19CAC16 – DATA MINING AND DATA WAREHOUSING

Prepared by

S.NITHYANANTH, ASP

DEPARTMENT OF MCA

MUTHAYAMMAL ENGINEERING COLLEGE,RASIPURAM.

OBJECTIVES

- To gain knowledge on data mining and the need for pre-processing.
- To characterize the kinds of patterns that can be discovered by association rule mining.
- To implement classification techniques on large datasets.
- To analyze various clustering techniques in real world applications.
- To get exposed to the concepts of data warehousing architecture and implementation.

UNIT I DATA MINING & DATA PREPROCESSING

Data Mining–Concepts , DBMS versus Data mining , kinds of Data, Applications, Issues and Challenges–Need for Data Pre-processing – Data Cleaning – Data Integration and Transformation – Data Reduction – Data Discretization and Concept Hierarchy Generation.

UNIT II ASSOCIATION RULE MINING AND CLASSIFICATION BASICS

Introduction to Association rules – Association Rule Mining – Mining Frequent Itemsets with and without Candidate Generation – Mining Various Kinds of Association Rules - Classification versus Prediction – Data Preparation for Classification and Prediction.

UNIT III
CLASSIFICATION AND PREDICTION TECHNIQUES

Classification by Decision Tree – Bayesian Classification – Rule Based Classification – Bayesian Belief Networks – Classification by Back Propagation – Support Vector Machines – K-Nearest Neighbor Algorithm –Linear Regression, Nonlinear Regression, Other Regression-Based Methods.

UNIT IV
CLUSTERING TECHNIQUES

Cluster Analysis – Partitioning Methods: k-Means and k- Mediods – Hierarchical Methods: Agglomerative and Divisive – Density–Based Method: DBSCAN – Model Based Clustering Methods- Clustering High Dimensional Data - Outlier Analysis.

UNIT V DATA WAREHOUSE

Need for Data Warehouse – Database versus Data Warehouse – Multidimensional Data Model – Schemas for Multidimensional Databases – OLAP operations – OLAP versus OLTP– Data Warehouse Architecture .

OUTCOMES

- Identify data mining techniques in building intelligent model.
- Illustrate association mining techniques on transactional databases.
- Apply classification and clustering techniques in real world applications.
- Evaluate various mining techniques on complex data objects.
- Design, create and maintain data warehouses.

REFERENCES

1. Daniel T. Larose, Chantal D. Larose, "Data mining and Predictive analytics," Second Edition, Wiley Publication, 2015.
2. G. K. Gupta, “Introduction to Data Mining with Case Studies”, Eastern Economy Edition, Prentice Hall of India, Third Edition, 2014.

UNIT – I

DATA MINING & DATA PREPROCESSING

OVERVIEW

- Data Mining–Concepts
- DBMS versus Data Mining
- Kinds of Data in Data Mining
- Applications of Data Mining
- Issues in Data Mining
- Challenges in Data Mining
- Need for Data Pre-processing
- Data Cleaning
- Data Integration
- Data Transformation
- Data Reduction
- Data Discretization and Concept Hierarchy Generation.

Data Mining–Concepts

Data Mining Definition :

Data mining is a process of **extracting and discovering patterns in large data sets.**

- **Data mining (Knowledge Discovery from Data)**
  - Extraction of interesting **(<u>non-trivial,</u> <u>implicit,</u> <u>previously unknown</u> and <u>potentially useful)</u>** patterns or knowledge from huge amount of data.
- **Alternative names :**
  - **Knowledge Discovery in Databases (KDD),** knowledge extraction, data/pattern analysis, business intelligence, etc.

Data Mining

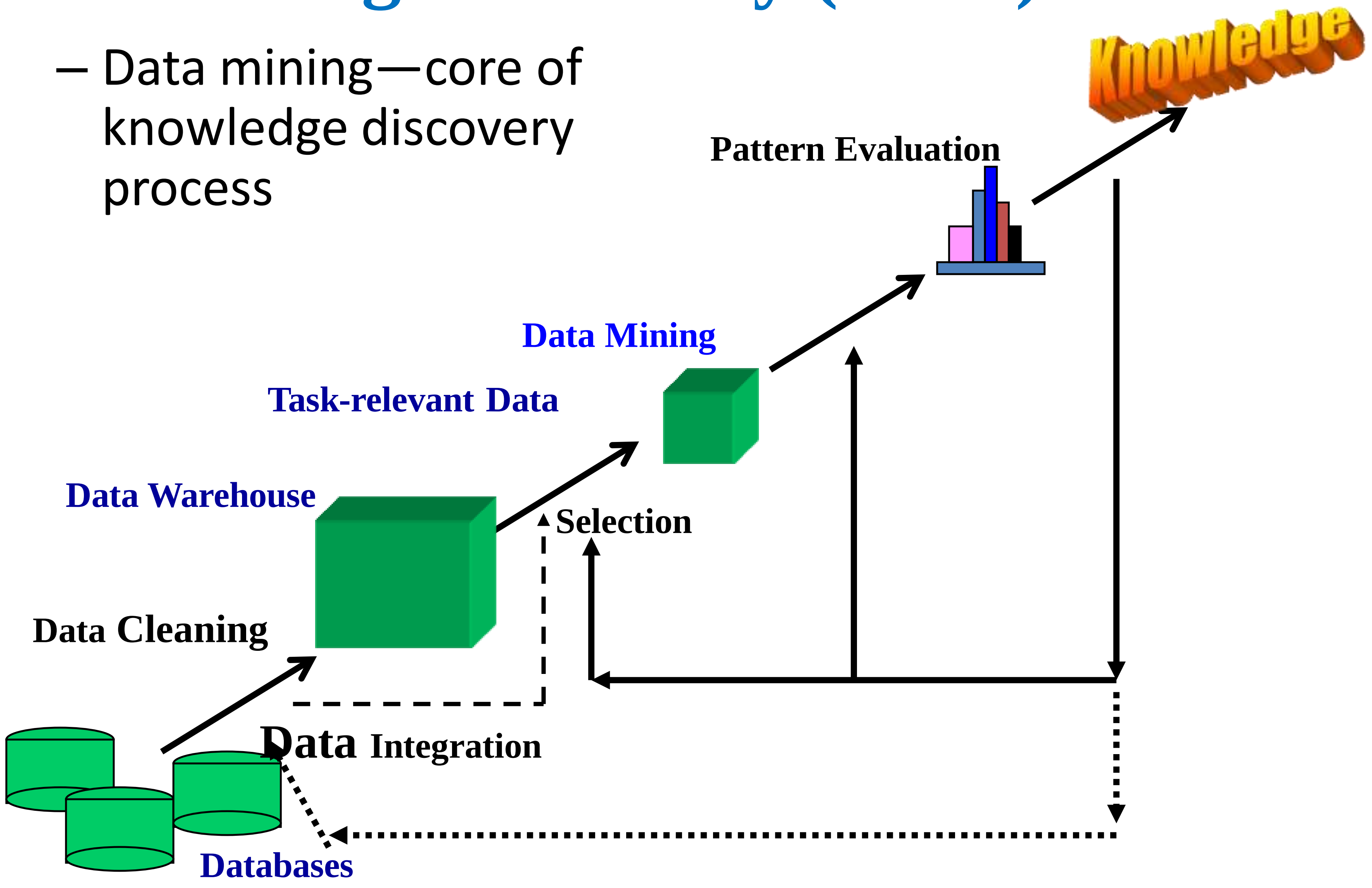
- Process of sorting through large amounts of data and picking out relevant information
- Process of analyzing data from different perspectives and summarizing it into useful information
- Discovering hidden value in database
- It is non-trivial process of identifying valid, novel, useful and understandable patterns in data
- Extracting or mining knowledge from large amounts of data

Advantages of Data Mining :

- Increasing revenue.
- Understanding customer segments and preferences.
- Acquiring new customers.
- Improving cross-selling and up-selling.
- Detecting fraud.
- Identifying credit risks.
- Monitoring operational performance.

Knowledge Discovery (KDD) Process

– Data mining—core of knowledge discovery process



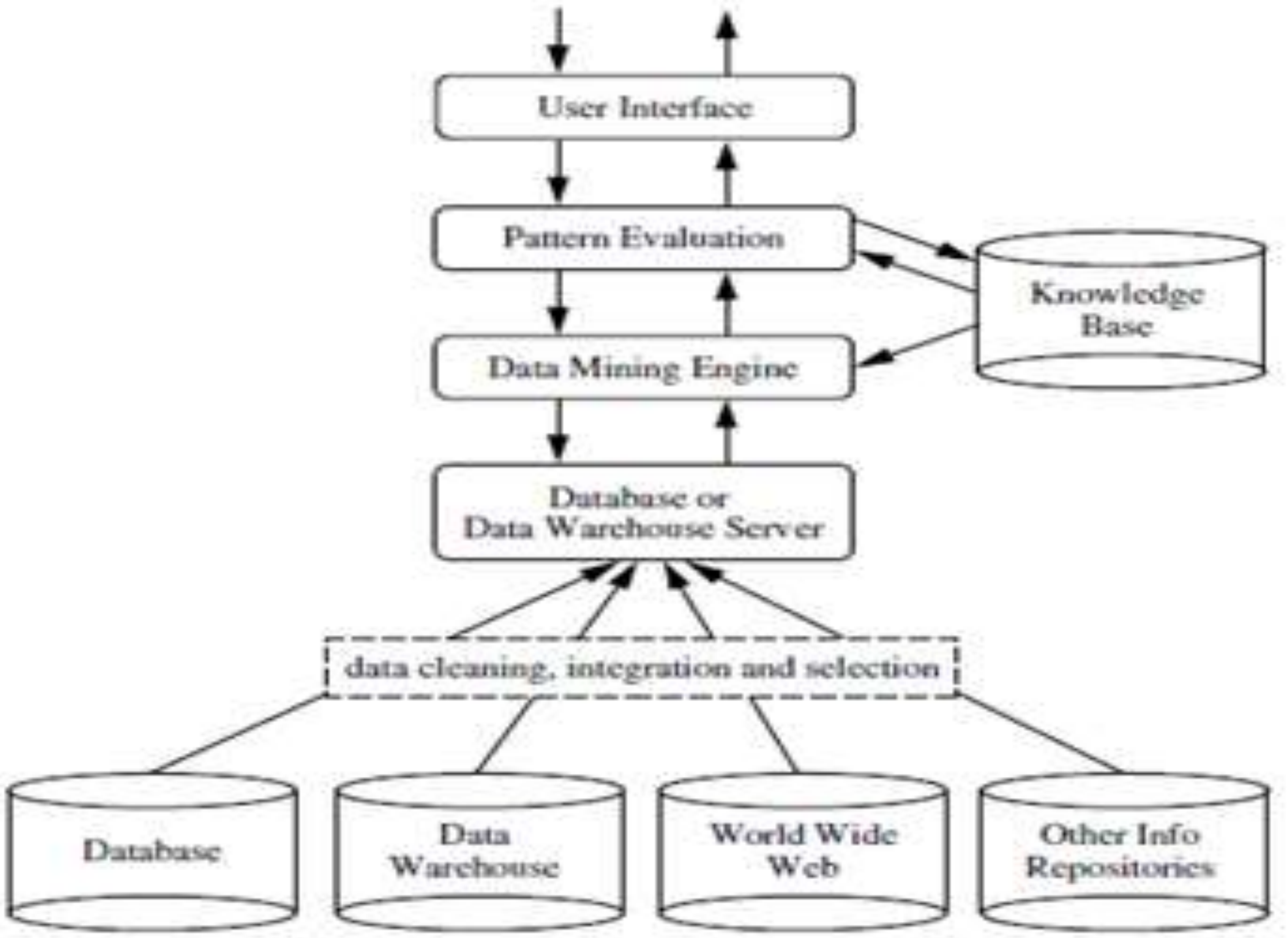
KDD Process Steps

1. Goal-Setting and Application Understanding.
2. Data Selection and Integration.
3. Data Cleaning and Preprocessing.
4. Data Transformation.
5. Data Mining.
6. Pattern Evaluation/Interpretation.
7. Knowledge Discovery and Use.

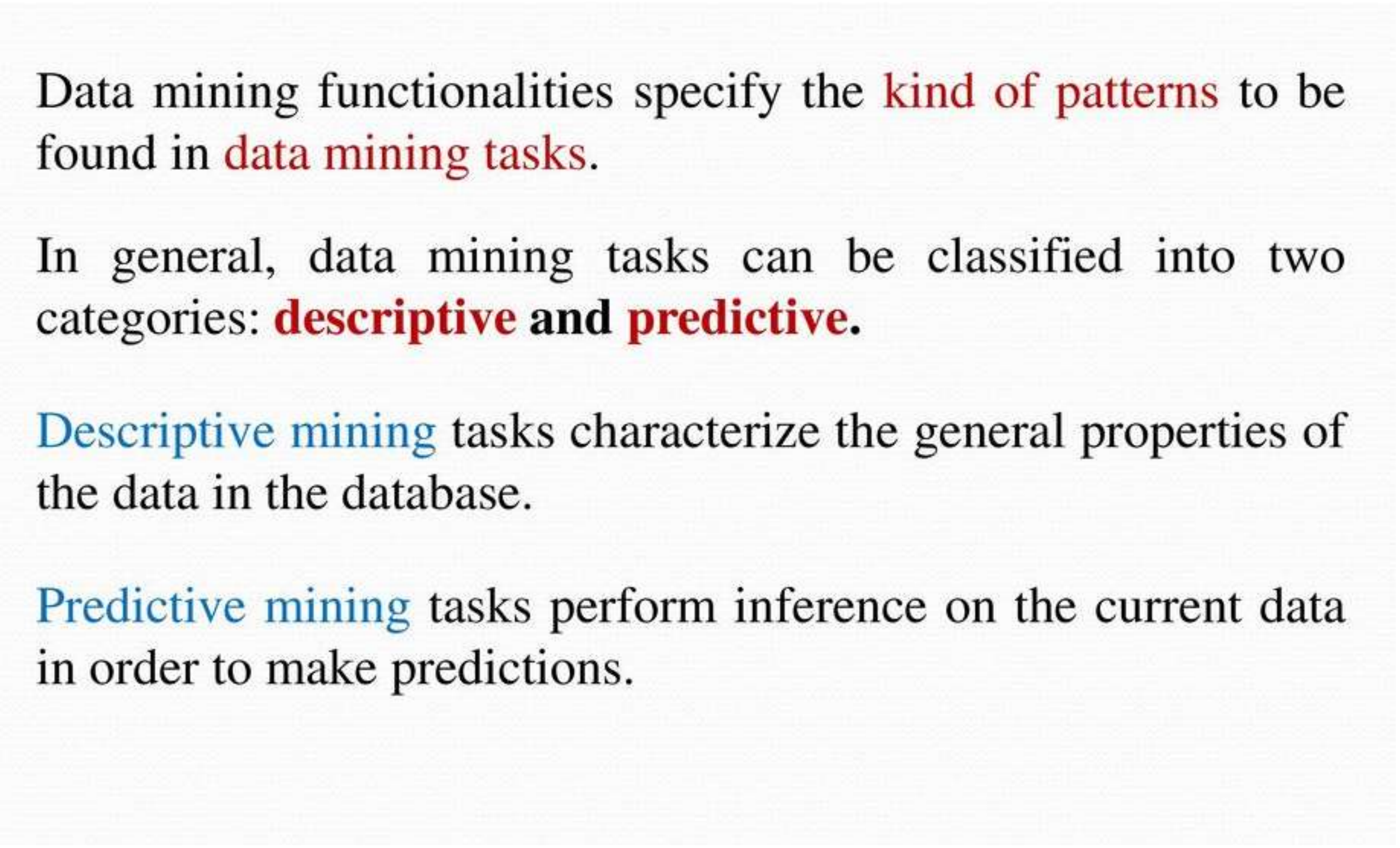
KDD Process: Several Key Elements

- Learning the application domain
- Creating a target data set
- Data cleaning and preprocessing
- Data reduction and transformation
- Choosing functions of data mining
  - summarization, classification, association, clustering
- Choosing the mining algorithm(s)
- Pattern evaluation and knowledge presentation
- Use of discovered knowledge

Architecture of Data Mining System



Data Mining Functionalities



Data mining functionalities 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.

Data mining functionalities, and the kinds of patterns they can discover, are:

Concept/Class Description: Characterization and Discrimination

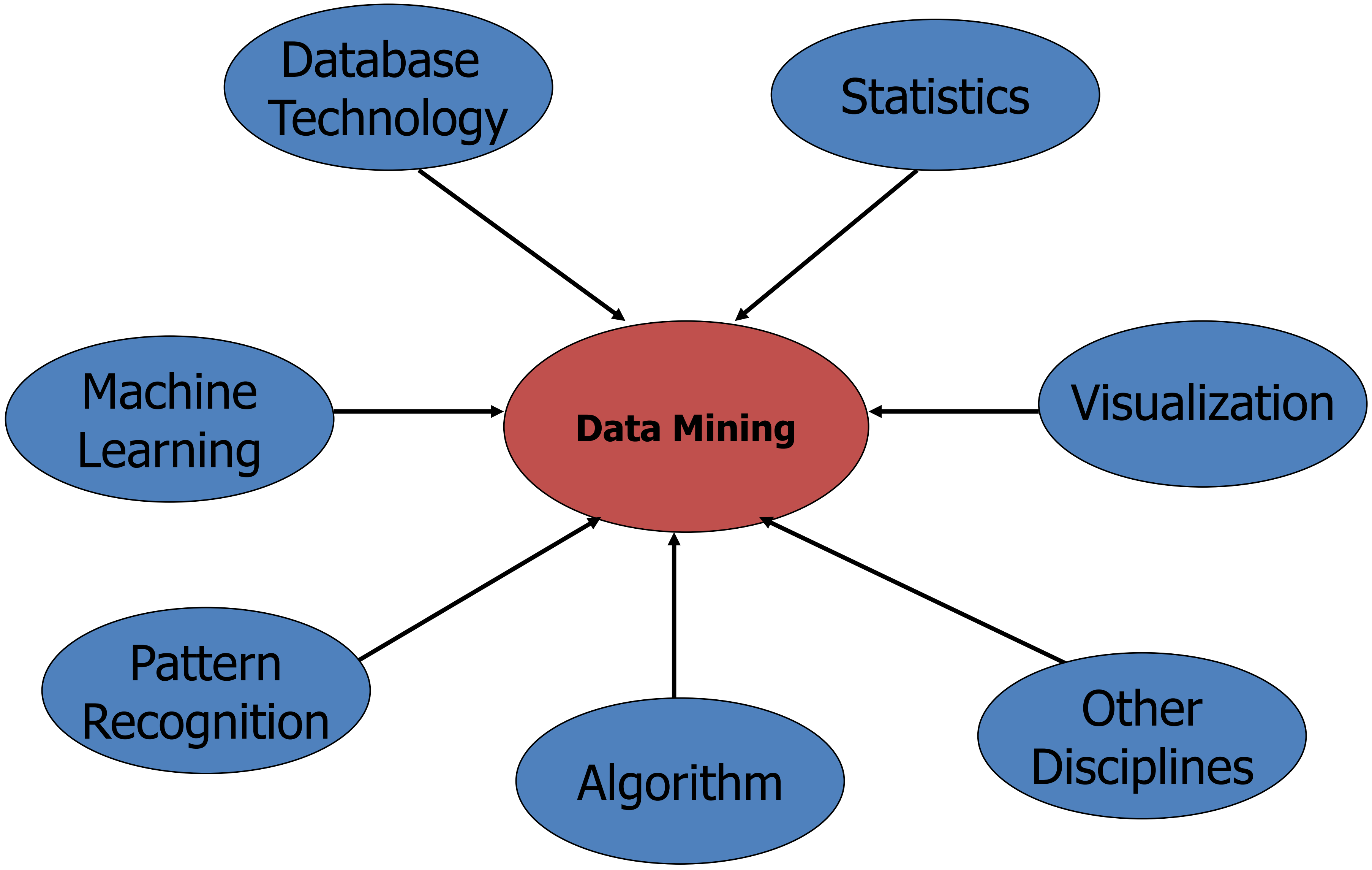
Mining Frequent Patterns, Associations, and Correlations

Classification and Prediction

Cluster Analysis

Outlier Analysis

Data Mining: Confluence of Multiple Disciplines



Evolution of Database Technology

- **1960s:**
  - Data collection, database creation and network DBMS
- **1970s:**
  - Relational data model, relational DBMS implementation
- **1980s:**
  - Application-oriented DBMS (spatial, scientific, engineering, etc.)

- **1990s:**
  - Data mining, data warehousing, multimedia databases, and Web databases
- **2000s**
  - Stream data management and mining
  - Data mining and its applications
  - Web technology (XML, data integration) and global information systems

DBMS versus Data Mining

DBMS	Data Mining
The database is the organized collection of data. data are stored in very large databases.	Data mining is analyzing data from different information to discover useful knowledge.
A Database may contain different levels of abstraction in its architecture.	Data mining deals with extracting useful and previously unknown information from raw data.
Typically, the three levels: external, conceptual and internal make up the database architecture.	The data mining process relies on the data compiled in the data warehousing phase in order to detect meaningful patterns.

Data Mining: On What Kinds of Data?

- **Database-Oriented Data Sets and Applications :**
  - Relational database, data warehouse, transactional database
- **Advanced Data sets and Advanced Applications :**
  - Time-series data, sequence data
  - Object-relational databases
  - Heterogeneous databases
  - Multimedia database
  - Text databases
  - The World-Wide Web

Kinds of Data in Data Mining

- **Relational databases**
- **Data warehouses**
- **Transactional Databases**
- **Advanced database systems**

  Object-relational

  Spacial and Temporal □

  Time-series □

  Multimedia Data Mining

  Text Mining□

  Web Mining □

Applications of Data Mining

- 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.

Data Mining Applications :

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

1. **Financial Data Analysis**
2. **Retail Industry**
3. **Telecommunication Industry**
4. **Biological Data Analysis**
5. **Other Scientific Applications**
6. **Intrusion Detection**

1. Financial Data Analysis :

- The financial data in banking and financial industry is generally reliable and of high quality which facilitates systematic data analysis and data mining.

1. **Design and construction of data warehouses for multidimensional data analysis and data mining.**
2. **Loan payment prediction and customer credit policy analysis.**
3. **Classification and clustering of customers for targeted marketing.**

2.Retail Industry :

- Data Mining has its great application in Retail Industry because it collects large amount of data from on sales, customer purchasing history, goods transportation, consumption and services.

1. **Design and Construction of data warehouses based on the benefits of data mining.**
2. **Multidimensional analysis of sales, customers, products, time and region.**
3. **Customer Retention.**
4. **Product recommendation and cross-referencing of items.**

3. Telecommunication Industry :

- The telecommunication industry is one of the most emerging industries providing various services such as fax, pager, cellular phone, internet messenger, images, e-mail, web data transmission, etc.

1.**Multidimensional Analysis of Telecommunication data.**

2. Identification of unusual patterns.

3. Multidimensional association and sequential patterns analysis.

4. Mobile Telecommunication services.

4. Biological Data Analysis :

- Biological data mining is a very important part of Bioinformatics.

1. **Semantic integration of heterogeneous databases.**
2. **Alignment, indexing, similarity search and comparative analysis.**
3. **Discovery of structural patterns and analysis of genetic networks.**
4. **Association and path analysis.**
5. **Visualization tools in genetic data analysis.**

5. Other Scientific Applications :

- Following are the applications of data mining in the field of Scientific Applications –

1. **Data Warehouses and Data Preprocessing.**
2. **Graph-based mining.**
3. **Visualization and domain specific knowledge.**

6. Intrusion Detection :

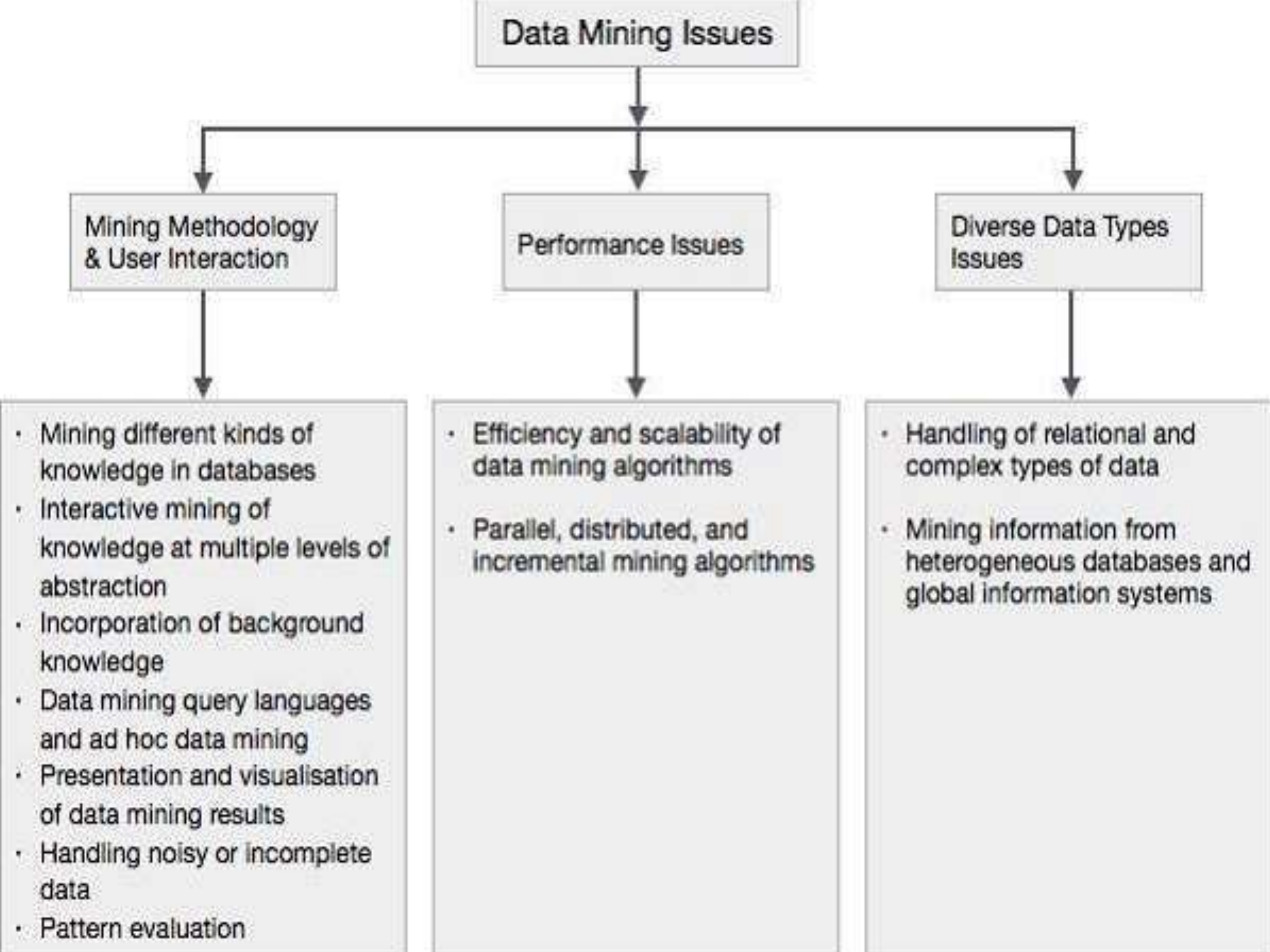
- Intrusion refers to any kind of action that threatens **integrity, confidentiality, or the availability** of network resources.

