Data Mining:
Concepts and Techniques

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

Why Data Mining?

- The Explosive Growth of Data: from terabytes to petabytes
  - Data collection and data availability
    - Automated data collection tools, database systems, Web, computerized society
  - Major sources of abundant data
    - Business: Web, e-commerce, transactions, stocks, …
    - Science: Remote sensing, bioinformatics, scientific simulation, …
    - Society and everyone: news, digital cameras, YouTube
- We are drowning in data, but starving for knowledge!
- "Necessity is the mother of invention"—Data mining—Automated analysis of massive data sets

Introduction

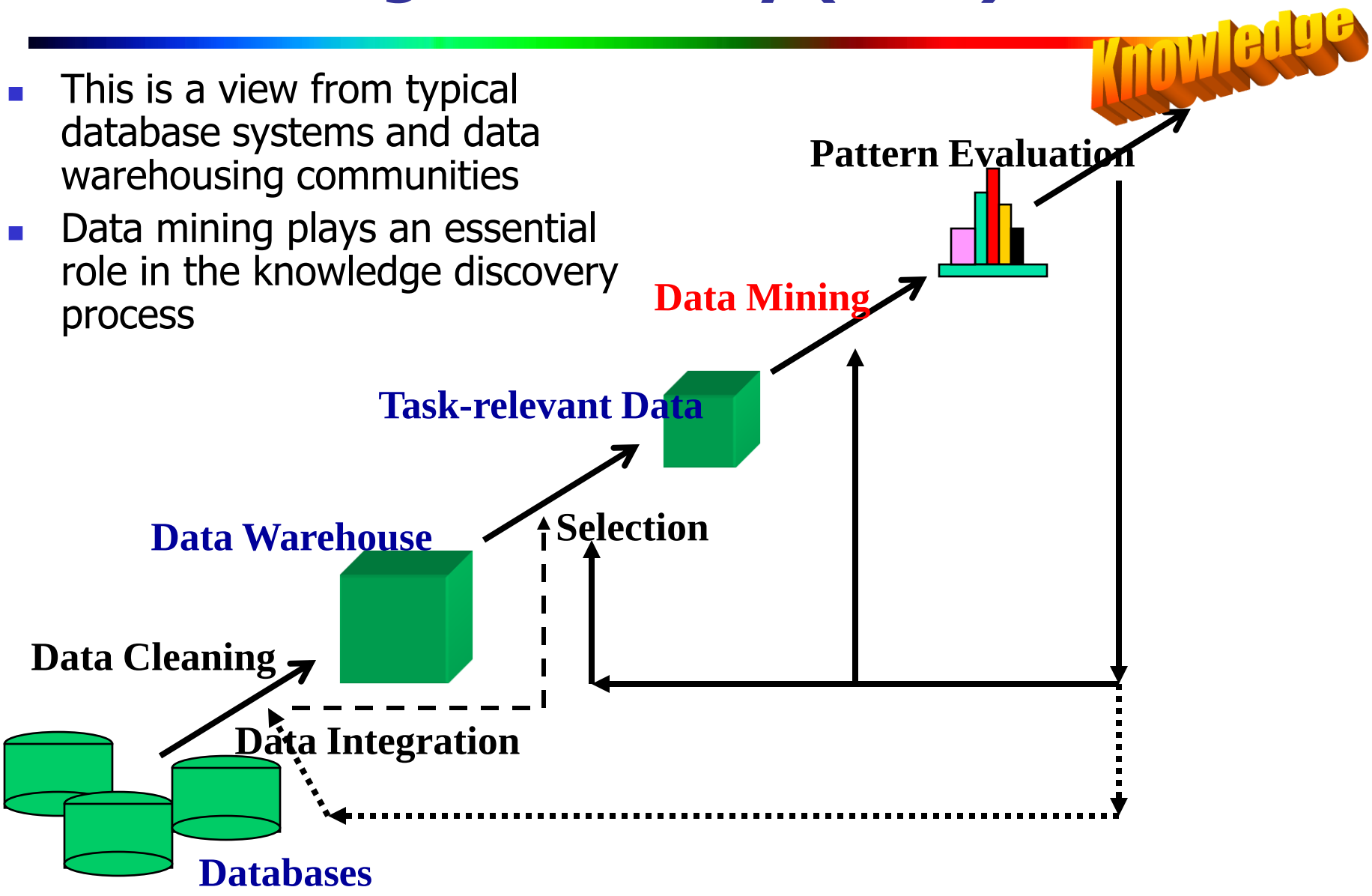
- Why Data Mining?
- What Is Data Mining? 
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

What Is Data Mining?

- Data mining (knowledge discovery from data)
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data

Knowledge Discovery (KDD) Process

- This is a view from typical database systems and data warehousing communities
- Data mining plays an essential role in the knowledge discovery process



Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

Multi-Dimensional View of Data Mining

- **Data to be mined**
  - Database data (extended-relational, object-oriented, heterogeneous, legacy), data warehouse, transactional data, stream, spatiotemporal, time-series, sequence, text and web, multi-media, graphs & social and information networks
- **Knowledge to be mined (or: Data mining functions)**
  - association, classification, clustering, trend/deviation, outlier analysis, etc.
  - Multiple/integrated functions and mining at multiple levels
- **Techniques utilized**
  - Data-intensive, data warehouse (OLAP), machine learning, statistics, pattern recognition, visualization, high-performance, etc.
- **Applications adapted**
  - Retail, telecommunication, banking, fraud analysis, bio-data mining, stock market analysis, text mining, Web mining, etc.

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

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
  - Data streams and sensor data
  - Time-series data, sequence data (incl. bio-sequences)
  - graphs, social networks and multi-linked data
  - Heterogeneous databases
  - Spatial data
  - Multimedia database
  - Text databases
  - The World-Wide Web

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

Data Mining Function: (1) Generalization

- Information integration and data warehouse construction
  - Data cleaning, transformation, integration, and multidimensional data model
- Data cube technology
  - Scalable methods for computing (i.e., materializing) multidimensional aggregates
  - OLAP (online analytical processing)
- Multidimensional concept description: Characterization and discrimination
  - Generalize, summarize, and contrast data characteristics, e.g., dry vs. wet region

Data Mining Function: (2) Association and Correlation Analysis

- Frequent patterns (or frequent itemsets)
  - What items are frequently purchased together in your Walmart?
- Association, correlation vs. causality
  - A typical association rule
    - Diaper → Beer [0.5%, 75%] (support, confidence)
  - Are strongly associated items also strongly correlated?
- How to mine such patterns and rules efficiently in large datasets?
- How to use such patterns for classification, clustering, and other applications?

Data Mining Function: (3) Classification

- Classification and label prediction
  - Construct models (functions) based on some training examples
  - Describe and distinguish classes or concepts for future prediction
    - E.g., classify countries based on (climate), or classify cars based on (gas mileage)
  - Predict some unknown class labels
- Typical methods
  - Decision trees, naïve Bayesian classification, support vector machines, neural networks, rule-based classification, pattern-based classification, logistic regression, …
- Typical applications:
  - Credit card fraud detection, direct marketing, classifying diseases, web-pages, …

Data Mining Function: (4) Cluster Analysis

- Unsupervised learning (i.e., Class label is unknown)
- Group data to form new categories (i.e., clusters), e.g., cluster houses to find distribution patterns
- Principle: Maximizing intra-class similarity & minimizing interclass similarity
- Many methods and applications

Data Mining Function: (5) Outlier Analysis

- Outlier analysis
  - Outlier: A data object that does not comply with the general behavior of the data
  - Noise or exception?
  - Methods: by product of clustering or regression analysis, …
  - Useful in fraud detection, rare events analysis

Time and Ordering: Sequential Pattern, Trend and Evolution Analysis

- Sequence, trend and evolution analysis
  - Trend, time-series, and deviation analysis: e.g., regression and value prediction
  - Sequential pattern mining
    - e.g., first buy digital camera, then buy large SD memory cards
  - Motifs and biological sequence analysis
    - Approximate and consecutive motifs
  - Similarity-based analysis
- Mining data streams
  - Ordered, time-varying, potentially infinite, data streams

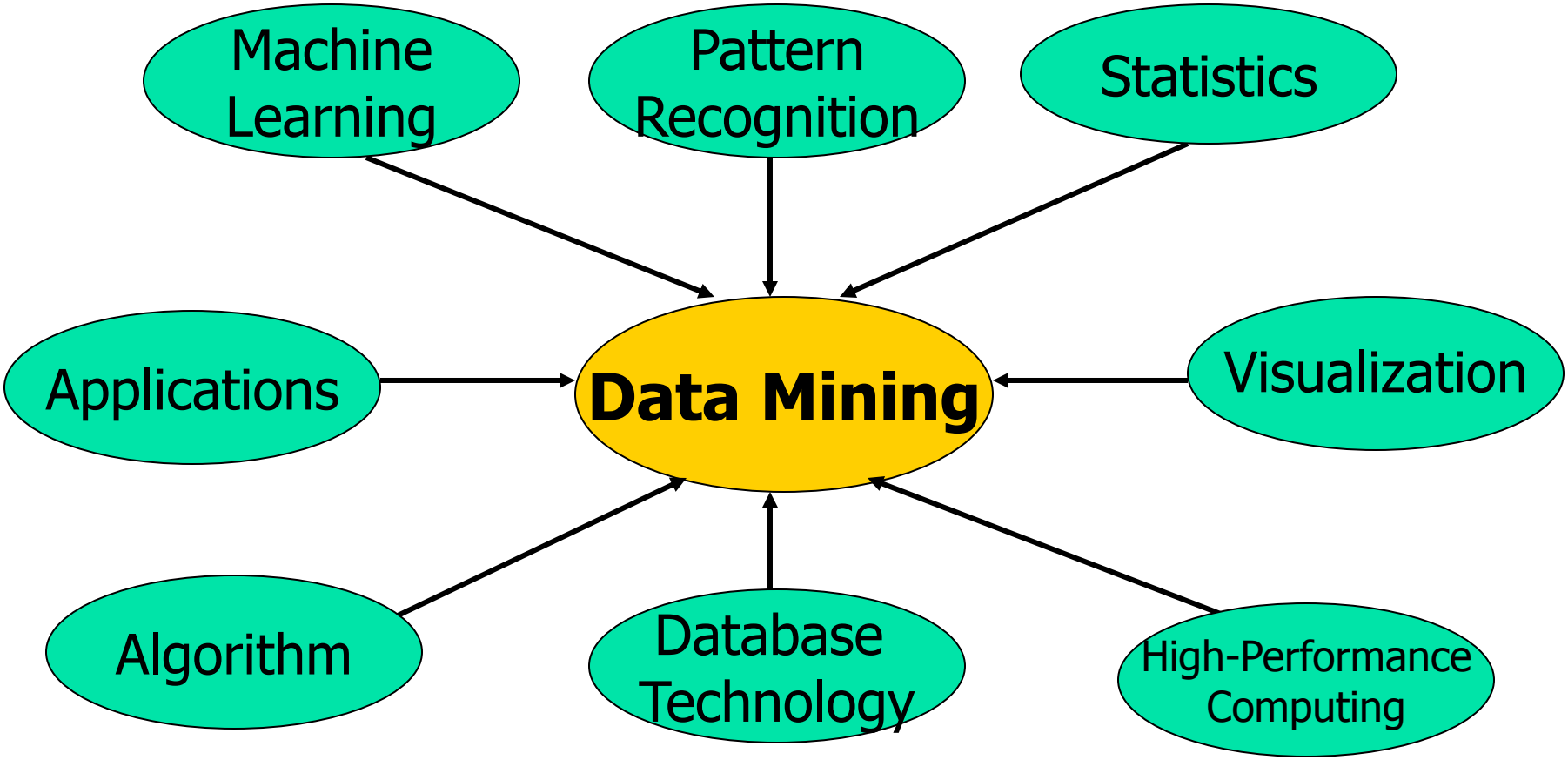
Structure and Network Analysis

- Graph mining
  - Finding frequent subgraphs (e.g., chemical compounds), trees (XML), substructures (web fragments)
- Information network analysis
  - Social networks: actors (objects, nodes) and relationships (edges)
    - e.g., author networks in CS, terrorist networks
  - Multiple heterogeneous networks
    - A person could be multiple information networks: friends, family, classmates, …
  - Links carry a lot of semantic information: Link mining
- Web mining
  - Web is a big information network: from PageRank to Google
  - Analysis of Web information networks
    - Web community discovery, opinion mining, usage mining, …

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used? 
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

Data Mining: Confluence of Multiple Disciplines



Why Confluence of Multiple Disciplines?

- Tremendous amount of data
  - Algorithms must be highly scalable to handle such as tera-bytes of data
- High-dimensionality of data
  - Micro-array may have tens of thousands of dimensions
- High complexity of data
  - Data streams and sensor data
  - Time-series data, temporal data, sequence data
  - Structure data, graphs, social networks and multi-linked data
  - Heterogeneous databases
  - Spatial, multimedia, text and Web data
- New and sophisticated applications

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted? 
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

Applications of Data Mining

- Web page analysis: from web page classification, clustering to PageRank algorithms
- Collaborative analysis & recommender systems
- Basket data analysis to targeted marketing
- Biological and medical data analysis: classification, cluster analysis (microarray data analysis), biological sequence analysis, biological network analysis

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining 
- A Brief History of Data Mining and Data Mining Society
- Summary

Major Issues in Data Mining (1)

- Mining Methodology
  - Mining various and new kinds of knowledge
  - Mining knowledge in multi-dimensional space
  - Data mining: An interdisciplinary effort
  - Boosting the power of discovery in a networked environment
  - Handling noise, uncertainty, and incompleteness of data
  - Pattern evaluation and pattern- or constraint-guided mining
- User Interaction
  - Interactive mining
  - Incorporation of background knowledge
  - Presentation and visualization of data mining results

Major Issues in Data Mining (2)

- Efficiency and Scalability
  - Efficiency and scalability of data mining algorithms
  - Parallel, distributed, stream, and incremental mining methods
- Diversity of data types
  - Handling complex types of data
  - Mining dynamic, networked, and global data repositories
- Data mining and society
  - Social impacts of data mining

Introduction

- Why Data Mining?
- What Is Data Mining?
- A Multi-Dimensional View of Data Mining
- What Kind of Data Can Be Mined?
- What Kinds of Patterns Can Be Mined?
- What Technology Are Used?
- What Kind of Applications Are Targeted?
- Major Issues in Data Mining
- A Brief History of Data Mining and Data Mining Society
- Summary

1.7 Data Mining Task Primitives

- How to construct a data mining query?
  - The primitives allow the user to interactively communicate with the data mining system during discovery to direct the mining process, or examine the findings

1.7 Data Mining Task Primitives

- The primitives specify:

  (1) The set of task-relevant data – which portion of the database to be used

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

1.7 Data Mining Task Primitives

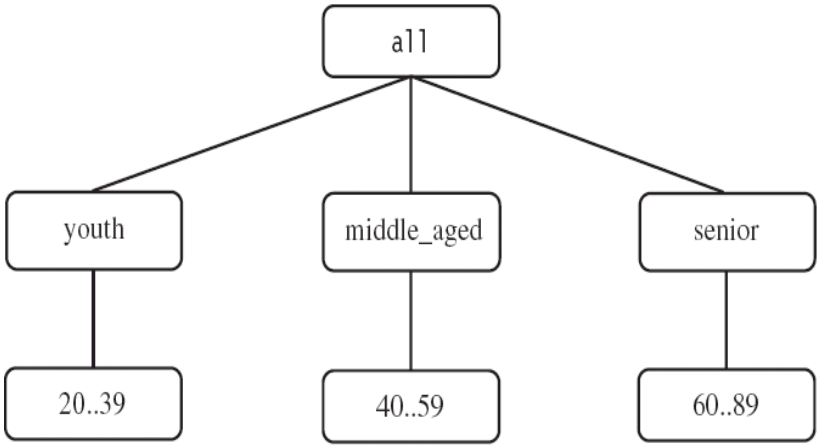
- The primitives specify:

  (2) The kind of knowledge to be mined – what DB functions to be performed

  - Characterization
  - Discrimination
  - Association
  - Classification/prediction
  - Clustering
  - Outlier analysis
  - Other data mining tasks

1.7 Data Mining Task Primitives

(3) The background knowledge to be used – what domain knowledge, concept hierarchies, etc.



(4) Interestingness measures and thresholds – support, confidence, etc.

