SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT-I Data Mining and Warehousing – SIT1301

DATA WAREHOUSING

Data warehousing Components –Building a Data warehouse –- Multi Dimensional Data Model – OLAP operations in Multi Dimensional Data model-Three Tier Data warehouse architecture- Schemas for multi dimensional data model-Online Analytical processing(OLAP)- OLAP vs OLTP Integrated OLAM and OLAP Architecture

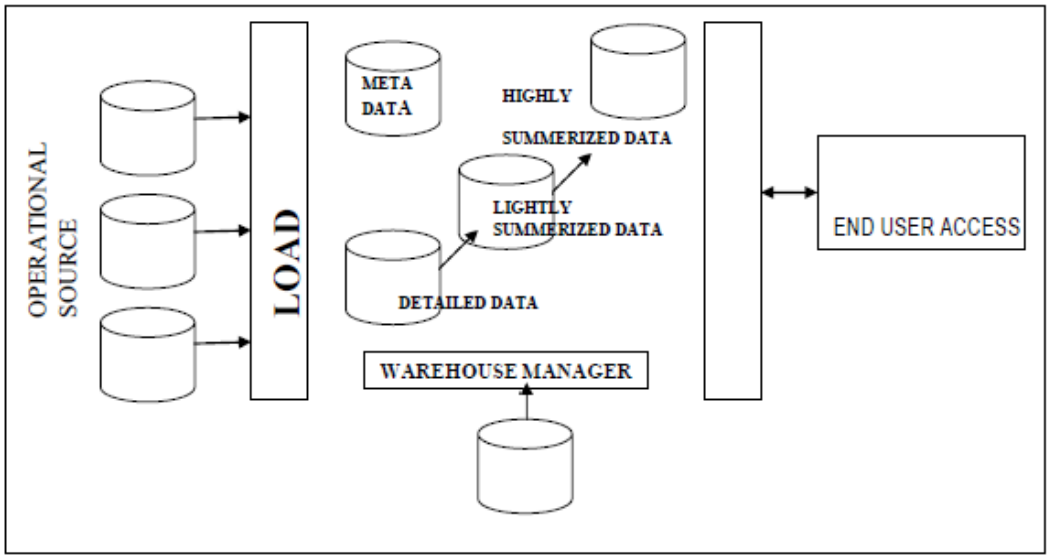
DATA WAREHOUSE COMPONENTS & ARCHITECTURE

The data in a data warehouse comes from operational systems of the organization as well as from other external sources. These are collectively referred to as ***source systems***. The data ***extracted*** from source systems is stored in a area called ***data staging area***, where the data is cleaned, ***transformed***, combined, de-duplicated to prepare the data for us in the data warehouse. The data staging area is generally a collection of machines where simple activities like sorting and sequential processing takes place. The data staging area does not provide any query or presentation services. As soon as a system provides query or presentation services, it is categorized as a ***presentation server***. A presentation server is the target machine on which the data is ***loaded*** from the data staging area organized and stored for direct querying by end users, report writers and other applications. The three different kinds of systems that are required for a data warehouse are:

1. Source Systems
2. Data Staging Area
3. Presentation servers

The data travels from source systems to presentation servers via the data staging area. The entire process is popularly known as ETL (extract, transform, and load) or ETT (extract, transform, and transfer). Oracle's ETL tool is called Oracle Warehouse Builder (OWB) and MS SQL Server's ETL tool is called Data Transformation Services (DTS).

A typical architecture of a data warehouse is shown below:



Each component and the tasks performed by them are explained below:

1. OPERATIONAL DATA

The sources of data for the data warehouse is supplied from:

(i) The data from the mainframe systems in the traditional network and hierarchical format.

(ii) Data can also come from the relational DBMS like Oracle, Informix.

(iii) In addition to these internal data, operational data also includes external data obtained from commercial databases and databases associated with supplier and customers.

2. LOAD MANAGER

The load manager performs all the operations associated with extraction and loading data into the data warehouse. These operations include simple transformations of the data to prepare the data for entry into the warehouse. The size and complexity of this component will vary between data warehouses and may be constructed using a combination of vendor data loading tools and custom built programs.

4. WAREHOUSE MANAGER

The warehouse manager performs all the operations associated with the management of data in the warehouse. This component is built using vendor data management tools and custom built programs. The operations performed by warehouse manager include:

(i) Analysis of data to ensure consistency

(ii) Transformation and merging the source data from temporary storage into data warehouse tables

(iii) Create indexes and views on the base table.

(iv) Denormalization

(v) Generation of aggregation

(vi) Backing up and archiving of data

In certain situations, the warehouse manager also generates query profiles to determine which indexes ands aggregations are appropriate.

4. QUERY MANAGER

The query manager performs all operations associated with management of user queries. This component is usually constructed using vendor end-user access tools, data warehousing monitoring tools, database facilities and custom built programs. The complexity of a query manager is determined by facilities provided by the end-user access tools and database.

5. DETAILED DATA

This area of the warehouse stores all the detailed data in the database schema. In most cases detailed data is not stored online but aggregated to the next level of details. However the detailed data is added regularly to the warehouse to supplement the aggregated data.

6. LIGHTLY AND HIGHLY SUMMERIZED DATA

The area of the data warehouse stores all the predefined lightly and highly summarized (aggregated) data generated by the warehouse manager. This area of the

warehouse is transient as it will be subject to change on an ongoing basis in order to respond to the changing query profiles. The purpose of the summarized information is to speed up the query performance. The summarized data is updated continuously as new data is loaded into the warehouse.

7. ARCHIVE AND BACK UP DATA

This area of the warehouse stores detailed and summarized data for the purpose of archiving and back up. The data is transferred to storage archives such as magnetic tapes or optical disks.

8. META DATA

The data warehouse also stores all the Meta data (data about data) definitions used by all processes in the warehouse. It is used for variety of purposed including:

(i) The extraction and loading process – Meta data is used to map data sources to a common view of information within the warehouse.

(ii) The warehouse management process – Meta data is used to automate the production of summary tables.

(iii) As part of Query Management process Meta data is used to direct a query to the most appropriate data source.

The structure of Meta data will differ in each process, because the purpose is different. More about Meta data will be discussed in the later Lecture Notes.

9. END-USER ACCESS TOOLS

The principal purpose of data warehouse is to provide information to the business managers for strategic decision-making. These users interact with the warehouse using end user access tools. The examples of some of the end user access tools can be:

(i) Reporting and Query Tools

(ii) Application Development Tools

(iii) Executive Information Systems Tools

(iv) Online Analytical Processing Tools

(v) Data Mining Tools

BUILDING A DATA WAREHOUSE

The ETL (Extract Transformation Load) process

In this section we will discussed about the 4 major process of the data warehouse. They are extract (data from the operational systems and bring it to the data warehouse), transform (the data into internal format and structure of the data warehouse), cleanse (to make sure it is ofsufficient quality to be used for decision making) and load (cleanse data is put into the data warehouse).

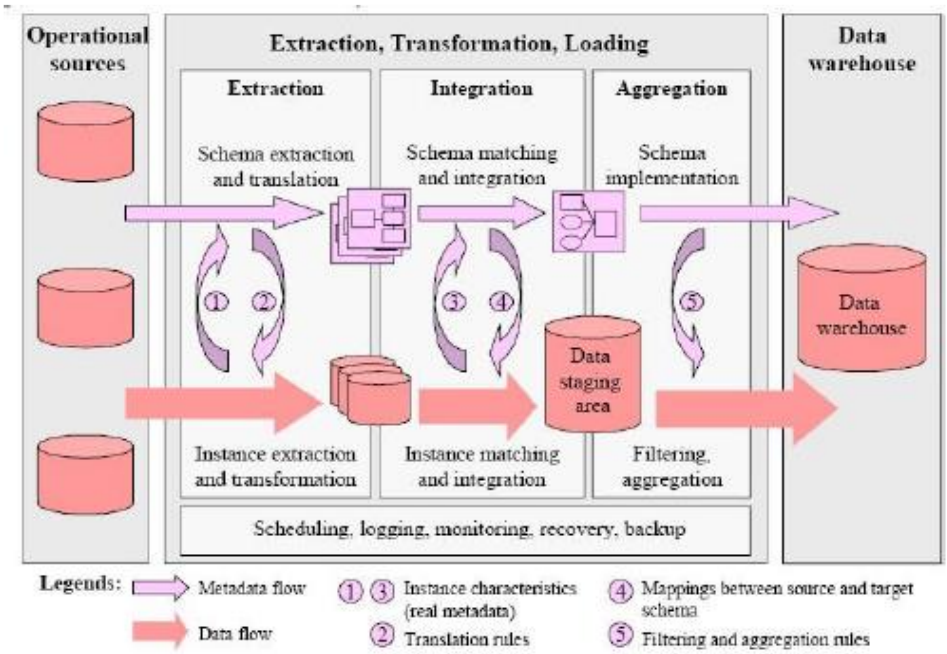


Figure 1. Steps of building a data warehouse: the ETL process

The four processes from extraction through loading often referred collectively as **Data Staging**.

EXTRACT

Some of the data elements in the operational database can be reasonably be expected to be useful in the decision making, but others are of less value for that purpose. For this reason, it is necessary to extract the relevant data from the operational database before bringing into the data warehouse. Many commercial tools are available to help with the

extraction process. **Data Junction** is one of the commercial products. The user of one of these tools typically has an easy- to-use windowed interface by which to specify the following:

(i) Which files and tables are to be accessed in the source database?

(ii) Which fields are to be extracted from them? This is often done internally by SQL Select statement.

(iii) What are those to be called in the resulting database?

(iv) What is the target machine and database format of the output?

(v) On what schedule should the extraction process be repeated?

TRANSFORM

The operational databases developed can be based on any set of priorities, which keeps changing with the requirements. Therefore those who develop data warehouse based on these databases are typically faced with inconsistency among their data sources. Transformation process deals with rectifying any inconsistency (if any).

One of the most common transformation issues is 'Attribute Naming Inconsistency'. It is common for the given data element to be referred to by different data names in different databases. Employee Name may be EMP_NAME in one database, ENAME in the other. Thus one set of Data Names are picked and used consistently in the data warehouse. Once all the data elements have right names, they must be converted to common formats. The conversion may encompass the following:

(i) Characters must be converted ASCII to EBCDIC or vise versa.

(ii) Mixed Text may be converted to all uppercase for consistency.

(iii) Numerical data must be converted in to a common format.

(iv) Data Format has to be standardized.

(v) Measurement may have to convert. (Rs/ $)

(vi) Coded data (Male/ Female, M/F) must be converted into a common format.

All these transformation activities are automated and many commercial products are available to perform the tasks. **DataMAPPER**from Applied Database Technologies is one such comprehensive tool.

CLEANSING

Information quality is the key consideration in determining the value of the information. The developer of the data warehouse is not usually in a position to change the quality of its underlying historic data, though a data warehousing project can put spotlight on the data quality issues and lead to improvements for the future. It is, therefore, usually necessary to go through the data entered into the data warehouse and make it as error free as possible. This process is known as **Data Cleansing**.

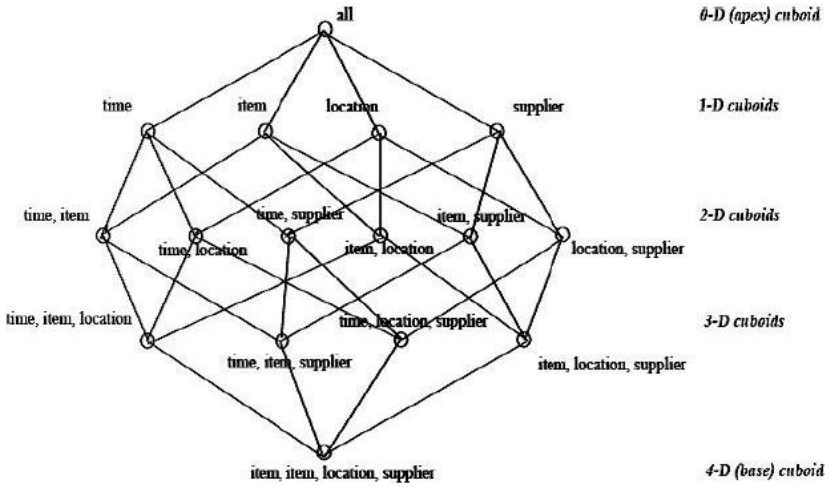
Data Cleansing must deal with many types of possible errors. These include missing data and incorrect data at one source; inconsistent data and conflicting data when two or more source are involved. There are several algorithms followed to clean the data, which will be discussed in the coming lecture notes.

LOADING

Loading often implies physical movement of the data from the computer(s) storing the source database(s) to that which will store the data warehouse database, assuming it is different. This takes place immediately after the extraction phase. The most common channel for data movement is a high-speed communication link. Ex: Oracle Warehouse Builder is the API from Oracle, which provides the features to perform the ETL task on Oracle Data Warehouse.

Multidimensional Data Model and its operation (OLAP operations)

The most popular data model for data warehouses is a multidimensional model. This model can exist in the form of a star schema, a snowflake schema, or a fact constellation schema. Let's have a look at each of these schema types.



OLAP operations on multidimensional data.

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

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

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

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

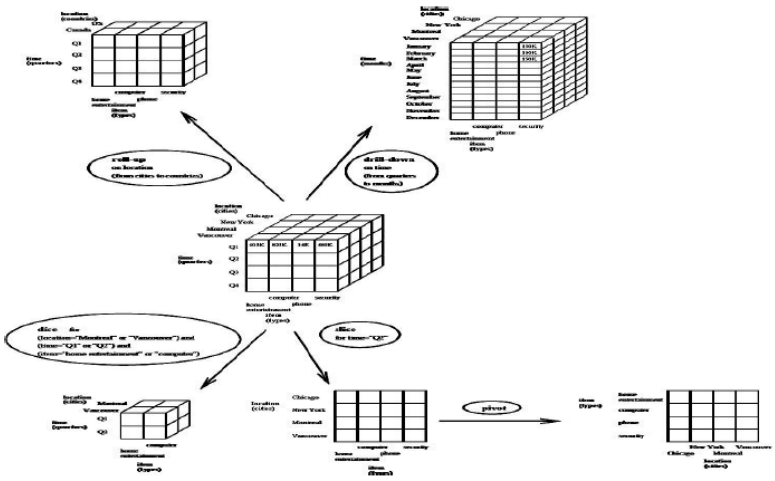
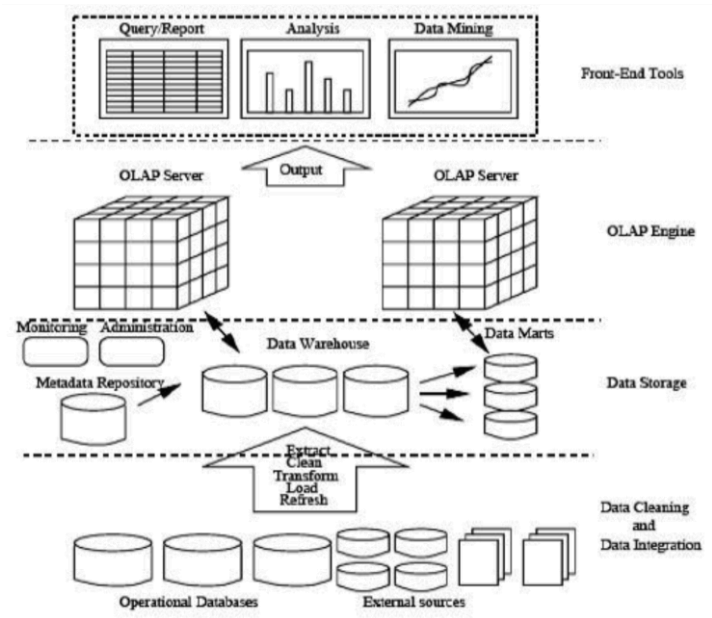


Figure : Examples of typical OLAP operations on multidimensional data.

Three-tier Data warehouse architecture



The bottom tier is a ware-house database server which is almost always a relational database system. The middle tier is an OLAP server which is typically implemented using either (1) a Relational OLAP (ROLAP) model, (2) a Multidimensional OLAP (MOLAP)

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

From the architecture point of view, there are three data warehouse models: the enterprise warehouse, the data mart, and the virtual warehouse.

- **Enterprise warehouse**: An enterprise warehouse collects all of the information about subjects spanning the entire organization. It provides corporate-wide data integration, usually from one or more operational systems or external information providers, and is cross-functional in scope. It typically contains detailed data as well as summarized data, and can range in size from a few gigabytes to hundreds of gigabytes, terabytes, or beyond.

- **Data mart:** A data mart contains a subset of corporate-wide data that is of value to a specific group of users. The scope is connected to specific, selected subjects. For example, a marketing data mart may connect its subjects to customer, item, and sales. The data contained in data marts tend to be summarized. Depending on the source of data, data marts can be categorized into the following two classes:

  (i) Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area.

  (ii) Dependent data marts are sourced directly from enterprise data warehouses.

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

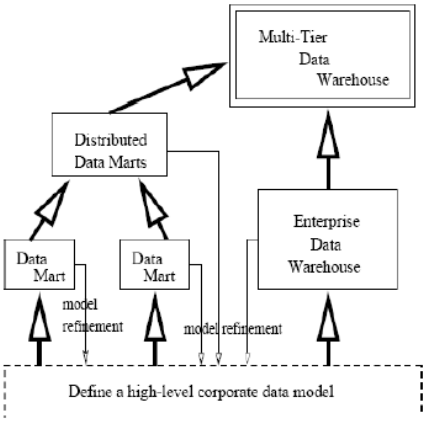


Figure: A recommended approach for data warehouse development

Multi-dimensional Schemas

- Two common multi-dimensional schemas are

1. Star schema

- Consists of a fact table with a single table for each dimension

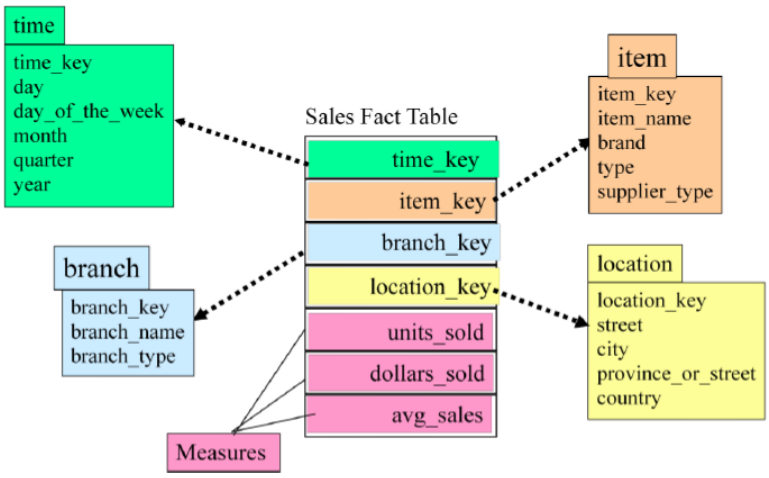
2. Snowflake Schema

- It is a variation of star schema, in which the dimensional tables from a star schema are organized into a hierarchy by normalizing them.

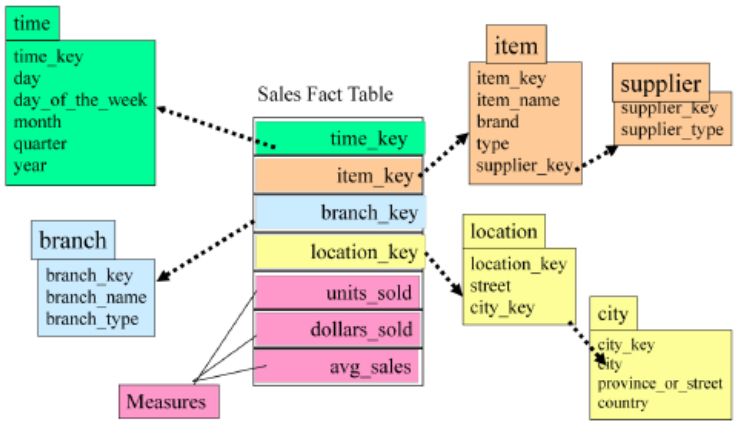
3. Fact constellations

- Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

Star Schema

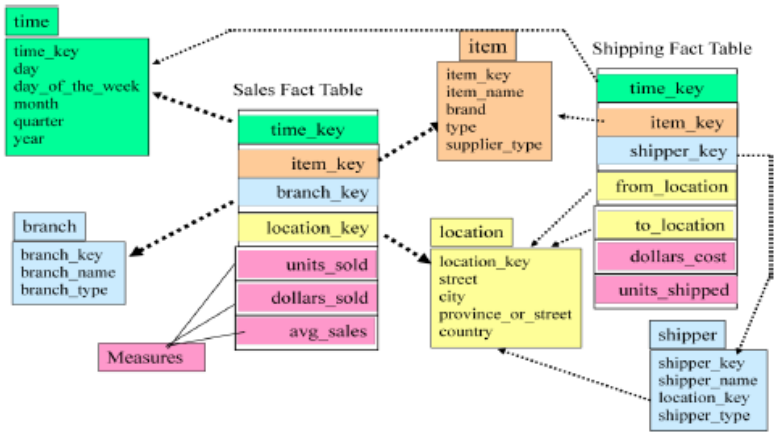


Snow flake Schema



Fact constellation

- Fact constellation is a set of tables that share some dimension tables. However, fact constellations limit the possible queries for the warehouse.



Features of OLTP and OLAP

The major distinguishing features between OLTP and OLAP are summarized as follows.

1. **Users and system orientation**: An OLTP system is customer-oriented and is used for transaction and query processing by clerks, clients, and information technology professionals. An OLAP system is market-oriented and is used for data analysis by knowledge workers, including managers, executives, and analysts.

2. **Data contents**: An OLTP system manages current data that, typically, are too detailed to be easily used for decision making. An OLAP system manages large amounts of historical data, provides facilities for summarization and aggregation, and stores and manages information at different levels of granularity. These features make the data easier for use in informed decision making.

3. **Database design**: An OLTP system usually adopts an entity-relationship (ER) data model and an application oriented database design. An OLAP system typically adopts either a star or snowflake model and a subject-oriented database design.

4. **View**: An OLTP system focuses mainly on the current data within an enterprise or department, without referring to historical data or data in different organizations. In contrast, an OLAP system often spans multiple versions of a database schema. OLAP systems also deal with information that originates from different organizations, integrating information from many data stores. Because of their huge volume, OLAP data are stored on multiple storage media.

5. **Access patterns**: The access patterns of an OLTP system consist mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms. However, accesses to OLAP systems are mostly read-only operations although many could be complex queries.

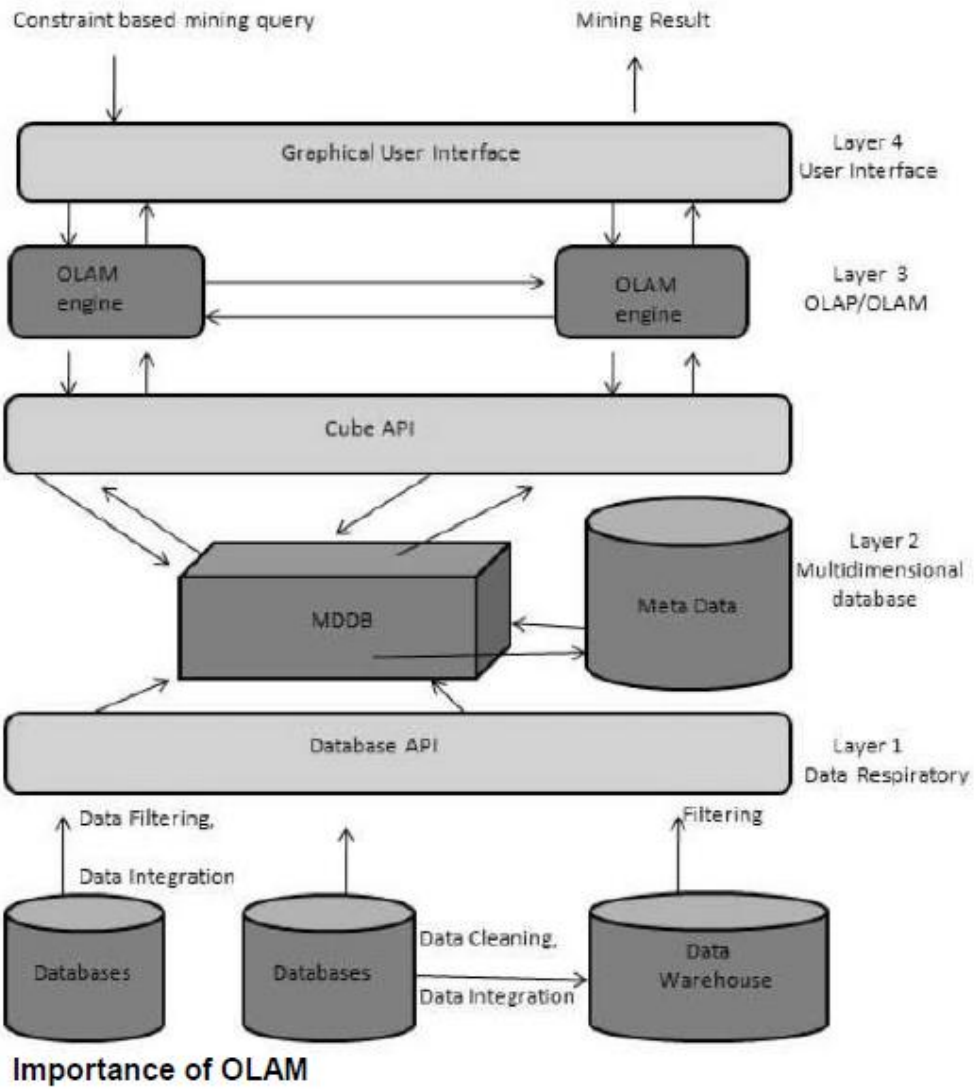
Comparison between OLTP and OLAP systems

Features	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst) long term informational requirements, decision support
Function	day-to-day operations	star/snowflake, subject-oriented
DB design	E-R based, application-oriented	historical; accuracy maintained over time
Data	current; guaranteed up-to-date	summarized, consolidated
Summarization	primitive, highly detailed	summarized, multidimensional
View	detailed, flat relational	complex query
Unit of work	short, simple transaction	mostly read

Access Focus	read / write	information out
Operations	index / hash on primary key	lots of scans
# of records accessed	tens	millions
# of uses	thousands	hundreds
DB size	100 MB to GB	100 GB to TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughout	query throughput, response time

Integrated OLAP and OLAM Architecture

Online Analytical Mining integrates with Online Analytical Processing with data mining and mining knowledge in multidimensional databases. Here is the diagram that shows the integration of both OLAP and OLAM –



Importance of OLAM

OLAM is important for the following reasons −

- **High quality of data in data warehouses** − The data mining tools are required to work on integrated, consistent, and cleaned data. These steps are very costly in the preprocessing of data. The data warehouses constructed by such preprocessing are valuable sources of high quality data for OLAP and data mining as well.
- **Available information processing infrastructure surrounding data warehouses** − Information processing infrastructure refers to accessing, integration, consolidation, and transformation of multiple heterogeneous databases, web-accessing and service facilities, reporting and OLAP analysis tools.

- **OLAP−based exploratory data analysis** − Exploratory data analysis is required for effective data mining. OLAM provides facility for data mining on various subset of data and at different levels of abstraction.

- **Online selection of data mining functions** − Integrating OLAP with multiple data mining functions and online analytical mining provide users with the flexibility to select desired data mining functions and swap data mining tasks dynamically.

QUESTIONS

PART A

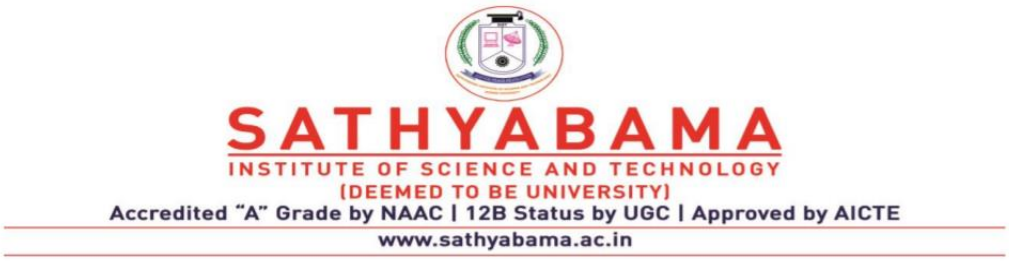
1. Explain three types of dimensions in spatial data cube?
2. List out the challenges for effective resource and knowledge discovery from web?
3. What is information retrieval? Explain the methods supporting information retrieval?
4. List out all the data mining applications?
5. "Is data mining a threat to privacy and data security"?
6. What are the data mining tasks supported by DBminer system?
7. Explain the trends in data mining?
8. Explain the commercially available data mining tools in the market?

PART B

1. Explain mining spatial databases in detail with appropriate example?
2. Explain how to mine text databases with example?
3. What is web usage mining? How to classify the web documents automatically?
4. Explain any two data mining applications with example?
5. Explain the social impacts of data mining with real-time example?
6. What is DBminer? Explain the system architecture. How KDD process supports the DBminer?

TEXT/REFERENCE BOOKS

1. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", 2nd Edition, Elsevier 2007.
2. Alex Berson and Stephen J. Smith, "Data Warehousing, Data Mining & OLAP", Tata McGrew Hill, 2007.
3. www.tutorialspoint.com/dwh/dwh.overview.htm
4. http://study.com/academy/lesson/data-warehousing-and-data-minig-information-for-business-intelligence.html
5. http://www.dei.unpd_it/-capt/SVMATERIALE/DWDM0405.pdf
6. Data mining Concepts and Techniques, Book by Jewei Han and Kamber.



SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT-2Data Mining and Warehousing – SIT1301

DATA MINING

Introduction – Data – Types of Data – Steps in KDD- System Architecture –Data Mining Functionalities – Classification of Data Mining Systems – Data Mining Task Primitives – Integration of a Data Mining System with a Data Warehouse – Issues –Data Preprocessing- Data Mining Application.

What is Data?

- Collection of data objects and their attributes
  - An attribute is a property or characteristic of an object
    - Examples: eye color of a person, temperature, etc.
    - Attribute is also known as variable, field, characteristic, or feature
- A collection of attributes describe an object
    - Object is also known as record, point, case, sample, entity, or instance

Attributes

Objects

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Attribute Values

- Attribute values are numbers or symbols assigned to an attribute
- Distinction between attributes and attribute values
  - Same attribute can be mapped to different attribute values
    - Example: height can be measured in feet or meters

– Different attributes can be mapped to the same set of values

- Example: Attribute values for ID and age are integers
- But properties of attribute values can be different

– ID has no limit but age has a maximum and minimum value

Types of Attributes

- There are different types of attributes

– Nominal

- Examples: ID numbers, eye color, zip codes

– Ordinal

- Examples: rankings (e.g., taste of potato chips on a scale from 1-10), grades, height in {tall, medium, short}

– Interval

- Examples: calendar dates, temperatures in Celsius or Fahrenheit.

