attributes, A and B, by computing the correlation coefficient (also known as Pearson's product moment coefficient, named after its inventor, Karl Pearson). This is

$$r_{A,B} = \frac{\sum_{i=1}^{n}(a_i - \overline{A})(b_i - \overline{B})}{(n-1)\sigma_A\sigma_B} = \frac{\sum_{i=1}^{n}(a_i b_i) - n\overline{A}\,\overline{B}}{(n-1)\sigma_A\sigma_B}$$

where n is the number of tuples, ai and bi are the respective means of A and B, σA and σB are the respective standard deviation of A and B, and Σ(aibi) is the sum of the AB cross-product.

Covariance (Numeric Data)

In probability theory and statistics, correlation and covariance are two similar measures for assessing how much two attributes change together. Consider two numeric attributes A and B, and a set of n observations {(a1, b1)... (an, bn)}. The mean values of A and B, respectively, are also known as the expected values on A and B, that is,

$$E(A) = \bar{A} = \frac{\sum_{i=1}^{n} a_i}{n}$$

and

$$E(B) = \bar{B} = \frac{\sum_{i=1}^{n} b_i}{n}.$$

The **covariance** between A and B is defined as

$$Cov(A, B) = E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^{n} (a_i - \bar{A})(b_i - \bar{B})}{n}.$$

If we compare equation for rA,B (correlation coefficient) with Equation for covariance, we see that

$$r_{A,B} = \frac{Cov(A, B)}{\sigma_A \sigma_B},$$

where σ_A and σ_B are the standard deviations of A and B, respectively. It can also be shown that

$$Cov(A, B) = E(A \cdot B) - \bar{A}\bar{B}.$$

-

4. Data transformation:-

A function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value can be identified with one of the new values

- Methods
  - Smoothing: Remove noise from data
  - Attribute/feature construction
    - New attributes constructed from the given ones
  - Aggregation: Summarization, data cube construction
  - Normalization: Scaled to fall within a smaller, specified range
    - min-max normalization
    - z-score normalization
    - normalization by decimal scaling
  - Discretization: Concept hierarchy climbing

Normalization

1. **Min-max normalization**: Performs a linear transformation on the original data. Suppose that min_A and max_A are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, v_i , of A to v_i' in the range [new_min_A, new_max_A] by computing

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

Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to

$$\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$$

2. Z-score normalization (or zero-mean normalization), the values for an attribute, A, are normalized based on the mean (i.e., average) and standard deviation of A. A value, vi, of A is normalized to vi' by computing

$$v' = \frac{v - \mu_A}{\sigma_A}$$

Ex. Suppose that the mean and standard deviation of the values for the attribute income are \$54,000 and \$16,000, respectively. With z-score normalization, a value of \$73,600 for income is transformed to

$$\frac{73{,}600 - 54{,}000}{16{,}000} = 1.225$$

3. 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, vi , of A is normalized to v0i by computing. Where j is the smallest integer such that Max(|v'|) < 1

$$v' = \frac{v}{10^j}$$

Ex: Suppose that the recorded values of A range from -986 to 917. The maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1000 (i.e., j D 3) so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

Discretization: Three types of attributes

- Nominal—values from an unordered set, e.g., color, profession
- Ordinal—values from an ordered set, e.g., military or academic rank
- Numeric—real numbers, e.g., integer or real numbers

Discretization: Divide the range of a continuous attribute into intervals

- Interval labels can then be used to replace actual data values
- Reduce data size by discretization
- Supervised vs. unsupervised
- Split (top-down) vs. merge (bottom-up)
- Discretization can be performed recursively on an attribute
- Prepare for further analysis, e.g., classification

Data Discretization Methods

Typical methods: All the methods can be applied recursively

- Binning
  - Top-down split, unsupervised
- Histogram analysis
  - Top-down split, unsupervised
- Clustering analysis (unsupervised, top-down split or bottom-up merge)
- Decision-tree analysis (supervised, top-down split)
- Correlation (e.g., □ 2) analysis (unsupervised, bottom-up merge)

Concept Hierarchy Generation

- Concept hierarchy organizes concepts (i.e., attribute values) hierarchically and is usually associated with each dimension in a data warehouse
- Concept hierarchies facilitate drilling and rolling in data warehouses to view data in multiple granularity
- Concept hierarchy formation: Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for age) by higher level concepts (such as youth, adult, or senior)
- Concept hierarchies can be explicitly specified by domain experts and/or data warehouse designers
- Concept hierarchy can be automatically formed for both numeric and nominal data. For numeric data, use discretization methods shown.

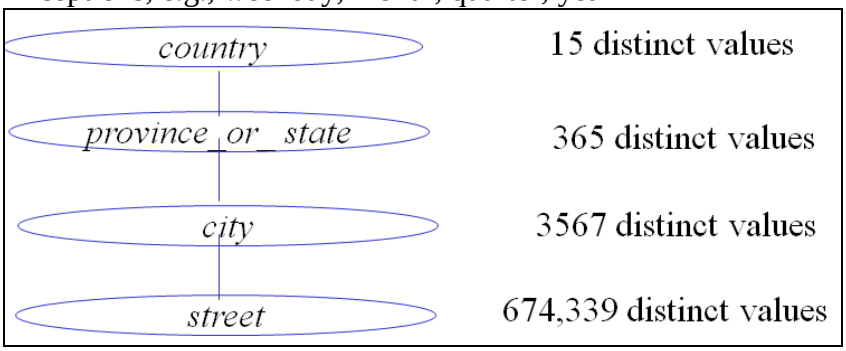
Concept Hierarchy Generation for Nominal Data

- Specification of a partial/total ordering of attributes explicitly at the schema level by users or experts
  - street < city < state < country
- Specification of a hierarchy for a set of values by explicit data grouping
  - {Urbana, Champaign, Chicago} < Illinois
- Specification of only a partial set of attributes
  - E.g., only street < city, not others
- Automatic generation of hierarchies (or attribute levels) by the analysis of the number of distinct values
  - E.g., for a set of attributes: {street, city, state, country}

Automatic Concept Hierarchy Generation

Some hierarchies can be automatically generated based on the analysis of the number of distinct values per attribute in the data set

- The attribute with the most distinct values is placed at the lowest level of the hierarchy
- Exceptions, e.g., weekday, month, quarter, year



4. Data Reduction

Obtain a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results. Why data reduction? — A database/data warehouse may store terabytes of data. Complex data analysis may take a very long time to run on the complete data set. The following Data reduction strategies can be applied on data.

- Dimensionality reduction, e.g., remove unimportant attributes
  - Wavelet transforms
  - Principal Components Analysis (PCA)
  - Feature subset selection, feature creation
- Numerosity reduction (some simply call it: Data Reduction)
  - Regression and Log-Linear Models
  - Histograms, clustering, sampling
  - Data cube aggregation
- Data compression

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. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results. Strategies for data reduction include the following.

1. Data cube aggregation	2. Dimension reduction	3. Data compression
4. Numerosity reduction	5. Regression and log-linear models	6. Histograms
7. Clustering	8. Sampling	

1. Data cube aggregation: - In Data cube aggregation, aggregation operations are applied to the data in the construction of a data cube.

Suppose AllElectronics have their data as sales per quarter for the years 1997 to 1999 as shown in fig(a) . But the management are interested in the annual sales (total per year), rather than the total per quarter. Thus the data can be aggregated so that the resulting data summarize the total

sales per year instead of per quarter. This aggregation is illustrated in Figure b. The resulting data set is smaller in volume, without loss of information necessary for the analysis task.

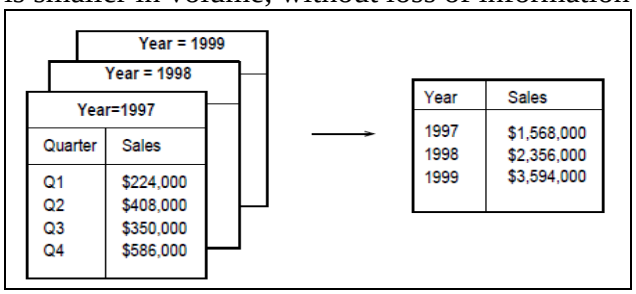


Fig (a)

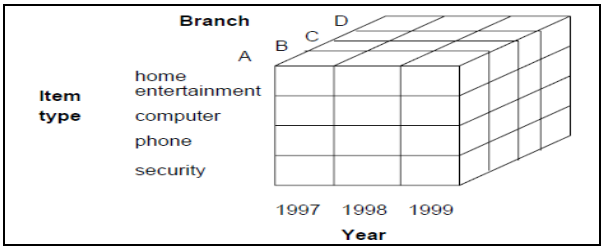


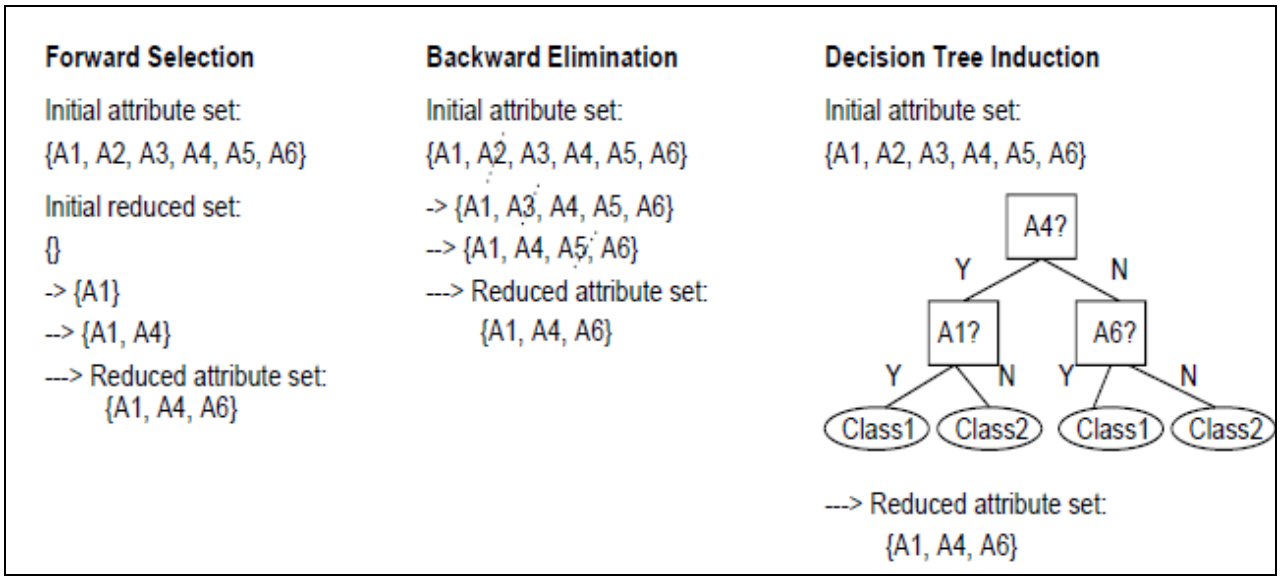
Fig (b)

Data cubes store multidimensional aggregated information. Data cube consists of many cells and each cell holds an aggregate data value at multiple levels of abstraction. 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. For the sales data represented in the cube , the apex cuboid would give one total i.e. the total sales for all three years, for all item types, and for all branches. 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.

2. Dimension reduction:- In Dimension reduction, irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed. Data sets for analysis may contain hundreds of attributes, many of which may be irrelevant to the mining task, or redundant. In analyzing customer music interest attributes such as the customer's telephone number are likely to be irrelevant and attributes such as age or music taste become relevant attributes.

The 'best' (and 'worst') attributes are typically selected using greedy methods. Some of the methods of attribute subset selection are



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

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

c. Combination forward selection and backward elimination:- The step-wise forward selection and backward elimination methods can be combined, where at each step one selects the best attribute and removes the worst from among the remaining attributes.
d. Decision tree induction:- Decision tree induction constructs a flow-chart-like structure where each internal (non-leaf) 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. When decision tree induction is used for attribute subset selection, a tree is constructed from the given data. All attributes that do not appear in the tree are assumed to be irrelevant. The set of attributes appearing in the tree form the reduced subset of attributes.

3. Data compression:- In Data compression encoding mechanisms are used to 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 compression technique used is called lossless. If, instead, we can reconstruct only an approximation of the original data, then the data compression technique is called lossy. Two popular and effective methods of lossy data compression are

1. Wavelet transforms and 2. Principal components analysis

a. Wavelet transforms: - The discrete wavelet transform (DWT) is a linear signal processing technique that, when applied to a data vector D, transforms it to a numerically different vector, D0, of wavelet coefficients. This technique is useful for data reduction if the wavelet transformed data are of the same length as the original data.
The DWT is closely related to the discrete Fourier transform (DFT), a signal processing technique involving sines and cosines. In general the DWT achieves better lossy compression. That is, if the same number of coefficients are retained for a DWT and a DFT of a given data vector, the DWT version will provide a more accurate approximation of the original data.
Popular wavelet transforms include the Daubechies-4 and the Daubechies-6 transforms. Wavelet transforms can be applied to multidimensional data, such as a data cube. Wavelet transforms give good results on sparse or skewed data, and data with ordered attributes.
There is only one DFT, yet there are several DWTs. The general algorithm for a discrete wavelet transform is as follows.
1. The length, L, of the input data vector must be an integer power of two. This condition can be met by padding the data vector with zeros, as necessary.
2. Each transform involves applying two functions. The first applies some data smoothing, such as a sum or weighted average. The second performs a weighted difference.
3. The two functions are applied to pairs of the input data, resulting in two sets of data of length L=2. In general, these respectively represent a smoothed version of the input data, and the high-frequency content of it.
4. The two functions are recursively applied to the sets of data obtained in the previous loop, until the resulting data sets obtained are of desired length.
5. A selection of values from the data sets obtained in the above iterations are designated the wavelet coefficients of the transformed data.

b.Principal components analysis:- Principal components analysis is a method of data compression. PCA can be used as a form of dimensionality reduction. However, unlike attribute subset selection, which reduces the attribute set size by retaining a subset of the initial set of attributes, PCA "combines" the essence of attributes by creating an alternative, smaller set of variables. The initial data can then be projected onto this smaller set.

PCA can be applied to ordered and unordered attributes, and can handle sparse data and skewed data. Multidimensional data of more than two dimensions can be handled by reducing the problem to two dimensions.

4. Numerosity reduction

Can we reduce the data volume by choosing alternative, `smaller' forms of data representation?" Techniques of numerosity reduction can indeed be applied for this purpose. These techniques may be parametric or non-parametric.

For parametric methods, a model is used to estimate the data, so that typically only the data parameters need be stored, instead of the actual data. (Outliers may also be stored). Log-linear models, which estimate discrete multidimensional probability distributions, are an example. Non-parametric methods for storing reduced representations of the data include histograms, clustering, and sampling.

Let's have a look at each of the numerosity reduction techniques mentioned above.

5. Regression and log-linear models

Regression and log-linear models can be used to approximate the given data. In linear regression, the data are modeled to a straight line. For example, a random variable, Y (called a response variable), can be modeled as a linear function of another random variable, X (called a predictor variable), with the equation $Y = mX+c$

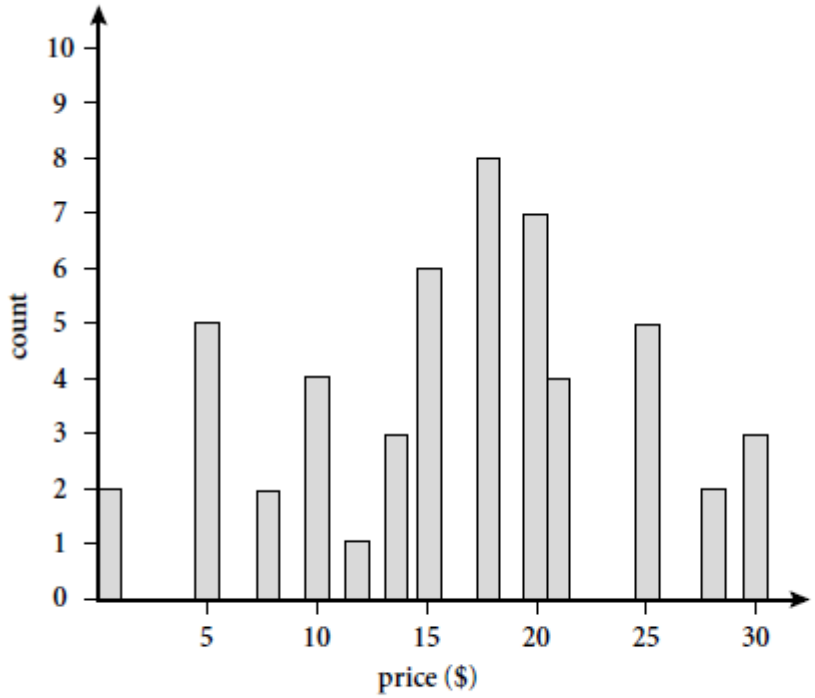
Multiple linear regression is an extension of linear regression allowing a response variable Y to be modeled as a linear function of a multidimensional feature vector. Log-linear models approximate discrete multidimensional probability distributions. The method can be used to

Estimate the probability of each cell in a base cuboid for a set of discretized attributes, based on the smaller cuboids making up the data cube lattice. This allows higher order data cubes to be constructed from lower order ones.

6. Histograms

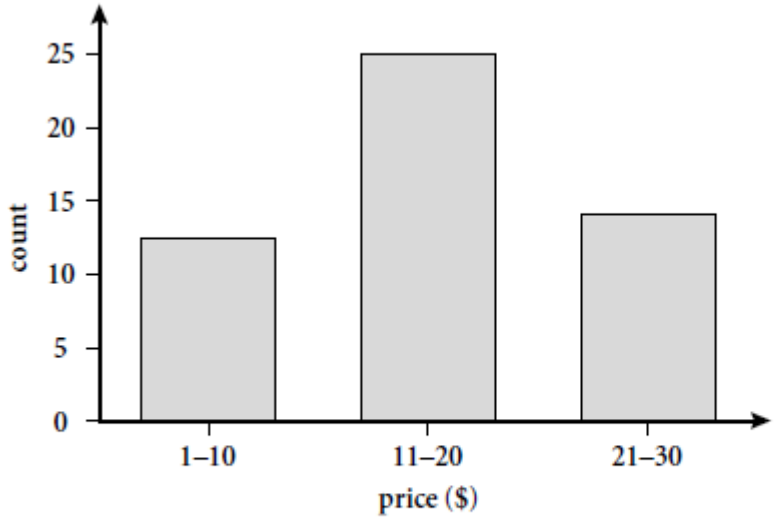
Histograms use binning to approximate data distributions and are a popular form of data reduction. A histogram

For an attribute A partitions the data distribution of A into disjoint subsets, or buckets. The buckets are displayed on a horizontal axis, while the height (and area) of a bucket typically represents the average frequency of the values represented by the bucket. If each bucket represents only a single attribute-value/frequency pair, the buckets are called singleton buckets. Often, buckets instead represent continuous ranges for the given attribute.



A histogram for *price* using singleton buckets—each bucket represents one price-value/frequency pair.

Example The following data are a list of prices of commonly sold items at AllElectronics (rounded to the nearest dollar). 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.



An equal-width histogram for *price*, where values are aggregated so that each bucket has a uniform width of $10.

How are the buckets determined and the attribute values partitioned? There are several partitioning rules, including the following.

1. Equi-width: In an equi-width histogram, the width of each bucket range is constant (such as the width of $10 for the buckets in Figure 3.8).

2. Equi-depth (or equi-height): In an equi-depth histogram, the buckets are created so that, roughly, the frequency of each bucket is constant (that is, each bucket contains roughly the same number of contiguous data samples).

3. V-Optimal: If we consider all of the possible histograms for a given number of buckets, the V-optimal histogram is the one with the least variance. Histogram variance is a weighted sum of the original values that each bucket represents, where bucket weight is equal to the number of values in the bucket.

4. MaxDiff: In a MaxDiff histogram, we consider the difference between each pair of adjacent values. A bucket boundary is established between each pair for pairs having the 1 largest differences, where _ is user-specified.

V-Optimal and MaxDiff_ histograms tend to be the most accurate and practical. Histograms are highly effective

At approximating both sparse and dense data, as well as highly skewed, and uniform data.

7. Clustering

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. The “quality" of a cluster may be represented by its diameter, the maximum distance between any two objects in the cluster

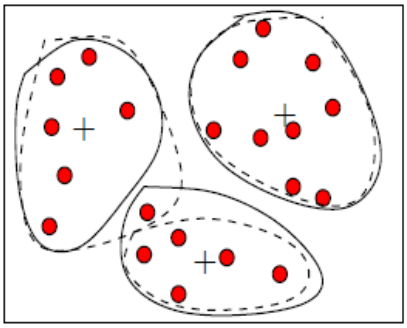


Figure 3.9: A 2-D plot of customer data with respect to customer locations in a city, showing three data clusters. Each cluster centroid is marked with a “+”.