(5) Visualization methods – what form to display the result, e.g. rules, tables, charts, graphs, …

1.7 Data Mining Task Primitives

- DMQL – Data Mining Query Language
  - Designed to incorporate these primitives
  - Allow user to interact with DM systems
  - Providing a standardized language like SQL

A Brief History of Data Mining Society

- 1989 IJCAI Workshop on Knowledge Discovery in Databases
  - Knowledge Discovery in Databases (G. Piatetsky-Shapiro and W. Frawley, 1991)
- 1991-1994 Workshops on Knowledge Discovery in Databases
  - Advances in Knowledge Discovery and Data Mining (U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, 1996)
- 1995-1998 International Conferences on Knowledge Discovery in Databases and Data Mining (KDD'95-98)
  - Journal of Data Mining and Knowledge Discovery (1997)
- ACM SIGKDD conferences since 1998 and SIGKDD Explorations
- More conferences on data mining
  - PAKDD (1997), PKDD (1997), SIAM-Data Mining (2001), (IEEE) ICDM (2001), etc.
- ACM Transactions on KDD starting in 2007

Conferences and Journals on Data Mining

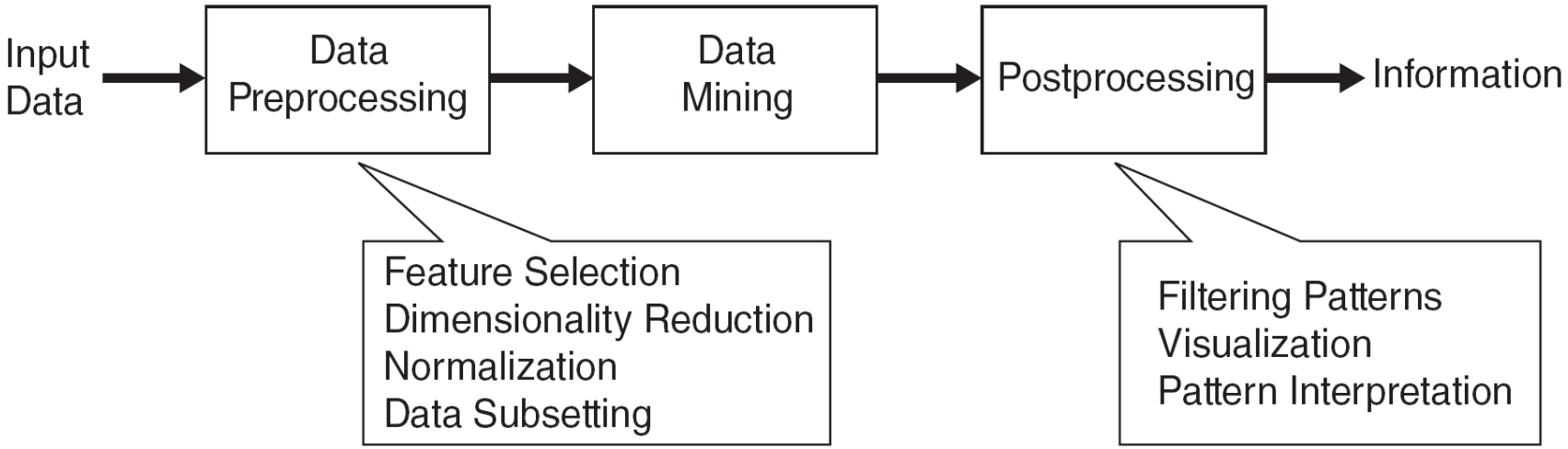
- KDD Conferences
  - ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases and Data Mining (KDD)
  - SIAM Data Mining Conf. (SDM)
  - (IEEE) Int. Conf. on Data Mining (ICDM)
  - European Conf. on Machine Learning and Principles and practices of Knowledge Discovery and Data Mining (ECML-PKDD)
  - Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)
  - Int. Conf. on Web Search and Data Mining (WSDM)
- Other related conferences
  - DB conferences: ACM SIGMOD, VLDB, ICDE, EDBT, ICDT, …
  - Web and IR conferences: WWW, SIGIR, WSDM
  - ML conferences: ICML, NIPS
  - PR conferences: CVPR,
- Journals
  - Data Mining and Knowledge Discovery (DAMI or DMKD)
  - IEEE Trans. On Knowledge and Data Eng. (TKDE)
  - KDD Explorations
  - ACM Trans. on KDD

Where to Find References? DBLP, CiteSeer, Google

- Data mining and KDD (SIGKDD: CDROM)
  - Conferences: ACM-SIGKDD, IEEE-ICDM, SIAM-DM, PKDD, PAKDD, etc.
  - Journal: Data Mining and Knowledge Discovery, KDD Explorations, ACM TKDD
- Database systems (SIGMOD: ACM SIGMOD Anthology—CD ROM)
  - Conferences: ACM-SIGMOD, ACM-PODS, VLDB, IEEE-ICDE, EDBT, ICDT, DASFAA
  - Journals: IEEE-TKDE, ACM-TODS/TOIS, JIIS, J. ACM, VLDB J., Info. Sys., etc.
- AI & Machine Learning
  - Conferences: Machine learning (ML), AAAI, IJCAI, COLT (Learning Theory), CVPR, NIPS, etc.
  - Journals: Machine Learning, Artificial Intelligence, Knowledge and Information Systems, IEEE-PAMI, etc.
- Web and IR
  - Conferences: SIGIR, WWW, CIKM, etc.
  - Journals: WWW: Internet and Web Information Systems,
- Statistics
  - Conferences: Joint Stat. Meeting, etc.
  - Journals: Annals of statistics, etc.
- Visualization
  - Conference proceedings: CHI, ACM-SIGGraph, etc.
  - Journals: IEEE Trans. visualization and computer graphics, etc.

What is Data Mining?

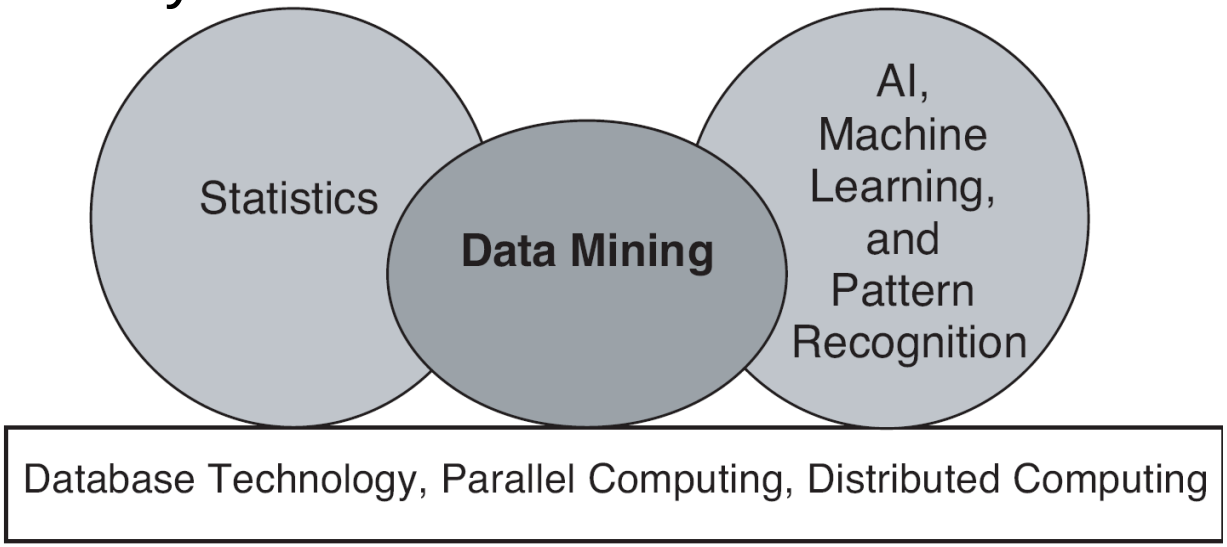
- Many Definitions
  - Non-trivial extraction of implicit, previously unknown and potentially useful information from data
  - Exploration & analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns



Origins of Data Mining

- Draws ideas from machine learning/AI, pattern recognition, statistics, and database systems

- Traditional techniques may be unsuitable due to data that is
  - Large-scale
  - High dimensional
  - Heterogeneous
  - Complex
  - Distributed



- A key component of the emerging field of data science and data-driven discovery

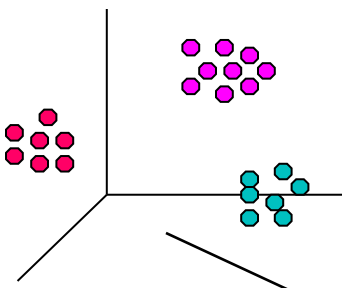
Data Mining Tasks

- Prediction Methods
  - Use some variables to predict unknown or future values of other variables.

- Description Methods
  - Find human-interpretable patterns that describe the data.

From [Fayyad, et.al.] Advances in Knowledge Discovery and Data Mining, 1996

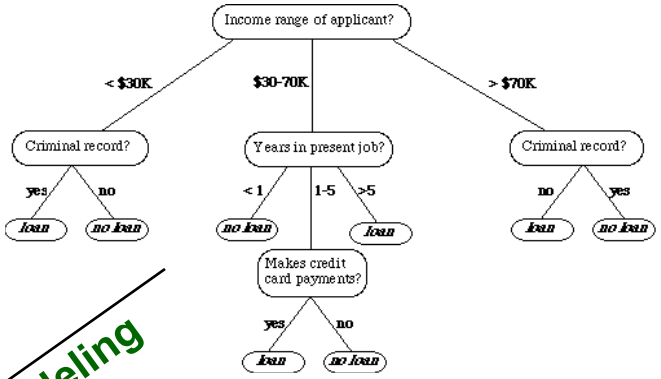
Data Mining Tasks ...



Clustering

Data

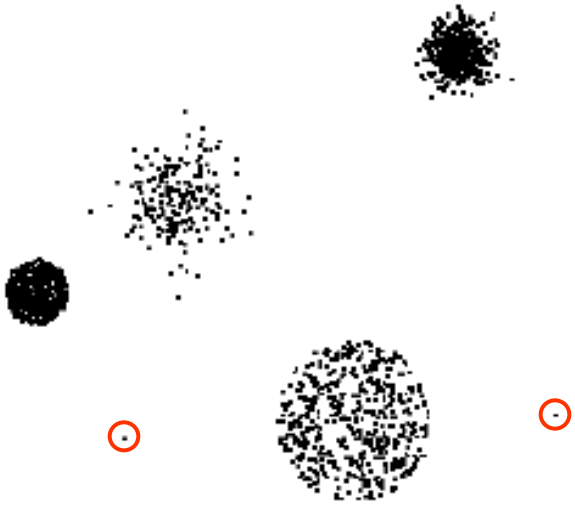
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes
11	No	Married	60K	No
12	Yes	Divorced	220K	No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



Predictive Modeling

Association Rules

Anomaly Detection



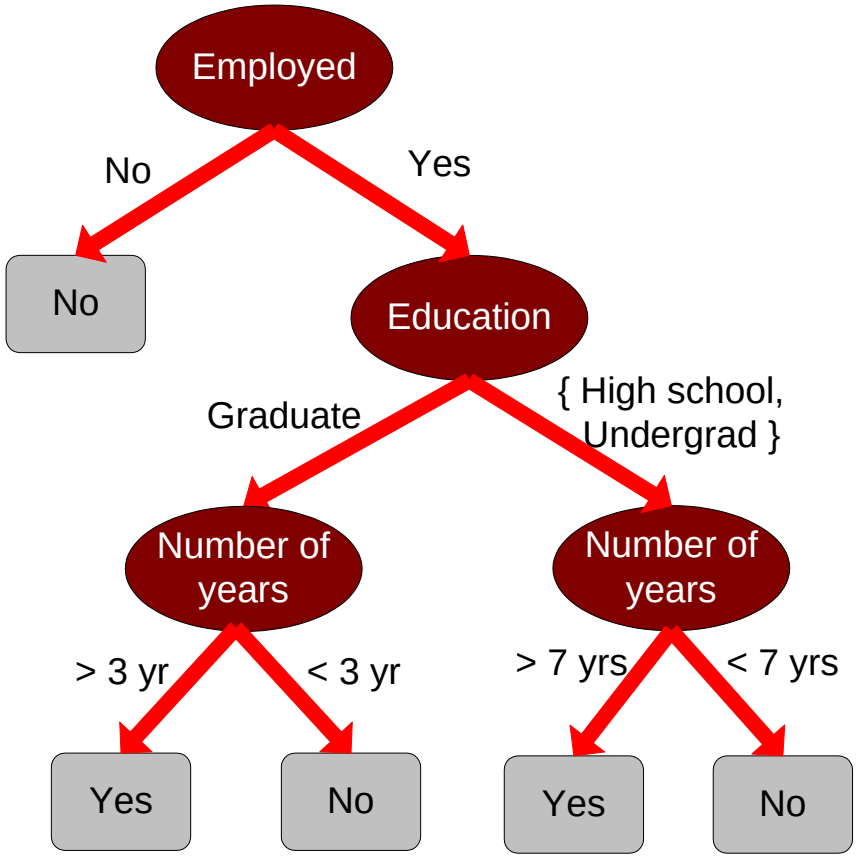
Predictive Modeling: Classification

- Find a model for class attribute as a function of the values of other attributes

Class

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Graduate	5	Yes
2	Yes	High School	2	No
3	No	Undergrad	1	No
4	Yes	High School	10	Yes
…	…	…	…	…

Model for predicting credit worthiness

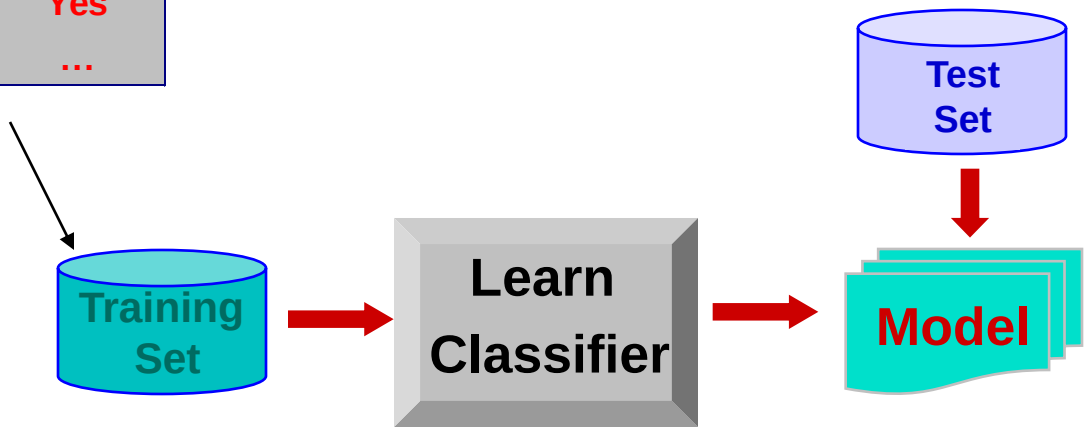


Classification Example

categorical categorical quantitative class

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Graduate	5	Yes
2	Yes	High School	2	No
3	No	Undergrad	1	No
4	Yes	High School	10	Yes
…	…	…	…	…

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Undergrad	7	?
2	No	Graduate	3	?
3	Yes	High School	2	?
…	…	…	…	…



Examples of Classification Task

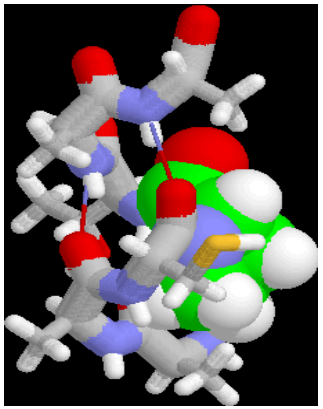
- Classifying credit card transactions as legitimate or fraudulent

- Classifying land covers (water bodies, urban areas, forests, etc.) using satellite data

- Categorizing news stories as finance, weather, entertainment, sports, etc

- Predicting tumor cells as benign or malignant

- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil

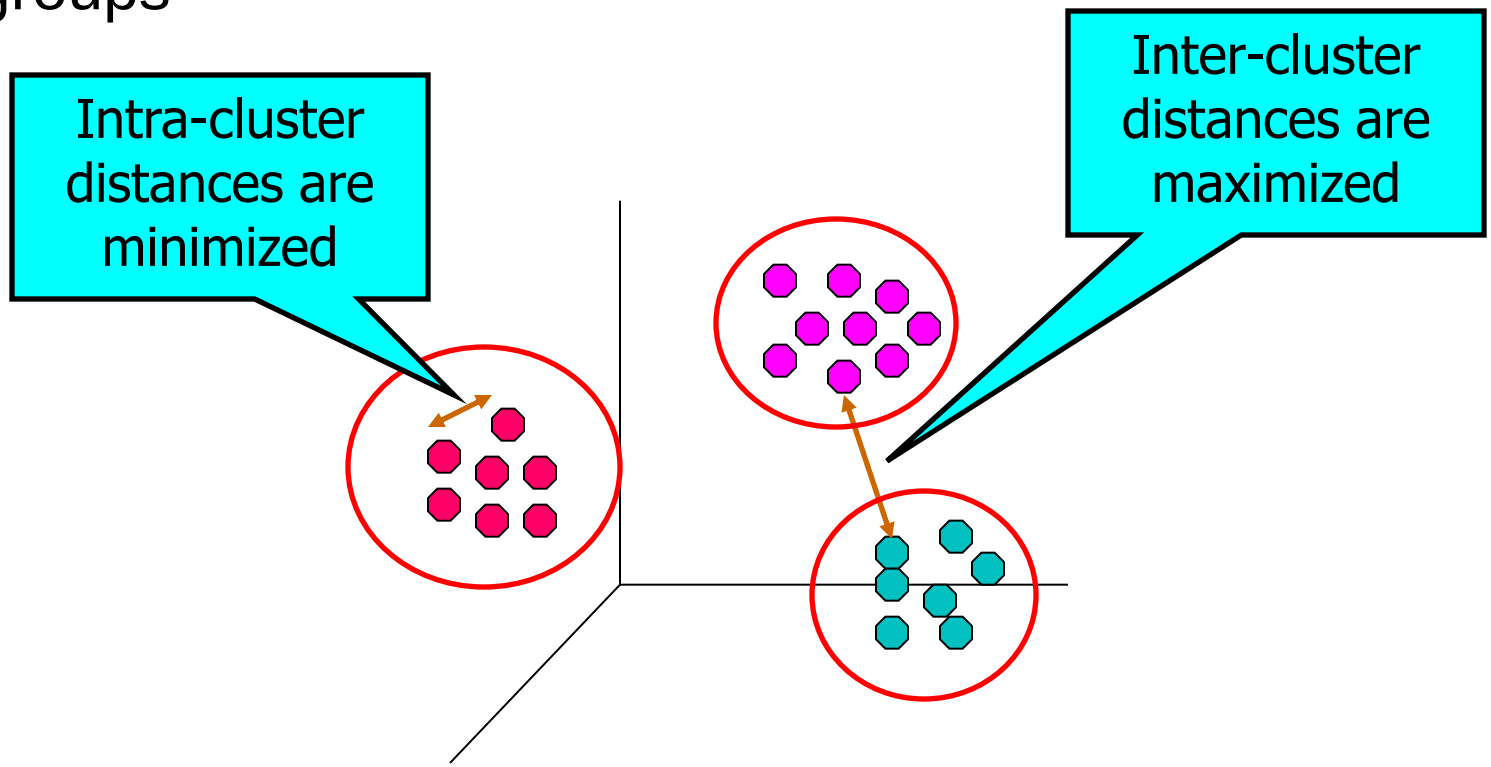


Classification: Application

- Fraud Detection
  - **Goal:** Predict fraudulent cases in credit card transactions.
  - **Approach:**
    - Use credit card transactions and the information on its account-holder as attributes.
      - When does a customer buy, what does he buy, how often he pays on time, etc
    - Label past transactions as fraud or fair transactions. This forms the class attribute.
    - Learn a model for the class of the transactions.
    - Use this model to detect fraud by observing credit card transactions on an account.

