LECTURE NOTES ON

DATA MINING& DATA WAREHOUSING

COURSE CODE:BCS-403

DEPT OF CSE & IT
VSSUT, Burla

SYLLABUS:

Module – I

Data Mining overview, Data Warehouse and OLAP Technology,Data Warehouse Architecture, Stepsfor the Design and Construction of Data Warehouses, A Three-Tier Data WarehouseArchitecture,OLAP,OLAP queries, metadata repository,Data Preprocessing – Data Integration and Transformation, Data Reduction,Data Mining Primitives:What Defines a Data Mining Task? Task-Relevant Data, The Kind of Knowledge to be Mined,KDD

Module – II

Mining Association Rules in Large Databases, Association Rule Mining, Market BasketAnalysis: Mining A Road Map, The Apriori Algorithm: Finding Frequent Itemsets Using Candidate Generation,Generating Association Rules from Frequent Itemsets, Improving the Efficiently of Apriori,Mining Frequent Itemsets without Candidate Generation, Multilevel Association Rules, Approaches toMining Multilevel Association Rules, Mining Multidimensional Association Rules for Relational Database and Data Warehouses,Multidimensional Association Rules, Mining Quantitative Association Rules, MiningDistance-Based Association Rules, From Association Mining to Correlation Analysis

Module – III

What is Classification? What Is Prediction? Issues RegardingClassification and Prediction, Classification by Decision Tree Induction, Bayesian Classification, Bayes Theorem, Naïve Bayesian Classification, Classification by Backpropagation, A Multilayer Feed-Forward Neural Network, Defining aNetwork Topology, Classification Based of Concepts from Association Rule Mining, OtherClassification Methods, k-Nearest Neighbor Classifiers, GeneticAlgorithms, Rough Set Approach, Fuzzy Set Approachs, Prediction, Linear and MultipleRegression, Nonlinear Regression, Other Regression Models, Classifier Accuracy

Module – IV

What Is Cluster Analysis, Types of Data in Cluster Analysis,A Categorization of Major Clustering Methods, Classical Partitioning Methods: k-Meansand k-Medoids, Partitioning Methods in Large Databases: From k-Medoids to CLARANS, Hierarchical Methods, Agglomerative and Divisive Hierarchical Clustering,Density-BasedMethods, Wave Cluster: Clustering Using Wavelet Transformation, CLIQUE:Clustering High-Dimensional Space, Model-Based Clustering Methods, Statistical Approach,Neural Network Approach.

Chapter-1

1.1 What Is Data Mining?

Data mining refers to extracting or mining knowledge from large amountsof data. The term is actually a misnomer. Thus, data miningshould have been more appropriately named as knowledge mining which emphasis on mining from large amounts of data.

It is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems.
The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use.

The key properties of data mining are

- Automatic discovery of patterns
- Prediction of likely outcomes
- Creation of actionable information
- Focus on large datasets and databases

1.2 The Scope of Data Mining

Data mining derives its name from the similarities between searching for valuable business information in a large database — for example, finding linked products in gigabytes of store scanner data — and mining a mountain for a vein of valuable ore. Both processes require either sifting through an immense amount of material, or intelligently probing it to find exactly where the value resides. Given databases of sufficient size and quality, data mining technology can generate new business opportunities by providing these capabilities:

Automated prediction of trends and behaviors. Data mining automates the process of finding predictive information in large databases. Questions that traditionally required extensive hands-on analysis can now be answered directly from the data — quickly. A typical example of a predictive problem is targeted marketing. Data mining uses data on past promotional mailings to identify the targets most likely to maximize return on investment in future mailings. Other predictive problems include forecasting bankruptcy and other forms of default, and identifying segments of a population likely to respond similarly to given events.

Automated discovery of previously unknown patterns. Data mining tools sweep through databases and identify previously hidden patterns in one step. An example of pattern discovery is the analysis of retail sales data to identify seemingly unrelated products that are often purchased together. Other pattern discovery problems include detecting fraudulent credit card transactions and identifying anomalous data that could represent data entry keying errors.

1.3 Tasks of Data Mining

Data mining involves six common classes of tasks:

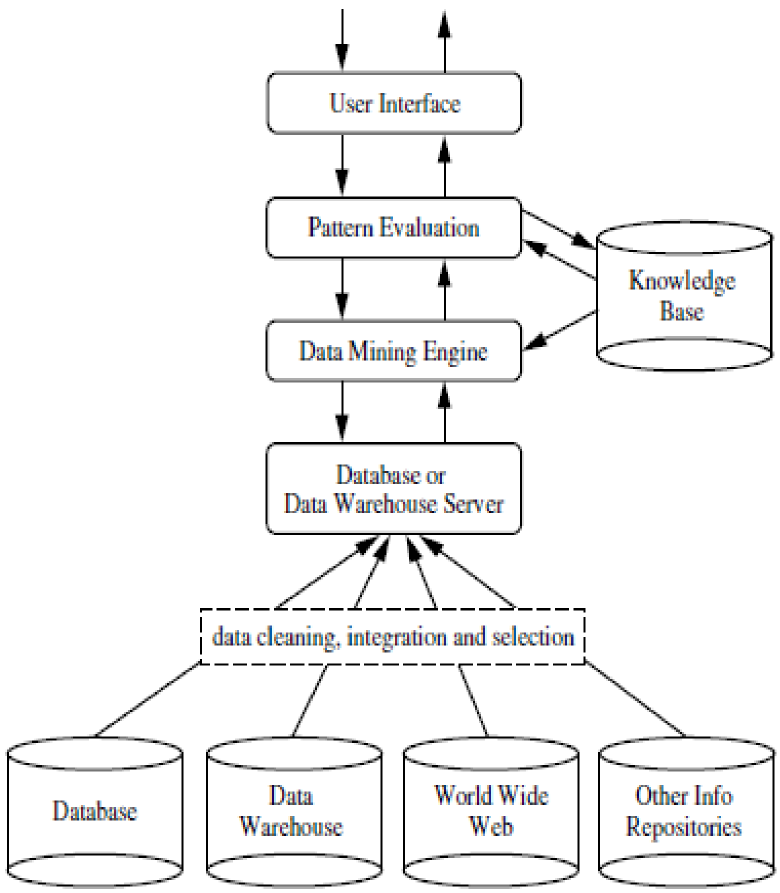
- **Anomaly detection (Outlier/change/deviation detection)** – The identification of unusual data records, that might be interesting or data errors that require further investigation.
- **Association rule learning (Dependency modelling)** – Searches for relationships between variables. For example a supermarket might gather data on customer purchasing habits. Using association rule learning, the supermarket can determine which products are frequently bought together and use this information for marketing purposes. This is sometimes referred to as market basket analysis.
- **Clustering** – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data.

- **Classification** – is the task of generalizing known structure to apply to new data. For example, an e-mail program might attempt to classify an e-mail as "legitimate" or as "spam".
- **Regression** – attempts to find a function which models the data with the least error.

- **Summarization** – providing a more compact representation of the data set, including visualization and report generation.

1.4 Architecture of Data Mining

A typical data mining system may have the following major components.



1. **Knowledge Base:**

This is the domain knowledge that is used to guide the search orevaluate the interestingness of resulting patterns. Such knowledge can include concepthierarchies, used to organize attributes or attribute values into different levels of abstraction. Knowledge such as user beliefs, which can be used to assess a pattern's interestingness based on its unexpectedness, may also be included. Other examples of domain knowledge are additional interestingness constraints or thresholds, and metadata (e.g., describing data from multiple heterogeneous sources).

2. **Data Mining Engine:**

This is essential to the data mining systemand ideally consists ofa set of functional modules for tasks such as characterization, association and correlationanalysis, classification, prediction, cluster analysis, outlier analysis, and evolutionanalysis.

3. **Pattern Evaluation Module:**

This component typically employs interestingness measures interacts with the data mining modules so as to focus thesearch toward interesting patterns. It may use interestingness thresholds to filterout discovered patterns. Alternatively, the pattern evaluation module may be integratedwith the mining module, depending on the implementation of the datamining method used. For efficient data mining, it is highly recommended to pushthe evaluation of pattern interestingness as deep as possible into the mining processso as to confine the search to only the interesting patterns.

4. **User interface:**

Thismodule communicates between users and the data mining system,allowing the user to interact with the system by specifying a data mining query ortask, providing information to help focus the search, and performing exploratory datamining based on the intermediate data mining results. In addition, this componentallows the user to browse database and data warehouse schemas or data structures,evaluate mined patterns, and visualize the patterns in different forms.

1.5 Data Mining Process:

Data Mining is a process of discovering various models, summaries, and derived values from a given collection of data.

The general experimental procedure adapted to data-mining problems involves the following steps:

1. State the problem and formulate the hypothesis

Most data-based modeling studies are performed in a particular application domain. Hence, domain-specific knowledge and experience are usually necessary in order to come up with a meaningful problem statement. Unfortunately, many application studies tend to focus on the data-mining technique at the expense of a clear problem statement. In this step, a modeler usually specifies a set of variables for the unknown dependency and, if possible, a general form of this dependency as an initial hypothesis. There may be several hypotheses formulated for a single problem at this stage. The first step requires the combined expertise of an application domain and a data-mining model. In practice, it usually means a close interaction between the data-mining expert and the application expert. In successful data-mining applications, this cooperation does not stop in the initial phase; it continues during the entire data-mining process.

2. Collect the data

This step is concerned with how the data are generated and collected. In general, there are two distinct possibilities. The first is when the data-generation process is under the control of an expert (modeler): this approach is known as a designed experiment. The second possibility is when the expert cannot influence the data- generation process: this is known as the observational approach. An observational setting, namely, random data generation, is assumed in most data-mining applications. Typically, the sampling distribution is completely unknown after data are collected, or it is partially and implicitly given in the data-collection procedure. It is very important, however, to understand how data collection affects its theoretical distribution, since such a priori knowledge can be very useful for modeling and, later, for the final interpretation of results. Also, it is important to make sure that the data used for estimating a model and the data used later for testing and applying a model come from the same, unknown, sampling distribution. If this is not the case, the estimated model cannot be successfully used in a final application of the results.

3. Preprocessing the data

In the observational setting, data are usually "collected" from the existing databses, data warehouses, and data marts. Data preprocessing usually includes at least two common tasks:

1. **Outlier detection (and removal)** – Outliers are unusual data values that are not consistent with most observations. Commonly, outliers result from measurement errors, coding and recording errors, and, sometimes, are natural, abnormal values. Such nonrepresentative samples can seriously affect the model produced later. There are two strategies for dealing with outliers:

a. Detect and eventually remove outliers as a part of the preprocessing phase, or
b. Develop robust modeling methods that are insensitive to outliers.

2. Scaling, encoding, and selecting features – Data preprocessing includes several steps such as variable scaling and different types of encoding. For example, one feature with the range [0, 1] and the other with the range [−100, 1000] will not have the same weights in the applied technique; they will also influence the final data-mining results differently. Therefore, it is recommended to scale them and bring both features to the same weight for further analysis. Also, application-specific encoding methods usually achieve dimensionality reduction by providing a smaller number of informative features for subsequent data modeling.

These two classes of preprocessing tasks are only illustrative examples of a large spectrum of preprocessing activities in a data-mining process.

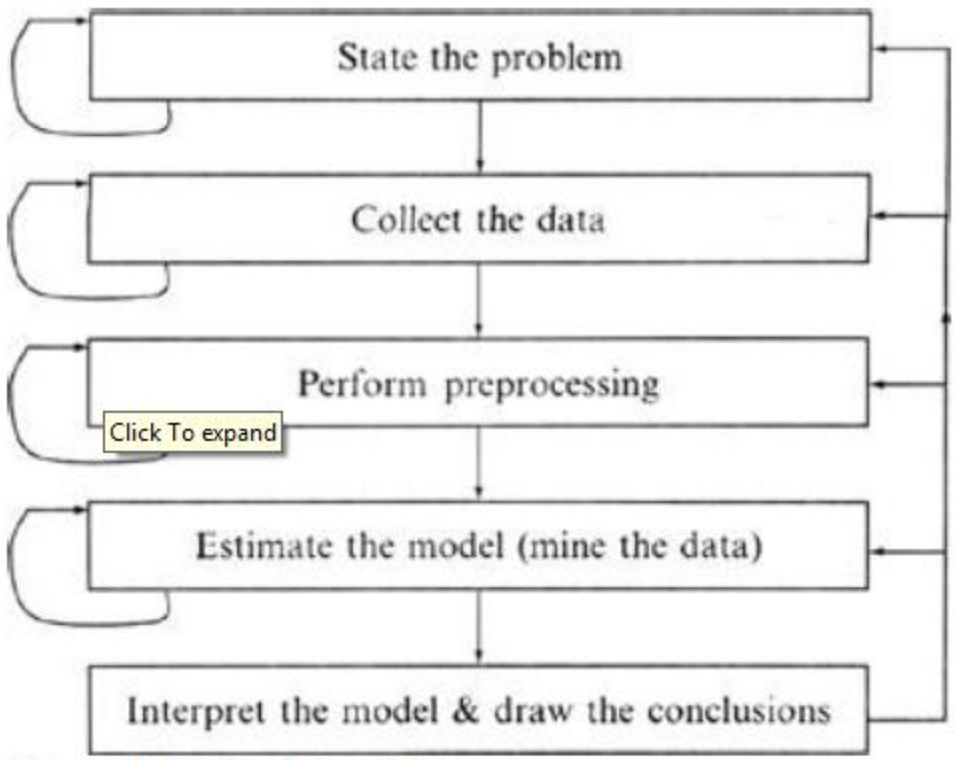
Data-preprocessing steps should not be considered completely independent from other data-mining phases. In every iteration of the data-mining process, all activities, together, could define new and improved data sets for subsequent iterations. Generally, a good preprocessing method provides an optimal representation for a data-mining technique by incorporating a priori knowledge in the form of application-specific scaling and encoding.

4. Estimate the model

The selection and implementation of the appropriate data-mining technique is the main task in this phase. This process is not straightforward; usually, in practice, the implementation is based on several models, and selecting the best one is an additional task. The basic principles of learning and discovery from data are given in Chapter 4 of this book. Later, Chapter 5 through 13 explain and analyze specific techniques that are applied to perform a successful learning process from data and to develop an appropriate model.

5. Interpret the model and draw conclusions

In most cases, data-mining models should help in decision making. Hence, such models need to be interpretable in order to be useful because humans are not likely to base their decisions on complex "black-box" models. Note that the goals of accuracy of the model and accuracy of its interpretation are somewhat contradictory. Usually, simple models are more interpretable, but they are also less accurate. Modern data-mining methods are expected to yield highly accurate results using highdimensional models. The problem of interpreting these models, also very important, is considered a separate task, with specific techniques to validate the results. A user does not want hundreds of pages of numeric results. He does not understand them; he cannot summarize, interpret, and use them for successful decision making.