1. **Development of data mining algorithm for intrusion detection.**
2. **Association and correlation analysis, aggregation to help select and build discriminating attributes.**
3. **Analysis of Stream data.**
4. **Distributed data mining.**
5. **Visualization and query tools.**

Issues in Data Mining

- Data mining is not an easy task, as the algorithms used can get very complex and data is not always available at one place.
- It needs to be integrated from various **heterogeneous data sources.**
- These factors also create some issues. **The major issues are –**

1. **Mining Methodology and User Interaction.**
2. **Performance Issues.**
3. **Diverse Data Types Issues.**


Data Mining Issues
Mining Methodology & User Interaction
Performance Issues
Diverse Data Types Issues
· 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
· Presentation and visualisation of data mining results
· Handling noisy or incomplete data
· Pattern evaluation
· Efficiency and scalability of data mining algorithms
· Parallel, distributed, and incremental mining algorithms
· Handling of relational and complex types of data
· Mining information from heterogeneous databases and global information systems

1.Mining Methodology and User Interaction :

It refers to the following kinds of issues –

- **Mining different kinds of knowledge in Databases** – Different users may be interested in different kinds of knowledge.
- **Interactive mining of knowledge at multiple levels of abstraction** – The data mining process needs to be interactive because it allows users to focus the search for patterns.
- **Incorporation of background knowledge** – Background knowledge may be used to express the discovered patterns not only in concise terms but at multiple levels of abstraction.

- **Data mining query languages and ad hoc data mining –** Data Mining Query language that allows the user to describe ad hoc mining tasks, should be integrated with a data warehouse query language.
- **Presentation and visualization of data mining results –** Once the patterns are discovered it needs to be expressed in high level languages, and visual representations.
- **Handling noisy or incomplete data –** The data cleaning methods are required to handle the noise and incomplete objects while mining the data regularities.
- **Pattern evaluation –** The patterns discovered should be interesting because either they represent common knowledge or lack novelty.

2. Performance Issues :

- **Efficiency and scalability of data mining algorithms** – In order to effectively extract the information from huge amount of data in databases, data mining algorithm must be efficient and scalable.
- **Parallel, distributed, and incremental mining algorithms** – The factors such as huge size of databases, wide distribution of data, and complexity of data mining methods motivate the development of parallel and distributed data mining algorithms.

3. Diverse Data Types Issues :

- **Handling of relational and complex types of data –** The database may contain complex data objects, multimedia data objects, spatial data, temporal data etc. It is not possible for one system to mine all these kind of data.
- **Mining information from heterogeneous databases and global information systems –** The data is available at different data sources on LAN or WAN. These data source may be structured, semi structured or unstructured.

Challenges in Data Mining

- Some of the Data Mining challenges are :

1. **Security and Social Challenges**
2. **Noisy and Incomplete Data**
3. **Distributed Data**
4. **Complex Data**
5. **Performance**
6. **Scalability and Efficiency of the Algorithms**
7. **Improvement of Mining Algorithms**
8. **Incorporation of Background Knowledge**

9. Data Visualization
10. Data Privacy and Security
11. User Interface
12. Mining dependent on Level of Abstraction
13. Integration of Background Knowledge
14. Mining Methodology Challenges

Data Pre-processing

Data Pre-processing :

- Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining.
- Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.
- Data in the real world is dirty.

Incomplete: Lacking attribute values, lacking certain attributes of interest, or containing only aggregate data.

Noisy: Containing Random errors or outliers.

Inconsistent: Containing discrepancies in codes or names.

Need for Data Pre - processing :

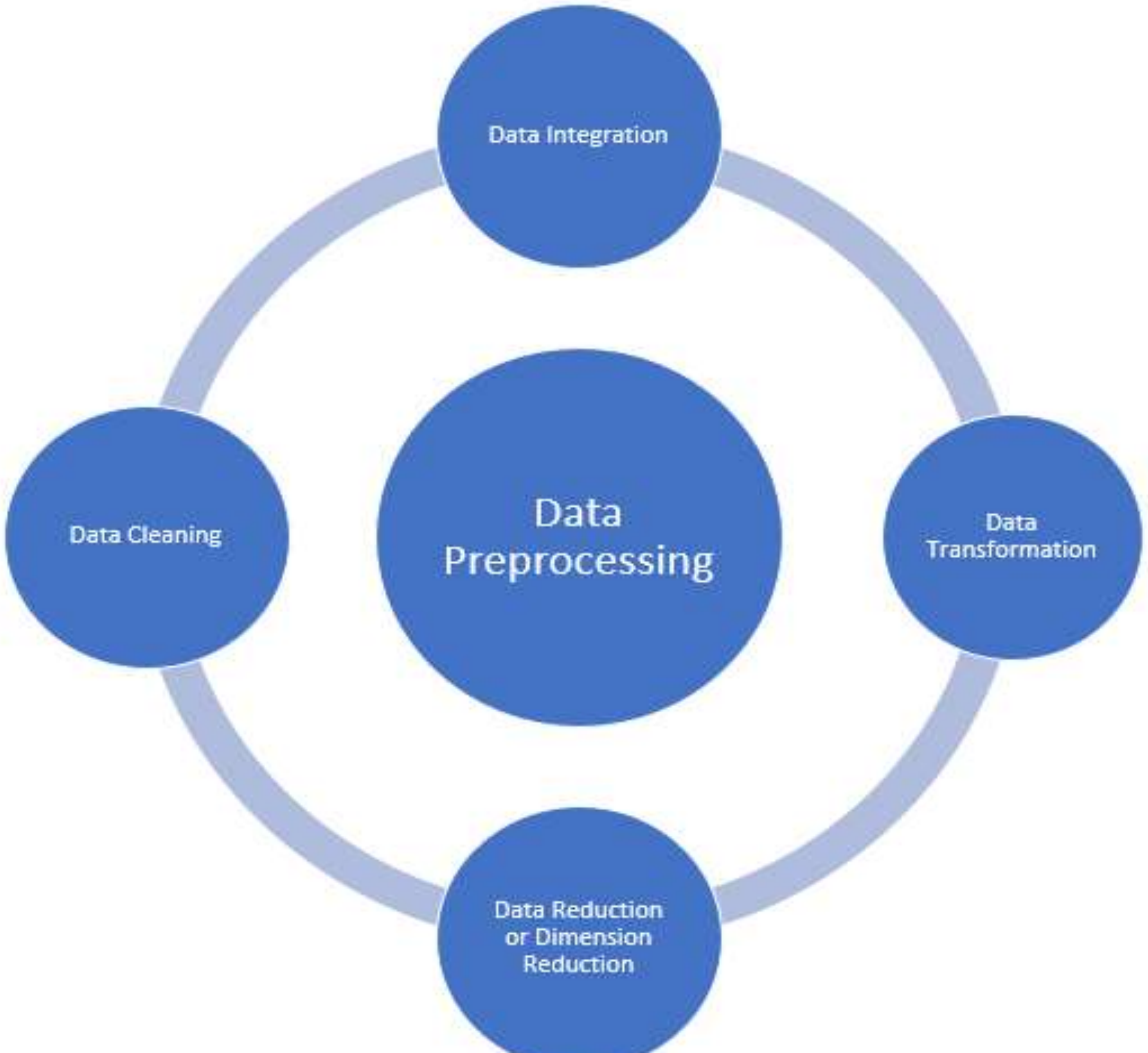
- **Yes Data Pre - Processing is need to check the data quality.**
- **The quality can be checked by the following :**

1. **Accuracy:** To check whether the data entered is correct or not.
2. **Completeness:** To check whether the data is available or not recorded.
3. **Consistency:** To check whether the same data is kept in all the places that do or do not match.
4. **Timeliness:** The data should be updated correctly.
5. **Believability:** The data should be trustable.
6. **Interpretability:** The understandability of the data.

Data Pre - processing Techniques

- The following **Four Types** of Data Pre - processing Techniques are :

1. Data Cleaning
2. Data Integration
3. Data Transformation
4. Data Reduction


Data Integration
Data Cleaning
Data Preprocessing
Data Transformation
Data Reduction or Dimension Reduction

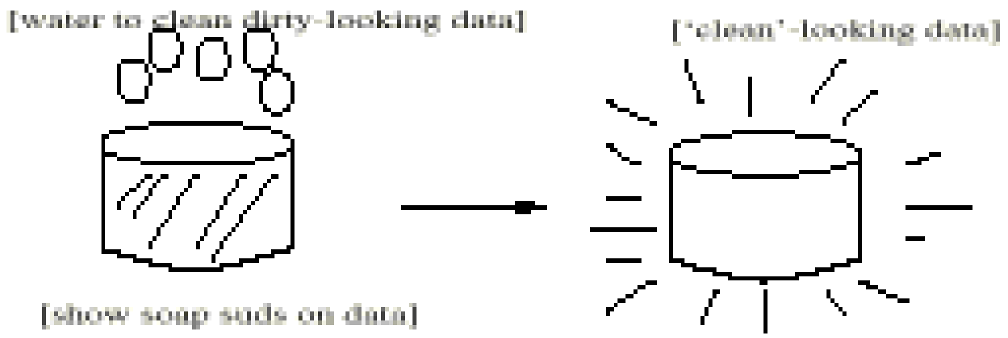
Major Tasks in Data Preprocessing

Major Tasks in Data Preprocessing

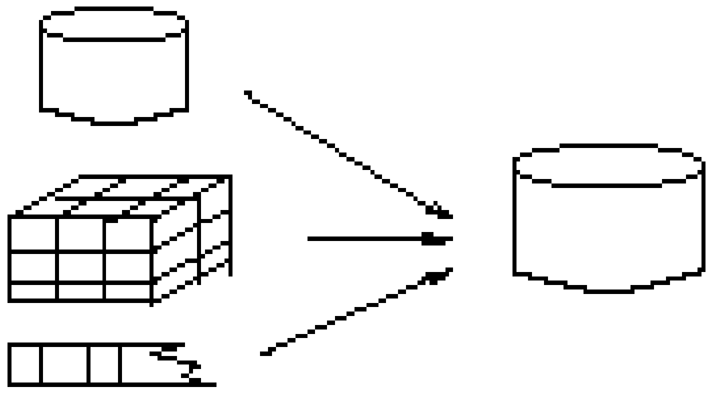
- **Data Cleaning** – Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies.
- **Data Integration** – Integration of multiple databases, data cubes, or files.
- **Data Transformation** – Normalization and aggregation.
- **Data Reduction** – Obtains reduced representation in volume but produces the same or similar analytical results.

Data Preprocessing Steps

Data Cleaning

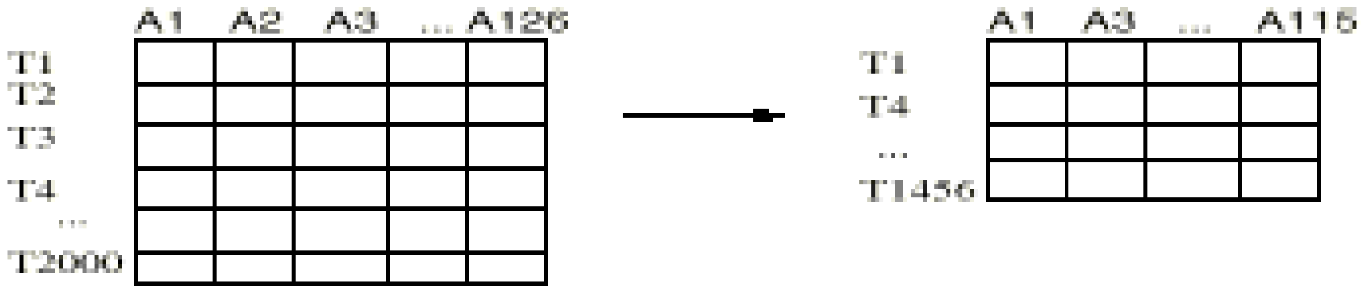


Data Integration



Data Transformation -2, 32, 100, 59, 48 $\longrightarrow$ -0.02, 0.32, 1.00, 0.59, 0.48

Data Reduction



1. Data Cleaning

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

Data cleaning tasks :

- **Fill in missing values.**
- **Identify outliers and smooth out noisy data.**
- **Correct inconsistent data.**

(a). Missing Data : Data is not always available

 E.g., many tuples have **no recorded value** for several attributes, such as : **customer income in sales data**

- **Missing data may be due to :**

1. inconsistent with other recorded data and thus deleted.
2. Data not entered due to misunderstanding.
3. Certain data may not be considered important at the time of entry.
4. Not register history or changes of the data.

• Handle Missing Data :

1. **Ignore the tuple: usually done when class label is missing (assuming the tasks in classification).**

2. Fill in the missing value manually.

3. Use a global constant to fill in the missing values.

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

5. Use the most probable value to fill in the missing value.

(b). Noisy Data: Random error or variance in a measured variable.

- Incorrect attribute values may due to

1. **faulty data collection instruments**
2. **Data entry problems**
3. **Data transmission problems**
4. **Technology limitation**
5. **Inconsistency in naming convention**

- Other data problems which requires data cleaning

1. **Duplicate records**
2. **Incomplete data**
3. **Inconsistent data**

- **Handle Noisy Data:**

1. Binning method:

□ first sort data and partition into (equi-depth) bins, then smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.

2. Regression : smooth by fitting the data into regression functions.

3. Clustering : This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters. Finally Detect and remove outliers.

1. Binning methods for Data Smoothing:

Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* **Partition into (equi-depth) bins:**

- **Bin 1:** 4, 8, 9, 15
- **Bin 2:** 21, 21, 24, 25
- **Bin 3:** 26, 28, 29, 34

* **Smoothing by bin means:**

- **Bin 1:** 9, 9, 9, 9
- **Bin 2:** 23, 23, 23, 23
- **Bin 3:** 29, 29, 29, 29

2. Regression : smooth by fitting the data into regression functions.

Linear regression: Y =a + b X□ Two parameters , a and b specify the line and are to be estimated by using the data at hand.

Multiple regression: Y = b0 + b1 X1 + b2 X2.

□ Many nonlinear functions can be transformed into the above.

Log-linear regression: The multi-way table of joint probabilities is approximated by a product of lower-order tables.

□ Probability: p(a, b, c, d) = ab ac ad □

3. Clustering :

- Partition data set into clusters, and one can store cluster representation only.
- Can be very effective if data is clustered but not if data is “smeared”.
- Can have hierarchical clustering and be stored in multi-dimensional index tree structures.

2. Data Integration

Data Integration – Integration of multiple databases, data cubes, or files.

- combines data from multiple sources into a coherent Data store.
- Data Integration is a data preprocessing technique that involves combining data from multiple heterogeneous data sources into a coherent data store and provide a unified view of the data.
- These sources may include multiple data cubes, databases, or flat files.

Issues in Data Integration:

1. **Schema Integration and Object Matching.**

2. Redundancy.

3. Detection and resolution of data value conflicts.

1. Schema Integration and Object Matching:

Integrate metadata from different sources.

- The real-world entities from multiple sources are matched referred to as the **entity identification problem.**
- **Entity identification problem:** identify real world entities from multiple data sources, **e.g., A.cust-id =B.cust-id**

2. Redundancy :

- Redundant data occur often when integration of multiple databases.
- An attribute may be redundant if it can be derived or obtaining from another attribute or set of attributes.
- Inconsistencies in attributes can also cause redundancies in the resulting data set.
- Some redundancies can be detected by correlation analysis.
- The same attribute may have different names in different databases Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality.

3. Detection and resolution of data value conflicts :

- This is the third important issue in data integration.
- Attribute values from different sources may differ for the same real-world entity.
- An attribute in one system may be recorded at a lower level abstraction than the “same” attribute in another.

3. Data Transformation

- Transform the data in appropriate forms suitable for mining process.
- "Transforming or consolidating data into mining suitable form is known as Data Transformation."
- This involves following ways:

Smoothing

Aggregation

Generalization

Normalization

Attribute construction

- **Smoothing:** remove noise from data.
- **Aggregation:** summarization, data cube construction.
- **Generalization:** concept hierarchy climbing.
- **Normalization:** It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0). scaled to fall within a small, specified range.

1. **Min-max normalization**
2. **Z-score normalization**
3. **Normalization by decimal scaling**

- **Attribute Selection:** In this strategy, new attributes are constructed from the given set of attributes to help the mining process.
- **Discretization:** This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.
- **Concept Hierarchy Generation:** Here attributes are converted from lower level to higher level in hierarchy. **For Example-The attribute “city” can be converted to “country”.**

4. Data Reduction

- “Data reduction techniques are applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of base data.”

- we uses data reduction technique. **It aims to increase the storage efficiency and reduce data storage and analysis costs.**

Data Reduction - Strategies

- Data cube aggregation
- Dimensionality Reduction
- Data Compression
- Numerosity Reduction
- Data Discretization and Concept Hierarchy Generation

1. Data cube aggregation : Aggregation operation is applied to data for the construction of the data cube.

- **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 it.

- **Reference appropriate levels**

 Use the smallest representation which is enough to solve the task.

2. Dimensionality Reduction :

- This reduce the size of data by encoding mechanisms.
- It can be lossy or lossless.
- If after reconstruction from compressed data, original data can be retrieved, such reduction are called **lossless reduction else it is called lossy reduction.**
- The **two effective methods** of dimensionality reduction are:

1. Wavelet Transforms

2. PCA (Principal Component Analysis).

3. Data Compression : Data compression is the process of encoding, restructuring or otherwise modifying data in order to reduce its size.

- **It can be lossy or lossless.**
- **Lossless Compression:** Not Occuring Data Losses.
- **Lossy Compression:** Occurs the Data Losses.

4. Numerosity Reduction :

- This enable to store the model of data instead of whole data,
- **for example: Regression Models.**
- **Linear regression**
- **Multiple regression**
- **Log-linear regression**

Data Discretization and Concept Hierarchy Generation

- **Data Discretization:** This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.
- Divide the range of a continuous attribute into intervals.
- Some classification algorithms only accept categorical attributes.
- Reduce data size by discretization.

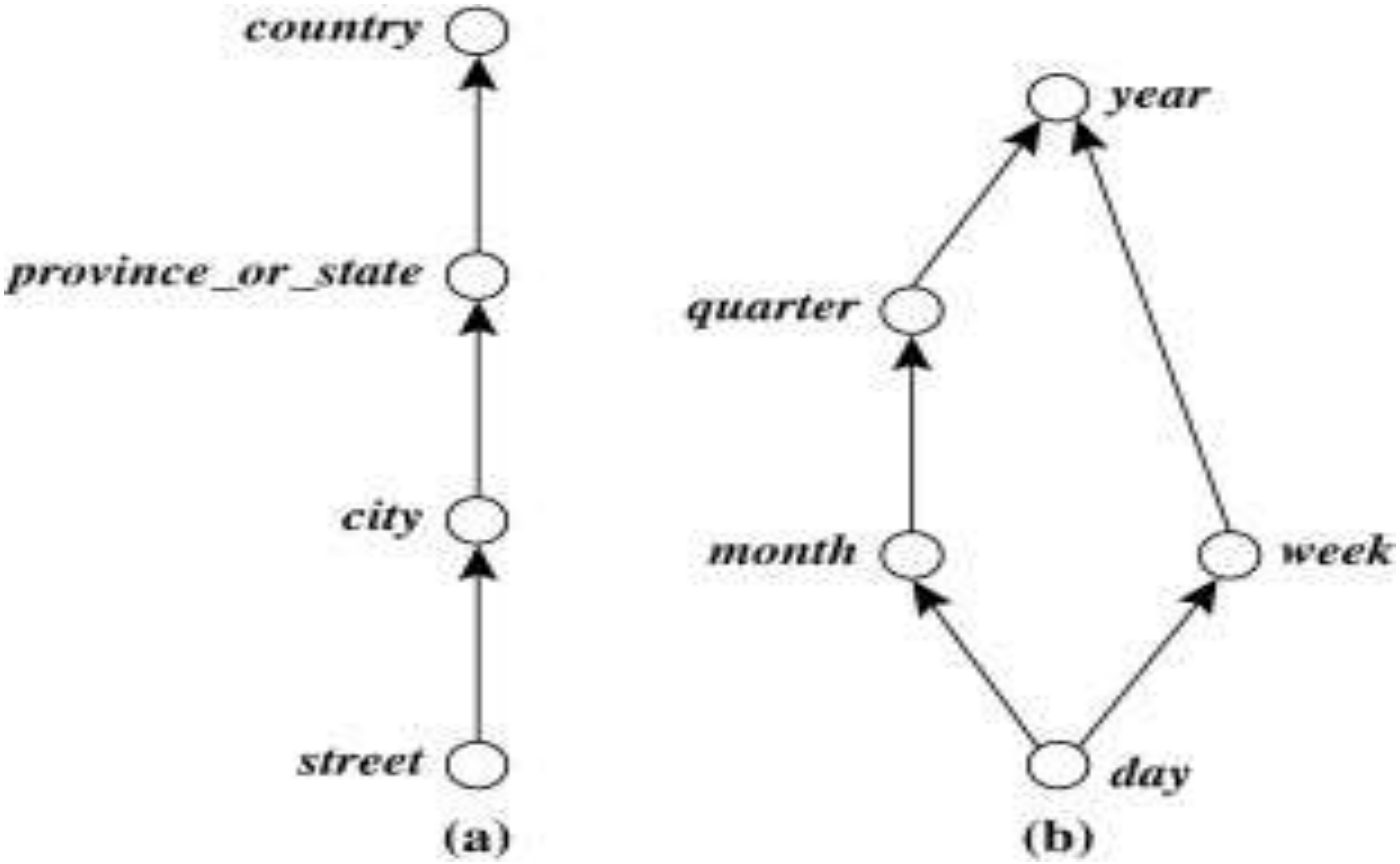
- Reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals.
- Interval labels can then be used to replace actual data values.
- **Three types of attributes:**
- **Nominal** — values from an unordered set.
- **Ordinal** — values from an ordered set.
- **Continuous** — It have a Real Numbers.

- **Concept Hierarchy Generation :** Attributes are converted from lower level to higher level in hierarchy.

- **For Example- The attribute “city” can be converted to “country”.**

- Reduce the data by collecting and replacing low level concepts.
- such as numeric values for the **(attribute - age)** by higher level concepts **(such as young, middle-aged, or old).**

Example for Concept Hierarchy Generation



UNIT II

UNIT II ASSOCIATION RULE MINING AND CLASSIFICATION BASICS

Introduction to Association rules – Association Rule Mining – Mining Frequent Itemsets with and without Candidate Generation – Mining Various Kinds of Association Rules - Classification versus Prediction – Data Preparation for Classification and Prediction.

- Basic concepts
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Summary

Association rule mining

- Proposed by Agrawal et al in 1993.
- It is an important data mining model studied extensively by the database and data mining community.
- Assume all data are categorical.
- No good algorithm for numeric data.
- Initially used for Market Basket Analysis to find how items purchased by customers are related.

Bread $\rightarrow$ Milk [sup = 5%, conf = 100%]

The model: data

- $I = \{i_1, i_2, \ldots, i_m\}$: a set of *items*.
- Transaction *t* :
  - *t* a set of items, and $t \subseteq I$.
- Transaction Database *T*: a set of transactions $T = \{t_1, t_2, \ldots, t_n\}$.