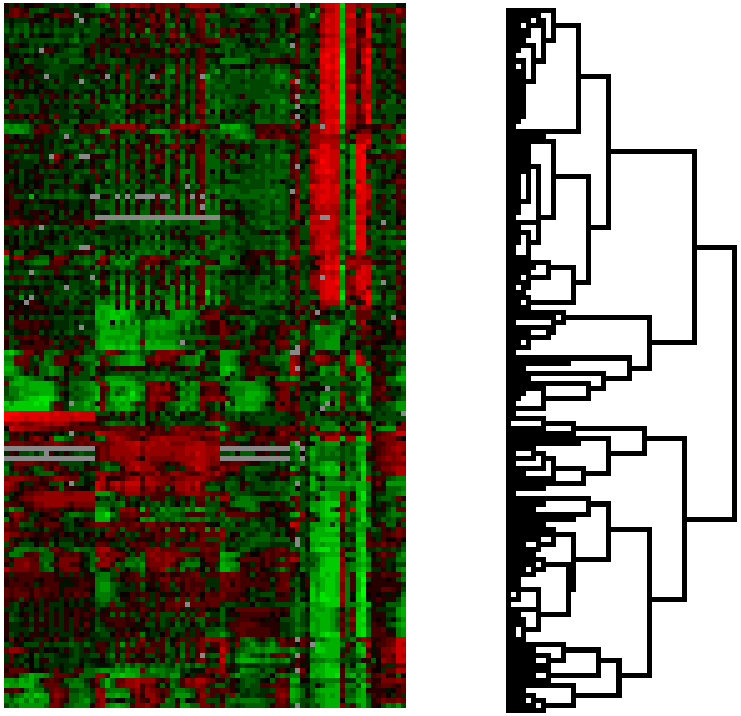
Clustering

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

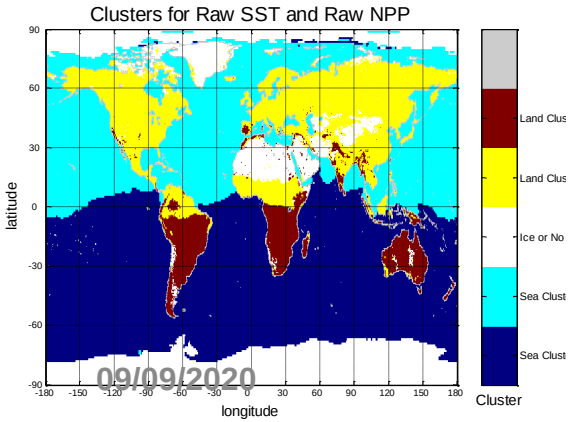


Applications of Cluster Analysis

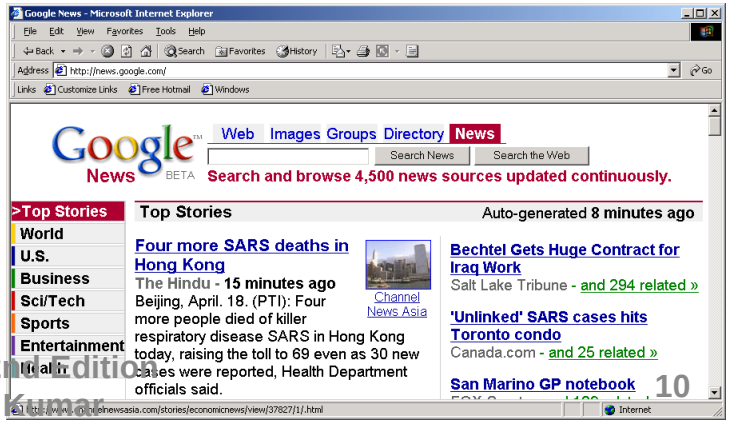
- **Understanding**
  - Custom profiling for targeted marketing
  - Group related documents for browsing
  - Group genes and proteins that have similar functionality
  - Group stocks with similar price fluctuations
- **Summarization**
  - Reduce the size of large data sets



Courtesy: Michael Eisen



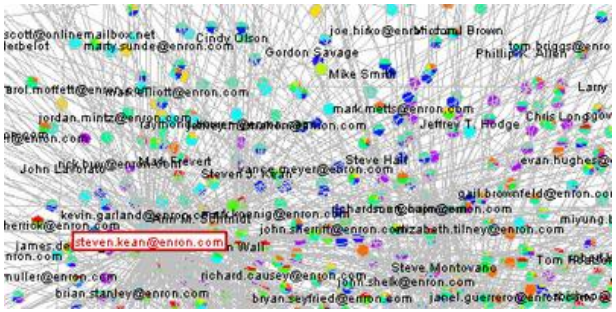
Use of K-means to partition Sea Surface Temperature (SST) and Net Primary Production (NPP) into clusters that reflect the Northern and Southern Hemispheres.



Clustering: Application

- Document Clustering:
  - **Goal:** To find groups of documents that are similar to each other based on the important terms appearing in them.
  - **Approach:** To identify frequently occurring terms in each document. Form a similarity measure based on the frequencies of different terms. Use it to cluster.

Enron email dataset



Association Rule Discovery: Definition

- Given a set of records each of which contain some number of items from a given collection
  - Produce dependency rules which will predict occurrence of an item based on occurrences of other items.

TID	*Items*
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Rules Discovered:

{Milk} --> {Coke}

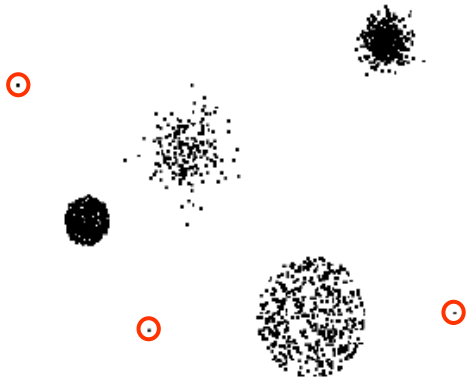
{Diaper, Milk} --> {Beer}

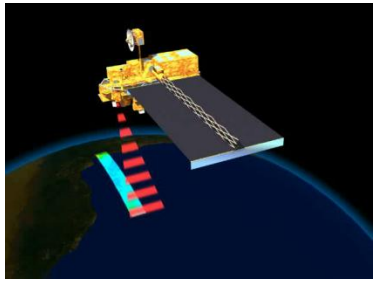
Association Analysis: Applications

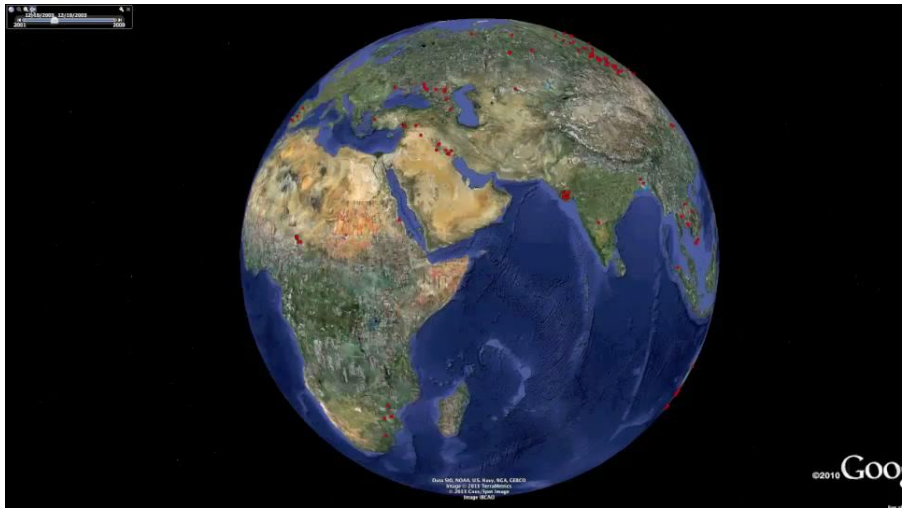
- Market-basket analysis
  - Rules are used for sales promotion, shelf management, and inventory management
- Telecommunication alarm diagnosis
  - Rules are used to find combination of alarms that occur together frequently in the same time period
- Medical Informatics
  - Rules are used to find combination of patient symptoms and test results associated with certain diseases

Deviation/Anomaly/Change Detection

- Detect significant deviations from normal behavior
- Applications:
  - Credit Card Fraud Detection
  - Network Intrusion Detection
  - Identify anomalous behavior from sensor networks for monitoring and surveillance.







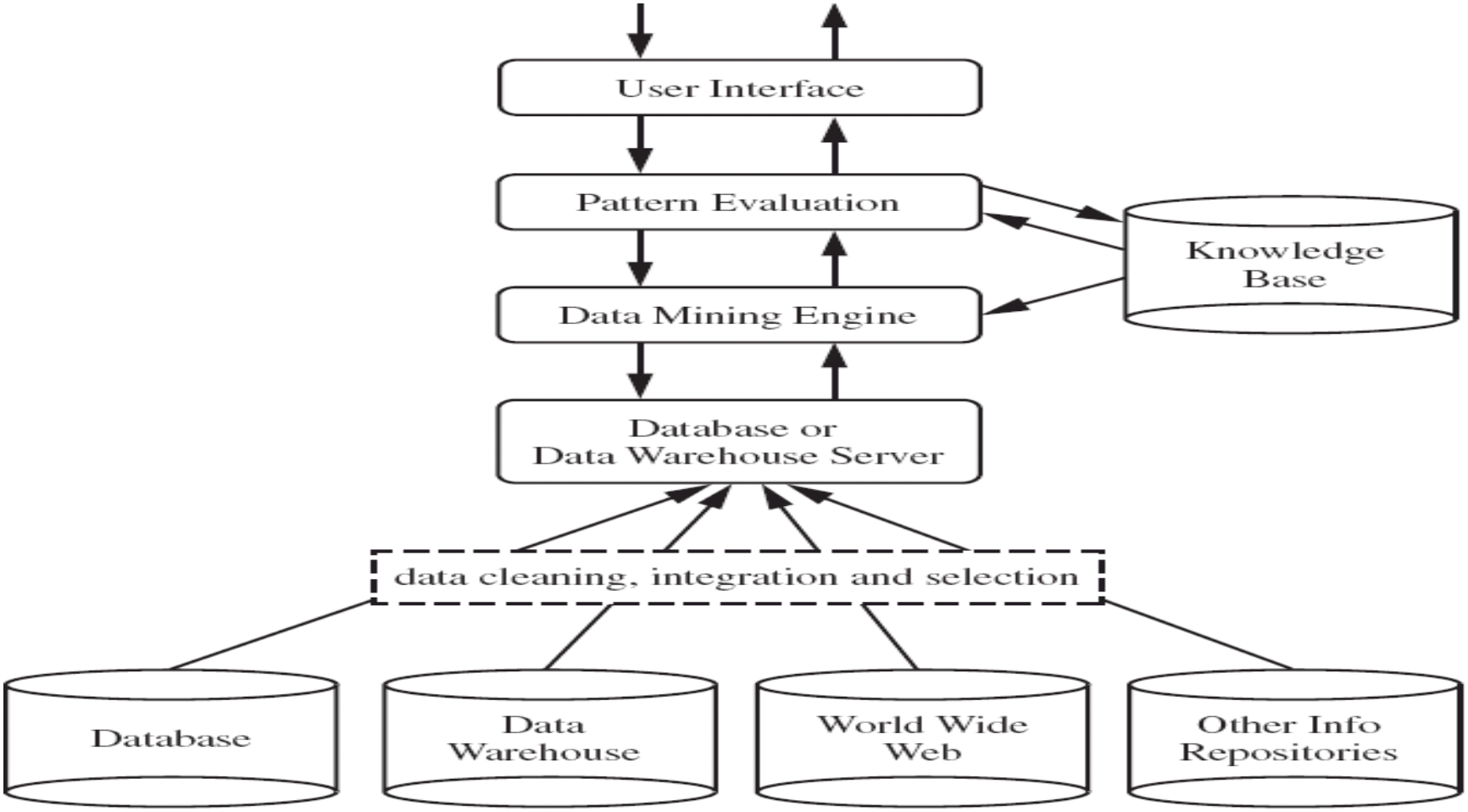
Motivating Challenges

- Scalability
- High Dimensionality
- Heterogeneous and Complex Data
- Data Ownership and Distribution
- Non-traditional Analysis

A typical DM System Architecture

- Database, data warehouse, WWW or other information repository (store data)
- Database or data warehouse server (fetch and combine data)
- Knowledge base (turn data into meaningful groups according to domain knowledge)
- Data mining engine (perform mining tasks)
- Pattern evaluation module (find interesting patterns)
- User interface (interact with the user)

A typical DM System Architecture



DEPARTMENT OF INFORMATION TECHNOLOGY, NITK, SURATHKAL

Course Code:	**IT-414**	**Course Name:**	**Data Warehousing and Data Mining**
Core/Elective/MLC:	Elective	**L-T-P:**	(3-0-2) 4
Pre-requisites:	Database System	**Contact Hours:**	8 AM to 5 PM
Type of course: (Lecture/Tutorial/ Seminar/Project)	Lecture/Project	**Course Assessment Methods: (Both Continuous and Semester-End Assessment)**	Mid Sem Exam=20% Lab Assignments=15% End Sem Exam=35% Course Project=30%

Course Description: The Course Covers the following main aspects

- What is data warehouse and data Mining, importance and challenges of Data Mining
- Data Warehouse Architecture and schemas
- Data Mining Techniques and its real world application

Course Objectives: The aim of the course is

- To understand the data warehouse architecture, data warehouse schemas, Multidimensional model
- To understand and apply different data preprocessing techniques
- To learn different feature selection methods
- To study different Association Rule mining methods to identify important association between frequent itemsets and to apply frequent itemset mining on the data
- To understand and apply different classification and clustering methods

Course Outcomes: After the completion of this course, the student will be able to:

CO1	Understand the differences of OLTP and OLAP, understand the data warehousing techniques, data warehouse architecture and process of constructing data warehouses with different models
CO2	Learn data mining techniques such as association rule mining, various classification and clustering techniques and apply for knowledge representation
CO3	Understand the different data preprocessing techniques along with feature selection methods
CO4	To apply DM techniques to real world dataset and Analyze results

Detailed Course Plan:

Weeks	**Topics**
Week 1 & 2	Introduction to data mining: Motivation and significance of data mining, data mining functionalities, interestingness measures, classification of data mining system, major issues in data mining, Data Mining task primitives, types of data for data mining
Week 3 & 4:	Association Analysis: Basic concepts: Frequent itemsets, Association rules, Market basket analysis example. Frequent ItemSet Mining methods: Apriori algorithm, Rule Generation, FP-growth Algorithm; Constraint based Association analysis.
Week 5 & 6	Compact Representation of Frequent ItemSets, Collosal itemset Mining methods, multilevel and multidimensional association rule mining
Week 7 & 8:	Data preprocessing: Why preprocess the data?, Data cleaning: Missing value, Noisy data; Data integration and transformation: Correlation analysis, Min Max, Z-score and decimal scaling normalization, ; Attribute subset selection. Filter based and Wrapper based feature selection methods, Chi square method, Heurastic methods for feature selection
Week 9 & 10:	PCA, OLTP/OLAP differences; Data Warehouse: What is Data Warehouse and data warehouse schemas, Data Cubes and multidimensional data model, data warehouse architecture
Week 11 & 12:	Classification and clustering methods: Density based clusters and comparison with partition and hierarchical methods, Association based classification, ensemble based classifiers, Boosted Decision tree classifier,
Week 13	Data mining on complex data and applications, trends , Research paper discussion.

Reference Books:

1 Han, J. and Kamber, M., "Data Mining - Concepts and Techniques", 3rd Ed., Morgan Kaufmann Series, (Elsevier), 2008.

2 Alex Berson , S. J. Smith, "Data Warehousing, Data Mining & OLAP" , McGraw Hill

3 Tan, P.N., Steinbach, M. and Kumar, V., "Introduction to Data Mining", Addison Wesley – Pearson, 2006

4 Pujari, A. K., "Data Mining Techniques", 4th Ed., Sangam Books.

5 Oded Maimon, Lior Rokach, The Data Mining and Knowledge Discovery Handbook, Springer, 2005.

6 S. Weiss and N. Indurkhya, Predictive Data-Mining: A Practical Guide, Morgan Kaufmann, 1998

7 S. Weiss, N. Indurkhya, T. Zhang and F. Damerau, Text Mining: Predictive Methods for Analyzing Unstructured Information, Springer, 2004.

Course Instructors:

Dr. Nagamma Patil

Outline

- Basics
- Market Basket Analysis: A Motivating Example
- Preliminaries
- Frequent Itemset Mining
- Apriori Algorithm
- Frequent Pattern growth (FP-growth) Algorithm
- Frequent Closed Itemset Mining
- Frequent Closed Itemset Mining from High Dimensional Data

Basics

- Frequent patterns are patterns (such as itemsets, subsequences, or substructures) that appear in a data set frequently.
- A set of items, such as milk and bread, that appear frequently together in a transaction data set is a frequent itemset.
- A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs frequently in a shopping history database, is a frequent sequential pattern.
- A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices. If a substructure occurs frequently, it is called a frequent structured pattern.