The Data mining Process

1.6 Classification of Data mining Systems:

The data mining system can be classified according to the following criteria:

- Database Technology
- Statistics
- Machine Learning
- Information Science
- Visualization
- Other Disciplines

Some Other Classification Criteria:

- Classification according to kind of databases mined
- Classification according to kind of knowledge mined
- Classification according to kinds of techniques utilized
- Classification according to applications adapted

Classification according to kind of databases mined

We can classify the data mining system according to kind of databases mined. Database system can be classified according to different criteria such as data models, types of data etc. And the data mining system can be classified accordingly. For example if we classify the database according to data model then we may have a relational, transactional, object- relational, or data warehouse mining system.

Classification according to kind of knowledge mined

We can classify the data mining system according to kind of knowledge mined. It is means data mining system are classified on the basis of functionalities such as:

- Characterization
- Discrimination
- Association and Correlation Analysis
- Classification
- Prediction
- Clustering
- Outlier Analysis
- Evolution Analysis

Classification according to kinds of techniques utilized

We can classify the data mining system according to kind of techniques used. We can describes these techniques according to degree of user interaction involved or the methods of analysis employed.

Classification according to applications adapted

We can classify the data mining system according to application adapted. These applications are as follows:

- Finance
- Telecommunications
- DNA
- Stock Markets
- E-mail

1.7 Major Issues In Data Mining:

•**Mining different kinds of knowledge in databases.** - The need of different users is not the same. And Different user may be in interested in different kind of knowledge. Therefore it is necessary for data mining to cover broad range of knowledge discovery task.

•**Interactive mining of knowledge at multiple levels of abstraction.** - The data mining process needs to be interactive because it allows users to focus the search for patterns, providing and refining data mining requests based on returned results.

•**Incorporation of background knowledge.** - To guide discovery process and to express the discovered patterns, the background knowledge can be used. Background knowledge may be used to express the discovered patterns not only in concise terms but at multiple level of abstraction.

- **Data mining query languages and ad hoc data mining.** - Data Mining Query language that allows the user to describe ad hoc mining tasks, should be integrated with a data warehouse query language and optimized for efficient and flexible data mining.

- **Presentation and visualization of data mining results.** - Once the patterns are discovered it needs to be expressed in high level languages, visual representations. This representations should be easily understandable by the users.

- **Handling noisy or incomplete data.** - The data cleaning methods are required that can handle the noise, incomplete objects while mining the data regularities. If data cleaning methods are not there then the accuracy of the discovered patterns will be poor.

- **Pattern evaluation.** - It refers to interestingness of the problem. The patterns discovered should be interesting because either they represent common knowledge or lack novelty.

- **Efficiency and scalability of data mining algorithms.** - In order to effectively extract the information from huge amount of data in databases, data mining algorithm must be efficient and scalable.

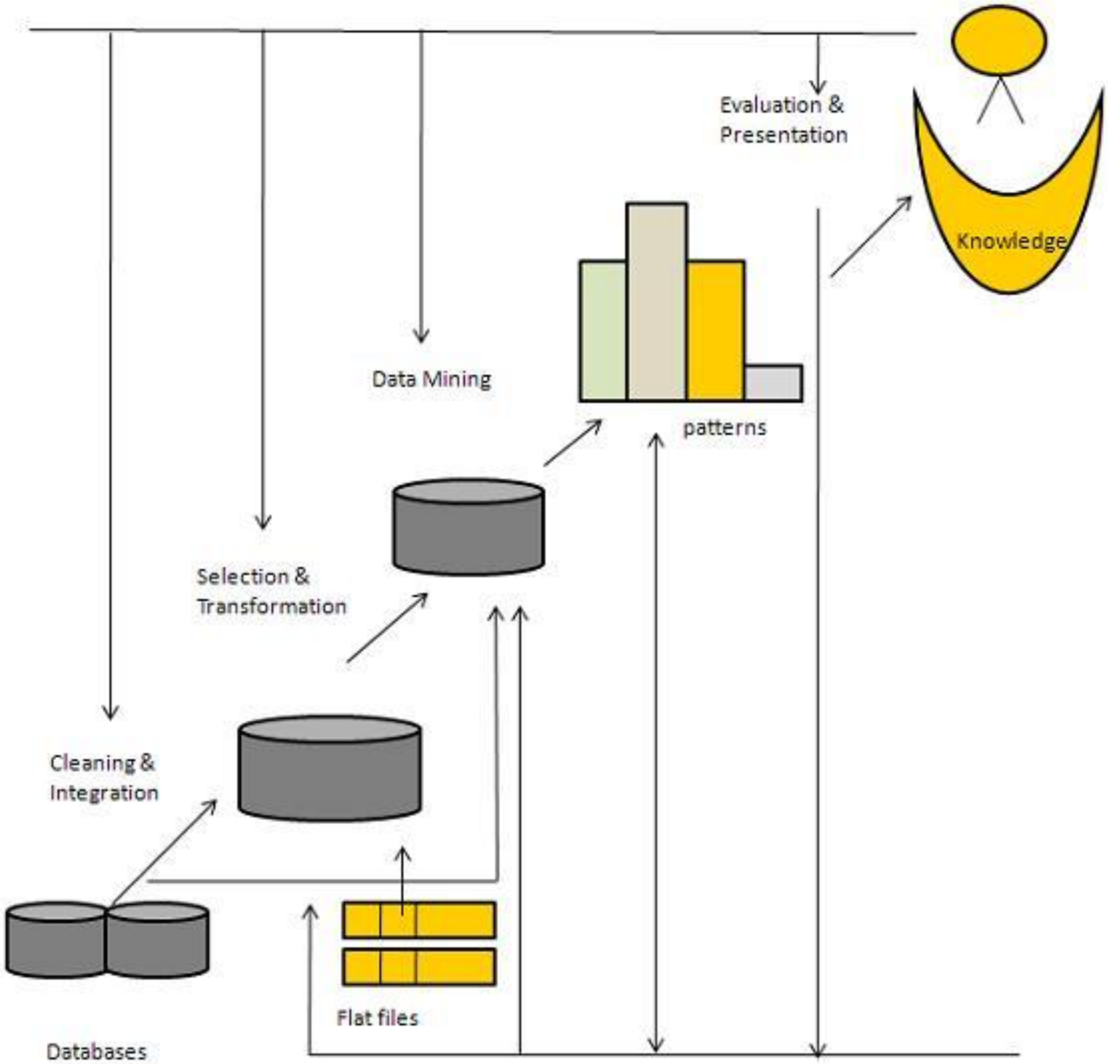
- **Parallel, distributed, and incremental mining algorithms.** - The factors such as huge size of databases, wide distribution of data,and complexity of data mining methods motivate the development of parallel and distributed data mining algorithms. These algorithm divide the data into partitions which is further processed parallel. Then the results from the partitions is merged. The incremental algorithms, updates databases without having mine the data again from scratch.

1.8 Knowledge Discovery in Databases(KDD)

Some people treat data mining same as Knowledge discovery while some people view data mining essential step in process of knowledge discovery. Here is the list of steps involved in knowledge discovery process:

- **Data Cleaning** - In this step the noise and inconsistent data is removed.
- **Data Integration** - In this step multiple data sources are combined.
- **Data Selection -** In this step relevant to the analysis task are retrieved from the database.
- **Data Transformation** - In this step data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.
- **Data Mining** - In this step intelligent methods are applied in order to extract data patterns.
- **Pattern Evaluation -** In this step, data patterns are evaluated.
- **Knowledge Presentation** - In this step,knowledge is represented.

The following diagram shows the process of knowledge discovery process:



Architecture of KDD

1.9 Data Warehouse:

A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process.

Subject-Oriented: A data warehouse can be used to analyze a particular subject area. For example, "sales" can be a particular subject.

Integrated: A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.

Time-Variant: Historical data is kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse. This contrasts with a transactions system, where often only the most recent data is kept. For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.

Non-volatile: Once data is in the data warehouse, it will not change. So, historical data in a data warehouse should never be altered.

1.9.1 Data Warehouse Design Process:

A data warehouse can be built using a *top-down approach*, a *bottom-up approach*, or a *combination of both*.

- The top-down approach starts with the overall design and planning. It is useful in cases where the technology is mature and well known, and where the business problems that must be solved are clear and well understood.

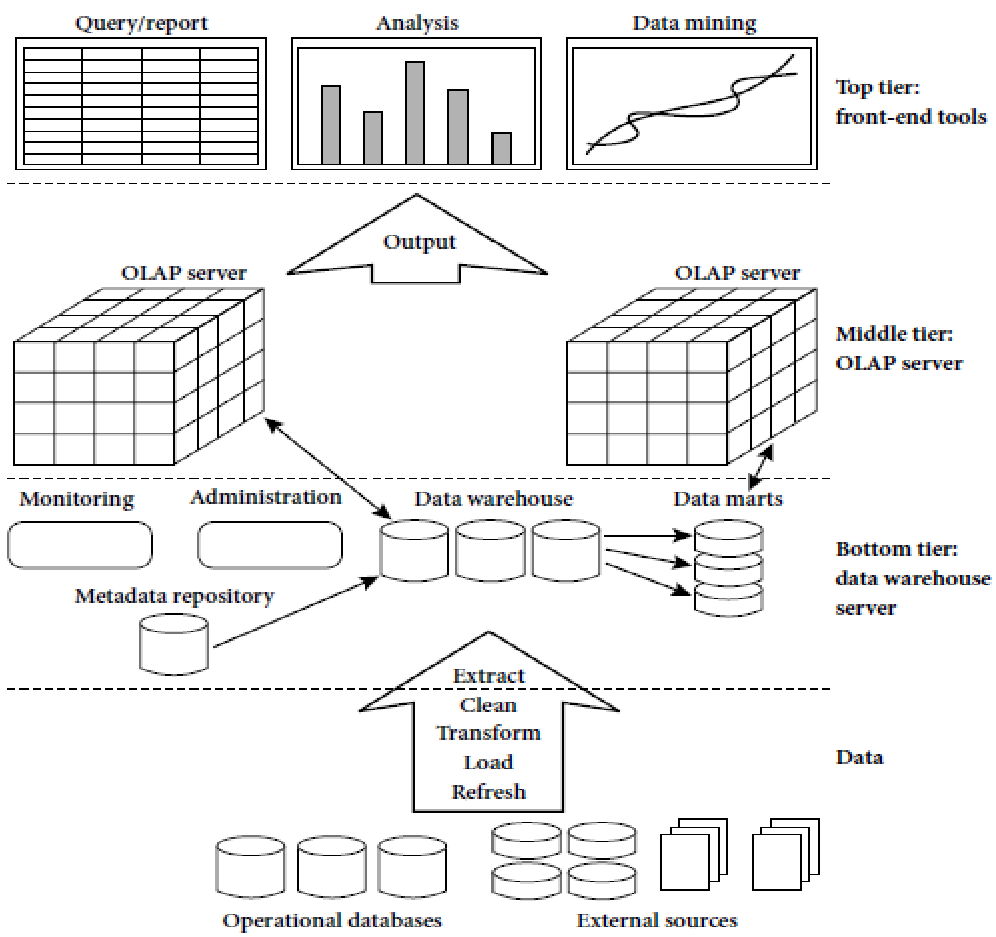
- The bottom-up approach starts with experiments and prototypes. This is useful in the early stage of business modeling and technology development. It allows an organization to move forward at considerably less expense and to evaluate the benefits of the technology before making significant commitments.

- In the combined approach, an organization can exploit the planned and strategic nature of the top-down approach while retaining the rapid implementation and opportunistic application of the bottom-up approach.

The warehouse design process consists of the following steps:

- Choose a business process to model, for example, orders, invoices, shipments, inventory, account administration, sales, or the general ledger. If the business process is organizational and involves multiple complex object collections, a data warehouse model should be followed. However, if the process is departmental and focuses on the analysis of one kind of business process, a data mart model should be chosen.
- Choose the grain of the business process. The grain is the fundamental, atomic level of data to be represented in the fact table for this process, for example, individual transactions, individual daily snapshots, and so on.
- Choose the dimensions that will apply to each fact table record. Typical dimensions are time, item, customer, supplier, warehouse, transaction type, and status.
- Choose the measures that will populate each fact table record. Typical measures are numeric additive quantities like dollars sold and units sold.

1.9.2 A Three Tier Data Warehouse Architecture:



Tier-1:

The bottom tier is a warehouse database server that is almost always a relationaldatabase system. Back-end tools and utilities are used to feed data into the bottomtier from operational databases or other external sources (such as customer profileinformation provided by external consultants). These tools and utilities performdataextraction, cleaning, and transformation (e.g., to merge similar data from differentsources into a unified format), as well as load and refresh functions to update thedata warehouse . The data are extracted using application programinterfaces known as gateways. A gateway is supported by the underlying DBMS andallows client programs to generate SQL code to be executed at a server.

Examplesof gateways include ODBC (Open Database Connection) and OLEDB (Open Linkingand Embedding for Databases) by Microsoft and JDBC (Java Database Connection).

This tier also contains a metadata repository, which stores information aboutthe data warehouse and its contents.

Tier-2:

The middle tier is an OLAP server that is typically implemented using either a relational OLAP (ROLAP) model or a multidimensional OLAP.

- OLAP model is an extended relational DBMS thatmaps operations on multidimensional data to standard relational operations.
- A multidimensional OLAP (MOLAP) model, that is, a special-purpose server that directly implements multidimensional data and operations.

Tier-3:

The top tier is a front-end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).

1.9.3 Data Warehouse Models:

There are three data warehouse models.

1. Enterprise warehouse:

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

2. Data mart:

- A data mart contains a subset of corporate-wide data that is of value to aspecific group of users. The scope is confined to specific selected subjects. For example,a marketing data mart may confine its subjects to customer, item, and sales. Thedata contained in data marts tend to be summarized.

- Data marts are usually implemented on low-cost departmental servers that areUNIX/LINUX- or Windows-based. The implementation cycle of a data mart ismore likely to be measured in weeks rather than months or years. However, itmay involve complex integration in the long run if its design and planning werenot enterprise-wide.

- Depending on the source of data, data marts can be categorized as independent ordependent. Independent data marts are sourced fromdata captured fromone or moreoperational systems or external information providers, or fromdata generated locallywithin a particular department or geographic area. Dependent data marts are sourceddirectly from enterprise data warehouses.

3. Virtual warehouse:

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

1.9.4 Meta Data Repository:

Metadata are data about data.When used in a data warehouse, metadata are the data thatdefine warehouse objects. Metadata are created for the data names anddefinitions of the given warehouse. Additional metadata are created and captured fortimestamping any extracted data, the source of the extracted data, and missing fieldsthat have been added by data cleaning or integration processes.