8. Sampling:

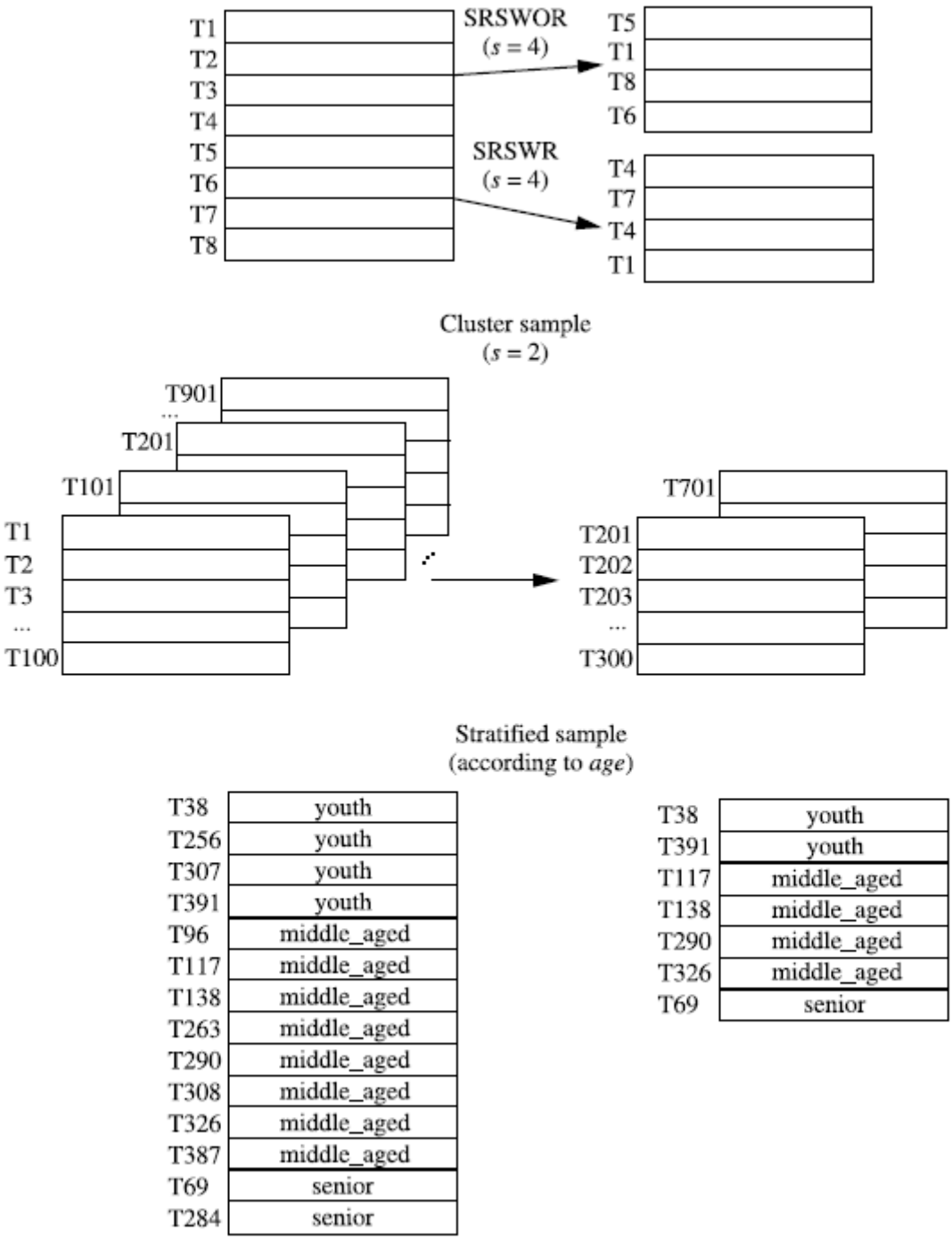
Sampling can be used as a data reduction technique since it allows a large data set to be represented by a much smaller random sample (or subset) of the data. Suppose that a large data set, D, contains N tuples. Let's have a look at some possible samples for D.

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. For example, tuples in a database are usually retrieved a page at a time, so
that each page can be considered a cluster. A reduced data representation can be obtained by applying, say, SRSWOR to the pages, resulting in a cluster sample of the tuples.
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. For example, a stratified sample may be obtained from customer data, where stratum is created for each customer age group. In this way, the age group having the smallest number of customers will
be sure to be represented.



Sampling can be used for data reduction.

Frequently Asked Questions

Short Answer Questions

1. Justify the role of pre-processing?
2. Explain about Binning Process?
3. Why Normalization is used in data transformation?
4. Explain about z-Score normalizations?

5. Define Concept Hierarchy?
6. Explain about role of Principal Component of Analysis (PCA)
7. Write about Wavelet transforms?

Long Answer Questions

1. Write a short notes on Data Cleaning,
2. Briefly explain Data Integration approaches used in data mining
3. Explain in detail about Transformation techniques used in data mining?
4. Explain about Data Reduction techniques?
5. Define and explain about Data Discretization and Concept Hierarchy Generation.

Exercise Problems

1. Suppose a group of 12 sales price records has been sorted as follows:
   5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215.
   Partition them into three bins by each of the following methods:
   (a) equal-frequency (equal-depth) partitioning
   (b) equal-width partitioning
   (c) Smoothing by bin means, bin median and bin boundaries
2. For the attribute age: 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.
   (a) Use min-max normalization to transform the value 35 for age onto the range [0.0, 1.0].
   (b) Use z-score normalization to transform the value 35 for age, where the standard deviation of age is 12.94 years.
   (c) Use normalization by decimal scaling to transform the value 35 for age.
   (d) Comment on which method you would prefer to use for the given data, giving reasons as to why.
3. . For the attribute age: 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.
   (a) Plot an equal-width histogram of width 10.
   (b) Sketch examples of each of the following sampling techniques: SRSWOR, SRSWR, Cluster sampling, and stratified sampling. Use samples of size 5 and the strata "Youth," "middle-aged," and "senior."

UNIT - III

Classification: Basic Concepts, General Approach to solving a classification problem, Decision Tree Induction: Working of Decision Tree, building a decision tree, methods for expressing an attribute test conditions, measures for selecting the best split, Algorithm for decision tree induction.

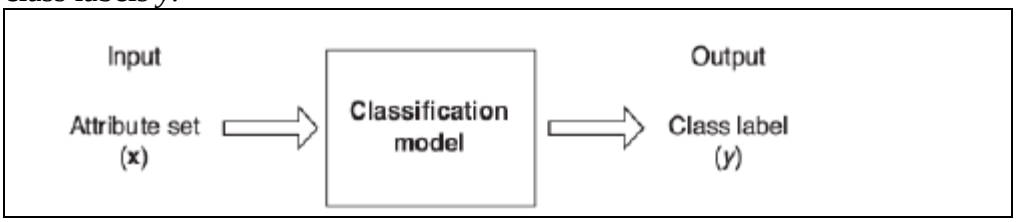
1. Classification: Basic concepts

Classification, which is the task of assigning objects to one of several predefined categories, is a pervasive problem that encompasses many diverse applications. **Examples**

- Detecting spam email messages based upon the message header and content
- Categorizing cells as malignant or benign based upon the results of MRI scans
- Classifying galaxies based upon their shapes.
- A classification model may be built to categorize bank loan applications as either safe or risky, while a prediction model may be built to predict the expenditures of potential customers on computer equipment given their income and occupation.

Data classification is a two-step process. In the first step, a model is built describing a predetermined set of data classes or concepts. The model is constructed by analyzing database tuples described by attributes. Each tuple is assumed to belong to a predefined class, as determined by one of the attributes, called the class label attribute. In the second step, the constructed model is used to classify the samples.

Classification is the task of learning a target function *f* that maps each attribute set X to one of the predefined class labels *y*.



The classification model is useful for the following purposes.

Descriptive Modeling A classification model can serve as an explanatory tool to distinguish between objects of different classes. **For example**, it would be useful for both biologists and others to have a descriptive model that summarizes the data shown in the following Table and explains what features define a **vertebrate as a mammal, reptile, bird, fish, or amphibian.**

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
human	warm-blooded	hair	yes	no	no	yes	no	mammal
python	cold-blooded	scales	no	no	no	no	yes	reptile
salmon	cold-blooded	scales	no	yes	no	no	no	fish
whale	warm-blooded	hair	yes	yes	no	no	no	mammal
frog	cold-blooded	none	no	semi	no	yes	yes	amphibian
komodo dragon	cold-blooded	scales	no	no	no	yes	no	reptile
bat	warm-blooded	hair	yes	no	yes	yes	yes	mammal
pigeon	warm-blooded	feathers	no	no	yes	yes	no	bird
cat	warm-blooded	fur	yes	no	no	yes	no	mammal
leopard shark	cold-blooded	scales	yes	yes	no	no	no	fish
turtle	cold-blooded	scales	no	semi	no	yes	no	reptile
penguin	warm-blooded	feathers	no	semi	no	yes	no	bird
porcupine	warm-blooded	quills	yes	no	no	yes	yes	mammal
eel	cold-blooded	scales	no	yes	no	no	no	fish
salamander	cold-blooded	none	no	semi	no	yes	yes	amphibian

Table. The vertebrate dataset

Predictive modeling a classification model can also be used to predict the class label of unknown records.

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
gila monster	cold-blooded	scales	no	no	no	yes	yes	?

2. General Approach to Solving a Classification Problem

A classification technique (or classifier) is a systematic approach for building classification models from an input data set. Examples include decision tree classifiers, rule-based classifiers, neural networks, support vector machines, and naive Bayes classifiers.

Each technique employs a learning algorithm to identify a model that best fits the relationship between the attribute set and class label of the input data. The model generated by a learning algorithm should both fit the input data well and correctly predict the class labels of records it has never seen before. Therefore, a key objective of the learning algorithm is to build models with good generalization capability; i.e., models that accurately predict the class labels of previously unknown records.

The following figure shows a general approach for solving classification problems.
First, a training set consisting of records whose class labels are known must be provided. The training set is used to build a classification model, which is subsequently applied to the test set, which consists of records with unknown class labels.

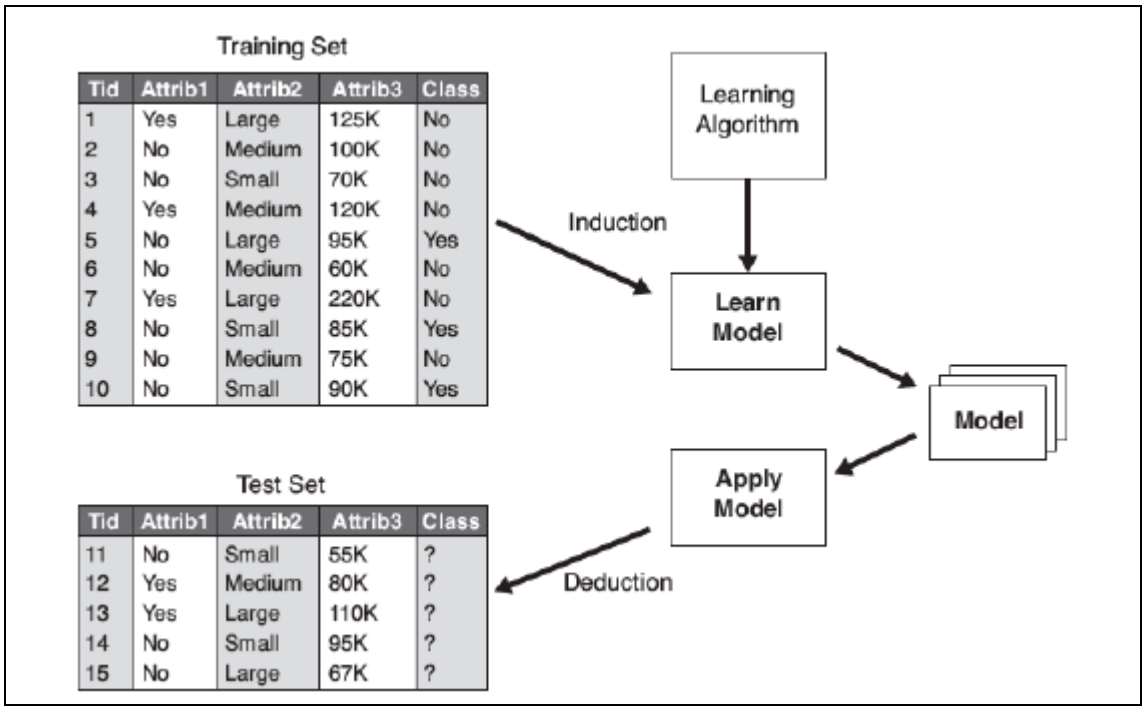


Figure: a General approach for building a classification model.

Evaluation of the performance of a classification model is based on the counts of test records correctly and incorrectly predicted by the model. These counts are tabulated in a table known as a confusion matrix as shown below.

		Predicted Class	
		$Class = 1$	$Class = 0$
Actual Class	$Class = 1$	f_{11}	f_{10}
	$Class = 0$	f_{01}	f_{00}

Table: Confusion matrix for a 2 class problem.

Based on the entries in the confusion matrix, the total number of correct predictions made by the model is *(f11* + foo) and the total number of incorrect predictions is *(f1o + f01).*

Although a confusion matrix provides the information needed to determine how well a classification model performs, summarizing this information with a single number would make it more convenient to compare the performance of different models. This can be done using a performance metric such as accuracy, which is defined as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}.$$

Equivalently, the performance of a model can be expressed in terms of its error rate, which is given by the following equation:

$$\text{Error rate} = \frac{\text{Number of wrong predictions}}{\text{Total number of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}.$$

Most classification algorithms seek models that attain the highest accuracy, or equivalently, the lowest error rate when applied to the test set.

3. Decision Tree Induction

How a Decision Tree Works

To illustrate how classification with a decision tree works, consider a simpler version of the vertebrate classification problem described in the previous section. Instead of classifying the vertebrates into five distinct groups of species, we assign them to two categories: mammals and non-mammals. Suppose a new species is discovered by scientists. How can we tell whether it is a mammal or a non-mammal? One approach is to pose a series of questions about the characteristics of the species.

The first question we may ask is whether the species is cold- or warm-blooded. If it is cold-blooded, then it is definitely not a mammal. Otherwise, it is either a bird or a mammal. In the latter case, we need to ask a follow-up question: Do the females of the species give birth to their young? Those that do give birth are definitely mammals, while those that do not are likely to be non-mammals.

The previous example illustrates how we can solve a classification problem by asking a series of carefully crafted questions about the attributes of the test record. Each time we receive an answer, a follow-up question is asked until we reach a conclusion about the class label of the record. The series of questions and their possible answers can be organized in the form of a decision tree, which is a hierarchical structure consisting of nodes and directed edges.

The following Figure shows the decision tree for the mammal classification problem. The tree has three types of nodes:

• A **root node** that has no incoming edges and zero or more outgoing edges.

• **Internal nodes**, each of which has exactly one incoming edge and two or more outgoing edges.
• **Leaf or terminal nodes**, each of which has exactly one incoming edge and no outgoing edges.

In a decision tree, each leaf node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics.

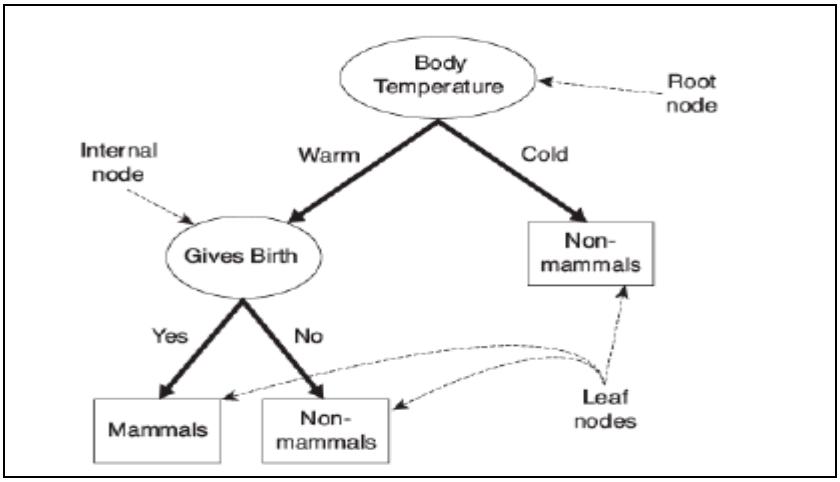
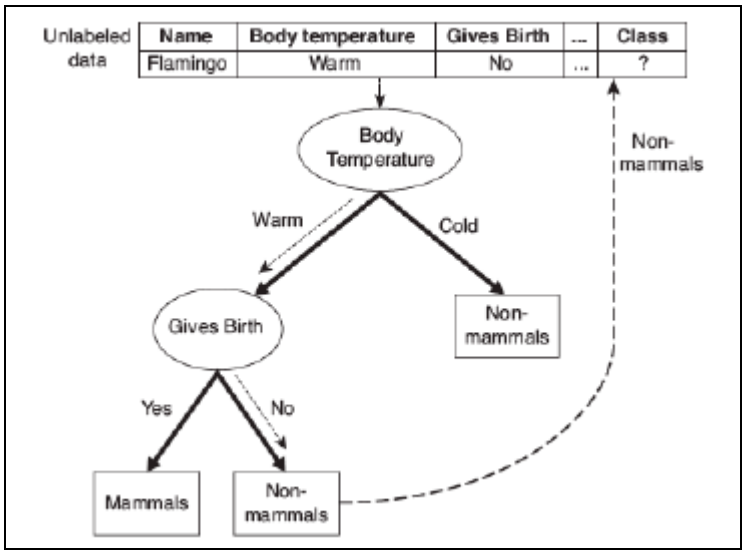


Figure: A decision tree for the mammal classification problem.

For example, the root node shown in above figure uses the attribute body Temperature to separate warm-blooded from cold-blooded vertebrates. Since all cold-blooded vertebrates are non-mammals, a leaf node labeled Non-mammals is created as the right child of the root node. If the vertebrate is warm-blooded, a subsequent attribute, Gives Birth, is used to distinguish mammals from other warm-blooded creatures, which are mostly birds.

Classifying a test record is straightforward once a decision tree has been constructed. Starting from the root node, we apply the test condition to the record and follow the appropriate branch based on the outcome of the test.



4. How to Build a Decision Tree: Hunt's Algorithm

In Hunt's algorithm, a decision tree is grown in a recursive fashion by partitioning the training records into successively purer subsets. Let D_t be the set of training records that are associated with

node t and $y = \{y_1, y_2, \ldots, y_c\}$ be the class labels. The following is a recursive definition of Hunt's algorithm.

Step 1: If all the records in D_t, belong to the same class y_t then t is a leaf node labeled as y_t.
Step 2: If D_t contains records that belong to more than one class, an attribute test condition is selected to partition the records into smaller subsets. A child node is created for each outcome of the test condition and the records in D_t are distributed to the children based on the outcomes. The algorithm is then recursively applied to each child node.

In the example shown in the following Figure, each record contains the personal information of a borrower along with a class label indicating whether the borrower has defaulted on loan payments.

	binary	categorical	continuous	class
Tid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Figure: Training set for predicting borrowers who will default on loan payments.

The initial tree for the classification problem contains a single node with class label Defaulted = No *(see following Figure* (a)), which means that *most* of the borrowers successfully repaid their loans. The tree, however, needs to be refined since the root node contains records from both classes. The records M" subsequently divided into smaller subsets based on the outcomes of the Home Owner test condition, as shown in following Figure (b). Hunt's algorithm is then applied recursively to each child of the root node. From the training set given in above Figure, notice that all borrowers who are homeowners successfully repaid their loans. The left child of the root is, therefore, a leaf node labeled Defaulted = No (see following Figure (b)). For the right child, we need to continue applying the recursive step of Hunt's algorithm until all the records belong to the same class. The trees resulting from each recursive step are shown in following Figures (c) and (d).

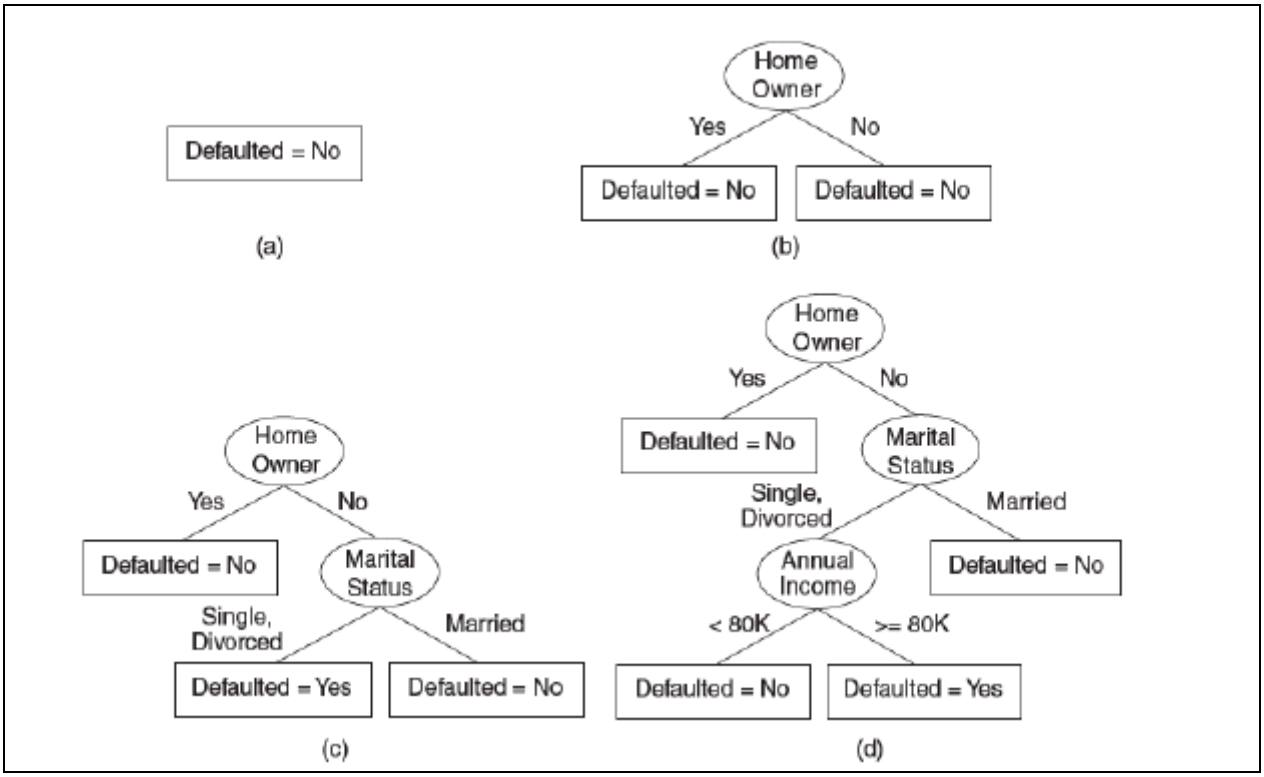


Figure: Hunts algorithm for inducing decision trees.

Design Issues of Decision Tree Induction

A learning algorithm for inducing decision trees must address the following two issues.