Market Basket Analysis: A Motivating Example

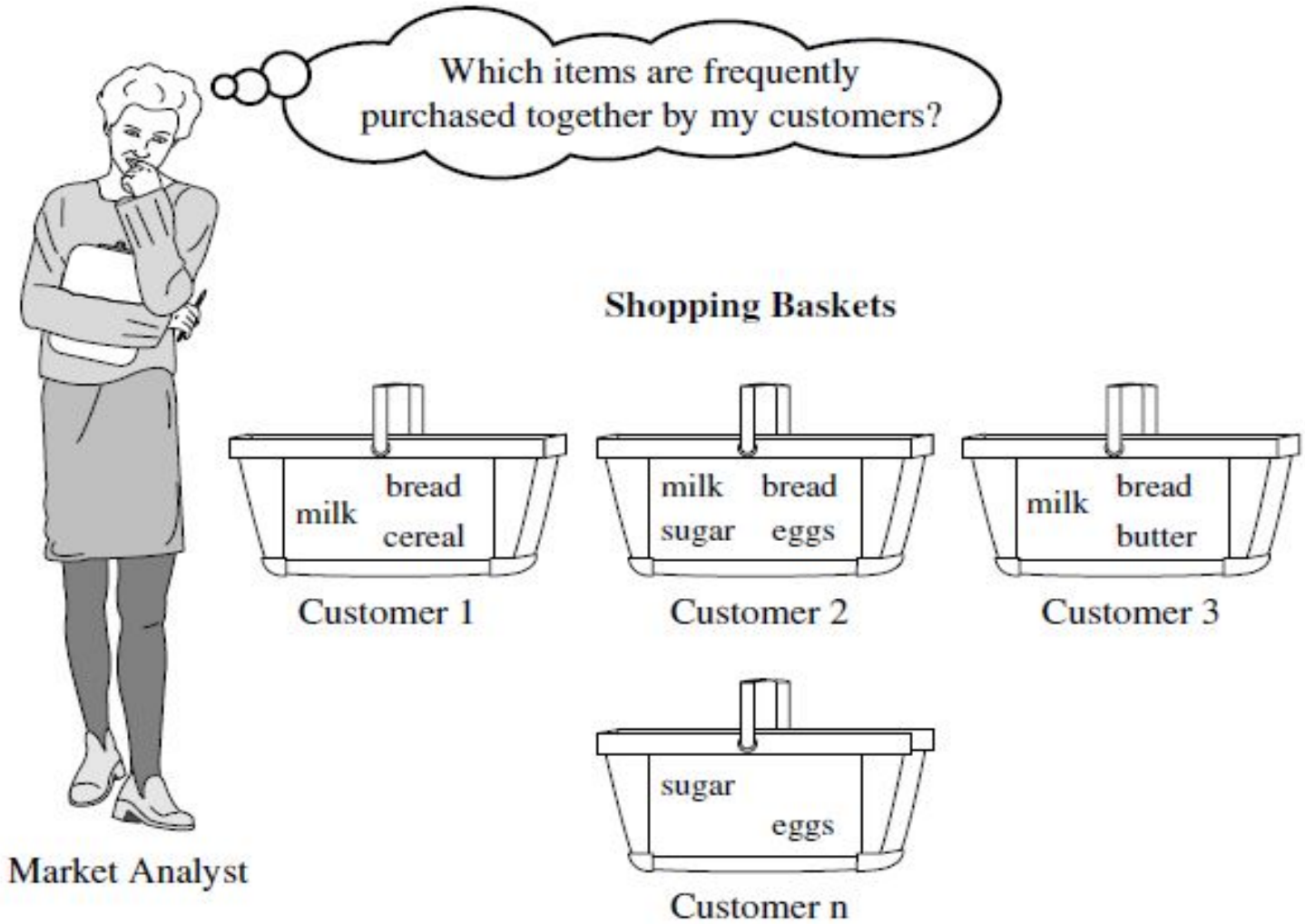


Figure 1. Apriori

Market Basket Analysis

- Frequent itemset mining leads to the discovery of associations and correlations among items in large transactional or relational data sets.
- A typical example of frequent itemset mining is market basket analysis.
- Data: collection of transactions of customers.
- Goal: find sets of products frequently occurring together.
- The discovery of associations helps in many business decision making processes, such as catalog design and customer shopping behavior analysis.

Applications

- Market basket analysis.
- Catalog design.
- Customer shopping behavior analysis.
- Web log analysis.
- DNA sequence analysis.
- Sale campaign analysis.
- Software bug detection.
- Chemical Compound Prediction.
- Text analysis.

Preliminaries

Let the Dataset *D* consist of *m* number of transactions (rows) and *n* of attributes or products (features)

- R = $\{r_1, r_2, \ldots r_m\}$
- F = $\{f_1, f_2, \ldots, f_n\}$
- Each row r_i has unique row identifier, *rid* and consist of set of products (features).
- A non-empty subset of features $X \subseteq$ F is defined as an itemset.
- Let $r(f_j)$ signify the rows in which j^{th} feature of the dataset is present.
- A non-empty subset of rids $Y \subseteq$ R is defined as rowset.
- Let $f(r_i)$ signify the features present in the i^{th} row of the dataset.

Preliminaries

Example 1

Table 1 shows an example of a Dataset D consisting of 8 rows, where each row is described with unique row identifier (rid), R= {1, 2, 3, 4, 5, 6, 7, 8} and 11 features, F= $\{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}\}$.

Table 1

Dataset D

row id (rid)	features
1	$f_1, f_2, f_4, f_6, f_{10}$
2	f_1, f_2, f_4, f_7, f_8
3	f_2, f_4, f_7, f_8
4	$f_1, f_2, f_6, f_8, f_9, f_{10}$
5	$f_1, f_3, f_4, f_7, f_8, f_{10}$
6	f_2, f_4, f_9
7	f_5, f_7
8	f_5, f_{11}

Preliminaries

Definition 1 (Support)

The number of rows in which an itemset X occurs is called the support of an itemset, denoted by *sup(X)*.

Example 2

In Table 1, the support of an itemset $X = \{f_2, f_4, f_7, f_8\}$, *sup(X)* is 2.

Definition 2 (Support Set)

The rows in which an itemset X occurs is called support set of an itemset, denoted by *supset(X)*.

Example 3

In Table 1, the support set of an itemset $X = \{f_2, f_4, f_7, f_8\}$, *supset(X)* is *23*.

Preliminaries

Definition 3 (Cardinality)

The number of items in an itemset X is called as the cardinality of an itemset, denoted by *card(X)*.

Example 4

In Table 1, the cardinality of an itemset $X = \{f_2, f_4, f_7, f_8\}$, *card(X)* is 4.

Definition 4 (Frequent Itemset)

An itemset X is called frequent itemset if and only if $sup(X) \geq minsup$, where *minsup* is user specified least support threshold.

Example 5

In Table 1, the itemset $X = \{f_2, f_8\}$ is frequent itemset with minimum support threshold set to 2, $sup(X) \geq 2$.

Preliminaries

Definition 5 (Association Rule)

Let A and B be the set of items. An association rule is an implication of the form $A \Rightarrow B$, where $A \subset \mathrm{F}$, $B \subset \mathrm{F}$ and $A \cap B = \emptyset$. The association rule $A \Rightarrow B$ holds in the dataset with **support** s and has **confidence** c.

Support s, is the percentage of transactions in D that contain A $\cup$ B (i.e., the union of sets A and B, or say, both A and B).

Confidence c, is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability, P(B|A).

$$support(A \Rightarrow B) = P(A \cup B) \tag{1}$$

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support(A \cup B)}{support(A)} \tag{2}$$

Frequent Itemset Mining

Table 2

Dataset D

TID	Items Bought
1	Beer, Nuts, Chips
2	Beer, Coffee, Chips
3	Beer, Chips, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Chips, Eggs, Milk

- Problem: To Mine the Frequent Itemsets with minimum support threshold (*minsup*) set to 50% and minimum confidence threshold (*minconf*) set to 50%.
- Frequent Itemsets are: Beer:3, Nuts:3, Chips:4, Eggs:3, {Beer, Chips}:3.
- Example of association rules
  Beer $\rightarrow$ Chips (60%, 100%).
  Chips $\rightarrow$ Beer (60%, 75%).

Frequent Itemset Mining

- Frequent Itemset Mining Algorithms
  - Apriori Algorithm
  - Frequent Pattern growth (FP-growth) algorithm
- Frequent Closed Itemset Mining Algorithms
- Frequent Maximal Itemset Mining Algorithms
- Frequent Colossal Itemset Mining Algorithms
- Frequent Colossal Closed Itemset Mining Algorithms

Apriori Algorithm

- Apriori is an algorithm proposed by R. Agrawal and R. Srikant in 1994 for mining frequent itemsets from transactional datasets for generating association rules.
- The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset properties.
- Apriori employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1)-itemsets.
- First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support.

Apriori Algorithm

- The resulting set is denoted L_1. Next, L_1 is used to find L_2, the set of frequent 2-itemsets, which is used to find L_3, and so on, until no more frequent k-itemsets can be found.
- The finding of each L_k requires one full scan of the database.
- To improve the efficiency of the level-wise generation of frequent itemsets, an important property called the Apriori property.
- Apriori property: All nonempty subsets of a frequent itemset must also be frequent.
- The property belongs to a special category of properties called antimonotone in the sense that if a set cannot pass a test, all of its supersets will fail the same test as well.

Apriori Algorithm

- Apriori Algorithm has two steps
  - The Join step
  - The Prune step
- **The Join step:**
  - To find L_k, a set of candidate k-itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k.
  - Apriori assumes that items within a transaction or itemset are sorted in lexicographic order.
  - The join, $L_{k-1} \bowtie L_{k-1}$, is performed, where members of L_{k-1} are joinable if their first (k-2) items are in common.

Apriori Algorithm

- **The Prune step:**
  - C_k is a superset of L_k, that is, its members may or may not be frequent, but all of the frequent k-itemsets are included in C_k.
  - A scan of the database to determine the count of each candidate in C_k would result in the determination of L_k.
  - Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset.
  - If any (k-1)-subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k.

Apriori Algorithm

Table 3
Dataset D

TID	List of items
1	I1, I2, I5
2	I2, I4
3	I2, I3
4	I1, I2, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I1, I2, I3, I5
9	I1, I2, I3

Apriori Algorithm

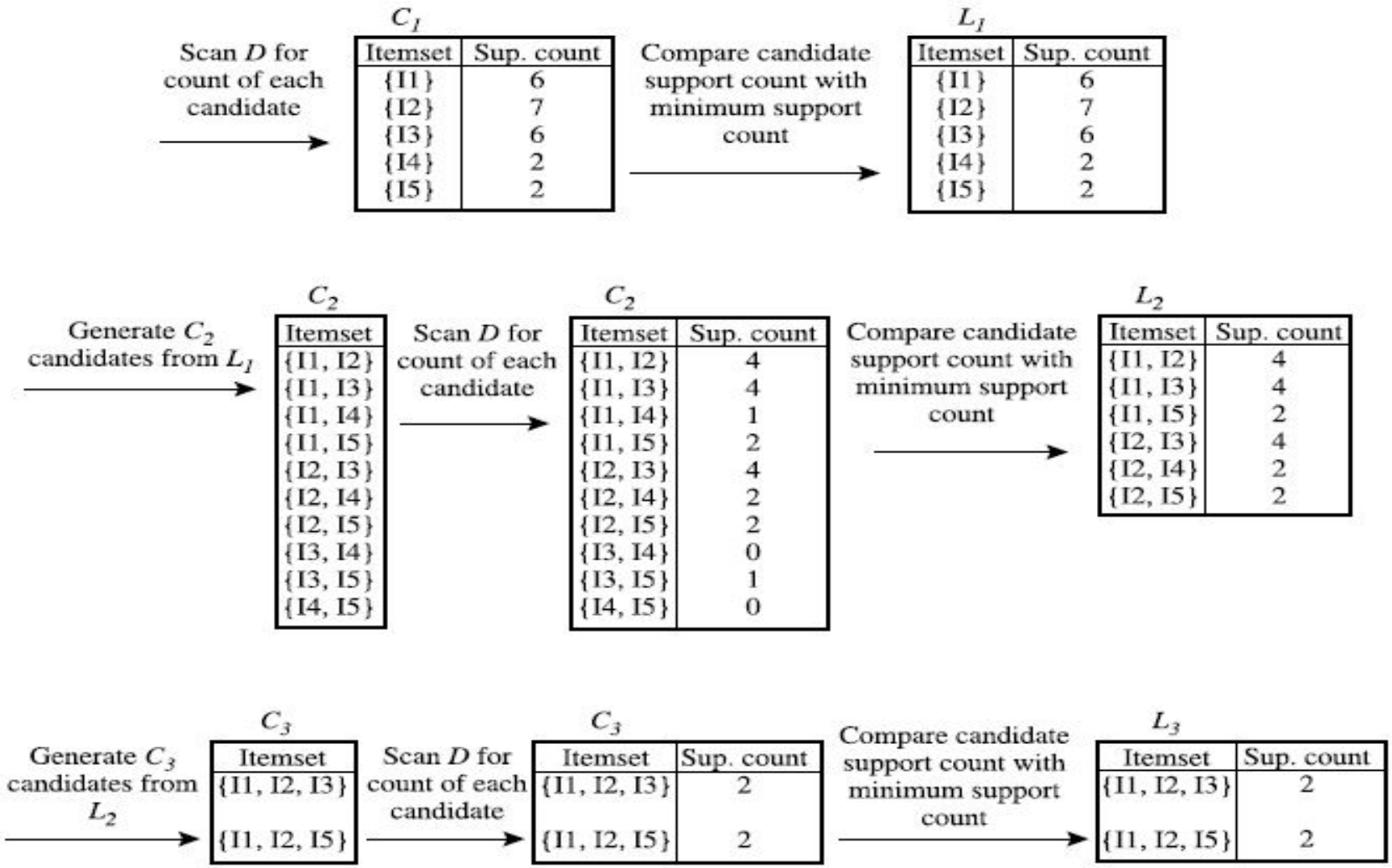


Figure 2. Steps Apriori Algorithm

Apriori Algorithm

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

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

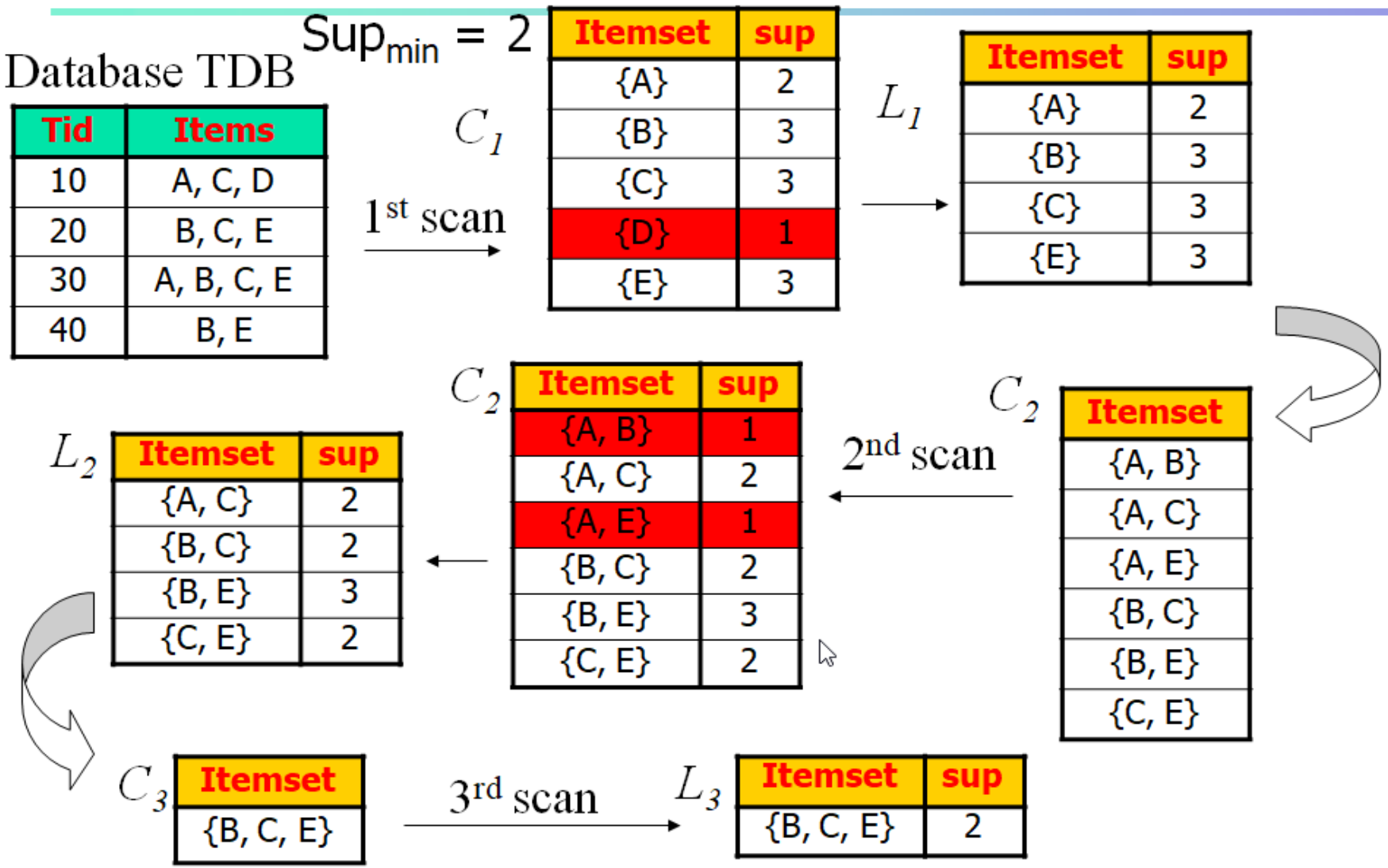
- The 2-item subsets of $\{I1, I2, I3\}$ are $\{I1, I2\}$, $\{I1, I3\}$, and $\{I2, I3\}$. All 2-item subsets of $\{I1, I2, I3\}$ are members of L_2. Therefore, keep $\{I1, I2, I3\}$ in C_3.
- The 2-item subsets of $\{I1, I2, I5\}$ are $\{I1, I2\}$, $\{I1, I5\}$, and $\{I2, I5\}$. All 2-item subsets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

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

Apriori Algorithm

- Generating association rules. The frequent itemset considered is {I1, I2, I5}
- The nonempty subsets of frequent itemset are {I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}.
- The resulting association rules are as shown below, each listed with its confidence:
  - $I1 \wedge I2 \Rightarrow I5$, confidence $= 2/4 = 50\%$
  - $I1 \wedge I5 \Rightarrow I2$, confidence $= 2/2 = 100\%$
  - $I2 \wedge I5 \Rightarrow I1$, confidence $= 2/2 = 100\%$
  - $I1 \Rightarrow I2 \wedge I5$, confidence $= 2/6 = 33\%$
  - $I2 \Rightarrow I1 \wedge I5$, confidence $= 2/7 = 29\%$
  - $I5 \Rightarrow I1 \wedge I2$, confidence $= 2/2 = 50\%$

Apriori Algorithm



FP-Tree/FP-Growth Algorithm

- Use a compressed representation of the database using an FP-tree
- Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets.