Transaction data: supermarket data

- Market basket transactions:

  t1: {bread, cheese, milk}

  t2: {apple, eggs, salt, yogurt}

  … …

  tn: {biscuit, eggs, milk}
- Concepts:
  - An *item*: an item/article in a basket
  - *I*: the set of all items sold in the store
  - A *transaction*: items purchased in a basket; it may have TID (transaction ID)
  - A *transactional dataset*: A set of transactions

Transaction data: a set of documents

- **A text document data set. Each document is treated as a "bag" of keywords**

doc1:	Student, Teach, School
doc2:	Student, School
doc3:	Teach, School, City, Game
doc4:	Baseball, Basketball
doc5:	Basketball, Player, Spectator
doc6:	Baseball, Coach, Game, Team
doc7:	Basketball, Team, City, Game

The model: rules

- A transaction *t* contains *X*, a set of items (itemset) in *I*, if $X \subseteq t$.
- An association rule is an implication of the form:

$$X \rightarrow Y, \text{ where } X, Y \subset I, \textit{ and } X \cap Y = \emptyset$$

- An itemset is a set of items.
  - E.g., X = {milk, bread, cereal} is an itemset.
- A *k*-itemset is an itemset with *k* items.
  - E.g., {milk, bread, cereal} is a 3-itemset

Rule strength measures

- Support: The rule holds with support *sup* in *T* (the transaction data set) if sup% of transactions contain $X \cup Y$.
  - *sup* = $\Pr(X \cup Y)$.
- Confidence: The rule holds in *T* with confidence *conf* if *conf*% of tranactions that contain *X* also contain *Y*.
  - *conf* = $\Pr(Y \mid X)$
- An association rule is a pattern that states when *X* occurs, *Y* occurs with certain probability.

Support and Confidence

- Support count: The support count of an itemset *X*, denoted by *X.count*, in a data set *T* is the number of transactions in *T* that contain *X*. Assume *T* has *n* transactions.
- Then,

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

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

Goal and key features

- **Goal:** Find all rules that satisfy the user-specified *minimum support* (minsup) and *minimum confidence* (minconf).
- **Key Features**
  - Completeness: find all rules.
  - No target item(s) on the right-hand-side
  - Mining with data on hard disk (not in memory)

An example

t1:	Beef, Chicken, Milk
t2:	Beef, Cheese
t3:	Cheese, Boots
t4:	Beef, Chicken, Cheese
t5:	Beef, Chicken, Clothes, Cheese, Milk
t6:	Chicken, Clothes, Milk
t7:	Chicken, Milk, Clothes

- Transaction data
- Assume:

  minsup = 30%

  minconf = 80%

- An example frequent *itemset*:

{Chicken, Clothes, Milk} [sup = 3/7]

- Association rules from the itemset:

Clothes $\rightarrow$ Milk, Chicken [sup = 3/7, conf = 3/3]

… …

Clothes, Chicken $\rightarrow$ Milk, [sup = 3/7, conf = 3/3]

Transaction data representation

- A simplistic view of shopping baskets,
- Some important information not considered. E.g,
  - the quantity of each item purchased and
  - the price paid.

Many mining algorithms

- There are a large number of them!!
- They use different strategies and data structures.
- Their resulting sets of rules are all the same.
  - Given a transaction data set *T*, and a minimum support and a minimum confident, the set of association rules existing in *T* is uniquely determined.
- Any algorithm should find the same set of rules although their computational efficiencies and memory requirements may be different.
- We study only one: the Apriori Algorithm

Road map

- Basic concepts
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Summary

The Apriori algorithm

- **Probably the best known algorithm**
- **Two steps**:
  - Find all itemsets that have minimum support (*frequent itemsets*, also called large itemsets).
  - Use frequent itemsets to generate rules.

- E.g., a frequent itemset

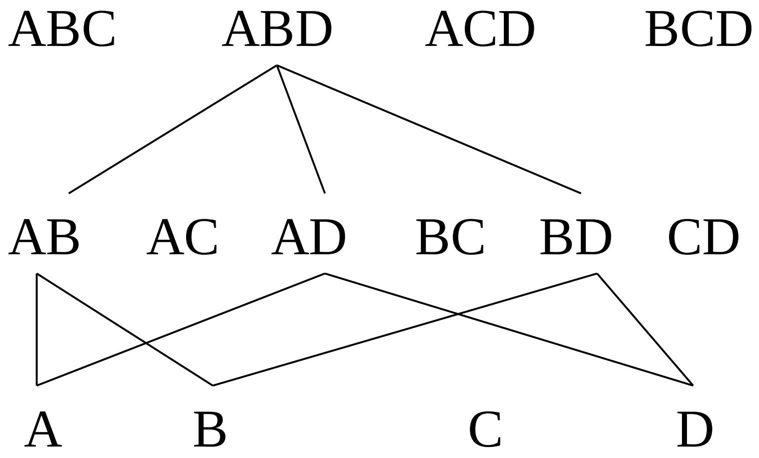
  {Chicken, Clothes, Milk} [sup = 3/7]

  and one rule from the frequent itemset

  Clothes $\rightarrow$ Milk, Chicken [sup = 3/7, conf = 3/3]

Step 1: Mining all frequent itemsets

- A frequent *itemset* is an itemset whose support is ≥ minsup.
- Key idea: The apriori property (downward closure property): any subsets of a frequent itemset are also frequent itemsets



The Algorithm

- Iterative algo. (also called level-wise search):
  Find all 1-item frequent itemsets; then all 2-item frequent itemsets, and so on.
  - In each iteration k, only consider itemsets that contain some k-1 frequent itemset.

- Find frequent itemsets of size 1: F_1
- From $k = 2$
  - C_k = candidates of size k: those itemsets of size k that could be frequent, given F_{k-1}
  - F_k = those itemsets that are actually frequent, $F_k \subseteq C_k$ (need to scan the database once).

Example – Finding frequent itemsets

Dataset T

minsup=0.5

TID	Items
T100	1, 3, 4
T200	2, 3, 5
T300	1, 2, 3, 5
T400	2, 5

itemset:count

1. scan T ➔ C_1: {1}:2, {2}:3, {3}:3, {4}:1, {5}:3

➔ F_1: {1}:2, {2}:3, {3}:3, {5}:3

➔ C_2: {1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}

2. scan T ➔ C_2: {1,2}:1, {1,3}:2, {1,5}:1, {2,3}:2, {2,5}:3, {3,5}:2

➔ F_2: **{1,3}**:2, **{2,3}**:2, **{2,5}**:3, **{3,5}**:2

➔ C_3: {2, 3,5}

3. scan T ➔ C_3: **{2, 3, 5}**:2 ➔ F_3: **{2, 3, 5}**

Details: ordering of items

- The items in *I* are sorted in lexicographic order (which is a total order).
- The order is used throughout the algorithm in each itemset.
- $\{w[1], w[2], \ldots, w[k]\}$ represents a k-itemset w consisting of items $w[1], w[2], \ldots, w[k]$, where $w[1] < w[2] < \ldots < w[k]$ according to the total order.

Details: the algorithm

```
Algorithm Apriori(T)
    $C_1 \leftarrow$ init-pass($T$);
    $F_1 \leftarrow \{f \mid f \in C_1, f.\text{count}/n \geq minsup\}$;    // n: no. of transactions in T
    for ($k$ = 2; $F_{k-1} \neq \emptyset$; $k$++) do
        $C_k \leftarrow$ candidate-gen($F_{k-1}$);
        for each transaction $t \in T$ do
           for each candidate $c \in C_k$ do
                if $c$ is contained in $t$ then
                  $c.count$++;
           end
        end
       $F_k \leftarrow \{c \in C_k \mid c.count/n \geq minsup\}$
    end
return $F \leftarrow \bigcup_k F_k$;
```

Apriori candidate generation

- The candidate-gen function takes F_{k-1} and returns a superset (called the candidates) of the set of all frequent *k*-itemsets. It has two steps
  - *join* step: Generate all possible candidate itemsets C_k of length *k*
  - *prune* step: Remove those candidates in C_k that cannot be frequent.

Candidate-gen function

```
Function candidate-gen($F_{k-1}$)
    $C_k \leftarrow \emptyset$;
    forall $f_1, f_2 \in F_{k-1}$
            with $f_1 = \{i_1, \ldots, i_{k-2}, i_{k-1}\}$
            and $f_2 = \{i_1, \ldots, i_{k-2}, i'_{k-1}\}$
            and $i_{k-1} < i'_{k-1}$ do
        $c \leftarrow \{i_1, \ldots, i_{k-1}, i'_{k-1}\}$;                // join $f_1$ and $f_2$
        $C_k \leftarrow C_k \cup \{c\}$;
        for each ($k$-1)-subset $s$ of $c$ do
            if ($s \notin F_{k-1}$) then
                delete $c$ from $C_k$;                         // prune
        end
    end
    return $C_k$;
```

An example

- F_3 = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {1, 3, 5}, {2, 3, 4}}

- After join
  - C_4 = {{1, 2, 3, 4}, {1, 3, 4, 5}}
- After pruning:
  - C_4 = {{1, 2, 3, 4}}
    because {1, 4, 5} is not in F_3 ({1, 3, 4, 5} is removed)

Step 2: Generating rules from frequent itemsets

- Frequent itemsets ≠ association rules
- One more step is needed to generate association rules
- For each frequent itemset *X*,

  For each proper nonempty subset *A* of *X*,
  - Let *B* = X - *A*
  - A $\rightarrow$ B is an association rule if
    - Confidence(A $\rightarrow$ B) ≥ minconf,

      support(A $\rightarrow$ B) = support(A∪B) = support(X)

      confidence(A $\rightarrow$ B) = support(A ∪ B) / support(A)

Generating rules: an example

- Suppose {2,3,4} is frequent, with sup=50%
  - Proper nonempty subsets: {2,3}, {2,4}, {3,4}, {2}, {3}, {4}, with sup=50%, 50%, 75%, 75%, 75%, 75% respectively
  - These generate these association rules:
    - $2,3 \rightarrow 4$, confidence=100%
    - $2,4 \rightarrow 3$, confidence=100%
    - $3,4 \rightarrow 2$, confidence=67%
    - $2 \rightarrow 3,4$, confidence=67%
    - $3 \rightarrow 2,4$, confidence=67%
    - $4 \rightarrow 2,3$, confidence=67%
    - All rules have support = 50%

Generating rules: summary

- To recap, in order to obtain A $\rightarrow$ B, we need to have support(A $\cup$ B) and support(A)
- All the required information for confidence computation has already been recorded in itemset generation. No need to see the data *T* any more.
- This step is not as time-consuming as frequent itemsets generation.

On Apriori Algorithm

Seems to be very expensive

- Level-wise search
- K = the size of the largest itemset
- It makes at most K passes over data
- In practice, K is bounded (10).
- The algorithm is very fast. Under some conditions, all rules can be found in linear time.
- Scale up to large data sets

More on association rule mining

- Clearly the space of all association rules is exponential, $O(2^m)$, where m is the number of items in *I*.
- The mining exploits sparseness of data, and high minimum support and high minimum confidence values.
- Still, it always produces a huge number of rules, thousands, tens of thousands, millions, ...

Road map

- Basic concepts
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Summary

Different data formats for mining

- The data can be in transaction form or table form

Transaction form:

a, b
a, c, d, e
a, d, f

Table form:

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

- Table data need to be converted to transaction form for association mining

From a table to a set of transactions

Table form:

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

⇒Transaction form:

(Attr1, a), (Attr2, b), (Attr3, d)

(Attr1, b), (Attr2, c), (Attr3, e)

candidate-gen can be slightly improved. Why?

Road map

- Basic concepts
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Summary

Problems with the association mining

- Single minsup: It assumes that all items in the data are of the same nature and/or have similar frequencies.
- Not true: In many applications, some items appear very frequently in the data, while others rarely appear.

  E.g., in a supermarket, people buy *food processor* and *cooking pan* much less frequently than they buy *bread* and *milk*.

Rare Item Problem

- If the frequencies of items vary a great deal, we will encounter two problems
  - If minsup is set too high, those rules that involve rare items will not be found.
  - To find rules that involve both frequent and rare items, minsup has to be set very low. This may cause combinatorial explosion because those frequent items will be associated with one another in all possible ways.

Multiple minsups model

- The minimum support of a rule is expressed in terms of *minimum item supports* (MIS) of the items that appear in the rule.
- Each item can have a minimum item support.
- By providing different MIS values for different items, the user effectively expresses different support requirements for different rules.

Minsup of a rule

- Let MIS(i) be the MIS value of item i. The *minsup* of a rule R is the lowest MIS value of the items in the rule.
- I.e., a rule R: $a_1, a_2, \ldots, a_k \rightarrow a_{k+1}, \ldots, a_r$ satisfies its minimum support if its actual support is $\geq$

  $\min(\text{MIS}(a_1), \text{MIS}(a_2), \ldots, \text{MIS}(a_r))$.

An Example

- Consider the following items:

  bread, shoes, clothes

  The user-specified MIS values are as follows:

  MIS(*bread*) = 2%   MIS(*shoes*) = 0.1%

  MIS(*clothes*) = 0.2%

  The following rule doesn't satisfy its minsup:

  clothes $\rightarrow$ *bread* [sup=0.15%,conf =70%]

  The following rule satisfies its minsup:

  clothes $\rightarrow$ *shoes* [sup=0.15%,conf =70%]

Downward closure property

- In the new model, the property no longer holds (?)

E.g., Consider four items 1, 2, 3 and 4 in a database. Their minimum item supports are

MIS(1) = 10%   MIS(2) = 20%

MIS(3) = 5%   MIS(4) = 6%

{1, 2} with support 9% is infrequent, but {1, 2, 3} and {1, 2, 4} could be frequent.

To deal with the problem

- We sort all items in I according to their MIS values (make it a total order).
- The order is used throughout the algorithm in each itemset.
- Each itemset w is of the following form:

  $\{w[1], w[2], \ldots, w[k]\}$, consisting of items,

  $w[1], w[2], \ldots, w[k]$,

  where $\text{MIS}(w[1]) \leq \text{MIS}(w[2]) \leq \ldots \leq \text{MIS}(w[k])$.

The MSapriori algorithm

```
Algorithm MSapriori(T, MS)
    $M \leftarrow$ sort($I$, $MS$);
    $L \leftarrow$ init-pass($M$, $T$);
    $F_1 \leftarrow \{\{i\} \mid i \in L, i.count/n \geq \text{MIS}(i)\}$;
    for ($k = 2$; $F_{k-1} \neq \emptyset$; $k$++) do
          if $k$=2 then
            $C_k \leftarrow$ level2-candidate-gen($L$)
          else $C_k \leftarrow$ MScandidate-gen($F_{k-1}$);
          end;
          for each transaction $t \in T$ do
             for each candidate $c \in C_k$ do
                 if $c$ is contained in $t$ then
                       $c.count$++;
                 if $c - \{c[1]\}$ is contained in $t$ then
                       $c.tailCount$++
             end
          end
       $F_k \leftarrow \{c \in C_k \mid c.count/n \geq MIS(c[1])\}$
    end
    return $F \leftarrow \bigcup_k F_k$;
```

Candidate itemset generation

- Special treatments needed:
  - Sorting the items according to their MIS values
  - First pass over data (the first three lines)
    - Let us look at this in detail.
  - Candidate generation at level-2
    - Read it in the handout.
  - Pruning step in level-*k* (*k* > 2) candidate generation.
    - Read it in the handout.

First pass over data

- It makes a pass over the data to record the support count of each item.
- It then follows the sorted order to find the first item *i* in *M* that meets MIS(*i*).
  - *i* is inserted into *L*.
  - For each subsequent item *j* in *M* after *i*, if *j.count*/*n* $\geq$ MIS(*i*) then *j* is also inserted into *L*, where *j.count* is the support count of *j* and *n* is the total number of transactions in *T*. Why?
- *L* is used by function level2-candidate-gen

First pass over data: an example

- Consider the four items 1, 2, 3 and 4 in a data set. Their minimum item supports are:

  MIS(1) = 10% MIS(2) = 20%

  MIS(3) = 5% MIS(4) = 6%
- Assume our data set has 100 transactions. The first pass gives us the following support counts:

  {3}.*count* = 6, {4}.*count* = 3,

  {1}.*count* = 9, {2}.*count* = 25.
- **Then** L = {3, 1, 2}, and F_1 = {{3}, {2}}
- Item 4 is not in L because 4.*count*/*n* < MIS(3) (= 5%),
- {1} is not in F_1 because 1.*count*/*n* < MIS(1) (= 10%).

Rule generation

- The following two lines in MSapriori algorithm are important for rule generation, which are not needed for the Apriori algorithm

```
if $c$ – {$c$[1]} is contained in $t$ then
    $c$.tailCount++
```

- Many rules cannot be generated without them.
- Why?

On multiple minsup rule mining

- Multiple minsup model subsumes the single support model.
- It is a more realistic model for practical applications.
- The model enables us to found rare item rules yet without producing a huge number of meaningless rules with frequent items.
- By setting MIS values of some items to 100% (or more), we effectively instruct the algorithms not to generate rules only involving these items.

Road map

- Basic concepts
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Summary

Mining class association rules (CAR)

- Normal association rule mining does not have any target.
- It finds all possible rules that exist in data, i.e., any item can appear as a consequent or a condition of a rule.
- However, in some applications, the user is interested in some targets.
  - E.g, the user has a set of text documents from some known topics. He/she wants to find out what words are associated or correlated with each topic.

Problem definition

- Let *T* be a transaction data set consisting of *n* transactions.
- Each transaction is also labeled with a class *y*.
- Let *I* be the set of all items in *T*, *Y* be the set of all class labels and $I \cap Y = \emptyset$.
- A **class association rule** (**CAR**) is an implication of the form

  $X \rightarrow y$, where $X \subseteq I$, and $y \in Y$.
- The definitions of **support** and **confidence** are the same as those for normal association rules.

An example

- **A text document data set**

doc 1:	Student, Teach, School	: Education
doc 2:	Student, School	: Education
doc 3:	Teach, School, City, Game	: Education
doc 4:	Baseball, Basketball	: Sport
doc 5:	Basketball, Player, Spectator	: Sport
doc 6:	Baseball, Coach, Game, Team	: Sport
doc 7:	Basketball, Team, City, Game	: Sport

- Let *minsup* = 20% and *minconf* = 60%. The following are two examples of class association rules:

  Student, School $\rightarrow$ Education [sup= 2/7, conf = 2/2]

  game $\rightarrow$ Sport [sup= 2/7, conf = 2/3]

Mining algorithm

- Unlike normal association rules, CARs can be mined directly in one step.
- The key operation is to find all **ruleitems** that have support above *minsup*. A **ruleitem** is of the form:

  (*condset*, *y*)

  where **condset** is a set of items from *I* (*i.e., condset* $\subseteq$ *I*), and $y \in Y$ is a class label.
- Each ruleitem basically represents a rule:

  $condset \rightarrow y$,
- The Apriori algorithm can be modified to generate CARs

Multiple minimum class supports

- The multiple minimum support idea can also be applied here.
- The user can specify different minimum supports to different classes, which effectively assign a different minimum support to rules of each class.
- For example, we have a data set with two classes, Yes and No. We may want
  - rules of class Yes to have the minimum support of 5% and
  - rules of class No to have the minimum support of 10%.
- By setting minimum class supports to 100% (or more for some classes), we tell the algorithm not to generate rules of those classes.
  - This is a very useful trick in applications.

UNIT III

UNIT III
CLASSIFICATION AND PREDICTION TECHNIQUES

Classification by Decision Tree – Bayesian Classification – Rule Based Classification – Bayesian Belief Networks – Classification by Back Propagation – Support Vector Machines – K-Nearest Neighbor Algorithm –Linear Regression, Nonlinear Regression, Other Regression-Based Methods.

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Classification by back propagation
- Lazy learners (or learning from your neighbors)
- Frequent-pattern-based classification
- Other classification methods
- Prediction
- Accuracy and error measures

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

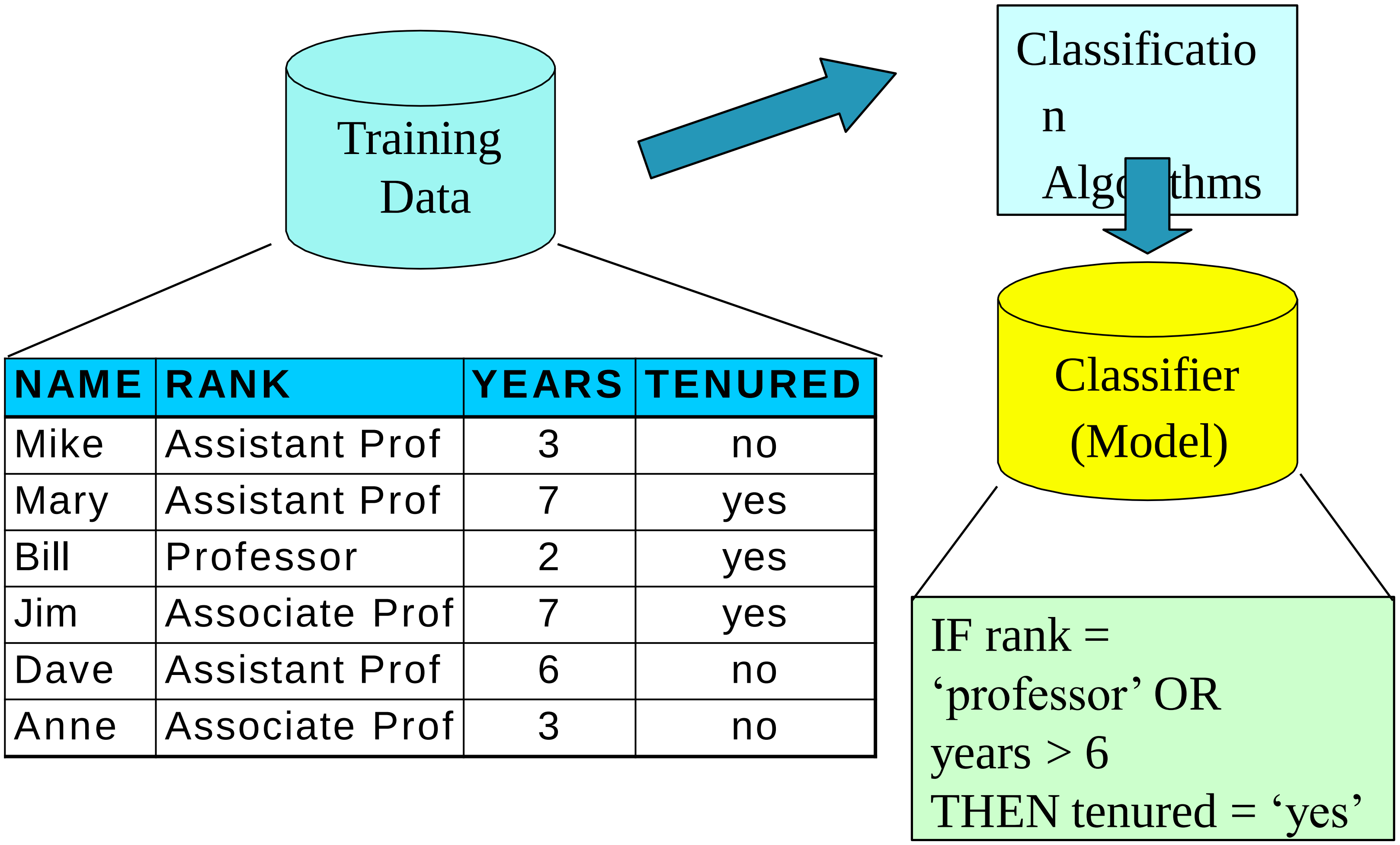
Classification vs. Prediction

- Classification
  - predicts categorical class labels (discrete or nominal)
  - 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
- Prediction
  - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical applications
  - Credit/loan approval:
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is

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 is 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
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

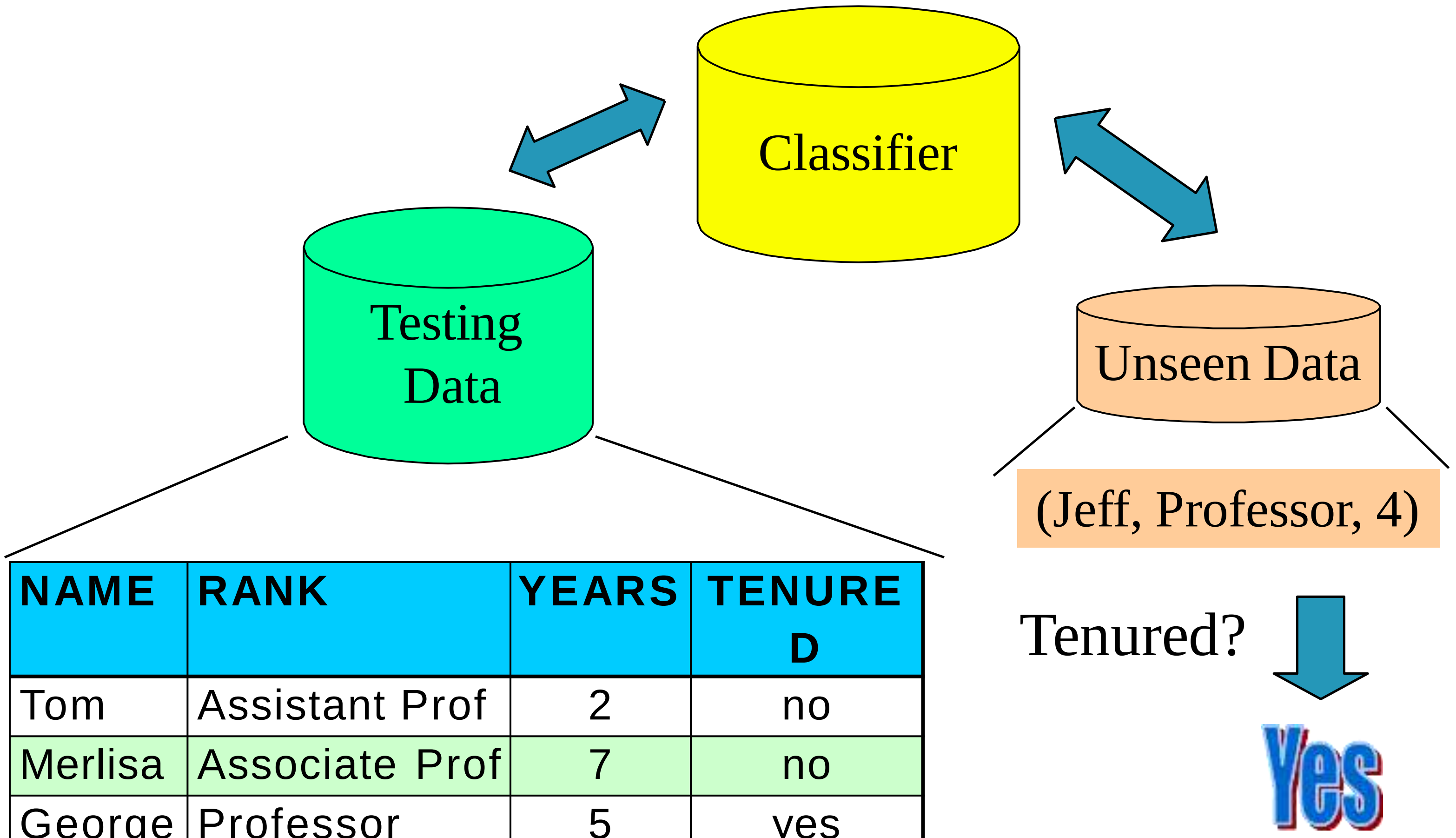
Process (1): Model Construction



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

IF rank = 'professor' OR
years > 6
THEN tenured = 'yes'