1. How should the training records be split? Each recursive step of the tree-growing process must select an attribute test condition to divide the records into smaller subsets.
2. How should the splitting procedure stop? A stopping condition is needed to terminate the tree-growing process. A possible strategy is to continue expanding a node until either all the records belong to the same class or all the records have identical attribute values.

5. Methods for Expressing Attribute Test Conditions

Decision tree induction algorithms must provide a method for expressing an attribute test condition and its corresponding outcomes for different attribute types.

Binary Attributes The test condition for a binary attribute generates two **potential outcomes, as shown in following Figure.**

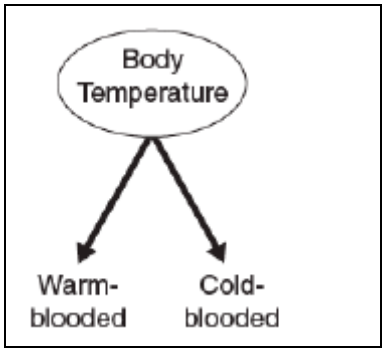
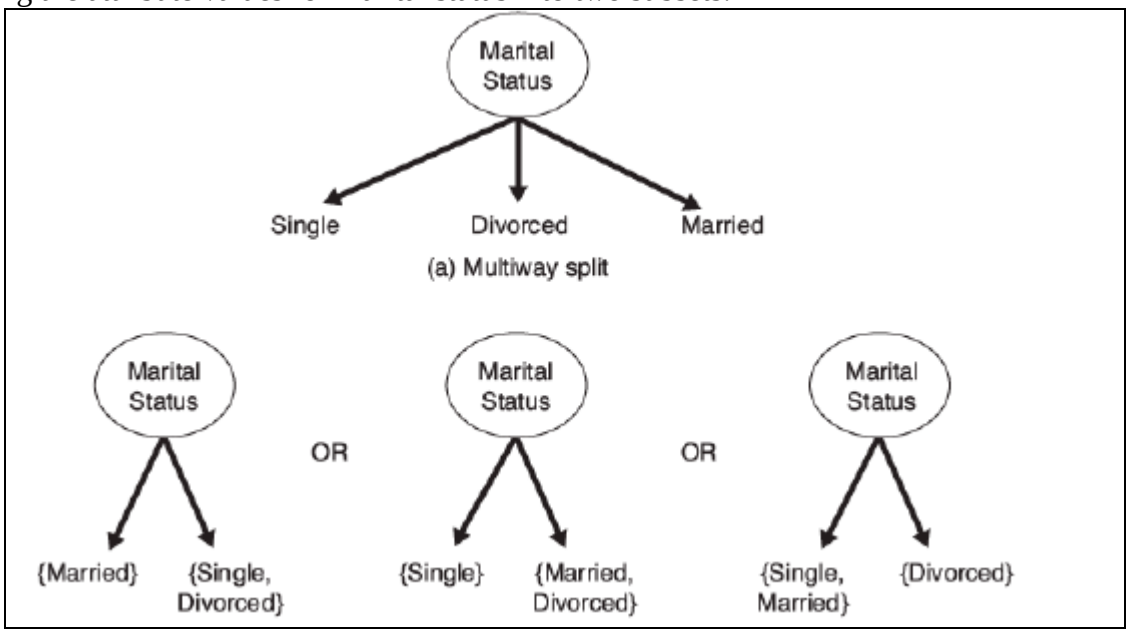


Figure: Test condition for binary attributes.

Nominal Attributes Since a nominal attribute can have many values its test condition can be expressed in two ways, as shown in following Figure. For a multi-way split (following Figure (a), the number of outcomes depends on the number of distinct values for the corresponding attribute.

For example, if an attribute such as marital status has three distinct values----single, married, or divorced-its test condition will produce a three-way split. On the other hand, some decision tree algorithms, such as a CART, produce only binary splits by considering all 2^{k-1} - 1 ways of creating a binary partition of *k* attribute values. Following Figure (b) illustrates three different ways of grouping the attribute values for marital status into two subsets.



(b) Binary split {by grouping attribute values}

Figure: Test conditions for nominal attributes.

Ordinal Attributes Ordinal attributes can also produce binary or multi-way splits. Ordinal attribute values can be grouped as long as the grouping does not violate the order property of the attribute values. The following Figure illustrates various ways of splitting training records based on the Shirt Size attribute.

The groupings are shown in Figures (a) and (b) preserve the order among the attribute values, whereas the grouping is shown in Figure (c) violates this property because it combines the attribute values Small and Large into the same partition while Medium and Extra Large are combined into another **partition.**

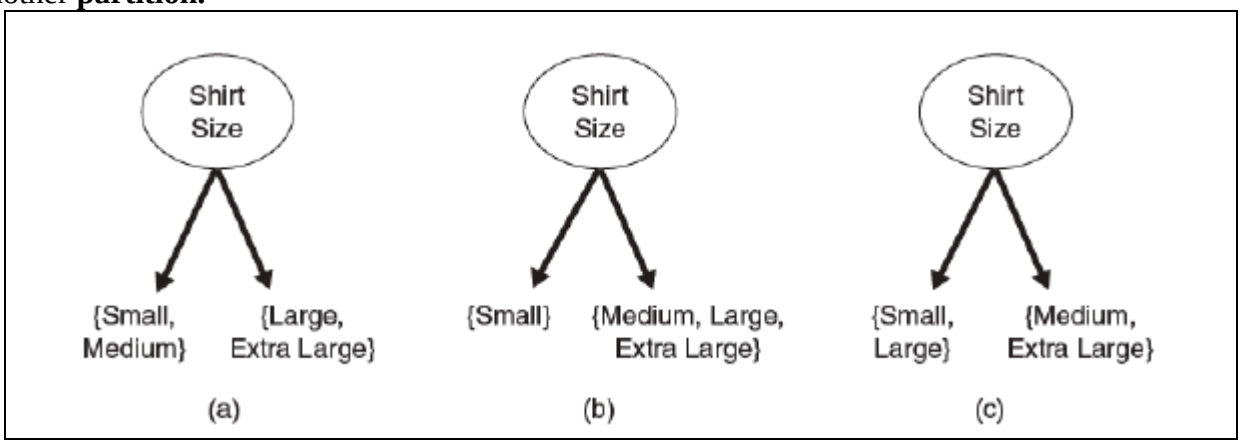


Figure: Different ways of grouping ordinal attribute values.

Continuous Attributes For continuous attributes, the test condition can be expressed as a comparison test *(A < v)* or *(A ≥ v)* with binary outcomes, or **a range query with outcomes of the form** $Vi \leq A < Vi+l$, **for** i = **1, ... , *k*. The** difference between these approaches is shown in the following Figure

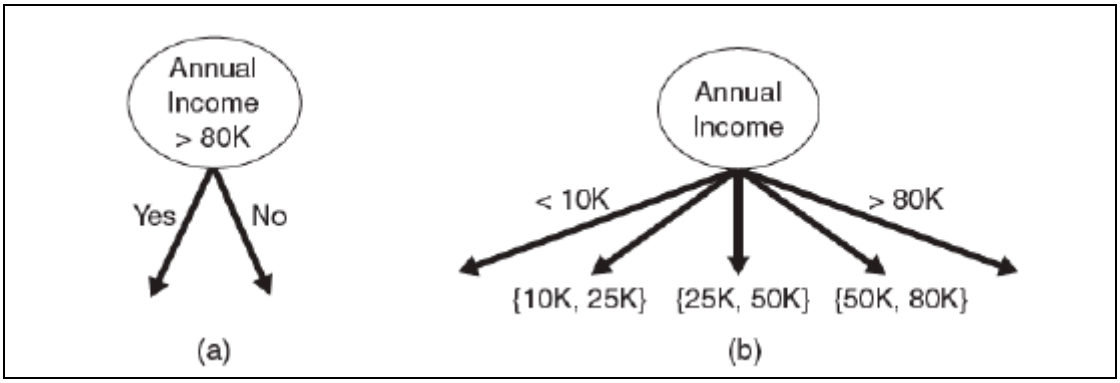


Figure: Test condition for continuous attributes.

6. Measures for Selecting the Best Split

Attribute selection measure:- The information gain measure is used to select the test attribute at each node in the tree. Such a measure is referred to as an attribute selection measure or a measure of the goodness of split. The attribute with the highest information gain (or greatest entropy reduction) is chosen as the test attribute for the current node.

Let S be a set consisting of s data samples. Suppose the class label attribute has m distinct values defining m distinct classes, Ci (for i = 1,..,m). Let si be the number of samples of S in class Ci. The expected information needed to classify a given sample is given by:

$$I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

The entropy or expected information based on the partitioning into subsets by A is given by:

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j} + \cdots + s_{mj}}{s} I(s_{1j}, \ldots, s_{mj}).$$

The smaller the entropy value is, the greater the purity of the subset partitions. The encoding information that would be gained by branching on A is

$$Gain(A) = I(s_1, s_2, \ldots, s_m) - E(A).$$

Example: Induction of a decision tree. The following Table presents a training set of data tuples taken from the All Electronics customer database. (The data are adapted from [Quinlan 1986b]). The class label attribute buys a computer, has two distinct values (namely {yes, no}), therefore, there are two distinct classes (m = 2). Let C1 correspond to the class yes and class C2 correspond to no. There are 9 samples of class yes and 5 samples of class no. To compute the information gain of each attribute, we first use first Equation to compute the expected information needed to classify a given sample. This is:

$$I(s_1, s_2) = I(9, 5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

Next, we need to compute the entropy of each attribute. Let's start with the attribute age. We need to look at the distribution of yes and no samples for each value of age. We compute the expected information for each of these distributions.

rid	age	income	student	credit_rating	Class: buys_computer
1	<30	high	no	fair	no
2	<30	high	no	excellent	no
3	30-40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	30-40	low	yes	excellent	yes
8	<30	medium	no	fair	no
9	<30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<30	medium	yes	excellent	yes
12	30-40	medium	no	excellent	yes
13	30-40	high	yes	fair	yes
14	>40	medium	no	excellent	no

Table. Training set

Using second equation the expected information needed to classify a given sample if the samples are partitioned according to age is

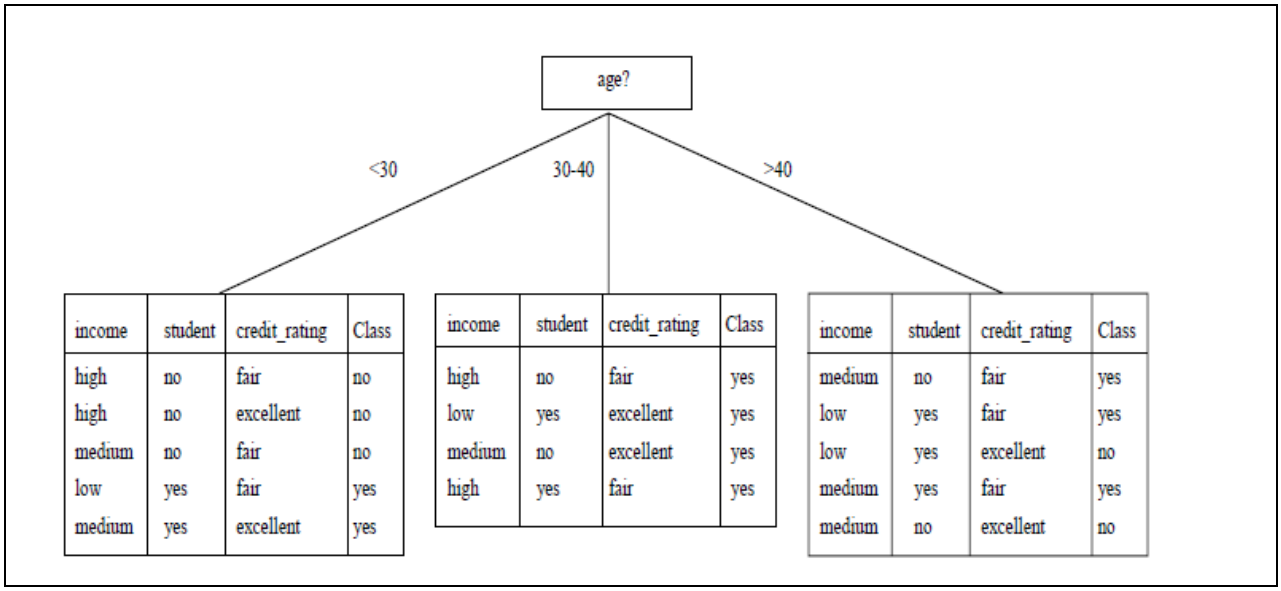
$$\begin{array}{llll} \text{for } age = \text{“}<30\text{”}: & s_{11} = 2 & s_{21} = 3 & I(s_{11}, s_{21}) = 0.971 \\ \text{for } age = \text{“}30\text{-}40\text{”}: & s_{12} = 4 & s_{22} = 0 & I(s_{12}, s_{22}) = 0 \\ \text{for } age = \text{“}>40\text{”}: & s_{13} = 3 & s_{23} = 2 & I(s_{13}, s_{23}) = 0.971 \end{array}$$

$$E(age) = \frac{5}{14} I(s_{11}, s_{21}) + \frac{4}{14} I(s_{12}, s_{22}) + \frac{5}{14} I(s_{13}, s_{23}) = 0.694.$$

Hence, the gain in information from such a partition would be:

$$Gain(age) = I(s_1, s_2) - E(age) = 0.246$$

Similarly, we can compute Gain (income) = 0.029, Gain(student) = 0.151, and Gain(credit rating) = 0.048. Since age has the highest information gain among the attributes, it is selected as the test attribute. A node is created and labeled with age, and branches are grown for each of the attribute's values. The samples are then partitioned accordingly, as shown in following Figure.

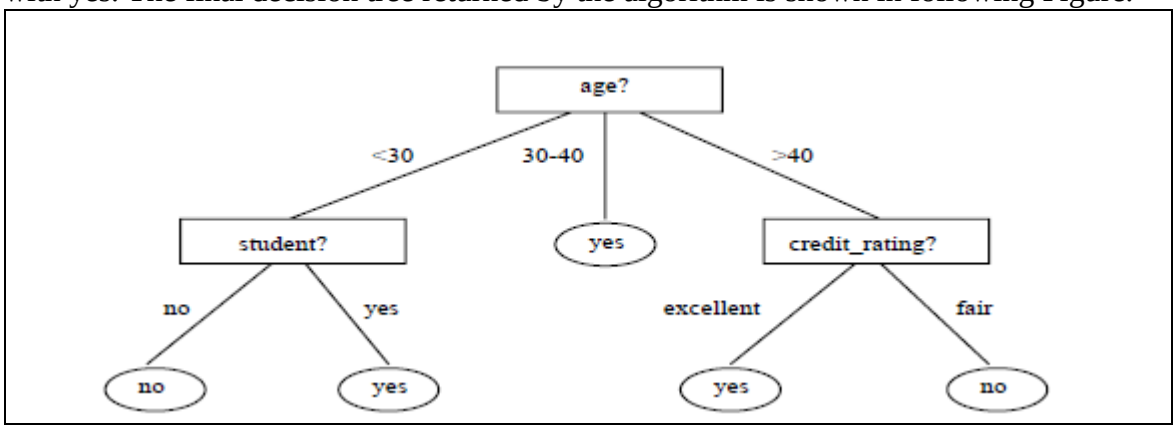


income	student	credit_rating	Class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	Class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

income	student	credit_rating	Class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

Notice that the samples falling into the partition for age = 30-40 all belong to the same class. Since they all belong to class yes, a leaf should, therefore, be created at the end of this branch and labeled with yes. The final decision tree returned by the algorithm is shown in following Figure.



Some examples of impurity measures include

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)],$$

Where c is the number of classes and 0 log2 0 = 0 in entropy calculations. We provide several examples of computing the different impurity measures.

Node N_1	Count
Class=0	0
Class=1	6

$\text{Gini} = 1 - (0/6)^2 - (6/6)^2 = 0$
$\text{Entropy} = -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$
$\text{Error} = 1 - \max[0/6, 6/6] = 0$

Node N_2	Count
Class=0	1
Class=1	5

$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$
$\text{Entropy} = -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$
$\text{Error} = 1 - \max[1/6, 5/6] = 0.167$

Node N_3	Count
Class=0	3
Class=1	3

$\text{Gini} = 1 - (3/6)^2 - (3/6)^2 = 0.5$
$\text{Entropy} = -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$
$\text{Error} = 1 - \max[3/6, 3/6] = 0.5$

Splitting of Binary Attributes

Consider the diagram shown in following Figure. Suppose there are two ways to split the data into smaller subsets. Before splitting, the Gini index is 0.5 since there are an equal number of records from both classes. If attribute *A* is chosen to split the data, the Gini index for node N1 is 0.4898, and for node N2, it is 0.480. The weighted average of the Gini index for the descendent nodes is (7/12) x 0.4898 + (5/12) x 0.480 = 0.486. Similarly, we can show that the weighted average of the Gini index for attribute *B* is 0.375. Since the subsets for attribute *B* have a smaller Gini index, it is preferred over attribute *A*.

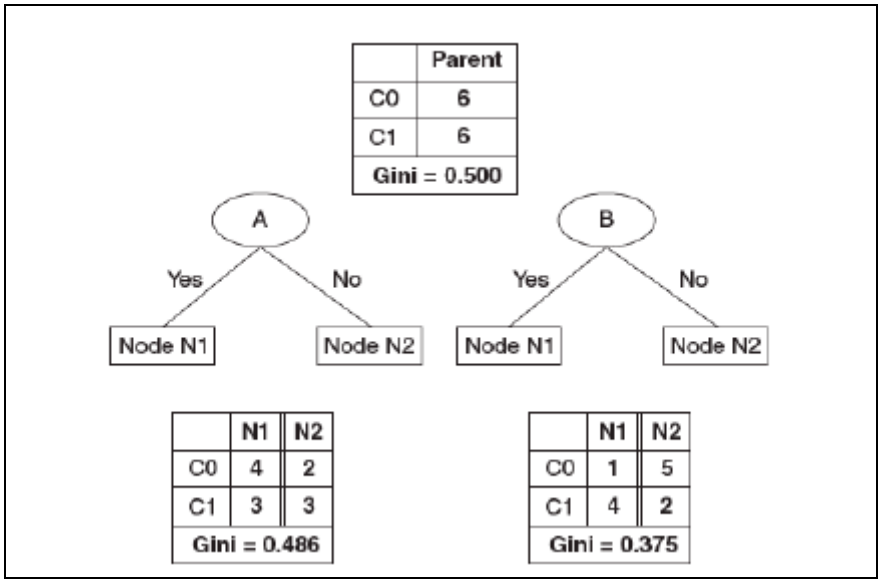


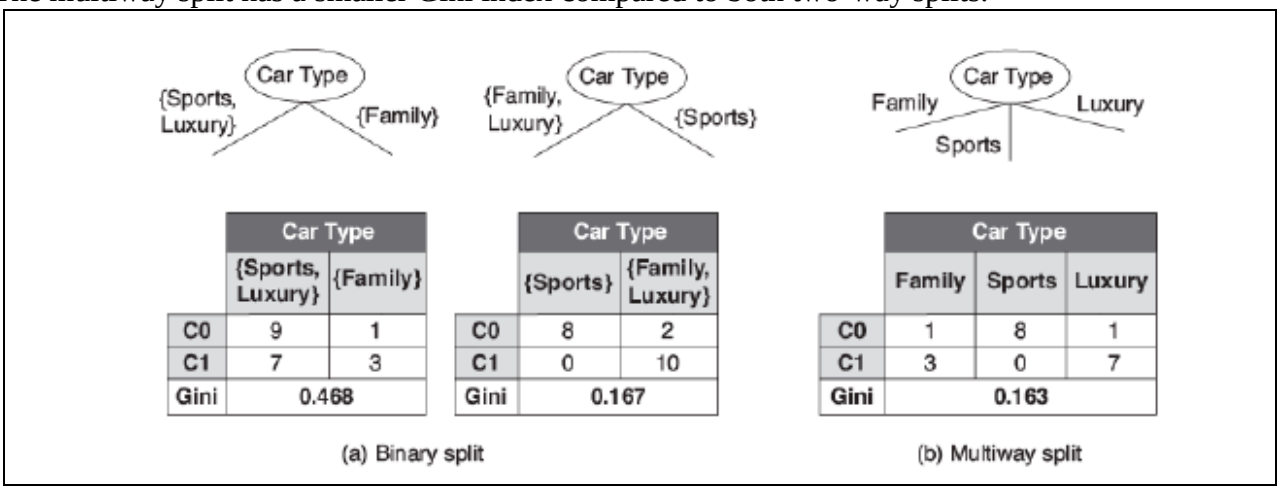
Figure: Splitting binary attributes.

Splitting of Nominal Attributes

As previously noted, a nominal attribute can produce either binary or multi-way split. The computation of the Gini index for a binary split is similar to that shown for determining binary attributes. For the first binary grouping of the Car Type attribute, the Gini index of {Sports, Luxury} is 0.4922 and the Gini index of {Family} is 0.3750. The weighted average Gini index for the grouping is equal to 16/20 X 0.4922 +4/20 X 0.3750 = 0.468.

Similarly, for the second binary grouping of {Sports} and {Family. Luxury}, the weighted average Gini index is 0.167. The second grouping has a lower Gini index because its corresponding subsets are much purer.

For the multiway split, the Gini index is computed for every attribute value. Since Gini({Family}) = 0.375, Gini({Sports}) = 0, and Gini({Luxury}) = 0.219, the overall Gini index for the multi way split is equal to 4/20 X 0.375 + 8/20 X 0 + 8/20 X 0.219 = 0.163.
The multiway split has a smaller Gini index compared to both two-way splits.



	Car Type	
	{Sports, Luxury}	{Family}
C0	9	1
C1	7	3
Gini	0.468	

	Car Type	
	{Sports}	{Family, Luxury}
C0	8	2
C1	0	10
Gini	0.167	

	Car Type		
	Family	Sports	Luxury
C0	1	8	1
C1	3	0	7
Gini	0.163		

(a) Binary split (b) Multiway split

Figure: Splitting nominal attributes.