Building the FP-Tree

1. Scan data to determine the support count of each item.

   Infrequent items are discarded, while the frequent items are sorted in decreasing support counts.

2. Make a second pass over the data to construct the FPtree.

   As the transactions are read, before being processed, their items are sorted according to the above order.

FP-tree Example: step 1

Step 1: Scan DB for the first time to generate L

(minimum support=3)

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

L

Item	*frequency*
f	*4*
c	*4*
a	*3*
b	*3*
m	*3*
p	*3*

By-Product of First Scan of Database

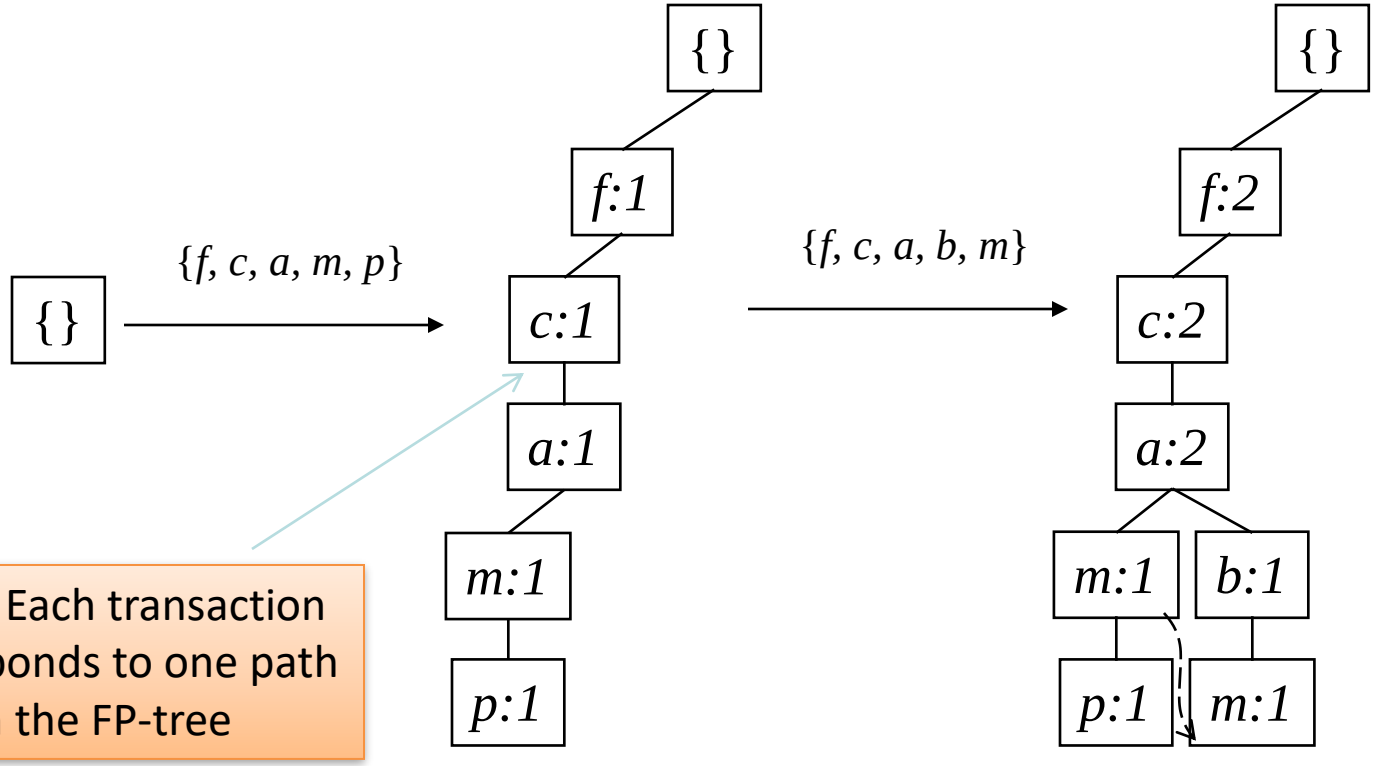
FP-tree Example: step 2

Step 2: scan the DB for the second time, order frequent items in each transaction

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

FP-tree Example: step 2

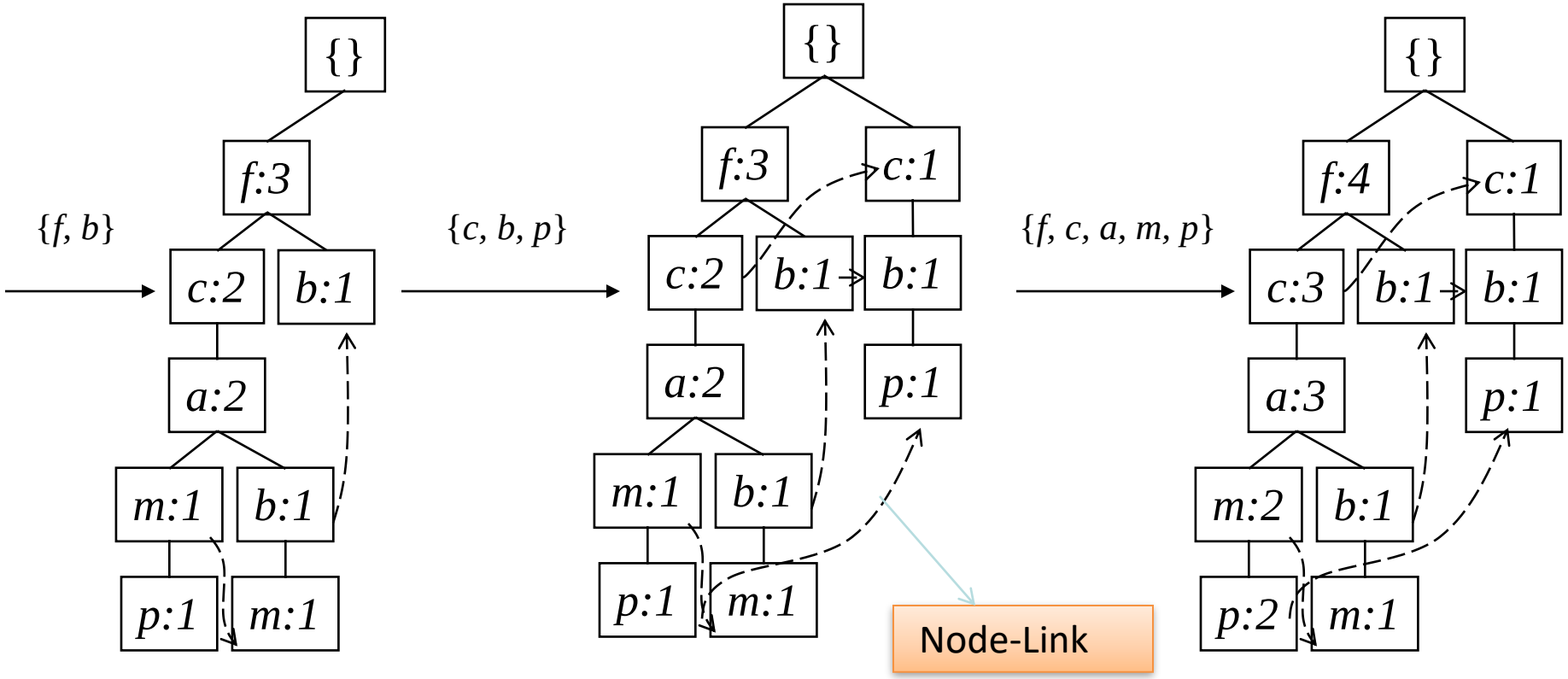
Step 2: construct FP-tree



NOTE: Each transaction corresponds to one path in the FP-tree

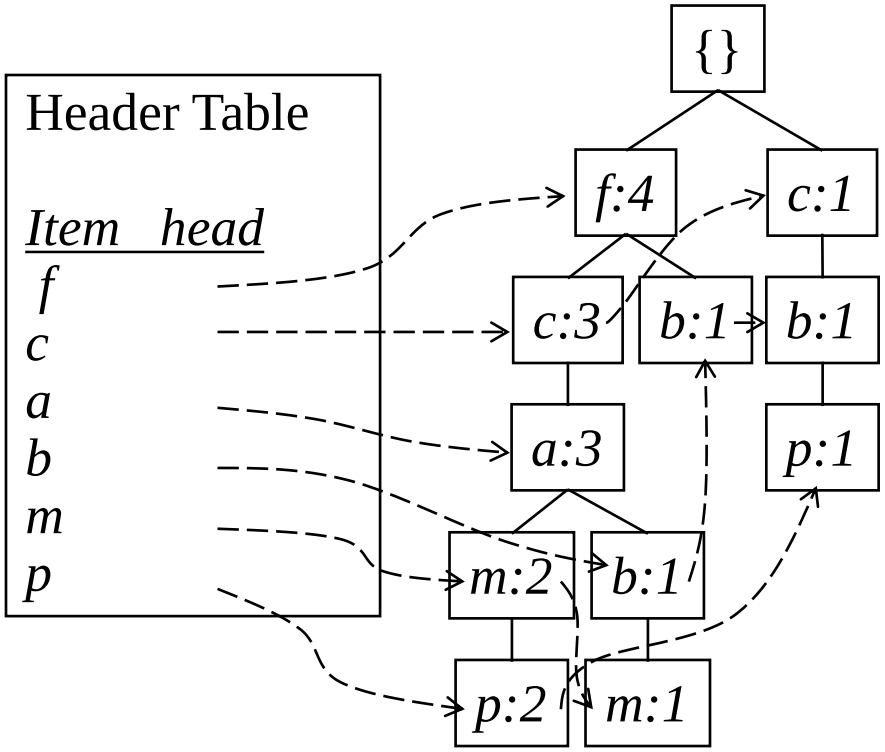
FP-tree Example: step 2

Step 2: construct FP-tree



Construction Example

Final FP-tree



FP-growth:
Mining Frequent Patterns Using FP-tree

3 Major Steps

Starting the processing from the end of list L:

Step 1:

Construct **conditional pattern base** for each item in the header table

Step 2

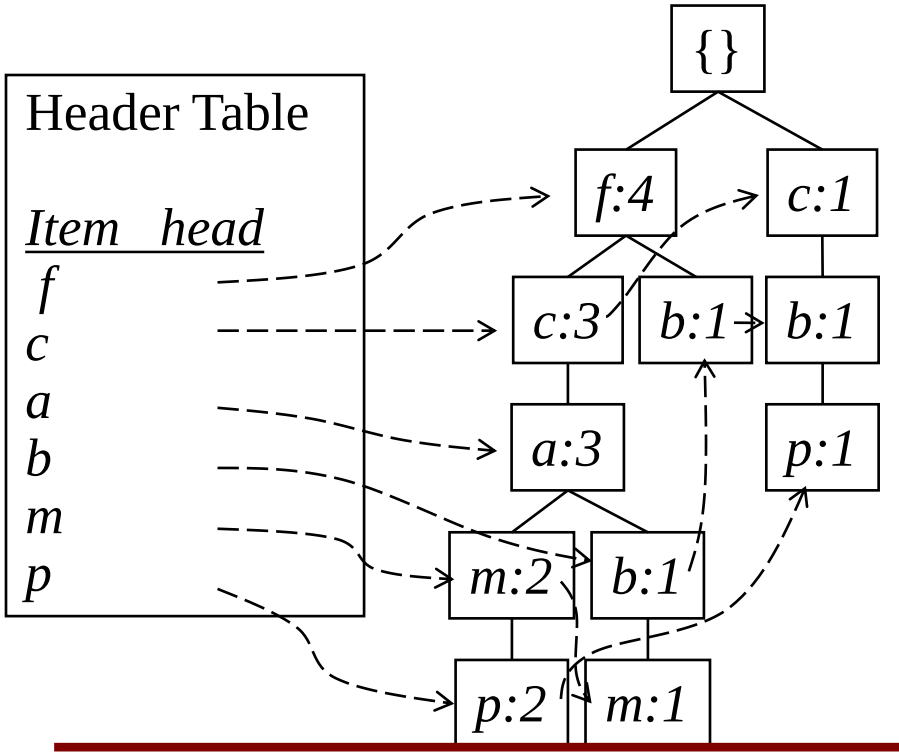
Construct **conditional FP-tree** from each conditional pattern base

Step 3

Recursively mine conditional FP-trees and grow frequent patterns obtained so far. If the conditional FP-tree contains a single path, simply enumerate all the patterns

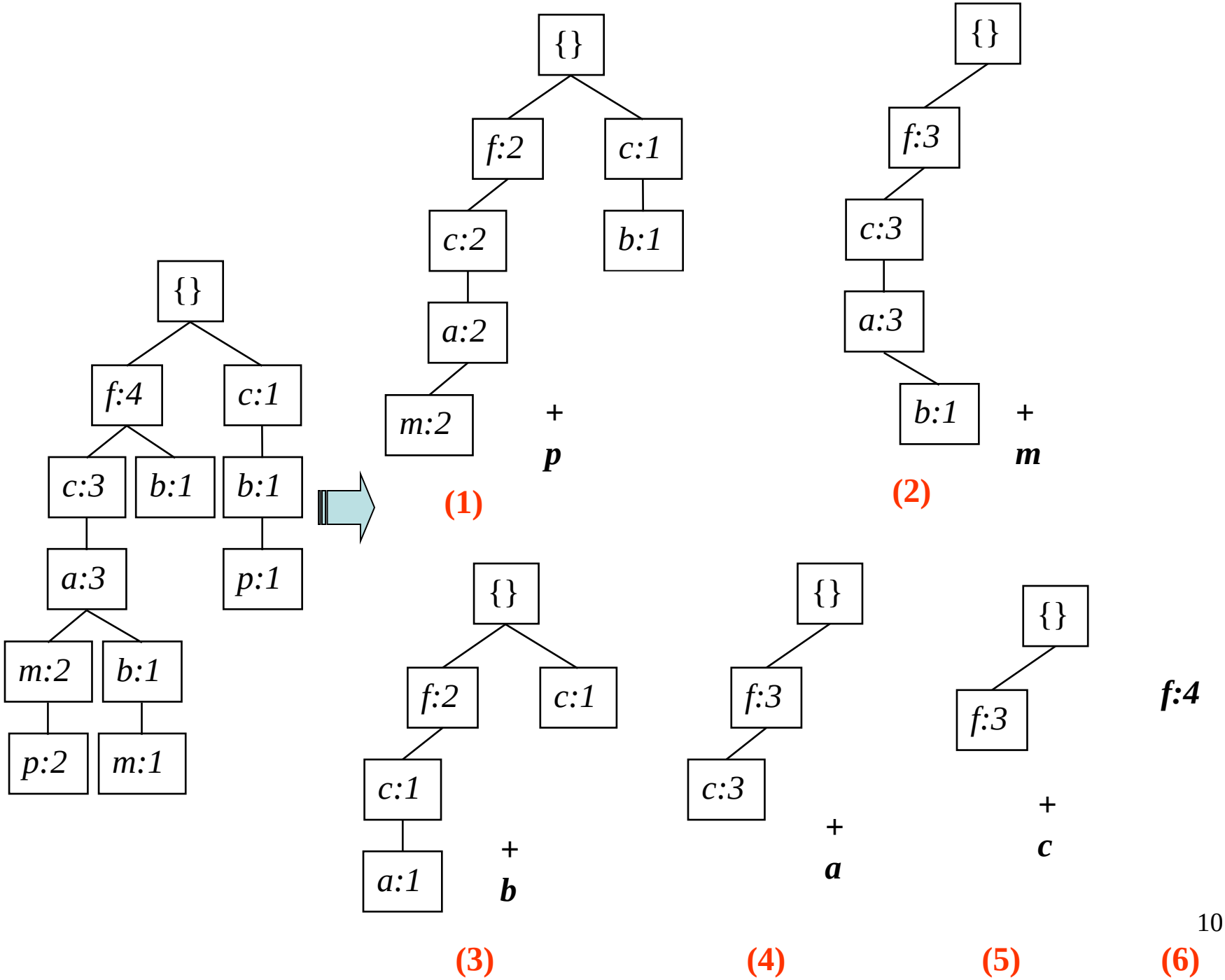
Step 1: Construct Conditional Pattern Base