Process (2): Using the Model in Prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Issues: Data Preparation

- Data cleaning
  - Preprocess data in order to reduce noise and handle missing values
- Relevance analysis (feature selection)
  - Remove the irrelevant or redundant attributes
- Data transformation
  - Generalize and/or normalize data

Issues: Evaluating Classification Methods

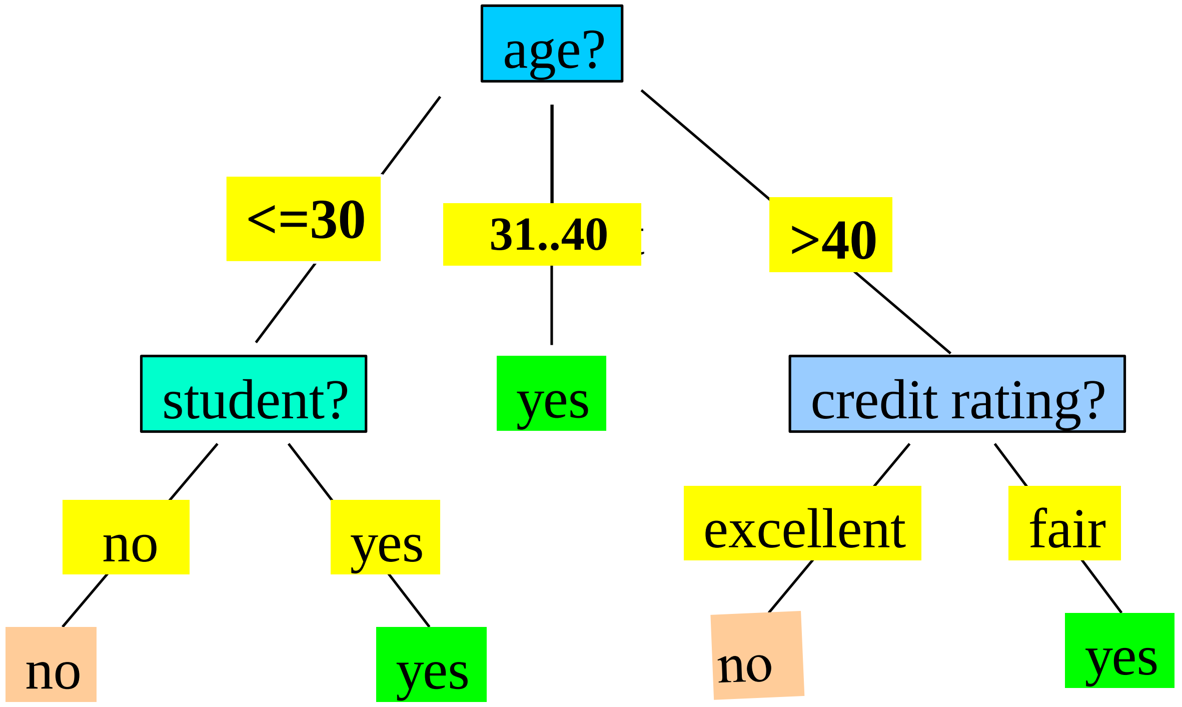
- Accuracy
  - classifier accuracy: predicting class label
  - predictor accuracy: guessing value of predicted attributes
- Speed
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- Robustness: handling noise and missing values
- Scalability: efficiency in disk-resident databases
- Interpretability
  - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Decision Tree Induction: Training Dataset

This follows an example of Quinlan's ID3 (Playing Tennis)

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

Output: A Decision Tree for *"buys_computer"*



Algorithm for Decision Tree Induction

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

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

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

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

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

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

  - Information gained by branching on attribute A

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

Attribute Selection: Information Gain

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

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

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

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

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

$\frac{5}{14} I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

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

Similarly,

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

Computing Information-Gain for Continuous-Value Attributes

- Let attribute A be a continuous-valued attribute
- Must determine the best split point for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible split point
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the minimum expected information requirement for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A $\leq$ split-point, and D2 is the set of tuples in D satisfying A $>$ split-point

Gain Ratio for Attribute Selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

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

  - GainRatio(A) = Gain(A)/SplitInfo(A)
- Ex

$$SplitInfo_A(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 0.926$$

  - gain_ratio(income) = 0.029/0.926 = 0.031
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini index (CART, IBM IntelligentMiner)

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

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

where p_j is the relative frequency of class j in D

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

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

- Reduction in Impurity: $$\Delta gini(A) = gini(D) - gini_A(D)$$

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

Gini index (CART, IBM IntelligentMiner)

- Ex. D has 9 tuples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D$_1$: {low, medium} and 4 in D$_2$

$$gini_{income\in\{low,medium\}}(D) = \left(\frac{10}{14}\right)Gini(D_1) + \left(\frac{4}{14}\right)Gini(D$$

$$\begin{aligned} &= \frac{10}{14}(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2) + \frac{4}{14}(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2) \\ &= 0.450 \\ &= Gini_{income\ \in\ \{high\}}(D) \end{aligned}$$

but gini$_{\{medium,high\}}$ is 0.30 and thus the best since it is the lowest

- All attributes are assumed continuous-valued
- May need other tools, e.g., clustering, to get the possible split values
- Can be modified for categorical attributes

Comparing Attribute Selection Measures

- The three measures, in general, return good results but
  - Information gain:
    - biased towards multivalued attributes
  - Gain ratio:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - Gini index:
    - biased to multivalued attributes
    - has difficulty when # of classes is large
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

Classification in Large Databases

- Classification—a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why decision tree induction in data mining?
  - relatively faster learning speed (than other classification methods)
  - convertible to simple and easy to understand classification rules
  - can use SQL queries for accessing databases
  - comparable classification accuracy with other methods

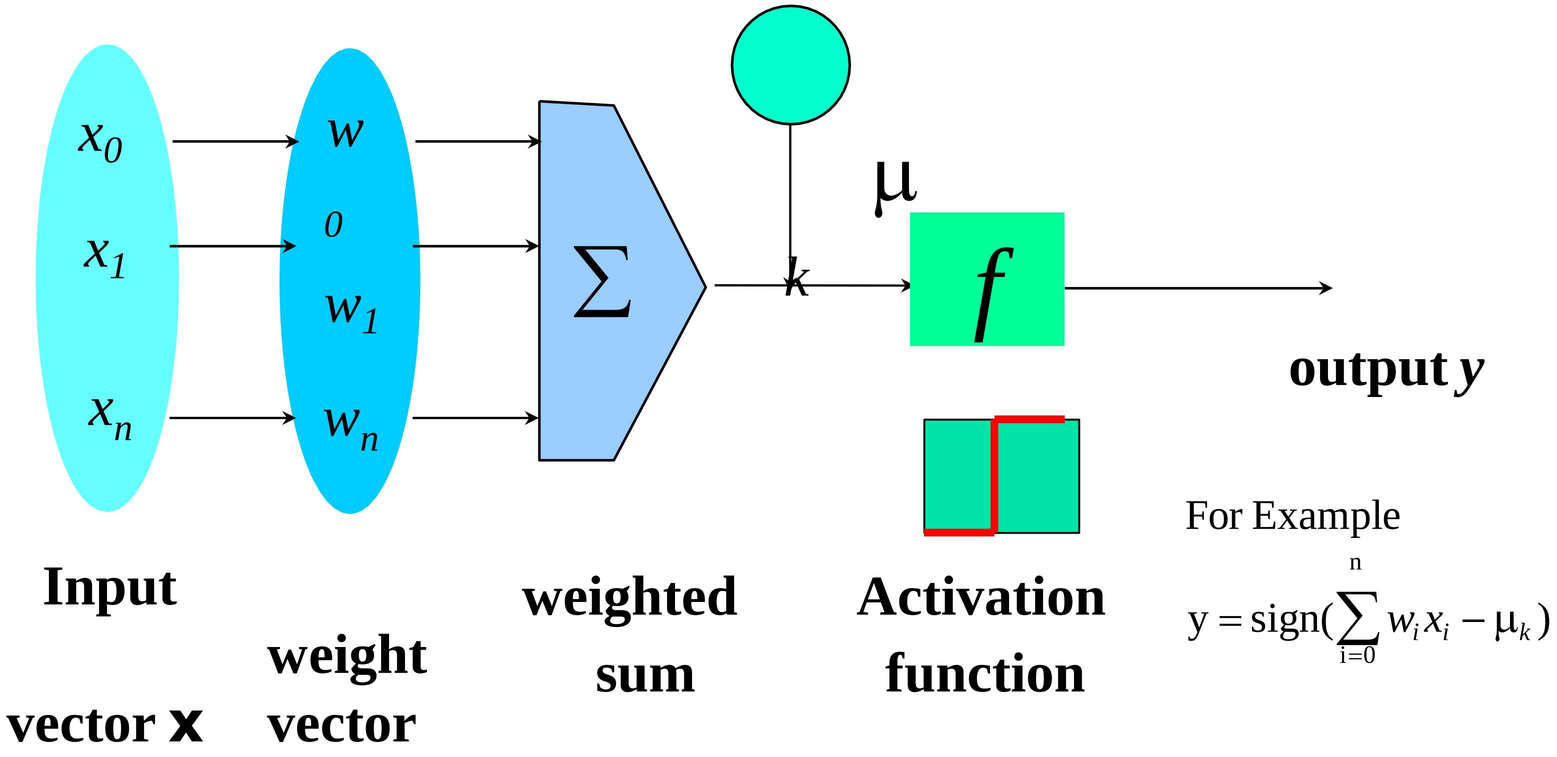
Classification by Backpropagation

- 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)

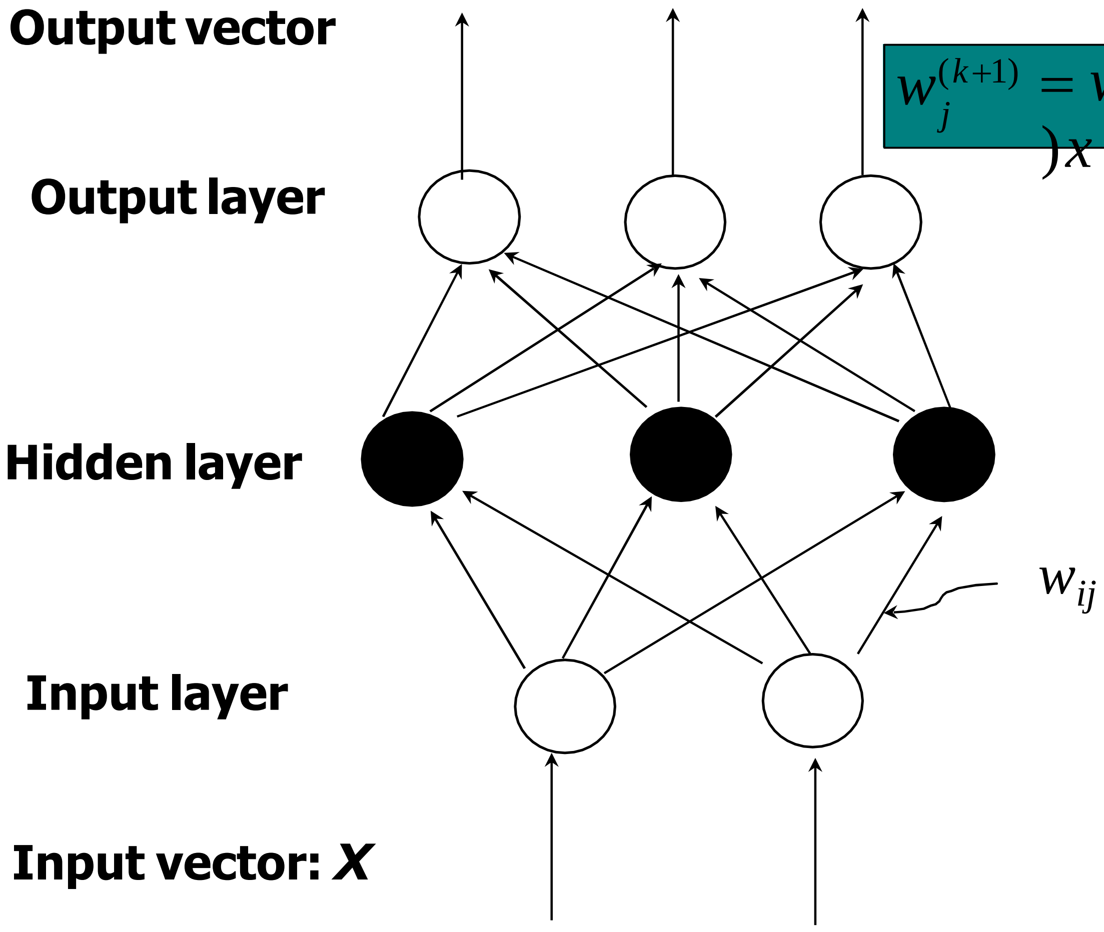


$$y = \text{sign}\left(\sum_{i=0}^{n} w_i x_i - \mu_k\right)$$

- 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

$$w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$$



How A Multi-Layer Neural Network Works?

- 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.)

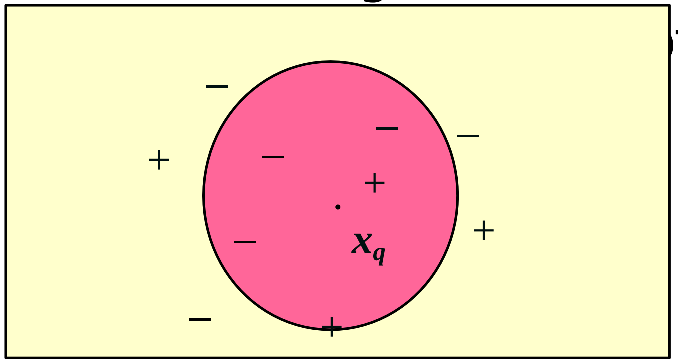
Lazy vs. Eager Learning

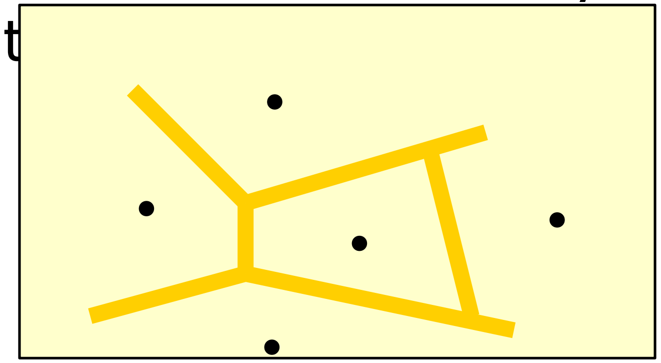
- Lazy vs. eager learning
  - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - Eager learning (the above discussed methods): Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist($\mathbf{X_1}$, $\mathbf{X_2}$)
- Target function could be discrete- or real- valued
- For discrete-valued, k-NN returns the most common value among the k training examples nearest to x_q
- Vonoroi diagram: the decision surface induced by

f t





Discussion on the *k*-NN Algorithm

- k-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the k nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query x_q
    - Give greater weight to closer neighbors $w \equiv \frac{1}{d(x_q, x_i)^2}$
- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Genetic Algorithms (GA)

- Genetic Algorithm: based on an analogy to biological evolution
- An initial **population** is created consisting of randomly generated rules
  - Each rule is represented by a string of bits
  - E.g., if A_1 and $\neg A_2$ then C_2 can be encoded as 100
  - If an attribute has $k > 2$ values, k bits can be used
- Based on the notion of survival of the **fittest**, a new population is formed to consist of the fittest rules and their offsprings
- The fitness of a rule is represented by its classification accuracy on a set of training examples
- Offsprings are generated by crossover and mutation
- The process continues until a population P evolves when each rule in P satisfies a prespecified threshold
- Slow but easily parallelizable

What Is 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

Linear Regression

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

  $y = w_0 + w_1 x$

  where w_0 (y-intercept) and w_1 (slope) are regression coefficients
- Method of least squares: estimates the best-fitting straight line

$$w_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2} \qquad w_0 = \bar{y} - w_1 \bar{x}$$

- Multiple linear regression: involves more than one predictor variable
  - Training data is of the form ($\mathbf{X_1}$, y_1), ($\mathbf{X_2}$, y_2),…, ($\mathbf{X_{|D|}}$, $y_{|D|}$)
  - Ex. For 2-D data, we may have: $y = w_0 + w_1 x_1 + w_2 x_2$
  - Solvable by extension of least square method or using SAS, S-Plus
  - Many nonlinear functions can be transformed into the above

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,

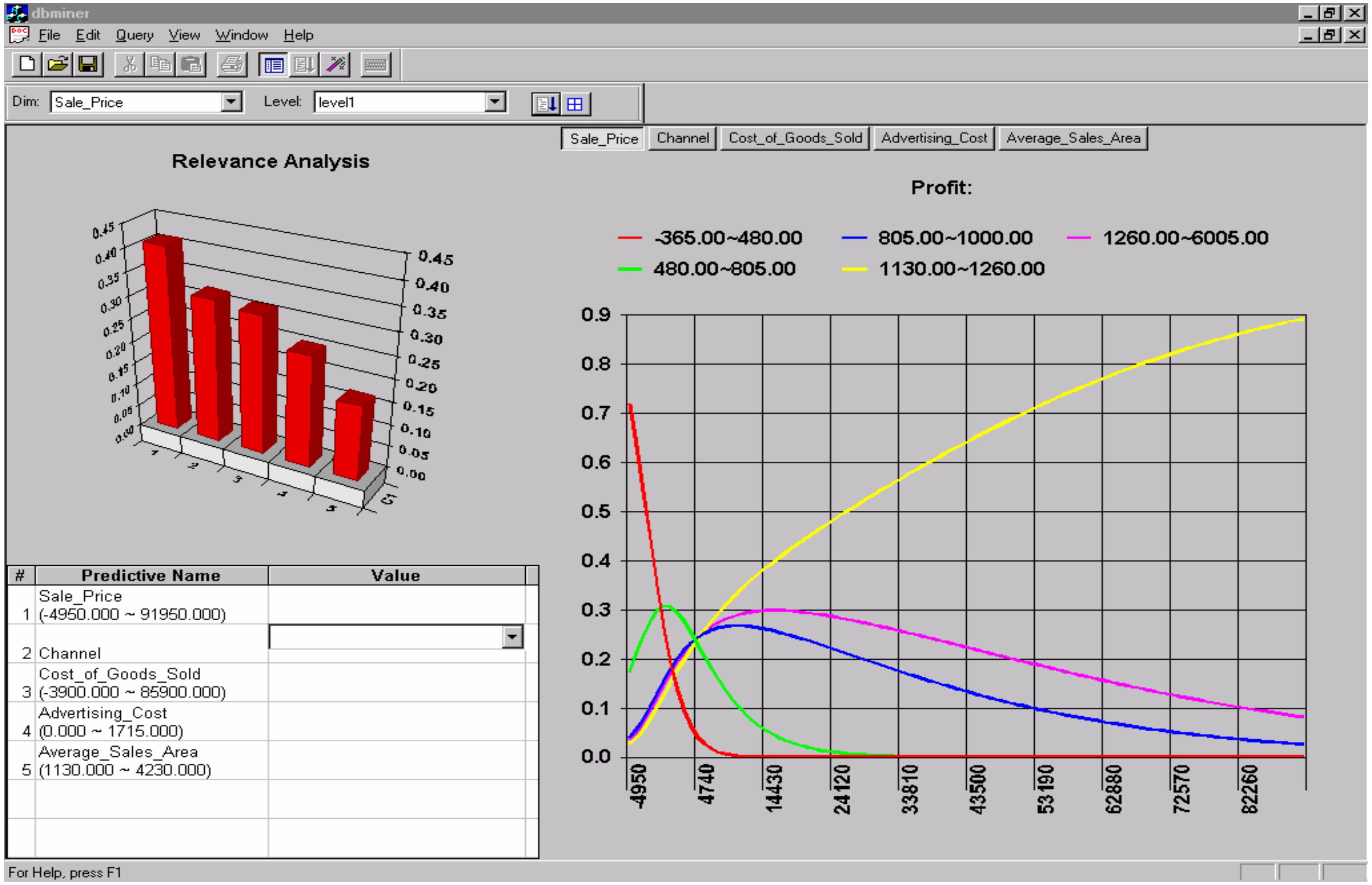
  $y = w_0 + w_1\, x + w_2\, x^2 + w_3\, x^3$

  convertible to linear with new variables: $x_2 = x^2$, $x_3 = x^3$ $y = w_0 + w_1\, x + w_2\, x_2 + w_3\, x_3$
- 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

Other Regression-Based Models

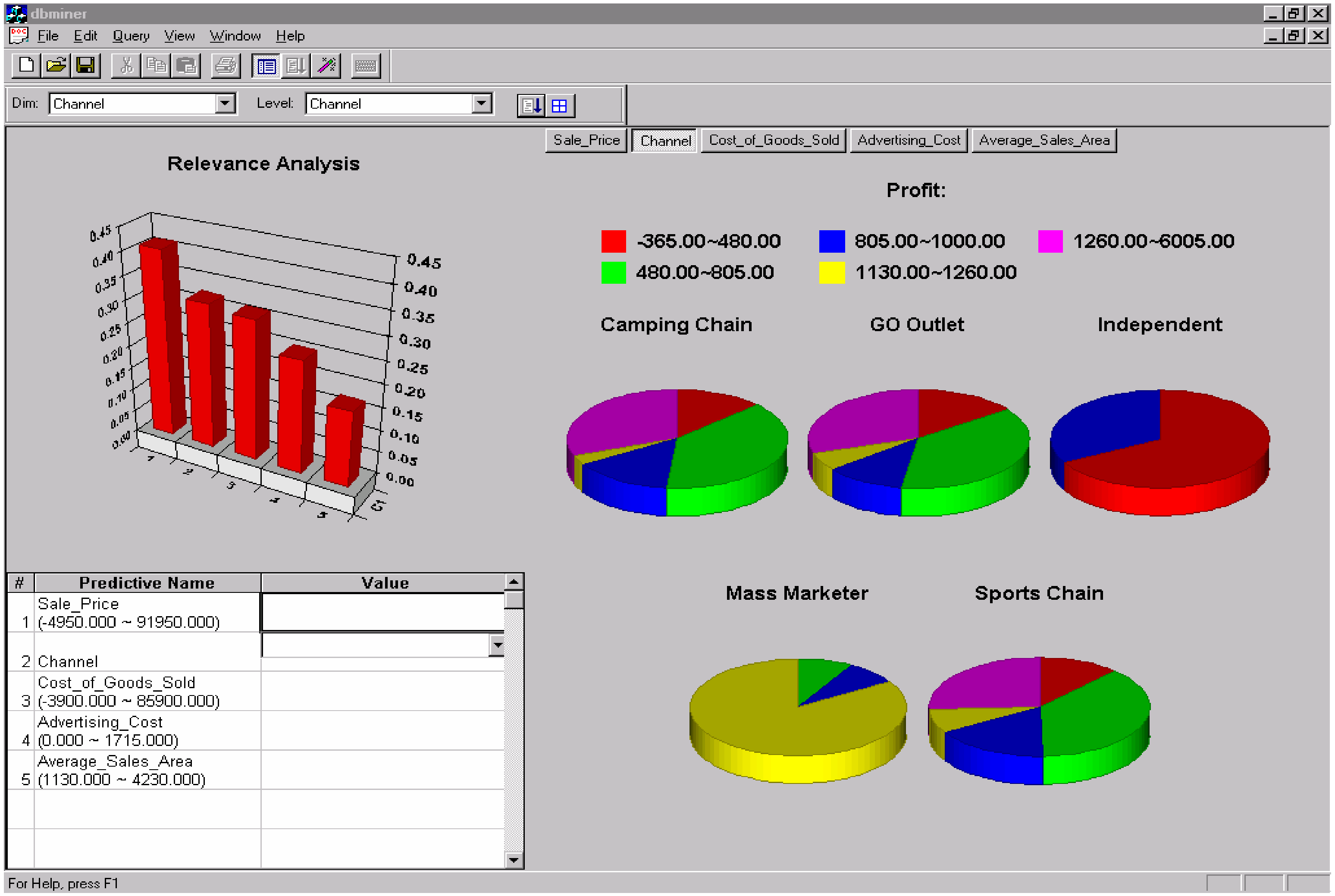
- Generalized linear model:
  - Foundation on which linear regression can be applied to modeling categorical response variables
  - Variance of y is a function of the mean value of y, not a constant
  - Logistic regression: models the prob. of some event occurring as a linear function of a set of predictor variables
  - Poisson regression: models the data that exhibit a Poisson distribution
- Log-linear models: (for categorical data)
  - Approximate discrete multidimensional prob. distributions
  - Also useful for data compression and smoothing
- Regression trees and model trees
  - Trees to predict continuous values rather than class labels