– Ratio

Examples: temperature in Kelvin, length, time, counts

Evolution of Database Technology

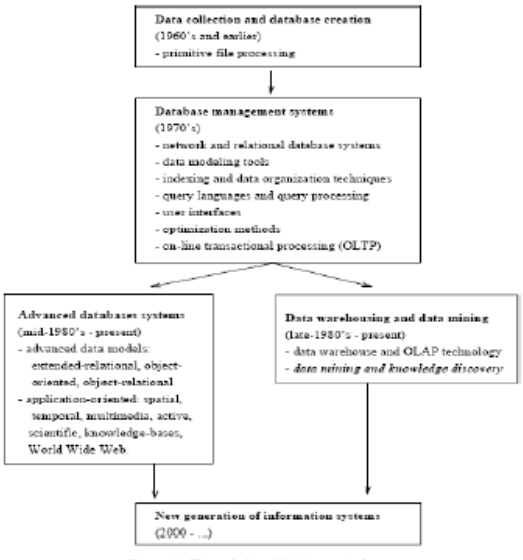


Figure 1.1: The evolution of database technology

Knowledge Discovery in Databases or KDD

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

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

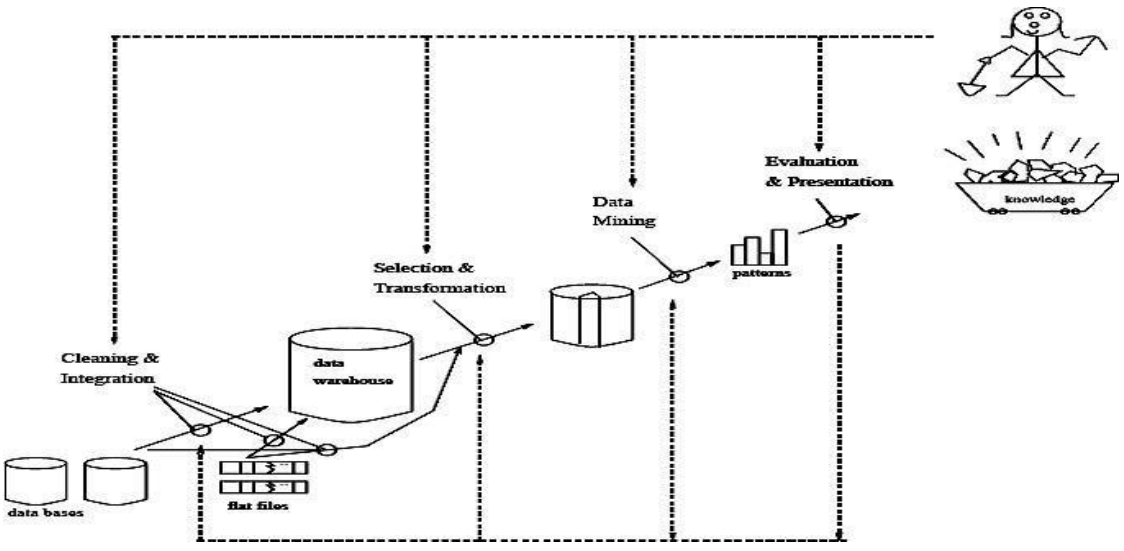


Figure: Data mining as a process of knowledge discovery.

Architecture of a typical data mining system.

The architecture of a typical data mining system may have the following major components

1. **Database, data warehouse, or other information repository.** This is one or a set of databases, data warehouses, spread sheets, or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.

2. **Database or data warehouse server.** The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.

3. **Knowledge base.** This is the domain knowledge that is used to guide the search, or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, 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.

4. **Data mining engine**. This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association analysis, classification, evolution and deviation analysis.

5. **Pattern evaluation module**. This component typically employs interestingness measures and interacts with the data mining modules so as to focus the search towards interesting patterns. It may access interestingness thresholds stored in the knowledge base. Alternatively, the pattern evaluation module may be integrated with the mining module, depending on the implementation of the data mining method used.

6. **Graphical user interface.** This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results.

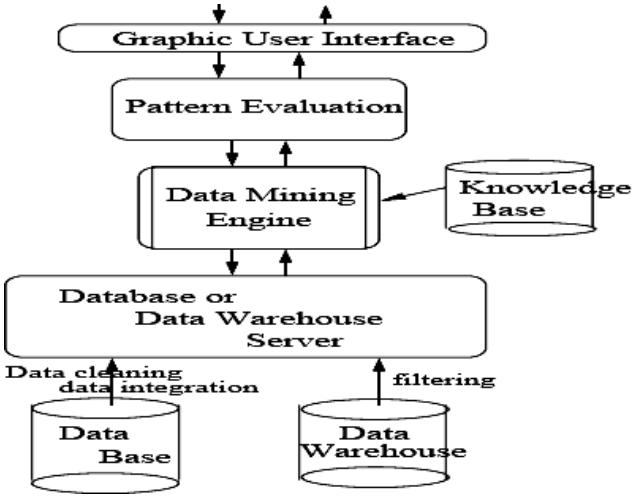


Figure: Architecture of a typical data mining system

Data mining functionalities

Data mining functionalities are used to specify the kind of patterns to be found in data mining tasks. In general, data mining tasks can be classified into two categories.

Descriptive and Predictive

Descriptive mining tasks characterize the general properties of the data in the database. Predictive mining tasks perform inference on the current data in order to make predictions.

In some cases, users may have no idea of which kinds of patterns in their data may be interesting, and hence may like to search for several different kinds of patterns in parallel. Thus it is important to have a data mining system that can mine multiple kinds of patterns to accommodate di_erent user expectations or applications. Furthermore, data mining systems should be able to discover patterns at various granularities. To encourage interactive and exploratory mining, users should be able to easily \play" with the output patterns, such as by mouse clicking. Operations that can be speci_ed by simple mouse clicks include adding or dropping a dimension (or an attribute), swapping rows and columns (pivoting, or axis rotation), changing dimension representations (e.g., from a 3-D cube to a sequence of 2-D cross tabulations, or crosstabs), or using OLAP roll-up or drill-down operations alongdimensions. Such operations allow data patterns to be expressed from different angles of view and at multiple levels of abstraction.

Data mining systems should also allow users to specify hints to guide or focus the search for interesting patterns. Since some patterns may not hold for all of the data in the database, a measure of certainty or \trustworthiness" is usually associated with each discovered pattern.

Data mining functionalities, and the kinds of patterns they can discover, are described below.

Concept/class description: characterization and discrimination

Data can be associated with classes or concepts. For example, in the AllElectronics store, classes of items for sale include computers and printers, and concepts of customers include bigSpenders and budgetSpenders. It can be useful to describe individual classes and concepts in summarized, concise, and yet precise terms. Such descriptions of a class or a concept are called class/concept descriptions. These descriptions can be derived via (1) data characterization, by summarizing the data of the class under study (often called the target class) in general terms, or (2) data discrimination, by comparison of the target class with one or a set of comparative classes (often called the contrasting classes), or (3) both data characterization and discrimination.

Data characterization is a summarization of the general characteristics or features of a target class of data. The data corresponding to the user-specified class are typically collected by a database query. For example, to study the characteristics of software products whose sales increased by 10% in the last year, one can collect the data related to such products by executing an SQL query. There are several methods for e_ective data summarization and characterization. For instance, the data cube- based OLAP roll-up operation can be used to perform user-controlled data summarization along a specified dimension. This process is further detailed in Chapter 2 which discusses data warehousing. An attribute- oriented induction technique can be used to perform data generalization and characterization without step-by-step user interaction. The output of data characterization can be presented in various forms. Examples include pie charts, bar charts, curves, multidimensional data cubes, and multidimensional tables, including crosstabs. The resulting descriptions can also be presented as generalized relations, or in rule form (called characteristic rules).

Association analysis

Association analysis is the discovery of association rules showing attribute-value conditions that occur frequently together in a given set of data. Association analysis is widely used for market basket or transaction data analysis. More formally, association rules are of the form X) Y , i.e., \A1 ^ _ _ _ ^Am !B1 ^ _ _ _^Bn", where Ai (for i 2 f1; : : :;mg) and Bj (for j 2 f1; :: :; ng) are attribute-value pairs. The association rule X) Y is interpreted as \database tuples that satisfy the conditions in X are also likely to satisfy the conditions in Y ".

An association between more than one attribute, or predicate (i.e., age, income, and buys). Adopting the terminology used in multidimensional databases, where each attribute is referred to as a dimension,the above rule can be referred to as a multidimensional association rule. Suppose, as a marketing manager of AllElectronics, you would like to determine which items are frequently purchased together within the same transactions. An example of such a rule is

Contains

(T; \computer"))contains(T; \software") [support = 1%; confidence = 50%]

meaning that if a transaction T contains \computer", there is a 50% chance that it contains\software" as well, and 1% of all of the transactions contain both. This association rule involves a single attribute or predicate (i.e., contains) which repeats. Association rules thatcontain a single predicate are referred to as single-dimensional association rules. Dropping the predicate notation, the above rule can be written simply as \computer) software [1%, 50%]".

Classification and prediction

Classification is the processing of finding a set of models (or functions) which describe and distinguish data classes or concepts, for the purposes of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known). The derived model may be represented in various forms, such as classi_cation (IF-THEN) rules, decision trees, mathematical formulae, or neural networks. A decision tree is aow-

chart-like tree structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distributions. Decision trees can be easily converted to classi_cation rules. A neural network is a collection of linear threshold units that can be trained to distinguish objects of different classes.Classification can be used for predicting the class label of data objects. However, in many applications, one may like to predict some missing or unavailable data values rather than class labels. This is usually the case when the predicted values are numerical data, and is often specifically referred to as prediction. Although prediction may refer to both data value prediction and class label prediction, it is usually con_ned to data value prediction and thus is distinct from classification. Prediction also encompasses the identification of distribution trends based on the available data. Classification and prediction may need to be preceded by relevance analysis which attempts to identify at tributes that do not contribute to the classification or prediction process.

Clustering analysis

Clustering analyzes data objects without consulting a known class label. In general, the class labels are not present in the training data simply because they are not known to begin with. Clustering can be used to generate such labels. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the interclass similarity. That is, clusters of objects are formed so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Each cluster that is formed can be viewed as a class of objects, from which rules can be derived.

Evolution and deviation analysis

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

Interestingness Patterns

A data mining system has the potential to generate thousands or even millions of patterns, or rules. This raises some serious questions for data mining:

A pattern is interesting if (1) it is easily understood by humans, (2) valid on new or test data with some degree of certainty, (3) potentially useful, and (4) novel. A pattern is also interesting if it validates a hypothesis that the user sought to con_rm. An interesting pattern

represents knowledge. Several objective measures of pattern interestingness exist. These are based on the structure of discovered patterns and the statistics underlying them. An objective measure for association rules of the form XU Y is rule support, representing the percentage of data samples that the given rule satisfies. Another objective measure for association rules is confidence, which assesses the degree of certainty of the detected association. It is defined as the conditional probability that a pattern Y is true given that X is true. More formally, support and confidence aredefined as

support(X) Y) = Prob{XUY}g

confidence (X) Y) = Prob{Y |X}g

A classification of data mining systems

Data mining is an interdisciplinary field, the conuence of a set of disciplines including database systems, statistics, machine learning, visualization, and information science. Moreover, depending on the data mining approach used, techniques from other disciplines may be applied, such as neural networks, fuzzy and/or rough set theory, knowledge representation, inductive logic programming, or high performance computing. Depending on the kinds of data to be mined or on the given data mining application, the data mining system may also integrate techniques from spatial data analysis, information retrieval, pattern recognition, image analysis, signal processing, computer graphics, Web technology, economics, or psychology. Because of the diversity of disciplines contributing to data mining, data mining research is expected to generatea large variety of data mining systems. Therefore, it is necessary to provide a clear classi_cation of data mining systems. Such a classi_cation may help potential users distinguish data mining systems and identify those that best match their needs. Data mining systems can be categorized according to various criteria, as follows._ Classi_cation according to the kinds of databases mined. A data mining system can be classi_ed according to the kinds of databases mined. Database systems themselves can be classi_ed according to di_erent criteria (such as data models, or the types of data or applications involved), each of which may require its own data mining technique. Data mining systems can therefore be classi_ed accordingly.

For instance, if classifying according to data models, we may have a relational, transactional, object-oriented, object-relational, or data warehouse mining system. If classifying according to the special types of data handled, we may have a spatial, time-series,

text, or multimedia data mining system, or a World-Wide Web mining system. Other system types include heterogeneous data mining systems, and legacy data mining systems.

Classification according to the kinds of knowledge mined. Data mining systems can be categorized according to the kinds of knowledge they mine, i.e., based on data mining functionalities, such as characterization, discrimination, association, classi_cation, clustering, trend and evolution analysis, deviation analysis, similarity analysis, etc. A comprehensive data mining system usually provides multiple and/or integrated data mining functionalities. Moreover, data mining systems can also be distinguished based on the granularity or levels of abstraction of the knowledge mined, including generalized knowledge (at a high level of abstraction), primitive-level knowledge (at a raw data level), or knowledge at multiple levels (considering several levels of abstraction). An advanced data mining system should facilitate the discovery of knowledge at multiple levels of abstraction.

Classification according to the kinds of knowledge mined

Data mining systems can be categorized according to the kinds of knowledge they mine, i.e., based on data mining functionalities, such as characterization, discrimination, association,classi_cation, clustering, trend and evolution analysis, deviation analysis, similarity analysis, etc. A comprehensive data mining system usually provides multiple and/or integrated data mining functionalities.Moreover, data mining systems can also be distinguished based on the granularity or levels of abstraction of the knowledge mined, including generalized knowledge (at a high level of abstraction), primitive-level knowledge (at a raw data level), or knowledge at multiple levels (considering several levels of abstraction). An advanced data mining system should facilitate the discovery of knowledge at multiple levels of abstraction.

Classification according to the kinds of techniques utilized

Data mining systems can also be categorized according to the underlying data mining techniques employed. These techniques can be described according to the degree of user interaction involved (e.g., autonomous systems, interactive exploratory systems, query-driven systems), or the methods of data analysis employed (e.g., database-oriented or data warehouse-oriented techniques, machine learning, statistics, visualization, pattern recognition, neural networks, and so on). A sophisticated data mining system will often adopt

multiple data mining techniques or work out an e_ective, integrated technique which combines the merits of a few individual approaches.

Data mining primitives.

A data mining query is defined in terms of the following primitives

2. **Task-relevant data:** This is the database portion to be investigated. For example, suppose that you are a manager of All Electronics in charge of sales in the United States and Canada.In particular, you would like to study the buying trends of customers in Canada. Rather than mining on the entire database. These are referred to as relevant attributes

3. **The kinds of knowledge to be mined:** This specifies the data mining functions to be performed, such as characterization, discrimination, association, classification, clustering, or evolution analysis. For instance, if studying the buying habits of customers in Canada, you may choose to mine associations between customer profiles and the items that these customers like to buy

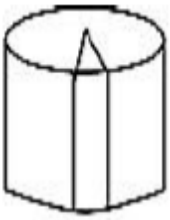
4. **Background knowledge:** Users can specify background knowledge, or knowledge about the domain to be mined. This knowledge is useful for guiding the knowledge discovery process, and for evaluating the patterns found. There are several kinds of background knowledge.

5. **Interestingness measures:** These functions are used to separate uninteresting patterns from knowledge. They may be used to guide the mining process, or after discovery, to evaluate the discovered patterns. Different kinds of knowledge may have different interestingness measures.

6. **Presentation and visualization of discovered patterns:** This refers to the form in which discovered patterns are to be displayed. Users can choose from different forms for knowledge presentation, such as rules, tables, charts, graphs, decision trees, and cubes.

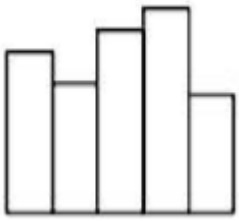
Figure : Primitives for specifying a data mining task.

$$Ent(S) - E(T, S) > \delta$$



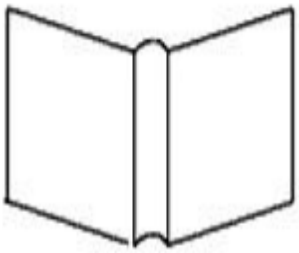
Task –relevant data

- database or data warehouse name
- database tablets or data warehouse cubes
- conditions for data selection
- relevant attributes or dimensions
- data grouping criteria



Knowledge type to be mined

- characterization
- discrimination
- association
- classification / prediction
- clustering

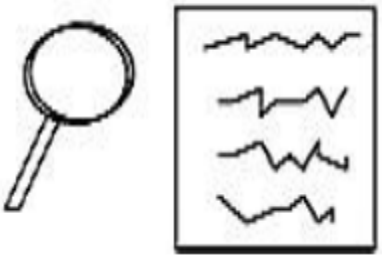


Background knowledge

- concept hierarchies
- user beliefs about relationships in the data

Pattern interestingness measurements

- simplicity
- certainty (e.g. confidence)
- utility (e.g. support)
- novelty



Visualization of discovered patterns

- rules, tables, reports, charts graphs, decision trees and cubes
- drill-down and roll-up

INTEGRATION OF DATAMINING SYSTEM WITH DATA WAREHOUSE

DB and DW systems, possible integration schemes include

- no coupling,
- loose coupling,
- semitight coupling, and
- tight coupling.

1. No coupling

❖ *No coupling* means that a DM system will not utilize any function of a DB or DW system.

❖ It may fetch data from a particular source (such as a file system), process data using some data mining algorithms, and then store the mining results in another file.

2. Loose coupling

- *Loose coupling* means that a DM system will use some facilities of a DB or DW system, fetching data from a data repository ,performing data mining, and then storing the mining results either in a file or in a designated place in a database or data Warehouse.
- Loose coupling is better than no coupling because it can fetch any portion of data stored in databases or data warehouses by using query processing, indexing, and other system facilities.
- It is difficult to achieve high scalability and good performance with large data sets.

3. Semitight coupling

- *Semitight coupling* means that besides linking a DM system to a DB/DW system, a few essential data mining primitives can be provided in the DB/DW system.
- These primitives can include sorting, indexing, aggregation, histogram analysis, multi way join, and precomputation of some essential statistical measures, such as sum, count, max, min,standard deviation.

4. Tight coupling

- *Tight coupling* means that a DM system is smoothly integrated into the DB/DW system.
- The data mining subsystem is treated as one functional component of information system.

Data mining queries and functions are optimized based on mining query analysis, data structures, indexing schemes, and query processing methods of a DB or DW system

Major issues in data mining

The scope of this book addresses major issues in data mining regarding mining methodology, user interaction, performance, and diverse data types. These issues are introduced below:

1. **Mining methodology and user-interaction issues.** These reect the kinds of knowledge mined, the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad-hoc mining, and knowledge visualization.

Mining different kinds of knowledge in databases.

Since different users can be interested in different kinds of knowledge, data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterization, discrimination, association, classification, clustering, trend and deviation analysis, and similarity analysis. These tasks may use the same database in different ways and require the development of numerous data mining techniques.

Interactive mining of knowledge at multiple levels of abstraction.

Since it is difficult to know exactly what can be discovered within a database, the data mining process should be interactive. For databases containing a huge amount of data, appropriate sampling technique can first be applied to facilitate interactive data exploration. Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results. Specifically, knowledge should be mined by drilling- down, rolling-up, and pivoting through the data space and knowledge space

interactively, similar to what OLAP can do on data cubes. In this way, the user can interact with the data mining system to view data and discovered patterns at multiple granularities and from different angles.

Incorporation of background knowledge.

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

Data mining query languages and ad-hoc data mining.

Relational query languages (such as SQL) allow users to pose ad-hoc queries for data retrieval. In a similar vein, high-level data mining query languages need to be developed to allow users to describe ad-hoc data mining tasks by facilitating the speci_cation of the relevant sets of data for analysis, the domain knowledge, the kinds of knowledge to be mined, and the conditions and interestingness constraints to be enforced on the discovered patterns. Such a language should be integrated with a database or data warehouse query language, and optimized for e_cient and exible data mining.

Presentation and visualization of data mining results.

Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that the knowledge can be easily understood and directly usable by humans. This is especially crucial if the data mining system is to be interactive. This requires the system to adopt expressive knowledge representation techniques, such as trees, tables, rules, graphs, charts, crosstabs, matrices, or curves.

Handling outlier or incomplete data.

The data stored in a database may reect outliers | noise, exceptional cases, or incomplete data objects. These objects may confuse the analysis process, causing over_tting of the data to the knowledge modelconstructed. As a result, the accuracy of the discovered patterns can be poor. Data cleaning methods and data analysis methods which can handle outliers are required. While most methods discard outlier data, such data may be of interest in

itself such as in fraud detection for _nding unusual usage of tele-communication services or credit cards. This form of data analysis is known as outlier mining.

Pattern evaluation: the interestingness problem.

A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, representing common knowledge or lacking novelty. Several challenges remain regarding the development of techniques to assess the interestingness of discovered patterns, particularly with regard to subjective measures which estimate the value of patterns with respect to a given user class, based on user beliefs or expectations. The use of interestingness measures to guide the discovery process and reduce the search space is another active area of research.

2. **Performance issues.** These include efficiency, scalability, and parallelization of data mining algorithms.

Efficiency and scalability of data mining algorithms.

To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable. That is, the running time of a data mining algorithm must be predictable and acceptable in large databases. Algorithms with exponential or even medium-order polynomial complexity will not be of practical use. From a database perspective on knowledge discovery, efficiency and scalability are key issues in the implementation of data mining systems. Many of the issues discussed above under mining methodology and user-interaction must also consider efficiency and scalability.

Parallel, distributed, and incremental updating algorithms.

The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of paralleland distributed data mining algorithms. Such algorithms divide the data into partitions, which are processed in parallel. The results from the partitions are then merged. Moreover, the high cost of some data mining processes promotes the need for incremental data miningalgorithms which incorporate database updates without having to mine the entire data againfrom scratch". Such algorithms perform knowledge modification incrementally to amend and strengthen what was previously discovered.

3. Issues relating to the diversity of database types.

Handling of relational and complex types of data.

There are many kinds of data stored in databases and data warehouses. Since relational databases and data warehouses are widely used, the development of efficient and effective data mining systems for such data is important. However, other databases may contain complex data objects, hypertext and multimedia data, spatial data, temporal data, or transaction data. It is unrealistic to expect one system to mine all kinds of data due to the diversity of data types and different goals of data mining. Specific data mining systems should be constructed for mining specific kinds of data. Therefore, one may expect to have different data mining systems for different kinds of data.

Mining information from heterogeneous databases and global information systems. Local and wide-area computer networks (such as the Internet) connect many sources of data, forming huge, distributed, and heterogeneous databases. The discovery of knowledge from di_erent sources of structured, semi-structured, or unstructured data with diverse data semantics poses great challenges to data mining. Data mining may help disclose high-level data regularities in multiple heterogeneous databases that are unlikely to be discovered by simple query systems and may improve information exchange and interoperability in heterogeneous databases.

Data Preprocessing

Data cleaning

Data cleaning routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data.

(i) Missing values

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

2. **Fill in the missing value manually:** In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.

3. **Use a global constant to fill in the missing value:** Replace all missing attribute values by the same constant, such as a label like “Unknown". If missing values are replaced by, say, “Unknown", then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common - that of “Unknown". Hence, although this method is simple, it is not recommended.

4. **Use the attribute mean to fill in the missing value:** For example, suppose that the average income of All Electronics customers is $28,000. Use this value to replace the missing value for income.

5. **Use the attribute mean for all samples belonging to the same class as the given tuple:** For example, if classifying customers according to credit risk, replace the missing value with the average income value for customers in the same credit risk category as that of the given tuple.

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

(ii) Noisy data

Noise is a random error or variance in a measured variable.

1. Binning methods

Binning methods smooth a sorted data value by consulting the “neighborhood", or values around it. The sorted values are distributed into a number of 'buckets', or bins. Because binning methods consult the neighborhood of values, they perform local smoothing. Figure illustrates some binning techniques.

In this example, the data for price are first sorted and partitioned into equi-depth bins (of depth 3). In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is 9. Therefore, each original value in this bin is replaced by the value 9. Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value.

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

(ii) Partition into (equi-width) bins:

- Bin 1: 4, 8, 15
- Bin 2: 21, 21, 24
- Bin 3: 25, 28, 34

(iii) Smoothing by bin means:

- Bin 1: 9, 9, 9,
- Bin 2: 22, 22, 22
- Bin 3: 29, 29, 29

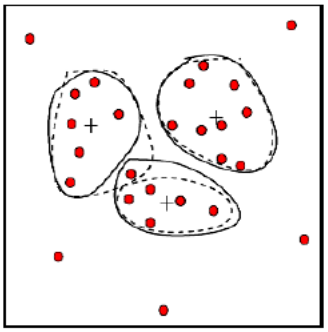
(iv) Smoothing by bin boundaries:

- Bin 1: 4, 4, 15
- Bin 2: 21, 21, 24
- Bin 3: 25, 25, 34

2. Clustering

Outliers may be detected by clustering, where similar values are organized into groups or “clusters”. Intuitively, values which fall outside of the set of clusters may be considered outliers.

Figure: Outliers may be detected by clustering analysis.



3. Combined computer and human inspection: Outliers may be identified through a combination of computer and human inspection. In one application, for example, an information-theoretic measure was used to help identify outlier patterns in a handwritten character database for classification. The measure's value reflected the “surprise" content of the predicted character label with respect to the known label.

Outlier patterns may be informative or “garbage". Patterns whose surprise content is above a threshold are output to a list. A human can then sort through the patterns in the list to identify the actual garbage ones

4. **Regression:** Data can be smoothed by fitting the data to a function, such as with regression. Linear regression involves finding the “best" line to fit two variables, so that one variable can be used to predict the other. Multiple linear regression is an extension of linear regression, where more than two variables are involved and the data are fit to a multidimensional surface.

(iii) Inconsistent data

There may be inconsistencies in the data recorded for some transactions. Some data inconsistencies may be corrected manually using external references. For example, errors made at data entry may be corrected by performing a paper trace. This may be coupled with routines designed to help correct the inconsistent use of codes. Knowledge engineering tools may also be used to detect the violation of known data constraints. For example, knownfunctional dependencies between attributes can be used to find values contradicting the functional constraints.

Data transformation.

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

1. **Normalization,** where the attribute data are scaled so as to fall within a small specified range, such as -1.0 to 1.0, or 0 to 1.0.

There are three main methods for data normalization :**min-max normalization, z-score normalization, and normalization by decimal scaling.**

(i) **Min-max normalization** performs a linear transformation on the original data. Suppose that minA and maxA are the minimum and maximum values of an attribute A. Min-max normalization maps a value v of A to v0 in the range [new minA; new maxA] by computing

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

(ii) **z-score normalization (or zero-mean normalization),** the values for an attribute A are normalized based on the mean and standard deviation of A. A value v of A is normalized to v0 by computing where mean A and stand dev A are the mean and standard deviation, respectively, of attribute A. This method of normalization is useful when the actual minimum and maximum of attribute A are unknown, or when there are outliers which dominate the min-max normalization.

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

(iii) **Normalization by decimal scaling** normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value ofA. A value v of A is normalized to v0by computing where j is the smallest integer such thatMax(v'|) < 1.

2. **Smoothing,** which works to remove the noise from data? Such techniques include binning, clustering, and regression.

(i) Binning methods

Binning methods smooth a sorted data value by consulting the ”neighborhood", or values around it. The sorted values are distributed into a number of 'buckets', or bins. Because binning methods consult the neighborhood of values, they perform local smoothing. Figure illustrates some binning techniques.

In this example, the data for price are first sorted and partitioned into equi-depth bins (of depth 3). In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is 9. Therefore, each original value in this bin is replaced by the value 9. Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value.

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

(ii) Partition into (equi-width) bins:

- Bin 1: 4, 8, 15

- Bin 2: 21, 21, 24
- Bin 3: 25, 28, 34

(iii) Smoothing by bin means:

- Bin 1: 9, 9, 9,
- Bin 2: 22, 22, 22
- Bin 3: 29, 29, 29

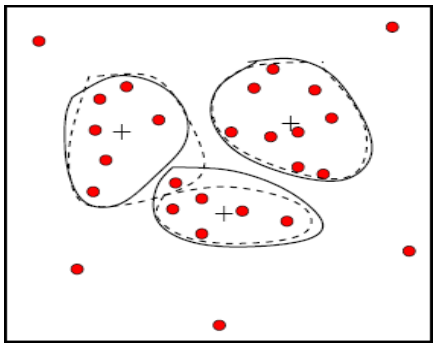
(iv) Smoothing by bin boundaries:

- Bin 1: 4, 4, 15
- Bin 2: 21, 21, 24
- Bin 3: 25, 25, 34

(ii) Clustering

Outliers may be detected by clustering, where similar values are organized into groups or "clusters". Intuitively, values which fall outside of the set of clusters may be considered outliers.

Figure: Outliers may be detected by clustering analysis.



3. **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 annual total amounts.

4. **Generalization of the data**, where low level or 'primitive' (raw) data are replaced by higher level concepts through the use of concept hierarchies. For example, categorical attributes, like street, can be generalized to higher level concepts, like city or county.

Data reduction

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

Strategies for data reduction include the following.

1. **Data cube aggregation,** where aggregation operations are applied to the data in the construction of a data cube.
2. **Dimension reduction**, where irrelevant, weakly relevant or redundant attributes or dimensions may be detected and removed.
3. **Data compression**, where encoding mechanisms are used to reduce the data set size.
4. **Numerosity reduction,** where the data are replaced or estimated by alternative, smaller data representations such as parametric models (which need store only the model parameters instead of the actual data), or nonparametric methods such as clustering, sampling, and the use of histograms.
5. **Discretization and concept hierarchy generation,** where raw data values for attributes are replaced by ranges or higher conceptual levels. Concept hierarchies allow the mining of data at multiple levels of abstraction, and are a powerful tool for data mining.

Data Cube Aggregation

- The lowest level of a data cube
  - the aggregated data for an individual entity of interest
  - e.g., a customer in a phone calling data warehouse.
- Multiple levels of aggregation in data cubes
  - Further reduce the size of data to deal with
- Reference appropriate levels
  - Use the smallest representation which is enough to solve the task
- Queries regarding aggregated information should be answered using data cube, when possible

Dimensionality Reduction

Feature selection (i.e., attribute subset selection):

- Select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
- reduce # of patterns in the patterns, easier to understand

Heuristic methods

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

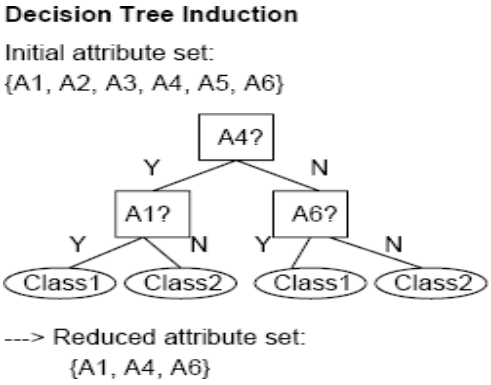
   Forward Selection
   Initial attribute set:
   {A1, A2, A3, A4, A5, A6}
   Initial reduced set:
   {}
   -> {A1}
   --> {A1, A4}
   ---> Reduced attribute set:
   {A1, A4, A6}

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

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

3. **Combination forward selection and backward elimination:** The step-wise forward selection and backward elimination methods can be combined, where at each step one selects the best attribute and removes the

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



Data compression

In data compression, data encoding or transformations are applied so as to obtain a reduced or ”compressed" representation of the original data. If the original data can be reconstructed from the compressed data without any loss of information, the data compression technique used is called lossless. If, instead, we can reconstruct only an approximation of the original data, then the data compression technique is called lossy. The two popular and effective methods of lossy data compression**: wavelet transforms, and principal components analysis.**

Wavelet transforms

The discrete wavelet transform (DWT) is a linear signal processing technique that, when applied to a data vector D, transforms it to a numerically different vector, D0, of wavelet coefficients. The two vectors are of the same length.

The DWT is closely related to the discrete Fourier transform (DFT), a signal processing technique involving sines and cosines. In general, however, the DWT achieves better lossy compression.

The general algorithm for a discrete wavelet transform is as follows.

1. The length, L, of the input data vector must be an integer power of two. This condition can be met by padding the data vector with zeros, as necessary.
2. Each transform involves applying two functions. The first applies some data smoothing, such as a sum or weighted average. The second performs a weighted difference.
3. The two functions are applied to pairs of the input data, resulting in two sets of data of length L=2. In general, these respectively represent a smoothed version of the input data, and the high-frequency content of it.
4. The two functions are recursively applied to the sets of data obtained in the previous loop, until the resulting data sets obtained are of desired length.
5. A selection of values from the data sets obtained in the above iterations are designated the wavelet coefficients of the transformed data.

Principal components analysis

Principal components analysis (PCA) searches for c k-dimensional orthogonal vectors that can best be used to represent the data, where c << N. The original data is thus projected onto a much smaller space, resulting in data compression. PCA can be used as a form of dimensionality reduction. The initial data can then be projected onto this smaller set.

The basic procedure is as follows.