A metadata repository should contain the following:

- A description of the structure of the data warehouse, which includes the warehouse schema, view, dimensions, hierarchies, and derived data definitions, as well as data mart locations and contents.

- Operational metadata, which include data lineage (history of migrated data and the sequence of transformations applied to it), currency of data (active, archived, or purged), and monitoring information (warehouse usage statistics, error reports, and audit trails).

- The algorithms used for summarization, which include measure and dimension definitionalgorithms, data on granularity, partitions, subject areas, aggregation, summarization,and predefined queries and reports.

- The mapping from the operational environment to the data warehouse, which includessource databases and their contents, gateway descriptions, data partitions, data extraction, cleaning, transformation rules and defaults, data refresh and purging rules, andsecurity (user authorization and access control).

- Data related to system performance, which include indices and profiles that improvedata access and retrieval performance, in addition to rules for the timing and scheduling of refresh, update, and replication cycles.

- Business metadata, which include business terms and definitions, data ownershipinformation, and charging policies.

1.10 OLAP(Online analytical Processing):

- OLAP is an approach to answering multi-dimensional analytical (MDA) queries swiftly.
- OLAP is part of the broader category of business intelligence, which also encompasses relational database, report writing and data mining.
- OLAP tools enable users to analyze multidimensional data interactively from multiple perspectives.

OLAP consists of three basic analytical operations:

- Consolidation (Roll-Up)
- Drill-Down

- Slicing And Dicing

- Consolidation involves the aggregation of data that can be accumulated and computed in one or more dimensions. For example, all sales offices are rolled up to the sales department or sales division to anticipate sales trends.

- The drill-down is a technique that allows users to navigate through the details. For instance, users can view the sales by individual products that make up a region's sales.

- Slicing and dicing is a feature whereby users can take out (slicing) a specific set of data of the OLAP cube and view (dicing) the slices from different viewpoints.

1.10.1 Types of OLAP:

1. Relational OLAP (ROLAP):

- ROLAP works directly with relational databases. The base data and the dimension tables are stored as relational tables and new tables are created to hold the aggregated information. It depends on a specialized schema design.
- This methodology relies on manipulating the data stored in the relational database to give the appearance of traditional OLAP's slicing and dicing functionality. In essence, each action of slicing and dicing is equivalent to adding a "WHERE" clause in the SQL statement.
- ROLAP tools do not use pre-calculated data cubes but instead pose the query to the standard relational database and its tables in order to bring back the data required to answer the question.
- ROLAP tools feature the ability to ask any question because the methodology does not limit to the contents of a cube. ROLAP also has the ability to drill down to the lowest level of detail in the database.

2. Multidimensional OLAP (MOLAP):

- MOLAP is the 'classic' form of OLAP and is sometimes referred to as just OLAP.

- MOLAP stores this data in an optimized multi-dimensional array storage, rather than in a relational database. Therefore it requires the pre-computation and storage of information in the cube - the operation known as processing.

- MOLAP tools generally utilize a pre-calculated data set referred to as a data cube. The data cube contains all the possible answers to a given range of questions.

- MOLAP tools have a very fast response time and the ability to quickly write back data into the data set.

3. Hybrid OLAP (HOLAP):

- There is no clear agreement across the industry as to what constitutes Hybrid OLAP, except that a database will divide data between relational and specialized storage.
- For example, for some vendors, a HOLAP database will use relational tables to hold the larger quantities of detailed data, and use specialized storage for at least some aspects of the smaller quantities of more-aggregate or less-detailed data.
- HOLAP addresses the shortcomings of MOLAP and ROLAP by combining the capabilities of both approaches.
- HOLAP tools can utilize both pre-calculated cubes and relational data sources.

1.11 Data Preprocessing:

1.11.1 Data Integration:

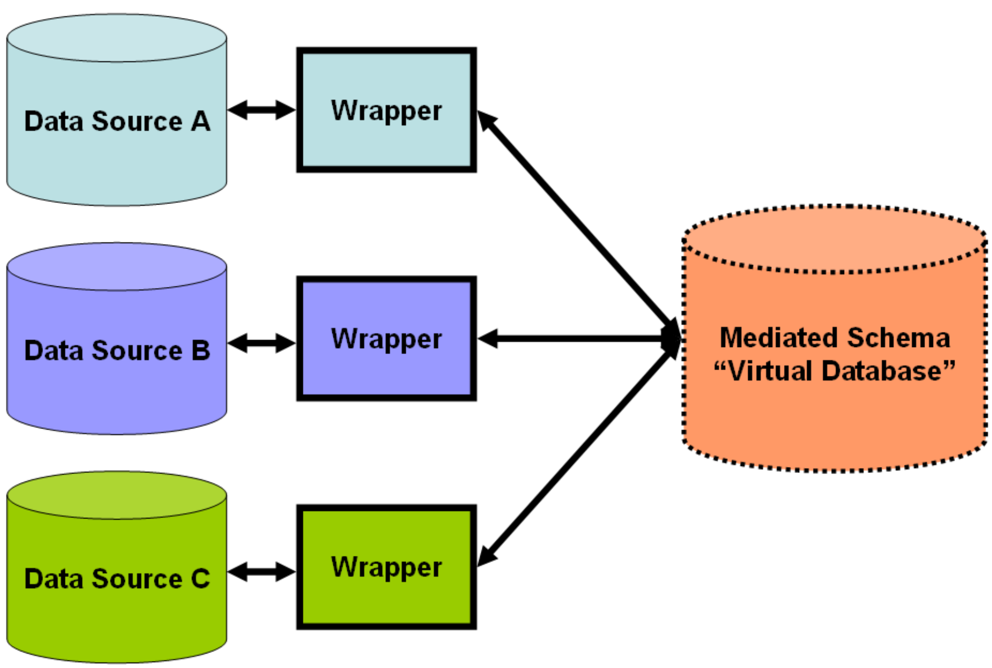
It combines datafrom multiple sources into a coherent data store, as in data warehousing. These sourcesmay include multiple databases, data cubes, or flat files.

The data integration systems are formally defined as triple<G,S,M>

Where G: The global schema

S:Heterogeneous source of schemas

M: Mapping between the queries of source and global schema



1.11.2 Issues in Data integration:

1. **Schema integration and object matching:**

   How can the data analyst or the computer be sure that customer id in one database and customer number in another reference to the same attribute.

2. **Redundancy:**

   An attribute (such as annual revenue, forinstance) may be redundant if it can be derived from another attribute or set ofattributes. Inconsistencies in attribute or dimension naming can also cause redundanciesin the resulting data set.

3. **detection and resolution of datavalue conflicts:**

   For the same real-world entity, attribute values fromdifferent sources may differ.

1.11.3 Data Transformation:

In data transformation, the data are transformed or consolidated into forms appropriatefor mining.

Data transformation can involve the following:

- **Smoothing**, which works to remove noise from the data. Such techniques includebinning, regression, and clustering.
- **Aggregation**, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annualtotal amounts. This step is typically used in constructing a data cube for analysis of the data at multiple granularities.

- **Generalization of the data**, where low-level or “primitive” (raw) data are replaced byhigher-level concepts through the use of concept hierarchies. For example, categoricalattributes, like street, can be generalized to higher-level concepts, like city or country.
- **Normalization**, where the attribute data are scaled so as to fall within a small specifiedrange, such as 1:0 to 1:0, or 0:0 to 1:0.
- **Attribute construction** (or feature construction),wherenewattributes are constructedand added from the given set of attributes to help the mining process.

1.11.4 Data Reduction:

Data reduction techniques can be applied to obtain a reduced representation of thedata set that ismuch smaller in volume, yet closely maintains the integrity of the originaldata. That is, mining on the reduced data set should be more efficient yet produce thesame (or almost the same) analytical results.

Strategies for data reduction include the following:

- **Data cube aggregation**, where aggregation operations are applied to the data in theconstruction of a data cube.
- **Attribute subset selection**, where irrelevant, weakly relevant, or redundant attributesor dimensions may be detected and removed.
- **Dimensionality reduction**, where encoding mechanisms are used to reduce the dataset size.
- **Numerosityreduction**,where the data are replaced or estimated by alternative, smallerdata representations such as parametric models (which need store only the modelparameters instead of the actual data) or nonparametric methods such as clustering,sampling, and the use of histograms.
- **Discretization and concept hierarchy generation**,where rawdata values for attributesare replaced by ranges or higher conceptual levels. Data discretization is a form ofnumerosity reduction that is very useful for the automatic generation of concept hierarchies.Discretization and concept hierarchy generation are powerful tools for datamining, in that they allow the mining of data at multiple levels of abstraction.

Chapter-2

2.1 Association Rule Mining:

- 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.
- Based on the concept of strong rules, RakeshAgrawal et al. introduced association rules.

Problem Definition:

The problem of association rule mining is defined as:

Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n binary attributes called *items*.

Let $D = \{t_1, t_2, \ldots, t_m\}$ 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.

A *rule* is defined as an implication of the form $X \Rightarrow Y$

where $X, Y \subseteq I$ and $X \cap Y = \emptyset$.

The sets of items (for short *itemsets*) X and Y are called *antecedent* (left-hand-side or LHS) and *consequent* (right-hand-side or RHS) of the rule respectively.

Example:

To illustrate the concepts, we use a small example from the supermarket domain. The set of items is $I = \{\mathrm{milk}, \mathrm{bread}, \mathrm{butter}, \mathrm{beer}\}$ and a small database containing the items (1 codes presence and 0 absence of an item in a transaction) is shown in the table.

An example rule for the supermarket could be $\{\mathrm{butter}, \mathrm{bread}\} \Rightarrow \{\mathrm{milk}\}$ meaning that if butter and bread are bought, customers also buy milk.

Example database with 4 items and 5 transactions

Transaction ID	milk	bread	butter	beer
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

2.1.1 Important concepts of Association Rule Mining:

- The **support** $\mathrm{supp}(X)$ of an itemset X is defined as the proportion of transactions in the data set which contain the itemset. In the example database, the itemset $\{\mathrm{milk}, \mathrm{bread}, \mathrm{butter}\}$ has a support of $1/5 = 0.2$ since it occurs in 20% of all transactions (1 out of 5 transactions).

- The **confidence** of a rule is defined

  $$\mathrm{conf}(X \Rightarrow Y) = \mathrm{supp}(X \cup Y)/\mathrm{supp}(X).$$

  For example, the rule $\{\mathrm{butter}, \mathrm{bread}\} \Rightarrow \{\mathrm{milk}\}$ has a confidence of $0.2/0.2 = 1.0$ in the database, which means that for 100% of the transactions containing butter and bread the rule is correct (100% of the times a customer buys butter and bread, milk is bought as well). Confidence can be interpreted as an estimate of the probability $P(Y|X)$, the probability of finding the RHS of the rule in transactions under the condition that these transactions also contain the LHS.

- The ***lift*** of a rule is defined as

$$\mathrm{lift}(X \Rightarrow Y) = \frac{\mathrm{supp}(X \cup Y)}{\mathrm{supp}(X) \times \mathrm{supp}(Y)}$$

or the ratio of the observed support to that expected if X and Y were independent. The rule $\{\mathrm{milk}, \mathrm{bread}\} \Rightarrow \{\mathrm{butter}\}$ has a lift of $\frac{0.2}{0.4 \times 0.4} = 1.25$.

- The **conviction** of a rule is defined as

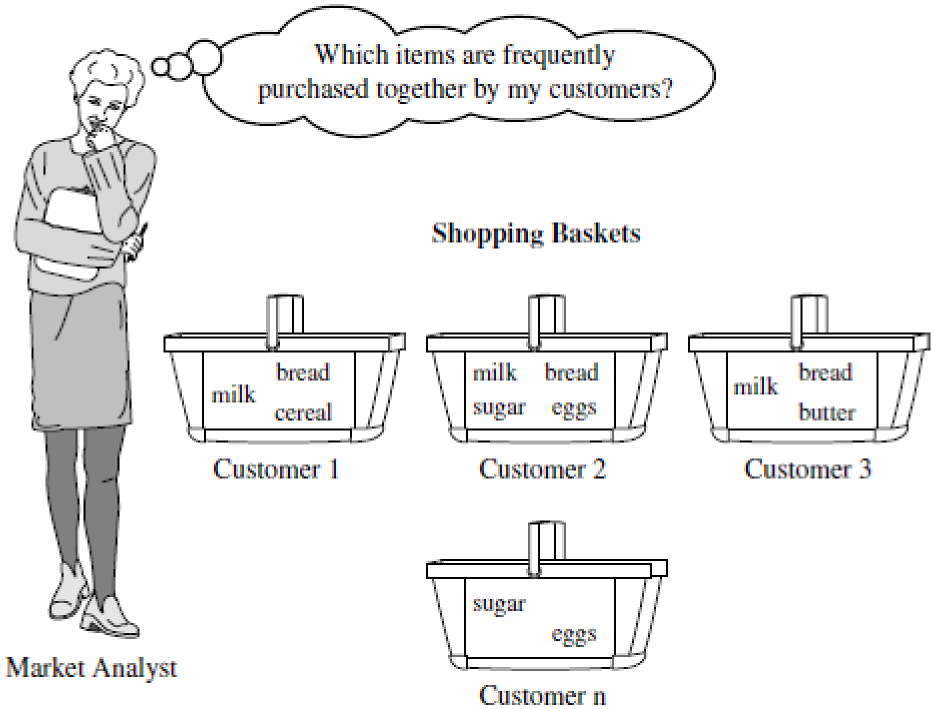
$$\mathrm{conv}(X \Rightarrow Y) = \frac{1 - \mathrm{supp}(Y)}{1 - \mathrm{conf}(X \Rightarrow Y)}.$$

The rule $\{\mathrm{milk}, \mathrm{bread}\} \Rightarrow \{\mathrm{butter}\}$ has a conviction of $\frac{1 - 0.4}{1 - .5} = 1.2$,

and can be interpreted as the ratio of the expected frequency that X occurs without Y (that is to say, the frequency that the rule makes an incorrect prediction) if X and Y were independent divided by the observed frequency of incorrect predictions.

2.2 Market basket analysis:

This processanalyzes customer buying habits by finding associations between the different items thatcustomers place in their shopping baskets. The discovery of such associationscan help retailers develop marketing strategies by gaining insight into which itemsare frequently purchased together by customers. For instance, if customers are buyingmilk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket. Such information can lead to increased sales by helping retailers doselective marketing and plan their shelf space.