Splitting of Continuous Attributes

Consider the example shown in Figure 4.16, in which the test condition Annual Income ≤ *v* is used to split the training records for the loan default classification problem.

Class	No	No	No	Yes	Yes	Yes	No	No	No	No
Annual Income										
Sorted Values →	60	70	75	85	90	95	100	120	125	220
Split Positions →	55	65	72	80	87	92	97	110	122	172	230

	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		0.300		0.343		0.375		0.400		0.420	

Figure: Splitting continuous attributes.

The best split position corresponds to the one that produces the smallest Gini index, i.e., *v* = 97. This procedure *is* less **expensive because it requires a constant amount of time to update the class** distribution at each candidate split position.

7. Algorithm for Decision Tree Induction

Algorithm presents a pseudo code for decision tree induction algorithm. The input to this algorithm is a set of training instances E along with the attribute set F. The details of this algorithm are explained below.

1. The createNode() function extends the decision tree by creating a new node. A node in the decision tree either has a test condition, denoted as node.test_cond, or a class label, denoted as node.label.
2. The find_best_split() function determines the attribute test condition for partitioning the training instances associated with a node. The splitting attribute chosen depends on the impurity measure used. The popular measures include entropy and the Gini index.
3. The Classify () function determines the class label to be assigned to a leaf node. For each leaf node t, let p(i|t) denote the fraction of training instances from class i associated with the node t.
4. The stopping_cond() function is used to terminate the tree-growing process by checking whether all the instances have identical class label or attribute values. Since decision tree classifiers employ a top-down, recursive partitioning approach for building a model, the number of training instances associated with a node decreases as the depth of the tree increases.

Algorithm: A skeleton decision tree induction algorithm.

TreeGrowth (*E*, *F*)
1: if stopping_cond (*E*, *F*) = *true* then
2: *leaf* = createNode().
3: *leaf.label* = Classify (*E*).
4: return *leaf*.
5: else
6: *root* = createNode().
7: *root.test cond* = find_best_split (*E*, *F*).
8: let *V* = *{v|v* is a possible outcome of *root.test cond }*.
9: for each $v \in V$ do
10: $E_v = \{e \mid root.test\_cond(e) = v \text{ and } e \in E\}$.
11: *child* = TreeGrowth(*Ev*, *F*).
12: add *child* as descendent of *root* and label the edge (*root* → *child*) as *v*.
13: end for
14: end if
15: return *root*.

Frequently Asked Questions

Short Answer Question

1. Define Classification?
2. Differentiate Training and Testing Data?
3. What are various data object types used in data mining?
4. Define about impurity measure used in data mining?
5. Define Accuracy and Error rate with formulae?

Long Answer Questions

1. Briefly explain about General Approach to solve a classification problem?
2. How a decision tree works and how to construct decision tree, explain about Hunt's Algorithm?
3. Explain the process of selecting the attribute test conditions for nominal, binary, ordinal and numeric attributes?
4. Briefly explain about measures for selecting the best split of attributes?
5. Write and explain decision tree induction algorithm?

Exercise Problems

1. Consider the given dataset

Customer ID	Gender	Car Type	Shirt size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Luxury	Extra Large	C1
7	F	Luxury	Small	C1
8	F	Luxury	Small	C1
9	F	Luxury	Medium	C1
10	F	Luxury	Medium	C1

(a) Compute the Gini index for the overall collection of training examples.
(b) Compute the Gini index for the Customer ID attribute.
(c) Compute the Gini index for the Gender attribute
(d) Compute the Gini index for the Car Type attribute using multiway split.
(e) Compute the Gini index for the Shirt Size attribute using multiway split.
(f) Which attribute is better, Gender, Car Type, or Shirt Size?
(g) Explain why Customer ID should not be used as the attribute test condition even though it has the lowest Gini.

UNIT-IV

Classification-Alternative Techniques: Bayes' Theorem, Naïve Bayesian Classification, Bayesian Belief Networks

1. Bayes' Theorem: Let X is a data tuple. In Bayesian terms, X is considered "evidence." As usual, 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.HjX/, the probability that the hypothesis H holds given the "evidence" or observed data tuple X. In other words, we are looking for the probability that tuple X belongs to class C, given that we know the attribute description of X.

P(H/X) is the posterior probability, or a posteriori probability, of H conditioned on X. In contrast, P(H) is the prior probability, or a priori probability, of H. Similarly, P(X/H) is the posterior probability of X conditioned on H. P(X) is the prior probability of X. Now Bayes theorem is defined as

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

2. Naive Bayesian Classification: 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, : : : , *xn*), depicting *n* measurements made on the tuple from *n* attributes, respectively, *A*1, *A*2, : : : , *An*.

2. Suppose that there are *m* classes, *C*1, *C*2, : : : , *Cm*. 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 *Ci* if and only if

$P(Ci|X) > P(Cj|X)$ for j!= *i*.

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

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

.

3. As *P(X)* is constant for all classes, only $P(X|Ci)/P(Ci)$ needs to be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, $P(C1)=P(C2)\ldots\ldots\ldots P(Cm)$, and we would therefore maximize $P(X|Ci)$.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute *P(X|Ci)*. To reduce computation in evaluating $P(X|Ci)$, the naive assumption of class-conditional independence is made.

5. To predict the class label of *X*, $P(X|Ci)P(Ci)$ is evaluated for each class *Ci* . The classifier predicts that the class label of tuple *X* is the class *Ci* if and only if

$$P(X|Ci)/P(Ci) > P(X|Cj)P(Cj) \text{ for } 1 < j < m, j \neq i.$$

In other words, the predicted class label is the class *Ci* for which *P.X*j*Ci*/*P.Ci*/ is the maximum.

"How effective are Bayesian classifiers?" Various empirical studies of this classifier in comparison to decision tree and neural network classifiers have found it to be comparable in some domains. In theory, Bayesian classifiers have the minimum error rate in comparison to all other classifiers.

However, in practice this is not always the case, owing to inaccuracies in the assumptions made for its use, such as class-conditional independence, and the lack of available probability data.

Example: Let's understand it using an example. Below we have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

outlook	temperature	humidity	windy	play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	TRUE	no

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Problem: Players will play if weather is sunny. Is this statement is correct? We can solve it using above discussed method of posterior probability.

P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)

Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes

3. Bayesian Belief Networks

Bayesian belief networks—probabilistic graphical models, which unlike naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes. Bayesian belief networks can be used for classification.

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. For brevity, we will refer to them as belief networks. A belief network is defined by two components—a directed acyclic graph and a set of conditional probability tables (Figure).

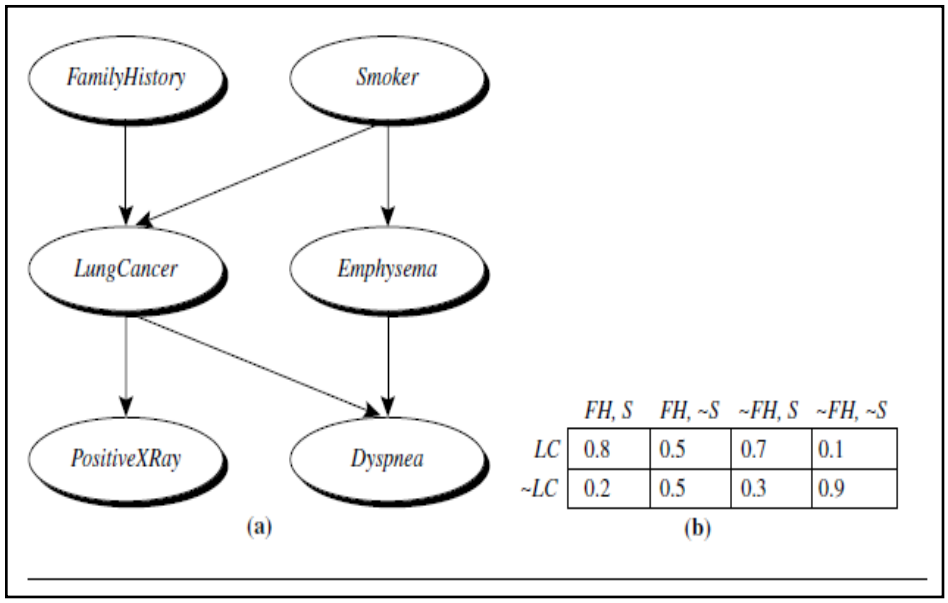


Figure: Simple Bayesian belief network. (a) A proposed causal model, represented by a directed acyclic graph. (b) The conditional probability table for the values of the variable *LungCancer* (*LC*) showing each possible combination of the values of its parent nodes, *FamilyHistory* (*FH*) and *Smoker* (*S*).

The arcs in Figure (a) allow a representation of causal knowledge. For example, having lung cancer is influenced by a person's family history of lung cancer, as well as whether or not the person is a smoker. Note that the variable *PositiveXRay* is independent of whether the patient has a family history of lung cancer or is a smoker, given that we know the patient has lung cancer. In other words, once we know the outcome of the variable *LungCancer*, then the variables *FamilyHistory* and *Smoker* do not provide any additional information regarding *PositiveXRay*. The arcs also show that the variable *LungCancer* is conditionally independent of *Emphysema*, given its parents, *FamilyHistory* and *Smoker*.

A belief network has one **conditional probability table (CPT)** for each variable.
The CPT for a variable *Y* specifies the conditional distribution *P.Y*j*Parents.Y*//, where *Parents.Y*/ are the parents of *Y*. Figure 9.1(b) shows a CPT for the variable *LungCancer*. The conditional probability for each known value of *LungCancer* is given for each possible combination of the values of its parents. For instance, from the upper leftmost and bottom rightmost entries, respectively, we see that
P(LungCancer =yes | FamilyHistory= yes, Smoker =yes) = 0.8
P(LungCancer =no| FamilyHistory =no, Smoker =no)D=0.9
Belief networks have been used to model a number of well-known problems. One example is genetic linkage analysis (e.g., the mapping of genes onto a chromosome). Other **applications** that have benefited from the use of belief networks include computer vision (e.g., image restoration and stereo vision), document and text analysis, decision support systems, and sensitivity analysis.

Frequently Asked Questions
Short Answer Questions
1. Define Bayes Theorem?
2. Define Naïve Bayes Classifier?
3. What are the Advantages of Bayesian Belief Networks?

Long Answer Questions
1. Explain Bayes Theorem implementation in Data Mining?
2. Briefly explain about Naïve Bayes Classified with an example?
3. Write a Short note on Bayesian Belief Networks?

Exercise Problem
1. Consider a binary classification problem with the following set of attributes and attribute values:
• Air Conditioner = {Working, Broken}, • Engine = {Good, Bad}, • Mileage = {High, Medium, Low}
• Rust = {Yes, No}
Suppose a rule-based classifier produces the following rule set:
Mileage = High →Value = Low
Mileage = Low →Value = High
Air Conditioner = Working, Engine = Good →Value = High
Air Conditioner = Working, Engine = Bad → Value = Low
Air Conditioner = Broken → Value = Low
(a) Are the rules mutually exclusive? Answer: No
(b) Is the rule set exhaustive? Answer: Yes
(c) Is ordering needed for this set of rules?
Answer: Yes because a test instance may trigger more than one rule.
(d) Do you need a default class for the rule set?
Answer: No because every instance is guaranteed to trigger at least one rule.

2. (a) Suppose the fraction of undergraduate students who smoke is 15% and the fraction of graduate students who smoke is 23%. If one-fifth of the college students are graduate students and the rest are undergraduates, what is the probability that a student who smokes is a graduate student?

Answer:

Given P (S|UG) = 0.15, P (S|G) = 0.23, P (G) = 0.2, P (UG) = 0.8. We want to compute P (G|S).
According to Bayesian Theorem,

$$P(G|S) = (0.23 \times 0.2) / (0.15 \times 0.8 + 0.23 \times 0.2) = 0.277.$$

(b) Given the information in part (a), is a randomly chosen college student more likely to be a graduate or undergraduate student? **Answer:** An undergraduate student, because P(UG) > P(G).

(c) Repeat part (b) assuming that the student is a smoker.

Answer: An undergraduate student because P(UG|S) > P(G|S).

(d) Suppose 30% of the graduate students live in a dorm but only 10% of the undergraduate students live in a dorm. If a student smokes and lives in the dorm, is he or she more likely to be a graduate or undergraduate student? You can assume independence between students who live in a dorm and those who smoke.

Answer:

First, we need to estimate all the probabilities.
P (D|UG) = 0.1, P (D|G) = 0.3.
P (D) = P (UG).P (D|UG) +P (G).P (D|G) = 0.8∗0.1+0.2∗0.3 = 0.14.
P(S) = P (S|UG) P (UG) +P (S|G) P (G) = 0.15∗0.8+0.23∗0.2 = 0.166.
P (DS|G) = P (D|G) × P (S|G) = 0.3 × 0.23 = 0.069 (using conditional independent assumption)
P (DS|UG) = P (D|UG) × P (S|UG) = 0.1 × 0.15 = 0.015.
We need to compute P (G|DS) and P (UG|DS).
P (G|DS) = 0.069 × 0.2 / P (DS) = 0.0138/ P (DS)
P (UG|DS) = 0.015 × 0.8/P (DS) =0.012/P (DS)
Since P (G|DS) > P (UG|DS), he/she is more likely to be a graduate student.

3. Consider the data set shown in Table

Record	A	B	C	Class
1	0	0	0	+
2	0	0	1	−
3	0	1	1	−
4	0	1	1	−
5	0	0	1	+
6	1	0	1	+
7	1	0	1	−
8	1	0	1	−
9	1	1	1	+
10	1	0	1	+

(a) Estimate the conditional probabilities for P (A|+), P (B|+), P (C|+), P (A|−), P (B|−), and P (C|−).

Answer:

P (A = 1|−) = 2/5 = 0.4, P (B = 1|−) = 2/5 = 0.4,
P(C = 1|−) = 1, P (A = 0|−) = 3/5 = 0.6,
P (B = 0|−) = 3/5 = 0.6, P(C = 0|−) = 0; P (A = 1|+) = 3/5 = 0.6,
P (B = 1|+) = 1/5 = 0.2, P(C = 1|+) = 2/5 = 0.4,
P (A = 0|+) = 2/5 = 0.4, P (B = 0|+) = 4/5 = 0.8,
P (C = 0|+) = 3/5 = 0.6.

UNIT-V

Association analysis: problem definition, frequent itemset generation: The Apriori principle, frequent itemset generation in the Apriori algorithm, candidate generation and pruning, support counting, rule generation, compact representation of frequent itemsets, FP-Growth algorithms.

1. Association analysis:

Association rule mining is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction. The uncovered relationships can be represented in the form of association rules or sets of frequent items. The following table illustrates an example of Market Basket Transactions.

TID	Items
1	{Bread, Milk}
2	{Bread, Diaper, Butter, Eggs}
3	{Milk, Diaper, Butter, Coke}
4	{Bread, Milk, Diaper, Butter }
5	{Bread, Milk, Diaper, Cola}

Table. An example of market basket transactions

Problem Definition: Definition: The problem of association rule mining is defined as:
Let I = {i1, i2….in} be a set of n binary attributes called items.
Let D = {t1, t2…tn} be a set of transactions called the database.
Each transaction in D has a unique transaction ID and contains a subset of the items in I. Binary Representation Market basket data can be represented in a binary format as shown in the following Table, where each row corresponds to a transaction and each column corresponds to an item. An item can be treated as a binary **variable whose value is one if the item is present in a transaction and zero** otherwise. Because the presence of an item in a transaction is often considered more important than its absence, an item is an asymmetric binary variable.

TID	Bread	Milk	Diaper	Butter	Eggs	Cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

Table: A binary 0/1 representation of market basket data

In association analysis, a collection of zero or more items is termed an itemset. If an itemset contains *k* items, it is called a k-itemset. For instance, {Butter, Diapers, Milk} is an example of a 3-item set. The null (or empty) set is an item set that does not contain any items. The transaction width is defined as the number of items present in a transaction.

Association Rule: An association rule is an implication expression of the form X →Y, where X and Y are disjoint item sets, i.e., X ∩Y = 0. The strength of an association rule can be measured in terms of its support and confidence. Support determines how often a rule is applicable to a given data set, while confidence determines how frequently items in Y appear in transactions that contain X. The formal definitions of these metrics are,

$$\text{Support, } s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N};$$
$$\text{Confidence, } c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

Example:
Consider the rule {Milk, Diapers} → {Butter}. Since the support count for **{Milk, Diapers, Butter} is 2** and the total number of transactions is 5, the rule's **support is 2/5 =0.4.**

The rule's confidence is obtained by dividing the support count for by the support count for {Milk, Diapers}. Since there are 3 transactions that contain milk and diapers, **the confidence for this rule is 2/3=0.67.**

Association Rule Discovery
Given a set of transactions *T,* find all the rules having support ≥ *minsup* and confidence ≥ *minconf,* where *minsup* and *minconf* are the corresponding support and confidence thresholds.
A brute-force approach for mining association rules is to compute the support and confidence for every possible rule. This approach is prohibitively expensive because there are exponentially many rules that can be extracted from a data set. More specifically, the total number of possible rules extracted from a data set that contains *d* items is

$$\mathbf{R= 3^d – 2^{d+1}+1}$$

Even for the small data set shown in the first Table, this approach requires us to compute the support and confidence for $\mathbf{3^6 - 2^7+ 1 = 602}$ **rules**. More than 80% of the rules are discarded after applying *minsup* = 20% and *minconf* = 50%, thus making most of the computations become wasted.

If the itemset is infrequent, then all six candidate rules can be pruned immediately without having to compute their confidence values. Therefore, a common strategy adopted by many association rule mining algorithms is to decompose the problem into two major subtasks:
1. Frequent Itemset Generation, whose objective is to find all the itemsets that satisfy the *minsup* threshold, these item sets are called frequent itemsets.
2. Rule Generation, whose objective is to extract all the high-confidence rules from the frequent itemsets found in the previous step. These rules are called strong rules.

2. Frequent itemset generation

A lattice structure can be used to enumerate the list of all possible item sets. The following Figure shows an itemset lattice for *1= {a, b, c, d, e}*. In general, a data set that contains *k* items can potentially generate up to 2^k - 1 frequent itemsets, excluding the null set.

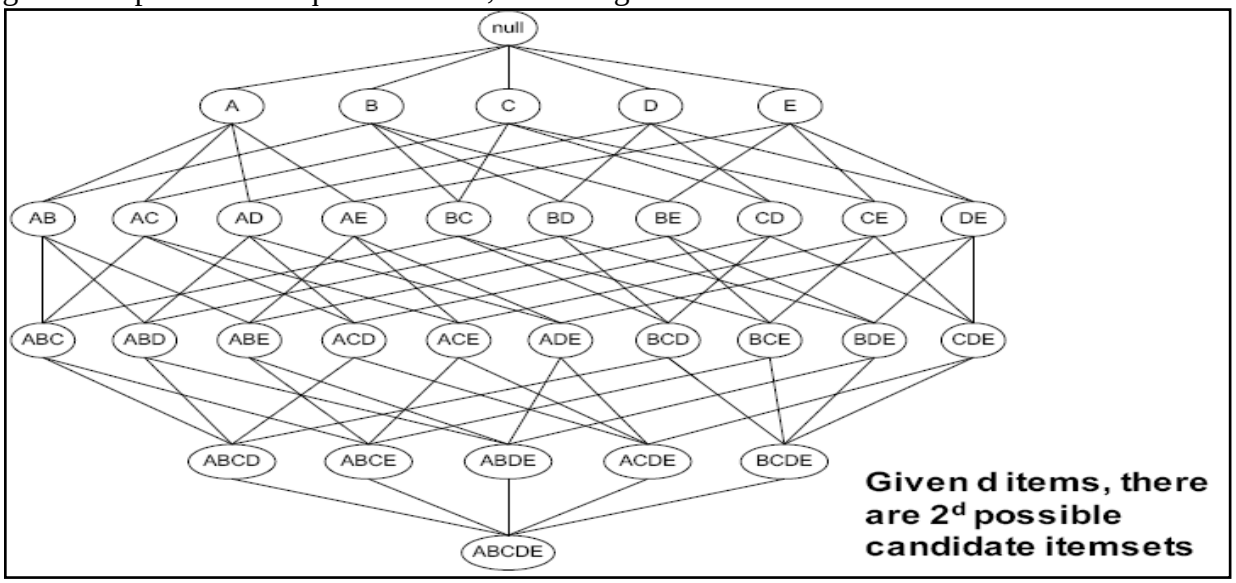


Figure: An item set lattice.

A brute-force approach for finding frequent itemsets is to determine the support count for every candidate itemset in the lattice structure. To do this, we need to compare each candidate against every transaction, an operation that is shown in following Figure. If the candidate is contained in a transaction, its support count will be incremented. For example, the support for {Bread, Milk} is incremented three times because the itemset is contained in transactions 1, 4, and 5.Such an approach can be very expensive because it requires *O(NMw)* comparisons, where *N* is the number of transactions, M = 2^k -1 is the number of candidate itemsets, and *w* is the maximum transaction width.

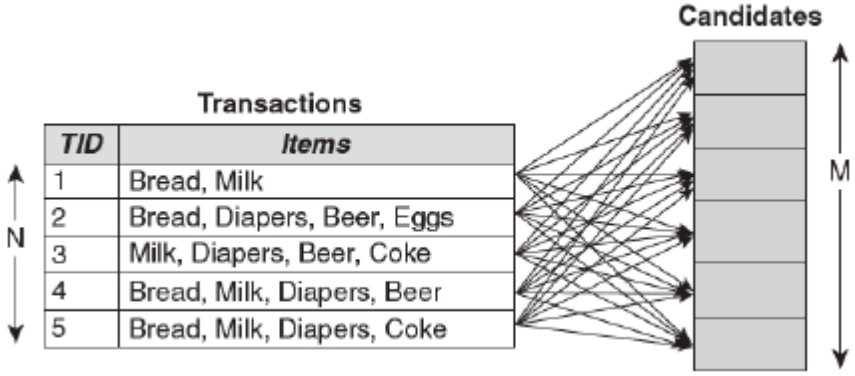


Figure: Counting the support of candidate itemsets

There are several ways to reduce the computational complexity of frequent item set generation

1. Reduce the number of candidate itemsets *(M).*
2. Reduce the number of comparisons.

The Apriori principle: If an itemset is frequent, then all of its subsets must also be frequent.