1. The input data are normalized, so that each attribute falls within the same range. This step helps ensure that attributes with large domains will not dominate attributes with smaller domains.
2. PCA computes N orthonormal vectors which provide a basis for the normalized input data. These are unit vectors that each point in a direction perpendicular to the others. These vectors are referred to as the principal components. The input data are a linear combination of the principal components.

3. The principal components are sorted in order of decreasing “significance" or strength. The principal components essentially serve as a new set of axes for the data, providing important information about variance.

4. Since the components are sorted according to decreasing order of “significance", the size of the data can be reduced by eliminating the weaker components, i.e., those with low variance. Using the strongest principal components, it should be possible to reconstruct a good approximation of the original data.

Numerosityreduction

Regression and log-linear models

Regression and log-linear models can be used to approximate the given data. In linear regression, the data are modeled to fit a straight line. For example, a random variable, Y (called a response variable), can be modeled as a linear function of another random variable, X (called a predictor variable), with the equation where the variance of Y is assumed to be constant. These coefficients can be solved for by the method of least squares, which minimizes the error between the actual line separating the data and the estimate of the line.

Multiple regression is an extension of linear regression allowing a response variable Y to be modeled as a linear function of a multidimensional feature vector.

Log-linear models approximate discrete multidimensional probability distributions. The method can be used to estimate the probability of each cell in a base cuboid for a set of discretized attributes, based on the smaller cuboids making up the data cube lattice

Histograms

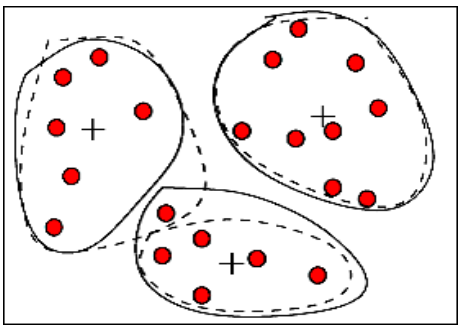
A histogram for an attribute A partitions the data distribution of A into disjoint subsets, or buckets. The buckets are displayed on a horizontal axis, while the height (and area) of a bucket typically reects the average frequency of the values represented by the bucket.

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

2. Equi-depth (or equi-height): In an equi-depth histogram, the buckets are created so that, roughly, the frequency of each bucket is constant (that is, each bucket contains roughly the same number of contiguous data samples).
3. V-Optimal: If we consider all of the possible histograms for a given number of buckets, the V-optimal histogram is the one with the least variance. Histogram variance is a weighted sumof the original values that each bucket represents, where bucket weight is equal to the number of values in the bucket.
4. MaxDiff: In a MaxDiff histogram, we consider the difference between each pair of adjacent values. A bucket boundary is established between each pair for pairs having the ß – 1 largest differences, where is user-specified.

Clustering

Clustering techniques consider data tuples as objects. They partition the objects into groups or clusters, so that objects within a cluster are “similar" to one another and “dissimilar" to objects in other clusters. Similarity is commonly defined in terms of how “close" the objects are in space, based on a distance function. The “quality" of a cluster may be represented by its diameter, the maximum distance between any two objects in the cluster. Centroid distance is an alternative measure of cluster quality, and is defined as the average distance of each cluster object from the cluster centroid.



Sampling

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

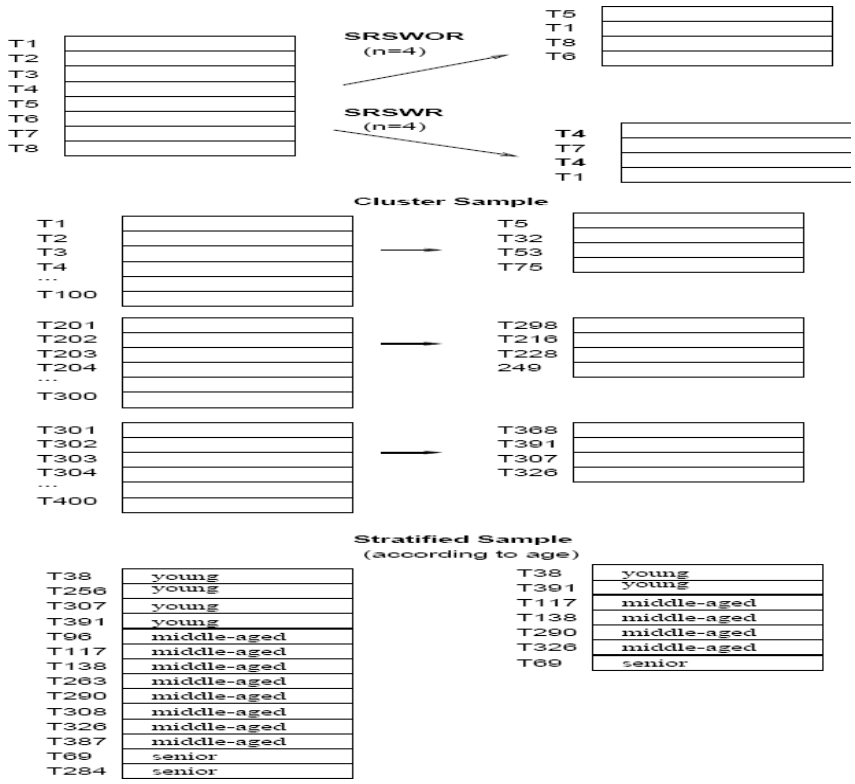
1. **Simple random sample without replacement (SRSWOR) of size n:** This is created by drawing n of the N tuples from D (n < N), where the probably of drawing any tuple in D is 1=N, i.e., all tuples are equally likely.

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

3. **Cluster sample:** If the tuples in D are grouped into M mutually disjoint “clusters", then a SRS of m clusters can be obtained, where m < M. A reduced data representation can be obtained by applying, say, SRSWOR to the pages, resulting in a cluster sample of the tuples.

4. **Stratified sample:** If D is divided into mutually disjoint parts called “strata", a stratified sample of D is generated by obtaining a SRS at each stratum. This helps to ensure a representative sample, especially when the data are skewed. For example, a stratified sample may be obtained from customer data, where stratum is created for each customer age group.

Figure : Sampling can be used for data reduction.



Major Issues in Data Warehousing and Mining

- Mining methodology and user interaction
  - Mining different kinds of knowledge in databases
  - Interactive mining of knowledge at multiple levels of abstraction
  - Incorporation of background knowledge
  - Data mining query languages and ad-hoc data mining
  - Expression and visualization of data mining results
  - Handling noise and incomplete data
  - Pattern evaluation: the interestingness problem
- Performance and scalability
  - Efficiency and scalability of data mining algorithms
  - Parallel, distributed and incremental mining methods
- Issues relating to the diversity of data types
  - Handling relational and complex types of data
  - Mining information from heterogeneous databases and global information systems (WWW)

- Issues related to applications and social impacts
  - Application of discovered knowledge
    - Domain-specific data mining tools

Data Mining Application.

- Web page analysis: from web page classification, clustering to PageRank & HITS algorithms
- Collaborative analysis & recommender systems
- Basket data analysis to targeted marketing
- Biological and medical data analysis: classification, cluster analysis (microarray data analysis), biological sequence analysis, biological network analysis
- Data mining and software engineering (e.g., IEEE Computer, Aug. 2009 issue)
- From major dedicated data mining systems/tools (e.g., SAS, MS SQL-Server Analysis Manager, Oracle Data Mining Tools) to invisible data mining

QUESTIONS

PART - A

1. Define data mining.
2. Define data warehouse.
3. Why preprocessing of data is required?
4. Explain Descriptive and Predictive data mining task.
5. What is meant bry Interestingness measure.
6. Explain the smoothing techniques.
7. Explain data transformation in detail.
8. Explain normalization in detail.
9. Explain data reduction.
10. Explain parametric methods and non-parametric methods of reduction.

PART - B

1. Explain the various datamining issues.
2. Explain the data mining functionalities.
3. Explain datamining primitives.

4. What is data mining? Explain the steps in Knowledge Discovery.
5. Explain the different types of data repositories on which mining can be performed.
6. Explain with appropriate examples, the data mining functionalities.
7. What are the important issues involved in data mining system?
8. Explain in detail i) data cleaning, ii) data integration and iii) data transformation
9. How data reduction can be done in a warehouse?
10. How can we integrate the Data Mining System with a Data Warehouse.

TEXT/REFERENCE BOOKS

1. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", 2nd Edition, Elsevier 2007.
2. Alex Berson and Stephen J. Smith, "Data Warehousing, Data Mining & OLAP", Tata McGrew Hill, 2007.
3. www.tutorialspoint.com/dwh/dwh.overview.htm
4. http://study.com/academy/lesson/data-warehousing-and-data-minig-information-for-business-intelligence.html
5. http://www.dei.unpd_it/-capt/SVMATERIALE/DWDM0405.pdf
6. Data mining Concepts and Techniques, Book by Jewei Han and Kamber.

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT-III Data Mining and Warehousing – SIT1301

ASSOCIATION RULE MINING

3.1 Mining Frequent Patterns
3.2 Associations and Correlations
3.3 Mining Methods
3.4 Finding Frequent Item set using Candidate Generation
3.5 Generating Association Rules from Frequent Item sets
3.6 Mining Frequent Item set without Candidate Generation
3.7 Mining various kinds of association rules
3.8 Mining Multi-Level Association Rule
3.9 Mining Multi-Dimensional Association Rule
3.10 Mining Correlation analysis
3.11 Constraint based association mining.

3.1 **Frequent patterns** are patterns (e.g., itemsets, subsequences, or substructures) that appear frequently in a data set. For example, a set of items, such as milk and bread, that appear frequently together in a transaction data set is a ***frequent itemset***.

A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs fre- quently in a shopping history database, is a (*frequent*) ***sequential pattern***.

A *substructure* can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a **(*frequent*) *structured pattern***.

Association Mining

- Association rule mining:
  - Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.

- Applications:
  - Basket data analysis, cross-marketing, catalog design, loss-leader analysis, clustering, classification, etc.
- Examples.
  - Rule form: ―Body ® Head [support, confidence]‖.
  - buys(x, ―diapers‖) ® buys(x, ―beers‖) [0.5%, 60%]
  - major(x, ―CS‖) ^ takes(x, ―DB‖) ® grade(x, ―A‖) [1%, 75%]

3.2 Association and Correlations

Association Rule: Basic Concepts

- Given: (1) database of transactions, (2) each transaction is a list of items (purchased by a customer in a visit)
- Find: all rules that correlate the presence of one set of items with that of another set of items
  - E.g., 98% of people who purchase tires and auto accessories also get automotive services done
  - Applications
    - * $\Rightarrow$ *Maintenance Agreement* (What the store should do to boost Maintenance Agreement sales)
    - *Home Electronics* $\Rightarrow$ * (What other products should the store stocks up?)
    - Attached mailing in direct marketing
    - Detecting - ping-pong‖ing of patients, faulty - collisions‖

Rule Measures: Support and Confidence

- Find all the rules *X & Y* $\Rightarrow$ *Z* with minimum confidence and support
  - support, *s*, probability that a transaction contains {X 4 Y 4 Z}
  - confidence, *c*, conditional probability that a transaction having {X 4 Y} also contains *Z*

Let minimum support 50%, and minimum confidence 50%, we have

- $A \Rightarrow C$ (50%, 66.6%)
- $C \Rightarrow A$ (50%, 100%)

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Association Rule Mining: A Road Map

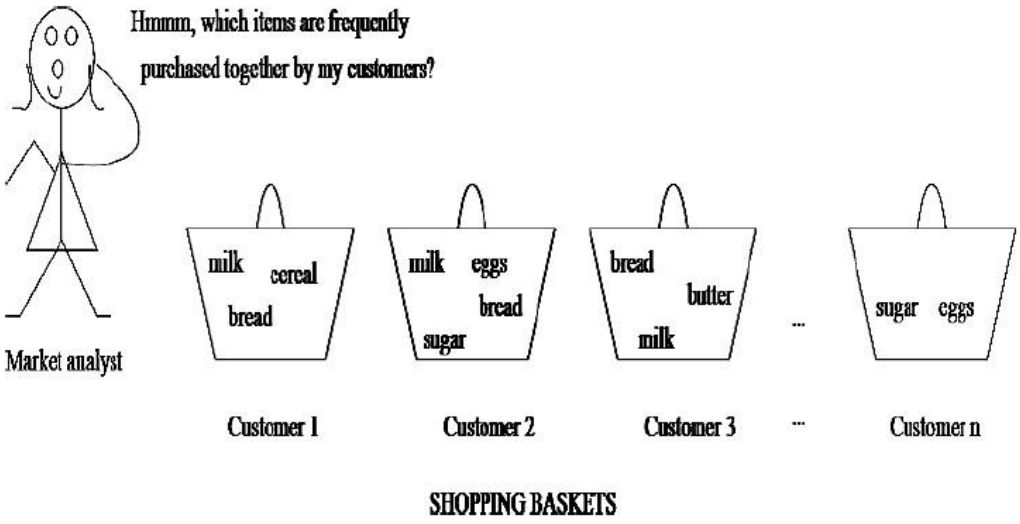
- Boolean vs. quantitative associations (Based on the types of values handled)
  - buys(x, ―SQLServer‖) ^ buys(x, ―DMBook‖) ® buys(x, ―DBMiner‖) [0.2%, 60%]
  - age(x, ―30..39‖) ^ income(x, ―42..48K‖) ® buys(x, ―PC‖) [1%, 75%]
- Single dimension vs. multiple dimensional associations (see ex. Above)
- Single level vs. multiple-level analysis
  - What brands of beers are associated with what brands of diapers?
- Various extensions
  - Correlation, causality analysis
- Association does not necessarily imply correlation or causality
  - Maxpatterns and closed itemsets
  - Constraints enforced
- E.g., small sales (sum < 100) trigger big buys (sum > 1,000)?

Market – Basket analysis

A market basket is a collection of items purchased by a customer in a single transaction, which is a well-defined business activity. For example, a customer's visits to a grocery store or an online purchase from a virtual store on the Web are typical customer transactions. Retailers accumulate huge collections of transactions by recording business activities over time. One

common analysis run against a transactions database is to find sets of items, or *itemsets*, that appear together in many transactions. A business can use knowledge of these patterns to improve the Placement of these items in the store or the layout of mail- order catalog page and Web pages. An itemset containing *i* items is called an *i- itemset*. The percentage of transactions that contain an itemset is called the itemset's *support*. For an itemset to be interesting, its support must be higher than a user-specified minimum. Such itemsets are said to be frequent.

Figure : Market basket analysis



Computer ⇒ financial_management_ software [support = 2%, confidence = 60%]

Rule support and confidence are two measures of rule interestingness. They respectively reflect the usefulness and certainty of discovered rules. A support of 2% for association Rule means that 2% of all the transactions under analysis show that computer and financial management software are purchased together. A confidence of 60% means that 60% of the customers who purchased a computer also bought the software. Typically, association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold.

3.3 Mining Methods

- Mining Frequent Pattern without candidate generation
- Mining Frequent Pattern without candidate generation

3.4 Mining Frequent Patterns with candidate Generation

The method that mines the complete set of frequent itemsets with candidate generation.

Apriori property & The Apriori Algorithm. Apriori property

- All nonempty subsets of a frequent item set most also be frequent.
  - An item set I does not satisfy the minimum support threshold, min-sup, then I is not frequent, i.e., support(I) < min-sup
  - If an item A is added to the item set I then the resulting item set (I U A) can not occur more frequently than I.
- Monotonic functions are functions that move in only one direction.
- This property is called anti-monotonic.
- If a set can not pass a test, all its supersets will fail the same test as well.
- This property is monotonic in failing the test.

The Apriori Algorithm

- Join Step: Ck is generated by joining Lk-1with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset

Method

1) L_1 = find_frequent_1 itemsets(D);
2) for (k = 2; $L_{k-1} \neq \phi$; k++) {
3) C_k = apriori_gen (L_{k-1}, min_sup);
4) For each transaction t $\in$ D { // scan D for counts
5) C_t = subset (C_k, t); // get the subsets of t that are candidates
6) for each candidate c $\in$ C_t
7) c.count++;
8) }
9) $L_k = \{c \in C_k | c.count \geq min\_sup\}$
10) }
11) return $L = U_k L_L$;

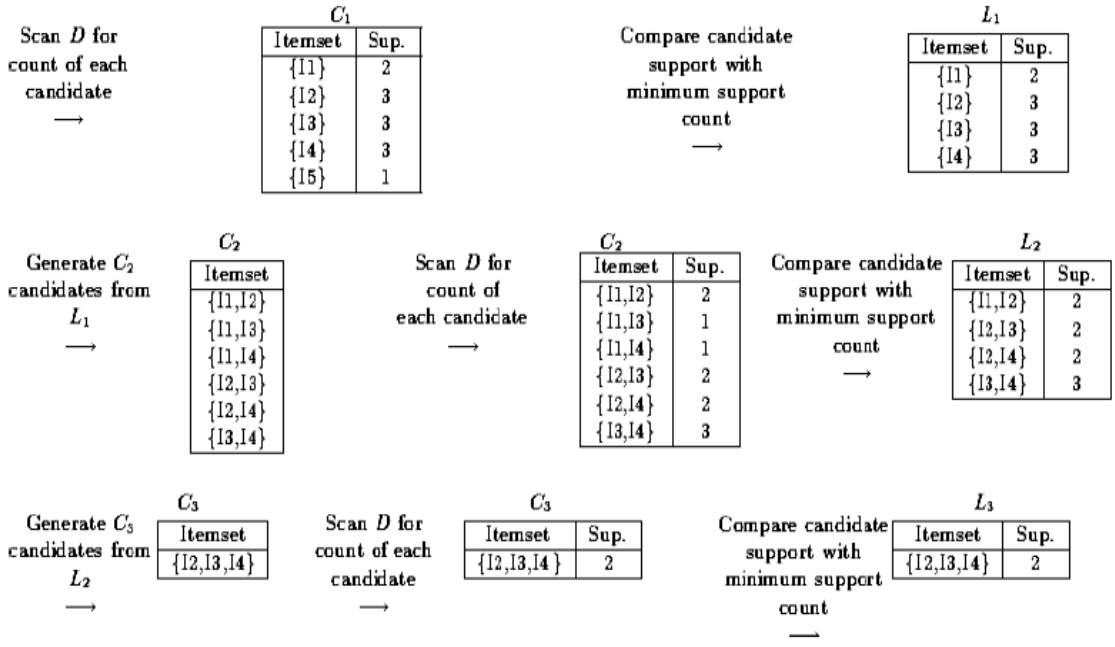
Procedure a priori_gen (L_{k-1}: frequent (k=t) itemsets; min_sup; minimum support)

1) for each itemset $l_1 \in L_{k-1}$
2) for each itemset $l_2 \in L_{k-1}$
3) If ($l_1[1] = l_2[1]$) ^ ($l_1[2] = l_2[2]$) ^…^ ($l_1[k-2] = l_2[k-2]$) ^ ($l_1[k-1] < l_2[k-1]$) that {
4) c = l_1 x l_2; // join step: generate candidates
5) if has_infrequent_subset (c, L_{k-1}) then
6) Delete c, // prune step: remove unfruitful candidate
7) else add c to C_k;
8) }
9) Return C_k;

Procedure has_infrequent_subset (c: candidate k itemsetm; L_{k-1}: frequent (k-1) itemsets); // use prior knowledge

1) for each (k-1) subset s of c
2) if s $\notin L_{k-1}$ then
3) return TRUE;
4) return FALSE;

Example



3.6 Mining Frequent Item set without Candidate Generation

Frequent Pattern Growth Tree Algorithm

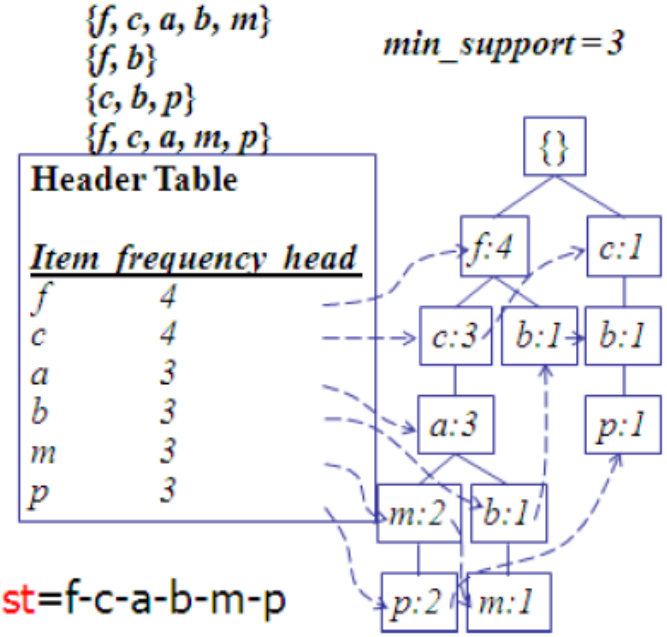
It grows long patterns from short ones using local frequent items

- "abc" is a frequent pattern
- Get all transactions having "abc": DB|abc
- "d" is a local frequent item in DB | abc € abcd is a frequent pattern

Construct FP-tree from a Transaction Database

TID	Items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o, w}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

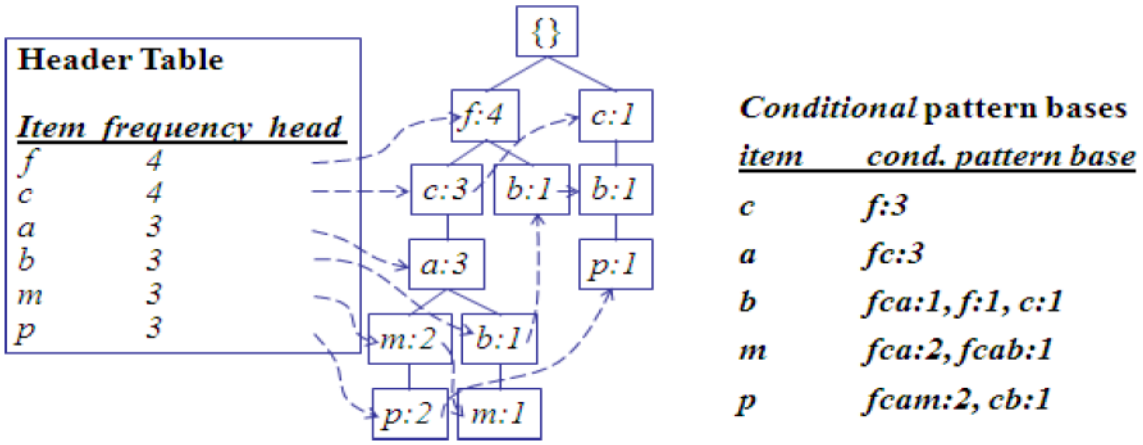
1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list
3. Scan DB again, construct FP-tree



F-list=f-c-a-b-m-p

Find Patterns Having P from P-conditional Database

- Starting at the frequent item header table in the FP-tree
- Traverse the FP-tree by following the link of each frequent item P
- Accumulate all of transformed prefix paths of items *p* to form *P*'s conditional pattern base



Benefits of the FP-tree Structure

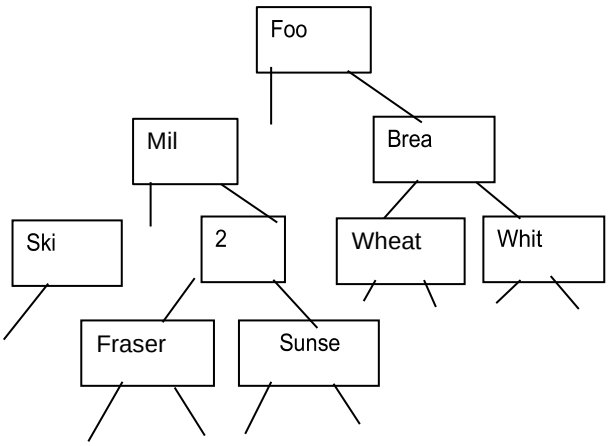
- Completeness:
  - never breaks a long pattern of any transaction
  - preserves complete information for frequent pattern mining
- Compactness
  - reduce irrelevant information—infrequent items are gone
  - frequency descending ordering: more frequent items are more likely to be shared
  - never be larger than the original database (if not count node-links and counts)
  - Example: For Connect-4 DB, compression ratio could be over 100

3.7 Mining various kinds of Association Rule

- Mining Multi-level association rule
- **Mining Multi dimensional Association Rule**

Mining multilevel association rules from transactional databases.

- Items often form hierarchy.
- Items at the lower level are expected to have lower support.
- Rules regarding itemsets atappropriate levels could be quite useful.
- Transaction database can be encoded based on dimensions and levels
- We can explore shared multi-level mining



TID	Items
T1	{111, 121, 211, 221}
T2	{111, 211, 222, 323}
T3	{112, 122, 221, 411}
T4	{111, 121}
T5	{111, 122, 211, 221, 413}

3.8 Mining Multi-Level Associations

- A top_down, progressive deepening approach:
  - First find high-level strong rules:
    milk ® bread [20%, 60%].
  - Then find their lower-level ―weaker‖ rules:
    2% milk ® wheat bread [6%, 50%].
- Variations at mining multiple-level association rules.
  - Level-crossed association rules:
    2% *milk ® Wonder wheat bread*
  - Association rules with multiple, alternative hierarchies**:**
    2% *milk ® Wonder bread*

Multi-level Association: Uniform Support vs. Reduced Support

- Uniform Support: the same minimum support for all levels
  - + One minimum support threshold. No need to examine itemsets containing any item whose ancestors do not have minimum support.
  - Lower level items do not occur as frequently. If support threshold
- too high ⇒ miss low level associations
- too low ⇒ generate too many high level associations
- Reduced Support: reduced minimum support at lower levels
  - There are 4 search strategies:
    - Level-by-level independent
    - Level-cross filtering by k-itemset
    - Level-cross filtering by single item
    - Controlled level-cross filtering by single item

Multi-level Association: Redundancy Filtering

- Some rules may be redundant due to ―ancestor‖ relationships between items.
- Example
  - milk ⇒ wheat bread [support = 8%, confidence = 70%]
  - 2% milk ⇒ wheat bread [support = 2%, confidence = 72%]
- We say the first rule is an ancestor of the second rule.
- A rule is redundant if its support is close to the ―expected‖ value, based on the rule's ancestor

Multi-Level Mining: Progressive Deepening

- A top-down, progressive deepening approach:
  - First mine high-level frequent items: milk (15%), bread (10%)
  - Then mine their lower-level ―weaker‖ frequent itemsets: 2% milk (5%), wheat bread (4%)
- Different min_support threshold across multi-levels lead to different algorithms:
  - If adopting the same *min_support* across multi-levels then toss *t* if any of *t*'s ancestors is infrequent.

– If adopting reduced *min_support* at lower levels then examine only those descendents whose ancestor's support is frequent/non-negligible.

3.9 Mining Multidimensional Association mining

Mining our *AllElectronics* database, we may discover the Boolean association rule *buys(X*, "*digital camera*"*) ⇒buys(X*, "*HP printer*"*).* (7.6)

Following the terminology used in multidimensional databases,

single- dimensional or **intradimensional association rule** because it contains a single distinct predicate (e.g., *buys*) with multiple occurrences (i.e., the predicate occurs more than once within the rule). Such rules are commonly mined from transactional data.

Considering each database attribute or warehouse dimension as a predicate, we can therefore mine association rules containing *multiple* predicates such as

age(X, "20 . . . 29"*)* A*occupation(X*, "*student*"*) ⇒buys(X*, "*laptop*"*).*

Association rules that involve two or more dimensions or predicates can be referred to as **multidimensional association rules**. Rule contains three predicates (*age*, *occupation*, and *buys*), each of which occurs *only once* in the rule. Hence, we say that it has **no repeated predicates**.

Multidimensional association rules with no repeated predicates are called **interdimensional association rules**. We can also mine multidimensional association rules with repeated predicates, which contain multiple occurrences of some predicates. These rules are called **hybrid-dimensional association rules**.

An example of such a rule is the following, where the predicate *buys* is repeated:

age(X, "20 . . . 29"*)* A*buys(X*, "*laptop*"*) ⇒buys(X*, "*HP printer*"*).*

Database attributes can be nominal or quantitative. The values of **nominal** (or categorical) attributes are "names of things." Nominal attributes have a finite number of possible values, with no ordering among the values (e.g., *occupation, brand, color*).

Quantitative attributes are numeric and have an implicit ordering among values (e.g., *age, income, price*). Techniques for mining multidimensional association rules can be categorized into two basic approaches regarding the treatment of quantitative attributes.

In the first approach, *quantitative attributes are discretized using predefined concept hierarchies*. This discretization occurs before mining. For instance, a concept hierarchy for *income* may be used to replace the original numeric values of this attribute by interval labels such as "0..20K," "21K..30K," "31K..40K," and so on.

Here, discretization is *static* and predetermined. Chapter 3 on data preprocessing gave several techniques for discretizing numeric attributes. The discretized numeric attributes, with their interval labels, can then be treated as nominal attributes (where each interval is considered a category).

Mining Quantitative Association Rules

- Determine the number of partitions for each quantitative attribute
- Map values/ranges to consecutive integer values such that the order is preserved
- Find the support of each value of the attributes, and combine when support is less than MaxSup. Find frequent itemsets, whose support is larger than MinSup
- Use frequent set to generate association rules
- Pruning out uninteresting rules

Partial Completeness

- R : rules obtained before partition
- R': rules obtained after partition
- Partial Completeness measures the maximum distance between a rule in R and its closest generalization in R'
- $\hat{X}$ is a generalization of itemset X: if

$$\forall x \in \text{attributes (X)} [< x, l, u > \in X \wedge x, l', u' > \in \hat{X} \Rightarrow l' \leq l \leq u \leq u']$$

- The distance is defined by the ratio of support

K-Complete

- *C* : the set of frequent itemsets
- For any K ≥ 1, *P* is K-complete w.r.t *C* if:
  1. P C
  2. For any itemset *X* (or its subset) in *C*, there exists a generalization whose support is no more than *K* times that of *X* (or its subset)
- The smaller K is, the less the information lost

3.10 Correlation Analysis

- Interest (correlation, lift)
  - taking both P(A) and P(B) in consideration
  - P(A^B)=P(B)*P(A), if A and B are independent events
  - A and B negatively correlated, if the value is less than 1; otherwise A and B positively correlated

X2 Correlation

- X2 measures correlation between categorical attributes

X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

Itemset	Support	Interest
X,Y	25%	2
X,Z	37.50%	0.9
Y,Z	12.50%	0.57

$$X =^2 = \sum \frac{(observe\_expe\ cted)^2}{\exp ected}$$

	game	not game	sum(row)
video	4000(4500)	3500(3000)	7500
not video	2000(1500)	500 (1000)	2500
sum(col.)	6000	4000	10000

- expected(i,j) = count(row i) * count(column j) / N
- X2 = (4000 - 4500)2 / 4500 - (3500 - 3000)2 / 3000 - (2000 - 1500)2 / 1500 - (500 -1000)2 / 1000 = 555.6
- X2 > 1 and observed value of (game, video) < expected value, there is a negative correlation

Numeric correlation

- Correlation concept in statistics
  - Used to study the relationship existing between 2 or more numeric variables
  - A correlation is a measure of the linear relationship between variables Ex: number of hours spent studying in a class with grade received
  - Outcomes:
    - → positively related
    - → Not related
    - → negatively related
  - Statistical relationships
    - Covariance
    - Correlation coefficient

3.11 Constraint-Based Association Mining

- Interactive, exploratory mining giga-bytes of data?
  - Could it be real? — Making good use of constraints!
- What kinds of constraints can be used in mining?
  - Knowledge type constraint: classification, association, etc.
  - Data constraint: SQL-like queries
- Find product pairs sold together in Vancouver in Dec.'98.
  - Dimension/level constraints:
- in relevance to region, price, brand, customer category.
  - Rule constraints
- small sales (price < $10) triggers big sales (sum > $200).
  - Interestingness constraints:
- strong rules (min_support ≥ 3%, min_confidence ≥ 60%).