- Starting at the bottom of frequent-item header table in the FP-tree
- Traverse the FP-tree by following the link of each frequent item
- Accumulate all of **transformed prefix paths** of that item to form a **conditional pattern base**



***Conditional* pattern bases**

item	*cond. pattern base*
p	***fcam:2, cb:1***
m	***fca:2, fcab:1***
b	***fca:1, f:1, c:1***
a	***fc:3***
c	***f:3***
f	***{}***


{}
f:4
c:1
c:3
b:1
b:1
a:3
p:1
m:2
b:1
p:2
m:1
{}
f:2
c:1
c:2
b:1
a:2
m:2
+
p
(1)
{}
f:3
c:3
a:3
b:1
+
m
(2)
{}
f:2
c:1
c:1
a:1
+
b
(3)
{}
f:3
c:3
+
a
(4)
{}
f:3
+
c
(5)
f:4
(6)

Conditional Pattern Bases and Conditional FP-Tree

Item	Conditional pattern base	Conditional FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)}|p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)}|m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Empty	Empty

order of L

min_sup = 3

1	*f, c, a, m, p*
2	*f, c, a, b, m*
3	*f, b*
4	*c, b, p*
5	*f, c, a, m, p*

1	*f, c, a, m*
4	*c, b*
5	*f, c, a, m*

+ *p* → | | |
|---|---|
| 1 | *c* |
| 4 | *c* |
| 5 | *c* |

+ *p* → *p*: 3, *cp*: 3

1	*f, c, a*
2	*f, c, a, b*
5	*f, c, a*

+ *m* → | | |
|---|---|
| 1 | *f, c, a* |
| 2 | *f, c, a* |
| 5 | *f, c, a* |

+ *m* → *m*: 3, *fm*: 3, *cm*: 3, *am*: 3, *fcm*: 3, *fam*: 3, *cam*: 3, *fcam*: 3

2	*f, c, a*
3	*f*
4	*c*

+ *b* → *b*: 3

1	*f, c*
2	*f, c*
5	*f, c*

+ *a* → *a*: 3, *fa*: 3, *ca*: 3, *fca*: 3

1	*f*
2	*f*
4	
5	*f*

+ *c* → *c*: 4, *fc*: 3

f: 1,2,3,5 → *f*: 4

Discusion

- Advantages of FP-Growth
  - only 2 passes over data-set
  - no candidate generation
  - much faster than Apriori
- Disadvantages of FP-Growth
  - FP-Tree may not fit in memory!!
  - FP-Tree is expensive to build

Challenges

- A major challenge in mining frequent itemsets from a large data set is the fact that such mining often generates a huge number of itemsets satisfying the minimum support threshold, especially when *minsup* is set low.
- This is because if an itemset is frequent, each of its subsets is frequent as well.
- A long itemset will contain a combinatorial number of shorter, frequent sub-itemsets.
- For example, a frequent itemset of length 100, such as {a1, a2, , , , a100}, contains $\binom{100}{1}$ =100 frequent 1-itemsets: a1, a2, , , , , a100, $\binom{100}{2}$ frequent 2-itemsets: (a1, a2), (a1, a3), , , , , (a99, a100), and so on. The total number of frequent itemsets that it contains is thus,

$$\binom{100}{1} + \binom{100}{2} + ... + \binom{100}{100} = 2^{100} - 1 \approx 1.27X10^{30} \qquad (3)$$

Frequent Closed Itemset Mining

Definition 6 (Frequent Closed Itemset)

An itemset X is called frequent closed itemset if and only if it is frequent and there exists no proper superset X'', $(X \subset X'')$ such that support of X is same as the support of X'', $sup(X)=sup(X'')$.

Definition 7 (Frequent Maximal Itemset)

An itemset X is called frequent maximal itemset if and only if it is frequent and there exists no proper superset X'', $(X \subset X'')$.

- Suppose that a transaction database has only two transactions: $\{\langle$a1, a2, , , , , a100$\rangle$; $\langle$a1, a2, : : : , a50$\rangle\}$. Let the minimum support count threshold be $minsup = 1$.
- Ttwo closed frequent itemsets and their support counts, that is, C = {{a1, a2, : : : , a100} : 1; {a1, a2, : : : , a50g : 2}}.

Frequent Closed Itemset Mining

- Suppose that a transaction database has only two transactions: $\{\langle$a1, a2, , , , , a100$\rangle$; $\langle$a1, a2, : : : , a50$\rangle\}$. Let the minimum support count threshold be $minsup = 1$.
- Two closed frequent itemsets and their support counts, that is, C = $\{\{$a1, a2, : : : , a100$\}$: 1; $\{$a1, a2, , , , , a50g : 2$\}\}$.
- There is one maximal frequent itemset M = $\{\{$a1, a2, , , , , a100$\}$: 1$\}$. (We cannot include $\{$a1, a2, , , , , a50$\}$ as a maximal frequent itemset because it has a frequent super-set, $\{$a1, a2, , , , , a100$\}$.)

Frequent Closed Itemset Mining from High Dimensional Dataset

- The conventional algorithms mine frequent itemsets, frequent closed itemset and frequent maximal itemset from the transactional datasets.
- In the modern era, the abundant data across variety of domains, including bioinformatics has led to the new form of dataset known as a high dimensional dataset, whose data characteristics are different from that of transactional datasets.
- The high dimensional datasets consist of less number of rows and considerably large number of features.
- The amount of information that can be extracted from high dimensional datasets is potentially huge, but extraction of information and knowledge from these datasets is a non-trivial task.

Frequent Closed Itemset Mining from High Dimensional Dataset

- The conventional algorithms adopt feature enumeration based approach for mine frequent closed itemsets.
- The conventional algorithms face an uphill task in mining frequent closed itemsets from the high dimensional datasets.
- To overcome the inefficiency and uphill task of these algorithms, sequential row enumerated algorithms were proposed to mine FCI from high dimensional datasets.
- This problem of inefficiency can be solved to the greater extent by parallel row enumerated algorithms.

Frequent Colossal Itemset Mining

- The result of frequent closed itemset mining algorithms includes small and mid-sized itemsets, which does not enclose valuable and complete information in many applications.
- In application dealing with high dimensional datasets such as bioinformatics, association rule mining gives greater importance to the large-sized itemsets called as colossal itemsets.

Definition 8 (Frequent Colossal Itemset)

An itemset X is called frequent colossal itemset if and only if it is frequent and $card(X) \geq mincard$, where $mincard$ is user specified least cardinality threshold.

Example 6

In Table 1, the itemset $X = \{f_1, f_2, f_6, f_{10}\}$ is frequent colossal itemset with minimum support threshold set to 2 and minimum cardinality threshold set to 4, $sup(X) \geq 2$ and $card(X) \geq 4$.

Frequent Colossal Closed Itemset Mining

Definition 9 (Frequent Colossal Closed Itemset)

An itemset X is called frequent colossal closed itemset if and only if it is frequent closed and $card(X) \geq mincard$, where $mincard$ is user specified least cardinality threshold.

Example 7

In Table 1, the itemset $X = \{f_2, f_4, f_7, f_8\}$, is frequent colossal closed itemset with minimum support threshold set to 2 and minimum cardinality threshold set to 4, $sup(X) \geq 2$ and $card(X) \geq 4$.

Apriori Algorithm

- Apriori: A Candidate Generation-and-Test Approach
- Improving the Efficiency of Apriori
- FPGrowth: A Frequent Pattern-Growth Approach
- ECLAT: Frequent Pattern Mining with Vertical Data Format
- Mining Close Frequent Patterns and Maxpatterns
- The disadvantage of Apriori algorithm are:
  - It may need to generate a huge number of candidate sets.
  - It may need to repeatedly scan the database and check a large set of candidates by pattern matching.
  - Breadth-first search approach

FP-growth Algorithm

- Frequent Pattern growth (FP-growth) Algorithm was proposed by J. Han, J. Pei, and Y. Yin.
- Mining frequent itemsets without candidate generation. FP-growth adopts depth-first search approach.
- The first scan of the database is the same as Apriori, which derives the set of frequent items (1-itemsets) and their support counts (frequencies).
- Let the minimum support count be 2. The set of frequent items is sorted in the order of descending support count.

FP-growth Algorithm

- An FP-tree is then constructed as follows. First, create the root of the tree, labeled with "null." Scan database D a second time.

- The items in each transaction are processed according to descending support count order and a branch is created for each transaction.

Table 4

Dataset *D*

TID	List of items
1	I1, I2, I5
2	I2, I4
3	I2, I3
4	I1, I2, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I1, I2, I3, I5
9	I1, I2, I3

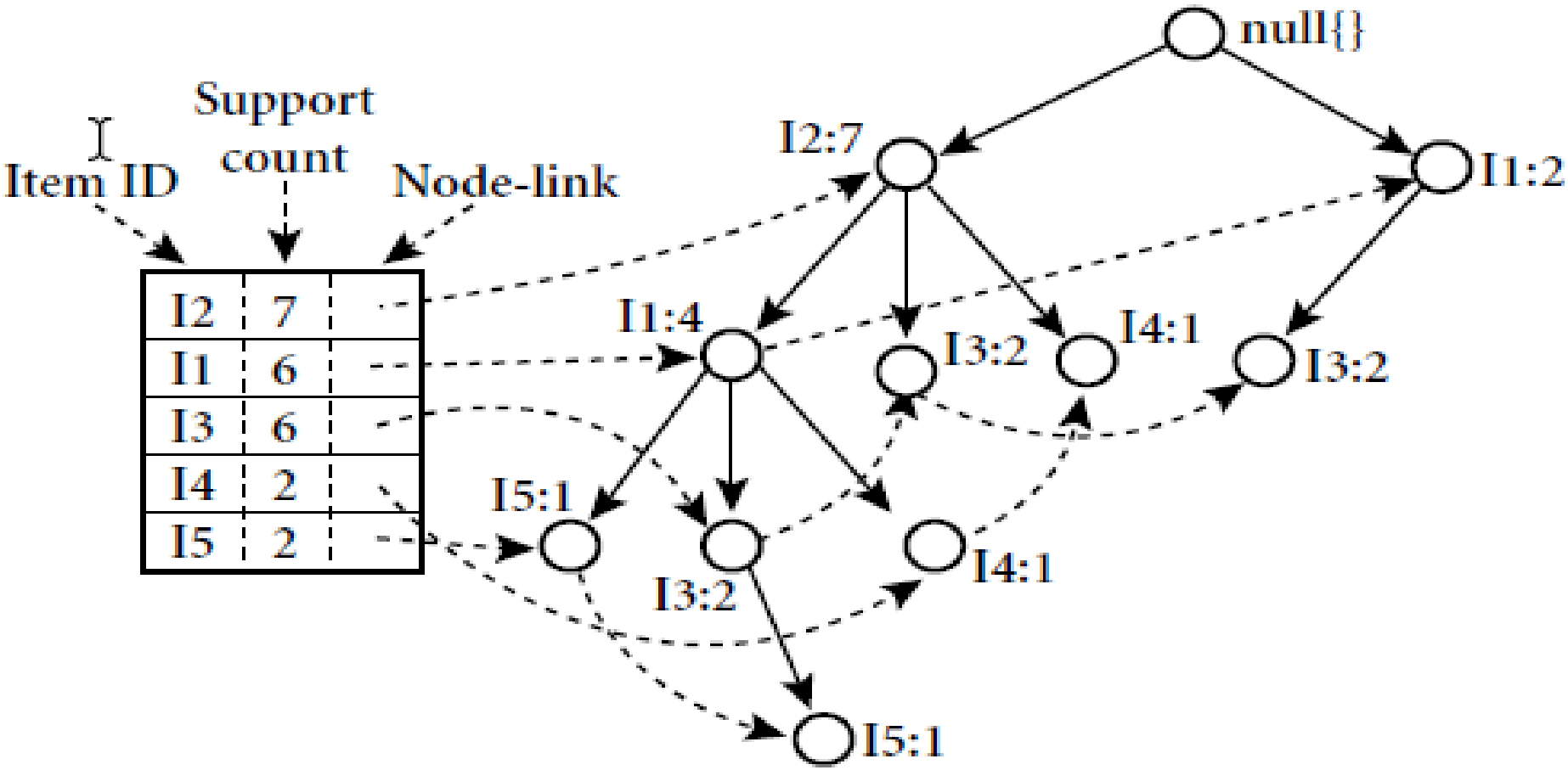
FP-growth Algorithm

- The set of frequent items is sorted in the order of descending support count. Thus, we have {{I2: 7}, {I1: 6}, {I3: 6}, {I4: 2}, {I5: 2}}.
- The items in each transaction are processed according to descending support count order and a branch is created for each transaction.

Table 5
Dataset D

TID	List of items
1	I2, I1, I5
2	I2, I4
3	I2, I3
4	I2, I1, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I2, I1, I3, I5
9	I2, I1, I3

FP-growth Algorithm



An FP-tree registers compressed, frequent pattern information.

FP-growth Algorithm

- The FP-tree is mined as follows.
- Start from each frequent length-1 pattern (as an initial suffix pattern).
- Construct its conditional pattern base (a "subdatabase," which consists of the set of prefix paths in the FP-tree co-occurring with the suffix pattern).
- Then construct its (conditional) FP-tree, and perform mining recursively on such a tree.
- The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-tree.

FP-growth Algorithm

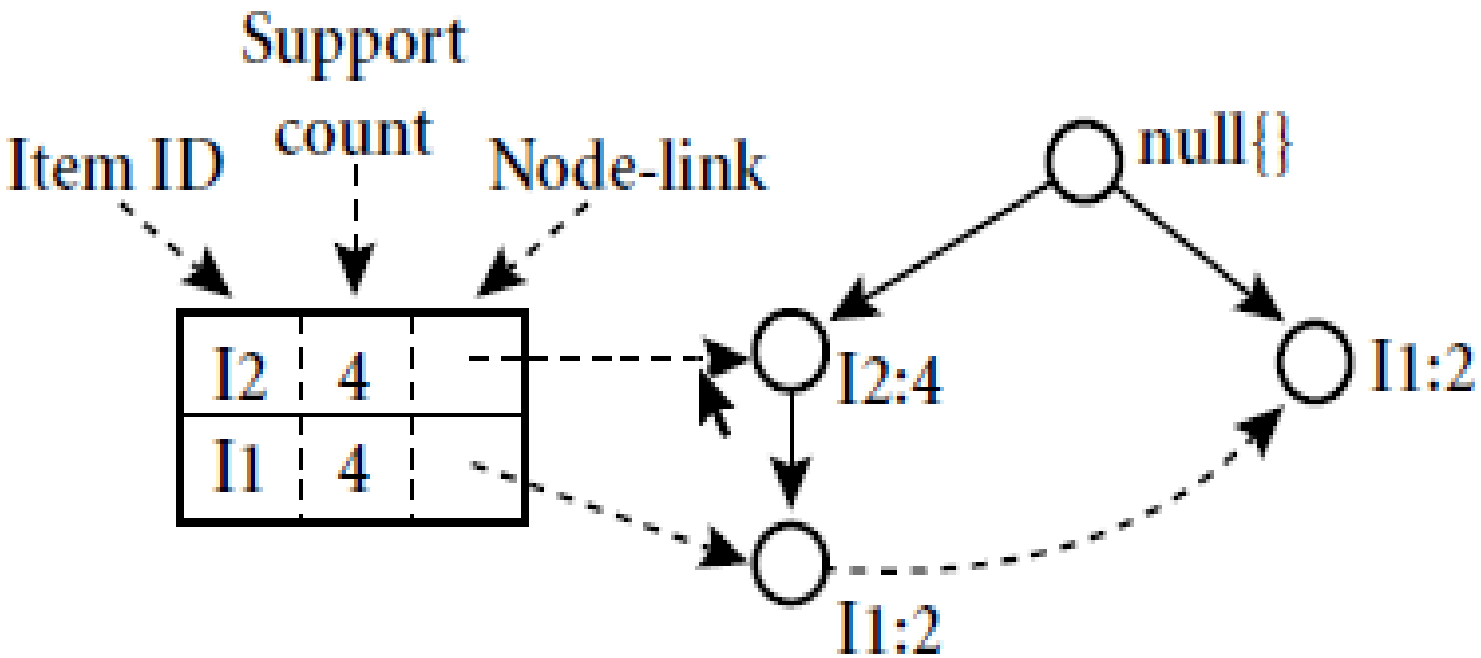
Mining the FP-tree by creating conditional (sub-)pattern bases.

Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	⟨I2: 2, I1: 2⟩	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	⟨I2: 2⟩	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	⟨I2: 4, I1: 2⟩, ⟨I1: 2⟩	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}
I1	{{I2: 4}}	⟨I2: 4⟩	{I2, I1: 4}

FP-growth Algorithm

- Lets first consider I5. I5 occurs in two branches of the FP-tree.
- The paths formed by these branches are $\langle$I2, I1, I5: 1$\rangle$ and $\langle$I2, I1, I3, I5: 1$\rangle$.
- Considering I5 as a suffix, its corresponding two prefix paths are $\langle$I2, I1: 1$\rangle$ and $\langle$I2, I1, I3: 1$\rangle$, which form its conditional pattern base.
- Its conditional FP-tree contains only a single path, $\langle$I2: 2, I1: 2$\rangle$; I3 is not included because its support count of 1 is less than the minimum support count.
- The single path generates all the combinations of frequent patterns: $\{$I2, I5: 2$\}$, $\{$I1, I5: 2$\}$, $\{$I2, I1, I5: 2$\}$.

FP-growth Algorithm



The conditional FP-tree associated with the conditional node I3.

FP-growth Algorithm

- I3's conditional pattern base is $\{\{I2, I1: 2\}, \{I2: 2\}, \{I1: 2\}\}$.
- Its conditional FP-tree has two branches, $\langle I2: 4, I1: 2\rangle$ and $\langle I1: 2\rangle$
- Generates the set of patterns, $\{\{I2, I3: 4\}, \{I1, I3: 4\}, \{I2, I1, I3: 2\}\}$.