For ex consider the itemset lattice shown in following Figure. Suppose {c, *d,* e} is a frequent *itemset.* Clearly, any transaction that contains *{c, d, e}* must also contain *its* subsets, *{c,d}, {c,e}, {d,e},* {c}, *{d},* and {e}. As a result, if *{c, d, e}* is frequent, then all subsets of *{c, d, e}* (i.e., the shaded itemsets in this figure) must *also* be frequent.

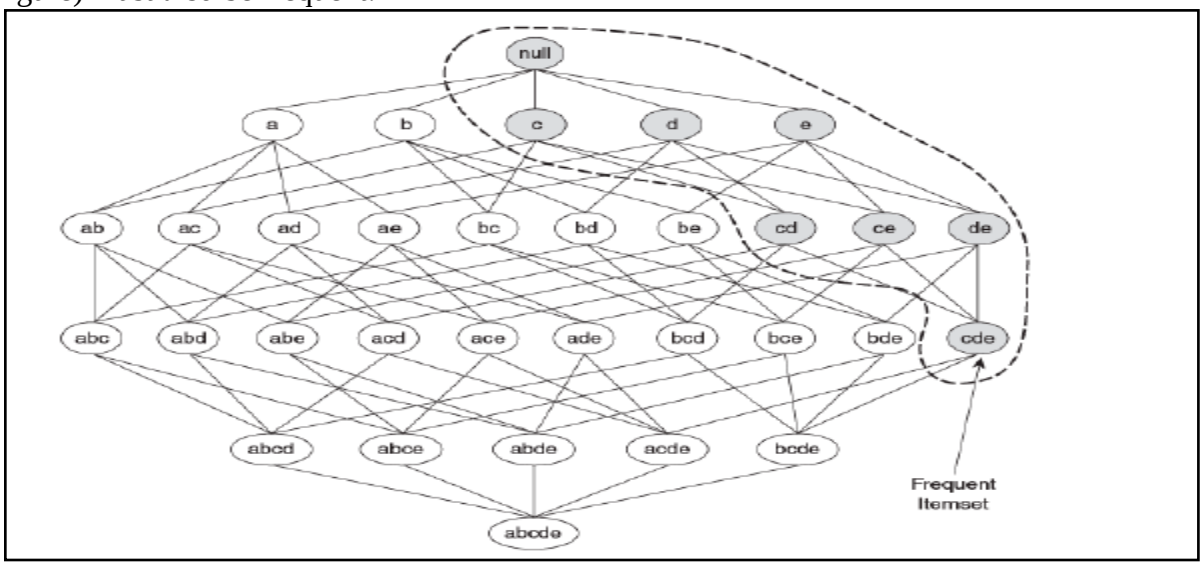


Figure: An illustration of the *Apriori* principle.

If *{c, d,* e} is frequent, then all subsets of this itemset are frequent,

Conversely, if an itemset such as *{a, b}* is infrequent, then all of its supersets must be infrequent too. *For ex* in the following Figure, the entire sub graph containing the supersets of *{a, b}* can be pruned immediately once *{a, b}* is found to be infrequent.

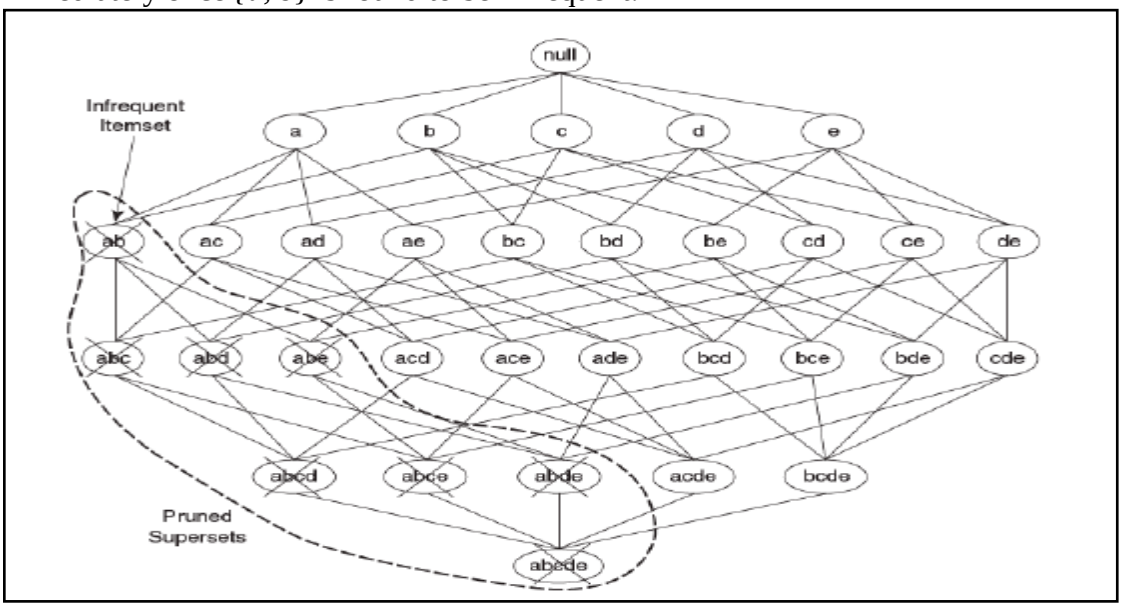


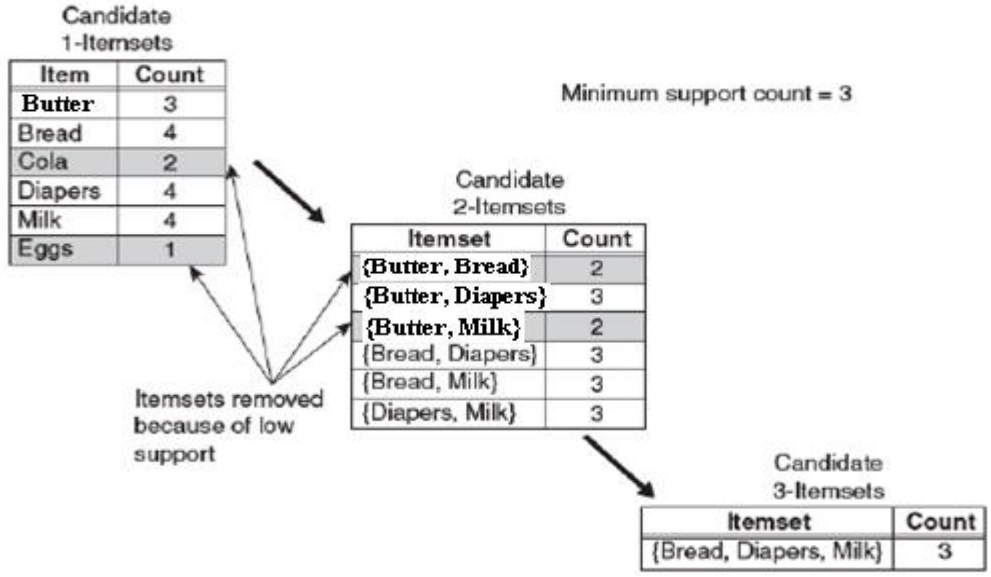
Figure: An illustration of support-based pruning.

if *{a, b}* is infrequent, then all supersets of *{a, b}* are infrequent. This strategy of trimming the exponential search space based on the support measure is known as support-based pruning. In this, the support for an itemset never exceeds the support for its subsets. This property is also known as the anti-monotone property of the support measure.

3. Frequent itemset generation in the Apriori algorithm

Apriori is an influential algorithm for mining frequent itemsets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset properties, as we shall see below. Apriori employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1)-itemsets. First, the set of frequent 1-itemsets is found. This set is denoted L1. L1 is used to find L2, the frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-itemsets can be found. The finding of each Lk requires one full scan of the database. To improve the efficiency of the level-wise generation of frequent itemsets, an important property called the Apriori property, presented below, is used to reduce the search space.

Suppose that the minimum transaction support count required is 2 (i.e., min sup = 50%). The following Figure provides a high-level illustration of the frequent itemset generation part of the *Apriori* algorithm for the transactions shown in the first Table. We assume that the support threshold is 60%, which is equivalent to a minimum support count equal to 3.



Initially, every item is considered as a candidate l-itemset. After counting their supports, the candidate itemsets {Cola} and {Eggs} are discarded because they appear in fewer than three transactions. In the next iteration, candidate 2-itemsets are generated using only the frequent l-itemsets because the *Apriori* principle ensures that all supersets of the infrequent l-itemsets must be infrequent. Because there are only four frequent l-itemsets, the number of candidate 2-itemsets generated by the algorithm is 6. Two of these six candidates, {Butter, Bread} and {Butter, Milk}, are subsequently found to be infrequent after computing their support values. The remaining four candidates are frequent, and thus will be used to generate candidate 3-itemsets. Without support-based pruning, there are 20 candidate 3-itemsets that can be formed using the six items given in this example. With the *Apriori* principle, we only need to keep candidate 3-itemsets whose subsets are frequent. The only candidate that has this property is {Bread, Diapers, Milk}.

The effectiveness of the *Apriori* pruning strategy can be shown by counting the number of candidate itemsets generated. A brute-force strategy of enumerating all itemsets (up to size 3) as candidates will produce

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

candidates. With the *Apriori* principle, this number decreases to

$$\binom{6}{1} + \binom{4}{2} + 1 = \mathbf{6} + 6 + 1 = 13$$

Candidates, which represents a 68% reduction in the number of candidates item sets even in this simple example. The pseudo code for the frequent itemset generation part of the *Apriori* algorithm is shown in Algorithm

```
1: k = 1.
2: F_k = { i | i ∈ I ∧ σ({i}) ≥ N × minsup}.    {Find all frequent 1-itemsets}
3: repeat
4:    k = k + 1.
5:    C_k = apriori-gen(F_{k-1}).    {Generate candidate itemsets}
6:    for each transaction t ∈ T do
7:       C_t = subset(C_k, t).    {Identify all candidates that belong to t}
8:       for each candidate itemset c ∈ C_t do
9:          σ(c) = σ(c) + 1.    {Increment support count}
10:      end for
11:   end for
12:   F_k = { c | c ∈ C_k ∧ σ(c) ≥ N × minsup}.    {Extract the frequent k-itemsets}
13: until F_k = ∅
14: Result = ⋃ F_k.
```

4. Candidate Generation and Pruning

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

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

Next, we will briefly describe several candidate generation procedures, including the one used by the apriori-gen function.

Brute-Force Method: The brute-force method considers every k-itemset as a potential candidate and then applies the candidate pruning step to remove any unnecessary candidates.

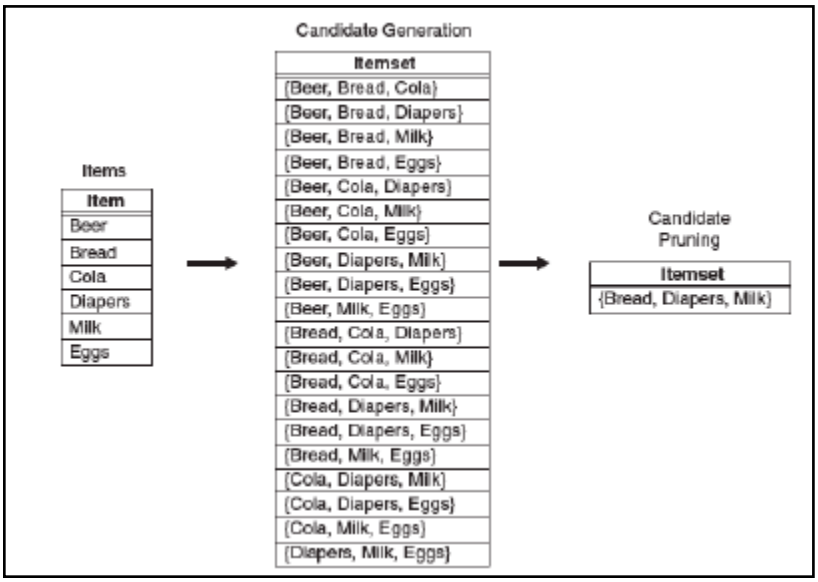


Figure: A brute-force method for generating candidate 3itemsets.

F $_{k-1}$ X F$_1$ Method: An alternative method for candidate generation is to extend each frequent *(k* - l)-itemset with other frequent items. The following Figure illustrates how a frequent 2-itemset such as {Butter, Diapers} can be augmented with a frequent item such as Bread to produce a candidate 3-itemset {Butter, Diapers, Bread}.

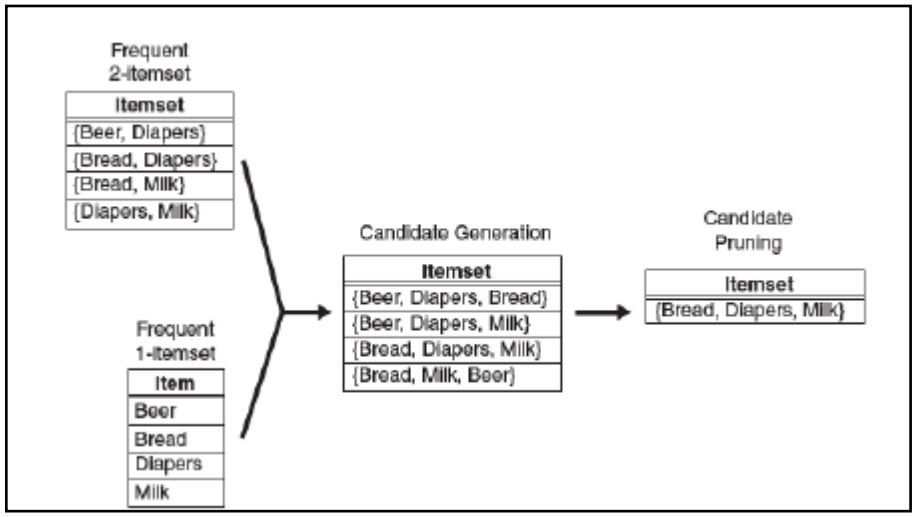
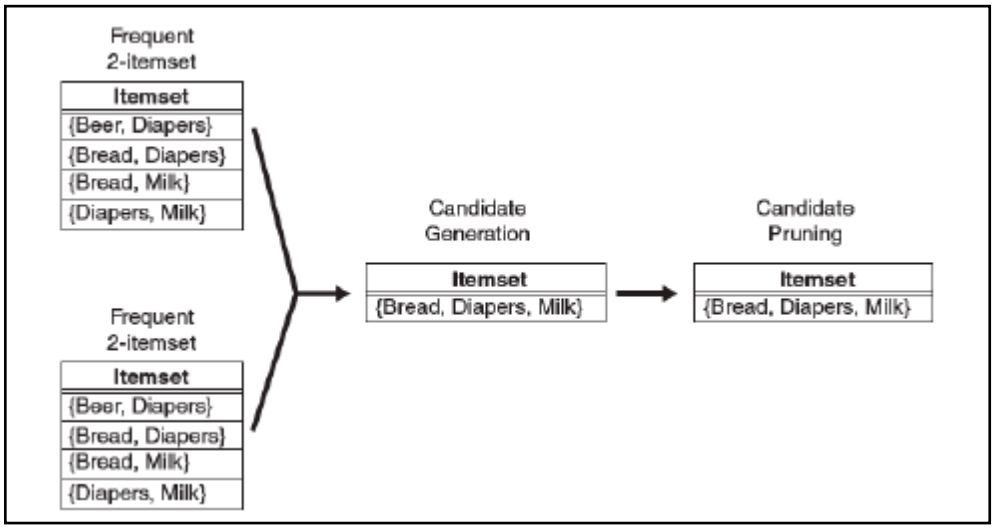


Figure: Generating and pruning candidate k itemsets by merging a frequent *(k - 1)*-itemset with a frequent item.

Note that some of the candidates are unnecessary because their subsets are infrequent. While this procedure is a substantial improvement over the brute-force method, it can still produce a large number of unnecessary candidates. For example, the candidate itemset obtained by merging {Butter, Diapers} with {Milk} is unnecessary because one of its subsets, {Butter, Milk}, is infrequent.

F $_{k-1}$ x F $_{k-1}$ Method: The candidate generation procedure in the apriori-gen function merges a pair of frequent *(k* -1)-itemsets only if their first *k* - 2 items are identical. Let A = $\{a_1, a_2, \ldots, a_{k-1}\}$ and *B* = $\{b_1, b_2, \ldots, b_{k-1}\}$ be a pair of frequent *(k* - 1) itemsets. *A* and *B* are merged if they satisfy the following conditions:

$a_i = b_i$ (for $i = 1, 2, \ldots, k-2$) and $a_{k-1} \neq b_{k-1}$.

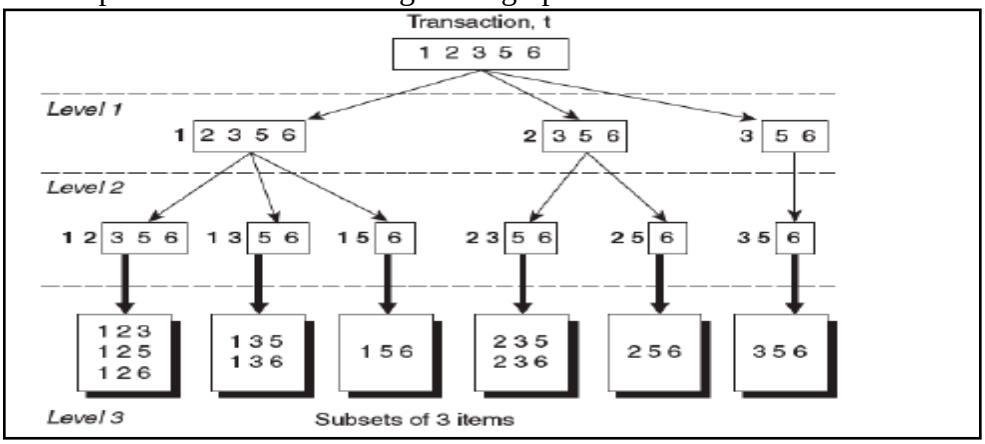


In the above Figure the frequent itemsets {Bread, Diapers} and {Bread, Milk} are merged to form a candidate 3-itemset {Bread, Diapers, Milk}. The algorithm does not have to merge {Butter, Diapers} with {Diapers, Milk} because the first item in both itemsets is different. Indeed, if {Butter, Diapers, Milk} is a viable candidate, it would have been obtained by merging {Butter, Diapers} with {Butter, Milk} instead.

5. Support Counting

One approach for doing this is to compare each transaction against every candidate itemset and to update the support counts of candidates contained in the transaction. This approach is computationally expensive, especially when the numbers of transactions and candidate itemsets are large.

An alternative approach is to enumerate the itemsets contained in each transaction and use them to update the support counts of their respective candidate itemsets. To illustrate, consider a transaction *t* that contains five items, {1, 2, 3,5, 6}. There are 10 itemsets of size 3 contained in this transaction. The following Figure shows a systematic way for enumerating the 3-itemsets contained in *t.* assuming that each itemset keeps its items in increasing lexicographic order.



6. Rule Generation

Here we describe how to extract association rules efficiently from a given frequent itemset. Each frequent k-itemset, Y, can produce up to 2^{k-2} association rules, ignoring rules that have empty antecedents or consequents ($\varphi \rightarrow Y$) or ($Y \rightarrow \varphi$).

An association rule can be extracted by partitioning the itemset Y into two non-empty subsets, *X* and Y - *X*, such that $X \rightarrow Y - X$ satisfies the confidence threshold.

Example: Let $X = \{1, 2, 3\}$ be a frequent itemset. There are six candidate association rules that can be generated from *X:* {1,2} => {3}, {1, 3} => {2}, {2,3} => {1}, {1=>{2,3}, {2} => {1,3}, and {3} => {1,2}. As each of their support is identical to the support for *X*, the rules must satisfy the support threshold.

Theorem: *If a rule* $X \rightarrow Y - X$ *does not satisfy the confidence threshold, then any* rule $X' \rightarrow Y - X'$, *where X' is a subset of X, must not satisfy the confidence threshold as well.*
For example, if *{acd}* $\rightarrow$ *{b}* and *{abd}* $\rightarrow$ *{c}* are high-confidence rules, then the candidate rule *{ad}* $\rightarrow$*{be}* is generated by merging the consequents of both rules. The following Figure shows a lattice structure for the association rules generated from the frequent itemset *{a, b,* c, *d}.* If any node in the lattice has low confidence, then according to above Theorem, the entire sub graph spanned by the node can be pruned immediately. Suppose the confidence for *{bcd}*$\rightarrow${*a*} is low. All the rules containing item *a* in its consequent, including *{cd}*$\rightarrow${*ab*}, *{bd}* $\rightarrow$ *{ae}*, *{be}*$\rightarrow${*ad*}, and *{d}* $\rightarrow$ *{abe}* can be discarded.