Rule Constraints in Association Mining

- Two kind of rule constraints:
  - Rule form constraints: meta-rule guided mining.
- P(x, y) ^ Q(x, w) ® takes(x, ―database systems‖).
  - Rule (content) constraint: constraint-based query optimization (Ng, et al., SIGMOD'98).
- sum(LHS) < 100 ^ min(LHS) > 20 ^ count(LHS) > 3 ^ sum(RHS) > 1000
- 1-variable vs. 2-variable constraints (Lakshmanan, et al. SIGMOD'99):
  - 1-var: A constraint confining only one side (L/R) of the rule, e.g., as shown above.
  - 2-var: A constraint confining both sides (L and R).
- sum(LHS) < min(RHS) ^ max(RHS) < 5* sum(LHS)

Constrain-Based Association Query

- Database: (1) trans (TID, Itemset), (2) itemInfo (Item, Type, Price)
- A constrained asso. query (CAQ) is in the form of {(*S1, S2*)|*C* },
  - where C is a set of constraints on S1, S2 including frequency constraint
- A classification of (single-variable) constraints:
  - Class constraint: : S ⊂ A. *e.g. S ⊂Item*
  - Domain constraint:
    - *S θ v, θ∈{ =, ≠, <, ≤, >, ≥ }. e.g. S.Price < 100*
    - *v θ S, θ is ∈ or ∉ e.g. snacks ∉ S. Typev*
    - *V θ S, or S θ V, θ∈ {⊆, ⊂, ⊄, =, ≠)*
    - *- e.g. {snacks, sodas} ⊆ S.Type*
  - Aggregation constraint: *agg(S) θ v*, where *agg* is in {*min, max, sum, count, avg*}, and *θ∈{ =, ≠, <, ≤, >, ≥ }.*
    - *e.g. count(S1.Type) = 1 , avg(S2.Price) < 100*

Constrained Association Query Optimization Problem

- Given a CAQ = { (*S1, S2*) | *C* }, the algorithm should be :
  - sound: It only finds frequent sets that satisfy the given constraints C
  - complete: All frequent sets satisfy the given constraints C are found

- A naïve solution:
  - Apply Apriori for finding all frequent sets, and then to test them for constraint satisfaction one by one.
- Our approach:
  - Comprehensive analysis of the properties of constraints and try to push them as deeply as possible inside the frequent set computation.

Categories of Constraints.

1. Anti-monotone and Monotone Constraints

- constraint Ca is anti-monotone iff. for any pattern S not satisfying Ca, none of the super-patterns of S can satisfy Ca
- A constraint Cm is monotone iff. for any pattern S satisfying Cm, every super-pattern of S also satisfies it

2. Succinct Constraint

- A subset of item Is is a succinct set, if it can be expressed as σp(I) for some selection predicate p, where σ is a selection operator
- SP⊆2I is a succinct power set, if there is a fixed number of succinct set I1, …, Ik ⊆I,s.t. SP can be expressed in terms of the strict power sets of I1, …, Ik using union and minus
- A constraint Cs is succinct provided SATCs(I) is a succinct power set

3. Convertible Constraint

- Suppose all items in patterns are listed in a total order R
- A constraint C is convertible anti-monotone iff a pattern S satisfying the constraint implies that each suffix of S w.r.t. R also satisfies C
- A constraint C is convertible monotone iff a pattern S satisfying the constraint implies that each pattern of which S is a suffix w.r.t. R also satisfies C

Property of Constraints: Anti-Monotone

- Anti-monotonicity: *If a set S violates the constraint, any superset of S violates the constraint.*
- Examples:
  - *sum(S.Price)* ≤*v* is anti-monotone
  - *sum(S.Price)* ≥*v* is not anti-monotone
  - *sum(S.Price)* = *v* is partly anti-monotone
- Application:
  - Push ―*sum(S.price)* ≤ 1000‖ deeply into iterative frequent set computation.

Example of Convertible Constraints: Avg(S) θ V

- Let R be the value descending order over the set of items
  - E.g. I={9, 8, 6, 4, 3, 1}
- Avg(S) ≥ v is convertible monotone w.r.t. R
  - If S is a suffix of S1, avg(S1) □ avg(S)
    - {8, 4, 3} is a suffix of {9, 8, 4, 3}
    - avg({9, 8, 4, 3})=6 ≥ avg({8, 4, 3})=5
  - If S satisfies avg(S) ≥v, so does S1
    - {8, 4, 3} satisfies constraint avg(S) ≥ 4, so does {9, 8, 4, 3}

Property of Constraints: Succinctness

- Succinctness:
  - For any set S1 and S2 satisfying C, S1 □ S2 satisfies C
  - Given A1 is the sets of size 1 satisfying C, then any set S satisfying C are based on A1 , i.e., it contains a subset belongs to A1 ,
- Example :
  - *sum(S.Price)* ≥*v* is not succinct
  - *min(S.Price)* ≤*v* is succinct

- Optimization:
  - If C is succinct, then C is pre-counting prunable. The satisfaction of the constraint alone is not affected by the iterative support counting.
    - ed based on the training set
    - Unsupervised learning (clustering)
    - The class labels of training data is unknown
    - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data.

QUESTIONS

PART A

1. What is association rule mining? Explain with example?
2. How association rules are mined in large databases?
3. Design a method that mines the complete set of frequent item sets without candidate generation?
4. Explain iceberg queries with example?
5. Differentiate mining quantitative association rules and distance based association rules?
6. Explain how to improve the efficiency of apriori algorithm?
7. List out different kinds of constraint based association mining?
8. How to transform from association analysis to correlation analysis?

PART B

1. Explain market basket analysis with an motivating example for association rule mining?
2. Define association rule mining and explain how the apriori algorithm works with suitable examples
3. Explain mining multi level association rules from transactional database?
4. Explain FP-Growth Method: Mining Frequent Itemsets without Candidate Generation?
5. What is the principle of multi-level association? How do you mine data from relational databases and data warehouses for multidimensional association rules?

TEXT/REFERENCE BOOKS

1. Jiawei Han and Micheline Kamber, “Data Mining Concepts and Techniques”, 2nd Edition, Elsevier 2007.
2. Alex Berson and Stephen J. Smith, “Data Warehousing, Data Mining & OLAP”, Tata McGrew Hill, 2007.

3. www.tutorialspoint.com/dwh/dwh.overview.htm
4. http://study.com/academy/lesson/data-warehousing-and-data-minig-information-for-business-intelligence.html
5. http://www.dei.unpd_it/-capt/SVMATERIALE/DWDM0405.pdf
6. Data mining Concepts and Techniques, Book by Jewei Han and Kamber.

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT-IV Data Mining and Warehousing – SIT1301

CLASSIFICATION AND PREDICTION

4.1 Classification and Prediction

– predicts categorical class labels
– classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
– models continuous-valued functions, i.e., predicts unknown or missing values

- **Typical applications**

– Credit approval
– Target marketing
– Medical diagnosis
– Fraud detection

Classification - A Two-Step Process

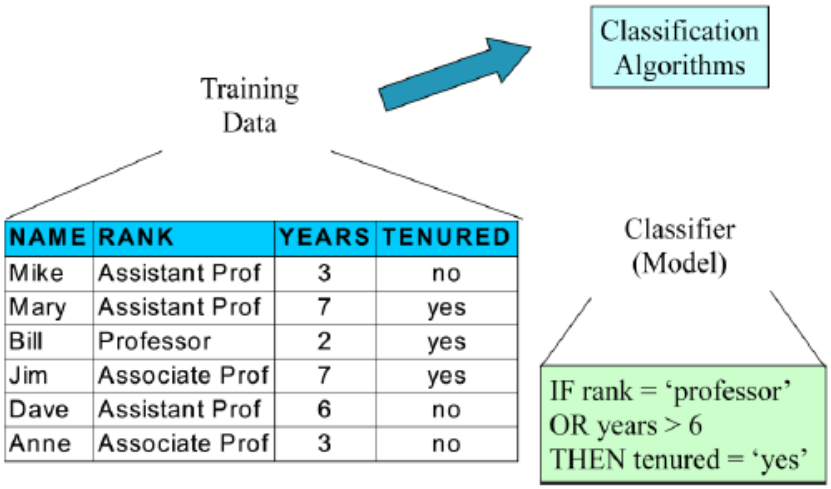
- Model construction: describing a set of predetermined classes
  – Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  – The set of tuples used for model construction: training set

  - The model is represented as classification rules, decision trees, or mathematical formulae

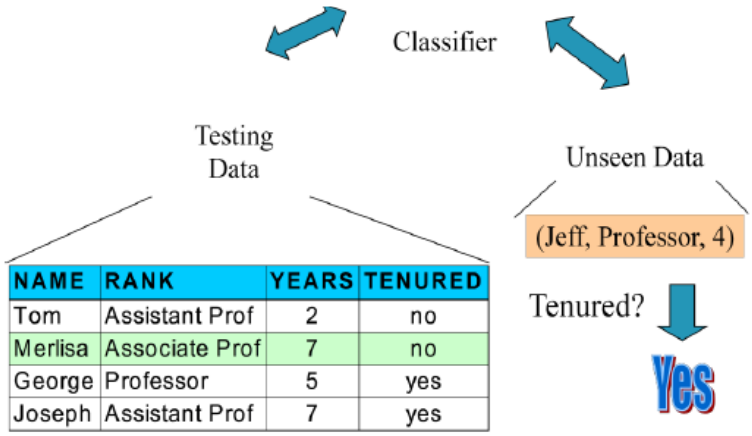
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model

- The known label of test sample is compared with the classified result from the model
- Accuracy rate is the percentage of test set samples that are correctly classified by the model
- Test set is independent of training set, otherwise over-fitting will occur

Process (1): Model Construction



Process (2): Using the Model in Prediction



Supervised vs. Unsupervised Learning

- Supervised learning(classification)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set

- Unsupervised learning(clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

4.2 Issues Regarding Classification and Prediction

This describes issues regarding preprocessing the data of classification and prediction. Criteria for the comparison and evaluation of classification methods are also described.

Preparing the Data for Classification and Prediction

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

Data Cleaning: This refers to the preprocessing of data in order to remove or reduce noise (by applying smoothing techniques) and the treatment of missing values (e.g., by replacing a missing value with the most commonly occurring value for that attribute, 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.

Relevance Analysis: Many of the attributes in the data may be irrelevant to the classification or prediction task. For example, data recording the day of the week on which a bank loan application was filed is unlikely to be relevant to the success of the application. Furthermore, other attributes may be redundant. Hence, relevance analysis may be performed on the data with the aim of removing any irrelevant or redundant attributes from the learning process. In machine learning, this step is known as feature selection. Including such attributes may otherwise slow down, and possibly mislead, the learning step.

Ideally, the time spent on relevance analysis, when added to the time spent on learning from the resulting "reduced" feature subset should be less than the time that would have been spent on learning from the original set of features. Hence, such analysis can help improve classification efficiency and scalability.

Data Transformation: The data can be generalized to higher – level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous – valued attributes. For example, numeric values for the attribute income may be generalized to discrete ranges such as low, medium, and high. Similarly, nominal – valued attributes like street, can be generalized to higher – level concepts, like city. Since generalization compresses the original training data, fewer input / output operations may be involved during learning.

The data may also be normalized, 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 – 1.0 to 1.0, or 0.0 to 1.0. In methods that use distance measurements, for example, this would prevent attributes with initially large ranges (like, say, income) from outweighing attributes with initially smaller ranges (such as binary attributes).

Comparing Classification Methods

Classification and prediction methods can be compared and evaluated according to the following criteria:

Predictive Accuracy: This refers to the ability of the model to correctly predict the class label of new or previously unseen data.

Speed: This refers to the computation costs involved in generating and using the model.

Robustness: This is the ability of the model to make correct predictions given noisy data or data with missing values.

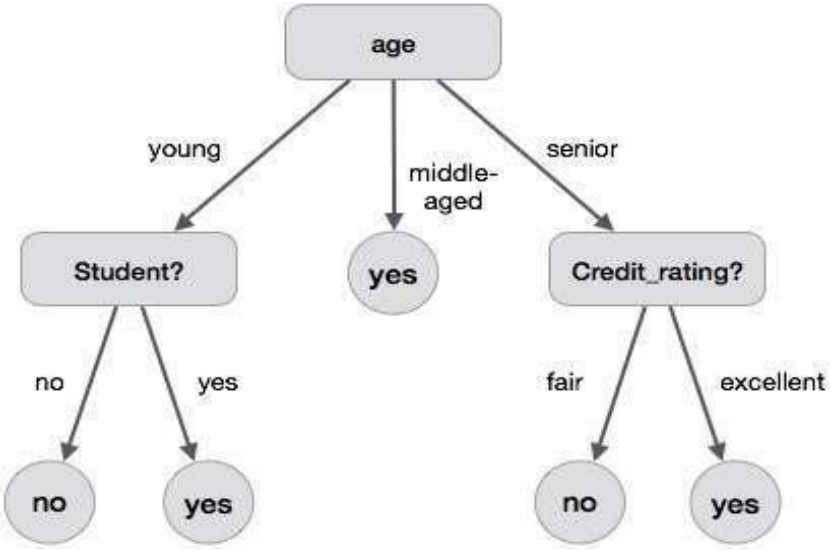
Scalability: This refers to the ability to construct the model efficiently given large amount of data.

Interpretability: This refers to the level of understanding and insight that is provided by the model.

4.3 Classification by Decision Tree Induction

A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node.

The following decision tree is for the concept buys_computer that indicates whether a customer at a company is likely to buy a computer or not. Each internal node represents a test on an attribute. Each leaf node represents a class.



The benefits of having a decision tree are as follows :

- It does not require any domain knowledge.
- It is easy to comprehend.
- The learning and classification steps of a decision tree are simple and fast.

The expected information needed to classify a tuple in *D* is given by

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

Then, for each attribute A,

$$\text{Info}_A(D) = \sum_{i=1}^{V} \frac{||D_i|}{|D|} \times \text{Info}(D_j),$$

where Dj / D is the weight of the jth partition.

Info A (D) is the expected information required to classify a tuple from D based on the partitioning by A. The smaller the expected information, the greater the purity of the partitions.

Information gain is defined as the difference between the original information requirement (i.e., based on just the proportion of classes) and the new requirement (i.e., obtained after partitioning on A). That is,

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

//Generating a decision tree form training tuples of data partition D

Algorithm: Generate_decision_tree 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 that the data tuples into individual classes. This criterion includes a splitting_attribute and either a splitting point or splitting subset.

Output

A Decision Tree

Method

```
create a node N;

if tuples in D are all of the same class, C then return N
 as leaf node labeled with class C;

 if attribute_list is empty then return N as
    leaf node withlabeled
          with majority class in D;|| majority voting

 apply attribute_selection_method(D, attribute_list) to find
 the best splitting_criterion;
 label node N with splitting_criterion;

 if splitting_attribute is discrete-valued and
    multiway splitsallowedthen          // no restricted to binarytrees
```

```
    attribute_list = splitting attribute; // remove splittingattribute for each
    outcome j of splitting criterion

      // partition the tuples and grow subtrees for each partition
      let Dj be the set of data tuples in D satisfying outcome j; // a partition

      if Dj is empty then
            attach a leaf labeled with the majority class in
            D to node N;
      else
           attach the node returned by Generate decision
           tree(Dj, attribute list) to nodeN;
    end for
return N;
```

Tree Pruning

Tree pruning is performed in order to remove anomalies in the training data due to noise or outliers. The pruned trees are smaller and less complex.

Tree Pruning Approaches

Here is the Tree Pruning Approaches listed below −

- **Pre-pruning** − The tree is pruned by halting its construction early.
- **Post-pruning** - This approach removes a sub-tree from a fully grown tree.

Cost Complexity

The cost complexity is measured by the following two parameters –

- Number of leaves in the tree, and
- Error rate of the tree.

Example

Class-labeled Training Tuples from the all Electronics Customer Database

RID	Age	Income	Student	Credit-rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

$$\text{Info}(D) = \frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits.}$$

$$\text{Info}_{age}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right)$$

$$+ \frac{4}{14} \times \left(-\frac{4}{4}\log_2\frac{4}{4}\right)$$

$$+ \frac{5}{14} \times \left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right)$$

= 0.694 bits.

Gain(age) = Info(D) – Info_{age}(D) = 0.940 – 0.694 = 0.246 bits.

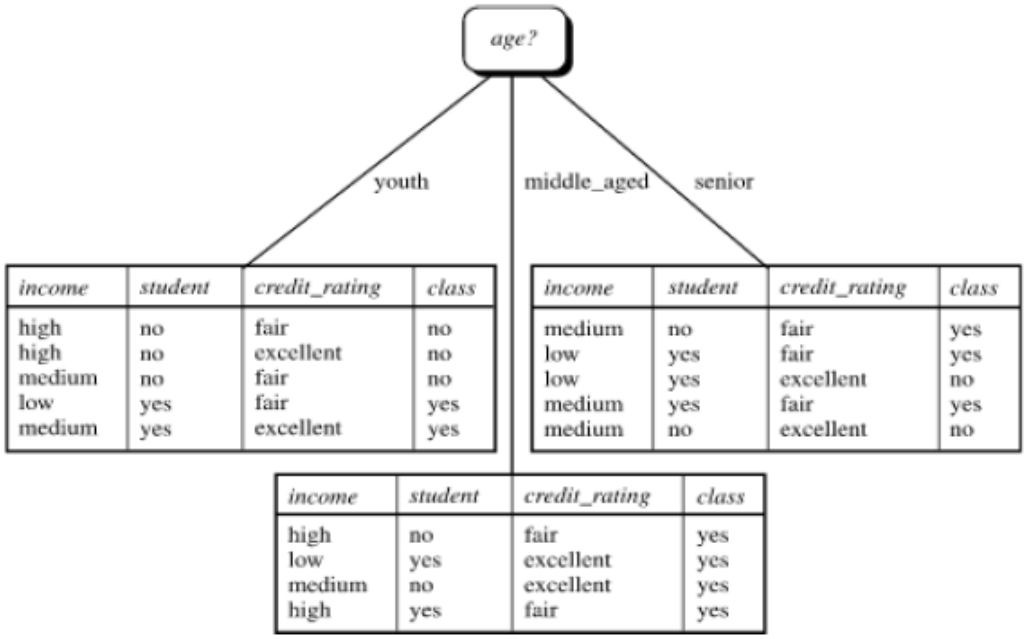
Similarly,

Gain(income)=0.029

Gain(Credit_rating)=0.1

51 Gain(Student)=0.048

Therefore out of all Gain values obtained, the attribute Age has gained a higher value, and hence it proves itself to be the best splitting attribute. Hence, the decision tree would look like the one given below:



income	*student*	*credit_rating*	*class*
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	*student*	*credit_rating*	*class*
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

The branch middle-aged is classified as pure class (YES). Repeat the same process for the youth and senior branch until all the branches of decision tree turned to pure class.

4.4 Bayesian Classification

Bayesian framework assumes that we always have a prior distribution for everything.

- The prior may be very vague.
- When we see some data, we combine our prior distribution with a likelihood term to get a posterior distribution.
- The likelihood term takes into account how probable the observed data is given the parameters of the model.
  - It favors parameter settings that make the data likely.
  - It fights the prior
  - With enough data the likelihood terms always win.

4.5 Baye's Theorem

P(H|X) is the posterior probability, or a posteriori probability, of H conditioned on X. For example, suppose our world of data tuples is confined to customer described by the attributes age and income, respectively, and that X is a 35-year-old customer will buy a computer. Then P(H|X) reflects the probability that customer X will buy a computer given that we know the customer's age and income.

In contrast, P(H) is the prior probability, or a priori probability, of H. For our example, this is the probability that any given customer will buy a computer, regardless of age, income, or any other information, for that matter.

Bayes' theorem is useful in that it provides a way of calculating the posterior probability P(H|X), from P(H), P(X|H), and P(X).

Bayes, theorem is

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)}$$

Naïve Bayesian Classification

The naïve Bayesian classifies, 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 nu an n-dimensional attribute vector, X = (x_1, x_2, …,x_n), depicting n measurements made on the tuple from n attributes, respectively, A_1, A_2,…A_n.

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

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

Thus. We maximize $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is classed the maximum posterior hypothesis. By Bayes' Theorem (Eq.)

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$$

3. As $P(X)$ is constant for all classes, only $P(X|C_i)P(C_i)$ needs to be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is $P(C_1) = P(C_2) = \ldots = P(C_m)$, and we would therefore maximize $P(X|C_i)$. Otherwise, we maximum $P(X|C_i)P(C_i)$. Note that the class prior probabilities may be estimated by $P(C_i) = |C_{i,D}|/|D|$, where $|C_{i,D}|$ is the number of training tuples of class C_i in D.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute $P(X|C_i)$. To reduce computation in evaluating $P(X|C_i)$, the naïve assumption of class-conditional independence is made. This presumes that the attributes values are conditionally independent of one another, given the class label of the tuple (i.e., that there are no dependence relationships among the attributes). Thus,

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

$$= P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(_n|C_i)$$

We can easily estimate the probabilities $P(x_1|C_i) \times \ldots \times P(x_n|C_i)$ from the training tuples. Recall that there x_k refers to the value of attribute A_k for tuple X. For each attribute, we look at whether the attribute is categorical or continuous-valued. For instance, to compute $P(X|C_i)$, we consider the following:

(a) If A_k is categorical, then $P(x_k|C_i)$ is the number of tuples of class C_i in D having the value x_k for A_k, divided by $|C_{i,D}|$, the number of tuples of class C_i in D.

(b) If A_k continuous-valued, then we need to do a bit more work, but the calculation is pretty straightforward. A continuous-valued attribute is typically assumed to have a Gaussian distribution with a mean μ and standard deviation σ, defined by

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

So that

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

Given database

Class-labeled Training Tuples from the all Electronics Customer Database

RID	Age	Income	Student	Credit-rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Example

Predicting a class label using naïve Bayesian classification. We wish to predict the class label of a tuple using naïve Bayesian classification given the same training data as in Example for decision tree induction. The training data were shown earlier in Table. The data tuples are described by the attributes age, income, student, and credit+ratings. The class label attribute, buys_computer, has two distinct values (namely, (yes, no). Let C_1 correspond to the class buys_ computer = yes and C_2 correspond to buys_computer = no. The tuple we wish to classify is

X = (age = youth, income = medium, student = yes, credit_rating = fair)

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

P(buys_computer = yes) = 9/14 = 0.643
P(buys_computer = no) = 5/14 = 0.357

To compute $P(X|C_i)$, for I = 1, 2, we compute the following conditional probabilities:

P(age = youth | buys_computer = yes)	=	2/9 = 0.222
P(age = youth | buys_computer = no)	=	3/5 = 0.600
P(income = medium | buys_computer = yes)	=	4/9 = 0.444
P(income = medium | buys_computer = no)	=	2/5 = 0.400
P(student = yes | buys_computer = yes)	=	6/9 = 0.667
P(student = yes | buys_computer = no)	=	1/5 = 0.200
P(credit_rating = fair | buys_computer = yes)	=	6/9 = 0.667
P(credit_rating = fair | buys_computer = no)	=	2/5 = 0.400

Using these probabilities, we obtain

P(X|buys_computer = yes) = P(age = youth | buys_computer = yes)
× P(income = medium | buys_computer = yes)
× P(student = yes | buys_computer = yes)
× P(credit_rating = fair | buys_computer = yes)
= 0.222 × 0.444 × 0.667 × 0.667 = 0.044.

Similary,

P(X|buys_computer = no) = 0.600 × 0.400 × 0.200 × 0.400 = 0.019.

To find the class, C_i, that maximize $P(X|C_i)P(C_i)$, we compute

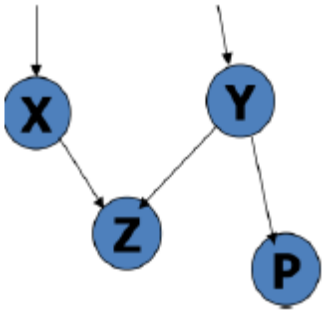
P(X|buys_computer = yes) P(buys_computer = yes) = 0.044 × 0.643 = 0.028
P(X|buys_computer = no) P(buys_computer = no) = 0.019 × 0.357 = 0.007

Therefore, the naïve Bayesian classifier predicts buys_computer = yes typle X.

4.7 Bayesian networks

A Bayesian network, Bayes network, belief network, Bayes(ian) model or probabilistic directed acyclic graphical model is a probabilistic graphical model (a type of statistical model) that represents a set of variables and their conditional dependencies via a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

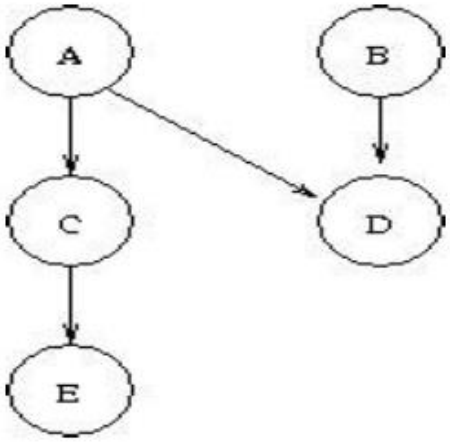
- Bayesian belief network allows a *subset* of the variables conditionally independent
- A graphical model of causal relationships
- Represents dependency among the variables
- Gives a specification of joint probability distribution



- Nodes, random variables
- Links: dependency
- X and Y are the parents of Z, and Y is the parent of P
- No dependency between Z and P
- Has no loops or cycles

Bayesian Net Example

Consider the following Bayesian network:



Thus, the independence expressed in this Bayesian net are that A and B are (absolutely) independent.

C is independent of B given A.

D is independent of C given A and B.

E is independent of A, B, and D given C.

Suppose that the net further records the following probabilities:

Prob(A=T) = 0.3

Prob(B=T) = 0.6

Prob(C=T|A=T) = 0.8

Prob(C=T|A=F) = 0.4

Prob(D=T|A=T,B=T) = 0.7

Prob(D=T|A=T,B=F) = 0.8

Prob(D=T|A=F,B=T) = 0.1

Prob(D=T|A=F,B=F) = 0.2

Prob(E=T|C=T) = 0.7

Prob(E=T|C=F) = 0.2

Some sample computations:

Prob(D=T)

P(D=T) =

P(D=T,A=T,B=T) + P(D=T,A=T,B=F) + P(D=T,A=F,B=T) + P(D=T,A=F,B=F) = P(D=T|A=T,B=T) P(A=T,B=T) + P(D=T|A=T,B=F) P(A=T,B=F) + P(D=T|A=F,B=T) P(A=F,B=T) + P(D=T|A=F,B=F) P(A=F,B=F) =
(since A and B are independent absolutely)

P(D=T|A=T,B=T) P(A=T) P(B=T) + P(D=T|A=T,B=F) P(A=T) P(B=F) + P(D=T|A=F,B=T) P(A=F) P(B=T) + P(D=T|A=F,B=F) P(A=F) P(B=F) =
0.7*0.3*0.6 + 0.8*0.3*0.4 + 0.1*0.7*0.6 + 0.2*0.7*0.4 = 0.32

Prob(A=T|C=T)

P(A=T|C=T) = P(C=T|A=T)P(A=T) / P(C=T).

Now, P(C=T) = P(C=T,A=T) + P(C=T,A=F) = P(C=T|A=T)P(A=T) + P(C=T|A=F)P(A=F) = 0.8*0.3+ 0.4*0.7 = 0.52

So P(C=T|A=T)P(A=T) / P(C=T) = 0.8*0.3/0.52= 0.46.

4.8 Rule Based Classification

Using IF-THEN Rules for Classification

- Represent the knowledge in the form of IF-THEN rules

  R: IF *age* = youth AND *student* = yes THEN *buys_computer* = yes

- Rule antecedent/precondition vs. rule consequent Assessment of a rule: *coverage* and *accuracy*
- ncovers = # of tuples covered byR
- ncorrect = # of tuples correctly classified byR

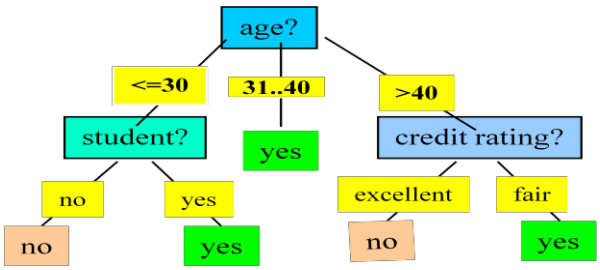
coverage(R) = ncovers /|D| /* D: training data set */

accuracy(R) = ncorrect /ncovers

- If more than one rule is triggered, need conflict resolution
- Size ordering: assign the highest priority to the triggering rules that has the ― toughest‖ requirement (i.e., with the *most attribute test*)
- Class-based ordering: decreasing order of *prevalence or misclassification cost per class*
- Rule-based ordering (decision list): rules are organized into one long priority list, according to some measure of rule quality or by experts

Rule Extraction from a Decision Tree

- Rules are easier to understand than large trees
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction: the leaf holds the class prediction
- Rules are mutually exclusive and exhaustive



- **Example: Rule extraction from our *buys_computer*decision-tree**

  IF *age* = young AND *student=no* THEN *buys_computer = no*

  IF *age* = young AND *student=yes* THEN *buys_computer = yes*

  IF *age*=mid-age THEN *buys_computer =yes*

  IF *age* = old AND *credit_rating = excellent* THEN *buys_computer = yes*

  IF *age* = young AND *credit_rating=fair* THEN *buys_computer = no*

Rule Extraction from the Training Data

- Sequential covering algorithm: Extracts rules directly from training data
- Typical sequential covering algorithms: FOIL, AQ, CN2,RIPPER
- Rules are learned *sequentially*, each for a given class Ci will cover many tuples of Ci but none (or few) of the tuples of other classes
- Steps:
  - Rules are learned one at a time
  - Each time a rule is learned, the tuples covered by the rules are removed
  - The process repeats on the remaining tuples unless *termination condition*, e.g., when no more training examples or when the quality of a rule returned is below a user-specified threshold
- Comp. w. decision-tree induction: learning a set of rule s*imultaneously*

4.9 Classification by Back propagation

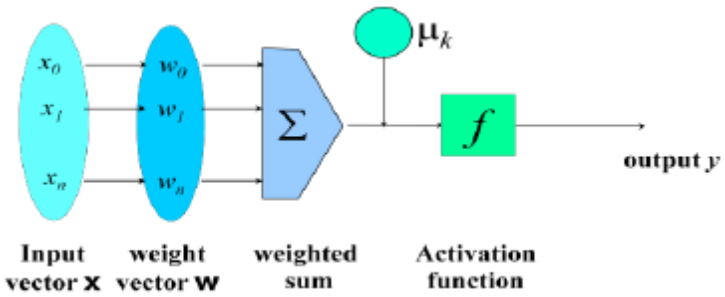
- Backpropagation: A **neural network** learning algorithm
- Started by psychologists and neurobiologists to develop and test computational analogues of neurons
- A neural network: A set of connected input/output units where each connection has a **weight** associated with it
- During the learning phase, the **network learns by adjusting the weights** so as to be able to predict the correct class label of the input tuples

- Also referred to as **connectionist learning** due to the connections between units

Neural Network as a Classifier

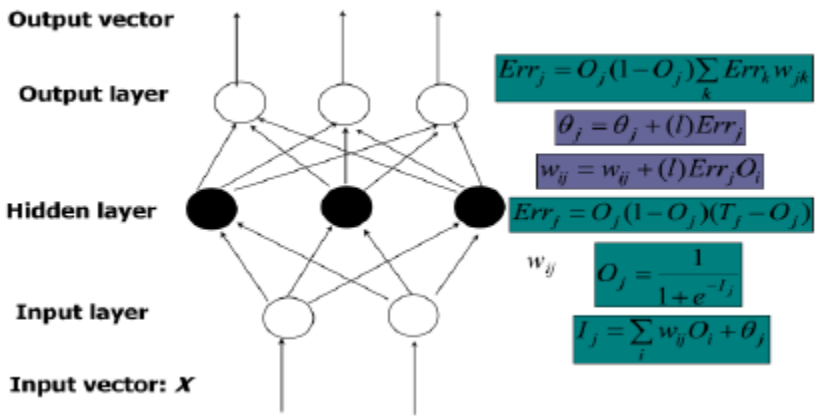
- Weakness
    - Long training time
    - Require a number of parameters typically best determined empirically, e.g., the network topology or “structure”.
    - Poor interpretability: Difficult to interpret the symbolic meaning behind the learned weights and of ``hidden units" in the network
- Strength
    - High tolerance to noisy data
    - Ability to classify untrained patterns
    - Well-suited for continuous-valued inputs and outputs
    - Successful on a wide array of real-world data
    - Algorithms are inherently parallel
    - Techniques have recently been developed for the extraction of rules from trained neural networks

A Neuron (= a perceptron)



- The *n*-dimensional input vector x is mapped into variable y by means of the scalar product and a nonlinear function mapping

A Multi-Layer Feed-Forward Neural Network



- The inputs to the network correspond to the attributes measured for each training tuple
- Inputs are fed simultaneously into the units making up the input layer
- They are then weighted and fed simultaneously to a hidden layer
- The number of hidden layers is arbitrary, although usually only one
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction
- The network is feed-forward in that none of the weights cycles back to an input unit or to an output unit of a previous layer
- From a statistical point of view, networks perform nonlinear regression: Given enough hidden units and enough training samples, they can closely approximate any function

Backpropagation

- Iteratively process a set of training tuples & compare the network's prediction with the actual known target value
- For each training tuple, the weights are modified to minimize the mean squared error between the network's prediction and the actual target value
- Modifications are made in the - backwards‖ direction: from the output layer, through each hidden layer down to the first hidden layer, hence—backpropagation‖
- Steps

- Initialize weights (to small random #s) and biases in the network
- Propagate the inputs forward (by applying activation function)
- Backpropagate the error (by updating weights and biases)
- Terminating condition (when error is very small, etc.)