Prediction: Numerical Data



#	Predictive Name	Value
1	Sale_Price (-4950.000 ~ 91950.000)	
2	Channel	
3	Cost_of_Goods_Sold (-3900.000 ~ 85900.000)	
4	Advertising_Cost (0.000 ~ 1715.000)	
5	Average_Sales_Area (1130.000 ~ 4230.000)	

Prediction: Categorical Data



#	Predictive Name	Value
1	Sale_Price (-4950.000 ~ 91950.000)	
2	Channel	
3	Cost_of_Goods_Sold (-3900.000 ~ 85900.000)	
4	Advertising_Cost (0.000 ~ 1715.000)	
5	Average_Sales_Area (1130.000 ~ 4230.000)	

Classifier Accuracy Measures

Real class\Predicted class	C_1	$\sim C_1$
C_1	True positive	False negative
$\sim C_1$	False positive	True negative

Real class\Predicted class	buy_computer = yes	buy_computer = no	total	recognition(%)
buy_computer = yes	6954	46	7000	99.34
buy_computer = no	412	2588	3000	86.27
total	7366	2634	10000	95.52

- Accuracy of a classifier M, acc(M): percentage of test set tuples that are correctly classified by the model M
  - Error rate (misclassification rate) of M = 1 – acc(M)
  - Given m classes, $CM_{i,j}$, an entry in a **confusion matrix**, indicates # of tuples in class i that are labeled by the classifier as class j
- Alternative accuracy measures (e.g., for cancer diagnosis)

  sensitivity = t-pos/pos /* true positive recognition rate */

  specificity = t-neg/neg /* true negative recognition rate */

  precision = t-pos/(t-pos + f-pos)

  accuracy = sensitivity * pos/(pos + neg) + specificity * neg/(pos + neg)

  - This model can also be used for cost-benefit analysis

Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- **Loss function**: measures the error betw. y_i and the predicted value y_i'
  - Absolute error: $| y_i - y_i'|$
  - Squared error: $(y_i - y_i')^2$

$$\frac{\sum_{i=1}^{d} | y_i - y_i'|}{d} \qquad \frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$$

$$\frac{\sum_{i=1}^{d} | y_i - y_i'|}{\sum_{i=1}^{d} | y_i - \bar{y}|} \qquad \frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$$

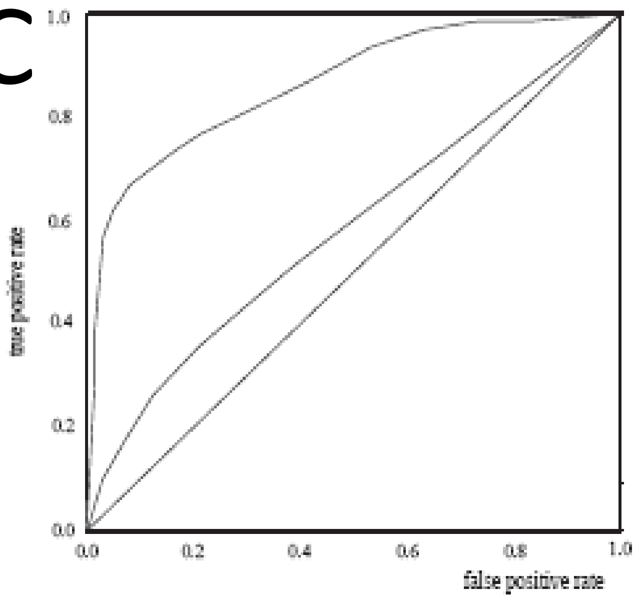
- Test error (generalization error): the average loss over the test set

Evaluating the Accuracy of a Classifier or Predictor (I)

- Holdout method
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- Cross-validation (k-fold, where k = 10 is most popular)
  - Randomly partition the data into k mutually exclusive subsets, each approximately equal size
  - At i-th iteration, use D_i as test set and others as training set
  - Leave-one-out: k folds where k = # of tuples, for small sized data
  - Stratified cross-validation: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Model Selection: ROC Curves

- ROC (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model



- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

UNIT – IV

CLUSTERING TECHNIQUES

UNIT IV
CLUSTERING TECHNIQUES

Cluster Analysis – Partitioning Methods: k-Means and k- Mediods – Hierarchical Methods: Agglomerative and Divisive – Density–Based Method: DBSCAN – Model Based Clustering Methods- Clustering High Dimensional Data - Outlier Analysis.

OVERVIEW

- Cluster Analysis: Basic Concepts
- Types of Data in Cluster Analysis
- A Categorization of Major Clustering Methods
    1. Partitioning Methods
    2. Hierarchical Methods
    3. Density-Based Methods
    4. Grid-Based Methods
    5. Model-Based Clustering Methods
- Clustering High- Dimensional Data
- Constraint-Based Cluster Analysis
- Outlier Analysis

Cluster Analysis: Basic Concepts

What is Cluster Analysis?

- Cluster: A collection of data objects.
  - similar (or related) to one another within the same group.
  - dissimilar (or unrelated) to the objects in other groups.
- Cluster analysis (or *clustering, data segmentation, …*)
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters.
- Clustering is a Unsupervised learning Concepts.
  - As a stand-alone tool to get insight into data distribution.
  - As a preprocessing step for other algorithms.

Applications of Clustering

- **Biology**: Taxonomy of living things like kingdom, phylum, class, order, family, genus and species.
- **Information retrieval**: To document clustering.
- **Land use**: Identification of areas of similar land use in an earth observation database.
- **Marketing**: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs.
- **Climate:** Understanding earth climate, find patterns of atmospheric and ocean.
- **Economic Science**: Used for Market Research.

What Is Good Clustering?

- A good clustering method will produce high quality clusters.
  - High intra-class similarity: cohesive within clusters.
  - Low inter-class similarity: distinctive between clusters.
- The quality of a clustering method depends on :
  - The similarity measure used by the method.
  - Its implementation, and
  - Its ability to discover some or all of the hidden patterns.

Types of Data in Cluster Analysis

1. **Interval-Scaled variables**
2. **Binary variables**
3. **Nominal, Ordinal, and Ratio variables**
4. **Variables of mixed types**

1. Interval-Scaled variables :

Interval-scaled variables are continuous measurements of a roughly linear scale.

Example:

weight and height, latitude and longitude coordinates (e.g., when clustering houses), and weather temperature.

2. Binary variables :

A binary variable is a variable that can take only 2 values.

Example : Generally gender variables can take 2 variables **male and female**.

3. Nominal, Ordinal, and Ratio variables:

Nominal (or) Categorical variables

A generalization of the binary variable in that it can take more than 2 states.

Example : red, yellow, blue, green.

Ordinal Variables:

An ordinal variable can be discrete or continuous.

Example : Rank.

Ratio variables :

It is a positive measurement on a nonlinear scale, approximately at an exponential scale.

Example : Ae^Bt or A^e-Bt.

4. Variables of mixed types :

A database may contain all the six types of variables symmetric binary, asymmetric binary, nominal, ordinal, interval, and ratio. Those combinedly called as **mixed-type variables**.

A Categorization of Major Clustering Methods

- **Partitioning Methods:**
  - Construct various partitions and then evaluate them by some Condition, e.g., minimizing the sum of square errors.
  - Typical methods: k-means, k- medoids , CLARANS
- **Hierarchical Methods:**
  - Create a hierarchical decomposition of the set of data (or objects) using some Condition.
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- **Density-Based Methods:**
  - Based on connectivity and density functions.
  - Typical methods: DBSACN, OPTICS, DenClue

- **Grid-Based Methods**:
  - based on a multiple-level granularity structure.
  - Typical methods: STING, WaveCluster, CLIQUE

- **Model-Based Clustering Methods**:
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB

1. Partitioning Methods:

- Construct various partitions and then evaluate them by some Condition, e.g., minimizing the sum of square errors.

- Typical methods: k-means, k- medoids , CLARANS

- Partitioning a database ***D*** of ***n*** objects into a set of ***k*** clusters, such that the sum of squared distances is minimized (where c_i is the centroid or medoid of cluster C_i)

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (p - c_i)^2$$

- Given *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - **Global optimal**: exhaustively enumerate all partitions
  - **Heuristic methods**: *k-means* and *k-medoids* algorithms

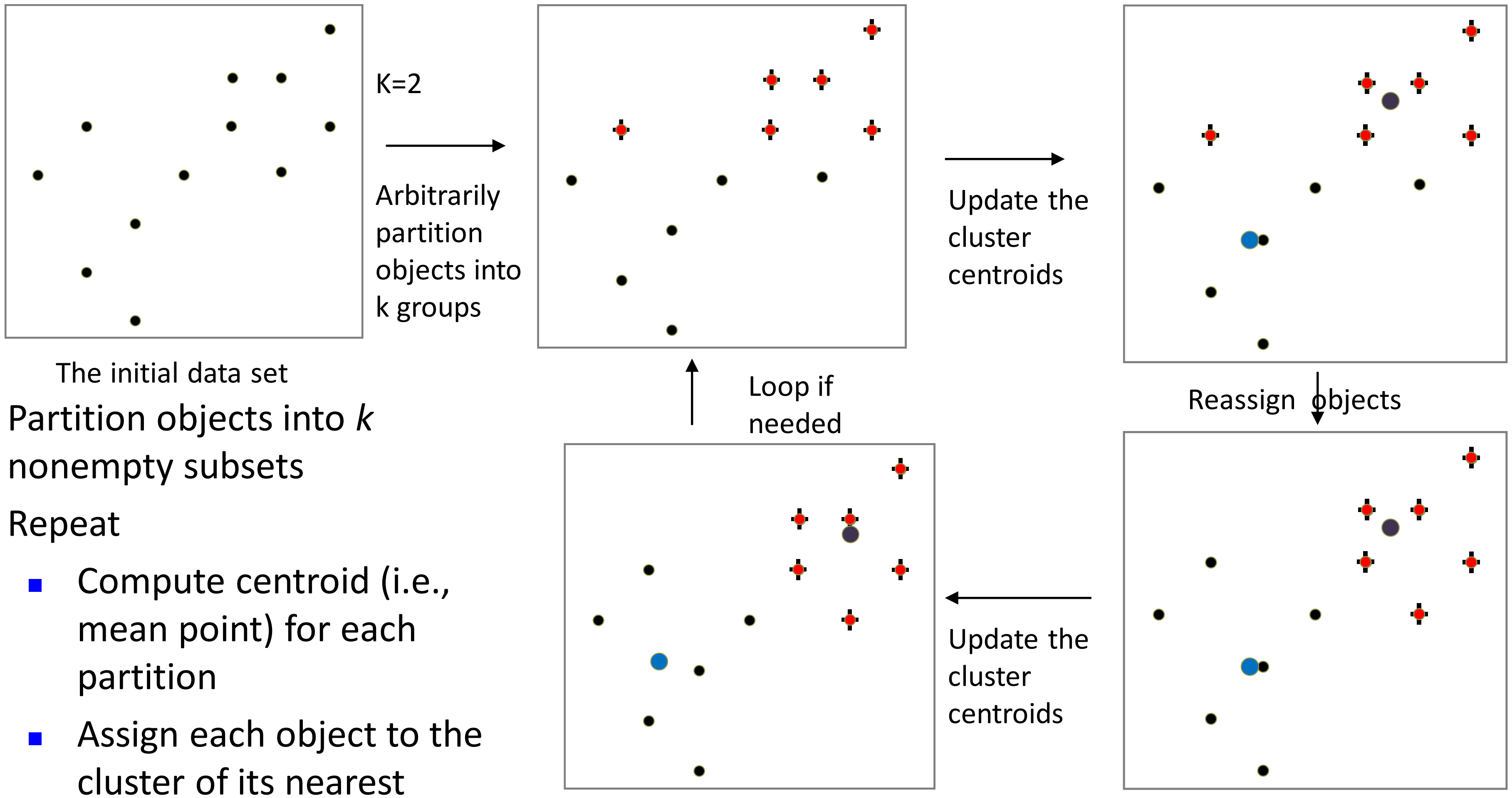
k-means : Each cluster is represented by the center of the cluster

k-medoids **or PAM** (Partition around medoids) : Each cluster is represented by one of the objects in the cluster

The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  - Partition objects into *k* nonempty subsets
  - Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point.
  - Go back to Step 2, stop when the assignment does not change.

An Example of *K-Means* Clustering


The initial data set
K=2
Arbitrarily partition objects into k groups
Update the cluster centroids
Reassign objects
Update the cluster centroids
Loop if needed

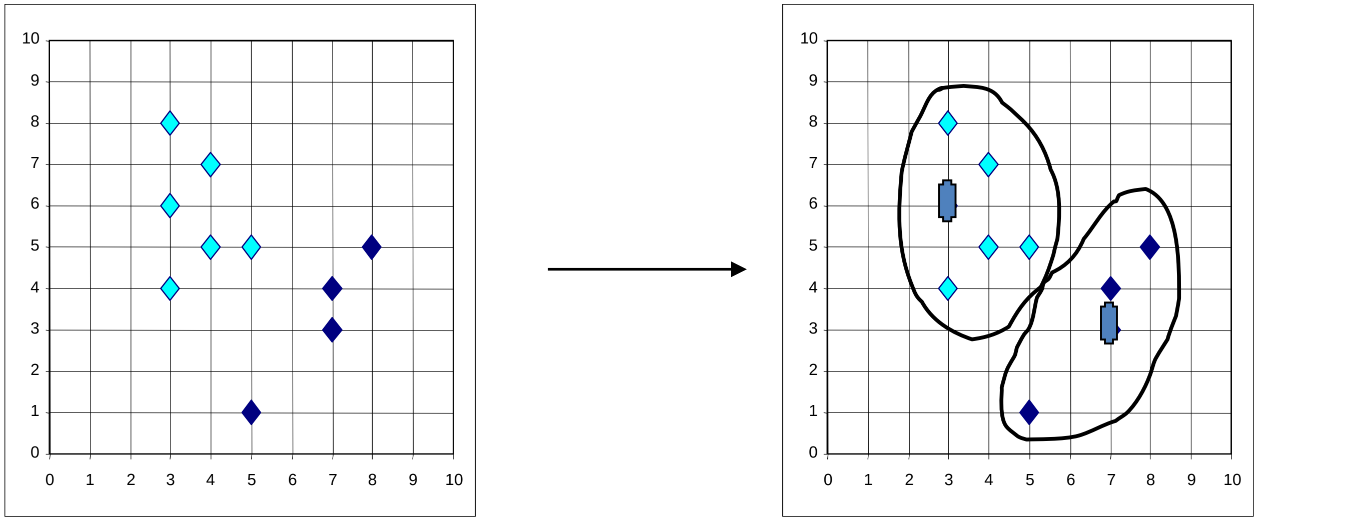
- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

- **Strength:** *Efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
    - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- **Weakness:**
    - Applicable only to objects in a continuous n-dimensional space
        - Using the k-modes method for categorical data.
        - In comparison, k-medoids can be applied to a wide range of data.
    - Need to specify *k,* the *number* of clusters, in advance (there are ways to automatically determine the best k .
    - Sensitive to noisy data and *outliers.*
    - Not suitable to discover clusters with *non-convex shapes.*

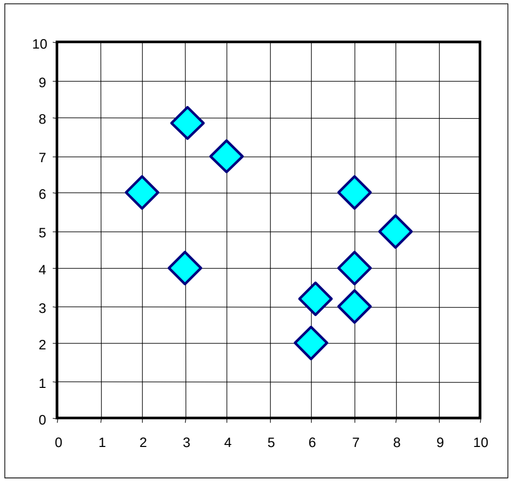
The K-Medoids Clustering Method

- *K-Medoids* Clustering: Find *representative* objects (**medoids**) in clusters
  - ***PAM* (Partitioning Around Medoids)**
    - Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering.
    - ***PAM*** works effectively for small data sets, but does not scale well for large data sets (due to the computational complexity).
- Efficiency improvement on PAM
  - ***CLARA* (Kaufmann & Rousseeuw, 1990): PAM on samples.**
  - ***CLARANS* (Ng & Han, 1994): Randomized re-sampling**.

- **K-Medoids:** Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.

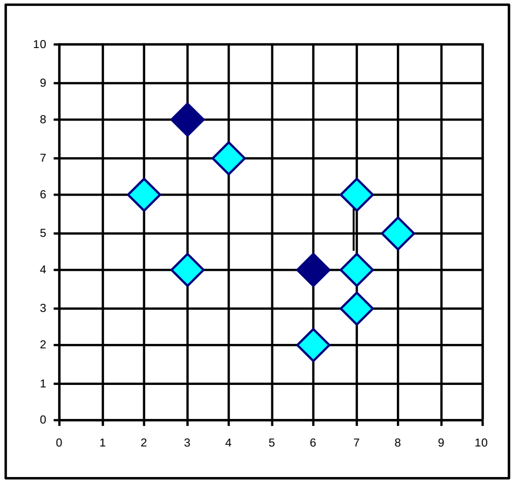


PAM: A Typical K-Medoids Algorithm



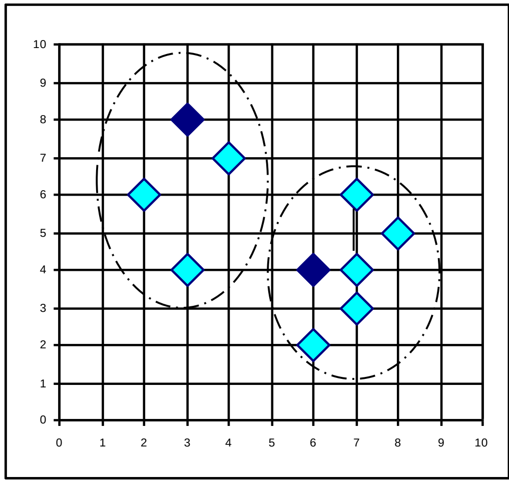
K=2

Arbitrary choose k object as initial medoids

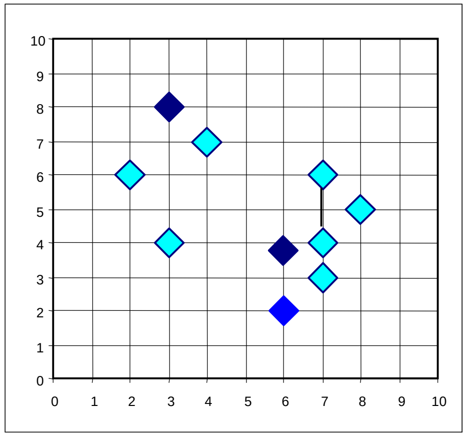


Assign each remaining object to nearest medoids

Total Cost = 20

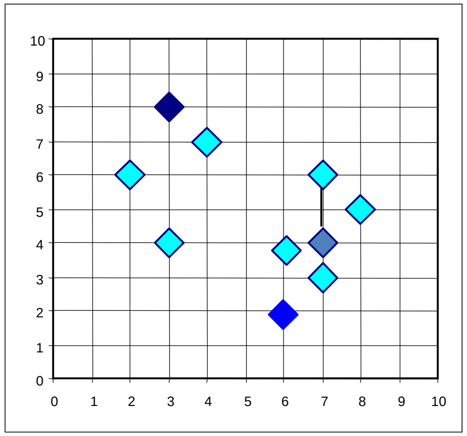


Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

If quality is improved.

Do loop

Until no change

2. Hierarchical Methods:

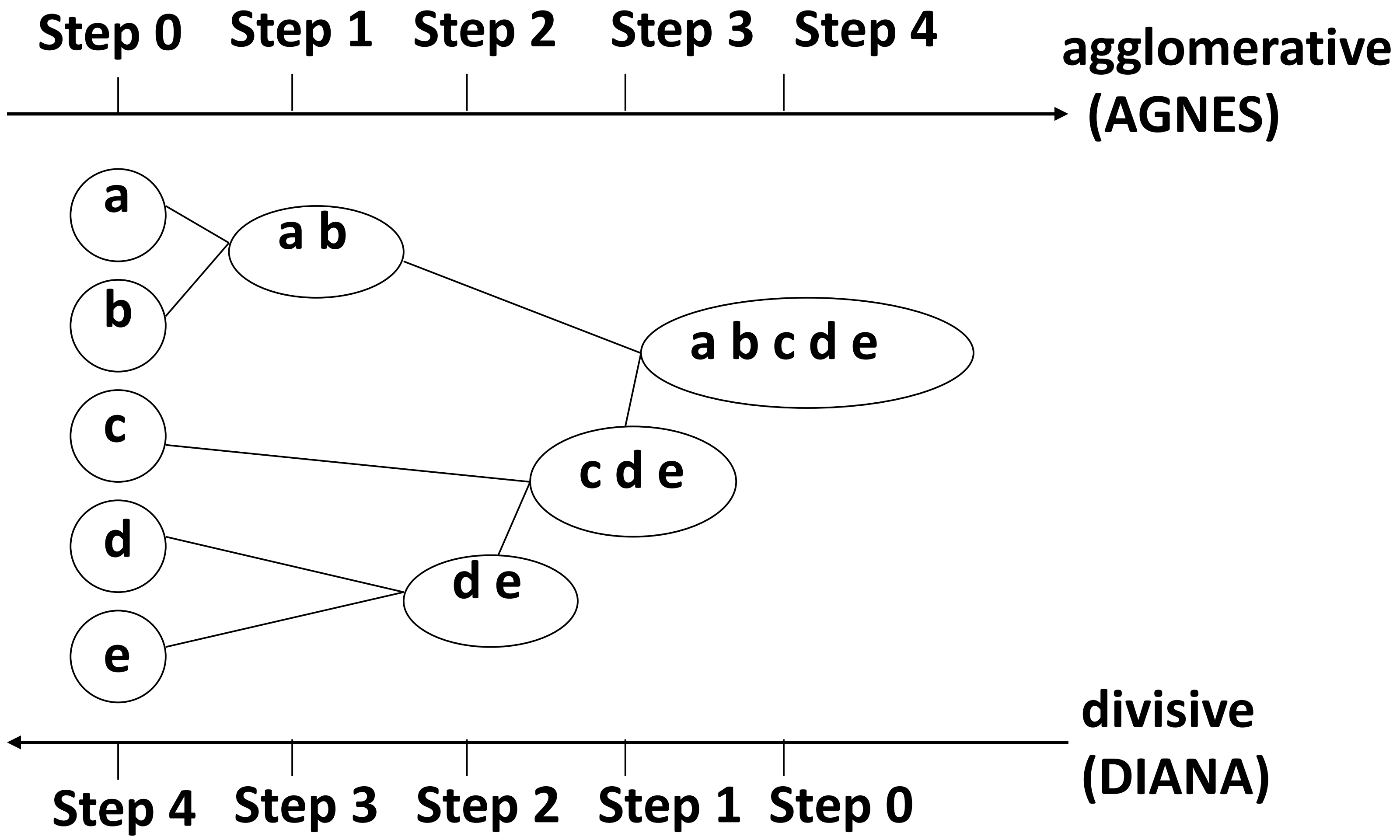
– Create a hierarchical decomposition of the set of data (or objects) using some Condition.

– Typical methods: Agnes, Diana, BIRCH, CAMELEON

– Use distance matrix as clustering criteria.

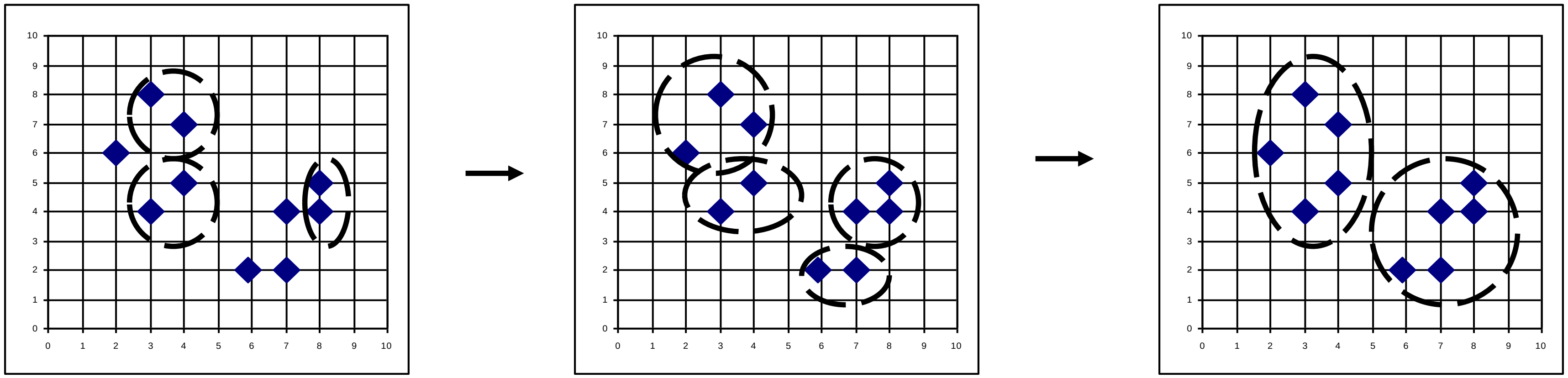
– This method does not require the number of clusters ***k*** as an input, but needs a termination condition.