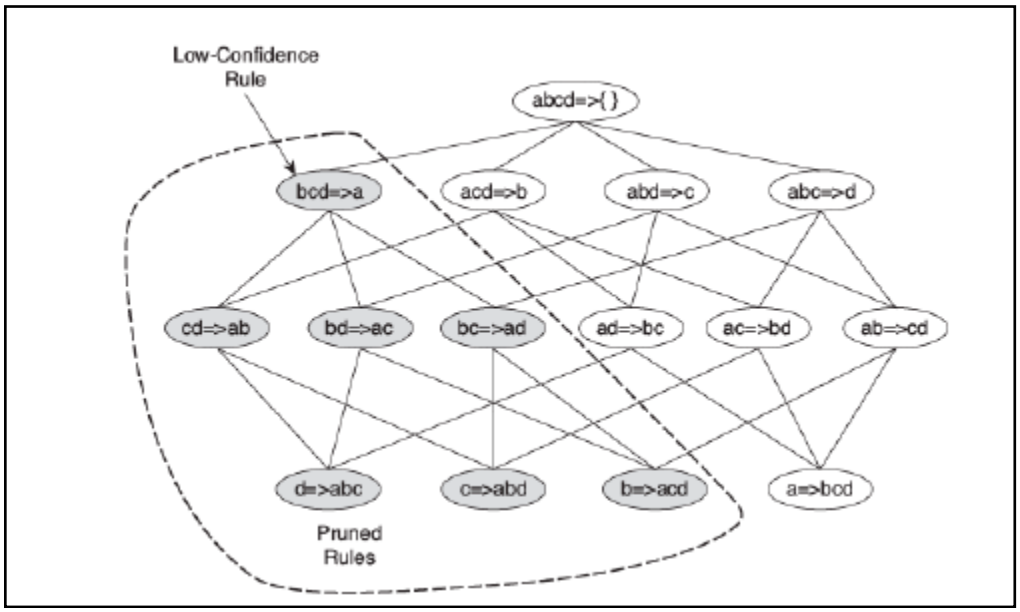


Figure: Pruning of association rules using the confidence measure.

7. Compact Representation of Frequent Itemsets

It is useful to identify a small representative set of itemsets from which all other frequent itemsets can be derived. Two such representations are as follows.

Maximal Frequent Item sets

A maximal frequent itemset is defined as a frequent itemset for which none of its immediate supersets are frequent. Consider the item set lattice shown in following Figure. The itemsets in the lattice are divided into two groups: those that are frequent and those that are infrequent. A frequent itemset order, which is represented by a dashed line, is also illustrated in the diagram. Every itemset located above the border is frequent; while those located below the border (the shaded nodes) are infrequent. Among the itemsets residing near the border, *{a, d}*, *{a,* c, e}, and *{b,* c, *d,* e} are considered to be maximal frequent itemsets because their immediate supersets are infrequent. An itemset such as *{a, d}* is maximal frequent because all of its immediate supersets, *{a, b, d}*, *{a,* c, *d},* and *{a,d,e},* are infrequent.

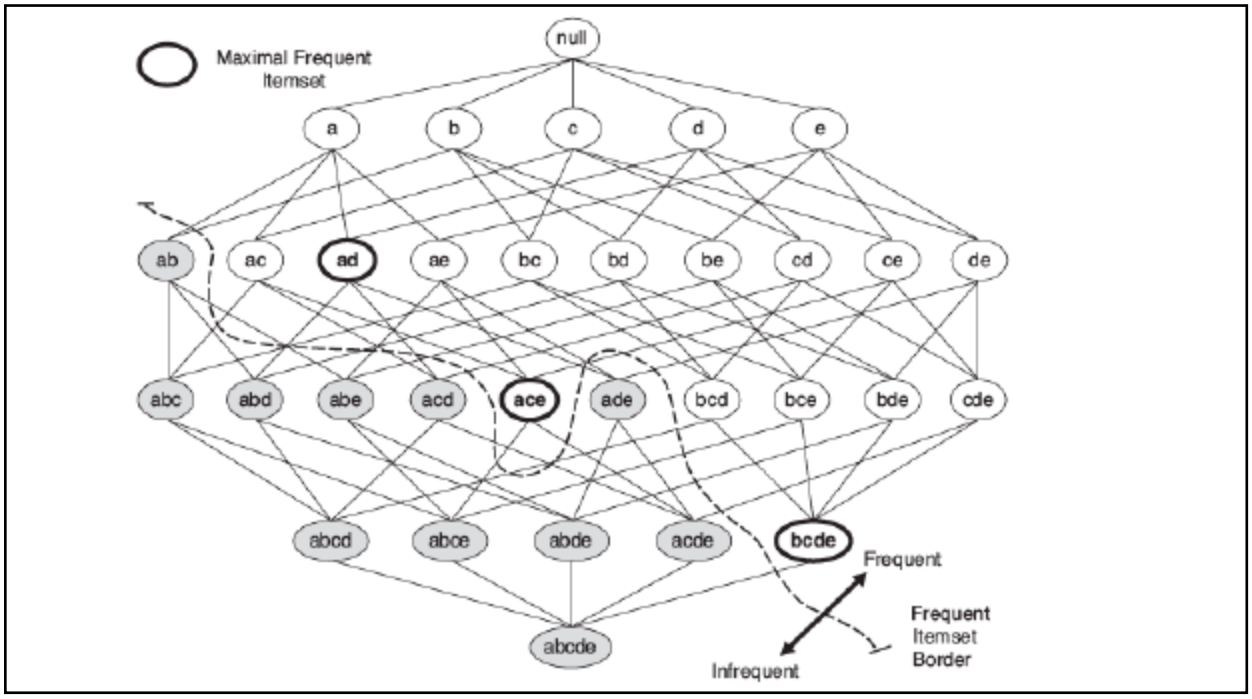


Figure Maximal frequent itemset.

Closed Frequent Item sets: An item set *X* is closed if none of its immediate supersets has exactly the same support count as *X*

Examples of closed itemsets are shown in following Figure. To better illustrate the support count of each itemset, we have associated each node (itemset) in the lattice with a list of its corresponding transaction IDs.

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

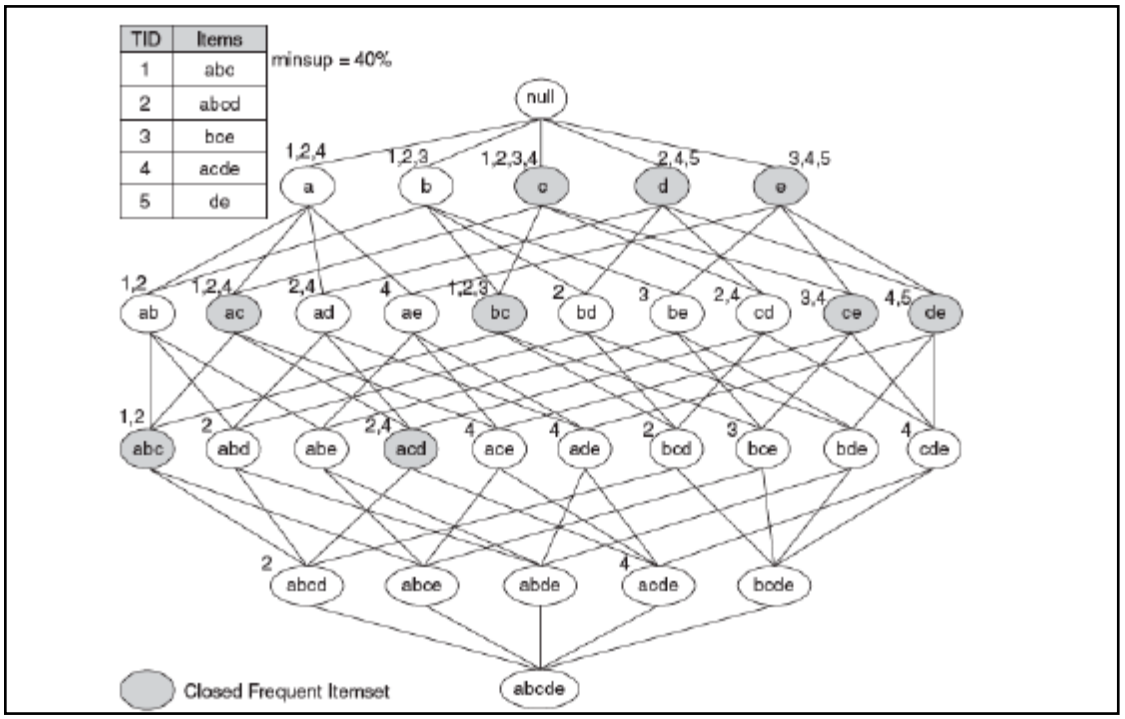


Figure An example of the closed frequent itemsets (with minimum support count equal to 40%).

8. FP-Growth Algorithm

This section presents an alternative algorithm called FP-growth that takes a radically different approach to discovering frequent itemsets. The algorithm does not subscribe to the generate-and-test paradigm of *Apriori.* Instead, it encodes the data set using a compact data structure called an FP- tree and extracts frequent itemsets directly from this structure.

FP-Tree Representation

An FP-tree is a compressed representation of the input data. It is constructed by reading the data set one transaction at a time and mapping each transaction onto a path in the FP-tree. As different transactions can have several items in common, their paths may overlap. The more the paths overlap with one another, the more compression we can achieve using the FP-tree structure.

1. After reading the first transaction, {a, b}, the nodes labeled as a and b are created. A path is then formed from null →a→b to encode the transaction. Every node along the path has a frequency count of 1.
2. After reading the second transaction, {b, c, d}, a new set of nodes is created for items b, c, and d. A path is then formed to represent the transaction by connecting the nodes null → b → c → d.
3. This process continues until every transaction has been mapped onto one of the paths given in the FP-tree. The resulting FP-tree after reading all the transactions is shown at the bottom of Figure.

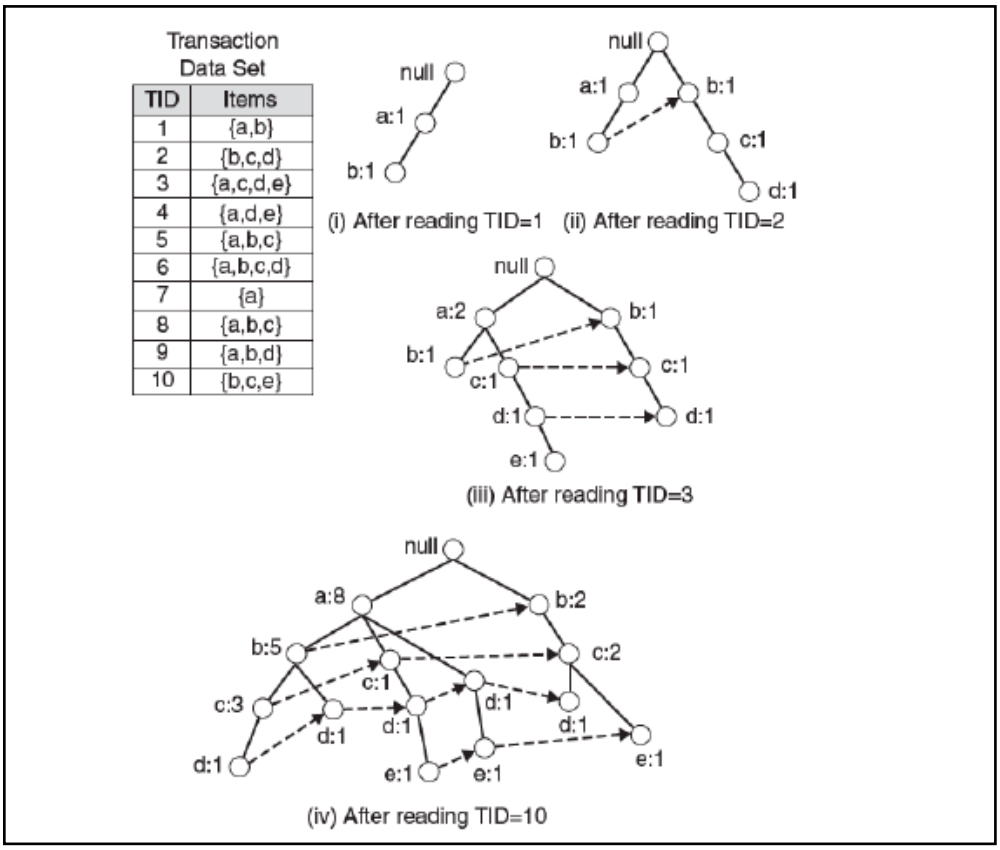


Figure: Construction of an FP-tree.

Frequent Item set Generation in FP-Growth Algorithm

FP-growth is an algorithm that generates frequent itemsets from an FP-tree by exploring the tree in a bottom-up fashion. Given the example tree is shown in above Figure, the algorithm looks for frequent itemsets ending in e first, followed by *d*, c, *b*, and finally, *a*. We can derive the frequent itemsets ending with a particular item. The extracted paths are shown in the following Figure

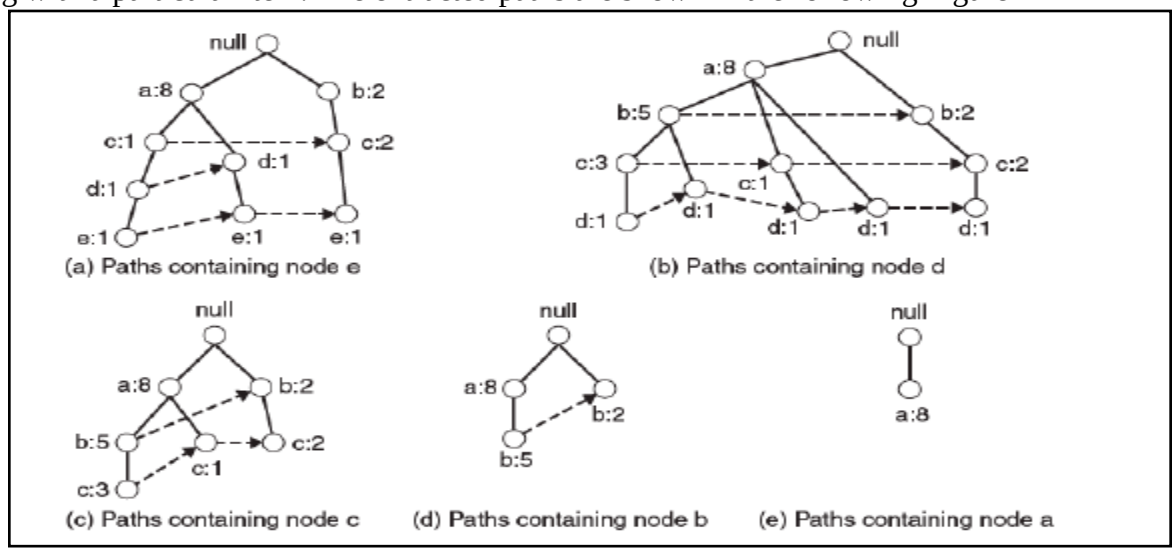


Figure: Decomposing the frequent item set ending in e, d, e, b, and a

FP-growth finds all the frequent itemsets ending with a particular suffix by employing a divide-and-conquer strategy to split the problem into smaller subproblems..

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

FP-growth is an interesting algorithm because it illustrates how a compact representation of the transaction data set helps to efficiently generate frequent

Frequently Asked Questions

Short Answer Questions

1. Define association analysis?
2. Binary Representation of Market Basket Analysis?
3. Differentiate Apriori Principle and Apriori Algorithm?
4. Define Maximal Frequent Itemsets?
5. Define Closed Frequent Itemsets?

Long Answer Questions

1. Explain Apriori Principle with an example?
2. Briefly explain Frequent Item Set generation using apriori algorithm?
3. Write a short note on Rule generation with an example?
4. Draw and explain compact representation of frequent item sets?
5. Explain FP-Growth Algorithm with an example?

Exercise Problems

1. Consider the data set shown in Table

Customer_ID	Transaction_ID	Items_Bought
1	1	{a, d, e}
1	24	{a, b, c, e}
2	12	{a, b, d, e}
2	31	{a, c, d, e}
3	15	{b, c, e}
3	22	{b, d, e}
4	29	{c, d}
4	40	{a, b, c}
5	33	{a, d, e}
5	38	{a, b, e}

(a) Compute the support for itemsets {e}, {b, d}, and {b, d, e} by treating each transaction ID as a market basket. **Answer:** s ({e}) = 8/10= 0.8, s ({b, d}) =2/10= 0.2, s ({b, d, e}) =2/10= 0.2

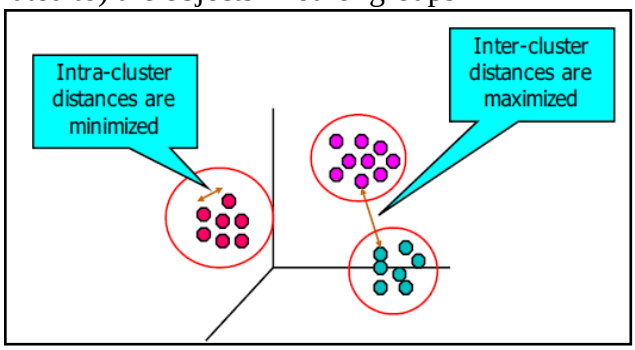
(b) Use the results in part (a) to compute the confidence for the association rules {b, d} $\longrightarrow$ {e} and {e} $\rightarrow$ {b, d}. Is confidence a symmetric measure?
Answer: c (bd $\rightarrow$ e) = 0.2/0.2= 100%, c (e $\rightarrow$ bd) = 0.2/0.8= 25%

UNIT- VI

Cluster Analysis: Basic Concepts and Algorithms: Overview: What Is Cluster Analysis? Different Types of Clustering, Different Types of Clusters; K-means: The Basic K-means Algorithm, K-means Additional Issues, Bisecting K-means, Strengths and Weaknesses; Agglomerative Hierarchical Clustering: Basic Agglomerative Hierarchical Clustering Algorithm DBSCAN: Traditional Density Center-Based Approach, DBSCAN Algorithm, Strengths and Weaknesses.

1. What Is Cluster Analysis

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Applications of Cluster Analysis

- Group related documents for browsing,
- Group genes and proteins that have similar functionality, or
- Group stocks with similar price fluctuations

What is not Cluster Analysis?

● Simple segmentation

Dividing students into different registration groups alphabetically, by last name

● Results of a query

Groupings are a result of an external specification

Clustering is a grouping of objects based on the data

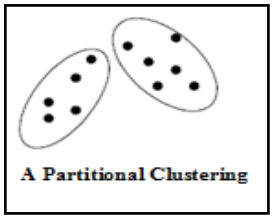
● Supervised classification: Have class label information

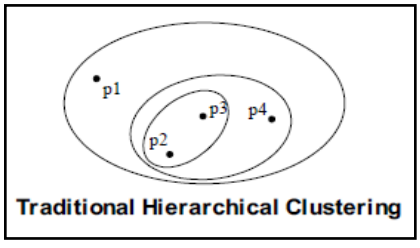
● Association Analysis: Local vs. global connections

2. Different Types of Clustering: A clustering is a set of clusters. There are two types of Clustering techniques. Hierarchical Clustering and Partitional Clustering, important distinction between hierarchical and partitional sets of clusters

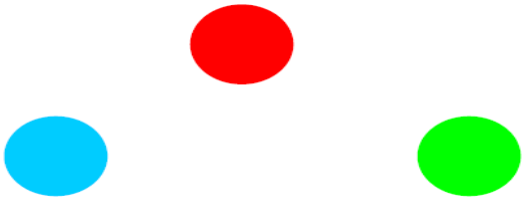
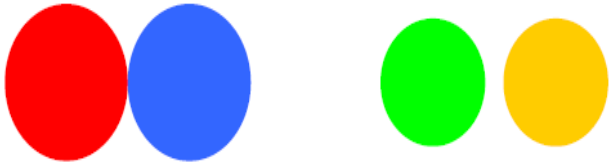
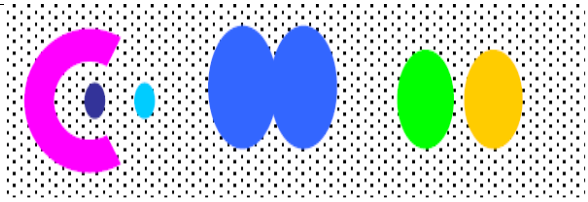
● Partitional Clustering– A division of data objects into non--overlapping subsets (clusters) such that each data object is in exactly one subset

● Hierarchical clustering– A set of nested clusters organized as a hierarchical tree





3. Different Types of Clusters

Types of Clusters	Example Cluster
Well--separated clusters: A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.	 3 well-separated clusters
Center--based clusters: A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster – The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster	 4 center-based clusters
Contiguous clusters: A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.	 8 contiguous clusters
Density--based clusters: A cluster is a dense region of points, which is separated by low--density regions, from other regions of high density. – Used when the clusters are irregular or intertwined, and when noise and outliers are present.	 6 density-based clusters
Property or Conceptual Clusters: Finds clusters that share some common property or represent a particular concept.	 2 Overlapping Circles

4. K-means: The Basic K-means Algorithm

The K-means clustering technique is simple, and we begin with a description of the basic algorithm. We first choose *K* initial centroids, where *K* is a user specified parameter, namely, the number of clusters desired.

- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

Basic K-means Algorithm

Step 1: Select K points as initial centroids
Step 2: **repeat**
Step 3: Form K clusters by assigning each point to its closest centroid
Step 4: Recompute the centroid of each cluster
Step 5: **Until** Centroids do not change

K-Means Algorithm Steps:

- Step 1: Assume the two mean points for the given cluster.
- Step 2: Using the Euclidean distance formula, calculate the distance.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance } [(x), (a)] = \sqrt{(x-a)^2}$$

- Step 3: Tabulate the data with reference to the clusters.
- Step 4: Display the clusters.
- Step 5 : recalculate the mean for the new cluster and repeat the steps 2 to 4.
- Step 6: similar repetitive clusters are formed, then stop.

Example:

- Apply K-Mean Clustering for the following data sets for two clusters. Dataset {2,4,10,12,3,20,30,11,25}

- Step 1: Assume the two mean points for the given cluster. Let us assume that $M_1 = 4$ and $M_2 = 11$.
- Step 2: Using the Euclidean distance formula, calculate the distance.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance } [(x), (a)] = \sqrt{(x-a)^2}$$

Dataset {2,4,10,12,3,20,30,11,25}

$$\text{Distance } [(x), (a)] = \sqrt{(x-a)^2}$$

Dataset {2,4,10,12,3,20,30,11,25}

$M_1 = 4$ and $M_2 = 11$.

where D_1 is the distance from M1

D_2 is the distance from M2

D

$$D_1(2) = \sqrt{(x-a)^2} = \sqrt{(2-4)^2} = 2$$

$$D_2(2) = \sqrt{(x-a)^2} = \sqrt{(2-11)^2} = 9$$

$M_1 = 4$ and $M_2 = 11$.