Example:

If customers who purchase computers also tend to buy antivirussoftware at the same time, then placing the hardware display close to the software displaymay help increase the sales of both items. In an alternative strategy, placing hardware andsoftware at opposite ends of the store may entice customers who purchase such items topick up other items along the way. For instance, after deciding on an expensive computer,a customer may observe security systems for sale while heading toward the software displayto purchase antivirus software and may decide to purchase a home security systemas well. Market basket analysis can also help retailers plan which items to put on saleat reduced prices. If customers tend to purchase computers and printers together, thenhaving a sale on printers may encourage the sale of printers *as well as* computers.

2.3 Frequent Pattern Mining:

Frequent patternmining can be classified in various ways, based on the following criteria:

1. **Based on the completeness of patterns to be mined:**

- We can mine the complete set of frequent itemsets, the closed frequent itemsets, and the maximal frequent itemsets, given a minimum support threshold.
- We can also mine constrained frequent itemsets, approximate frequent itemsets,near-match frequent itemsets, top-k frequent itemsets and so on.

2. **Based on the levels of abstraction involved in the rule set:**

Some methods for associationrule mining can find rules at differing levels of abstraction.

For example, supposethat a set of association rules mined includes the following rules where X is a variablerepresenting a customer:

buys(X, "computer"))=>buys(X, "HP printer") (1)

buys(*X*, "*laptop computer*")) =>*buys*(*X*, "*HP printer*") (2)

In rule (1) and (2), the items bought are referenced at different levels ofabstraction (e.g., "*computer*" is a higher-level abstraction of "*laptop computer*").

3. **Based on the number of data dimensions involved in the rule:**

- If the items or attributes in an association rule reference only one dimension, then it is a single-dimensional association rule.
  buys(X, "computer"))=>buys(X, "antivirus software")

- If a rule references two or more dimensions, such as the dimensions age, income, and buys, then it is amultidimensional association rule. The following rule is an exampleof a multidimensional rule:
  age(X, "30,31…39") ^ income(X, "42K,…48K"))=>buys(X, "high resolution TV")

4. **Based on the types of values handled in the rule:**

- If a rule involves associations between the presence or absence of items, it is a Boolean association rule.
- If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule.

5. **Based on the kinds of rules to be mined:**

- Frequent pattern analysis can generate various kinds of rules and other interesting relationships.
- Association rule mining cangenerate a large number of rules, many of which are redundant or do not indicatea correlation relationship among itemsets.
- The discovered associations can be further analyzed to uncover statistical correlations, leading to correlation rules.

6. **Based on the kinds of patterns to be mined:**

- Many kinds of frequent patterns can be mined from different kinds of data sets.
- Sequential pattern mining searches for frequent subsequences in a sequence data set, where a sequence records an ordering of events.
- For example, with sequential pattern mining, we can study the order in which items are frequently purchased. For instance, customers may tend to first buy a PC, followed by a digitalcamera,and then a memory card.
- Structuredpatternminingsearches for frequent substructuresin a structured data set.
- Single items are the simplest form of structure.
- Each element of an itemsetmay contain a subsequence, a subtree, and so on.
- Therefore, structuredpattern mining can be considered as the most general formof frequent pattern mining.

2.4 Efficient Frequent Itemset Mining Methods:

2.4.1 Finding Frequent Itemsets Using Candidate Generation:The Apriori Algorithm

- Apriori is a seminal algorithm proposed by R. Agrawal and R. Srikant in 1994 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.
- 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 by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted $L1$.Next, $L1$ is used to find $L2$, the set of 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 $L_k$requires one full scan of the database.
- A two-step process is followed in Aprioriconsisting of joinand prune action.

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D, a database of transactions;
- min_sup, the minimum support count threshold.

Output: L, frequent itemsets in D.

Method:

```
L1 = find_frequent_1-itemsets(D);
for (k = 2; L(k-1) ≠ φ; k++) {
    Ck = apriori_gen(L(k-1));
    for each transaction t ∈ D { // scan D for counts
        Ct = subset(Ck, t); // get the subsets of t that are candidates
        for each candidate c ∈ Ct
            c.count++;
    }
    Lk = {c ∈ Ck | c.count ≥ min_sup}
}
return L = ∪k Lk;

procedure apriori_gen(L(k-1):frequent (k − 1)-itemsets)
    for each itemset l1 ∈ L(k-1)
        for each itemset l2 ∈ L(k-1)
            if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧ ... ∧ (l1[k − 2] = l2[k − 2]) ∧ (l1[k − 1] < l2[k − 1]) then {
                c = l1 ⋈ l2; // join step: generate candidates
                if has_infrequent_subset(c, L(k-1)) then
                    delete c; // prune step: remove unfruitful candidate
                else add c to Ck;
            }
    return Ck;

procedure has_infrequent_subset(c: candidate k-itemset;
            L(k-1): frequent (k − 1)-itemsets); // use prior knowledge
    for each (k − 1)-subset s of c
        if s ∉ L(k-1) then
            return TRUE;
    return FALSE;
```

Example:

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

There are nine transactions in this database, that is, $|D| = 9$.

Steps:

1. In the first iteration of the algorithm, each item is a member of the set of candidate1-itemsets, C1. The algorithm simply scans all of the transactions in order to countthe number of occurrences of each item.

2. Suppose that the minimum support count required is 2, that is, min sup = 2. The set of frequent 1-itemsets, L1, can thenbe determined. It consists of the candidate 1-itemsets satisfying minimum support.In our example, all of the candidates in C1 satisfy minimum support.

3. To discover the set of frequent 2-itemsets, L2, the algorithm uses the join L1 on L1 togenerate a candidate set of 2-itemsets, C2.No candidates are removed fromC2 during the prune step because each subset of thecandidates is also frequent.

4.Next, the transactions inDare scanned and the support count of each candidate itemsetInC2 is accumulated.

5. The set of frequent 2-itemsets, L2, is then determined, consisting of those candidate2-itemsets in C2 having minimum support.

6. The generation of the set of candidate 3-itemsets,C3, Fromthejoin step, we first getC3 =L2x L2 = ({I1, I2, I3}, {I1, I2, I5}, {I1, I3, I5}, {I2, I3, I4},{I2, I3, I5}, {I2, I4, I5}. Based on the Apriori property that all subsets of a frequentitemsetmust also be frequent, we can determine that the four latter candidates cannotpossibly be frequent.

7.The transactions in D are scanned in order to determine L3, consisting of those candidate 3-itemsets in C3 having minimum support.

8.The algorithm uses L3x L3 to generate a candidate set of 4-itemsets, C4.

Scan D for count of each candidate → C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare candidate support count with minimum support count → L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generate C_2 candidates from L_1 → C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

Scan D for count of each candidate → C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

Compare candidate support count with minimum support count → L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

Generate C_3 candidates from L_2 → C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

Scan D for count of each candidate → C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare candidate support count with minimum support count → L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

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

(a) Join: $C_3 = L_2 \bowtie L_2 = \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\} \bowtie \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\}$
$= \{\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\}\}$.

(b) Prune using the Apriori property: All nonempty subsets of a frequent itemset must also be frequent. Do any of the candidates have a subset that is not frequent?

- The 2-item subsets of {I1, I2, I3} are {I1, I2}, {I1, I3}, and {I2, I3}. All 2-item subsets of {I1, I2, I3} are members of L_2. Therefore, keep {I1, I2, I3} in C_3.
- The 2-item subsets of {I1, I2, I5} are {I1, I2}, {I1, I5}, and {I2, I5}. All 2-item subsets of {I1, I2, I5} are members of L_2. Therefore, keep {I1, I2, I5} in C_3.
- The 2-item subsets of {I1, I3, I5} are {I1, I3}, {I1, I5}, and {I3, I5}. {I3, I5} is not a member of L_2, and so it is not frequent. Therefore, remove {I1, I3, I5} from C_3.
- The 2-item subsets of {I2, I3, I4} are {I2, I3}, {I2, I4}, and {I3, I4}. {I3, I4} is not a member of L_2, and so it is not frequent. Therefore, remove {I2, I3, I4} from C_3.
- The 2-item subsets of {I2, I3, I5} are {I2, I3}, {I2, I5}, and {I3, I5}. {I3, I5} is not a member of L_2, and so it is not frequent. Therefore, remove {I2, I3, I5} from C_3.
- The 2-item subsets of {I2, I4, I5} are {I2, I4}, {I2, I5}, and {I4, I5}. {I4, I5} is not a member of L_2, and so it is not frequent. Therefore, remove {I2, I4, I5} from C_3.

(c) Therefore, $C_3 = \{\{I1, I2, I3\}, \{I1, I2, I5\}\}$ after pruning.

Generation and pruning of candidate 3-itemsets, C_3, from L_2 using the Apriori property.

2.4.2 Generating Association Rules from Frequent Itemsets:

Once the frequent itemsets from transactions in a database *D* have been found, it is straightforward to generate strong association rules from them.

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

The conditional probability is expressed in terms of itemset support count, where $support\_count(A \cup B)$ is the number of transactions containing the itemsets $A \cup B$, and $support\_count(A)$ is the number of transactions containing the itemset A. Based on this equation, association rules can be generated as follows:

- For each frequent itemset l, generate all nonempty subsets of l.
- For every nonempty subset s of l, output the rule "$s \Rightarrow (l - s)$" if $\frac{support\_count(l)}{support\_count(s)} \geq min\_conf$, where min_conf is the minimum confidence threshold.

Example:

Generating association rules. Let's try an example based on the transactional data for *AllElectronics* shown in Table 5.1. Suppose the data contain the frequent itemset l = {I1, I2, I5}. What are the association rules that can be generated from l? The nonempty subsets of l are {I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}. The resulting association rules are as shown below, each listed with its confidence:

$$\begin{array}{ll}
I1 \wedge I2 \Rightarrow I5, & confidence = 2/4 = 50\% \\
I1 \wedge I5 \Rightarrow I2, & confidence = 2/2 = 100\% \\
I2 \wedge I5 \Rightarrow I1, & confidence = 2/2 = 100\% \\
I1 \Rightarrow I2 \wedge I5, & confidence = 2/6 = 33\% \\
I2 \Rightarrow I1 \wedge I5, & confidence = 2/7 = 29\% \\
I5 \Rightarrow I1 \wedge I2, & confidence = 2/2 = 100\%
\end{array}$$

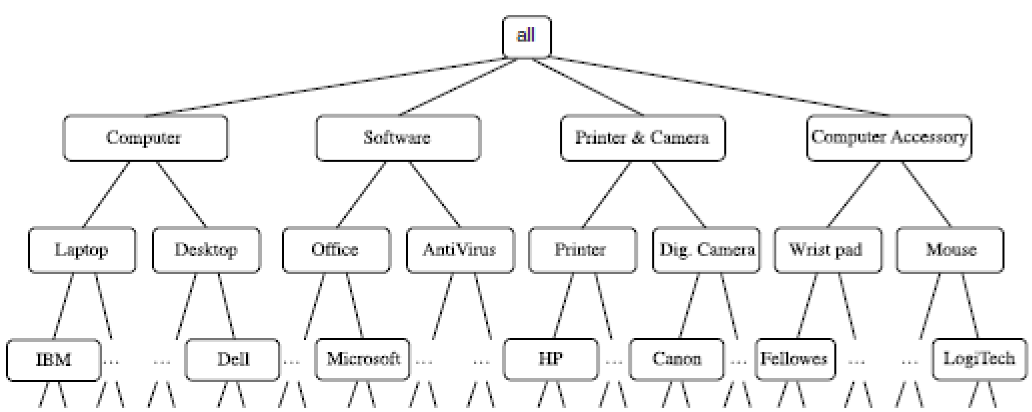
2.5 Mining Multilevel Association Rules:

- For many applications, it is difficult to find strong associations among data items at low or primitive levels of abstraction due to the sparsity of data at those levels.
- Strong associations discovered at high levels of abstraction may represent commonsense knowledge.
- Therefore, data mining systems should provide capabilities for mining association rules at multiple levels of abstraction, with sufficient flexibility for easy traversal amongdifferentabstraction spaces.

- Association rules generated from mining data at multiple levels of abstraction arecalled multiple-level or multilevel association rules.
- Multilevel association rules can be mined efficiently using concept hierarchies under a support-confidence framework.
- In general, a top-down strategy is employed, where counts are accumulated for the calculation of frequent itemsets at each concept level, starting at the concept level 1 and working downward in the hierarchy toward the more specific concept levels,until no more frequent itemsets can be found.

A concepthierarchy defines a sequence of mappings froma set of low-level concepts to higherlevel,more general concepts. Data can be generalized by replacing low-level conceptswithin the data by their higher-level concepts, or ancestors, from a concept hierarchy.

TID	*Items Purchased*
T100	IBM-ThinkPad-T40/2373, HP-Photosmart-7660
T200	Microsoft-Office-Professional-2003, Microsoft-Plus!-Digital-Media
T300	Logitech-MX700-Cordless-Mouse, Fellowes-Wrist-Rest
T400	Dell-Dimension-XPS, Canon-PowerShot-S400
T500	IBM-ThinkPad-R40/P4M, Symantec-Norton-Antivirus-2003
...	...



A concept hierarchy for *AllElectronics* computer items.

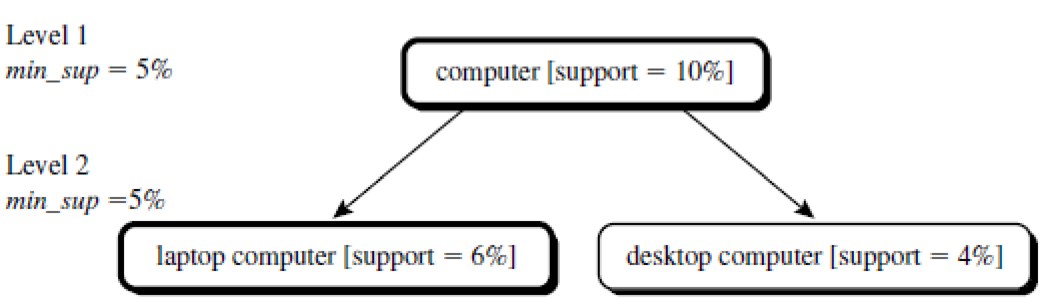
The concept hierarchy has five levels, respectively referred to as levels 0to 4, starting with level 0 at the root node for all.

- Here, Level 1 includes computer, software, printer&camera, and computer accessory.
- Level 2 includes laptop computer, desktop computer, office software, antivirus software
- Level 3 includes IBM desktop computer, . . . , Microsoft office software, and so on.
- Level 4 is the most specific abstraction level of this hierarchy.