Hierarchical Clustering



AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical packages, e.g., Splus.
- Use the **single-link** method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity.
- Go on in a non-descending fashion.
- Eventually all nodes belong to the same cluster.

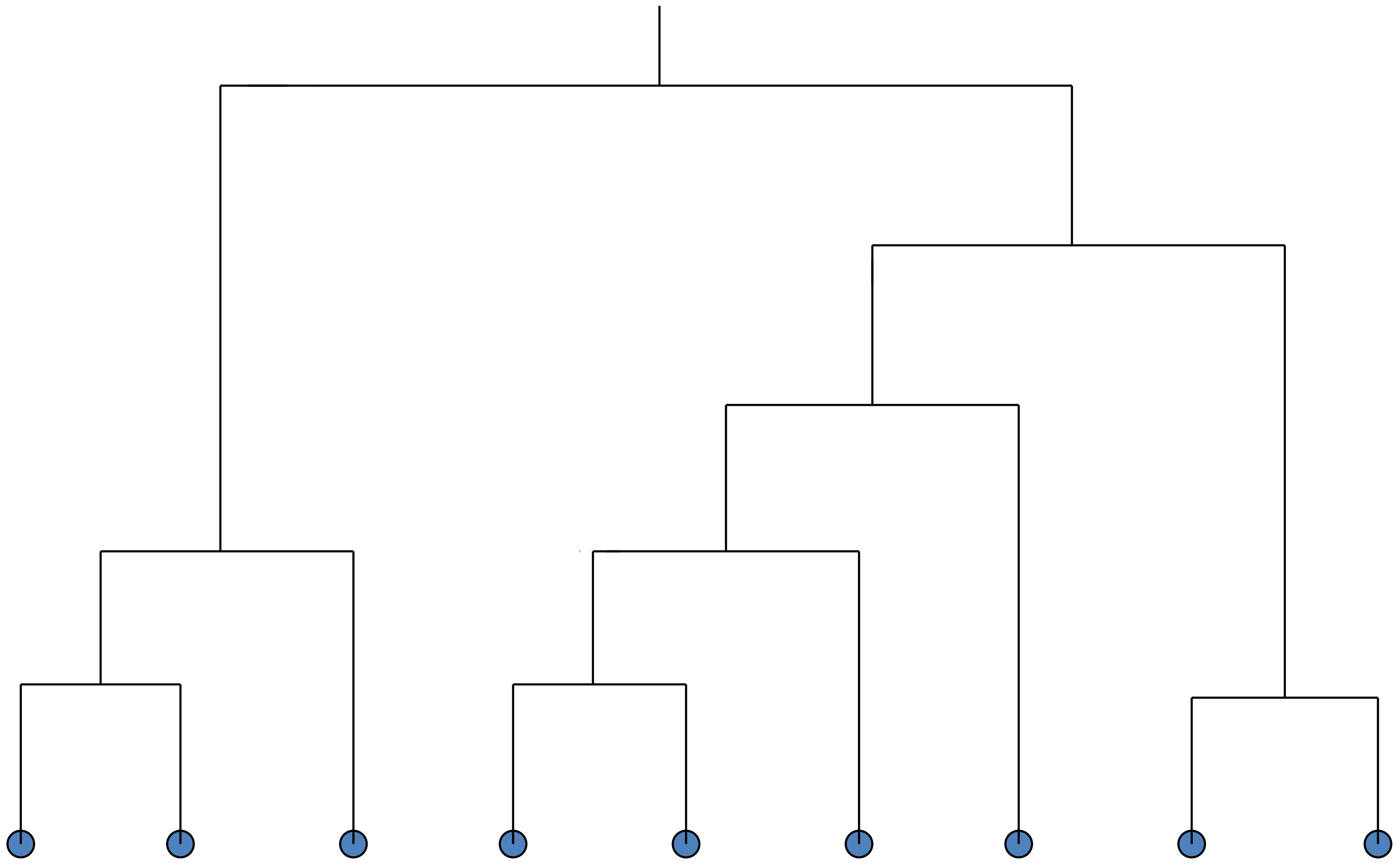


Dendrogram:

- Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram.

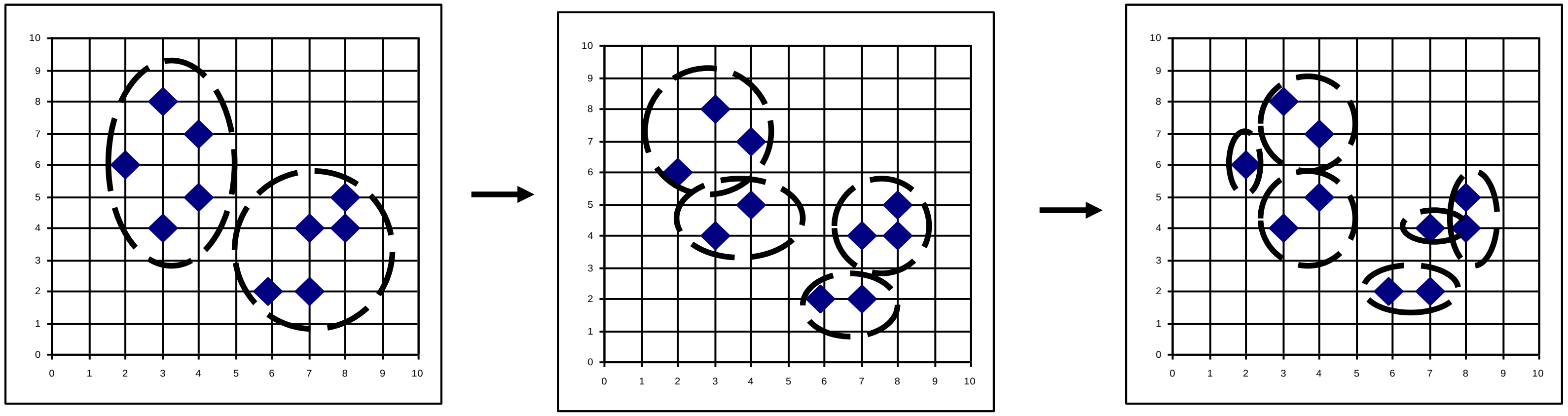
- A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.

Dendrogram: Shows How Clusters are Merged



DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990).
- Implemented in statistical analysis packages, e.g., Splus.
- Inverse order of AGNES.
- Eventually each node forms a cluster on its own.



Extensions to Hierarchical Clustering

- **Major weakness of agglomerative clustering methods**
  - Can never undo what was done previously.
  - Do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects.
- Integration of hierarchical & distance-based clustering
  - **BIRCH (1996):** uses CF-tree and incrementally adjusts the quality of sub-clusters.
  - **CHAMELEON (1999):** Hierarchical clustering using dynamic modeling.

BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

- Incrementally construct a CF **(Clustering Feature)** tree, a hierarchical data structure for multiphase clustering.
  - **Phase 1**: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - **Phase 2**: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- ***Scales linearly***: finds a good clustering with a single scan and improves the quality with a few additional scans.
- ***Weakness****:* handles only numeric data, and sensitive to the order of the data record.

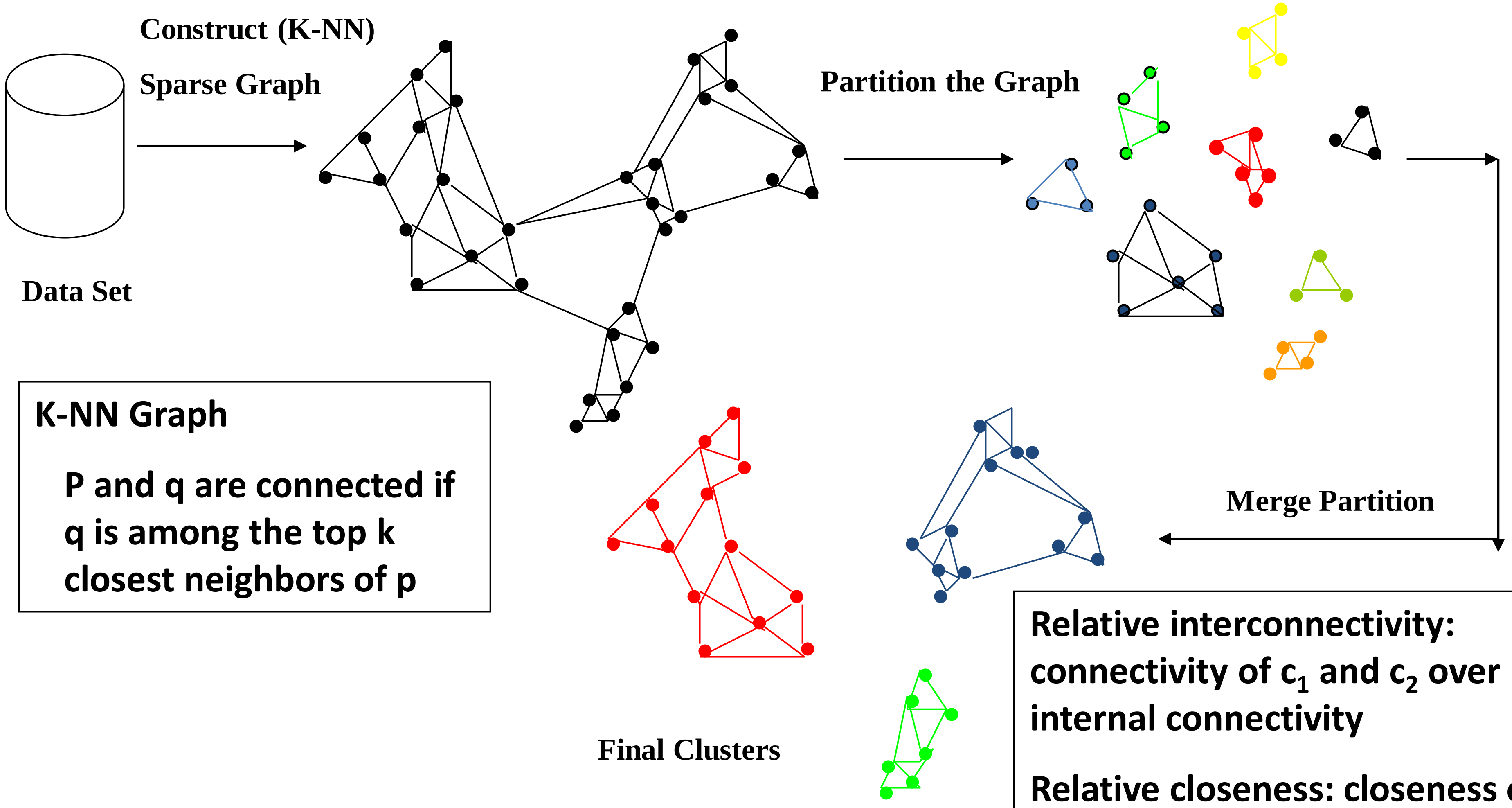
CF-Tree in BIRCH

- **Clustering Feature:**
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view.
  - Registers crucial measurements for computing cluster and utilizes storage efficiently.
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or “children”
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: max # of children.
  - Threshold: max diameter of sub-clusters stored at the leaf nodes.

CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- CHAMELEON: G. Karypis, E. H. Han, and V. Kumar, 1999 .
- Measures the similarity based on a dynamic model.
  - Two clusters are merged only if the *interconnectivity* and *closeness (proximity)* between two clusters are high *relative to* the internal interconnectivity of the clusters and closeness of items within the clusters .
- Graph-based, and a two-phase algorithm
  1. Use a graph-partitioning algorithm: cluster objects into a large number of relatively small sub-clusters.
  2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters.

Overall Framework of CHAMELEON



K-NN Graph

P and q are connected if q is among the top k closest neighbors of p

Relative interconnectivity: connectivity of c_1 and c_2 over internal connectivity

Relative closeness: closeness of c_1 and c_2 over internal closeness

3. Density Based Methods:

– Based on connectivity and density functions.

– Typical methods: DBSACN, OPTICS, DenClue

Clustering based on density (local cluster condition), such as density-connected points.

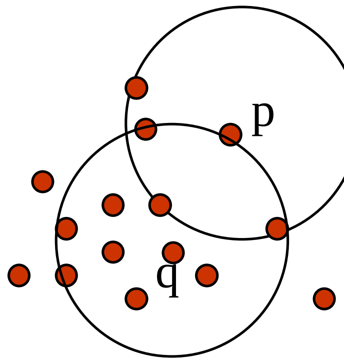
Major Features:

- **Discover clusters of arbitrary shape**
- **Handle noise**
- **One scan**
- **Need density parameters as termination condition**

Density-Based Clustering: Basic Concepts

- **Two parameters*:***
  - *Eps*: Maximum radius of the neighbourhood.
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point.
- $N_{Eps}(p)$: {q belongs to D | dist(p,q) ≤ Eps}
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if
  - *p* belongs to $N_{Eps}(q)$
  - core point condition:

$$|N_{Eps}(q)| \geq MinPts$$

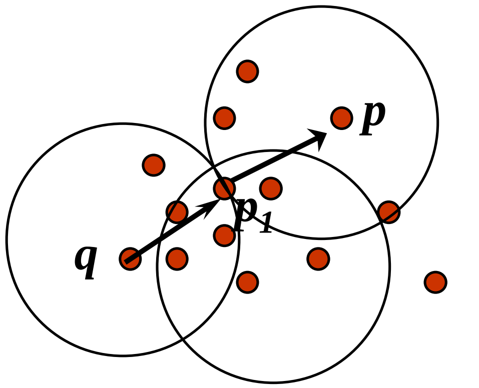


MinPts = 5

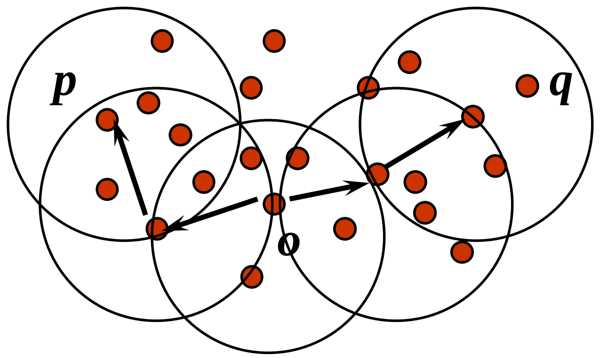
Eps = 1 cm

Density-Reachable and Density-Connected

- **Density-reachable:**
  - **A point *p* is density-reachable** from a point *q* w.r.t. *Eps*, *MinPts* if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i

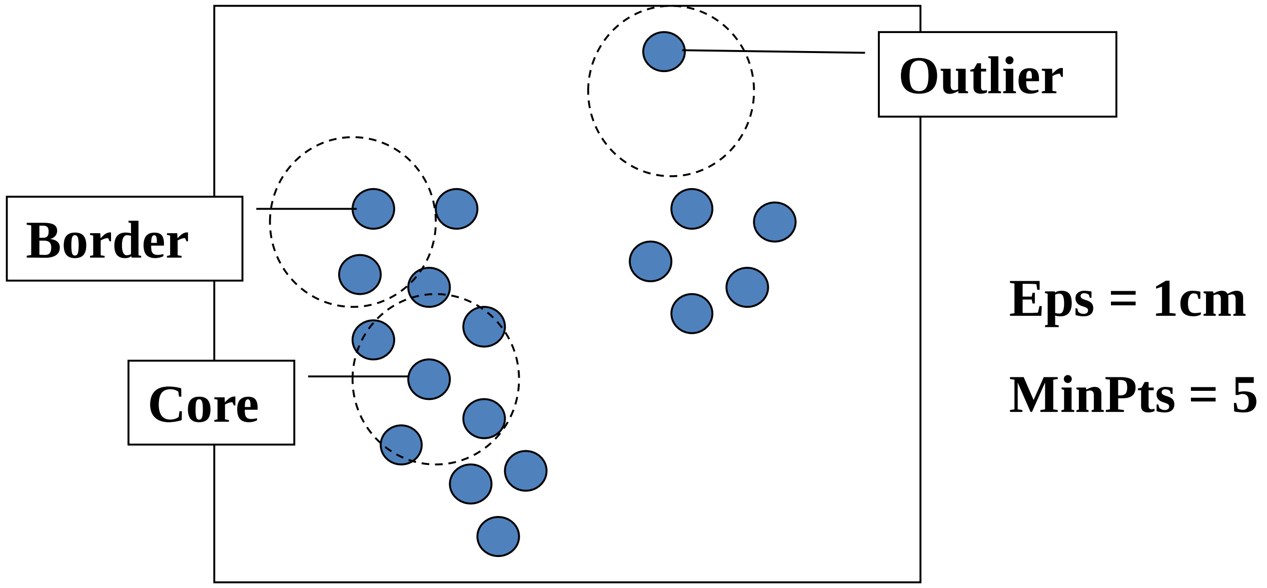


- **Density-connected :**
  - **A point *p* is density-connected** to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points.
- Discovers clusters of arbitrary shape in spatial databases with noise.



DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* w.r.t. *Eps* and *MinPts*
- If *p* is a core point, a cluster is formed.
- If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

DBSCAN: Sensitive to Parameters

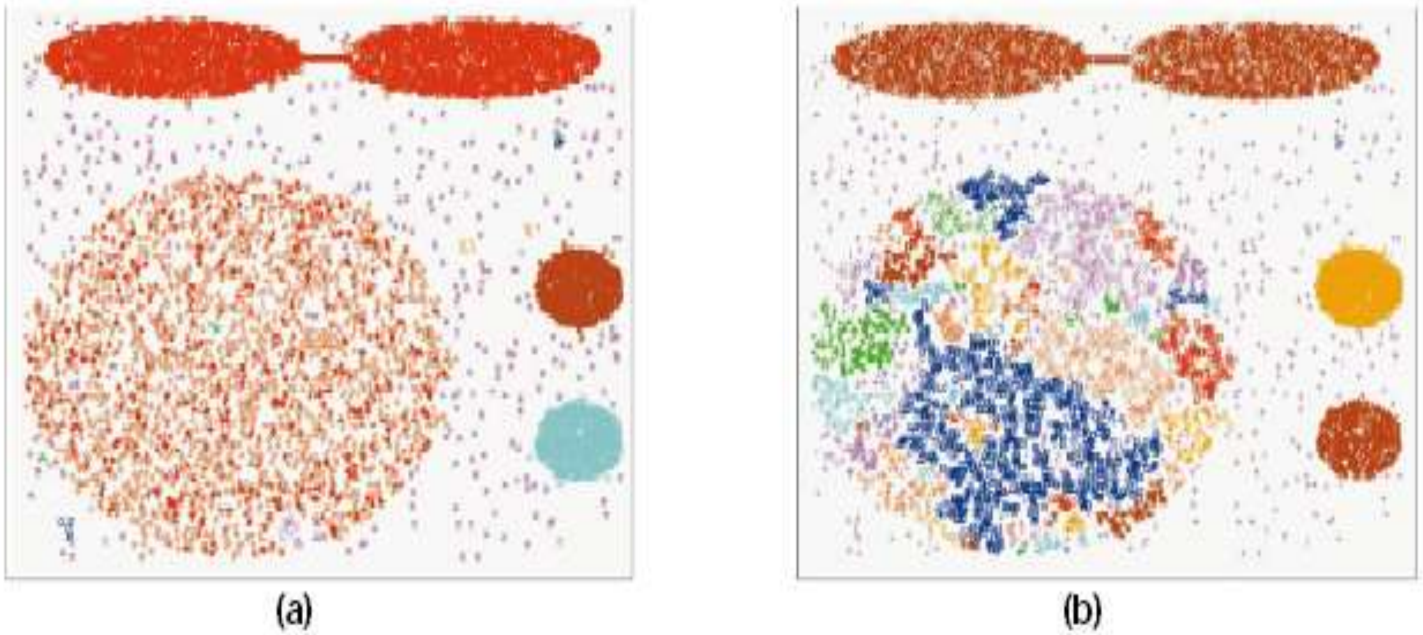


Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

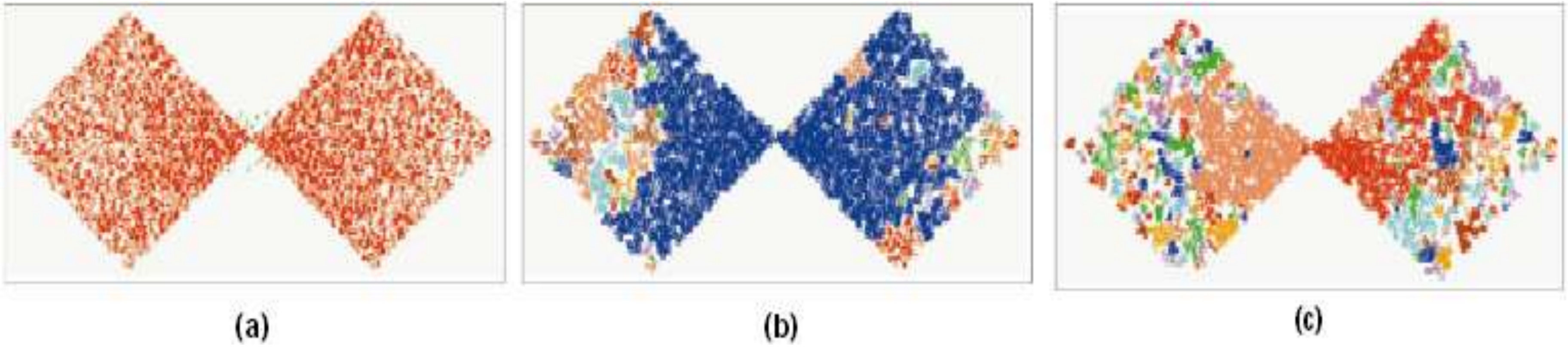
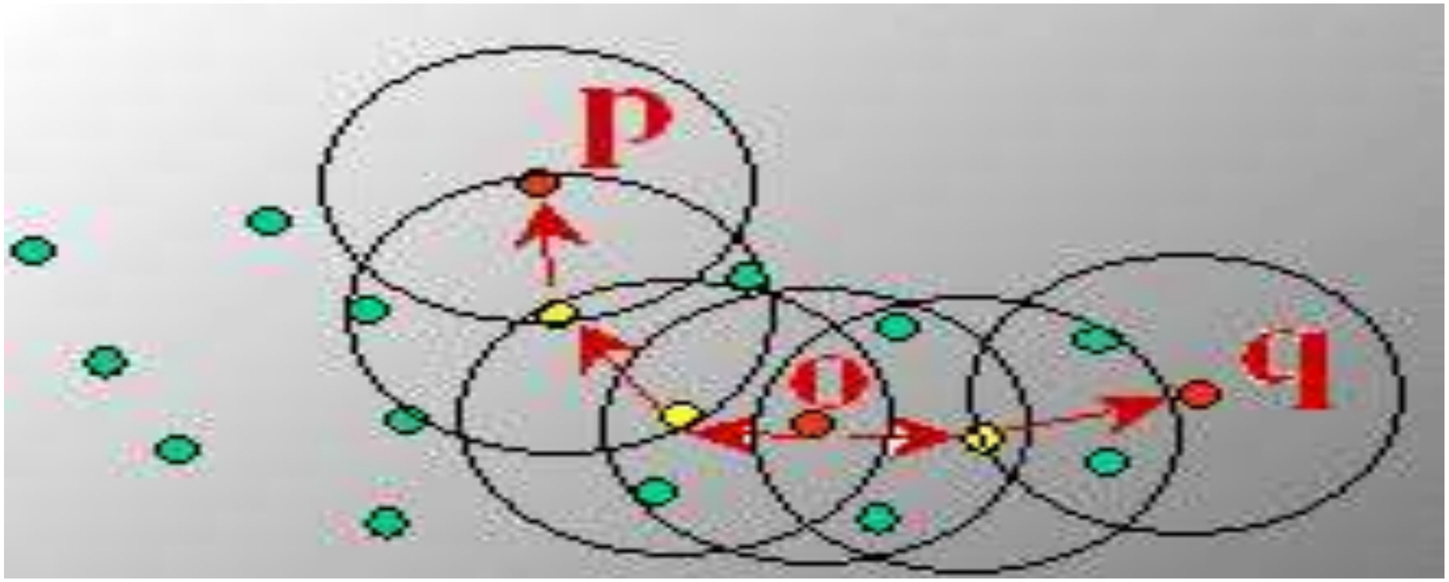


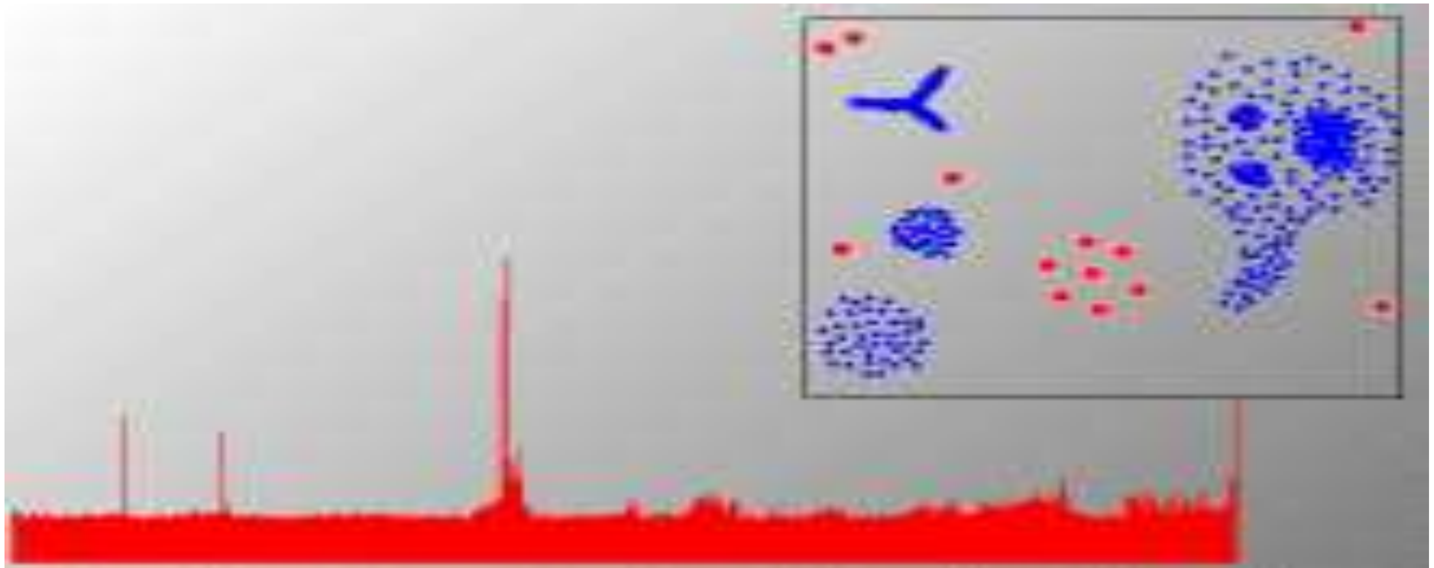
Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

OPTICS: A Cluster-Ordering Method (1999)

- **OPTICS: Ordering Points To Identify the Clustering Structure**
  - Produces a special order of the database write its density-based clustering structure.
  - This cluster-ordering contains info equivalent to the density-based clusterings corresponding to a broad range of parameter settings.
  - Good for both automatic and interactive cluster analysis, including finding **intrinsic clustering structure**.
  - Can be represented graphically or using visualization techniques.