Distance [(x), (a)] = $\sqrt{(x-a)^2}$

	2	4	10	12	3	20	30	11	25
D_1	2	0	6	8	1	16	26	7	21
D_2	9	7	1	1	8	9	19	0	14
Cluster	C1	C1	C2	C2	C1	C2	C2	C2	C2

The clusters are

C1 = {2,4,3}

C2 = {10,12,20,30,11,25}

- Step 4: Display the clusters.

  The clusters are

  C1 = {2,4,3}

  C2 = {10,12,20,30,11,25}

- Step 5 : Recalculate the mean for the new cluster and repeat the steps 2 to 4.

  Calculate the mean for the above two clusters.

  $M_1 = \frac{2+4+3}{3} = 3$

  $M_2 = \frac{10+12+20+30+11+25}{6} = 18$

The new clusters are

$M_1 = 3$ and $M_2 = 18$.

Distance [(x), (a)] = $\sqrt{(x-a)^2}$

	2	4	10	12	3	20	30	11	25
D_1	1	1	7	9	0	17	27	8	22
D_2	16	14	8	6	15	2	12	7	7
Cluster	C1	C1	C1	C2	C1	C2	C2	C2	C2

C1 = {2,4,3,10}

C2 = {12,20,30,11,25}

The clusters are

$C1 = \{2,4,3,10\}$

$C2 = \{12,20,30,11,25\}$

Calculate the mean for the above two clusters.

$$M_1 = \frac{2+4+3+10}{4} = 4.75$$

$$M_2 = \frac{12+20+30+11+25}{5} = 19.6$$

The new mean is.

$M_1 = 4.75$ and $M_2 = 19.6$.

Distance [(x), (a)] = $\sqrt{(x-a)^2}$

	2	4	10	12	3	20	30	11	25
D_1	2.75	0.75	5.25	7.25	1.75	15.25	25.25	6.25	20.25
D_2	17.6	15.6	9.6	7.6	16.6	0.4	10.4	8.6	5.4
Cluster	C1	C1	C1	C1	C1	C2	C2	C1	C2

$C1 = \{2,4,3,10,12,11\}$

$C2 = \{20,30,25\}$

The clusters are

$C1 = \{2,4,3,10,12,11\}$

$C2 = \{20,30, 25\}$

Calculate the mean for the above two clusters.

$$M_1 = \frac{2+4+3+10+11+12}{6} = 7$$

$$M_2 = \frac{20+30+25}{3} = 25$$

The new mean is.

$M_1 = 7$ and $M_2 = 25$.

Distance [(x), (a)] = $\sqrt{(x-a)^2}$

	2	4	10	12	3	20	30	11	25
D_1	5	3	3	5	4	13	23	4	18
D_2	23	21	15	13	22	5	5	14	0
Cluster	C1	C1	C1	C1	C1	C2	C2	C1	C2

The final clusters

C1 = {2,4,3,10,12,11}

C2 = {20,30,25}

Evaluating K-means Clusters

- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

- x is a data point in cluster C_i and m_i is the representative point for cluster C_i
  - can show that m_i corresponds to the center (mean) of the cluster
- Given two sets of clusters, we prefer the one with the smallest error
- One easy way to reduce SSE is to increase K, the number of clusters
  - A good clustering with smaller K can have a lower SSE than a poor clustering with higher K

Limitations of K-means

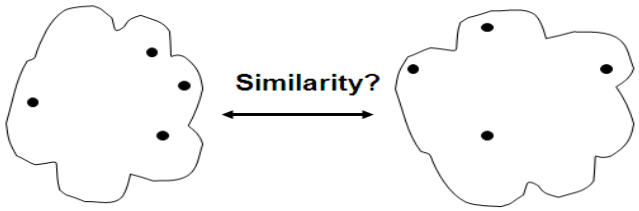
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

5.Bisecting K-means

- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering

1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3: Select a cluster from the list of clusters
4: **for** $i = 1$ to *number_of_iterations* **do**
5: Bisect the selected cluster using basic K-means
6: **end for**
7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains K clusters

How to Define Inter-Cluster Distance



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

- Starts with one cluster , individual item in its own cluster and iteratively merge clusters until all the items belong to one cluster.
- Bottom up approach is followed to merge the clusters together.
- Dendrograms are pictorially used to represent the HAC.

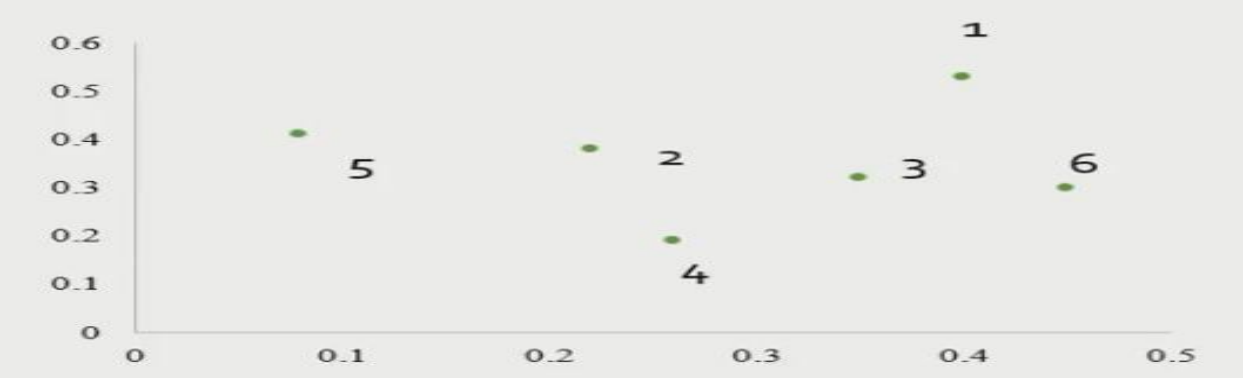
Single Linkage	This is the distance between the closest members of the two clusters.
Complete Linkage	This is the distance between the members that are farthest apart.
Average Linkage	This method involves looking at the distances between all pairs and averages all of these distances. This is also called Unweighted Pair Group Mean Averaging.

DENDROGRAM

- A tree like structure which represents hierarchical technique.
  - ✓ Leaf- Individual.
  - ✓ Root – One cluster.
- A cluster at level 1, is the merger of its child cluster at level i + 1.

Find the clusters using single link technique. Use Euclidean distance, and draw the dendrogram.

	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30



- Calculate Euclidean distance, create the distance matrix.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

Distance (P1,P2) $= \sqrt{(0.40-0.22)^2+(0.53-0.38)^2}$

(0.40,0.53), (0.22,0.38) $= \sqrt{(0.18)^2+(0.15)^2}$

$= \sqrt{0.0324+0.0225}$

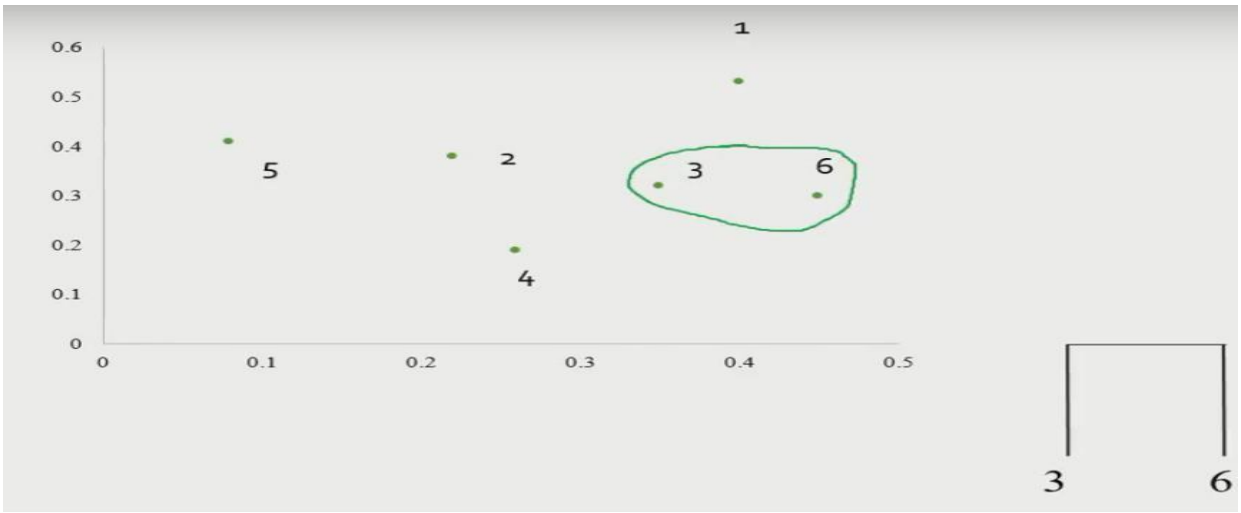
$= \sqrt{0.0549}$

$= 0.23$

- The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.24	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0



- To update the distance matrix MIN[dist(P3,P6),P1)]
- MIN(dist(P3,P1), (P6,P1))
  = min[(0.22,0.23)]
  = 0.22
- To update the distance matrix MIN[dist(P3,P6),P2)]
- MIN(dist(P3,P2), (P6,P2))
  = min[(0.15,0.25)]
  = 0.15

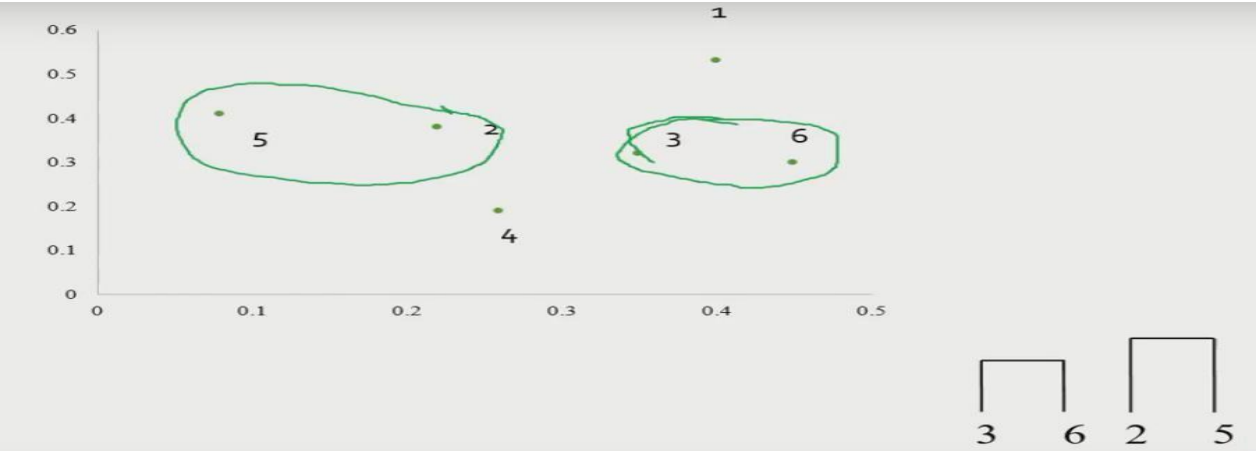
- To update the distance matrix MIN[dist(P3,P6),P4)]
- MIN(dist(P3,P4), (P6,P4))
  = min[(0.15,0.22)]
  = 0.15
- To update the distance matrix MIN[dist(P3,P6),P5)]
- MIN(dist(P3,P5), (P6,P5))
  = min[(0.28,0.39)]
  = 0.28

- The updated distance matrix for cluster P3, P6

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.22	0.15	0		
P4	0.37	0.20	0.15	0	
P5	0.34	0.14	0.28	0.29	0

- The distance matrix is

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.22	0.15	0		
P4	0.37	0.20	0.15	0	
P5	0.34	0.14	0.28	0.29	0



- To update the distance matrix MIN[dist(P2,P5),P1)]
- MIN[dist(P2,P1), (P5,P1)]

  = min[(0.23,0.34)]

  = 0.23
- To update the distance matrix MIN[dist(P2,P5),(P3,P6)]
- MIN[dist(P2,(P3,P6)), (P5,(P3,P6))]

  = min[(0.15,0.28)]

  = 0.15

- To update the distance matrix MIN[dist(P2,P5),P4)]
- MIN[dist(P2,P4), (P5,P4)]

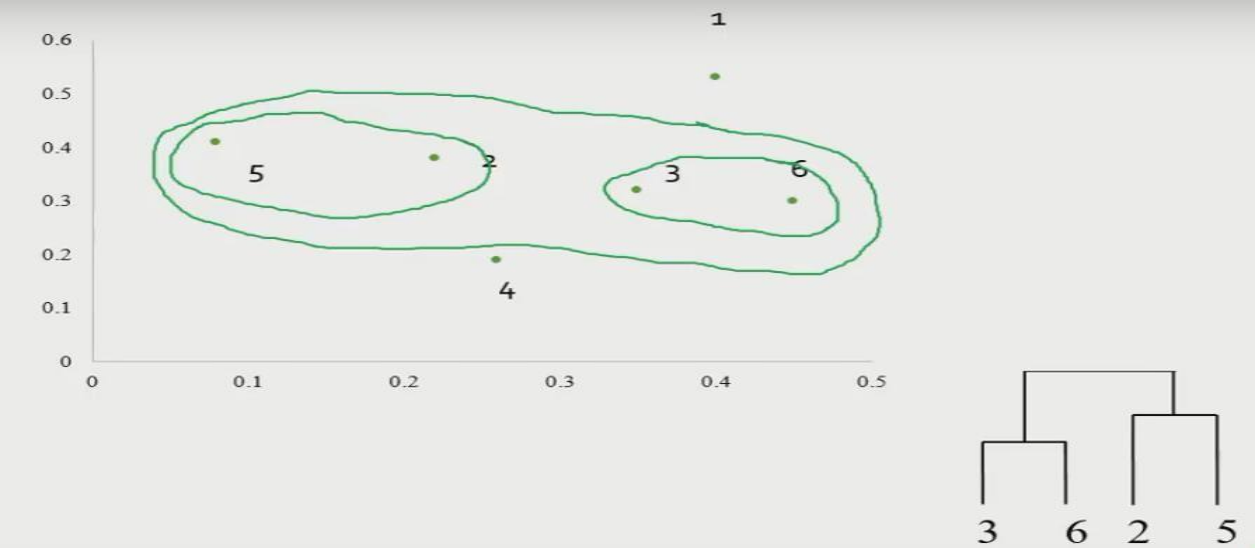
  = min[(0.20,0.29)]

  = 0.20

- The updated distance matrix for cluster P2,P5

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.23	0		
P3,P6	0.22	0.15	0	
P4	0.37	0.20	0.15	0

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.23	0		
P3,P6	0.22	0.15	0	
P4	0.37	0.20	0.15	0



- To update the distance matrix MIN[dist((P2,P5),(P3,P6)),P1]
- MIN[dist((P2,P5),P1), ((P3,P6),P1)]

  = min[(0.23,0.22)]

  = 0.22

- To update the distance matrix MIN[dist((P2,P5),(P3,P6)),P4]
- MIN[dist((P2,P5),P4), ((P3,P6),P4)]

  = min[(0.20,0.15)]

  = 0.15

- The updated distance matrix for cluster P2,P5,P3,P6

	P1	P2,P5,P3,P6	P4
P1	0		
P2,P5,P3,P6	0.22	0	
P4	0.37	0.15	0

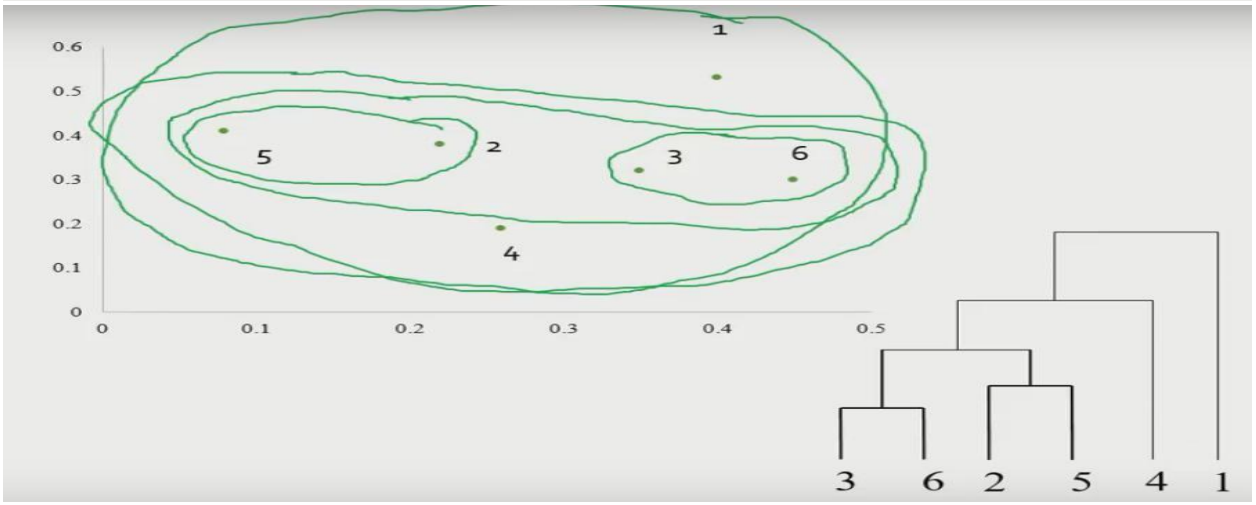
- To update the distance matrix MIN[dist(P2,P5,P3,P6),P4]
- MIN[dist((P2,P5,P3,P6),P1), (P4,P1)]

  = min[(0.22,0.37)]

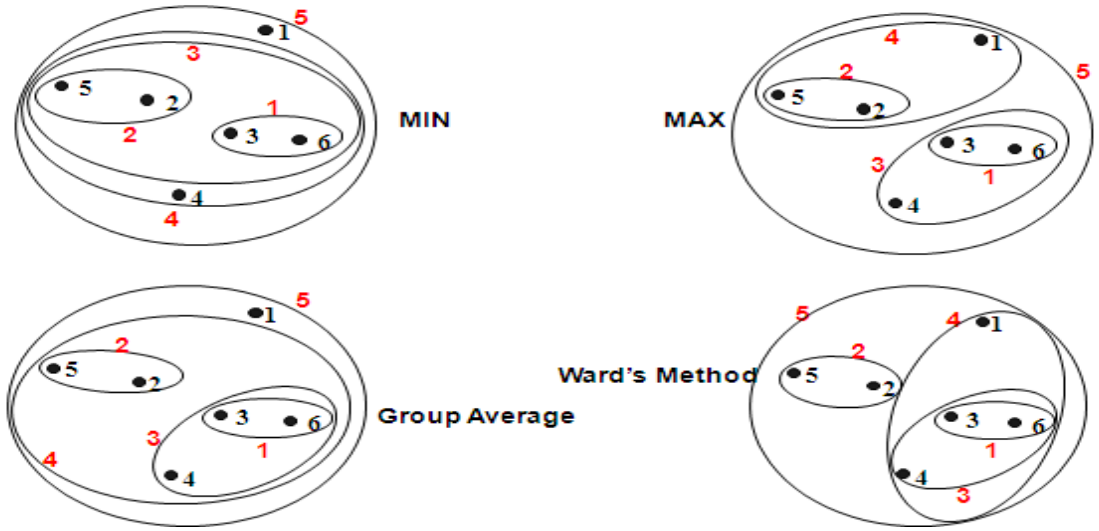
  = 0.22

- The updated distance matrix for cluster P2,P5,P3,P6,P4

	P1	P2,P5,P3,P6,P4
P1	0	
P2,P5,P3,P6,P4	0.22	0



Hierarchical Clustering: Comparison

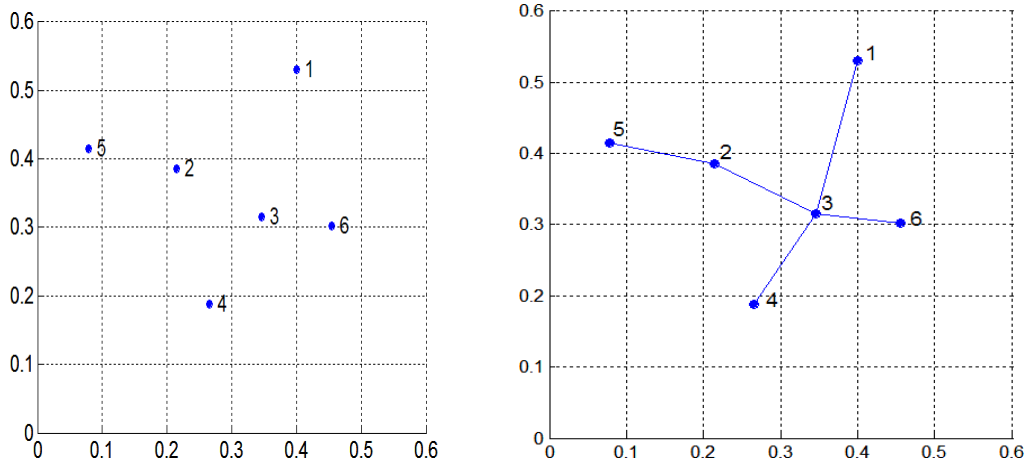


7. MST: Divisive Hierarchical Clustering

- Build MST (Minimum Spanning Tree)
  - Start with a tree that consists of any point
  - In successive steps, look for the closest pair of points (p, q) such that one point (p) is in the current tree but the other (q) is not
  - Add q to the tree and put an edge between p and q

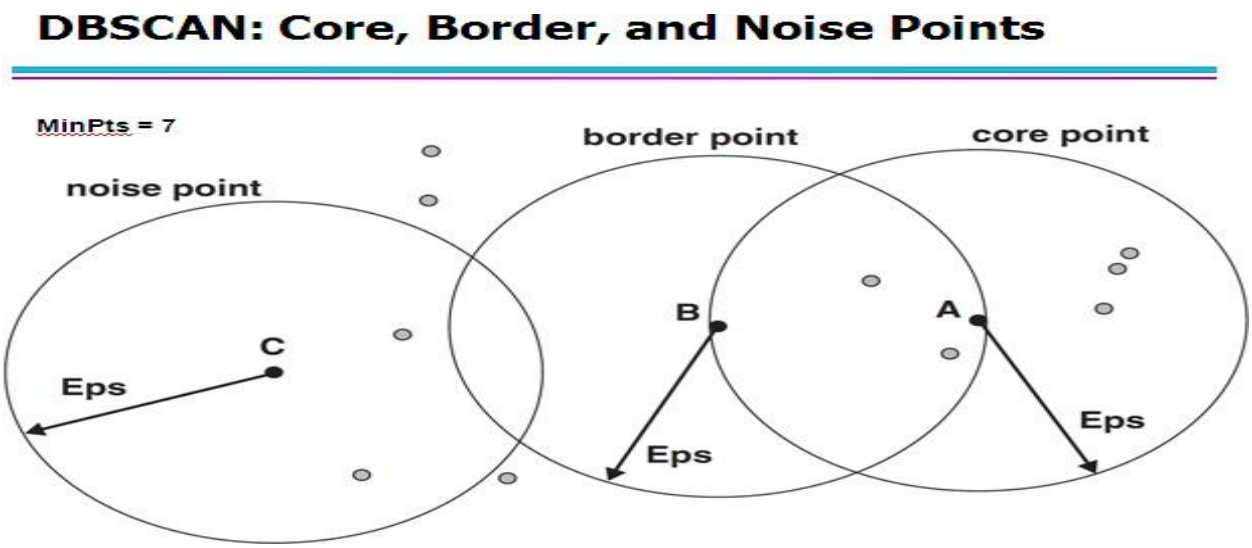
Algorithm 7.5 MST Divisive Hierarchical Clustering Algorithm

1: Compute a minimum spanning tree for the proximity graph.
2: **repeat**
3: Create a new cluster by breaking the link corresponding to the largest distance (smallest similarity).
4: **until** Only singleton clusters remain



8.DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has at least a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
    - Counts the point itself
  - A border point is not a core point, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point



- Eliminate noise points
- Perform clustering on the remaining points

```
current_cluster_label ← 1
for all core points do
  if the core point has no cluster label then
    current_cluster_label ← current_cluster_label + 1
    Label the current core point with cluster label current_cluster_label
  end if
  for all points in the Eps-neighborhood, except i^th the point itself do
    if the point does not have a cluster label then
      Label the point with cluster label current_cluster_label
    end if
  end for
end for
```

Frequently Asked Question

Short Answer Questions

1. Define Cluster Analysis?
2. What are Different Types of Clustering?
3. Explain Different Types of Clusters methods?
4. Explain K-means Additional Issues, Strengths and Weaknesses?
5. Explain about Strengths and Weaknesses of DBSCAN algorithm?