2.5.1 Approaches ForMining Multilevel Association Rules:

1.UniformMinimum Support:

- The same minimum support threshold is used when mining at each level of abstraction.
- When a uniform minimum support threshold is used, the search procedure is simplified.
- The method is also simple in that users are required to specify only one minimum support threshold.
- The uniform support approach, however, has some difficulties. It is unlikely thatitems at lower levels of abstraction will occur as frequently as those at higher levelsof abstraction.
- If the minimum support threshold is set too high, it could miss somemeaningful associations occurring at low abstraction levels. If the threshold is set too low, it may generate many uninteresting associations occurring at high abstractionlevels.

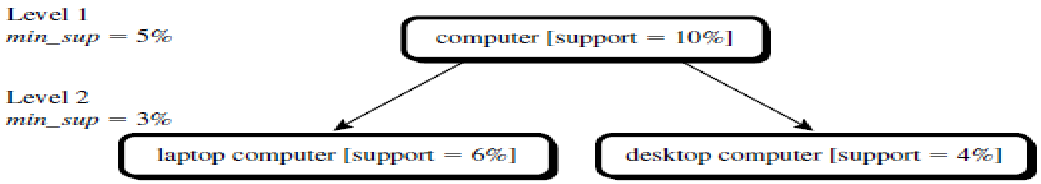


Multilevel mining with uniform support.

2.Reduced Minimum Support:

- Each level of abstraction has its own minimum support threshold.

- The deeper the level of abstraction, the smaller the corresponding threshold is.
- For example,the minimum support thresholds for levels 1 and 2 are 5% and 3%,respectively. In this way, "computer," "laptop computer," and "desktop computer" areall considered frequent.



3.Group-Based Minimum Support:

- Because users or experts often have insight as to which groups are more important than others, it is sometimes more desirable to set up user-specific, item, or group based minimal support thresholds when mining multilevel rules.
- For example, a user could set up the minimum support thresholds based on product price, or on items of interest, such as by setting particularly low support thresholds for laptop computersand flash drives in order to pay particular attention to the association patterns containing items in these categories.

2.6 Mining Multidimensional Association Rules from Relational Databases and Data Warehouses:

- Single dimensional or intradimensional association rule contains a single distinct predicate (e.g., buys)with multiple occurrences i.e., the predicate occurs more than once within the rule.

  buys(*X*, "*digital camera*")=>*buys*(*X*, "*HP printer*")

- Association rules that involve two or more dimensions or predicates can be referred to as multidimensional association rules.

age(X, "20…29")^occupation(X, "student")=>buys(X, "laptop")

- Above Rule contains three predicates (age, occupation,and buys), each of which occurs only once in the rule. Hence, we say that it has norepeated predicates.
- Multidimensional association rules with no repeated predicates arecalled interdimensional association rules.
- We can also mine multidimensional associationrules with repeated predicates, which contain multiple occurrences of some predicates.These rules are called hybrid-dimensional association rules. An example of sucha rule is the following, where the predicate buys is repeated:

  age(X, "20…29")^buys(X, "laptop")=>buys(X, "HP printer")

2.7 Mining Quantitative Association Rules:

- Quantitative association rules are multidimensional association rules in which the numeric attributes are *dynamically* discretized during the mining process so as to satisfy some mining criteria, such as maximizing the confidence or compactness of the rules mined.
- In this section, we focus specifically on how to mine quantitative association rules having two quantitative attributes on the left-hand side of the rule and one categorical attribute on the right-hand side of the rule. That is

*Aquan*1 ^*Aquan*2 =>*Acat*

where*Aquan*1 and *Aquan*2 are tests on quantitative attribute interval

*Acat*tests a categorical attribute fromthe task-relevantdata.

- Such rules have been referred to as two-dimensional quantitative association rules,

because they contain two quantitative dimensions.

- For instance, suppose you are curious about the association relationship between pairs of quantitative attributes, like customer age and income, and the type of television (such as *high-definition TV,* i.e., *HDTV*) that customers like to buy.

An example of such a 2-D quantitative association rule is

age(*X*, "30…39")^*income*(*X*, "42*K*…48*K*")=>*buys*(*X*, "*HDTV*")

2.8 From Association Mining to Correlation Analysis:

- A correlation measure can be used to augment the support-confidence framework for association rules. This leads to *correlation rules* of the form
  A=>*B* [*support*, *confidence*, *correlation*]
- That is, a correlation rule is measured not only by its support and confidence but alsoby the correlation between itemsets*A* and *B*. There are many different correlation measuresfrom which to choose. In this section, we study various correlation measures todetermine which would be good for mining large data sets.

- Lift is a simple correlation measure that is given as follows. The occurrence of itemset *A* is independent of the occurrence of itemset*B* if $P(A \cup B)$ = $P(A)P(B)$; otherwise, itemsets*A* and *B* are dependent and correlated as events. This definition can easily be extended to more than two itemsets.

The lift between the occurrence of *A* and *B* can bemeasured by computing

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

- If the lift(A,B) is less than 1, then the occurrence of A is negativelycorrelated with the occurrence of B.
- If the resulting value is greater than 1, then A and B are positively correlated, meaning that the occurrence of one implies the occurrence of the other.
- If the resulting value is equal to 1, then A and B are independent and there is no correlation between them.

Chapter-3

3.1 Classification and Prediction:

- Classification and prediction are two forms of data analysis that can be used to extractmodels describing important data classes or to predict future data trends.
- Classificationpredicts categorical (discrete, unordered) labels, *prediction* models continuousvaluedfunctions.
- For example, we can build a classification model to categorize bankloan applications as either safe or risky, or a prediction model to predict the expendituresof potential customers on computer equipment given their income and occupation.
- A predictor is constructed that predicts a continuous-valued function, or ordered value, as opposed to a categorical label.
- Regression analysis is a statistical methodology that is most often used for numeric prediction.
- Many classification and prediction methods have been proposed by researchers in machine learning, pattern recognition, and statistics.
- Most algorithms are memory resident, typically assuming a small data size. Recent data mining research has built on such work, developing scalable classification and prediction techniques capable of handling large disk-resident data.

3.1.1 Issues Regarding Classification and Prediction:

1.Preparing the Data for Classification and Prediction:

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

(i)Data cleaning:

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

(ii)Relevance analysis:

- Many of the attributes in the data may be *redundant*.
- Correlation analysis can be used to identify whether any two given attributes are statisticallyrelated.
- For example, a strong correlation between attributes *A*1 and *A*2 would suggest that one of the two could be removed from further analysis.
- A database may also contain *irrelevant* attributes. Attribute subset selection can be used in these cases to find a reduced set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.
- Hence, relevance analysis, in the form of correlation analysis and attribute subset selection, can be used to detect attributes that do not contribute to the classification or prediction task.
- Such analysis can help improve classification efficiency and scalability.

(iii)Data Transformation And Reduction

- The data may be transformed by normalization, particularly when neural networks or methods involving distance measurements are used in the learning step.
- Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1 to +1 or 0 to 1.
- The data can also be transformed by *generalizing* it to higher-level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous valuedattributes.

- For example, numeric values for the attribute *income* can be generalized to discrete ranges, such as *low*, *medium*, and *high*. Similarly, categorical attributes, like *street*, can be generalized to higher-level concepts, like *city*.
- Data can also be reduced by applying many other methods, ranging from wavelet transformation and principle components analysis to discretization techniques, such as binning, histogram analysis, and clustering.

3.1.2 Comparing Classification and Prediction Methods:

➢ **Accuracy:**

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

➢ **Speed:**

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

➢ **Robustness:**

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

➢ **Scalability:**

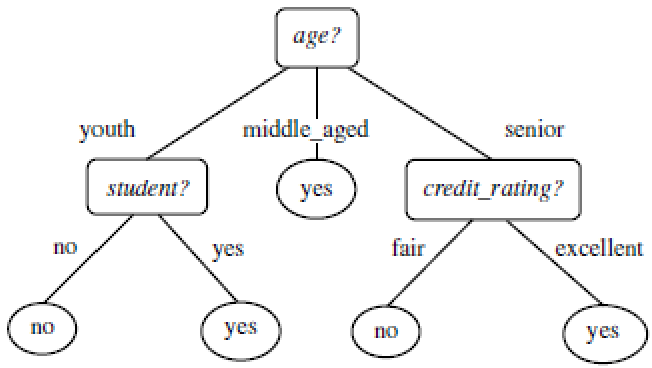
This refers to the ability to construct the classifier or predictor efficiently given large amounts of data.

➢ **Interpretability:**

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

3.2 Classification by Decision Tree Induction:

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



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

3.2.1 Algorithm For Decision Tree Induction:

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

Input:

- Data partition, D, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split point* or *splitting subset*.

Output: A decision tree.

Method:

(1) create a node N;
(2) **if** tuples in D are all of the same class, C **then**
(3)   return N as a leaf node labeled with the class C;
(4) **if** *attribute_list* is empty **then**
(5)   return N as a leaf node labeled with the majority class in D; // majority voting
(6) apply **Attribute_selection_method**(D, *attribute_list*) to **find** the "best" *splitting_criterion*;
(7) label node N with *splitting_criterion*;
(8) **if** *splitting_attribute* is discrete-valued **and**
    multiway splits allowed **then** // not restricted to binary trees
(9)   *attribute_list* $\leftarrow$ *attribute_list* $-$ *splitting_attribute*; // remove *splitting_attribute*
(10) **for each** outcome j of *splitting_criterion*
    // partition the tuples and grow subtrees for each partition
(11)   let D_j be the set of data tuples in D satisfying outcome j; // a partition
(12)   **if** D_j is empty **then**
(13)   attach a leaf labeled with the majority class in D to node N;
(14)   **else** attach the node returned by **Generate_decision_tree**(D_j, *attribute_list*) to node N;
    endfor
(15) return N;

The algorithm is called with three parameters:

- Data partition
- Attribute list
- Attribute selection method

- The parameter attribute list is a list ofattributes describing the tuples.
- Attribute selection method specifies a heuristic procedurefor selecting the attribute that "best" discriminates the given tuples according to class.
- The tree starts as a single node, *N*, representing the training tuples in *D*.

- If the tuples in *D* are all of the same class, then node *N* becomes a leaf and is labeledwith that class .
- Allof the terminating conditions are explained at the end of the algorithm.
- Otherwise, the algorithm calls Attribute selection method to determine the splitting criterion.
- The splitting criterion tells us which attribute to test at node N by determiningthe “best” way to separate or partition the tuples in D into individual classes.

There are three possible scenarios.Let *A* be the splitting attribute. *A* has *v* distinct values, $\{a_1, a_2, \ldots, a_v\}$, based on the training data.

1 A is discrete-valued:

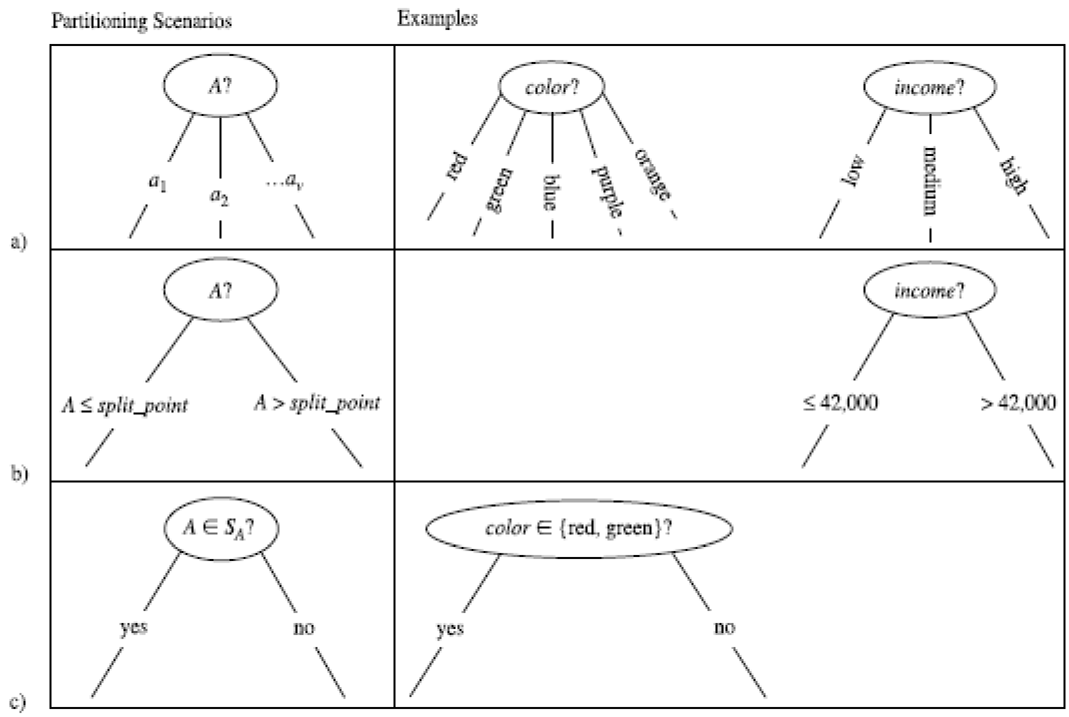
- In this case, the outcomes of the test at node N corresponddirectly to the known values of A.
- A branch is created for each known value, aj, of A and labeled with that value.
- Aneed not be considered in any future partitioning of the tuples.

2 A is continuous-valued:

In this case, the test at node N has two possible outcomes, corresponding to the conditions A <=split point and A >split point, respectively
wheresplit point is the split-point returned by Attribute selection method as part of the splitting criterion.

3 A is discrete-valued and a binary tree must be produced:

The test at node N is of the form“A€SA?”.
SA is the splitting subset for A, returned by Attribute selection methodas part of the splitting criterion. It is a subset of the known values of A.



(a)If A is Discrete valued (b)If A is continuous valued (c) IfA is discrete-valued and a binary tree must be produced:

3.3 Bayesian Classification:

- Bayesian classifiers are statistical classifiers.
- They can predictclass membership probabilities, such as the probability that a given tuple belongs toa particular class.
- Bayesian classification is based on Bayes' theorem.

3.3.1 Bayes' Theorem:

- Let X be a data tuple. In Bayesian terms, X is considered "evidence."and it is described by measurements made on a set of n attributes.

- Let H be some hypothesis, such as that the data tuple X belongs to a specified class C.
- For classification problems, we want to determine P(H|X), the probability that the hypothesis H holds given the "evidence" or observed data tuple X.
- P(H|X) is the posterior probability, or a posteriori probability, of H conditioned on X.
- Bayes' theorem is useful in that it providesa way of calculating the posterior probability, $P(H|\boldsymbol{X})$, from $P(H)$, $P(\boldsymbol{X}|H)$, and $P(\boldsymbol{X})$.

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

3.3.2 Naïve Bayesian Classification:

The naïve 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 = (x1, x2, …,xn), depicting n measurements made on the tuple from n attributes, respectively, A1, A2, …, An.