Density-Based Clustering: OPTICS & Its Applications





DENCLUE: Using Statistical Density Functions

- **DENsity-based CLUstEring** by Hinneburg & Keim (KDD'98)
- Using statistical density functions:
- **Major features:**
  - Solid mathematical foundation
  - Good for data sets with large amounts of noise
  - Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets
  - Significant faster than existing algorithm (e.g., DBSCAN)
  - But needs a large number of parameters

4. Grid Based Methods:

- Based on a multiple-level granularity structure.
- Typical methods: STING, WaveCluster, CLIQUE

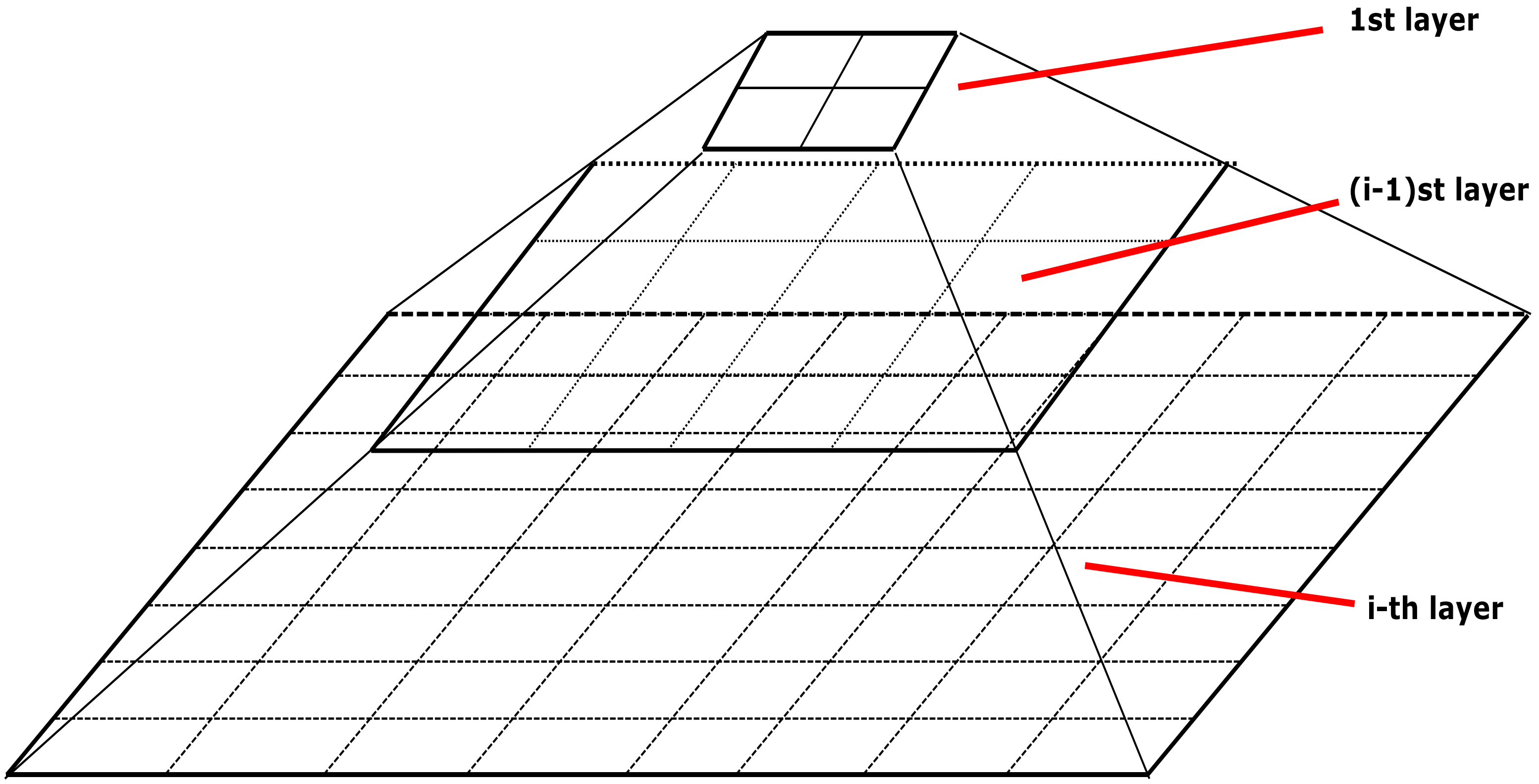
- Using multi-resolution grid data structure
- Several interesting methods are:
  - STING (a STatistical INformation Grid approach)
  - WaveCluster
    - A multi-resolution clustering approach using wavelet method
  - CLIQUE:
    - Both grid-based and subspace clustering

The STING Clustering Method

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level.
- Statistical info of each cell is calculated and stored before hand and is used to answer queries.
- 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.
- Use a top-down approach to answer spatial data queries.
- Start from a pre-selected layer—typically with a small number of cells.
- For each cell in the current level compute the confidence interval.

STING: A Statistical Information Grid Approach

- The spatial area is divided into rectangular cells.
- There are several levels of cells corresponding to different levels of resolution.

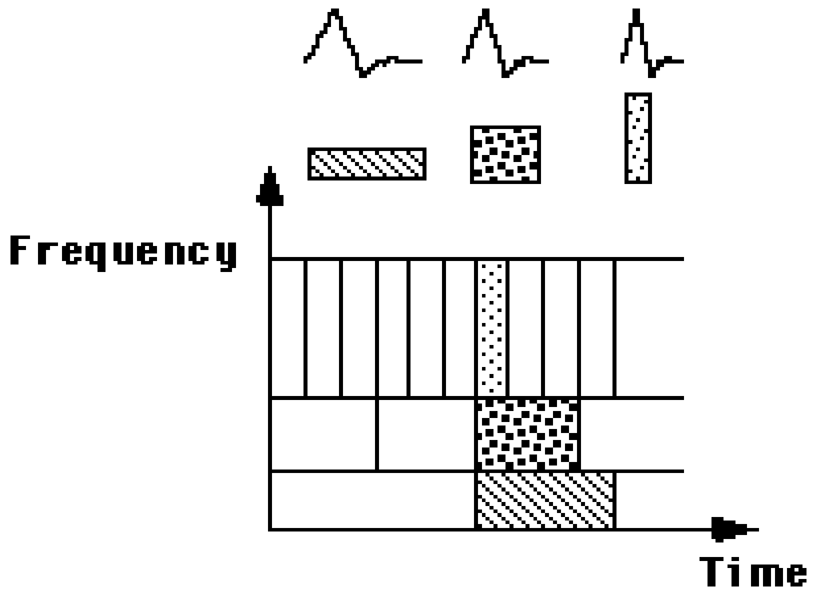


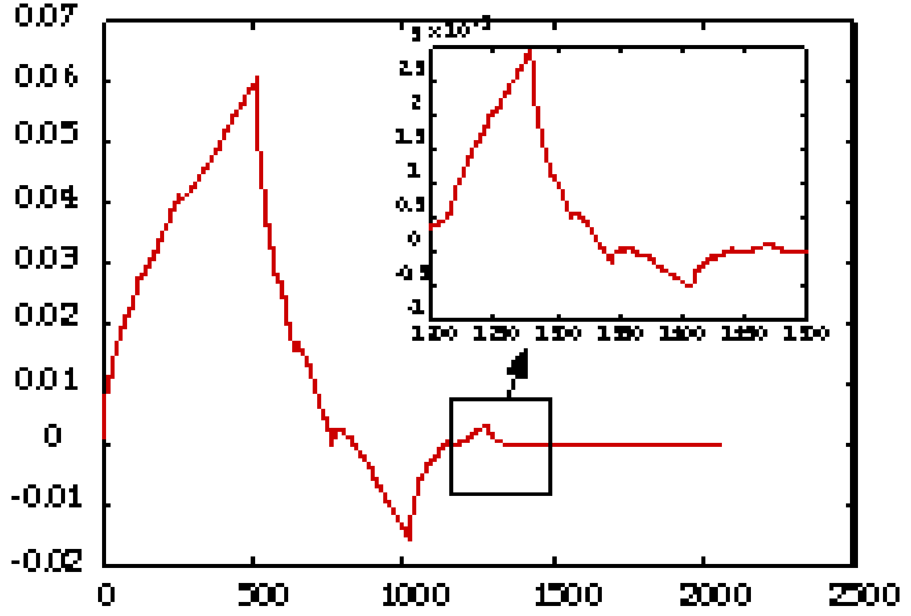
STING Algorithm and Its Analysis

- Remove the irrelevant cells from further consideration.
- When finish examining the current layer, proceed to the next lower level.
- Repeat this process until the bottom layer is reached.
- **Advantages:**
  - 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: Clustering by Wavelet Analysis (1998)

- A multi-resolution clustering approach which applies wavelet transform to the feature space; both grid-based and density-based.
- **Wavelet transform:** A signal processing technique that decomposes a signal into different frequency sub-band.
  - Data are transformed to preserve relative distance between objects at different levels of resolution.
  - Allows natural clusters to become more distinguishable.





The WaveCluster Algorithm

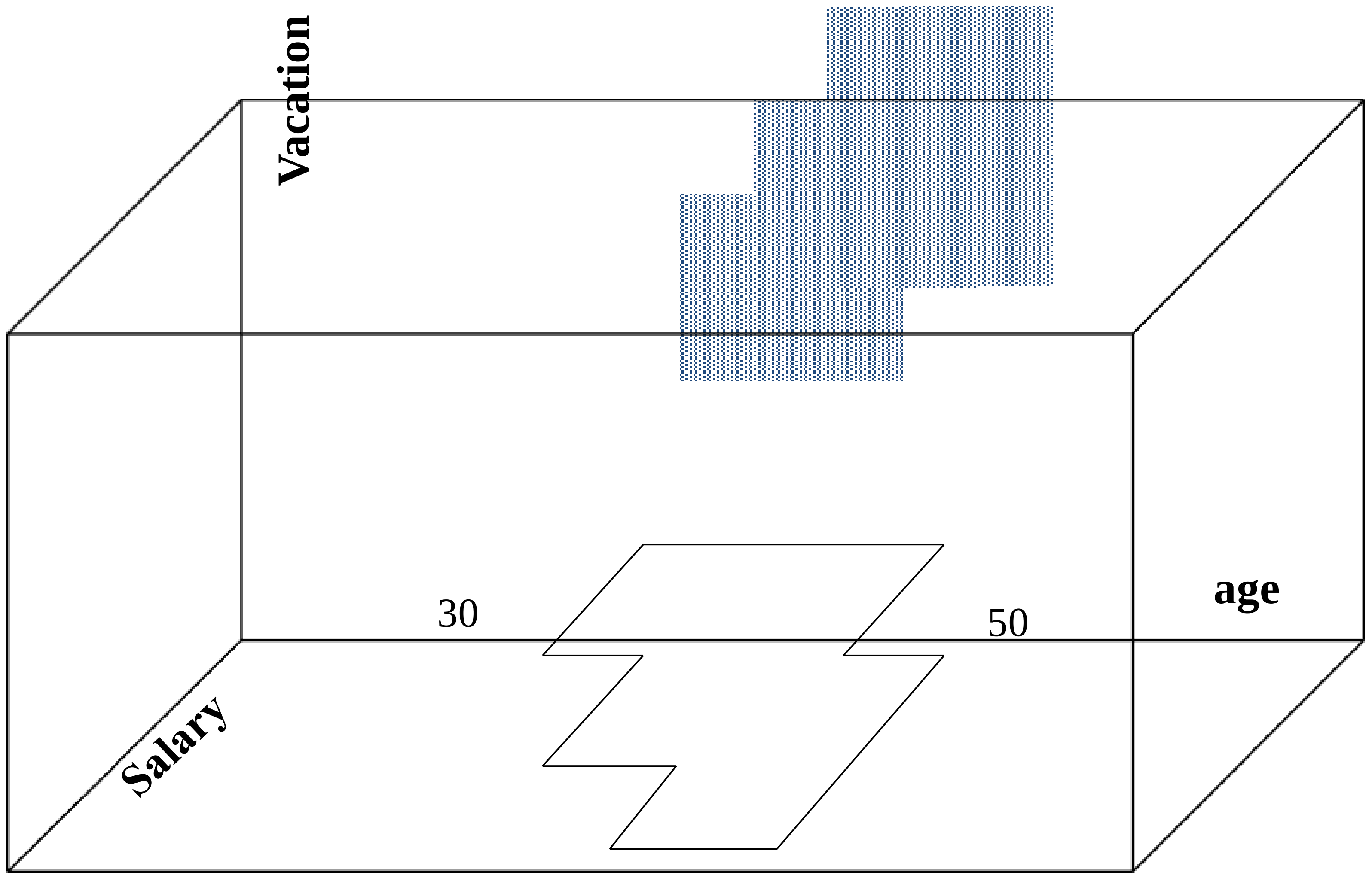
- **How to apply wavelet transform to find clusters**
  - Summarizes the data by imposing a multidimensional grid structure onto data space.
  - These multidimensional spatial data objects are represented in a n-dimensional feature space.
  - Apply wavelet transform on feature space to find the dense regions in the feature space.
  - Apply wavelet transform multiple times which result in clusters at different scales from fine to coarse .
- **Major Features:**
  - Complexity O(N)
  - Detect arbitrary shaped clusters at different scales.
  - Not sensitive to noise, not sensitive to input order.
  - Only applicable to low dimensional data.

CLIQUE (CLustering In QUEst)

- Automatically identifying subspaces of a 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 interval.
  - It partitions an m-dimensional data space into non-overlapping rectangular units.
  - A unit is dense if the fraction of 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.

CLIQUE: The Major Steps

- 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.


Vacation
30
50
age
Salary

Strength and Weakness of *CLIQUE*

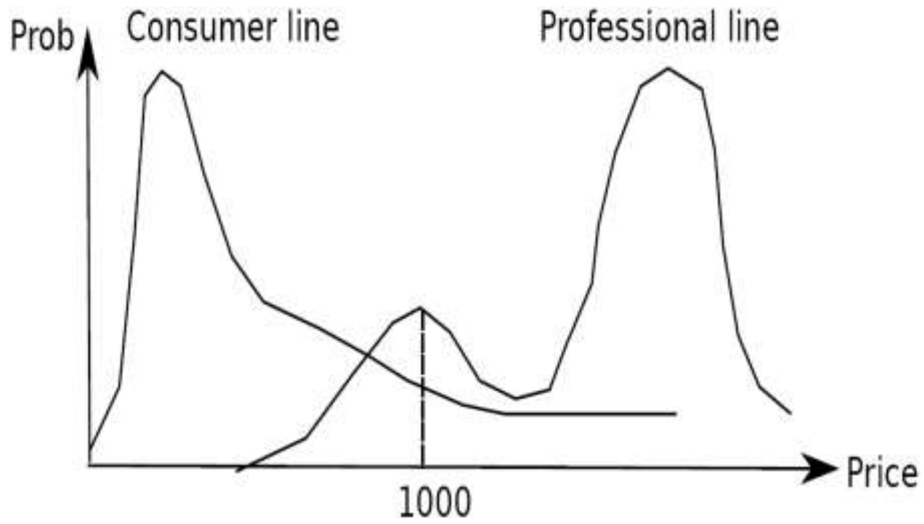
- Strength :
  - *Automatically finds subspaces of the highest dimensionality* such that high density clusters exist in those subspaces.
  - *Insensitive* to the order of records in input and does not presume some canonical data distribution.
  - Scales *linearly* with the size of input and has good scalability as the number of dimensions in the data increases.
- Weakness :
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method.

5. Model-Based Clustering Methods:

- A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other.
- Typical methods: EM, SOM, COBWEB

- Cluster analysis is to find hidden categories.
- A hidden category (i.e., *probabilistic cluster)* is a distribution over the data space, which can be mathematically represented using a probability density function (or distribution function).

Example :



- Ex. 2 categories for digital cameras sold
  - consumer line vs. professional line
  - density functions f_1, f_2 for C_1, C_2
  - obtained by probabilistic clustering

- A **mixture model** assumes that a set of observed objects is a mixture of instances from multiple probabilistic clusters, and conceptually each observed object is generated independently
- **Out task**: infer a set of *k* probabilistic clusters that is mostly likely to generate *D* using the above data generation process

The EM (Expectation Maximization) Algorithm

- **The k-means algorithm has two steps at each iteration:**
  - **Expectation Step** (E-step): Given the current cluster centers, each object is assigned to the cluster whose center is closest to the object: An object is *expected to belong to the closest cluster.*
  - **Maximization Step** (M-step): Given the cluster assignment, for each cluster, the algorithm *adjusts the center* so that *the sum of distance* from the objects assigned to this cluster and the new center is minimized.
- **The (EM) algorithm:** A framework to approach maximum likelihood or maximum a posteriori estimates of parameters in statistical models.
  - **E-step** assigns objects to clusters according to the current fuzzy clustering or parameters of probabilistic clusters.
  - **M-step** finds the new clustering or parameters that maximize the sum of squared error (SSE) or the expected likelihood.

Computing Mixture Models with EM

- **Given n objects O = {o_1, …, o_n}, we want to mine a set of parameters Θ = {θ_1, …, θ_k} s.t.,P(O|Θ) is maximized, where θ_j = (μ_j, σ_j) are the mean and standard deviation of the j-th univariate Gaussian distribution.**
- **We initially assign random values to parameters θ_j, then iteratively conduct the E- and M- steps until converge or sufficiently small change.**
- **At the E-step, for each object o_i, calculate the probability that o_i belongs to each distribution,**

$$P(\Theta_j|o_i,\mathbf{\Theta}) = \frac{P(o_i|\Theta_j)}{\sum_{l=1}^{k} P(o_i|\Theta_l)}$$

- At the M-step, adjust the parameters θ_j = (μ_j, σ_j) so that the expected likelihood P(**O**|**Θ**) is maximized.

$$\mu_j = \sum_{i=1}^{n} o_i \frac{P(\Theta_j|o_i,\mathbf{\Theta})}{\sum_{l=1}^{n} P(\Theta_j|o_l,\mathbf{\Theta})} = \frac{\sum_{i=1}^{n} o_i P(\Theta_j|o_i,\mathbf{\Theta})}{\sum_{i=1}^{n} P(\Theta_j|o_i,\mathbf{\Theta})} \qquad \sigma_j = \sqrt{\frac{\sum_{i=1}^{n} P(\Theta_j|o_i,\mathbf{\Theta})(o_i - u_j)^2}{\sum_{i=1}^{n} P(\Theta_j|o_i,\mathbf{\Theta})}}$$

Advantages and Disadvantages

- **Advantages**
  - Mixture models are more general than partitioning and fuzzy clustering .
  - Clusters can be characterized by a small number of parameters.
  - The results may satisfy the statistical assumptions of the generative models.
- **Disadvantages**
  - Converge to local optimal (overcome: run multi-times w. random initialization)
  - Computationally expensive if the number of distributions is large, or the data set contains very few observed data points.
  - Need large data sets.
  - Hard to estimate the number of clusters.

Self-Organizing Feature Map (SOM)

- **SOMs, also called topological ordered maps, or Kohonen Self-Organizing Feature Map (KSOMs)**
- It maps all the points in a high-dimensional source space into a 2 to 3-d target space, s.t., the distance and proximity relationship (i.e., topology) are preserved as much as possible.
- Similar to k-means: cluster centers tend to lie in a low-dimensional manifold in the feature space.
- Clustering is performed by having several units competing for the current object
  - The unit whose weight vector is closest to the current object wins
  - The winner and its neighbors learn by having their weights adjusted
- SOMs are believed to resemble processing that can occur in the brain
- Useful for visualizing high-dimensional data in 2- or 3-D space.

COBWEB

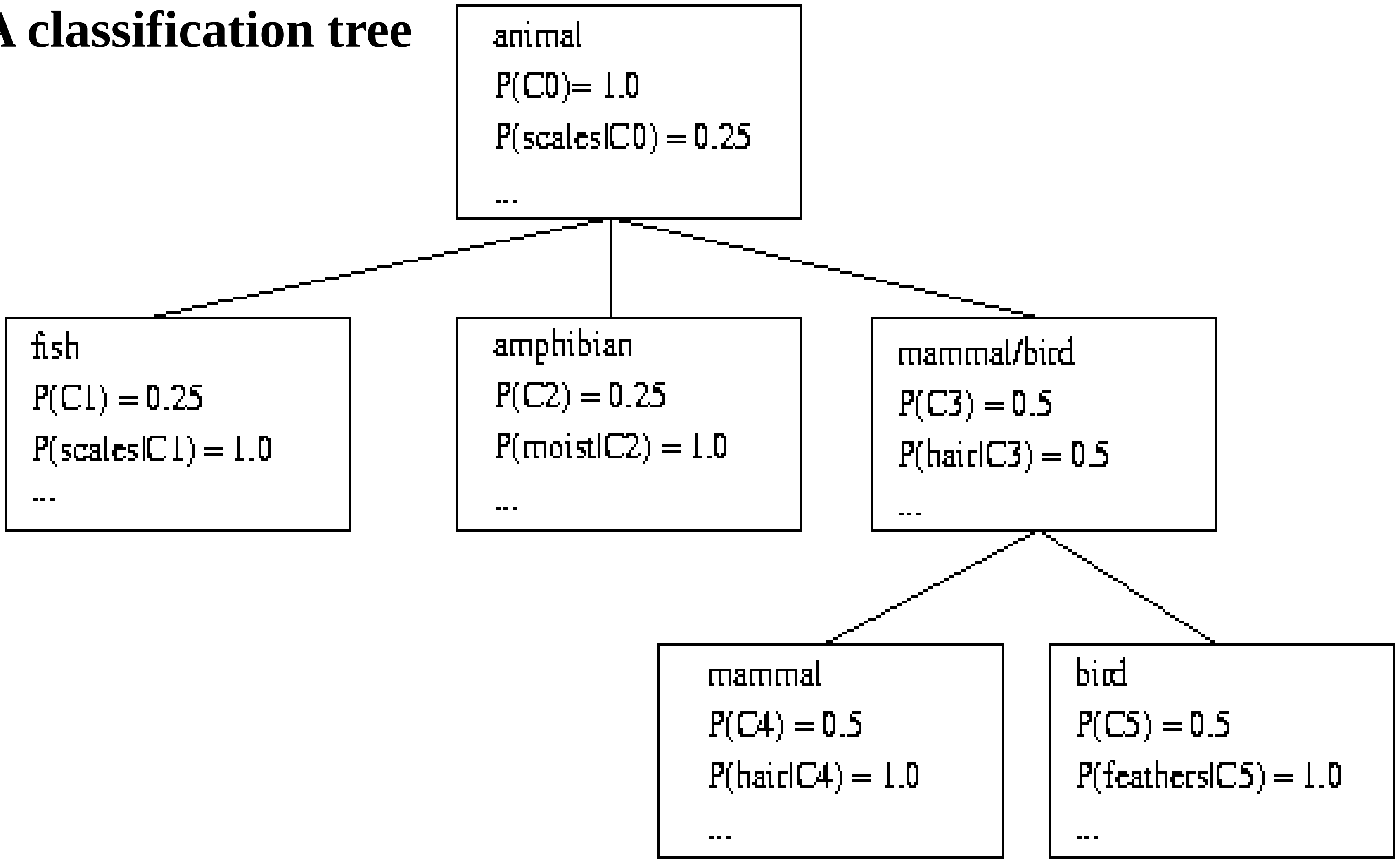
- **COBWEB**
  - A popular a simple method of incremental conceptual learning.
  - Creates a hierarchical clustering in the form of a classification tree.
  - Each node refers to a concept and contains a probabilistic description of that concept.