Long Answer Questions

1. Write and Explain about K-means, The Basic K-means Algorithm?
2. Write algorithm for Bisecting K-means and K-means as an Optimization Problem?
3. Explain about Basic Agglomerative Hierarchical Clustering Algorithm with an example?
4. Explain about DBSCAN Algorithm?

Exercise Problems

1. Consider five points {X1,X2,X3,X4,X5}with the following coordinates as a two dimensional sample for clustering : X1= (0.5, 2.5); X2= (0, 0); X3= (1.5, 1); X4= (5, 1); X5= (6, 2); Illustrate the K-means partitioning algorithms using the above data set.

(a) *Select an initial partition of k clusters containing randomly chosen samples and compute their centroids* Say, one selects two clusters and assigns to cluster C1 = (x1, x2, x4) and C2 = (x3, x5). Next, the centroids of the two clusters are determined:

M1 = {(0 + 0 + 5)/3, (2 + 0 + 0)/3} = {1.66, 0.66}
M2 = {(1.5 + 5)/2, (0 + 2)/2} = {3.25, 1.0}

(b) Compute the within-cluster variations:

$e_1^2 = [(0 - 1.66)^2 + (2 - 0.66)^2] + [(0 - 1.66)^2 + (0 - 0.66)^2] + [(5 - 1.66)^2 + (0 - 0.66)^2] = 19.36$

$e_2^2 = [(1.5 - 3.25)^2 + (0 - 1)^2] + [(5 - 3.25)^2 + (2 - 1)^2] = 8.12$

and the total error $E^2 = e_1^2 + e_2^2 = 19.36 + 8.12 = 27.48$

(c) Generate a new partition by assigning each sample to the closest cluster center

For example, the distance of x1 from the centroid M1 is d (M1, x1) = $(1.662 + 1.342)^{1/2} = 2.14$, while that for d(M2, x1) = 3.40. Thus, object x1 will be assigned to the group which has the smaller distance, namely C1. Similarly, one can compute distance measures of all other objects, and assign each object as shown in Table.

(d) Compute new cluster centers as centroids of the clusters. The new cluster centers are M1 = {0.5, 0.67} and M2 = {5.0, 1.0}

(e) Repeat steps (b) and (c) until an optimum value is found or until the cluster membership stabilizes

Table 8.9 Distance measures of the five objects with respect to the two groups

$d(M_1, x_1) = 2.14$	$d(M_2, x_1) = 3.40$	So assign $\Rightarrow x_1 \in C_1$
$d(M_1, x_2) = 1.79$	$d(M_2, x_2) = 3.40$	So assign $\Rightarrow x_2 \in C_1$
$d(M_1, x_3) = 0.83$	$d(M_2, x_3) = 2.01$	So assign $\Rightarrow x_3 \in C_1$
$d(M_1, x_4) = 3.41$	$d(M_2, x_4) = 2.01$	So assign $\Rightarrow x_4 \in C_2$
$d(M_1, x_5) = 3.60$	$d(M_2, x_5) = 2.01$	So assign $\Rightarrow x_5 \in C_2$

For the new clusters $C_1=(x_1, x_2, x_3)$ and $C_2=(x_4, x_5)$, the within-cluster variation and the total square errors are: $e_1^2 = 4.17$, $e_2^2 = 2.00$, $E^2 = 6.17$. Thus, the total error has decreased significantly just after one iteration. ■

2. The price of each item in a store is nonnegative. For each of the following cases, identify the kinds of constraint they represent and briefly discuss how to mine such association rules efficiently.

(a) Containing at least one Nintendo game

(b) Containing items the sum of whose prices is less than $150

(c) Containing one free item and other items the sum of whose prices is at least $200

(d) Where the average price of all the items is between $100 and $500

Answer:

(a) Containing at least one Nintendo game

The constraint is succinct and monotonic. This constraint can be mined efficiently using FP-growth as follows.

• All frequent Nintendo games are listed at the end of the list of frequent items L.

• Only those conditional pattern bases and FP-trees for frequent Nintendo games need to be derived from the global FP-tree and mined recursively.

(b) Containing items the sum of whose prices is less than $150

The constraint is antimonotonic. This constraint can be mined efficiently using Apriori as follows. Only candidates the sum of whose prices is less than $150 need to be checked.

(c) Containing one free item and other items the sum of whose prices is at least $200 The constraint is monotonic. (Or, sub constraints "containing one free item" and "the sum of whose prices is less than
$150" are succinct and monotonic, respectively.) This constraint can be mined efficiently using FP- growth as follows.

• Put all frequent free items at the end of the list of frequent items L.

• Only conditional pattern bases and FP-trees for frequent free items need to be derived from the global FP-tree and mined recursively. Other free items should be excluded from these conditional pattern bases and FP-trees.

• Once a pattern with items the sum of whose prices is at least $200, no further constraint checking for total price is needed in recursive mining.

• A pattern as well as its conditional pattern base can be pruned if the sum of the price of items in the pattern and the frequent ones in the pattern base is less than $200.

(d) Where the average price of all the items is between $100 and $150

The constraint is nonconvertible. (Or, the sub constraints "the average price is at least $100" and "the average price is at most $500" are convertible.) This constraint can be mined efficiently using FP-growth as follows.

• All frequent items are listed in price descending order. (If you use ascending order, you must rewrite the following two steps.)

• A pattern as well as its conditional pattern base can be pruned if the average price of items in the pattern and those frequent ones in pattern base with prices greater than $100 is less than $100.

• A pattern as well as its conditional pattern base can also be pruned if the average price of items in the pattern is greater than $500.

Additional Data- DATA Warehouse

1. **Define Data Warehouse:-**

A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process." The four keywords, subject-oriented, integrated, time-variant, and nonvolatile, distinguish data warehouses from other data repository systems, such as relational database systems, transaction processing systems, and file systems.

- ***Subject-oriented:*** A data warehouse is organized around major subjects rather than concentrating on the day-to-day operations. Hence, data warehouses provide a simple and concise view around particular subject 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, files, and on-line transaction records. Data cleaning and data integration techniques are applied to ensure consistency in data.
- ***Time-variant:*** Data are stored to provide information from a historical perspective (e.g., the past 5-10 years). Every data 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 when compared to the data at 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. **Differentiate OLAP Vs OLTP**

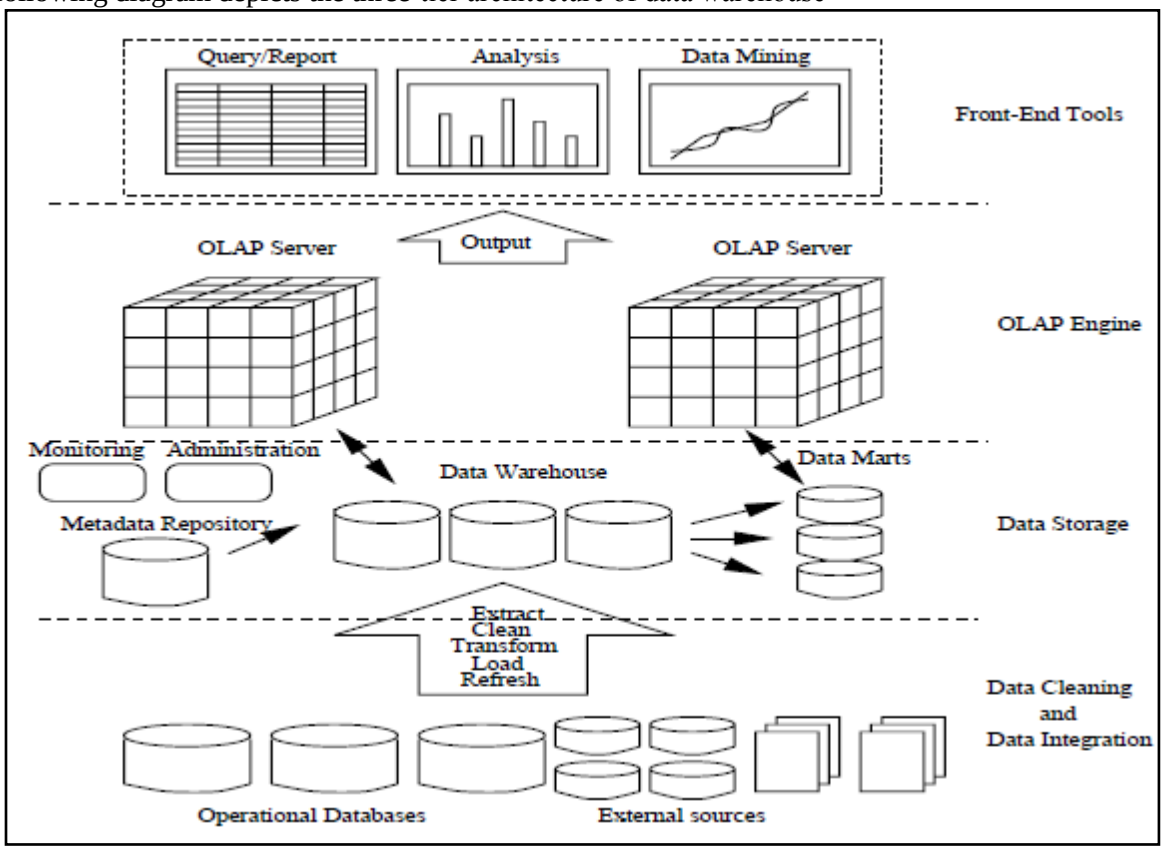
S. No.	Data Warehouse (OLAP)	Operational Database (OLTP)
1	Involves historical processing of information.	Involves day-to-day processing.
2	OLAP systems are used by knowledge workers such as executives, managers and analysts.	OLTP systems are used by clerks, DBAs, or database professionals.
3	Useful in analyzing the business.	Useful in running the business.
4	It focuses on Information out.	It focuses on Data in.
5	Based on Star Schema, Snowflake, Schema and Fact Constellation Schema.	Based on Entity Relationship Model.
6	Contains historical data.	Contains current data.
7	Provides summarized and consolidated data.	Provides primitive and highly detailed data.
8	Provides summarized and multidimensional view of data.	Provides detailed and flat relational view of data.
9	Number or users is in hundreds.	Number of users is in thousands.
10	Number of records accessed is in millions.	Number of records accessed is in tens.
11	Database size is from 100 GB to 1 TB	Database size is from 100 MB to 1 GB.
12	Highly flexible.	Provides high performance.

3. **Draw and Explain Three-Tier Data Warehouse Architecture**

Generally a data warehouses adopts three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** − The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** − In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.
    - By Relational OLAP (ROLAP), which is an extended relational database management system? The ROLAP maps the operations on multidimensional data to standard relational operations.
    - By Multidimensional OLAP (MOLAP) model, which directly implements the multidimensional data and operations?
- **Top-Tier** − this tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.

The following diagram depicts the three-tier architecture of data warehouse −



4. Data Warehousing - Schemas

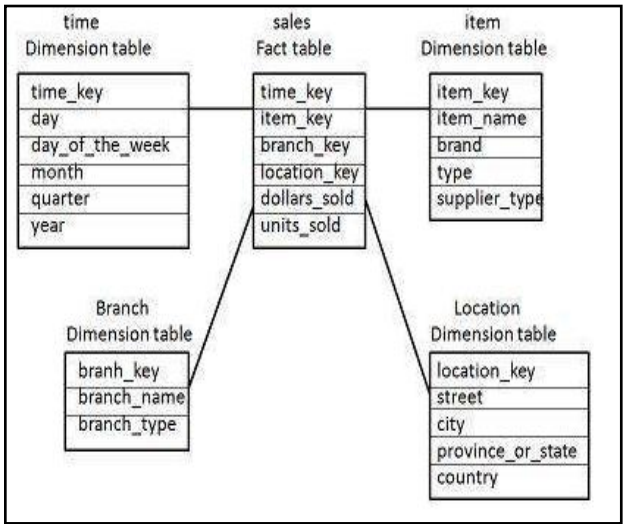
Schema is a logical description of the entire database. It includes the name and description of records of all record types including all associated data-items and aggregates. Much like a database, a data warehouse also requires to maintain a schema. A database uses relational model, while a data warehouse uses Star, Snowflake, and Fact Constellation schema. In this chapter, we will discuss the schemas used in a data warehouse.

Star Schema

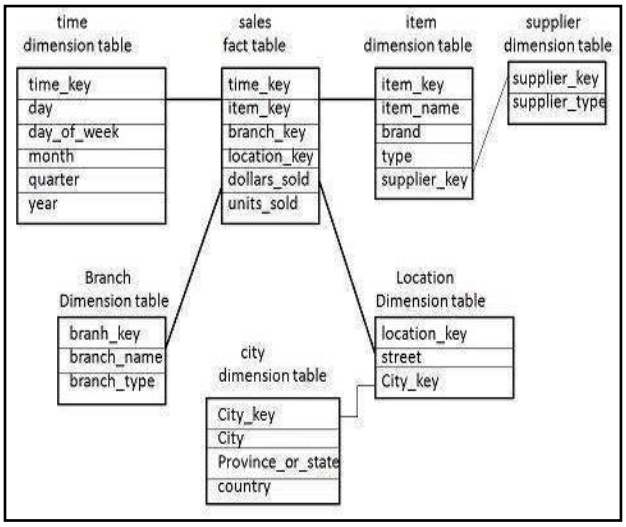
- Each dimension in a star schema is represented with only one-dimension table.
- This dimension table contains the set of attributes.
- The following diagram shows the sales data of a company with respect to the four dimensions, namely time, item, branch, and location.
- There is a fact table at the center. It contains the keys to each of four dimensions.
- The fact table also contains the attributes, namely dollars sold and units sold.
- Note − each dimension has only one dimension table and each table holds a set of attributes. For example, the location dimension table contains the attribute set {location_key, street, city, province_or_state, country}. This constraint may cause data redundancy. For example, "Vancouver" and "Victoria" both the cities are in the Canadian province of British Columbia. The entries for such cities may cause data redundancy along the attributes province_or_state and country.

Snowflake Schema

- Some dimension tables in the Snowflake schema are normalized.
- The normalization splits up the data into additional tables.
- Unlike Star schema, the dimensions table in a snowflake schema is normalized. For example, the item dimension table in star schema is normalized and split into two dimension tables, namely item and supplier table.
- Now the item dimension table contains the attributes item_key, item_name, type, brand, and supplier-key.
- The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier_key and supplier_type.
- Note − Due to normalization in the Snowflake schema, the redundancy is reduced and therefore, it becomes easy to maintain and the save storage space.



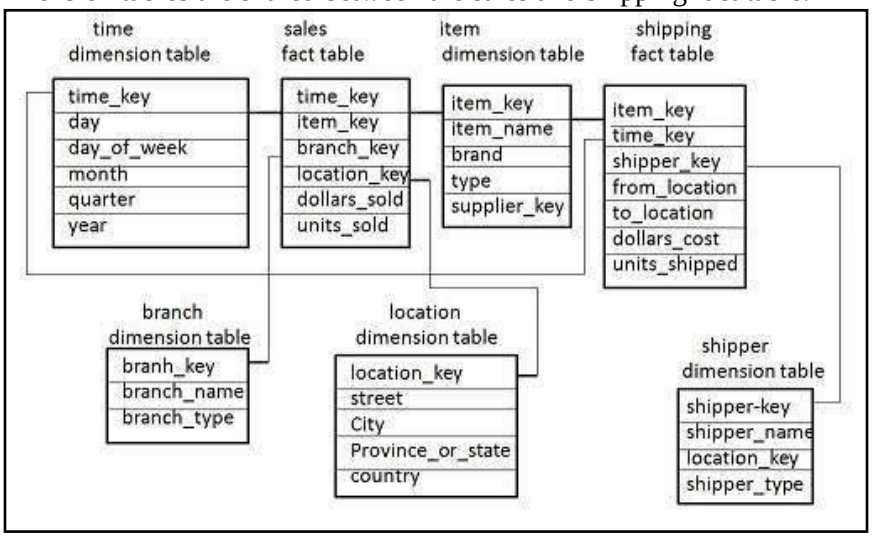
Star Schema



Snow Flake Schema

Fact Constellation Schema

- A fact constellation has multiple fact tables. It is also known as galaxy schema.
- The following diagram shows two fact tables, namely sales and shipping.
- The sales fact table is same as that in the star schema.
- The shipping fact table has the five dimensions, namely item_key, time_key, shipper_key, from_location, to_location.
- The shipping fact table also contains two measures, namely dollars sold and units sold.
- It is also possible to share dimension tables between fact tables. For example, time, item, and location dimension tables are shared between the sales and shipping fact table.



Schema Definition: Multidimensional schema is defined using Data Mining Query Language (DMQL). The two primitives, cube definition and dimension definition, can be used for defining the data warehouses and data marts.

Syntax for Cube Definition

define cube < cube_name > [< dimension-list > }: < measure_list >

Syntax for Dimension Definition

define dimension < dimension_name > as (< attribute_or_dimension_list >)

Star Schema Definition

The star schema that we have discussed can be defined using Data Mining Query Language (DMQL) as follows −

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

Snowflake schema can be defined using DMQL as follows −

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))

5. DATA WAREHOUSING – OLAP

Online Analytical Processing Server OLAP is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information.

Types of OLAP Servers: We have four types of OLAP servers

- Relational OLAP - ROLAP
- Multidimensional OLAP- MOLAP
- Hybrid OLAP- HOLAP
- Specialized SQL Servers

Relational OLAP

ROLAP servers are placed between relational back-end server and client front-end tools. T o store and manage warehouse data, ROLAP uses relational or extended-relational DBMS. ROLAP includes the following

- Implementation of aggregation navigation logic
- Optimization for each DBMS back end
- Additional tools and services

Multidimensional OLAP

MOLAP uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the data set is sparse. Therefore, many MOLAP server uses two levels of data storage representation to handle dense and sparse data sets.

Hybrid OLAP

Hybrid OLAP is a combination of both ROLAP and MOLAP. It offers higher scalability of ROLAP and faster computation of MOLAP. HOLAP servers allow to store the large data volumes of detailed information. T he aggregations are stored separately in MOLAP store.

Specialized SQL Servers

Specialized SQL servers provide advanced query language and query processing support for SQL queries over star and snowflake schemas in a read-only environment.

OLAP operations in the multidimensional data model:- In the multidimensional model, data are organized into multiple dimensions and each dimension contains multiple levels of abstraction defined by concept hierarchies. This organization provides users with the flexibility to view data from different perspectives. Some of the OLAP data cube operations are

- **Roll-up:** The roll-up operation (also called the "drill-up" operation) performs aggregation on a data cube, either by climbing-up a concept hierarchy for a dimension or by dimension reduction. The roll-up operation shown aggregates the data by ascending the location hierarchy from the level of city to the level of 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. The drill-down operation shown aggregates the data by descending the time hierarchy from the level of Quarter to the level of month.
- **Slice and Dice:** The slice operation performs a selection on one dimension of the given cube, resulting in a subcube. 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 subcube by performing a selection on two or more dimensions. Figure shows a dice operation on the central cube based on the following selection criteria which involves three dimensions: (location="Montreal" or"Vancouver") and (time="Q1" or "Q2") and (item="home entertainment" or "computer").
- **Pivot (Rotate):** Pivot (also called "rotate") is a visualization operation which rotates the data axes in view in order to provide an alternative presentation of the data.

- Drill across: Executes the queries involves more than on fact table.
- Drill through: This operation makes the use of relational SQL by converting the multidimensional data to standard relational operations.