- Efficiency of backpropagation: Each epoch (one interation through the training set) takes O(|D| * *w*), with |D| tuples and *w* weights, but # of epochs can be exponential to n, the number of inputs, in the worst case
- Rule extraction from networks: network pruning
- Simplify the network structure by removing weighted links that have the least effect on the trained network
- Then perform link, unit, or activation value clustering
- The set of input and activation values are studied to derive rules describing the relationship between the input and hidden unit layers
- Sensitivity analysis: assess the impact that a given input variable has on a network output. The knowledge gained from this analysis can be represented in rules.

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

Input:

D, a data set consisting of the training tuples and their associated target values;
l, the learning rate;
network, a multilayer feed-forward network.
Output: A trained neural network.
Method:
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 {
Oj = Ij ; // output of an input unit is its actual input value

for each hidΣden or output layer unit j {

(8) *Ij = i wij Oi + θj* ; //compute the net input of unit *j* with respect to the previous layer, *i*

Oj = 1 ; } // compute the output of each unit j

// Backpropagate the errors1:+ −*Ije*

for each unit j in the output layer

Errj Oj (1 Oj)(Tj O=j); //−compu−te the error

for each unit j in the hiddeΣn layers, from the last to the first hidden layer

Errj Oj (1 Oj) k=Errk w¯jk ; // compute the error with respect to the next higher layer, k

for each weight wij in network {

Owij = (l)Errj Oi ; // weight increment

wij = wij + Owij ; } // weight update

for each bias θj in network {

Oθj = (l)Errj ; // bias increment

θj = θj + Oθj ; } // bias update

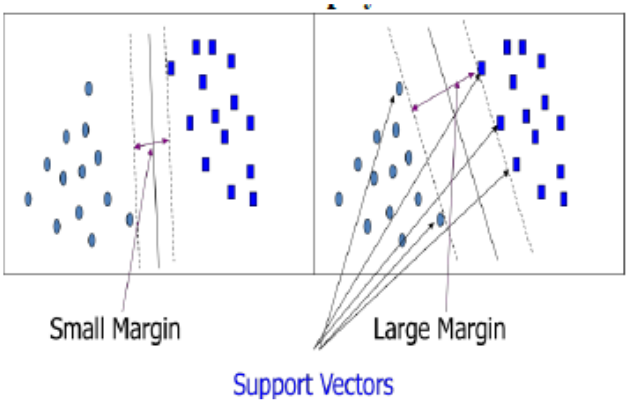
} }

4.9 SVM - Support Vector Machines

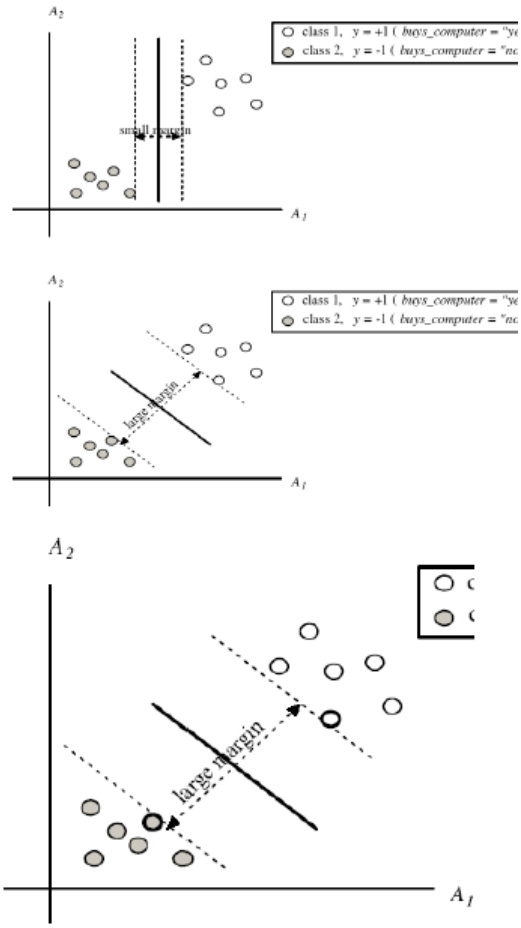
- A new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane (i.e., ―decision boundary‖)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using support vectors (―essential‖ training tuples) and margins (defined by the support vectors)
- Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Used both for classification and prediction

- Applications:
    - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests

SVM—General Philosophy



SVM—Margins and Support Vectors



SVM—Linearly Separable

- A separating hyperplane can be written as

  W ● X + b =0

  where W={w1, w2, …, wn} is a weight vector and b a scalar (bias)

- For 2-D it can be written as

  w0 + w1 x1 + w2 x2 =0

- The hyperplane defining the sides of the margin:

  H1: w0 + w1 x1 + w2 x2≥1 for yi= +1,and

  H2: w0 + w1 x1 + w2 x2 ≤ – 1 for yi = –1

- Any training tuples that fall on hyperplanes H1or H2 (i.e., the sides defining the margin) are support vectors

- This becomes a constrained (convex) quadratic optimization problem: Quadratic objective function and linear constraints *Quadratic Programming (QP)*

  Lagrangian multipliers

Why Is SVM Effective on High Dimensional Data?

- The complexity of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data

- The support vectors are the essential or critical training examples - they lie closest to the decision boundary(MMH)

- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found

- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality

- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

- **Associative Classification**

- Associative classification
  - Association rules are generated and analyzed for use in classification
  - Search for strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels
  - Classification: Based on evaluating a set of rules in the form of P1 ^ p2 … ^ pl □ ⎯A$_{class}$ = C‖(conf, sup)
- Why effective?
  - It explores highly confident associations among multiple attributes and may overcome some constraints introduced by decision-tree induction, which considers only one attribute at a time

In many studies, associative classification has been found to be more accurate than some traditional classification methods, such asC4.

4.11 Prediction

- (Numerical) prediction is similar to classification
  - construct a model
  - use model to predict continuous or ordered value for a given input
- Prediction is different from classification
  - Classification refers to predict categorical class label
  - Prediction models continuous-valued functions
- Major method for prediction: regression
  - model the relationship between one or more *independent* or predictor variables and a *dependent* or response variable
- Regression analysis
  - Linear and multiple regression
  - Non-linear regression

- Other regression methods: generalized linear model, Poisson regression, log-linear models, regression trees

4.12 Linear Regression

- Linear regression: involves a response variable y and a single predictor variable x

  $y = w0 + w1x$

  where w0 (y-intercept) and w1 (slope) are regression coefficients

- Method of least squares: estimates the best-fitting straight line
  - Multiple linear regression: involves more than one predictor variable
  - Training data is of the form (X1, y1), (X2, y2),…, (X|D|,y|D|)
  - Ex. For 2-D data, we may have: $y = w0 + w1\ x1 + w2x2$
  - Solvable by extension of least square method or using SAS,S-Plus
  - Many nonlinear functions can be transformed into the above

4.13 Nonlinear Regression

- Some nonlinear models can be modeled by a polynomial function
- A polynomial regression model can be transformed into linear regression model. For example,

  $y = w0 + w1\ x + w2\ x^2 + w3x^3$

  convertible to linear with new variables: $x2 = x^2$, $x3 = x^3$ $y = w0 + w1\ x + w2\ x2 + w3x3$
- Other functions, such as power function, can also be transformed to linear model
- Some models are intractable nonlinear (e.g., sum of exponential terms)
- possible to obtain least square estimates through extensive calculation on more complex formulae

QUESTIONS

PART A

1. What is the difference between classification and prediction?
2. Explain the issues involved in classification and prediction.
3. What are the different types of regression ?
4. What are the two steps involved in classification?
5. Illustrate how classification and prediction methods are compared to evaluate the performance.
6. How information gain is calculated. Describe the formula which is used to find the information gain?
7. What is tree pruning? Write down it's different approaches.
8. State bayes' theorem.
9. With example describe rule based classification.

PART B

1. How classification can be done using decision tree induction ?
2. Explain in detail the Bayesian classification with examples.
3. Give an overview of the different classification approaches.
4. Describe bayes belief network and state how it is used in classification.
5. What is back propagation? Describe that in detail.
6. Is SVM suitable for very large dataset. How it is used to classify a data? Describe them in detail.
7. What is linear regression? With suitable example, explain linear regression in detail.

TEXT/REFERENCE BOOKS

1. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", 2nd Edition, Elsevier 2007.
2. Alex Berson and Stephen J. Smith, "Data Warehousing, Data Mining & OLAP", Tata McGrew Hill, 2007.
3. www.tutorialspoint.com/dwh/dwh.overview.htm
4. http://study.com/academy/lesson/data-warehousing-and-data-minig-information-for-business-intelligence.html
5. http://www.dei.unpd_it/-capt/SVMATERIALE/DWDM0405.pdf
6. Data mining Concepts and Techniques, Book by Jewei Han and Kamber.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – V - Data Mining and Data Warehousing - SIT1301

V. CLUSTERING, APPLICATIONS AND TRENDS IN DATA MINING

Cluster analysis - Types of data in Cluster Analysis - Categorization of major clustering methods -Partitioning methods - K Means - K Medoids - Hierarchical methods - Density-based methods - Grid-based methods - Model based clustering methods - Constraint Based cluster analysis - Outlier analysis - Data Mining Spatial Applications.

CLUSTER ANALYSIS

Cluster is a group of objects that belongs to the same class. In other words, similar objects aregrouped in one cluster and dissimilar objects are grouped in another cluster.

What is Clustering?

Clustering is the process of making a group of abstract objects into classes of similar objects.

Points to Remember

- A cluster of data objects can be treated as one group.
- While doing cluster analysis, we first partition the set of data into groups based on datasimilarity and then assign the labels to the groups.
- The main advantage of clustering over classification is that, it is adaptable to changes andhelps single out useful features that distinguish different groups.

Applications of Cluster Analysis

- Clustering analysis is broadly used in many applications such as market research, pattern recognition, data analysis, and image processing.
- Clustering can also help marketers discover distinct groups in their customer base. And they can characterize their customer groups based on the purchasing patterns.
- In the field of biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionalities and gain insight into structures inherent to populations.
- Clustering also helps in identification of areas of similar land use in an earth observation database. It also helps in the identification of groups of houses in a city according to house type, value, and geographic location.

- Clustering also helps in classifying documents on the web for information discovery.
- Clustering is also used in outlier detection applications such as detection of credit card fraud.
- As a data mining function, cluster analysis serves as a tool to gain insight into the distribution of data to observe characteristics of each cluster.

Requirements of Clustering in Data Mining

The following points throw light on why clustering is required in data mining −

- **Scalability** − We need highly scalable clustering algorithms to deal with large databases.
- **Ability to deal with different kinds of attributes** − Algorithms should be capable to be applied on any kind of data such as interval-based (numerical) data, categorical, and binary data.
- **Discovery of clusters with attribute shape** − The clustering algorithm should be capable of detecting clusters of arbitrary shape. They should not be bounded to only distance measures that tend to find spherical cluster of small sizes.
- **High dimensionality** − The clustering algorithm should not only be able to handle low- dimensional data but also the high dimensional space.
- **Ability to deal with noisy data** − Databases contain noisy, missing or erroneous data.
  Some algorithms are sensitive to such data and may lead to poor quality clusters.
- **Interpretability** − The clustering results should be interpretable, comprehensible, and usable.

Types of Data in Cluster Analysis

1. Interval - scaled variable
2. Binary Variables
3. Nominal (Categorical) Variables
4. Ordinal Variables
5. Ratio-Scaled Variables
6. Variables of Mixed Types

CATEGORIZATION OF MAJOR CLUSTERING METHODS

Clustering methods can be classified into the following categories −

- Partitioning Method
- Hierarchical Method
- Density-based Method
- Grid-Based Method
- Model-Based Method
- Constraint-based Method

Partitioning Method

Suppose we are given a database of 'n' objects and the partitioning method constructs 'k' partition of data. Each partition will represent a cluster and k ≤ n. It means that it will classify thedata into k groups, which satisfy the following requirements −

- Each group contains at least one object.
- Each object must belong to exactly one group.

Points to remember −

- For a given number of partitions (say k), the partitioning method will create an initial partitioning.
- Then it uses the iterative relocation technique to improve the partitioning by moving objects from one group to other.

Hierarchical Methods

This method creates a hierarchical decomposition of the given set of data objects. We can classify hierarchical methods on the basis of how the hierarchical decomposition is formed. There are two approaches here −

- Agglomerative Approach
- Divisive Approach

Agglomerative Approach

This approach is also known as the bottom-up approach. In this, we start with each object forming a separate group. It keeps on merging the objects or groups that are close

to one another. It keep on doing so until all of the groups are merged into one or until the termination condition holds.

Divisive Approach

This approach is also known as the top-down approach. In this, we start with all of the objects in the same cluster. In the continuous iteration, a cluster is split up into smaller clusters. It is down until each object in one cluster or the termination condition holds. This method is rigid, i.e., oncea merging or splitting is done, it can never be undone.

Approaches to Improve Quality of Hierarchical Clustering

Here are the two approaches that are used to improve the quality of hierarchical clustering −

- Perform careful analysis of object linkages at each hierarchical partitioning.
- Integrate hierarchical agglomeration by first using a hierarchical agglomerative algorithm to group objects into micro-clusters, and then performing macro-clustering on the micro- clusters.

Density-based Method

This method is based on the notion of density. The basic idea is to continue growing the given cluster as long as the density in the neighborhood exceeds some threshold, i.e., for each data point within a given cluster, the radius of a given cluster has to contain at least a minimum number of points.

Grid-based Method

In this, the objects together form a grid. The object space is quantized into finite number of cells that form a grid structure.

Advantage

- The major advantage of this method is fast processing time.
- It is dependent only on the number of cells in each dimension in the quantized space.

Model-based methods

In this method, a model is hypothesized for each cluster to find the best fit of data for a given model. This method locates the clusters by clustering the density function. It reflects spatial distribution of the data points.

This method also provides a way to automatically determine the number of clusters based on standard statistics, taking outlier or noise into account. It therefore yields robust clustering methods.

Constraint-based Method

In this method, the clustering is performed by the incorporation of user or application-oriented constraints. A constraint refers to the user expectation or the properties of desired clustering results. Constraints provide us with an interactive way of communication with the clustering process. Constraints can be specified by the user or the application requirement.

PARTITIONING METHODS

K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

What is K-Means Algorithm?

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on. It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

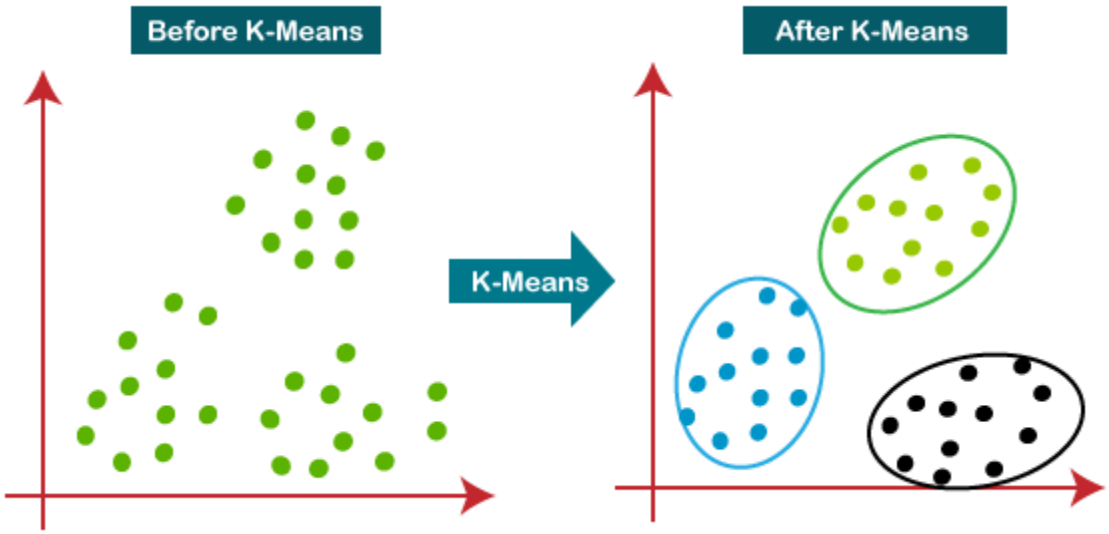
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

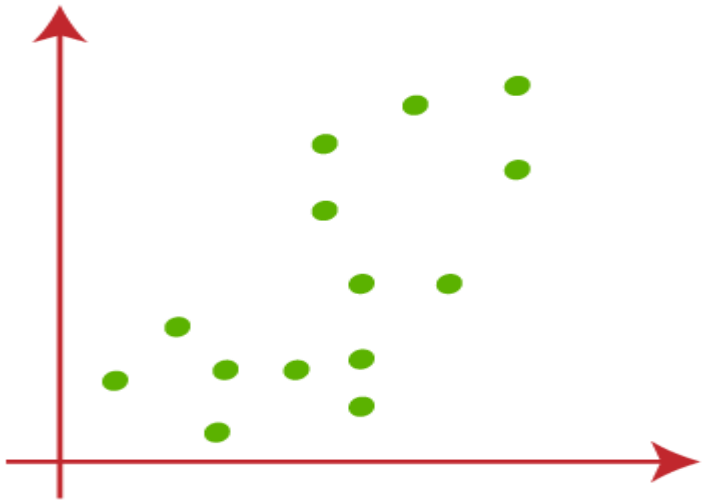
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

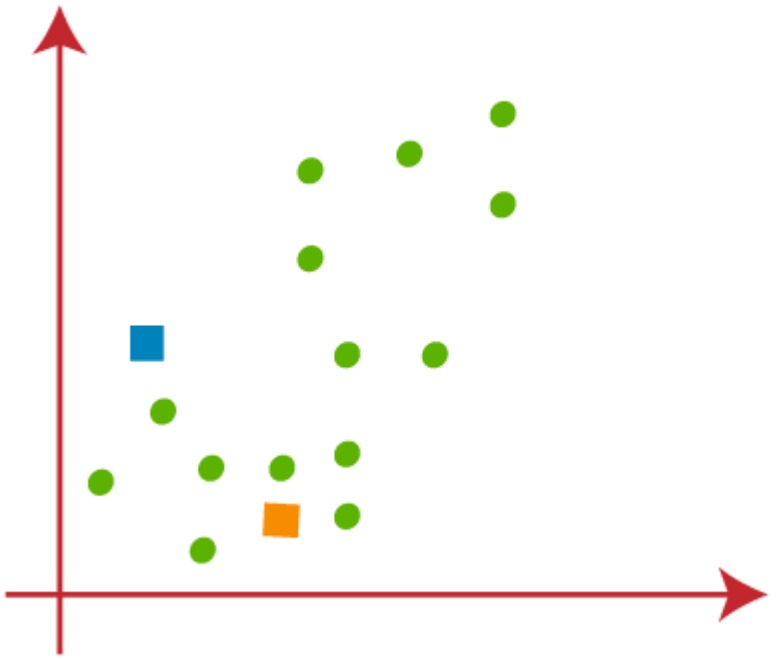
Let's understand the above steps by considering the visual plots:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

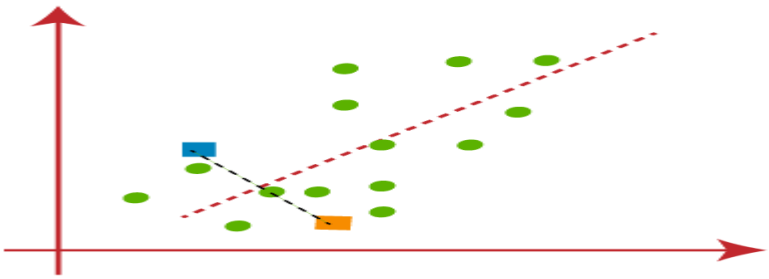


- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the

below image:

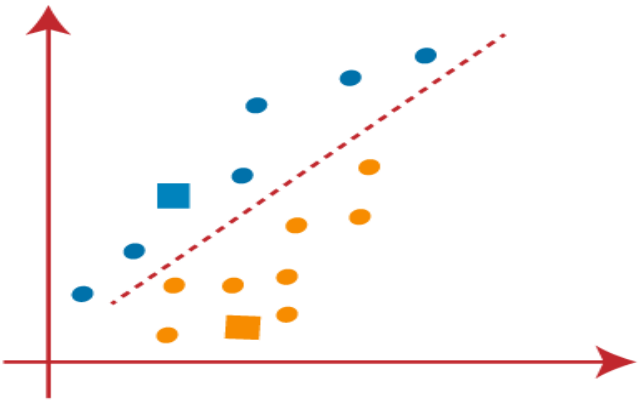


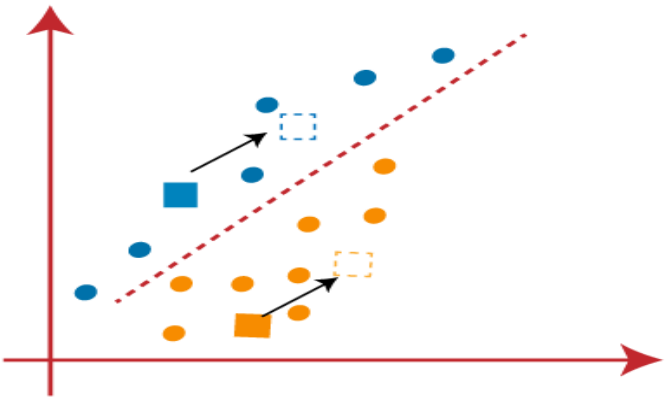
- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:



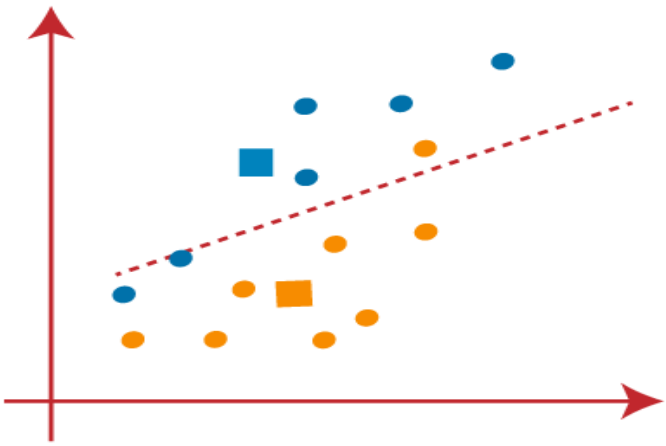
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.

- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:

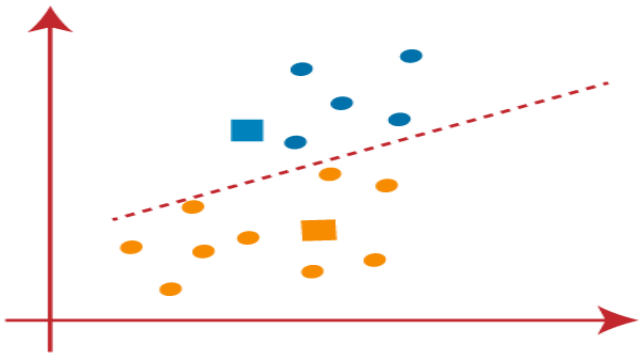




Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:

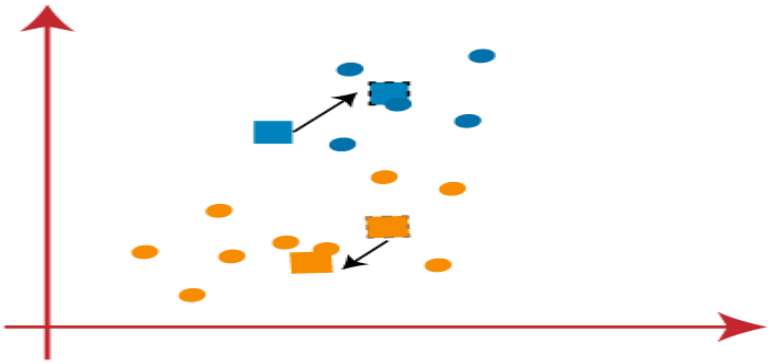


From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

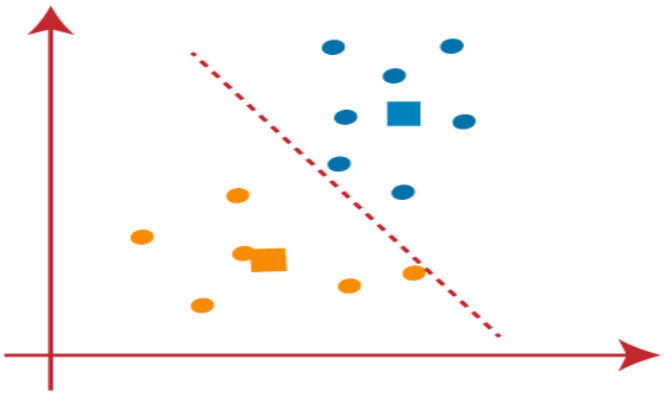


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

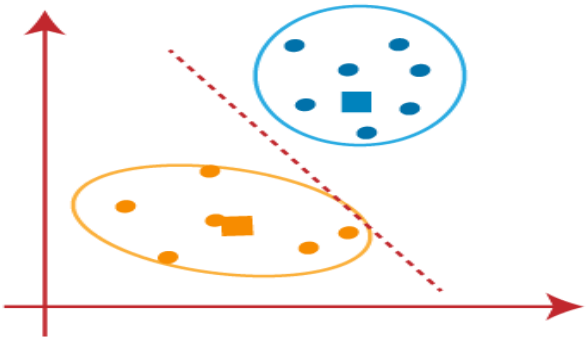
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



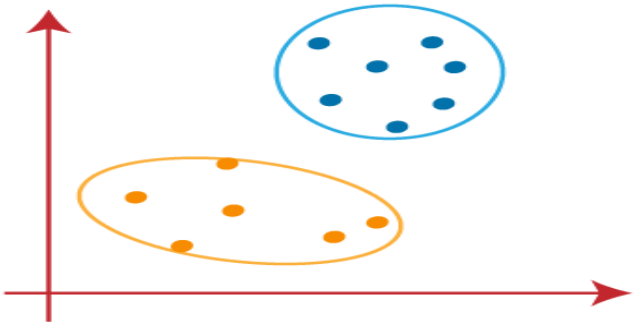
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS= \sum_{Pi \text{ in Cluster1}} distance(P_i\ C_1)^2 + \sum_{Pi \text{ in Cluster2}} distance(P_i\ C_2)^2 + \sum_{Pi \text{ in CLuster3}} distance(P_i\ C_3)^2$$

In the above formula of WCSS,

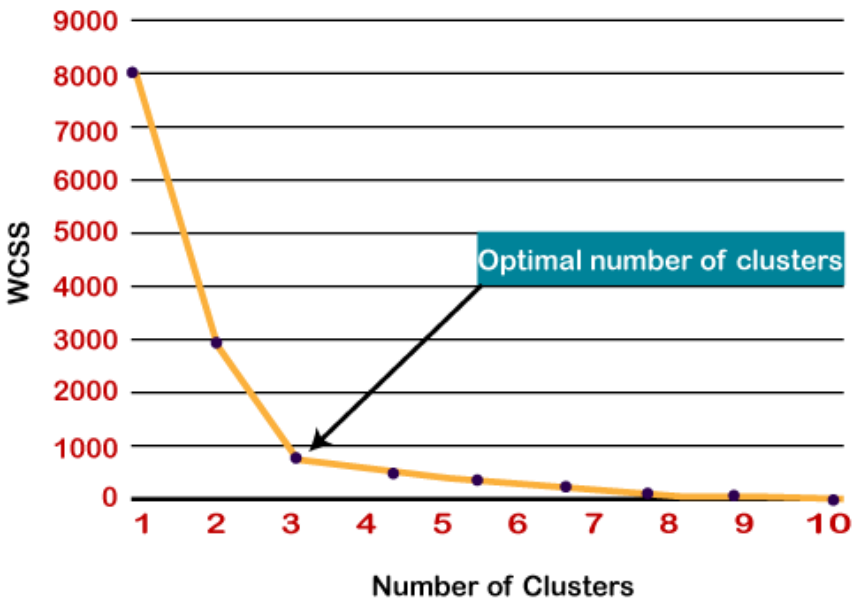
$\sum_{Pi \text{ in Cluster1}} distance(P_i\ C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



We can choose the number of clusters equal to the given data points. If we choose the number of clusters equal to the data points, then the value of WCSS becomes zero, and that will be the endpoint of the plot.

In the above section, we have discussed the K-means algorithm, now let's see how it can be implemented using Python.

Before implementation, let's understand what type of problem we will solve here. So, we have a dataset of **Mall_Customers**, which is the data of customers who visit the mall and spend there.

In the given dataset, we have **Customer_Id, Gender, Age, Annual Income ($), and Spending Score** (which is the calculated value of how much a customer has spent in the mall, the more the value, the more he has spent). From this dataset, we need to calculate some patterns, as it is an unsupervised method, so we don't know what to calculate exactly.