Limitations of COBWEB

1. The assumption that the attributes are independent of each other is often too strong because correlation may exist.
2. Not suitable for clustering large database data – skewed tree and expensive probability distributions.

COBWEB Clustering Method

A classification tree



Clustering High - Dimensional Data

- Clustering high-dimensional data (How high is high-D in clustering?)
  - **Many applications**: text documents, DNA micro-array data.
  - **Major challenges**:
    - Many irrelevant dimensions may mask clusters.
    - Distance measure becomes meaningless—due to equi-distance.
    - Clusters may exist only in some subspaces.

Clustering High-Dimensional Data

- METHODS
  - **Subspace-clustering**: Search for clusters existing in subspaces of the given high dimensional data space.
    - **CLIQUE, ProClus, and bi-clustering approaches.**
  - **Dimensionality reduction approaches**: Construct a much lower dimensional space and search for clusters there (may construct new dimensions by combining some dimensions in the original data)
    - Dimensionality reduction methods and spectral clustering.

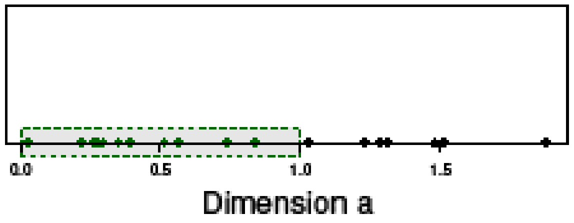
EXAMPLE

- Traditional distance measure could be dominated by noises in many dimensions.
- Ex. Which pairs of customers are more similar?

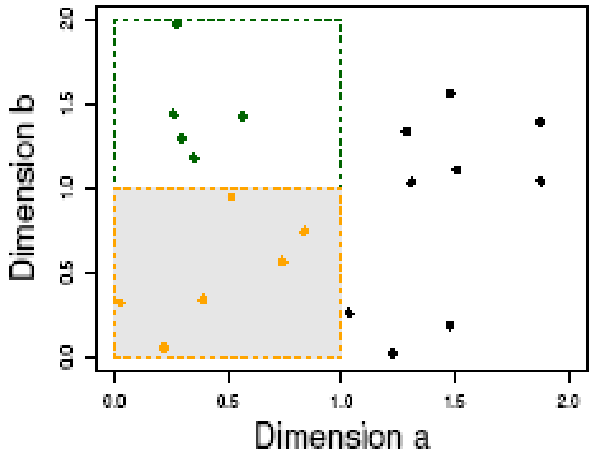
Customer	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
Ada	1	0	0	0	0	0	0	0	0	0
Bob	0	0	0	0	0	0	0	0	0	1
Cathy	1	0	0	0	1	0	0	0	0	1

The Curse of Dimensionality

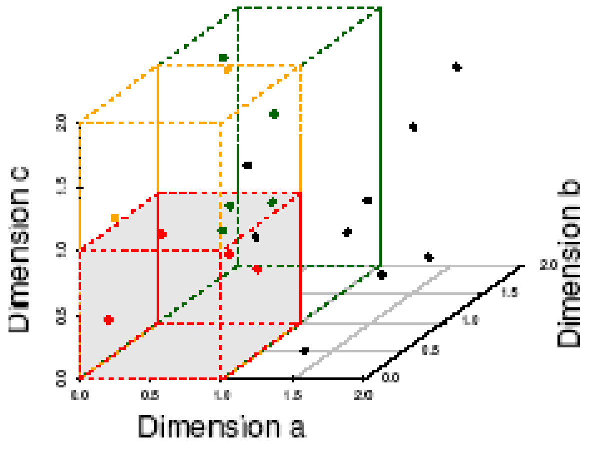
- Data in only one dimension is relatively packed.
- Adding a dimension "stretch" the points across that dimension, making them further apart.
- Adding more dimensions will make the points further apart—high dimensional data is extremely sparse.
- Distance measure becomes meaningless—due to equi-distance.



(a) 11 Objects in One Unit Bin



(b) 6 Objects in One Unit Bin



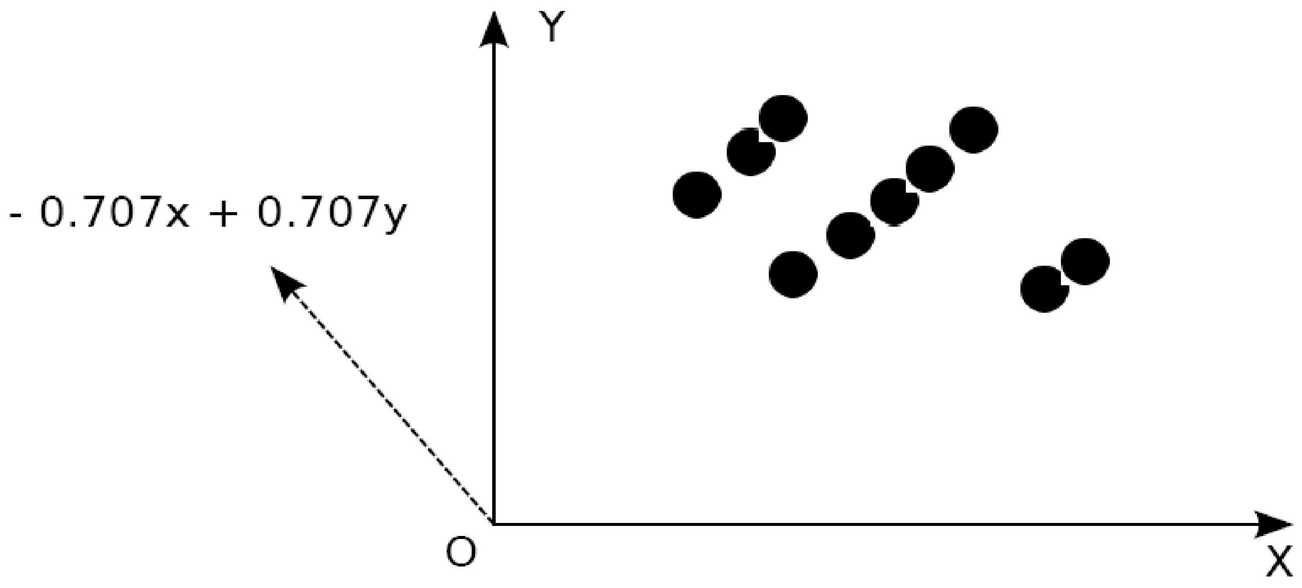
(c) 4 Objects in One Unit Bin

Subspace High-Dimensional Clustering Methods

- **Subspace search methods: Search various subspaces to find clusters**
  - Bottom-up approaches
  - Top-down approaches
- **Correlation-based clustering methods**
  - E.g., PCA based approaches
- **Bi-clustering methods**
  - Optimization-based methods
  - Enumeration methods

Dimensionality-Reduction Methods

- Dimensionality reduction: In some situations, it is more effective to construct a new space instead of using some subspaces of the original data.



EXAMPLE

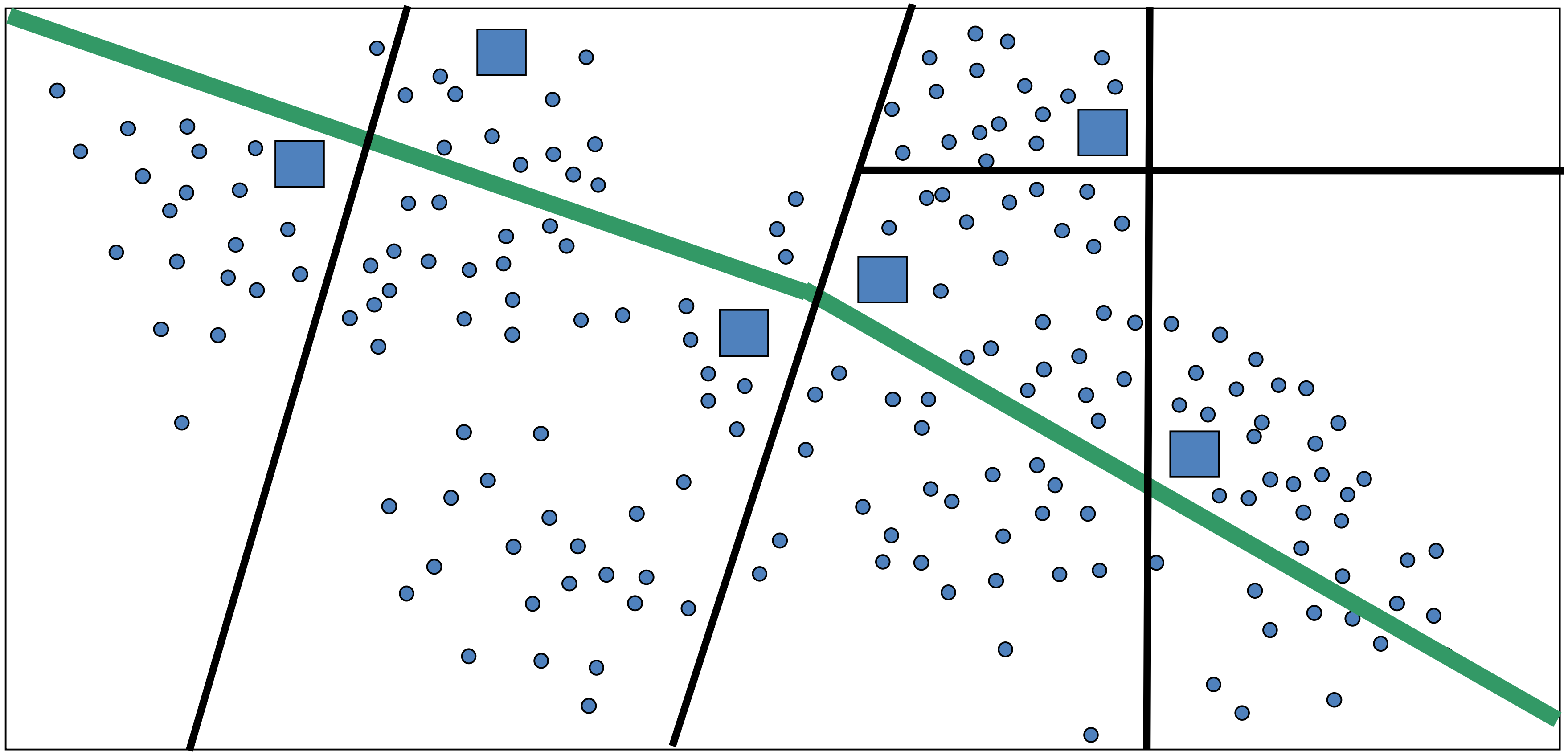
- To cluster the points in the right figure, any subspace of the original one, X and Y, cannot help, since all the three clusters will be projected into the overlapping areas in X and Y axes.
  - Construct a new dimension as the dashed one, the three clusters become apparent when the points projected into the new dimension.
- Dimensionality reduction methods
  - **Feature selection and extraction**: But may not focus on clustering structure finding.
  - **Spectral clustering**: Combining feature extraction and clustering (i.e., use the *spectrum* of the similarity matrix of the data to perform dimensionality reduction for clustering in fewer dimensions)
    - Normalized Cuts (Shi and Malik, CVPR'97 or PAMI'2000)
    - The Ng-Jordan-Weiss algorithm (NIPS'01)

Constraint-Based Cluster Analysis

- 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.

Why Constraint-Based Cluster Analysis?

- Need user feedback: Users know their applications the best.
- Less parameters but more user-desired constraints, e.g., an ATM allocation problem: obstacle & desired clusters.



Categorization of Constraints

- **Constraints on instances**: specifies how a pair or a set of instances should be grouped in the cluster analysis.
  - Must-link vs. cannot link constraints
    - must-link(x, y): x and y should be grouped into one cluster
  - Constraints can be defined using variables, e.g.,
    - cannot-link(x, y) if dist(x, y) > d
- **Constraints on clusters**: specifies a requirement on the clusters.
  - E.g.,specify the min # of objects in a cluster, the max diameter of a cluster, the shape of a cluster(e.g., a convex), # of clusters (e.g., k)
- **Constraints on similarity measurements**: specifies a requirement that the similarity calculation must respect.
  - E.g., driving on roads, obstacles (e.g., rivers, lakes)

Constraint-Based Clustering Methods (I) Handling Hard Constraints

- **Handling hard constraints**: Strictly respect the constraints in cluster assignments.
- **Example**: The COP-k-means algorithm
  - Generate super-instances for **must-link constraints.**
    - Compute the transitive closure of the must-link constraints.
    - To represent such a subset, replace all those objects in the subset by the mean.
    - The super-instance also carries a weight, which is the number of objects it represents.
  - Conduct modified k-means clustering to respect **cannot-link constraints.**
    - Modify the center-assignment process in k-means to a *nearest feasible center assignment.*
    - An object is assigned to the nearest center so that the assignment respects all cannot-link constraints.

Constraint-Based Clustering Methods (II) Handling Soft Constraints

- **Treated as an optimization problem**: When a clustering violates a soft constraint, a penalty is imposed on the clustering.
- **Overall objective**: Optimizing the clustering quality, and minimizing the constraint violation penalty.
- **Ex** : CVQE (Constrained Vector Quantization Error)
- **Algorithm**: Conduct k-means clustering while enforcing constraint violation penalties.
- **Objective function**: Sum of distance used in k-means, adjusted by the constraint violation penalties.

Outlier Analysis

- If the data objects does not belongs to any group it should be named as Outlier Analysis.
- An **outlier** is a data point that differs significantly from other observations.
- An outlier may be due to variability in the measurement or it may indicate experimental error.
- Outliers can occur by chance in any distribution, but they often indicate either measurement error or that the population has a heavy-tailed distribution.
- outlier analysis tries to find unusual patterns in any dataset.
- Most data mining methods discard outliers as noise or exceptions.

- The analysis of outlier data is referred to as outlier mining.
- Outliers may be detected using **statistical tests** that assume a distribution or probability model for the data.
- Many data mining algorithms try to minimize the influence of outliers or eliminate them all together.
- Outlier detection and analysis is an interesting data mining task.

Methods of Outlier Detection

- **There are four methods in outlier detection**.

  1. The statistical approach
  2. The distance-based approach
  3. The density-based local outlier approach
  4. The deviation-based approach

1. The Statistical Approach

- This approach assumes a distribution for the given data set and then identifies outliers with respect to the model using a discordancy test.
- A statistical discordancy test examines **Two Hypotheses**:

1. **A working hypothesis and**
2. **An alternative hypothesis.**

- **A working hypothesis, H**, is a statement that the entire data set of n objects comes from an initial distribution model, F, that is H:o_i ϵ F, where i= 1,2,....,n.
- **An alternative hypothesis, H,** which states that oi comes from another distribution model (G), is adopted.
- The result is very much dependent on which model is chosen because **oi** may be an outlier under one model and a perfectly valid value under another.

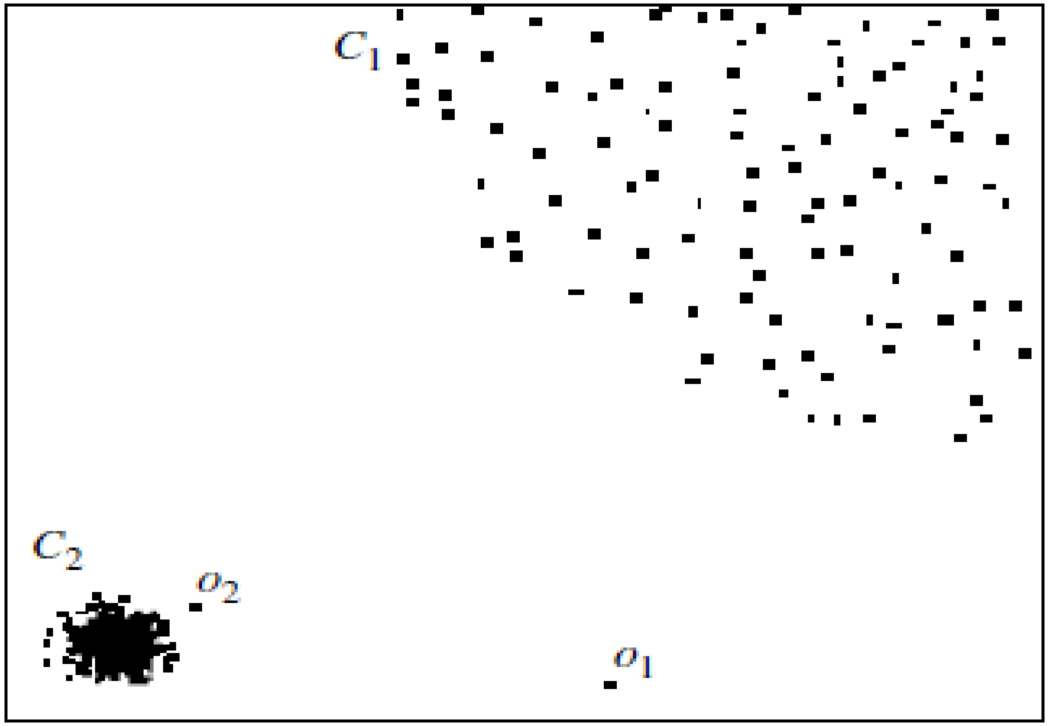
2. The Distance-Based Approach

- This approach generalizes the ideas behind discordancy testing for various standard distributions and its neighbors are defined based on their distance from the given object.
- Several efficient algorithms for mining distance-based outliers have been developed.

1. **Index-based algorithm**
2. **Nested-loop algorithm**
3. **Cell-based algorithm**

3. The Density-Based Local Outlier Approach

- Distance based outlier detection faces difficulty in identifying outliers if **data is not uniformly distributed**.
- Therefore this approach is used which depends on the overall distribution of the given set of data points.
- **Eg**: The below Figure shows a simple **2-D data set** containing **502 objects, with two obvious clusters.**
- Cluster **C1 contains 400 objects** while **Cluster C2 contains 100 objects.**
- **Two additional objects, o1 and o2 are clearly outliers.**



However, by **distance-based** outlier detection **o1 & o2** are the reasonable outliers.

4. The Deviation-Based Approach

- This approach identifies outliers by examining the main characteristics of objects in a group.
- Objects that "**deviate**" from this description are considered outliers.
- Hence, in this approach the term deviation is typically used to refer to outliers.
- There are **Two techniques** for deviation-based outlier detection.

    1. Sequentially compares objects in a set.

    2. An OLAP Data Cube Approach.

UNIT V

UNIT V DATA WAREHOUSE

Need for Data Warehouse – Database versus Data Warehouse – Multidimensional Data Model – Schemas for Multidimensional Databases – OLAP operations – OLAP versus OLTP– Data Warehouse Architecture .

OVERVIEW

- Definition
- Types
- Components
- Architecture
- Database Design
- OLAP & OLTP
- Metadata repository

OLTP vs. Warehousing

- Organized by transactions vs. Organized by particular subject
- More number of users vs. less
- Accesses few records vs. entire table
- Smaller database vs. Large database
- Normalised data structure vs. Unnormalized
- Continuous update vs. periodic update

Definition

- A datawarehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of managements decision making process.
- It is the process whereby organizations extract value from their informational assets through use of special stores called data warehouses

Types

- Operational Data Store: Operational data mirror. Eg: Item in stock.
- Enterprise data warehouse: Historical analysis, Complex pattern analysis.
- Data Marts

Uses of a datawarehouse

- Presentation of standard reports and graphs
- For dimensional analysis
- Data mining

Advantages

- Lowers cost of information access
- Improves customer responsiveness
- Identifies hidden business opportunities
- Strategic decision making

Roadmap to DataWarehousing

- Data extracted, transformed and cleaned
- Stored in a database - RDBMS, MDD
- Query and Reporting systems
- Executive Information System and Decision Support System

Data Extraction and Load

- Find sources of data : Tables, files, documents, commercial databases, emails, Internet
- Bad data Quality: Same name but different things, Different Units
- Tool to clean data - Apertus
- Tool to convert codes, aggregate and calculate derived values - SAS
- Data Reengineering tools

Metadata

- Database that describes various aspects of data in the warehouse
- Administrative Metadata: Source database and contents, Transformations required, History of Migrated data
- End User Metadata:

  Definition of warehouse data

  Descriptions of it

  Consolidation Hierarchy

Storage

- Relational databases
- MDD

  Measurements are numbers that quantify the business process

  Dimensions are attributes that describe measurements

Information Analysis & Delivery

- Speed up retrieval using query optimizers and bitmap indices
- Adhoc query - Simple query and analysis functions
- Managed Query - Business layer between end users and database
- Multidimensional - OLAP - support complex analysis of dimensional data

Information Analysis & Delivery

- EIS/DSS
  - Packaged queries and reports
  - Preplanned analytical functions
  - Answer specific questions
- Alerts
  - Specific indicators

Managing the Data Warehouse

- Data - Size storage needs
  Security
  Backups
  Tracking
- Process- Monitoring update process like changes in source, quality of data
  Accurate and upto date

Tools

- Data Extraction - SAS
- Data Cleaning - Apertus, Trillium
- Data Storage - ORACLE, SYBASE
- Optimizers - Advanced Parallel Optimizer
  Bitmap Indices
  Star Index

Tools

- Development tools to create applications

  IBM Visualizer, ORACLE CDE
- Relational OLAP

  Informix Metacube

Architecture

- Rehosting Mainframe Applications
  - Moving to lower cost microprocessors
  - Tools - Micro Focus COBOL
  - Lowers Cost
  - No transparent Access to data

Architecture

- Mainframe as server
  - 2-tier approach
  - Front end client & back end server
  - Power Builder, VB - Front end tools
  - Minimal investment in extra hardware
  - Data inconsistency hidden
  - Fat Client
  - Cannot be used if number of end users increase

Architecture

- Enterprise Information Architecture

  3 tier

  Source data on host computer

  Database servers like ORACLE,

  Essbase(MDD)

  Front-end tools - DSS/EIS

RDBMS

- RDBMS provide rapid response to queries
  - Bitmap index
  - Index structures
- Functionality added to conventional RDBMS like data extraction and replication

MDD

- Decision support environment
- Supports iterative queries
- Extensions to SQL - for high performance data warehousing
- Performance degrades as size increases
- Inability to incrementally load
- Loading is slow
- No agreed upon model

MDD

- No standard access method like SQL
- Minor changes require complete reorganization

Data Access Tools

- Simple relational query tools - Esperent
- DSS/EIS - EXPRESS used by financial specialists

Database Design

- Simple
- Data must be clean
- Query processing must be fast
- Fast loading

Star Schema

- Consists of a group of tables that describe the dimensions of the business arranged logically around a huge central table that contains all the accumulated facts and figures of the business.
- The smaller, outer tables are points of the star, the larger table the center from which the points radiate.

Star Schema

- Fact Table

  -Sales, Orders, Budget, Shipment

  Real values (numeric)
- Dimension Table

  -Period, Market, Product

  Character data
- Summary/Aggregate data

Star Schema

- Data you can trust

  Referrential Integrity

- Query Speed

  Fact table - Primary key

  Dimension table - all columns

  Query optimizer which understands star schema

Star Schema

- Load Processing
  Must be done offline
  Issue if aggregate data is stored

Variations of Star Schema

- Outboard tables
- Fact table families
- Multistar fact table

OLAP & OLTP

- Front end tool for MDD
- Slice Report
- Pivot Report
- Alert-reporting
- Time-based
- Exception reporting

Wide OLAP

- Generating (synthesizing) information as well as using it, and storing this additional information by updating the data source
- Modeling capabilities, including a calculation engine for deriving results and creating aggregations, consolidations and complex calculations
- Forecasting, trend analysis, optimization, statistical analysis

Relational OLAP

- Has a powerful SQL-generator
- Generates SQL optimized for the target database
- Rapidly changing dimensions

MDD OLAP

- Row level calculations
- Financial functions, currency conversions,
  interest calculations

Metadata

- User Oriented

  Definition of attributes

- System oriented

  Record and field edit procedure names

Uses of Metadata

- Map source system data to data warehouse tables
- Generate data extract, transform, and load procedures for import jobs
- Help users discover what data are in the data warehouse
- Help users structure queries to access data they need

Describing the data warehouse

- I/P - O/P object
  File/Table
  Archive Period
- Relationship
- Data element - Name, Defn., Type
- Relationship Member - Role, Participation Constraint
- Field Assignment

Extract Jobs

- Wholesale replace
- Wholesale append
- Update replace
- Update append

Planning

- Interviews
- Data quality
- Data Access
- Timeliness and history
- Data sources
- Decide on Architecture

Development Process

- Project Initiation
- Develop Enterprise Info. Architecture
- Design Data Warehouse Database
- Transform data
- Manage Metadata
- Develop User-Interface
- Manage Production

Evolution

- Support the current DW baseline
- Enhance current baseline capabilities
- Define new business requirements
- Implement new baseline

Mistakes

- Starting with the wrong sponsorship chain
- Setting expectations that cannot be met
- Believing that DW design is the same as Transactional Database Design
- Believing the Performance, Capacity Promises
- Believing that Once the Data Warehouse Is Up and Running Problems are finished

THANK YOU