2. Suppose that there are m classes, C1, C2, …, 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 naïve Bayesian classifier predicts that tuple X belongs to the class Ci if and only if

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

Thus we maximize P(Cij**X**). The classCifor which P(Cij**X**) is maximized is called the maximum posteriori hypothesis. By Bayes' theorem

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

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

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

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

We can easily estimate the probabilities $P(x1|Ci)$, $P(x2|Ci)$, : : : , $P(xn|Ci)$ fromthe trainingtuples. For eachattribute, we look at whether the attribute is categorical or continuous-valued. Forinstance, to compute $P(\boldsymbol{X}|Ci)$, we consider the following:

- If $A_k$is categorical, then $P(x_k|Ci)$ is the number of tuples of class Ciin D havingthe value $x_k$for A_k, divided by $|C_{i,D}|$ the number of tuples of class $C_i$in D.
- If $A_k$is continuous-valued, then we need to do a bit more work, but the calculationis pretty straightforward.

A continuous-valued attribute is typically assumed tohave a Gaussian distribution with a mean μ and standard deviation , defined by

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}).$$

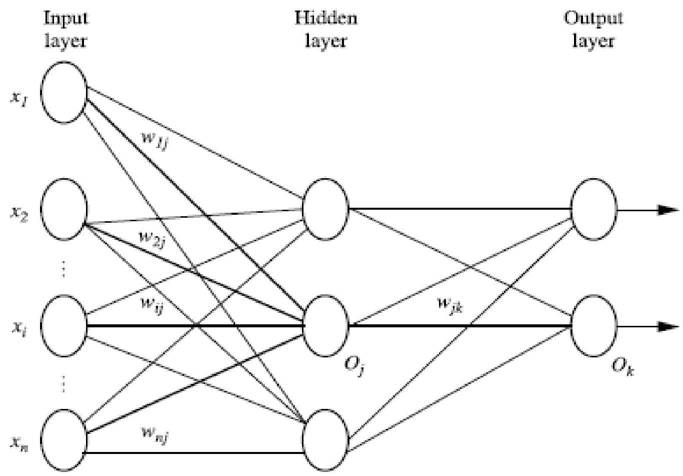
5. In order to predict the class label of X, $P(X\text{j}Ci)P(Ci)$ is evaluated for each class Ci.
The classifier predicts that the class label of tuple X is the class Ciif and only if

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

3.4 A Multilayer Feed-Forward Neural Network:

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

Example:



- The inputs to the network correspond to the attributes measured for each training tuple. The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer known as a hidden layer.
- The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary.
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction for given tuples

3.4.1 Classification by Backpropagation:

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

- Neural networks involve long training times and are therefore more suitable for applicationswhere this is feasible.
- Backpropagation learns by iteratively processing a data set of training tuples, comparing the network's prediction for each tuple with the actual known target value.
- The target value may be the known class label of the training tuple (for classification problems) or a continuous value (for prediction).
- For each training tuple, the weights are modified so as to minimize the mean squared errorbetween the network's prediction and the actual target value. These modifications are made in the "backwards" direction, that is, from the output layer, through each hidden layer down to the first hidden layer hence the name is backpropagation.
- Although it is not guaranteed, in general the weights will eventually converge, and the learning process stops.

Advantages:

- It include their high tolerance of noisy data as well as their ability to classify patterns on which they have not been trained.
- They can be used when you may have little knowledge of the relationships between attributesand classes.
- They are well-suited for continuous-valued inputs and outputs, unlike most decision tree algorithms.
- They have been successful on a wide array of real-world data, including handwritten character recognition, pathology and laboratory medicine, and training a computer to pronounce English text.
- Neural network algorithms are inherently parallel; parallelization techniques can be used to speed up the computation process.

Process:

Initialize the weights:

The weights in the network are initialized to small random numbers
ranging from-1.0 to 1.0, or -0.5 to 0.5. Each unit has a *bias* associated with it. The biases are similarly initialized to small random numbers.

Each training tuple, ***X***, is processed by the following steps.

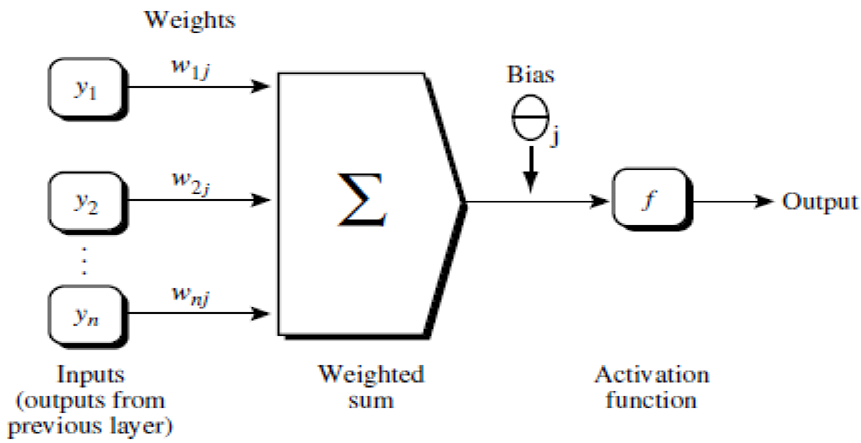
Propagate the inputs forward:

First, the training tuple is fed to the input layer of thenetwork. The inputs pass through the input units, unchanged. That is, for an input unit*j*, its output, *Oj*, is equal to its input value, *Ij*. Next, the net input and output of eachunit in the hidden and output layers are computed. The net input to a unit in the hiddenor output layers is computed as a linear combination of its inputs.
Each such unit has anumber of inputs to it that are, in fact, the outputs of the units connected to it in theprevious layer. Each connection has a weight. To compute the net input to the unit, each input connected to the unit ismultiplied by its correspondingweight, and this is summed.

$$I_j = \sum_i w_{ij} O_i + \theta_j,$$

where$w_{i,j}$is the weight of the connection from unit iin the previous layer to unit j;
$O_i$is the output of unit ifrom the previous layer
$\theta_j$is the bias of the unit & it actsas a threshold in that it serves to vary the activity of the unit.

Each unit in the hidden and output layers takes its net input and then applies an activation function to it.



Backpropagate the error:

The error is propagated backward by updating the weights and biases to reflect the error of the network's prediction. For a unit *j* in the output layer, the error Err_j is computed by

$$Err_j = O_j(1 - O_j)(T_j - O_j)$$

where O_j is the actual output of unit j, and T_j is the known target value of the giventraining tuple.

The error of a hidden layerunit *j* is

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{jk}$$

where w_{jk} is the weight of the connection from unit j to a unit k in the next higher layer, and Err_k is the error of unit k.

Weights are updatedby the following equations, where D*wi j* is the change in weight w_{ij}:

$$\Delta w_{ij} = (l)Err_j O_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

Biases are updated by the following equations below

$$\Delta\theta_j = (l)Err_j$$

$$\theta_j = \theta_j + \Delta\theta_j$$

Algorithm:

Input:

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

Output: A trained neural network.

Method:

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

3.5 k-Nearest-Neighbor Classifier:

- Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuplewith training tuples that are similar to it.
- The training tuples are describedby *n* attributes. Each tuple represents a point in an *n*-dimensional space. In this way,all of the training tuples are stored in an *n*-dimensional pattern space. When given anunknown tuple, a ***k***-nearest-neighbor classifier searches the pattern space for the *k* trainingtuples that are closest to the unknown tuple. These *k* training tuples are the *k* nearest neighbors of the unknown tuple.

- Closeness is defined in terms of a distance metric, such as Euclidean distance.
- The Euclidean distance between two points or tuples, say, $\boldsymbol{X}_1 = (x_{11}, x_{12}, \dots, x_{1n})$ and

$\boldsymbol{X}_2 = (x_{21}, x_{22}, \dots, x_{2n})$, is

$$dist(\boldsymbol{X_1}, \boldsymbol{X_2}) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}.$$

In other words, for each numeric attribute, we take the difference between the corresponding values of that attribute in tuple $\boldsymbol{X_1}$and in tuple $\boldsymbol{X_2}$, square this difference,and accumulate it. The square root is taken of the total accumulated distance count.

Min-Max normalization can be used to transforma value v of a numeric attribute A to v_0 in therange [0, 1] by computing

$$v' = \frac{v - min_A}{max_A - min_A},$$

where$min_A$and $max_A$are the minimum and maximum values of attribute A

- For k-nearest-neighbor classification, the unknown tuple is assigned the mostcommon class among its k nearest neighbors.
- When $k = 1$, the unknown tuple is assigned the class of the training tuple that is closest to it in pattern space.
- Nearestneighborclassifiers can also be used for prediction, that is, to return a real-valued prediction for a given unknown tuple.
- In this case, the classifier returns the averagevalue of the real-valued labels associated with the k nearest neighbors of the unknowntuple.

3.6 Other Classification Methods:

3.6.1 Genetic Algorithms:

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, geneticlearning starts as follows.

- An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes,C_1 andC_2.

- The rule “IF A_1 ANDNOT A_2 THENC_2” can be encoded as the bit string “100,” where the two leftmost bits represent attributes A_1 and A_2, respectively, and the rightmost bit represents the class.
- Similarly, the rule “IF NOT A_1 AND NOT A_2 THEN C_1” can be encoded as “001.”
- If an attribute has k values, where k > 2, then k bits may be used to encode the attribute’s values.

Classes can be encoded in a similar fashion.

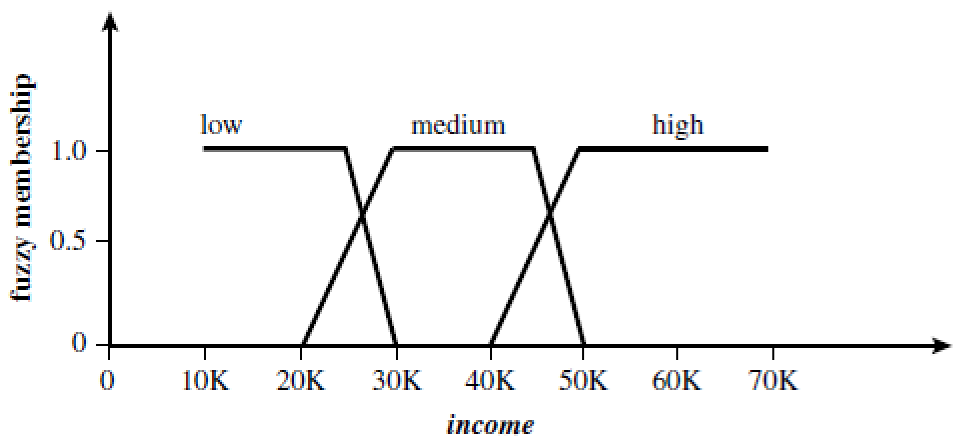
- Based on the notion of survival of the fittest, a new population is formed to consist of the fittest rules in the current population, as well as offspring of these rules.
- Typically, thefitness of a rule is assessed by its classification accuracy on a set of training samples.
- Offspring are created by applying genetic operators such as crossover and mutation.
- In crossover, substrings from pairs of rules are swapped to form new pairs of rules.
- Inmutation, randomly selected bits in a rule’s string are inverted.
- The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a pre specified fitness threshold.
- Genetic algorithms are easily parallelizable and have been used for classification as well as other optimization problems. In data mining, they may be used to evaluate the fitness of other algorithms.

3.6.2 Fuzzy Set Approaches:

- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree ofmembership that a certain value has in a given category. Each category then represents afuzzy set.
- Fuzzy logic systemstypically provide graphical tools to assist users in converting attribute values to fuzzy truthvalues.
- Fuzzy set theory is also known as possibility theory.

- It was proposed by LotfiZadeh in1965 as an alternative to traditional two-value logic and probability theory.
- It lets usworkat a high level of abstraction and offers a means for dealing with imprecise measurementof data.
- Most important, fuzzy set theory allows us to dealwith vague or inexact facts.
- Unlike the notion of traditional "crisp" sets where anelement either belongs to a set *S* or its complement, in fuzzy set theory, elements canbelong to more than one fuzzy set.
- Fuzzy set theory is useful for data mining systems performing rule-based classification.
- It provides operations for combining fuzzy measurements.
- Several procedures exist for translating the resulting fuzzy output into a *defuzzified*or crisp value that is returned by the system.
- Fuzzy logic systems have been used in numerous areas for classification, including market research, finance, health care, and environmental engineering.

Example:



3.7 Regression Analysis:

- Regression analysis can be used to model the relationship between one or moreindependent or predictor variables and a dependent or response variable which is continuous-valued.
- In the context of data mining, the predictor variables are theattributes of interest describing the tuple (i.e., making up the attribute vector).
- In general,the values of the predictor variables are known.

- The response variable is what we want to predict.

3.7.1 Linear Regression:

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

That is, $y = b+wx$

where the variance of *y* is assumed to be constant

b and *w* are regression coefficientsspecifying the Y-intercept and slope of the line.

- The regression coefficients, *w* and *b*, can also be thought of as weights, so that we can equivalently write, $y = w_0+w_1x$
- These coefficients can be solved for by the method of least squares, which estimates the best-fitting straight line as the one that minimizes the error between the actual data and the estimate of the line.
- Let *D* be a training set consisting of values of predictor variable, *x*, for some population and their associated values for response variable, *y*. The training set contains $|D|$ data points of the form $(x_1, y_1), (x_2, y_2), \ldots, (x_{|D|}, y_{|D|})$.

The regressioncoefficients can be estimated using this method with the following equations:

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

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

where *x* is the mean value of $x_1, x_2, \ldots, x_{|D|}$, and *y* is the mean value of $y_1, y_2, \ldots, y_{|D|}$.

The coefficients w_0 and w_1 often provide good approximations to otherwise complicated regression equations.

3.7.2 Multiple Linear Regression:

- It is an extension of straight-line regression so as to involve more than one predictor variable.

- It allows response variable y to be modeled as a linear function of, say, n predictor variables or attributes, A_1, A_2, …, An, describing a tuple, X.
- An example of a multiple linear regression model basedon two predictor attributes or variables, A_1 and A_2, is$y = w_0+w_1x_1+w_2x_2$

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

- Multiple regression problemsare instead commonly solved with the use of statistical software packages, such as SAS,SPSS, and S-Plus.

3.7.3 Nonlinear Regression:

- It can be modeled by adding polynomial terms to the basic linear model.
- By applying transformations to the variables, we can convert the nonlinear model into a linear one that can then be solved by the method of least squares.
- Polynomial Regression is a special case of multiple regression. That is, the addition of high-order terms like x^2, x^3, and so on, which are simple functions of the single variable, x, can be considered equivalent to adding new independent variables.