The steps to be followed for the implementation are given below:

- **Data Pre-processing**
- **Finding the optimal number of clusters using the elbow method**
- **Training the K-means algorithm on the training dataset**
- **Visualizing the clusters**

Step-1: Data pre-processing Step

The first step will be the data pre-processing, as we did in our earlier topics of Regression and Classification. But for the clustering problem, it will be different from other models. Let's discuss it:

- **Importing Libraries**

  As we did in previous topics, firstly, we will import the libraries for our model, which is part of data pre-processing. The code is given below:

```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
```

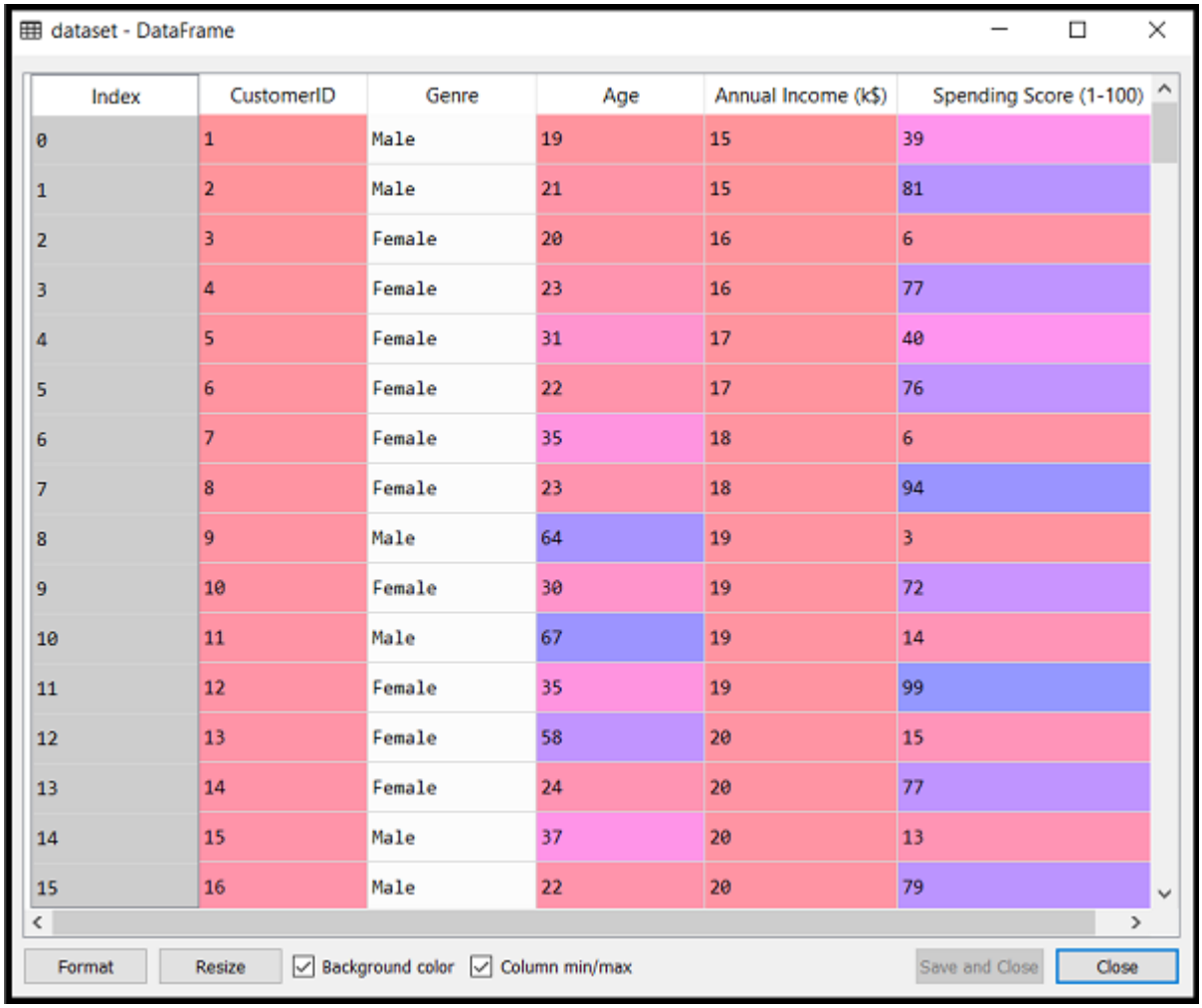
In the above code, the numpy we have imported for the performing mathematics calculation, **matplotlib** is for plotting the graph, and **pandas** are for managing the dataset.

- **Importing the Dataset:**

  Next, we will import the dataset that we need to use. So here, we are using the Mall_Customer_data.csv dataset. It can be imported using the below code:

```
# Importing the dataset
dataset = pd.read_csv('Mall_Customers_data.csv')
```

By executing the above lines of code, we will get our dataset in the Spyder IDE. The dataset looks like the below image:



Index	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13
15	16	Male	22	20	79

From the above dataset, we need to find some patterns in it.

- **Extracting Independent Variables**

Here we don't need any dependent variable for data pre-processing step as it is a clustering problem, and we have no idea about what to determine. So we will just add a line of code for the matrix of features.

```
x = dataset.iloc[:, [3, 4]].values
```

As we can see, we are extracting only 3rd and 4th feature. It is because we need a 2d plot to visualize the model, and some features are not required, such as customer_id.

Step-2: Finding the optimal number of clusters using the elbow method

In the second step, we will try to find the optimal number of clusters for our clustering problem. So, as discussed above, here we are going to use the elbow method for this purpose.

As we know, the elbow method uses the WCSS concept to draw the plot by plotting WCSS values on the Y-axis and the number of clusters on the X-axis. So we are going to calculate the value for WCSS for different k values ranging from 1 to 10. Below is the code for it:

```
#finding optimal number of clusters using the elbow method
from sklearn.cluster import KMeans
wcss_list= []  #Initializing the list for the values of WCSS
#Using for loop for iterations from 1 to 10.
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
    kmeans.fit(x)
    wcss_list.append(kmeans.inertia_)
mtp.plot(range(1, 11), wcss_list)
mtp.title('The Elobw Method Graph')
mtp.xlabel('Number of clusters(k)')
mtp.ylabel('wcss_list')
mtp.show()
```

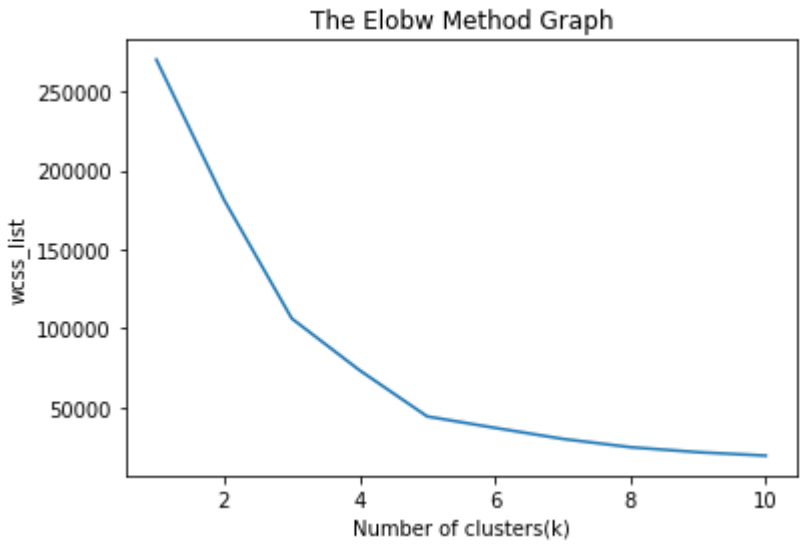
As we can see in the above code, we have used **the KMeans** class of sklearn. cluster library to form the clusters.

Next, we have created the **wcss_list** variable to initialize an empty list, which is used to contain the value of wcss computed for different values of k ranging from 1 to 10.

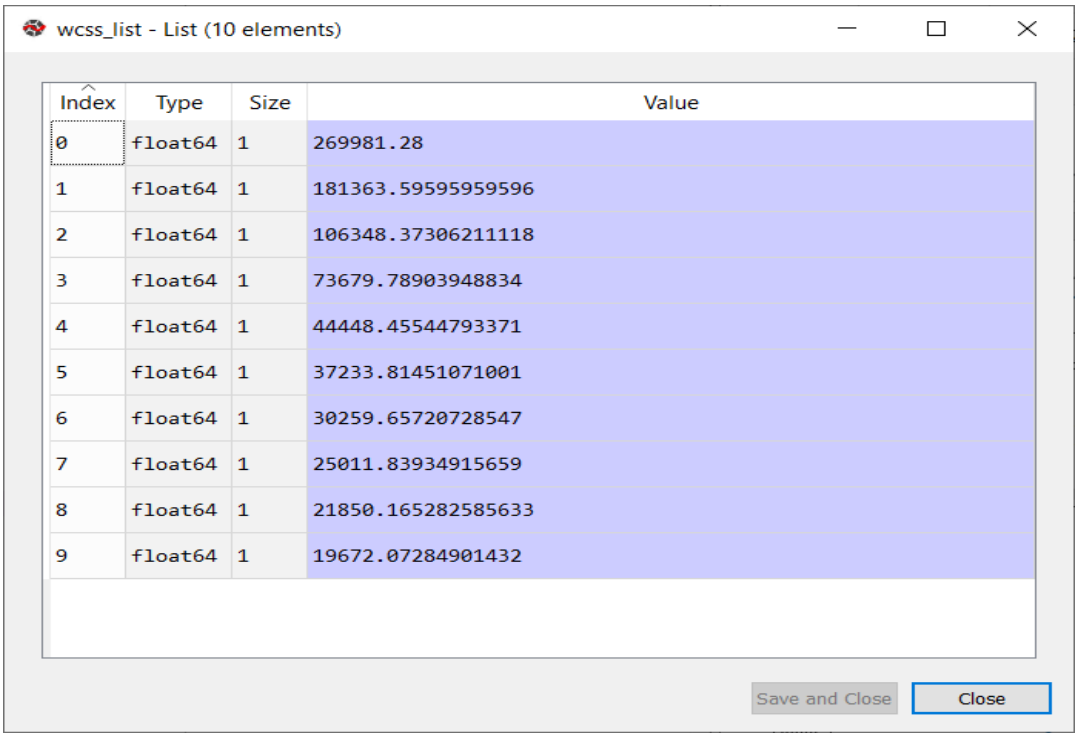
After that, we have initialized the for loop for the iteration on a different value of k ranging from 1 to 10; since for loop in Python, exclude the outbound limit, so it is taken as 11 to include 10^{th} value.

The rest part of the code is similar as we did in earlier topics, as we have fitted the model on a matrix of features and then plotted the graph between the number of clusters and WCSS.

Output: After executing the above code, we will get the below output:



From the above plot, we can see the elbow point is at **5. So the number of clusters here will be 5.**



Step- 3: Training the K-means algorithm on the training dataset

As we have got the number of clusters, so we can now train the model on the dataset.

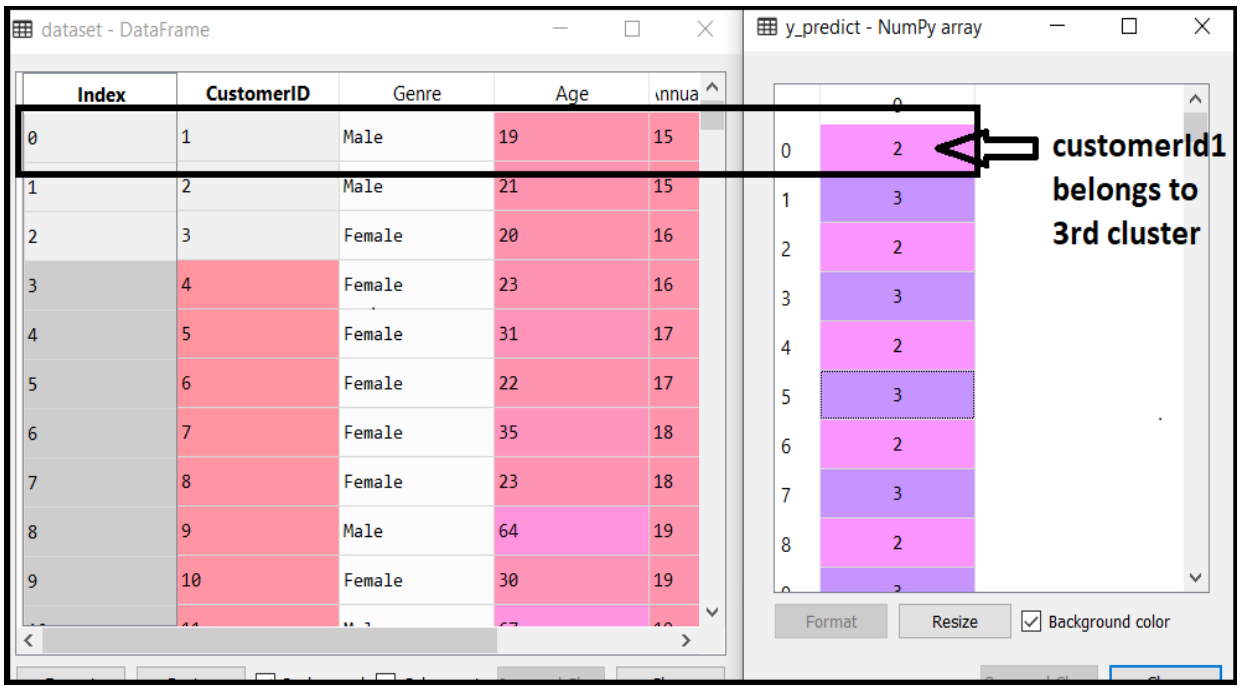
To train the model, we will use the same two lines of code as we have used in the above section, but here instead of using i, we will use 5, as we know there are 5 clusters that need to be formed. The code is given below:

```
#training the K-means model on a dataset
kmeans = KMeans(n_clusters=5, init='k-means++', random_state= 42)
y_predict= kmeans.fit_predict(x)
```

The first line is the same as above for creating the object of KMeans class.

In the second line of code, we have created the dependent variable **y_predict** to train the model.

By executing the above lines of code, we will get the y_predict variable. We can check it under **the variable explorer** option in the Spyder IDE. We can now compare the values of y_predict with our original dataset. Consider the below image:



From the above image, we can now relate that the CustomerID 1 belongs to a cluster

3(as index starts from 0, hence 2 will be considered as 3), and 2 belongs to cluster 4, and so on.

Step-4: Visualizing the Clusters

The last step is to visualize the clusters. As we have 5 clusters for our model, so we will visualize each cluster one by one.

To visualize the clusters will use scatter plot using mtp.scatter() function of matplotlib.

```
#visulaizing the clusters
mtp.scatter(x[y_predict == 0, 0], x[y_predict == 0, 1], s = 100, c = 'blue', label = 'Cluster 1') #for first cluster
mtp.scatter(x[y_predict == 1, 0], x[y_predict == 1, 1], s = 100, c = 'green', label = 'Cluster 2') #for second cluster
mtp.scatter(x[y_predict== 2, 0], x[y_predict == 2, 1], s = 100, c = 'red', label = 'Cluster 3') #for third cluster
mtp.scatter(x[y_predict == 3, 0], x[y_predict == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4') #for fourth cluster
mtp.scatter(x[y_predict == 4, 0], x[y_predict == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5') #for fifth cluster
mtp.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroid')
mtp.title('Clusters of customers')
mtp.xlabel('Annual Income (k$)')
mtp.ylabel('Spending Score (1-100)')
mtp.legend()
mtp.show()
```

In above lines of code, we have written code for each clusters, ranging from 1 to 5. The first coordinate of the mtp.scatter, i.e., x[y_predict == 0, 0] containing the x value for the showing the matrix of features values, and the y_predict is ranging from 0 to 1.

Output:

The output image is clearly showing the five different clusters with different colors. The clusters are formed between two parameters of the dataset; Annual income of customer and Spending. We can change the colors and labels as per the requirement or choice. We can also observe some points from the above patterns, which are given below:

- **Cluster1** shows the customers with average salary and average spending so we can categorize these customers as
- Cluster2 shows the customer has a high income but low spending, so we can categorize them as **careful**.
- Cluster3 shows the low income and also low spending so they can be categorized as sensible.
- Cluster4 shows the customers with low income with very high spending so they can be categorized as **careless**.
- Cluster5 shows the customers with high income and high spending so they can be categorized as target, and these customers can be the most profitable customers for the mall owner.

K - MEDOIDS ALGORITHM

K-Medoids (also called as Partitioning Around Medoid) algorithm was proposed in 1987 by Kaufman and Rousseeuw. A medoid can be defined as the point in the cluster, whose dissimilarities with all the other points in the cluster is minimum.

The dissimilarity of the medoid(Ci) and object(Pi) is calculated by using E = |Pi - Ci|

The cost in K-Medoids algorithm is given as

Algorithm:

1. Initialize: select k random points out of the n data points as the medoids.

2. Associate each data point to the closest medoid by using any common distance metric methods.

3. While the cost decreases:

For each medoid m, for each data o point which is not a medoid:

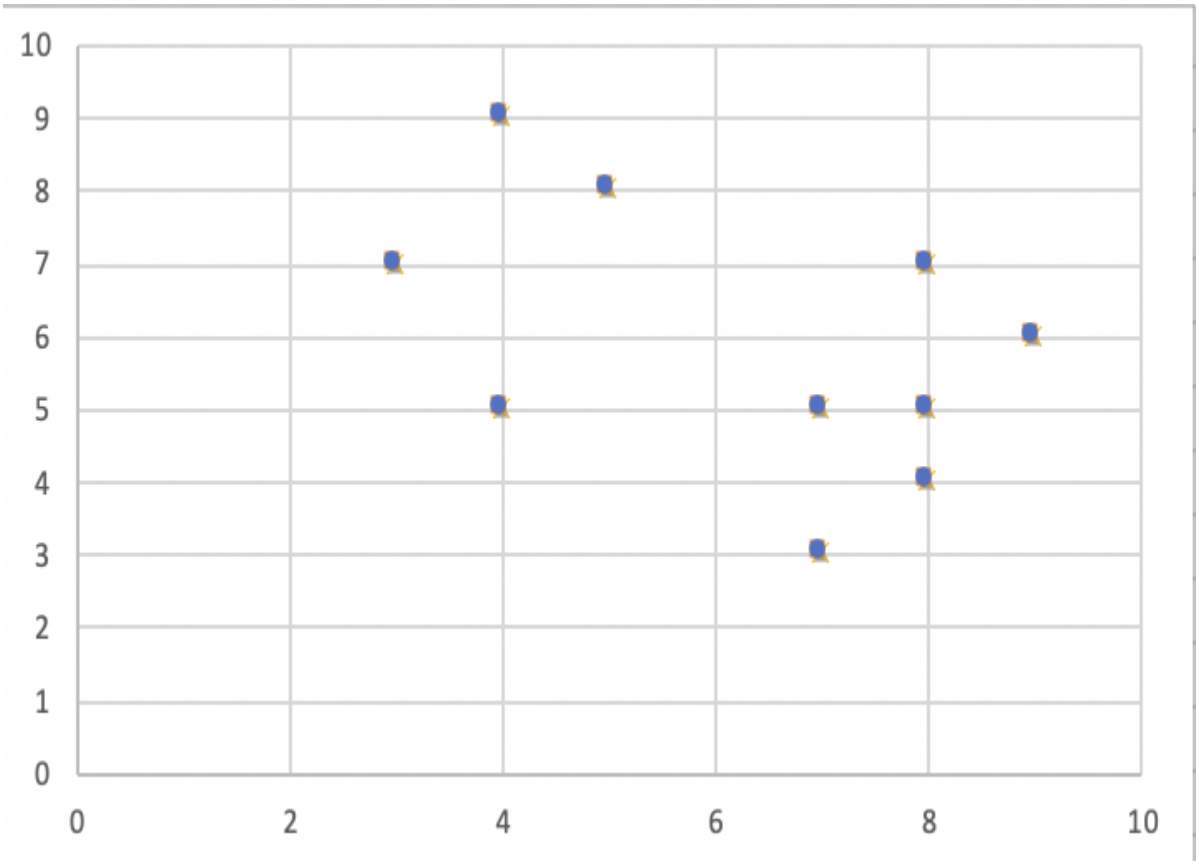
1. Swap m and o, associate each data point to the closest medoid, recompute the cost.

2. If the total cost is more than that in the previous step, undo the swap.

Let’s consider the following example:

	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	8	5
5	5	8
6	7	3
7	8	4
8	7	5
9	4	5

If a graph is drawn using the above data points, we obtain the following:



Step 1:

Let the randomly selected 2 medoids, so select k = 2 and let **C1 -(4, 5)** and **C2 -(8, 5)** are the two medoids.

Step 2: Calculating cost.

The dissimilarity of each non-medoid point with the medoids is calculated and tabulated:

	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	2
1	3	7	3	7
2	4	9	4	8
3	9	6	6	2
4	8	5	-	-
5	5	8	4	6
6	7	3	5	3
7	8	4	5	1
8	7	5	3	1
9	4	5	-	-

Each point is assigned to the cluster of that medoid whose dissimilarity is less.

The points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.

The Cost = (3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20

Step 3: randomly select one non-medoid point and recalculate the cost.

Let the randomly selected point be (8, 4). The dissimilarity of each non-medoid point with the medoids – C1 (4, 5) and C2 (8, 4) is calculated and tabulated.

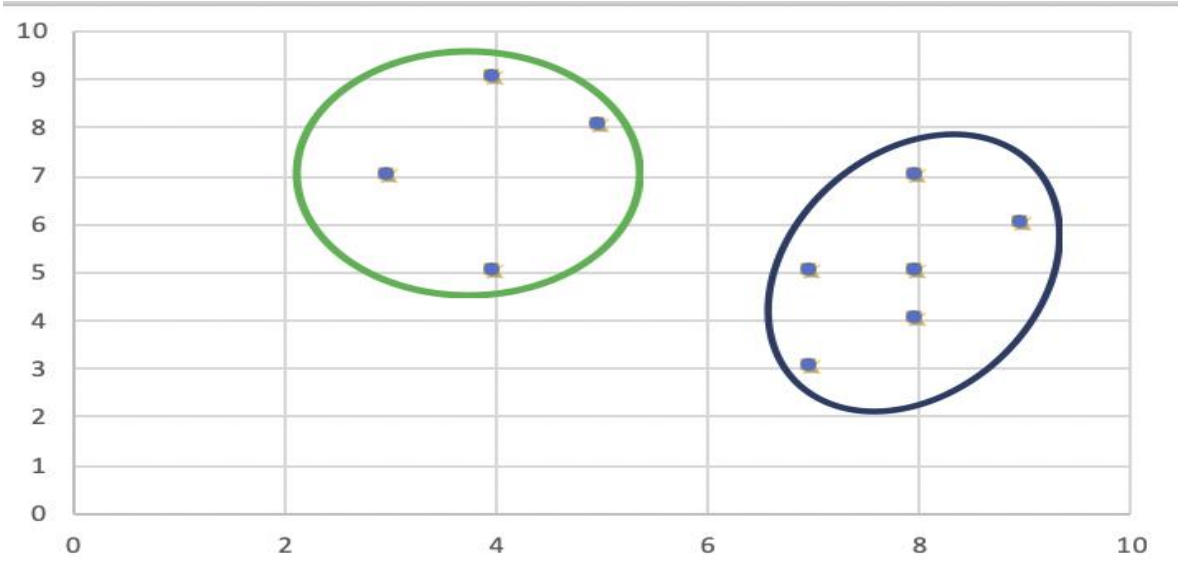
	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	3
1	3	7	3	8
2	4	9	4	9
3	9	6	6	3
4	8	5	4	1
5	5	8	4	7
6	7	3	5	2
7	8	4	-	-
8	7	5	3	2
9	4	5	-	-

Each point is assigned to that cluster whose dissimilarity is less. So, the points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.

The New cost = (3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22

Swap Cost = New Cost – Previous Cost = 22 – 20 and **2 >0**

As the swap cost is not less than zero, we undo the swap. Hence (3, 4) and (7, 4) are the final medoids. The clustering would be in the following way



The **time complexity** is .

Advantages:

1. It is simple to understand and easy to implement.
2. K-Medoid Algorithm is fast and converges in a fixed number of steps.
3. PAM is less sensitive to outliers than other partitioning algorithms.

Disadvantages:

1. The main disadvantage of K-Medoid algorithms is that it is not suitable for clustering non-spherical (arbitrary shaped) groups of objects. This is because it relies on minimizing the distances between the non-medoid objects and the medoid (the cluster centre) – briefly, it uses compactness as clustering criteria instead of connectivity.
2. It may obtain different results for different runs on the same dataset because the first k medoids are chosen randomly.

Hierarchical Clustering in Data Mining

A **Hierarchical clustering** method works via grouping data into a tree of clusters. Hierarchical clustering begins by treating every data points as a separate cluster. Then, it repeatedly executes the subsequent steps:

1. Identify the 2 clusters which can be closest together, and
2. Merge the 2 maximum comparable clusters. We need to continue these steps until all the clusters are merged together.

In Hierarchical Clustering, the aim is to produce a hierarchical series of nested clusters. A diagram called **Dendrogram** (A Dendrogram is a tree-like diagram that statistics the sequences of merges or splits) graphically represents this hierarchy and is an inverted tree that describes the order in which factors are merged (bottom-up view) or cluster are break up (top-down view).

The basic method to generate hierarchical clustering are:

1. Agglomerative:

Initially consider every data point as an **individual** Cluster and at every step, **merge** the nearest pairs of the cluster. (It is a bottom-up method). At first everydata set set is considered as individual entity or cluster. At every iteration, the clusters merge with different clusters until one cluster is formed.

Algorithm for Agglomerative Hierarchical Clustering is:

- Calculate the similarity of one cluster with all the other clusters (calculate proximity matrix)
- Consider every data point as a individual cluster
- Merge the clusters which are highly similar or close to each other.
- Recalculate the proximity matrix for each cluster
- Repeat Step 3 and 4 until only a single cluster remains.

Let’s see the graphical representation of this algorithm using a dendrogram.

Note:

This is just a demonstration of how the actual algorithm works no calculation has been performed below all the proximity among the clusters are assumed.

Let’s say we have six data points **A, B, C, D, E, F**.

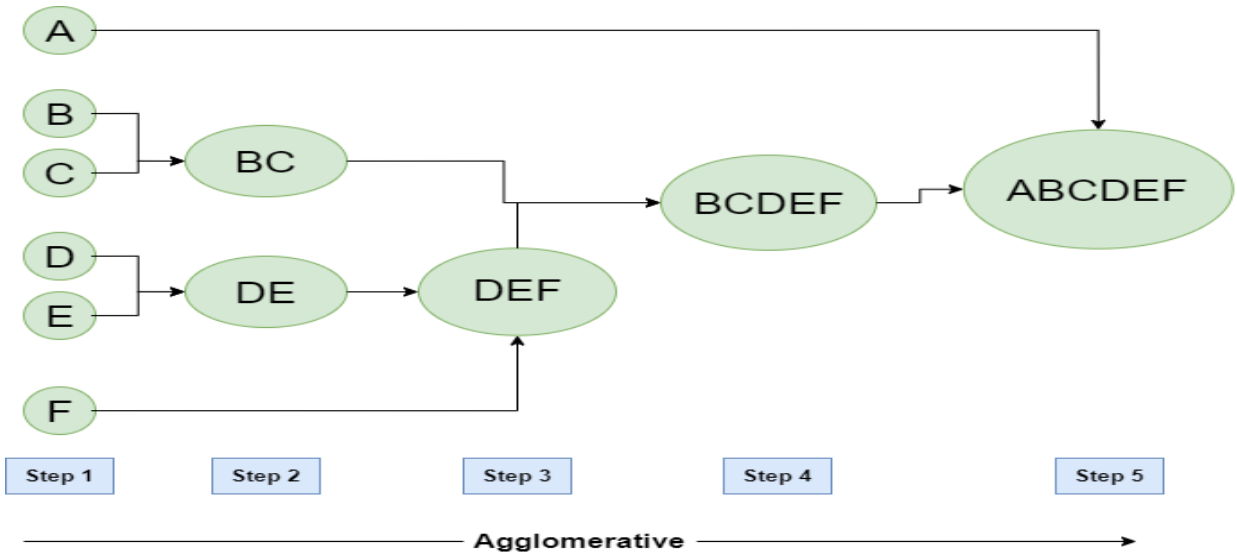


Figure – Agglomerative Hierarchical clustering

- **Step-1:**

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

- **Step-2:**

  In the second step comparable clusters are merged together to form a single cluster. Let’s say cluster (B) and cluster (C) are very similar to each other therefore we merge them in the second step similarly with cluster (D) and (E) and at last, we get the clusters

  [(A), (BC), (DE), (F)]

- **Step-3:**

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

- **Step-4:**

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

- **Step-5:**

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

2. Divisive:

We can say that the Divisive Hierarchical clustering is precisely the **opposite** of the Agglomerative Hierarchical clustering. In Divisive Hierarchical clustering, we take into account all of the data points as a single cluster and in every iteration, we separate the data points from the clusters which aren't comparable. In the end, we are left with N clusters.

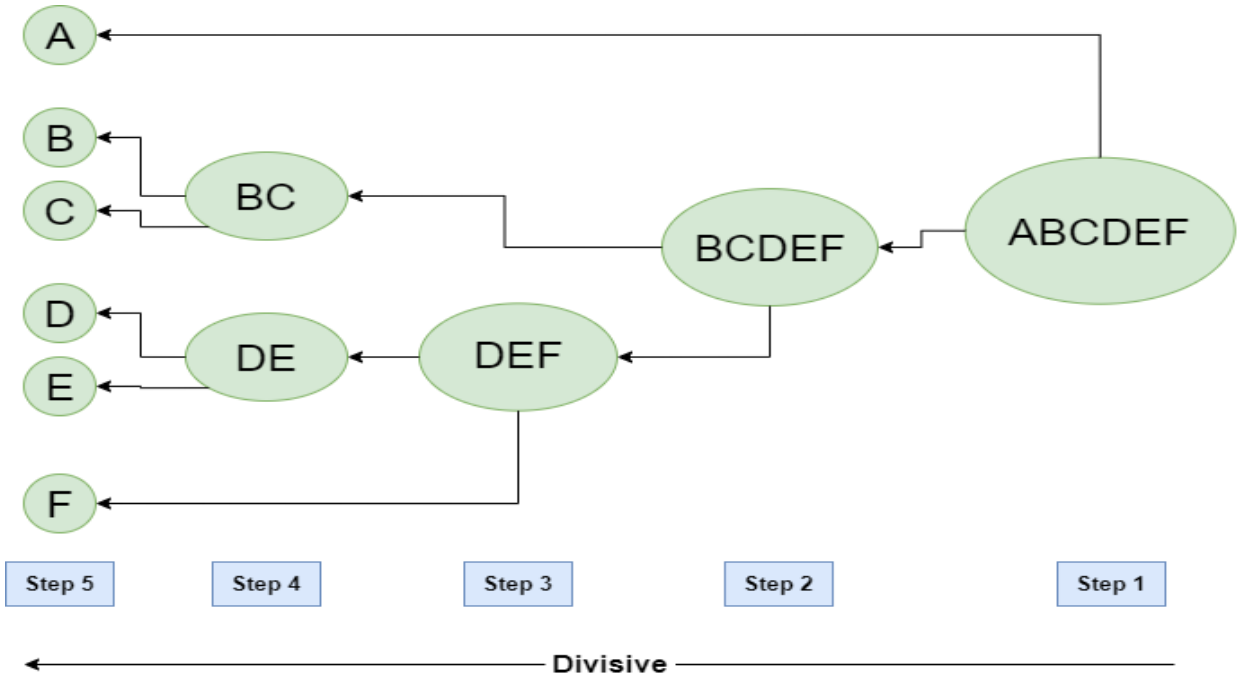


Figure – Divisive Hierarchical clustering

DENSITY BASED CLUSTERING METHOD - DB SCAN

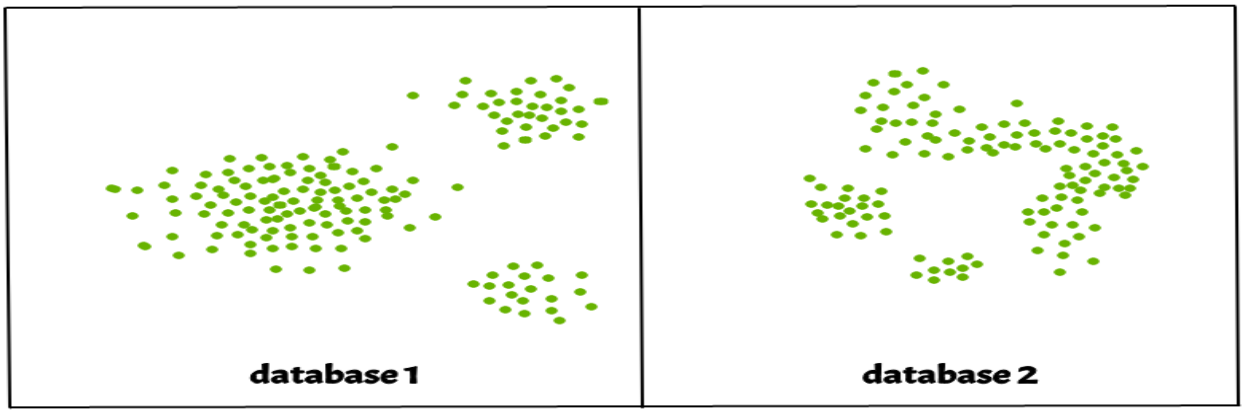
Clustering analysis or simply Clustering is basically an Unsupervised learning method that divides the data points into a number of specific batches or groups, such that the data points in the same groups have similar properties and data points in different groups have different

properties in some sense. It comprises of many different methods based on different evolution.