Transformation of a polynomial regression model to a linear regression model:

Consider a cubic polynomial relationship given by

$$y = w_0+w_1x+w_2x_2+w_3x_3$$

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

$x_1 = x$, $x_2 = x^2$,$x_3 = x^3$

It can then be converted to linear formby applying the above assignments,resulting in the equation$y = w_0+w_1x+w_2x_2+w_3x_3$

which is easily solved by themethod of least squares using software for regression analysis.

3.8 Classifier Accuracy:

- The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.
- In the pattern recognition literature, this is also referred to as the overall recognition rate of theclassifier, that is, it reflects how well the classifier recognizes tuples of the various classes.

- The error rate or misclassification rate of a classifier,M, which is simply 1-Acc(M), where Acc(M) is the accuracy of M.
- The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes.
- True positives refer to the positive tuples that were correctly labeled by the classifier.
- True negatives are the negative tuples that were correctly labeled by the classifier.
- False positives are the negative tuples that were incorrectly labeled.
- How well the classifier can recognize, for this sensitivity and specificity measures can be used.

Accuracy is a function of sensitivity and specificity.

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

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

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

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

wheret _posis the number of true positives

posis the number of positive tuples

t _negis the number of true negatives

negis the number of negative tuples,

f _posis the number of false positives

Chapter-4

4.1 Cluster Analysis:

- The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering.
- A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters.
- A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression.
- Cluster analysis tools based on k-means, k-medoids, and several methods have also been built into many statisticalanalysis software packages or systems, such as S-Plus, SPSS, and SAS.

4.1.1 Applications:

- Cluster analysis has been widely used in numerous applications, including market research, pattern recognition, data analysis, and image processing.
- In business, clustering can help marketers discover distinct groups in their customer bases and characterize customer groups based on purchasing patterns.
- In biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionality, and gain insight into structures inherent in populations.
- Clustering may also help in the identification of areas of similar land use in an earth observation database and in the identification of groups of houses in a city according to house type, value,and geographic location, as well as the identification of groups of automobile insurance policy holders with a high average claim cost.
- Clustering is also called data segmentation in some applications because clustering partitions large data sets into groups according to their *similarity*.

- Clustering can also be used for outlier detection,Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce.

4.1.2 Typical Requirements Of Clustering InData Mining:

- **Scalability:**

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

  Highly scalable clustering algorithms are needed.

- **Ability to deal with different types of attributes:**

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

- **Discovery of clusters with arbitrary shape:**

  Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density.

  However, a cluster could be of any shape. It is important to develop algorithms thatcan detect clusters of arbitrary shape.

- **Minimal requirements for domain knowledge to determine input parameters:**

  Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters). The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects. This not only burdens users, but it also makes the quality of clustering difficult to control.

- **Ability to deal with noisy data:**

  Most real-world databases contain outliers or missing, unknown, or erroneous data.

  Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

- **Incremental clustering and insensitivity to the order of input records:**

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

  That is, given a set of data objects, such an algorithm may return dramatically different clusterings depending on the order of presentation of the input objects.

  It is important to develop incremental clustering algorithms and algorithms thatare insensitive to the order of input.

- **High dimensionality:**

  A database or a data warehouse can contain several dimensionsor attributes.Many clustering algorithms are good at handling low-dimensional data,involving only two to three dimensions. Human eyes are good at judging the qualityof clustering for up to three dimensions. Finding clusters of data objects in highdimensionalspace is challenging, especially considering that such data can be sparseand highly skewed.

- **Constraint-based clustering:**

  Real-world applications may need to perform clustering under various kinds of constraints. Suppose that your job is to choose the locations for a given number of new automatic banking machines (ATMs) in a city. To decide upon this, you may cluster households while considering constraints such as the city's rivers and highway networks, and the type and number of customers per cluster. A challenging task is to find groups of data with good clustering behavior that satisfy specified constraints.

- **Interpretability and usability:**

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

4.2 Major Clustering Methods:

- Partitioning Methods
- Hierarchical Methods

- Density-Based Methods
- Grid-Based Methods
- Model-Based Methods

4.2.1 Partitioning Methods:

A partitioning method constructs *k* partitions of the data, where each partition represents a cluster and *k* <= *n*. That is, it classifies the data into *k* groups, which together satisfy the following requirements:

- Each group must contain at least one object, and
- Each object must belong to exactly one group.

A partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.

The general criterion of a good partitioning is that objects in the same cluster are close or related to each other, whereas objects of different clusters are far apart or very different.

4.2.2 Hierarchical Methods:

A hierarchical method creates a hierarchical decomposition ofthe given set of data objects. A hierarchical method can be classified as being eitheragglomerative or divisive, based on howthe hierarchical decomposition is formed.

- Theagglomerative approach, also called the bottom-up approach, starts with each objectforming a separate group. It successively merges the objects or groups that are closeto one another, until all of the groups are merged into one or until a termination condition holds.
- The divisive approach, also calledthe top-down approach, starts with all of the objects in the same cluster. In each successiveiteration, a cluster is split up into smaller clusters, until eventually each objectis in one cluster, or until a termination condition holds.

Hierarchical methods suffer fromthe fact that once a step (merge or split) is done,it can never be undone. This rigidity is useful in that it leads to smaller computationcosts by not having toworry about a combinatorial number of different choices.

There are two approachesto improving the quality of hierarchical clustering:

- ❖ Perform careful analysis ofobject “linkages” at each hierarchical partitioning, such as in Chameleon, or
- ❖ Integratehierarchical agglomeration and other approaches by first using a hierarchicalagglomerative algorithm to group objects into microclusters, and then performingmacroclustering on the microclusters using another clustering method such as iterative relocation.

4.2.3 Density-based methods:

- ❖ Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes.
- ❖ Other clustering methods have been developed based on the notion of density. Their general idea is to continue growing the given cluster as long as the density in the neighborhood exceeds some threshold; that is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points. Such a method can be used to filter out noise (outliers)and discover clusters of arbitrary shape.
- ❖ DBSCAN and its extension, OPTICS, are typical density-based methods that growclusters according to a density-based connectivity analysis. DENCLUE is a methodthat clusters objects based on the analysis of the value distributions of density functions.

4.2.4 Grid-Based Methods:

- ❖ Grid-based methods quantize the object space into a finite number of cells that form a grid structure.

- All of the clustering operations are performed on the grid structure i.e., on the quantized space. The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space.
- STING is a typical example of a grid-based method. Wave Cluster applies wavelet transformation for clustering analysis and is both grid-based and density-based.

4.2.5 Model-Based Methods:

- Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model.
- A model-based algorithm may locate clusters by constructing a density function that reflects the spatial distribution of the data points.
- It also leads to a way of automatically determining the number of clusters based on standard statistics, taking "noise" or outliers into account and thus yielding robust clustering methods.

4.3 Tasks in Data Mining:

- Clustering High-Dimensional Data
- Constraint-Based Clustering

4.3.1 Clustering High-Dimensional Data:

- It is a particularly important task in cluster analysis because many applications require the analysis of objects containing a large number of features or dimensions.
- For example, text documents may contain thousands of terms or keywords as features, and DNA micro array data may provide information on the expression levels of thousands of genes under hundreds of conditions.
- Clustering high-dimensional data is challenging due to the curse of dimensionality.
- Many dimensions may not be relevant. As the number of dimensions increases, thedata become increasingly sparse so that the distance measurement between pairs ofpoints become meaningless and the average density of points anywhere in the data islikely to be low. Therefore, a different clustering methodology needs to be developedfor high-dimensional data.

- CLIQUE and PROCLUS are two influential subspace clustering methods, which search for clusters in subspaces ofthe data, rather than over the entire data space.
- Frequent pattern–based clustering,another clustering methodology, extractsdistinct frequent patterns among subsets ofdimensions that occur frequently. It uses such patterns to group objects and generatemeaningful clusters.

4.3.2 Constraint-Based Clustering:

- It is a clustering approach that performs clustering by incorporation of user-specified or application-oriented constraints.
- A constraint expresses a user's expectation or describes properties of the desired clustering results, and provides an effective means for communicating with the clustering process.
- Various kinds of constraints can be specified, either by a user or as per application requirements.
- Spatial clustering employs with the existence of obstacles and clustering under user-specified constraints. In addition, semi-supervised clusteringemploys forpairwise constraints in order to improvethe quality of the resulting clustering.

4.4 Classical Partitioning Methods:

The mostwell-known and commonly used partitioningmethods are

- The *k*-Means Method
- k-Medoids Method

4.4.1 Centroid-Based Technique: The *K*-Means Method:

The k-means algorithm takes the input parameter, k, and partitions a set of n objects intok clusters so that the resulting intracluster similarity is high but the intercluster similarity is low.

Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity.

The *k*-means algorithm proceeds as follows.

- First, it randomly selects *k* of the objects, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean.
- It then computes the new mean for each cluster.
- This process iterates until the criterion function converges.

Typically, the square-error criterion is used, defined as

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

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

pis the point in space representing a given object

$m_i$is the mean of cluster C_i

4.4.1 The k-means partitioning algorithm:

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

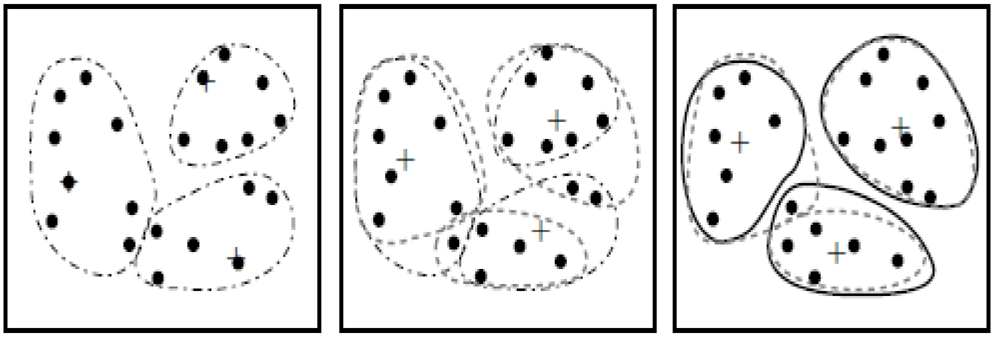
Input:

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

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) **repeat**
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) **until** no change;



Clustering of a set of objects based on the *k*-means method

4.4.2 The *k*-Medoids Method:

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

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

where$E$ is the sum of the absolute error for all objects in the data set

$\boldsymbol{p}$is the point inspace representing a given object in clusterC_j

$\mathbf{o_j}$is the representative object of C_j

- The initial representative objects are chosen arbitrarily. The iterative process of replacing representative objects by non representative objects continues as long as the quality of the resulting clustering is improved.
- This quality is estimated using a cost function that measures the average dissimilaritybetween an object and the representative object of its cluster.
- To determine whether a non representative object, oj random, is a good replacement for a current representativeobject, oj, the following four cases are examined for each of the nonrepresentative objects.

Case 1:

pcurrently belongs to representative object, o_j. If $o_j$is replaced by o_{random}asa representative object and p is closest to one of the other representative objects, o_i,$i\neq j$, then p is reassigned to o_i.

Case 2:

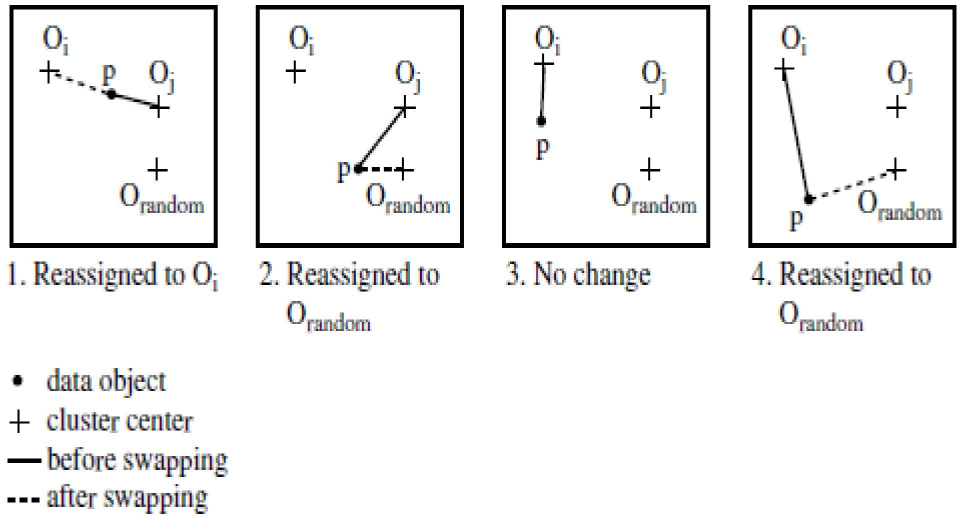
pcurrently belongs to representative object, oj. If ojis replaced by o_{random}asa representative object and p is closest to o_{random}, then p is reassigned to o_{random}.

Case 3:

pcurrently belongs to representative object, o_i, $i\neq j$. If ojis replaced by o_{random}as a representative object and p is still closest to o_i, then the assignment does notchange.

Case 4:

pcurrently belongs to representative object, oi, $i\neq j$. If $o_j$is replaced byo_{random}as a representative object and p is closest to o_{random}, then p is reassigned
too_{random}.



Four cases of the cost function for *k*-medoids clustering

4.4.2 The *k*-Medoids Algorithm:

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

Input:

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

Output: A set of k clusters.

Method:

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

The *k*-medoids method ismore robust than *k*-means in the presence of noise and outliers, because a medoid is lessinfluenced by outliers or other extreme values than a mean. However, its processing ismore costly than the *k*-means method.

4.5 Hierarchical Clustering Methods:

- A hierarchical clustering method works by grouping data objects into a tree of clusters.
- The quality of a pure hierarchical clusteringmethod suffers fromits inability to performadjustment once amerge or split decision hasbeen executed. That is, if a particular merge or split decision later turns out to have been apoor choice, the method cannot backtrack and correct it.

Hierarchical clustering methods can be further classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion.

4.5.1 Agglomerative hierarchical clustering:

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

4.5.2 Divisive hierarchical clustering:

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

4.6 Constraint-Based Cluster Analysis:

Constraint-based clustering finds clusters that satisfy user-specified preferences orconstraints. Depending on the nature of the constraints, constraint-based clusteringmay adopt rather different approaches.

There are a few categories of constraints.

- **Constraints on individual objects:**

  We can specify constraints on the objects to beclustered. In a real estate application, for example, one may like to spatially cluster only those luxury mansions worth over a million dollars. This constraint confines the setof objects to be clustered. It can easily be handled by preprocessing after which the problem reduces to an instance ofunconstrained clustering.

- **Constraints on the selection of clustering parameters:**

  A user may like to set a desired range for each clustering parameter. Clustering parameters are usually quite specific to the given clustering algorithm. Examples of parameters include k, the desired numberof clusters in a k-means algorithm; or e the radius and the minimum number of points in the DBSCAN algorithm. Although such user-specified parameters may strongly influence the clustering results, they are usually confined to the algorithm itself. Thus, their fine tuning and processing are usually not considered a form of constraint-based clustering.

- **Constraints on distance or similarity functions:**

  We can specify different distance orsimilarity functions for specific attributes of the objects to be clustered, or differentdistance measures for specific pairs of objects.When clustering sportsmen, for example,we may use different weighting schemes for height, body weight, age, and skilllevel. Although this will likely change the mining results, it may not alter the clusteringprocess per se. However, in some cases, such changes may make the evaluationof the distance function nontrivial, especially when it is tightly intertwined with the clustering process.

- **User-specified constraints on the properties of individual clusters:**
  A user may like tospecify desired characteristics of the resulting clusters, which may strongly influencethe clustering process.
- **Semi-supervised clustering based on partial supervision:**
  The quality of unsupervisedclustering can be significantly improved using some weak form of supervision.This may be in the formof pairwise constraints (i.e., pairs of objects labeled as belongingto the same or different cluster). Such a constrained clustering process is calledsemi-supervised clustering.

4.7 Outlier Analysis:

- There exist data objects that do not comply with the general behavior or model of the data. Such data objects, which are grossly different from or inconsistent with the remaining set of data, are called outliers.
- Many data mining algorithms try to minimize the influence of outliers or eliminate them all together. This, however, could result in the loss of important hidden information because one person's noise could be another person's signal. In other words, the outliers may be of particular interest, such as in the case of fraud detection, where outliers may indicate fraudulent activity. Thus, outlier detection and analysis is an interesting data mining task, referred to as outlier mining.
- It can be used in fraud detection, for example, by detecting unusual usage of credit cards or telecommunication services. In addition, it is useful in customized marketing for identifying the spending behavior of customers with extremely low or extremely high incomes, or in medicalanalysis for finding unusual responses to various medical treatments.

Outlier mining can be described as follows: Given a set of n data points or objectsand k, the expected number of outliers, find the top k objects that are considerablydissimilar, exceptional, or inconsistent with respect to the remaining data. The outliermining problem can be viewed as two subproblems:

- Define what data can be considered as inconsistent in a given data set, and
- Find an efficient method to mine the outliers so defined.

Types of outlier detection:

- Statistical Distribution-Based Outlier Detection
- Distance-Based Outlier Detection
- Density-Based Local Outlier Detection
- Deviation-Based Outlier Detection

4.7.1 Statistical Distribution-Based Outlier Detection:

The statistical distribution-based approach to outlier detection assumes a distributionor probability model for the given data set (e.g., a normal or Poisson distribution) andthen identifies outliers with respect to the model using a discordancy test. Application ofthe test requires knowledge of the data set parameters knowledge of distribution parameters such as the mean and variance and theexpected number of outliers.

A statistical discordancy test examines two hypotheses:

- A working hypothesis
- An alternative hypothesis

A working hypothesis, H, is a statement that the entire data set of n objects comes from an initial distribution model, F, that is,

$$H : o_i \in F, \quad \text{where } i = 1, 2, \ldots, n.$$

The hypothesis is retained if there is no statistically significant evidence supporting its rejection. A discordancy test verifies whether an object, oi, is significantly large (or small) in relation to the distribution F. Different test statistics have been proposed for use as a discordancy test, depending on the available knowledge of the data. Assuming that some statistic, T, has been chosen for discordancy testing, and the value of the statistic for object oi is vi, then the distribution of T is constructed. Significance probability, SP(vi)=Prob(T > vi), is evaluated. If SP(vi) is sufficiently small, then oi is discordant and the working hypothesis is rejected.

An alternative hypothesis, H, which states that o_i comes from another distribution model, G, is adopted. The result is very much dependent on which model F is chosen because $o_i$may be an outlier under one model and a perfectly valid value under another. The alternative distribution is very important in determining the power of the test, that is, the probability that the working hypothesis is rejected when oi is really an outlier.

There are different kinds of alternative distributions.

- **Inherent alternative distribution:**

  In this case, the working hypothesis that all of the objects come from distribution F is rejected in favor of the alternative hypothesis that all of the objects arise from another distribution, G:

  H :oi € G, where i = 1, 2,…, n

  F and G may be different distributions or differ only in parameters of the same distribution.

  There are constraints on the form of the *G* distribution in that it must have potential to produce outliers. For example, it may have a different mean or dispersion, or a longer tail.

- **Mixture alternative distribution:**

  The mixture alternative states that discordant values are not outliers in the F population, but contaminants from some other population,

  G. In this case, the alternative hypothesis is

$$\overline{H} : o_i \in (1-\lambda)F + \lambda G, \quad \text{where } i = 1, 2, \ldots, n.$$

- **Slippage alternative distribution:**

  This alternative states that all of the objects (apart from some prescribed small number) arise independently from the initial model, F, with its given parameters, whereas the remaining objects are independent observations from a modified version of F in which the parameters have been shifted.

There are two basic types of procedures for detecting outliers:

Block procedures:

In this case, either all of the suspect objects are treated as outliersor all of them are accepted as consistent.

Consecutive procedures:

An example of such a procedure is the *insideout*procedure. Its main idea is that the object that is least likely to be an outlier istested first. If it is found to be an outlier, then all of the more extreme values are alsoconsidered outliers; otherwise, the next most extreme object is tested, and so on. Thisprocedure tends to be more effective than block procedures.

4.7.2 Distance-Based Outlier Detection:

The notion of distance-based outliers was introduced to counter the main limitationsimposed by statistical methods. An object, o, in a data set, D, is a distance-based (DB)outlier with parameters pct and dmin,that is, a DB(pct;dmin)-outlier, if at least a fraction,pct, of the objects in D lie at a distance greater than dmin from o. In other words, rather thatrelying on statistical tests, we can think of distance-based outliers as thoseobjects that do not have enoughneighbors, where neighbors are defined based ondistance from the given object. In comparison with statistical-based methods, distancebased outlier detection generalizes the ideas behind discordancy testing for various standarddistributions. Distance-based outlier detection avoids the excessive computationthat can be associated with fitting the observed distribution into some standard distributionand in selecting discordancy tests.

For many discordancy tests, it can be shown that if an object, o, is an outlier accordingto the given test, then o is also a DB(pct, dmin)-outlier for some suitably defined pct anddmin.

For example, if objects that lie three or more standard deviations from the mean are considered to be outliers, assuming a normal distribution, then this definition can be generalized by a DB(0.9988, 0.13s) outlier.

Several efficient algorithms for mining distance-based outliers have been developed.

Index-based algorithm:

Given a data set, the index-based algorithm uses multidimensionalindexing structures, such as R-trees or k-d trees, to search for neighbors of eachobject ***o*** within radius *dmin*around that object. Let *M*be the maximum number ofobjects within the *dmin*-neighborhood of an outlier. Therefore, once*M*+1 neighborsof object ***o*** are found, it is clear that ***o*** is not an outlier. This algorithm has a worst-casecomplexity of *O*(*n*2*k*), where *n* is the number of objects in the data set and *k* is thedimensionality. The index-based algorithm scales well as *k* increases. However, thiscomplexity evaluation takes only the search time into account, even though the taskof building an index in itself can be computationally intensive.

Nested-loop algorithm:

The nested-loop algorithm has the same computational complexityas the index-based algorithm but avoids index structure construction and triesto minimize the number of I/Os. It divides the memory buffer space into two halvesand the data set into several logical blocks. By carefully choosing the order in whichblocks are loaded into each half, I/O efficiency can be achieved.

Cell-based algorithm:

To avoid$O(n^2)$ computational complexity, a cell-based algorithm was developed for memory-resident data sets. Its complexity is $O(c^k+n)$, where c is a constant depending on the number of cells and k is the dimensionality.

In this method, the data space is partitioned into cells with a side length equal to $\frac{dmin}{2\sqrt{k}}$.

Eachcell has two layers surrounding it. The first layer is one cell thick, while the secondis $\lceil 2\sqrt{k}-1 \rceil$ cells thick, rounded up to the closest integer. The algorithm countsoutliers on a cell-by-cell rather than an object-by-object basis. For a given cell, itaccumulates three counts—the number of objects in the cell, in the cell and the firstlayer together, and in the cell and both layers together. Let's refer to these counts ascell count, cell + 1 layer count, and cell + 2 layers count, respectively.

Let Mbe the maximum number ofoutliers that can exist in the dmin-neighborhood of an outlier.

- An object, **o**, in the current cell is considered an outlier only if cell + 1 layer countis less than or equal to M. If this condition does not hold, then all of the objectsin the cell can be removed from further investigation as they cannot be outliers.
- If cell_+ 2_layers_count is less than or equal to M, then all of the objects in thecell are considered outliers. Otherwise, if this number is more than M, then itis possible that some of the objects in the cell may be outliers. To detect theseoutliers, object-by-object processing is used where, for each object, **o**, in the cell,objects in the second layer of **o** are examined. For objects in the cell, only thoseobjects having no more than M points in their dmin-neighborhoods are outliers.The dmin-neighborhood of an object consists of the object's cell, all of its firstlayer, and some of its second layer.

A variation to the algorithm is linear with respect to n and guarantees that no morethan three passes over the data set are required. It can be used for large disk-residentdata sets, yet does not scale well for high dimensions.

4.7.3 Density-Based Local Outlier Detection:

Statistical and distance-based outlier detection both depend on the overall or globaldistribution of the given set of data points, D. However, data are usually not uniformlydistributed. These methods encounter difficulties when analyzing data with rather different

density distributions.

To define the local outlier factor of an object, we need to introduce the concepts ofk-distance, k-distance neighborhood, reachability distance,13 and local reachability density.

These are defined as follows:

The k-distance of an object p is the maximal distance that p gets from its k-nearestneighbors. This distance is denoted as k-distance(p). It is defined as the distance, d(p, o), between p and an object o 2 D, such that for at least k objects, o_0 2 D, it holds that d(p, o')_d(p, o). That is, there are at least k objects inDthat are as close asor closer to p than o, and for at most k-1 objects, o00 2 D, it holds that d(p;o'') <d(p, o).

That is, there are at most k-1 objects that are closer to p than o. You may bewondering at this point how k is determined. The LOF method links to density-basedclustering in that it sets k to the parameter rMinPts,which specifies the minimumnumberof points for use in identifying clusters based on density.

Here, MinPts (as k) is used to define the local neighborhood of an object, p.

The k-distance neighborhood of an object p is denoted $N_{kdistance(p)(}$p), or N_k(p)for short. By setting k to MinPts, we get N_{MinPts}(p). It contains the MinPts-nearestneighbors of p. That is, it contains every object whose distance is not greater than theMinPts-distance of p.

The reachability distance of an object p with respect to object o (where o is within theMinPts-nearest neighbors of p), is defined as reach

distMinPts(p, o) = max{MinPtsdistance(o), d(p, o)}.

Intuitively, if an object p is far away , then the reachabilitydistance between the two is simply their actual distance. However, if they are sufficientlyclose (i.e., where p is within the MinPts-distance neighborhood of o), thenthe actual distance is replaced by the MinPts-distance of o. This helps to significantlyreduce the statistical fluctuations of d(p, o) for all of the p close to o.

The higher thevalue of MinPts is, the more similar is the reachability distance for objects withinthe same neighborhood.

Intuitively, the local reachability density of p is the inverse of the average reachability density based on the MinPts-nearest neighbors of p. It is defined as

$$lrd_{MinPts}(\boldsymbol{p}) = \frac{|N_{MinPts}(\boldsymbol{p})|}{\Sigma_{\boldsymbol{o} \in N_{MinPts}(\boldsymbol{p})} reach\_dist_{MinPts}(\boldsymbol{p}, \boldsymbol{o})}.$$

The local outlier factor (LOF) of ***p*** captures the degree to which we call ***p*** an outlier.

It is defined as

$$LOF_{MinPts}(\boldsymbol{p}) = \frac{\Sigma_{\boldsymbol{o} \in N_{MinPts}(\boldsymbol{p})} \frac{lrd_{MinPts}(\boldsymbol{o})}{lrd_{MinPts}(\boldsymbol{p})}}{|N_{MinPts}(\boldsymbol{p})|}.$$

It is the average of the ratio of the local reachability density of ***p*** and those of ***p***'s *MinPts*-nearest neighbors. It is easy to see that the lower ***p***'s local reachability density is, and the higher the local reachability density of ***p***'s *MinPts*-nearest neighbors are, the higher *LOF*(***p***) is.

4.7.4 Deviation-Based Outlier Detection:

Deviation-based outlier detection does not use statistical tests or distance-basedmeasures to identify exceptional objects. Instead, it identifies outliers by examining themain characteristics of objects in a group.Objects that "deviate" fromthisdescription areconsidered outliers. Hence, in this approach the term deviations is typically used to referto outliers. In this section, we study two techniques for deviation-based outlier detection.The first sequentially compares objects in a set, while the second employs an OLAPdata cube approach.

Sequential Exception Technique:

The sequential exception technique simulates the way in which humans can distinguishunusual objects from among a series of supposedly like objects. It uses implicit redundancyof the data. Given a data set, D, of n objects, it builds a sequence of subsets,{D1, D2, …,Dm}, of these objects with 2<=m <= n such that

$$D_{j-1} \subset D_j, \quad \text{where } D_j \subseteq D.$$

Dissimilarities are assessed between subsets in the sequence. The technique introducesthe following key terms.

Exception set:

This is the set of deviations or outliers. It is defined as the smallestsubset of objects whose removal results in the greatest reduction of dissimilarity in the residual set.

Dissimilarity function:

This function does not require a metric distance between theobjects. It is any function that, if given a set of objects, returns a lowvalue if the objectsare similar to one another. The greater the dissimilarity among the objects, the higherthe value returned by the function. The dissimilarity of a subset is incrementally computedbased on the subset prior to it in the sequence. Given a subset of *n* numbers, $\{x_1, \ldots, x_n\}$, a possible dissimilarity function is the variance of the numbers in theset, that is,

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2,$$

where x is the mean of the n numbers in the set. For character strings, the dissimilarityfunction may be in the form of a pattern string (e.g., containing wildcard charactersthat is used to cover all of the patterns seen so far. The dissimilarity increases when the pattern covering all of the strings in D_{j-1} does not cover any string in D_j that isnot in D_{j-1}.

Cardinality function:

This is typically the count of the number of objects in a given set.

Smoothing factor:

This function is computed for each subset in the sequence. Itassesses how much the dissimilarity can be reduced by removing the subset from theoriginal set of objects.