E.g. K-Means (distance between points), Affinity propagation (graph distance), Mean-shift (distance between points), DBSCAN (distance between nearest points), Gaussian mixtures (Mahalanobis distance to centers), Spectral clustering (graph distance) etc.

Fundamentally, all clustering methods use the same approach i.e. first we calculate similarities and then we use it to cluster the data points into groups or batches. Here we will focus on **Density-based spatial clustering of applications with noise** (DBSCAN) clustering method.

Clusters are dense regions in the data space, separated by regions of the lower density of points. The **DBSCAN algorithm** is based on this intuitive notion of "clusters" and "noise". The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.



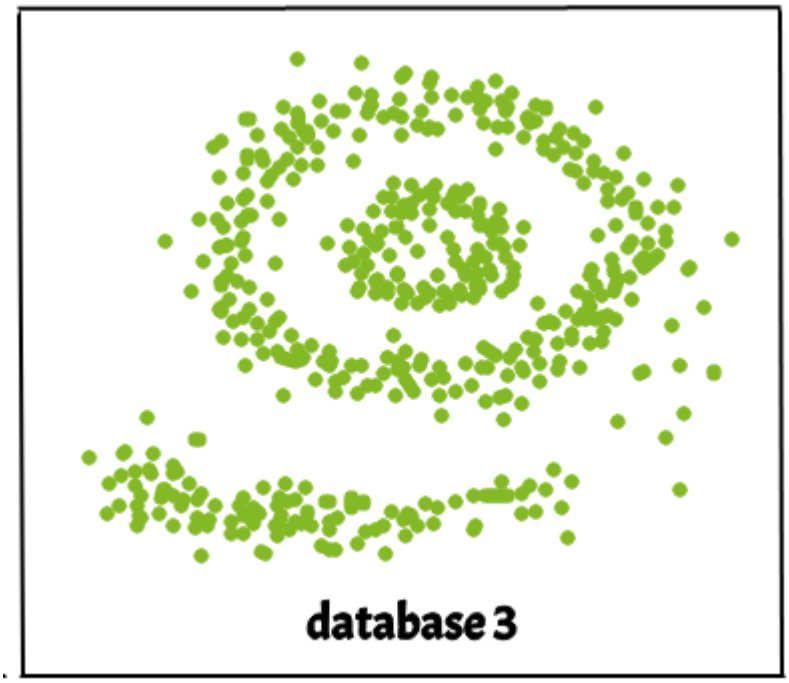
Why DBSCAN ?

Partitioning methods (K-means, PAM clustering) and hierarchical clustering work for finding spherical-shaped clusters or convex clusters. In other words, they are suitable only for compact and well-separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Real life data may contain irregularities, like –

i) Clusters can be of arbitrary shape such as those shown in the figure below.

ii) Data may contain noise.



The figure below shows a data set containing nonconvex clusters and outliers/noises. Given such data, k-means algorithm has difficulties for identifying these clusters with arbitrary shapes.

DBSCAN algorithm requires two parameters –

1. **eps** : It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to 'eps' then they are considered as neighbors. If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters will merge and majority of the data points will be in the same clusters. One way to find the eps value is based on the k-distance graph.
2. **MinPts**: Minimum number of neighbors (data points) within eps radius. Larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1. The minimum value of MinPts must be chosen at least 3.

   In this algorithm, we have 3 types of data points.

3.

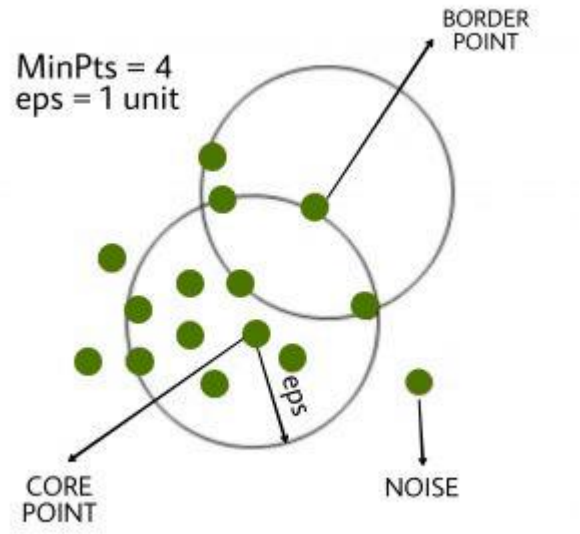
   Core Point*: A point is a core point if it has more than MinPts points within eps.*

   Border Point*: A point which has fewer than MinPts within eps but it is in the*

neighborhood of a core point.

***Noise or outlier**: A point which is not a core point or border point.*

4.



5.

DBSCAN algorithm can be abstracted in the following steps –

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPts neighbors.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.

   A point a and b are said to be density connected if there exist a point c which has a sufficient number of points in its neighbors and both the points a and b are within the eps distance. This is a chaining process. So, if b is neighbor of c, c is neighbor of d, d is neighbor of e, which in turn is neighbor of a implies that b is neighbor of a.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

Below is the DBSCAN clustering algorithm in pseudocode:

DBSCAN(dataset, eps, MinPts){

cluster index

```
C = 1
for each unvisited point p in dataset {
        mark p as visited
        # find neighbors
        Neighbors N = find the neighboring points of p

        if |N|>=MinPts:
            N = N U N'
            if p' is not a member of any cluster:
                add p' to cluster C
}
```

Implementation of above algorithm in Python :

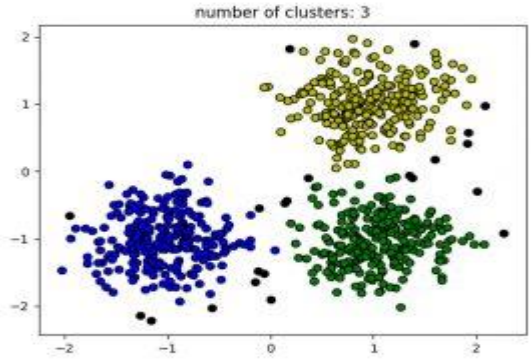
Here, we'll use the Python library sklearn to compute DBSCAN. We'll also use the matplotlib.pyplot library for visualizing clusters.

The dataset used can be found here.

```
import numpy as np
from sklearn.cluster import DBSCAN
from sklearn import metrics
from sklearn.datasets.samples_generator import make_blobs
from sklearn.preprocessing import StandardScaler
from sklearn import datasets

# Load data in X
db = DBSCAN(eps=0.3, min_samples=10).fit(X)
core_samples_mask = np.zeros_like(db.labels_, dtype=bool)
core_samples_mask[db.core_sample_indices_] = True
labels = db.labels_
```

```
# Number of clusters in labels, ignoring noise if present.
n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)

print(labels)

# Plot result
import matplotlib.pyplot as plt

# Black removed and is used for noise instead.
unique_labels = set(labels)
colors = ['y', 'b', 'g', 'r']
print(colors)
for k, col in zip(unique_labels, colors):
    if k == -1:
        # Black used for noise.
        col = 'k'

    class_member_mask = (labels == k)

    xy = X[class_member_mask & core_samples_mask]
    plt.plot(xy[:, 0], xy[:, 1], 'o', markerfacecolor=col,
                                 markeredgecolor='k',
                                 markersize=6)

    xy = X[class_member_mask & ~core_samples_mask]
    plt.plot(xy[:, 0], xy[:, 1], 'o', markerfacecolor=col,
                                 markeredgecolor='k',
                                 markersize=6)

plt.title('number of clusters: %d' %n_clusters_)
plt.show()
```

Output:

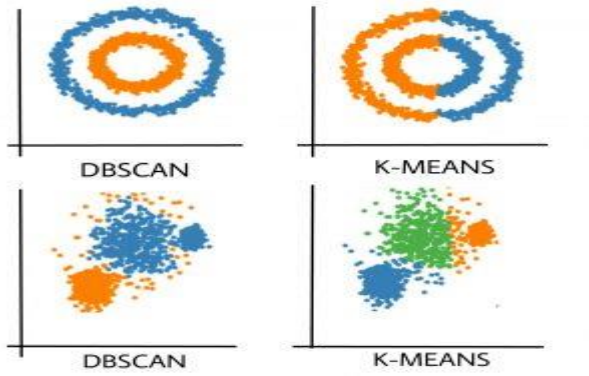


Black points represent outliers. By changing the eps and the MinPts , we can change the cluster configuration.

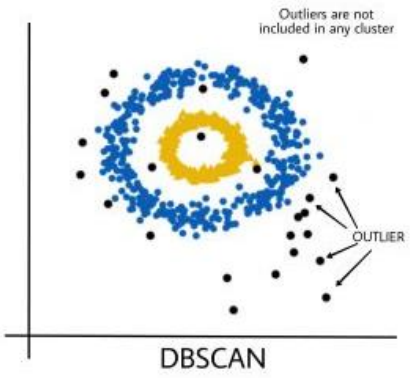
Now the question should be raised is – *Why should we use DBSCAN where K-Means is the widely used method in clustering analysis?*

Disadvantage Of K-MEANS:

1. K-Means forms spherical clusters only. This algorithm fails when data is not spherical (i.e. same variance in all directions).



2. K-Means algorithm is sensitive towards outlier. Outliers can skew the clusters in K-Means in very large extent.



3. K-Means algorithm requires one to specify the number of clusters a priory etc.

Basically, DBSCAN algorithm overcomes all the above-mentioned drawbacks of K-Means algorithm. DBSCAN algorithm identifies the dense region by grouping together data points that are closed to each other based on distance measurement.

OPTICS

OPTICS works like an extension of DBSCAN. The only difference is that it does not assign cluster memberships but stores the order in which the points are processed. So for each object stores: *Core distance* and Reachability distance. Order Seeds is called the record which constructs the output order.

Core Distance: The minimum value of ε which is present in the ε-neighborhood of a P is a core distance. Of course, it's needed to hold the minimum MinPts objects.

Reachability Distance: Reachability distance between p and q is defined as the least radius value that formulates p density reachable from q.

Algorithm

1. Randomly selects an unvisited point P
2. Selects all point's density reachable from P w.r.t Eps, MinPts.
3. Assign core distance & reachability distance = NULL
4. If P is not a core point
5. Move next point in the order Seeds list
6. If P is a core point
7. For each object q, in the ε — neighborhood of P
8. UPDATE reachability distance from P
9. If q is unvisited INSERT q into Order Seeds
10. Until no object is unvisited

Note: The algorithm above is taken from 'Review on Density-Based Clustering Algorithms for Big Data'.

Advantage

- It does not require density parameters.
- The clustering order is useful to extract the basic clustering information.

Disadvantage

- It only produces a cluster ordering.
- It can’t handle high dimensional data.

Grid-Based Clustering - STING, WaveCluster & CLIQUE

April 06, 2020

Grid-Based Clustering

Grid-Based Clustering method uses a multi-resolution grid data structure.

Several interesting methods

- **STING** (a **ST**atistical **IN**formation **G**rid approach)
- **WaveCluster** - A multi-resolution clustering approach using wavelet method
- **CLIQUE**

STING - A Statistical Information Grid Approach

STING was proposed by Wang, Yang, and Muntz.

In this method, the spatial area is divided into rectangular cells.

There are several levels of cells corresponding to different levels of resolution.For each cell, the high level is partitioned into several smaller cells in the next lower level.

The statistical info of each cell is calculated and stored beforehand and is used to answer queries.

The parameters of higher-level cells can be easily calculated from parameters of lower-level cell

- Count, mean, s, min, max
- Type of distribution—normal, uniform, etc.

Then using a top-down approach we need to answer spatial data queries. Then start from a pre-selected layer—typically with a small number of cells.

For each cell in the current level compute the confidence interval.
Now remove the irrelevant cells from further consideration.

When finishing examining the current layer, proceed to the next lower level.
Repeat this process until the bottom layer is reached.

Advantages:

It is Query-independent, easy to parallelize, incremental update.

O(K), where K is the number of grid cells at the lowest level.

Disadvantages:

All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected.

WaveCluster

It was proposed by Sheikholeslami, Chatterjee, and Zhang.

It is a multi-resolution clustering approach which applies wavelet transform to the feature space

- A wavelet transform is a signal processing technique that decomposes a signal into different frequency sub-band.
- It can be both grid-based and density-based method.

Input parameters:

- No of grid cells for each dimension
- The wavelet, and the no of applications of wavelet transform.

How to apply the wavelet transform to find clusters

- It summaries the data by imposing a multidimensional grid structure onto data space.
- These multidimensional spatial data objects are represented in an n-dimensional feature space.

- Now apply wavelet transform on feature space to find the dense regions in the feature space.
- Then apply wavelet transform multiple times which results in clusters at different scales from fine to coarse.

Why is wavelet transformation useful for clustering

- It uses hat-shape filters to emphasize region where points cluster, but simultaneously to suppress weaker information in their boundary.
- It is an effective removal method for outliers.
- It is of Multi-resolution method.
- It is cost-efficiency.

Major features:

- The time complexity of this method is O(N).
- It detects arbitrary shaped clusters at different scales.
- It is not sensitive to noise, not sensitive to input order.
- It only applicable to low dimensional data.

CLIQUE - Clustering In QUEst

It was proposed by Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98).
It is based on automatically identifying the subspaces of high dimensional data space that allow better clustering than original space.

CLIQUE can be considered as both density-based and grid-based:

- It partitions each dimension into the same number of equal-length intervals.
- It partitions an m-dimensional data space into non-overlapping rectangular units.
- A unit is dense if the fraction of the total data points contained in the unit exceeds the input model parameter.

- A cluster is a maximal set of connected dense units within a subspace.

Partition the data space and find the number of points that lie inside each cell of the partition.

Identify the subspaces that contain clusters using the Apriori principle.

Identify clusters:

- Determine dense units in all subspaces of interests.
- Determine connected dense units in all subspaces of interests.

Generate minimal description for the clusters:

- Determine maximal regions that cover a cluster of connected dense units for each cluster.
- Determination of minimal cover for each cluster.

Advantages

It automatically finds subspaces of the highest dimensionality such that high-density clusters exist in those subspaces.

It is insensitive to the order of records in input and does not presume some canonical data distribution.

It scales linearly with the size of input and has good scalability as the number of dimensions in the data increases.

Disadvantages

The accuracy of the clustering result may be degraded at the expense of the simplicity of the method.

Summary

Grid-Based Clustering -> It is one of the methods of cluster analysis which uses a multi-resolution grid data structure.

MODEL BASED CLUSTERING

The traditional clustering methods, such as hierarchical clustering and k-means clustering, are heuristic and are not based on formal models. Furthermore, k-means algorithm is commonly randomnly initialized, so different runs of k-means will often yield different results. Additionally, k-means requires the user to specify the the optimal number of clusters.

An alternative is **model-based clustering**, which consider the data as coming from a distribution that is mixture of two or more clusters (Fraley and Raftery 2002, Fraley et al. (2012)). Unlike k-means, the model-based clustering uses a soft assignment, where each data point has a probability of belonging to each cluster.

Concept of model-based clustering

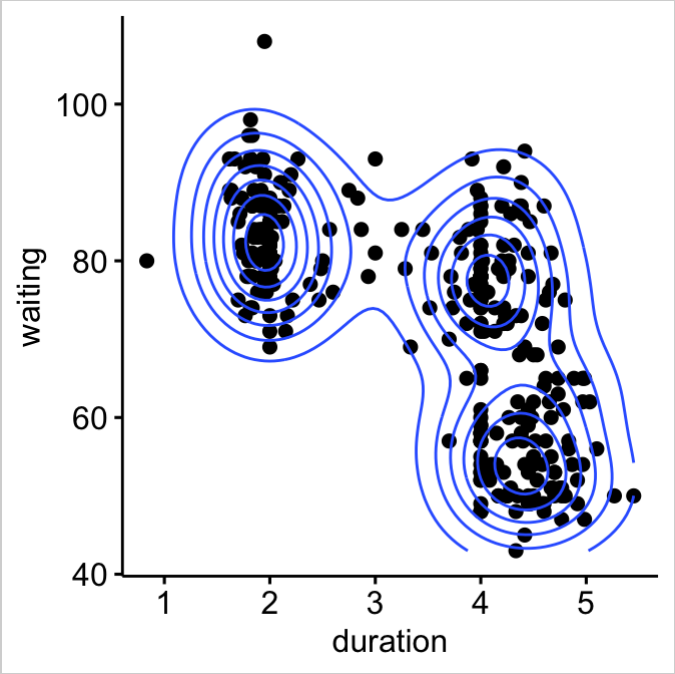
In model-based clustering, the data is considered as coming from a mixture of density.

Each component (i.e. cluster) k is modeled by the normal or Gaussian distribution which is characterized by the parameters:

- μkμk: mean vector,
- ∑k∑k: covariance matrix,
- An associated probability in the mixture. Each point has a probability of belonging to each cluster.

For example, consider the “old faithful geyser data” [in MASS R package], which can be illustrated as follow using the ggpubr R package:

```
# Load the data
library("MASS")

data("geyser")

# Scatter plot
library("ggpubr")

ggscatter(geyser, x = "duration", y = "waiting")+

 geom_density2d() # Add 2D density
```



The plot above suggests at least 3 clusters in the mixture. The shape of each of the 3 clusters appears to be approximately elliptical suggesting three bivariate normal distributions. As the 3 ellipses seems to be similar in terms of volume, shape and orientation, we might anticipate that the three components of this mixture might have homogeneous covariance matrices.

Estimating model parameters

The model parameters can be estimated using the *Expectation-Maximization* (EM) algorithm initialized by hierarchical model-based clustering. Each cluster k is centered at the means μkμk, with increased density for points near the mean.

Geometric features (shape, volume, orientation) of each cluster are determined by the covariance matrix ∑k∑k.

Different possible parameterizations of ∑k∑k are available in the R package *mclust* (see *?mclustModelNames*).

The available model options, in *mclust* package, are represented by identifiers including: EII, VII, EEI, VEI, EVI, VVI, EEE, EEV, VEV and VVV.

The first identifier refers to volume, the second to shape and the third to orientation. E stands for “equal”, V for “variable” and I for “coordinate axes”.

For example:

- EVI denotes a model in which the volumes of all clusters are equal (E), the shapes of the clusters may vary (V), and the orientation is the identity (I) or “coordinate axes.
- EEE means that the clusters have the same volume, shape and orientation in p-dimensional space.
- VEI means that the clusters have variable volume, the same shape and orientation equal to coordinate axes.

Choosing the best model

The *Mclust* package uses maximum likelihood to fit all these models, with different covariance matrix parameterizations, for a range of k components.

The best model is selected using the Bayesian Information Criterion or *BIC*. A large BIC score indicates strong evidence for the corresponding model.

Computing model-based clustering

We start by installing the *mclust* package as follow: *install.packages(“mclust”)*

Note that, model-based clustering can be applied on univariate or multivariate data.

Here, we illustrate model-based clustering on the diabetes data set [mclust package] giving three measurements and the diagnosis for 145 subjects described as follow:

```
library("mclust")

data("diabetes")

head(diabetes, 3)
```

```
##    class glucose insulin sspg
## 1 Normal      80     356  124
## 2 Normal      97     289  117
## 3 Normal     105     319  143
```

- class: the diagnosis: normal, chemically diabetic, and overtly diabetic. Excluded from the cluster analysis.
- glucose: plasma glucose response to oral glucose
- insulin: plasma insulin response to oral glucose
- sspg: steady-state plasma glucose (measures insulin resistance)

Model-based clustering can be computed using the function Mclust() as follow:

```
library(mclust)

df <- scale(diabetes[, -1]) # Standardize the data

mc <- Mclust(df)            # Model-based-clustering

summary(mc)                 # Print a summary
```

```
## ----------------------------------------------------

## Gaussian finite mixture model fitted by EM algorithm

## ----------------------------------------------------

##

## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 3 components:

##  log.likelihood   n df  BIC  ICL

##           -169 145 29 -483 -501

## Clustering table:

##  1  2  3

## 81 36 28
```

For this data, it can be seen that model-based clustering selected a model with three components (i.e. clusters). The optimal selected model name is VVV model. That is the three

components are ellipsoidal with varying volume, shape, and orientation. The summary contains also the clustering table specifying the number of observations in each clusters.

You can access to the results as follow:

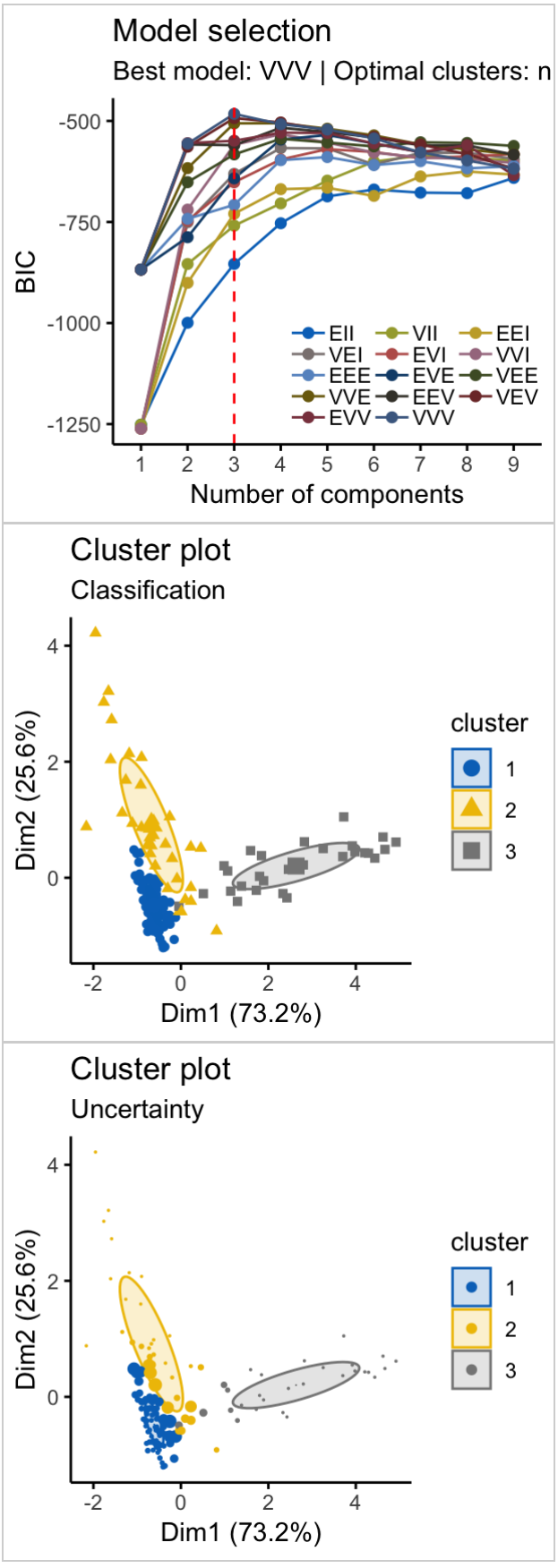
```
mc$modelName              # Optimal selected model ==> "VVV"

mc$G                      # Optimal number of cluster => 3

head(mc$z, 30)            # Probality to belong to a given cluster

head(mc$classification, 30) # Cluster assignement of each observation
```

Visualizing model-based clustering

Model-based clustering results can be drawn using the base function plot.Mclust() [in mclust package]. Here we'll use the function *fviz_mclust*() [in *factoextra* package] to create beautiful plots based on ggplot2.

In the situation, where the data contain more than two variables, *fviz_mclust*() uses a principal component analysis to reduce the dimensionnality of the data. The first two principal components are used to produce a scatter plot of the data. However, if you want to plot the data using only two variables of interest, let say here c("insulin", "sspg"), you can specify that in the *fviz_mclust*() function using the argument *choose.vars = c("insulin", "sspg")*.

```
library(factoextra)# BIC values used for choosing the number of clusters

fviz_mclust(mc, "BIC", palette = "jco")# Classification: plot showing the clustering

fviz_mclust(mc, "classification", geom = "point",

        pointsize = 1.5, palette = "jco")# Classification uncertainty

fviz_mclust(mc, "uncertainty", palette = "jco")
```



Note that, in the uncertainty plot, larger symbols indicate the more uncertain observations.

CONTRAINED CLUSTERING

Constrained clustering is a class of semi-supervised learning algorithms. Typically, constrained clustering incorporates either a set of must-link constraints, cannot-link constraints, or both, with a Data clustering algorithm. Both a must-link and a cannot-link constraint define a relationship between two data instances. A must-link constraint is used to specify that the two instances in the must-link relation should be associated with the same cluster. A cannot-link constraint is used to specify that the two instances in the cannot-link relation should *not* be associated with the same cluster. These sets of constraints acts as a guide for which a constrained clustering algorithm will attempt to find clusters in a data set which satisfy the specified must-link and cannot-link constraints. Some constrained clustering algorithms will abort if no such clustering exists which satisfies the specified constraints. Others will try to minimize the amount of constraint violation should it be impossible to find a clustering which satisfies the constraints. Constraints could also be used to guide the selection of a clustering model among several possible solutions. A cluster in which the members conform to all must-link and cannot-link constraints is called a **chunklet**.

Examples of constrained clustering algorithms include:

COP K-means

PCKmeans (Pairwise Constrained K-means)

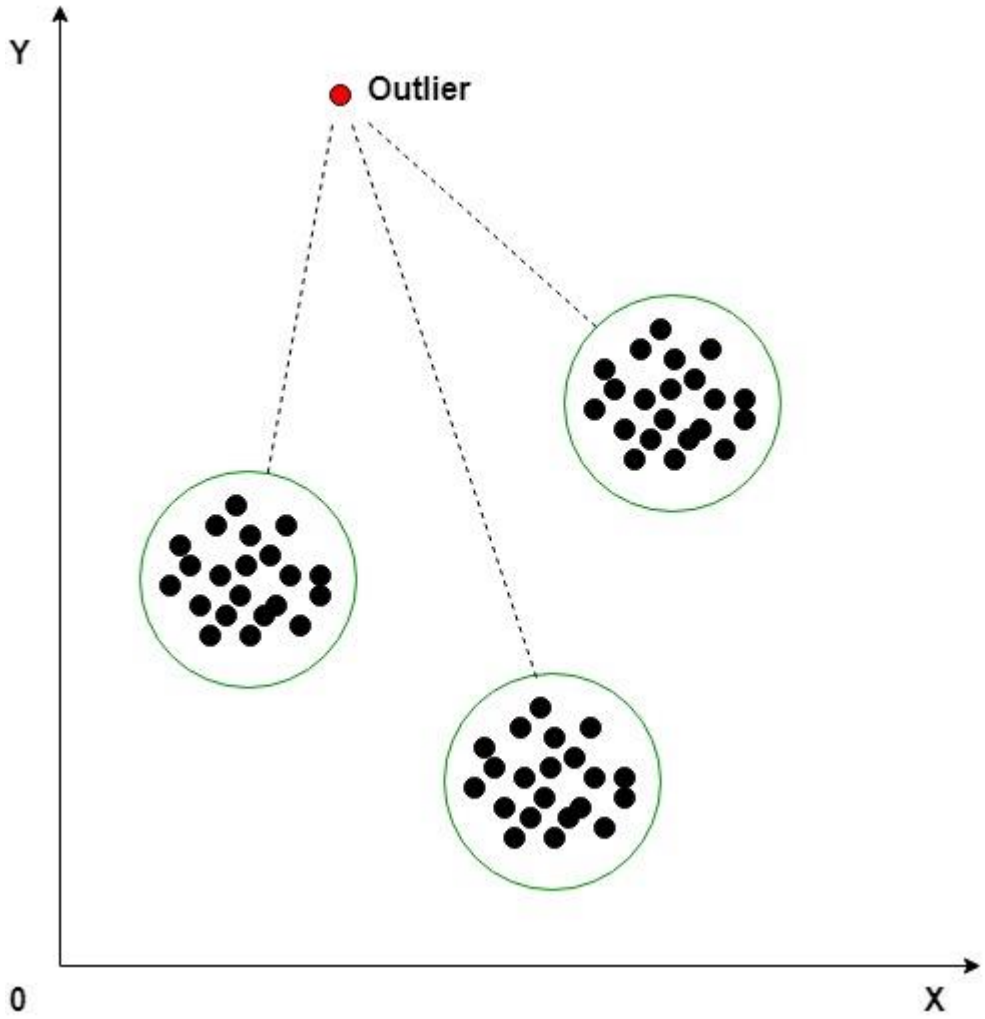
MWK-Means (Constrained Minkowski Weighted K-Means)

OUTLIER ANALYSIS

An **outlier** is an object that deviates significantly from the rest of the objects. They can be caused by measurement or execution error. The analysis of outlier data is referred to as outlier analysis or outlier mining.

Why outlier analysis?

Most data mining methods discard outliers noise or exceptions, however, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring one and hence, the outlier analysis becomes important in such case.



Detecting Outlier:

Clustering based outlier detection using distance to the closest cluster:

In the K-Means clustering technique, each cluster has a mean value. Objects belong to the cluster whose mean value is closest to it. In order to identify the Outlier, firstly we need to initialize the threshold value such that any distance of any data point greater than it from its nearest cluster identifies it as an outlier for our purpose. Then we need to find the distance of the test data to each cluster mean. Now, if the distance between the test data and the closest cluster to it is greater than the threshold value then we will classify the test data as an outlier.

Algorithm:

1. Calculate the mean of each cluster
2. Initialize the Threshold value

3. Calculate the distance of the test data from each cluster mean
4. Find the nearest cluster to the test data
5. If (Distance > Threshold) then, Outlier

Data Mining Applications

Data mining is widely used in diverse areas. There are a number of commercial data mining system available today and yet there are many challenges in this field. In this tutorial, we will discuss the applications and the trend of data mining.

Applications

Here is the list of areas where data mining is widely used −

- Financial Data Analysis
- Retail Industry
- Telecommunication Industry
- Biological Data Analysis
- Other Scientific Applications
- Intrusion Detection

1) Data Mining for Financial Data Analysis

- Design and construction of data warehouses for multidimensional data analysis and data mining
- Loan payment prediction and customer credit policy analysis
- Classification and clustering of customers for targeted marketing
- Detection of money laundering and other financial crimes
- Data Mining for the Retail Industry

2)A few examples of data mining in the retail industry

- Design and construction of data warehouses based on the benefits of data mining
- Multidimensional analysis of sales, customers, products, time, and region

- Analysis of the effectiveness of sales campaigns
- Customer retention—analysis of customer loyalty
- Product recommendation and cross-referencing of items

3) Data Mining for the Telecommunication Industry

- Multidimensional analysis of telecommunication data
- Fraudulent pattern analysis and the identification of unusual patterns
- Multidimensional association and sequential pattern analysis
- Mobile telecommunication services
- Use of visualization tools in telecommunication data analysis

4) Data Mining for Biological Data Analysis

- Semantic integration of heterogeneous, distributed genomic and proteomic databases
- Alignment, indexing, similarity search, and comparative analysis of multiple nucleotide , protein sequences.
- Discovery of structural patterns and analysis of genetic networks and protein pathways.
- Association and path analysis: identifying co-occurring gene sequences and linking genes to different stages of disease development.
- Visualization tools in genetic data analysis.

5) Data Mining in Scientific Applications

- Scientific data can be amassed at much higher speeds and lower costs.
- This has resulted in the accumulation of huge volumes of high-dimensional data, stream data, and heterogeneous data, containing rich spatial and temporal information.
- Scientific applications are shifting from the “hypothesize-and-test” paradigm toward a “collect and store data, mine for new hypotheses, confirm with data or experimentation” process.

6) Data Mining for Intrusion Detection

- Development of data mining algorithms for intrusion detection
- Association and correlation analysis, and aggregation to help select and build discriminating attributes
- Analysis of stream data
- Distributed data mining
- Visualization and querying tools

7) Trends in Data Mining

- Application exploration
- Scalable and interactive data mining methods
- Integration of data mining with database systems, data warehouse systems, and Webdatabase systems
- Standardization of data mining language
- Visual data mining
- Biological data mining
- Data mining and software engineering
- Web mining
- Distributed data mining
- Real-time or time-critical data mining
- Graph mining, link analysis, and social network analysis
- Multi relational and multi database data mining
- New methods for mining complex types of data
- Privacy protection and information security in data mining

8) Assessment of a Data mining System

1. Data types
2. System issues
3. Data sources
4. Data mining functions and methodologies
5. Coupling data mining with database and/or data warehouse systems.
6. Scalability
7. Visualization tools
8. Data mining query language and graphical user interface

9) Theoretical Foundations of Data Mining

- Data reduction
- Data compression
- Pattern discovery
- Probability theory
- Microeconomic view
- Inductive databases

10) Statistical Data Mining techniques

1. Regression
2. Generalized linear model
3. Analysis of variance
4. mixed effect model
5. Factor analysis
6. Discriminate analysis
7. Time series analysis
8. Survival analysis
9. Quality control

11) Visual and Audio Data Mining

- Visual data mining discovers implicit and useful knowledge from large data sets using data and/or knowledge visualization
- Data visualization and data mining can be integrated in the following ways:
- Data visualization
- Data mining result visualization
- Data mining process visualization
- Interactive visual data mining techniques

12) Security of Data Mining

- Data security enhancing techniques have been developed to help protect data
- Databases can employ a multilevel security model to classify and restrict data according to various security levels, with users permitted access to only their authorized level

- Privacy-sensitive data mining deals with obtaining valid data mining results without learning the underlying data values

Social Impacts of Data Mining

1 . Is Data Mining a Hype or Will It Be Persistent?

- Data mining is a technology
- Technological life cycle
    - Innovators
    - Early Adopters
    - Early Adopters
    - Chasm
    - Early Majority
    - Late Majority
    - Laggards

2. Data Mining: Managers' Business or Everyone's?

- Data mining will surely be an important tool for managers' decision making
    - Bill Gates: "Business @ the speed of thought"
- The amount of the available data is increasing, and data mining systems will be more affordable
- Multiple personal uses
    - Mine your family's medical history to identify genetically-related medical conditions
    - Mine the records of the companies you deal with
    - Mine data on stocks and company performance, etc.
- Invisible data mining
    - Build data mining functions into many intelligent tools

3. Social Impacts: Threat to Privacy and Data Security?

- Is data mining a threat to privacy and data security?
    - "Big Brother", "Big Banker", and "Big Business" are carefully watching you
- Profiling information is collected every time

• credit card, debit card, supermarket loyalty card, or frequent flyer card, or apply for any of the above

• You surf the Web, rent a video, fill out a contest entry form,

• You pay for prescription drugs, or present you medical care number when visiting the

doctor

• Collection of personal data may be beneficial for companies and consumers, there is also potential for misuse

• Medical Records, Employee Evaluations, etc.

4. Protect Privacy and Data Security

1. Fair information practices
   - International guidelines for data privacy protection
   - Cover aspects relating to data collection, purpose, use, quality, openness, individual participation, and accountability
   - Purpose specification and use limitation
   - Openness : Individuals have the right to know what information is collected about them, who has access to the data, and how the data are being used

2. Develop and use data security-enhancing techniques
   - Blind signatures
   - Biometric encryption
   - Anonymous databases

Examples Of Data Mining In Real Life

The importance of data mining and analysis is growing day by day in our real life. Today most organizations use data mining for analysis of Big Data.

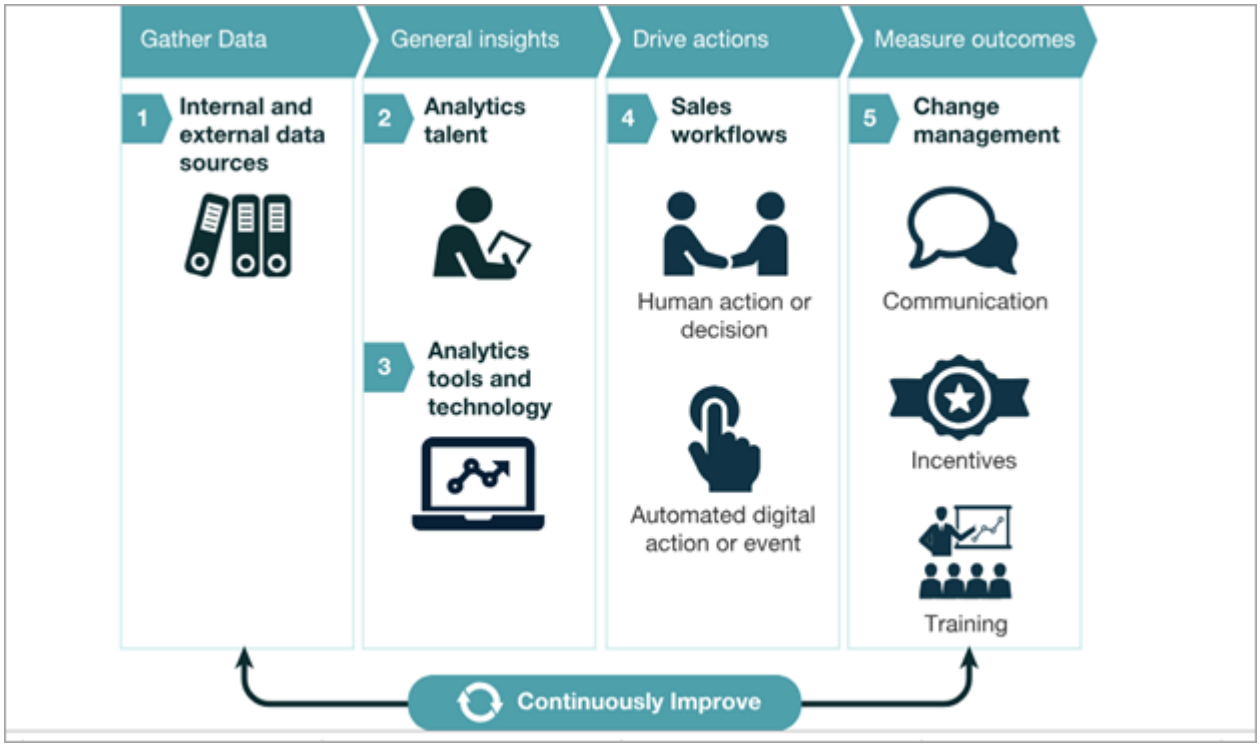
Let us see how these technologies benefit us.

#1) Mobile Service Providers

Mobile service providers use data mining to design their marketing campaigns and to retain customers from moving to other vendors.

From a large amount of data such as billing information, email, text messages, web data transmissions, and customer service, the data mining tools can predict “churn” that tells the customers who are looking to change the vendors.

With these results, a probability score is given. The mobile service providers are then able to provide incentives, offers to customers who are at higher risk of churning. This kind of mining is often used by major service providers such as broadband, phone, gas providers, etc.



#2) Retail Sector

Data Mining helps the supermarket and retail sector owners to know the choices of the customers. Looking at the purchase history of the customers, the data mining tools show the buying preferences of the customers.

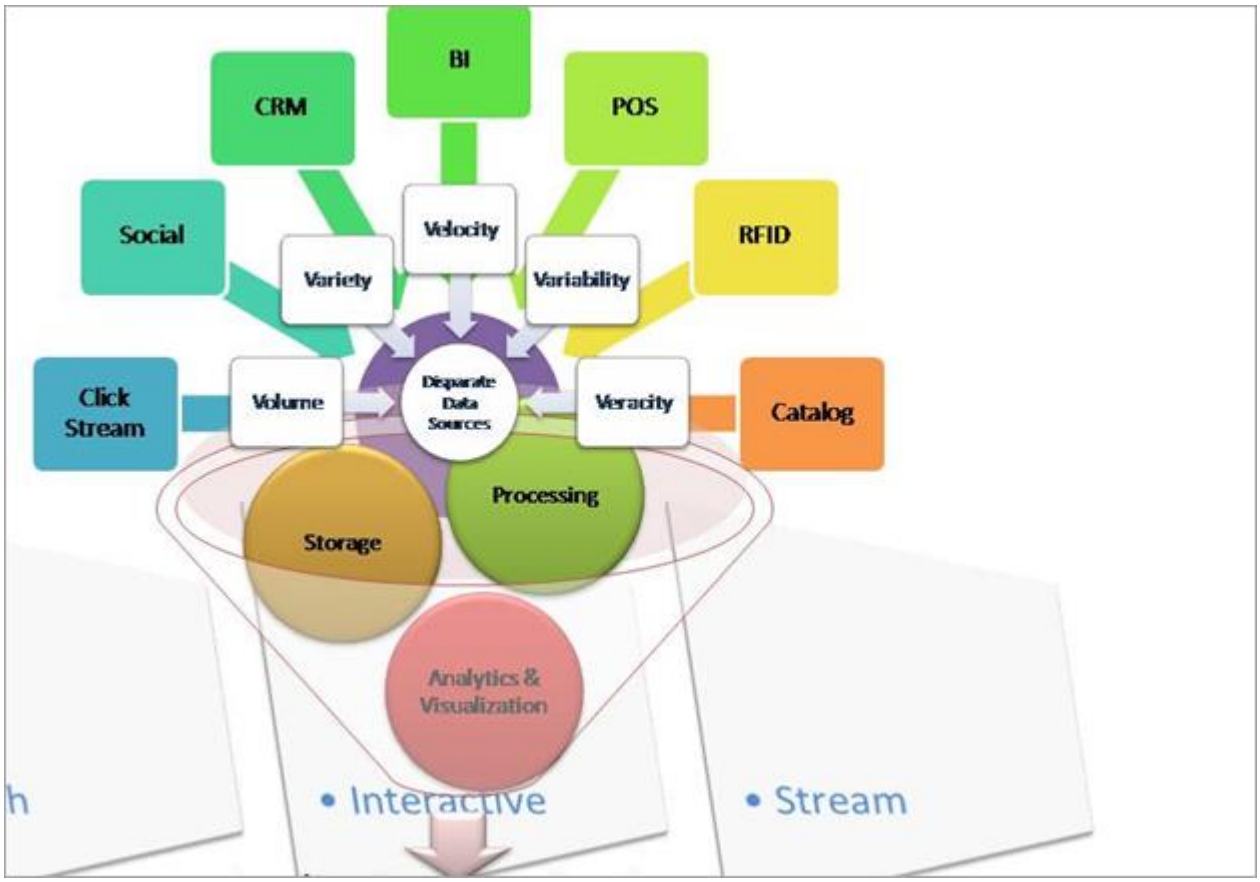
With the help of these results, the supermarkets design the placements of products on shelves and bring out offers on items such as coupons on matching products, and special discounts on some products.

These campaigns are based on RFM grouping. RFM stands for recency, frequency, and monetary grouping. The promotions and marketing campaigns are customized for these

segments. The customer who spends a lot but very less frequently will be treated differently from the customer who buys every 2-3 days but of less amount.

Data Mining can be used for product recommendation and cross-referencing of items.

Data Mining In Retail Sector From Different Data Sources.



#3) Artificial Intelligence

A system is made artificially intelligent by feeding it with relevant patterns. These patterns come from data mining outputs. The outputs of the artificially intelligent systems are also analyzed for their relevance using the data mining techniques.

The recommender systems use data mining techniques to make personalized recommendations when the customer is interacting with the machines. The artificial intelligence is used on mined data such as giving product recommendations based on the past purchasing history of the customer in Amazon.

#4) Ecommerce

Many E-commerce sites use data mining to offer cross-selling and upselling of their products. The shopping sites such as Amazon, Flipkart show “People also viewed”, “Frequently bought together” to the customers who are interacting with the site.

These recommendations are provided using data mining over the purchasing history of the customers of the website.

#5) Science And Engineering

With the advent of data mining, scientific applications are now moving from statistical techniques to using “collect and store data” techniques, and then perform mining on new data, output new results and experiment with the process. A large amount of data is collected from scientific domains such as astronomy, geology, satellite sensors, global positioning system, etc.

Data mining in computer science helps to monitor system status, improve its performance, find out software bugs, discover plagiarism and find out faults. Data mining also helps in analyzing the user feedback regarding products, articles to deduce opinions and sentiments of the views.

#6) Crime Prevention

Data Mining detects outliers across a vast amount of data. The criminal data includes all details of the crime that has happened. Data Mining will study the patterns and trends and predict future events with better accuracy.

The agencies can find out which area is more prone to crime, how much police personnel should be deployed, which age group should be targeted, vehicle numbers to be scrutinized, etc.

#7) Research

Researchers use Data Mining tools to explore the associations between the parameters under research such as environmental conditions like air pollution and the spread of diseases like asthma among people in targeted regions.

#8) Farming

Farmers use Data Mining to find out the yield of vegetables with the amount of water required by the plants.

#9) Automation

By using data mining, the computer systems learn to recognize patterns among the parameters which are under comparison. The system will store the patterns that will be useful in the future to achieve business goals. This learning is automation as it helps in meeting the targets through machine learning.

#10) Dynamic Pricing

Data mining helps the service providers such as cab services to dynamically charge the customers based on the demand and supply. It is one of the key factors for the success of companies.

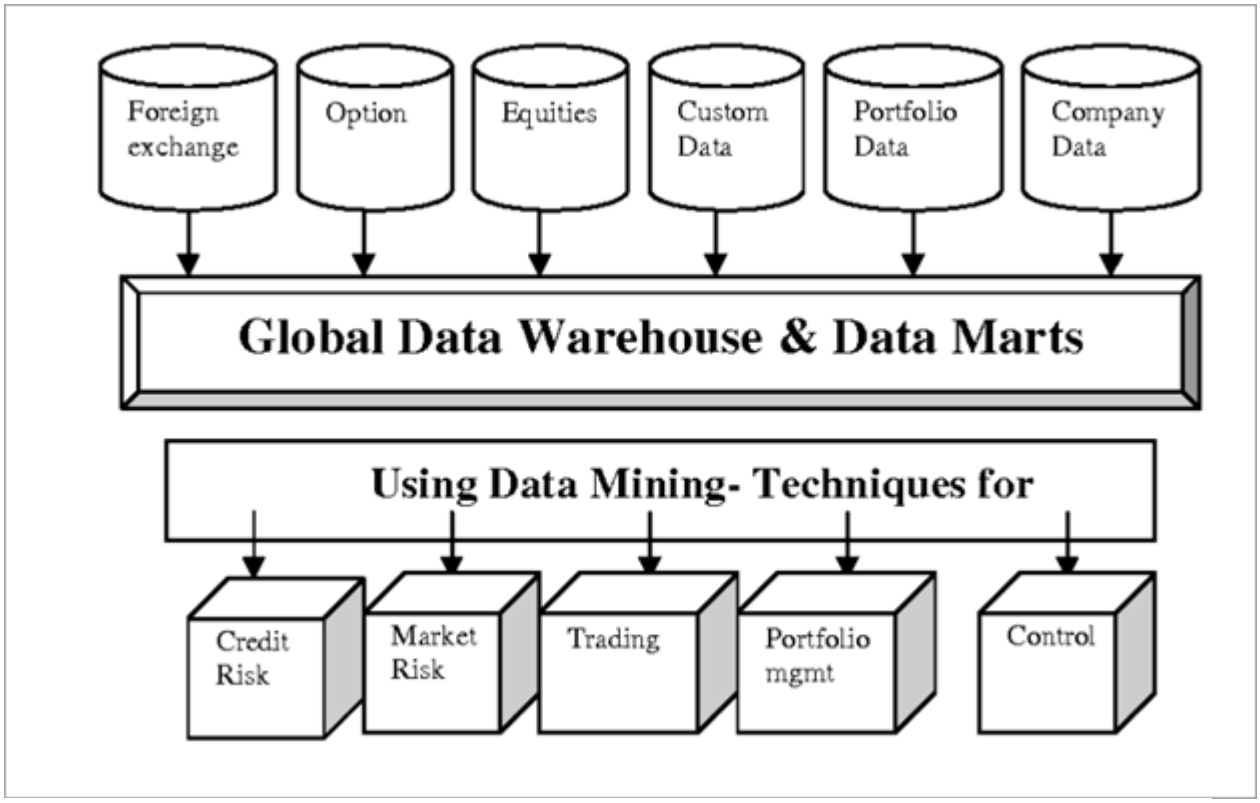
#11) Transportation

Data Mining helps in scheduling the moving of vehicles from warehouses to outlets and analyze the product loading patterns.

#12) Insurance

Data mining methods help in forecasting the customers who buy the policies, analyze the medical claims that are used together, find out fraudulent behaviors and risky customers.

Data Mining Examples In Finance



The finance sector includes banks, insurance companies, and investment companies. These institutions collect a huge amount of data. The data is often complete, reliable and of high quality and demands a systematic data analysis.

To store financial data, data warehouses that store data in the form of data cubes are constructed. To analyze this data, advanced data cube concepts are used. Data mining methods such as clustering and outlier analysis, characterization are used in financial data analysis and mining.

Some cases in finance where data mining is used are given below.

#1) Loan Payment Prediction

Data mining methods like attribute selection and attribute ranking will analyze the customer payment history and select important factors such as payment to income ratio, credit history, the term of the loan, etc. The results will help the banks decide its loan granting policy, and also grant loans to the customers as per factor analysis.

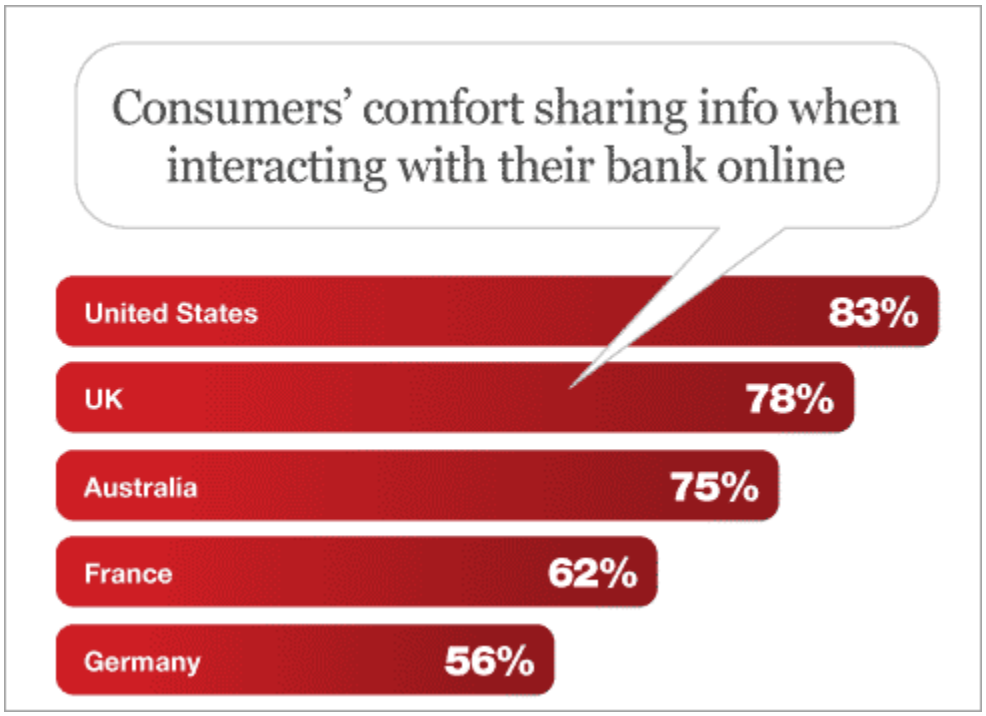
#2) Targeted Marketing

Clustering and classification data mining methods will help in finding the factors that influence the customer's decisions towards banking. Similar behavioral customers' identification will facilitate targeted marketing.

#3) Detect Financial Crimes

Banking data come from many different sources, various cities, and different bank locations. Multiple data analysis tools are deployed to study and to detect unusual trends like big value transactions. Data visualization tools, outlier analysis tools, clustering tools, etc are used to identify the relationships and patterns of action.

The figure below is a study from Infosys showing the customer's willingness to banking online system in different countries. Infosys used Big Data Analytics for this study.



Applications Of Data Mining In Marketing

Data mining boosts the company's marketing strategy and promotes business. It is one of the key factors for the success of companies. A huge amount of data is collected on sales, customer shopping, consumption, etc. This data is increasing day by day due to e-commerce.

Data mining helps to identify customer buying behavior, improve customer service, focus on customer retention, enhance sales, and reduce the cost of businesses.

Some examples of data mining in marketing are:

#1) Forecasting Market

To predict the market, the marketing professionals will use Data Mining techniques like regression to study customer behavior, changes, and habits, customer response and other factors like marketing budget, other incurring costs, etc. In the future, it will be easier for professionals to predict the customers in case of any factor changes.

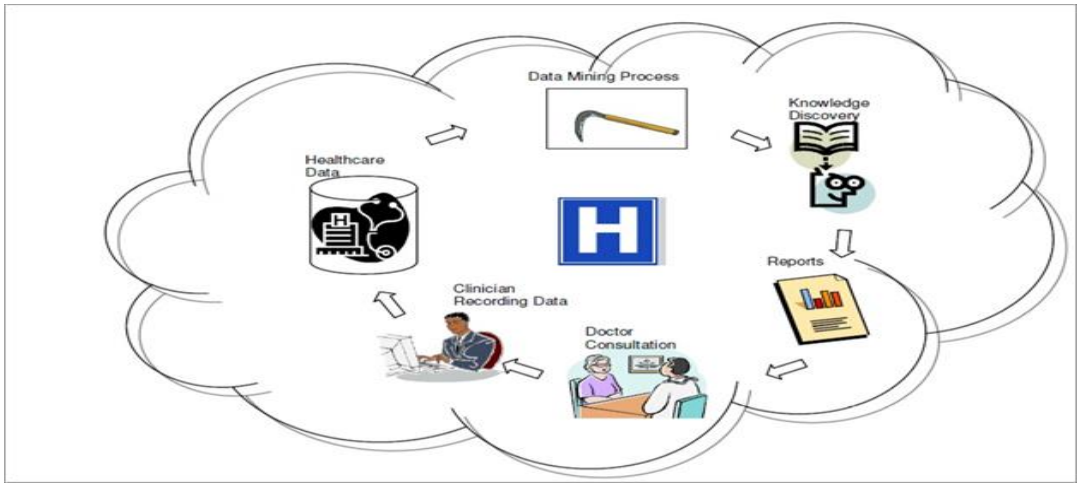
#2) Anomaly Detection

Data mining techniques are deployed to detect any abnormalities in data that may cause any kind of flaw in the system. The system will scan thousands of complex entries to perform this operation.

#3) System Security

Data Mining tools detect intrusions that may harm the database offering greater security to the entire system. These intrusions may be in the form of duplicate entries, viruses in the form of data by hackers, etc.

Examples Of Data Mining Applications In Healthcare



In healthcare, data mining is becoming increasingly popular and essential.

Data generated by healthcare is complex and voluminous. To avoid medical fraud and abuse, data mining tools are used to detect fraudulent items and thereby prevent loss.

Some data mining examples of the healthcare industry are given below for your reference.

#1) Healthcare Management

The data mining method is used to identify chronic diseases, track high-risk regions prone to the spread of disease, design programs to reduce the spread of disease. Healthcare professionals will analyze the diseases, regions of patients with maximum admissions to the hospital.

With this data, they will design the campaigns for the region to make people aware of the disease and see how to avoid it. This will reduce the number of patients admitted to hospitals.

#2) Effective Treatments

Using data mining, the treatments can be improved. By continuous comparison of symptoms, causes, and medicines, data analysis can be performed to make effective treatments. Data mining is also used for the treatment of specific diseases, and the association of side-effects of treatments.

#3) Fraudulent And Abusive Data

Data mining applications are used to find abnormal patterns such as laboratory, physician's results, inappropriate prescriptions, and fraudulent medical claims.

Data Mining And Recommender Systems

Recommender systems give customers with product recommendations that may be of interest to the users.

The recommended items are either similar to the items queried by the user in the past or by looking at the other customer preferences which have similar taste as the user. This approach is called a content-based approach and a collaborative approach appropriately.

Many techniques like information retrieval, statistics, machine learning, etc are used in recommender systems.

Recommender systems search for keywords, user profiles, user transactions, common features among items to estimate an item for the user. These systems also find the other users who have a similar history of buying and predict items that those users could buy.

There are many challenges in this approach. The recommendation system needs to search through millions of data in real-time.

There are two types of errors made by Recommender Systems:

False negatives and False positives.

False negatives are products that were not recommended by the system but the customer would want them. **False-positive** are products that were recommended by the system but not wanted by the customer. Another challenge is the recommendation for the users who are new without any purchasing history.

An intelligent query answering technique is used to analyze the query and provide generalized, associated information relevant to the query. **For Example:** Showing the review of restaurants instead of just the address and phone number of the restaurant searched for.

Data Mining For CRM (Customer Relationship Management)

Customer Relationship Management can be reinforced with data mining. Good customer Relations can be built by attracting more suitable customers, better cross-selling and up-selling, better retention.

Data Mining can enhance CRM by:

1. Data mining can help businesses create targeted programs for higher response and better ROI.
2. Businesses can offer more products and services as desired by the customers through up-selling and cross-selling thereby increasing customer satisfaction.
3. With data mining, a business can detect which customers are looking for other options. Using that information companies can build ideas to retain the customer from leaving.

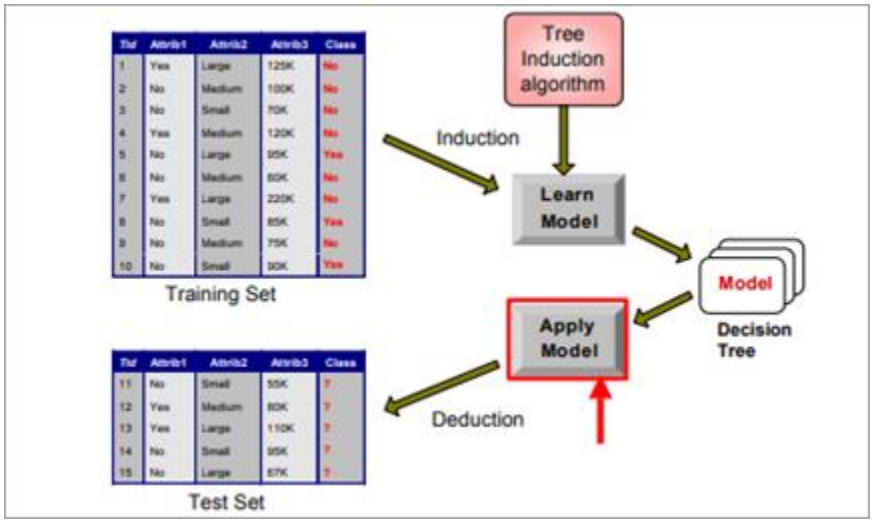
Data Mining helps CRM in:

1. **Database Marketing:** Marketing software enables companies to send messages and emails to customers. This tool along with data mining can do targeted marketing. With data mining, automation, and scheduling of jobs can be performed. It helps in better decision making. It will also help in technical decisions as to what kind of customers are interested in a new product, which market area is good for product launching.
2. **Customer Acquisition Campaign:** With data mining, the market professional will be able to identify potential customers who are unaware of the products or new buyers. They will be able to design the offers and initiatives for such customers.
3. **Campaign Optimization:** Companies use data mining for the effectiveness of the campaign. It can model customer responses to marketing offers.

Data Mining Using Decision Tree Example

Decision tree algorithms are called CART(Classification and Regression Trees). It is a supervised learning method. A tree structure is built on the features chosen, conditions for splitting and when to stop. Decision trees are used to predict the value of class variables based on learning from the previous training data.

The internal node represents an attribute and the leaf node represents a class label.



Following steps are used to build a Decision Tree Structure:

1. Place the best attribute at the top of the tree (root).
2. Subsets are created in such a way that each subset represents data with the same value for an attribute.
3. Repeat the same steps to find the leaf nodes of all branches.

To predict a class label, the record's attribute is compared with the root of the tree. On comparing, the next branch is chosen. The internal nodes are also compared in the same way until the leaf node reached predicts the class variable.

Some algorithms used for Decision Tree Induction include Hunt's Algorithm, CART, ID3, C4.5, SLIQ, and SPRINT.

Most Popular Example Of Data Mining: Marketing And Sales

Marketing and Sales are the domains in which companies have large volumes of data.

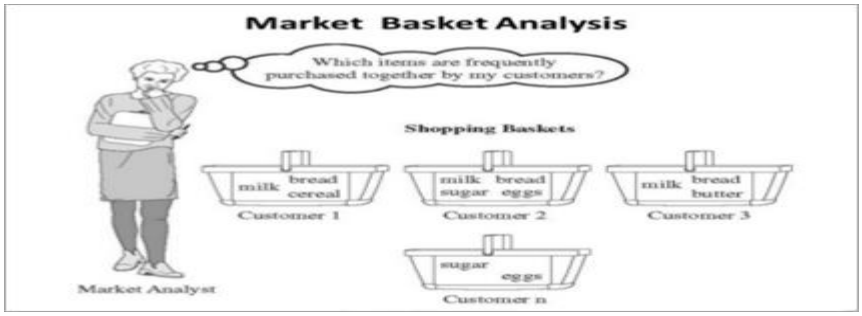
#1) Banks are the first users of data mining technology as it helps them with credit assessment. Data mining analyzes what services offered by banks are used by customers, what type of customers use ATM cards and what do they generally buy using their cards (for cross-selling).

Banks use data mining to analyze the transactions which the customer do before they decide to change the bank to reduce customer attrition. Also, some outliers in transactions are analyzed for fraud detection.

#2) Cellular Phone Companies use data mining techniques to avoid churning. Churning is a measure showing the number of customers leaving the services. It detects patterns that show how customers can benefit from the services to retain customers.

#3) Market Basket Analysis is the technique to find the groups of items that are bought together in stores. Analysis of the transactions show the patterns such as which things are bought together often like bread and butter, or which items have higher sales volume on certain days such as beer on Fridays.

This information helps in planning the store layouts, offering a special discount to the items that are less in demand, creating offers such as “buy 2 get 1 free” or “get 50% on second purchase” etc.



Big Companies Using Data Mining

Some online companies using data mining techniques are given below:

- **AMAZON:** Amazon uses Text Mining to find the lowest price of the product.
- **MC Donald’s:** McDonald’s uses big data mining to enhance its customer experience. It studies the ordering pattern of customers, waiting times, size of orders, etc.
- **NETFLIX:** Netflix finds out how to make a movie or a series popular among the customers using its data mining insights.

Conclusion

Data mining is used in diverse applications such as banking, marketing, healthcare, telecom industries, and many other areas.Data mining techniques help companies to gain knowledgeable information, increase their profitability by making adjustments in processes

and operations. It is a fast process which helps business in decision making through analysis of hidden patterns and trends.

QUESTIONS

PART A

1. What do you infer from the word cluser analysis? Highlight various types of Data in Cluster Analysis.
2. Compare and Contrast K- Means and K – Medoids.
3. Which is better? Agglomerative or Divisive Hierarchical Clustering. Illustrate.
4. Highlight the importance of Neural Network Approach in Clustering.
5. How do cluster high dimensional Data. Elaborate.
6. Comment on Constraint Based Cluster Analysis.
7. Justify the essence of outlier analysis.
8. Compare various clustering techniques.
9. How do you categorize Major Clustering Methods.
10. Illustrate Grid Based Clustering Methods.

PART B

1. Justify the need of Clustering Technique.
2. How can the data for a variable be standardized?
3. Compare Manhattan and Euclidean Distance.
4. Highlight various clustering techniques.
5. What do you infer from the term “Density based cluster”.
6. Comment on STING.

TEXT/REFERENCE BOOKS

1. Jiawei Han and Micheline Kamber, “Data Mining Concepts and Techniques”, 2nd Edition, Elsevier 2007.
2. Alex Berson and Stephen J. Smith, “Data Warehousing, Data Mining & OLAP”, Tata McGrew Hill, 2007.
3. www.tutorialspoint.com/dwh/dwh.overview.htm
4. http://study.com/academy/lesson/data-warehousing-and-data-minig-information-for-business-intelligence.html
5. http://www.dei.unpd_it/-capt/SVMATERIALE/DWDM0405.pdf
6. Data mining Concepts and Techniques, Book by Jewei Han and Kamber.