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation

What is a Data Warehouse?

- Defined in many different ways, but not rigorously.
  - A decision support database that is maintained separately from the organization's operational database
  - Support information processing by providing a solid platform of consolidated, historical data for analysis.
- "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process."—W. H. Inmon
- Data warehousing:
  - The process of constructing and using data warehouses

Data Warehouse—Subject-Oriented

- Organized around major subjects, such as customer, product, sales
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing
- Provides a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process

Data Warehouse—Integrated

- Constructed by integrating multiple, heterogeneous data sources
  - relational databases, flat files, on-line transaction records
- Data cleaning and data integration techniques are applied.
  - Ensure consistency in naming conventions, encoding structures, attribute measures, etc. among different data sources
    - E.g., Hotel price: currency, tax, breakfast covered, etc.
  - When data is moved to the warehouse, it is converted.

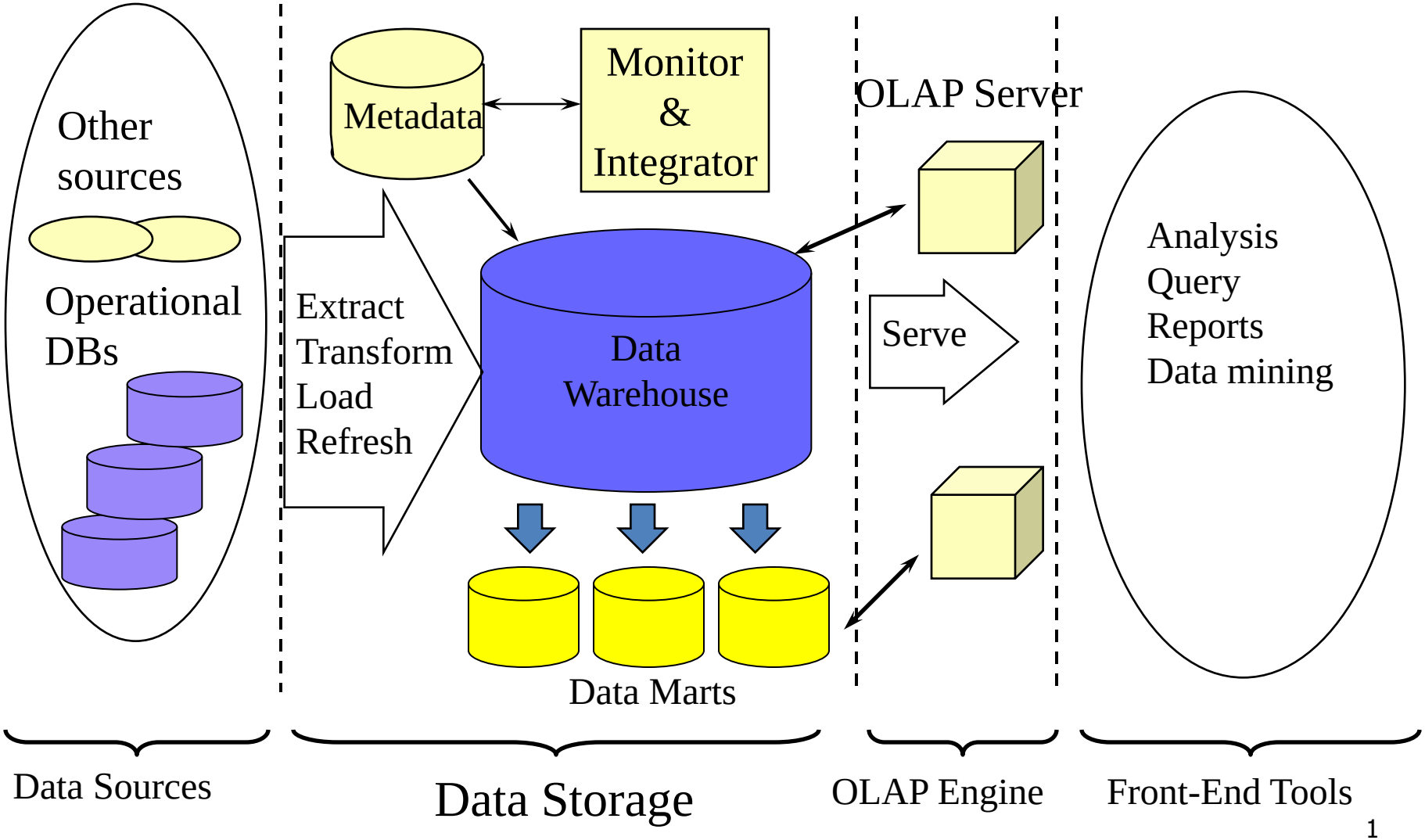
Data Warehouse—Nonvolatile

- A physically separate store of data transformed from the operational environment
- Operational update of data does not occur in the data warehouse environment
  - Does not require transaction processing, recovery, and concurrency control mechanisms
  - Requires only two operations in data accessing:
    - *initial loading of data* and *access of data*

OLTP vs. OLAP

	OLTP	OLAP
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

Data Warehouse: A Multi-Tiered Architecture



Three Data Warehouse Models

- Enterprise warehouse
  - collects all of the information about subjects spanning the entire organization

- Data Mart
  - a subset of corporate-wide data that is of value to a specific groups of users. Its scope is confined to specific, selected groups, such as marketing data mart

- Virtual warehouse
  - A set of views over operational databases
  - Only some of the possible summary views may be materialized

Extraction, Transformation, and Loading (ETL)

- **Data extraction**
  - get data from multiple, heterogeneous, and external sources
- **Data cleaning**
  - detect errors in the data and rectify them when possible
- **Data transformation**
  - convert data from legacy or host format to warehouse format
- **Load**
  - sort, summarize, consolidate, compute views, check integrity, and build indicies and partitions
- **Refresh**
  - propagate the updates from the data sources to the warehouse

Metadata Repository

- **Meta data** is the data defining warehouse objects. It stores:
- Description of the structure of the data warehouse
  - schema, view, dimensions, hierarchies, derived data defn, data mart locations and contents

- Operational meta-data
  - data lineage (history of migrated data and transformation path), currency of data (active, archived, or purged), monitoring information (warehouse usage statistics, error reports, audit trails)

- The algorithms used for summarization
- The mapping from operational environment to the data warehouse
- Data related to system performance
  - warehouse schema, view and derived data definitions
- Business data

- business terms and definitions, ownership of data, charging policies

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage

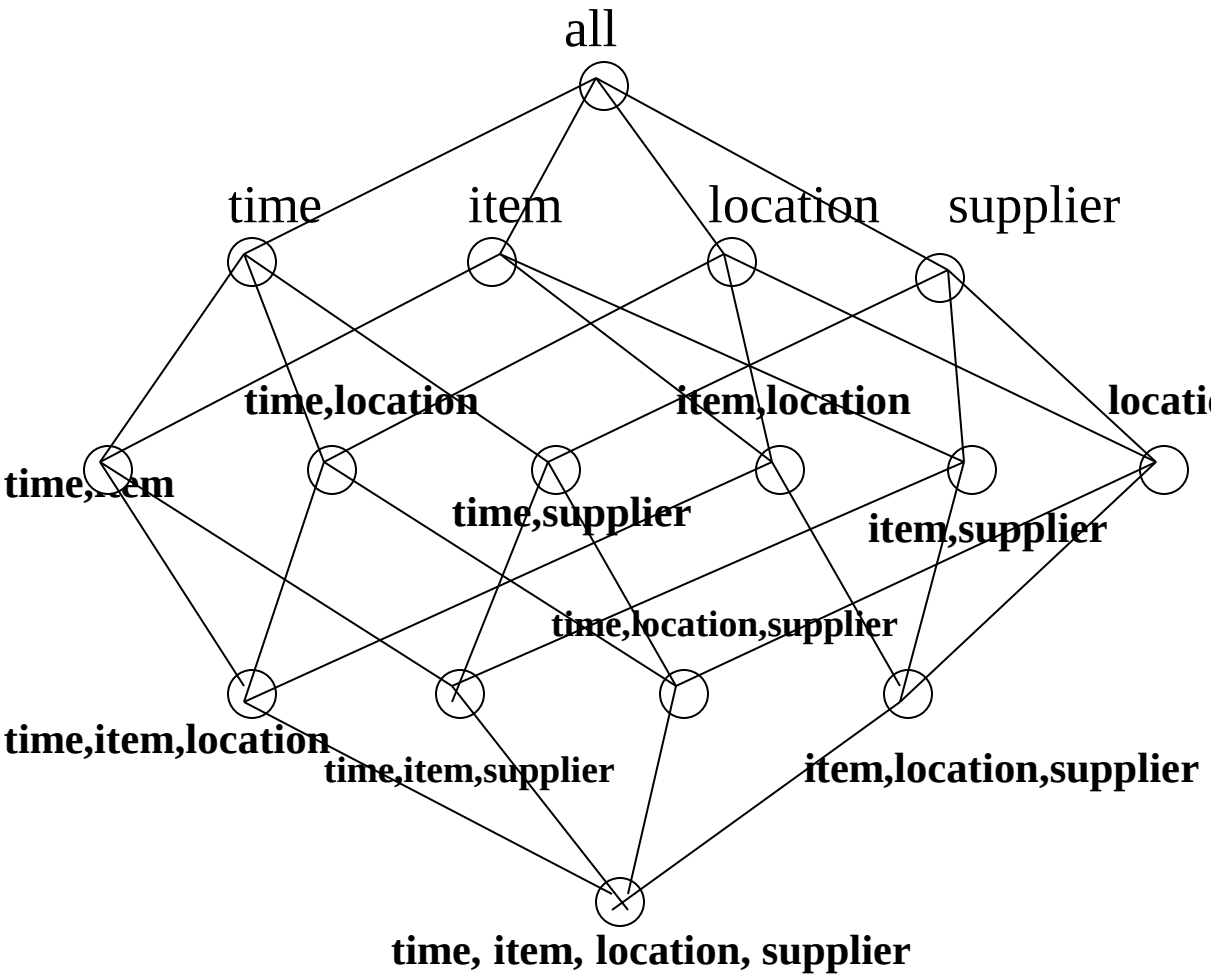
From Tables and Spreadsheets to Data Cubes

- A **data warehouse** is based on a multidimensional data model which views data in the form of a data cube
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
  - **Dimension tables**, such as item (item_name, brand, type), or time(day, week, month, quarter, year)

  - **Fact table** contains **measures** (such as dollars_sold) and keys to each of the related dimension tables

- In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.

Cube: A Lattice of Cuboids



0-D (*apex*) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

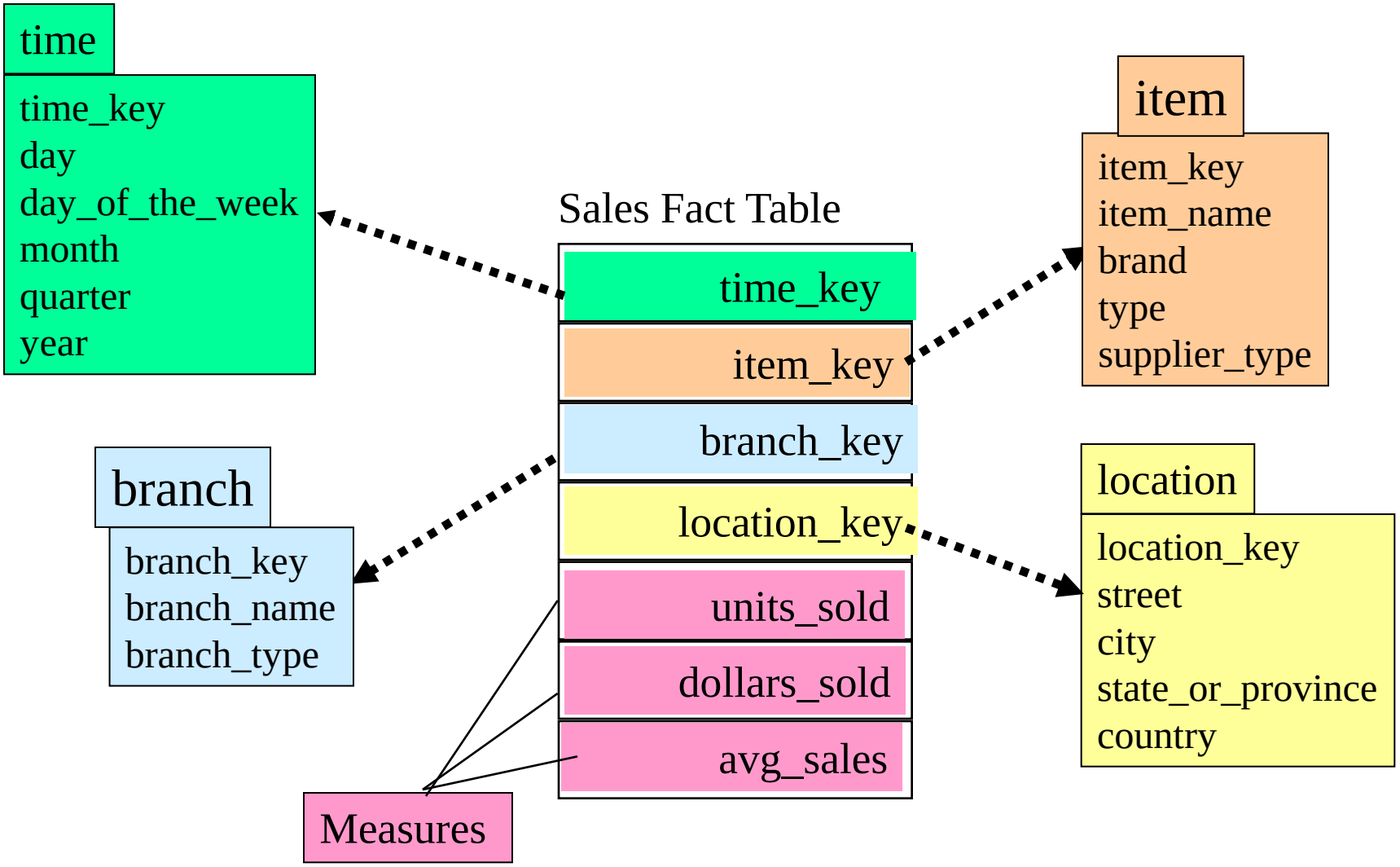
4-D (*base*) cuboid

Conceptual Modeling of Data Warehouses

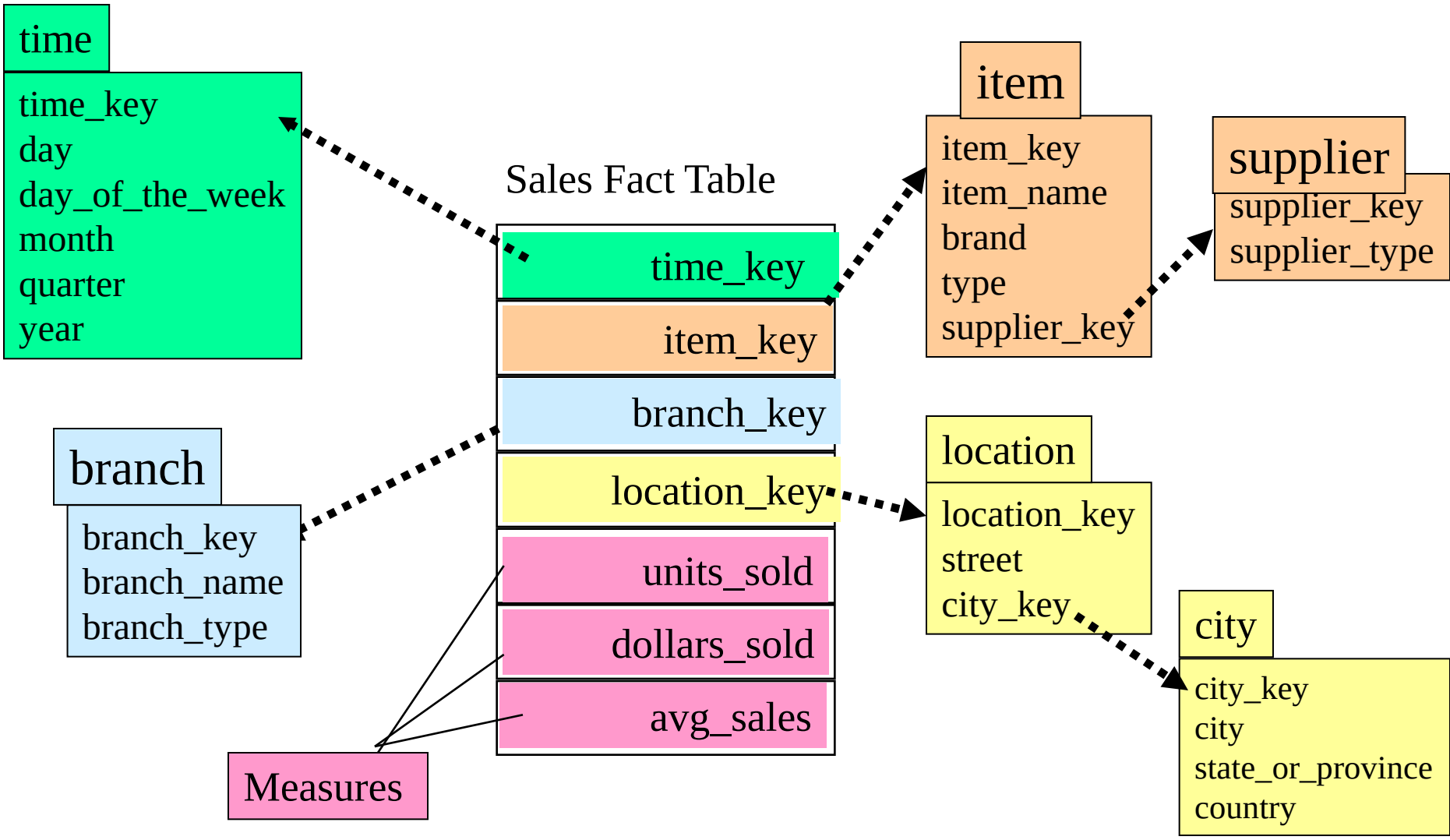
- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake

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

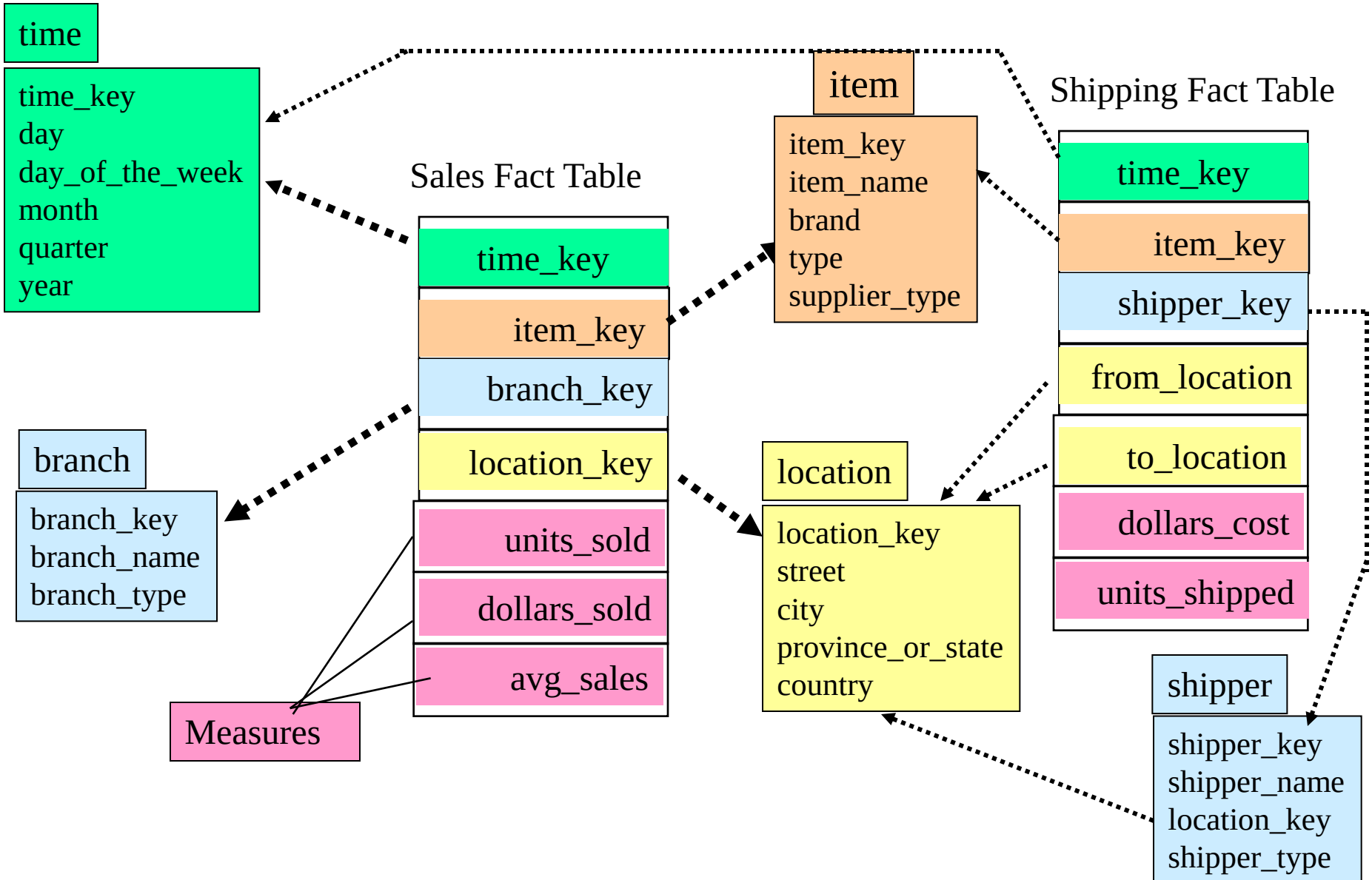
Example of **Star Schema**



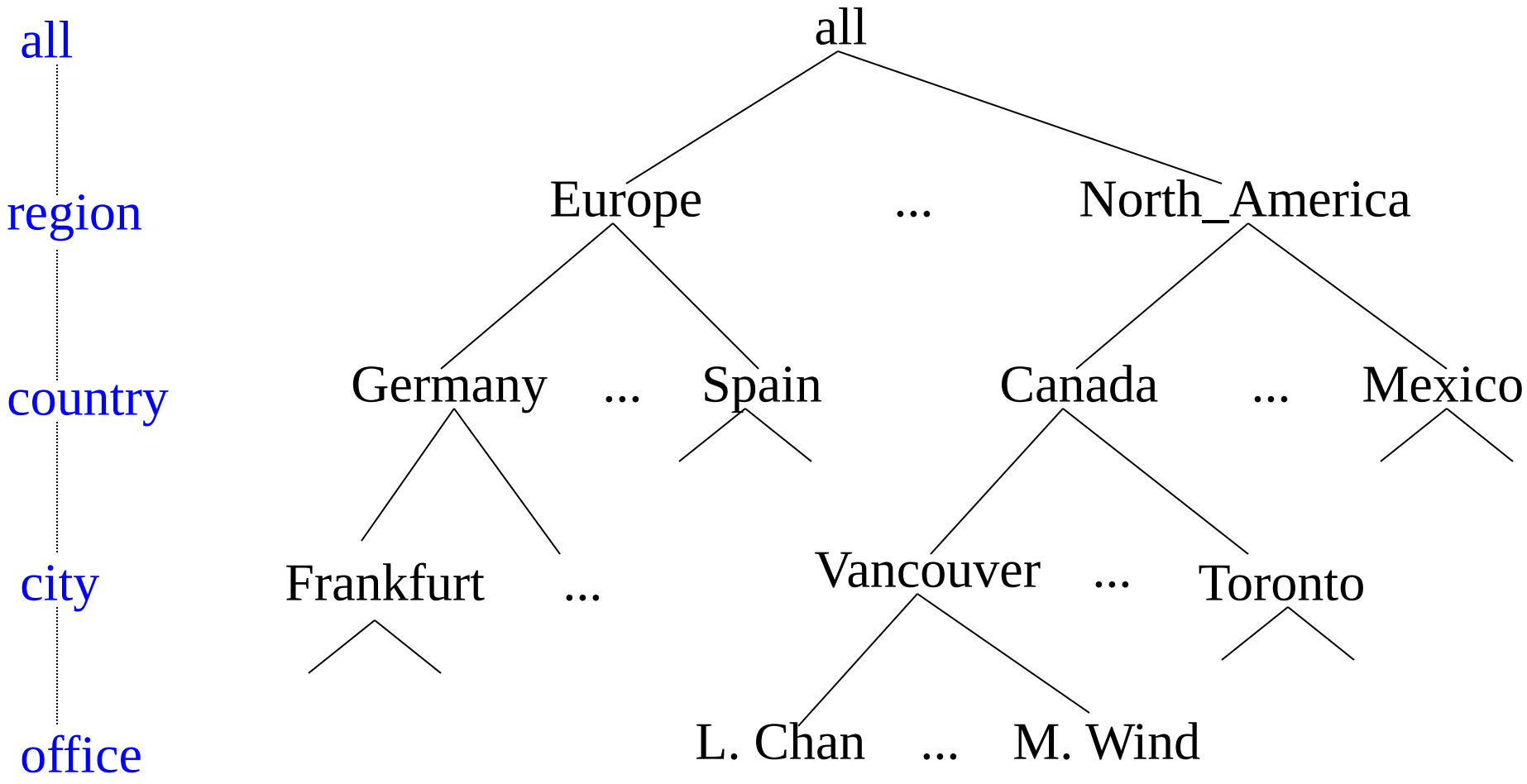
Example of **Snowflake Schema**



Example of **Fact Constellation**

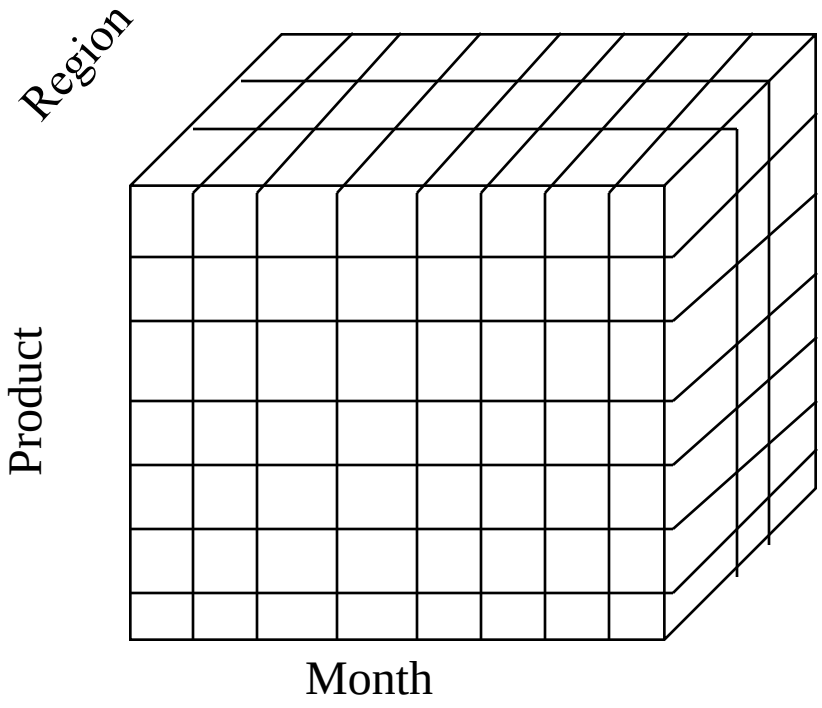


A Concept Hierarchy: **Dimension** (location)

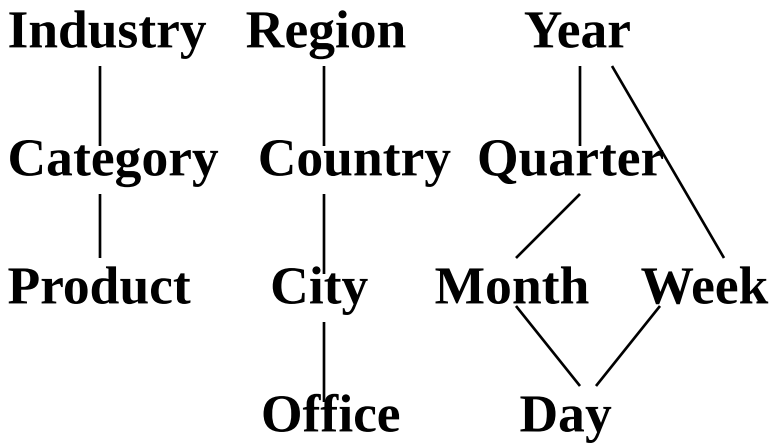


Multidimensional Data

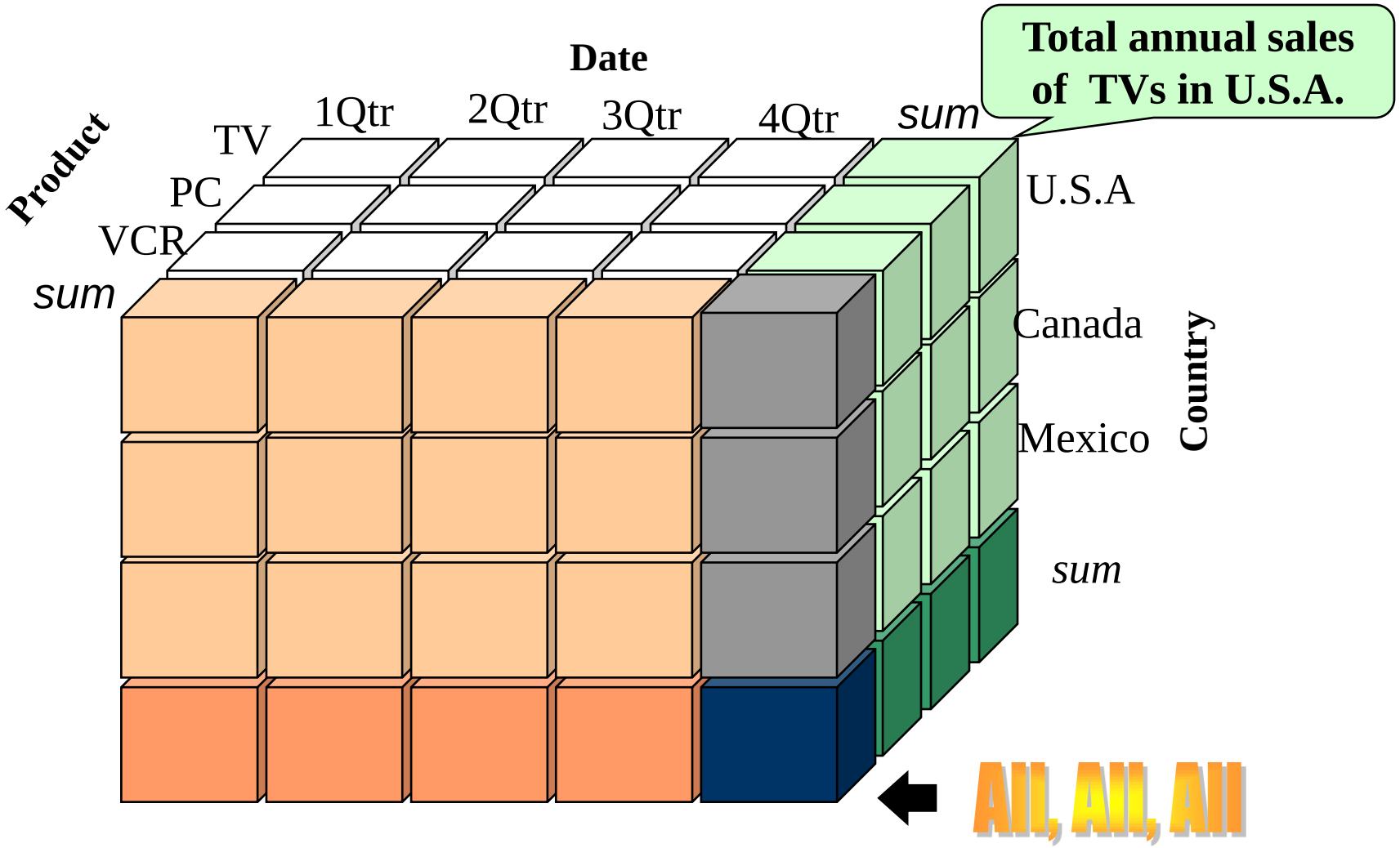
- Sales volume as a function of product, month, and region



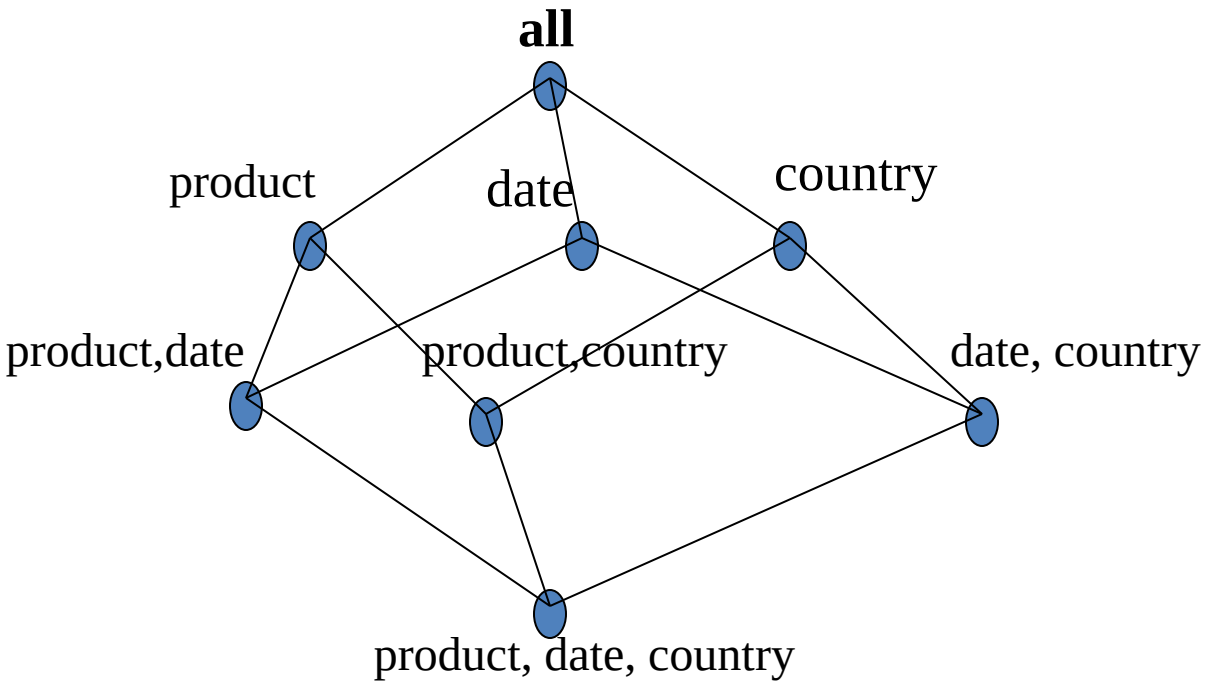
Dimensions: *Product, Location, Time*
Hierarchical summarization paths



A Sample Data Cube



Cuboids Corresponding to the Cube



0-D (*apex*) cuboid

1-D cuboids

2-D cuboids

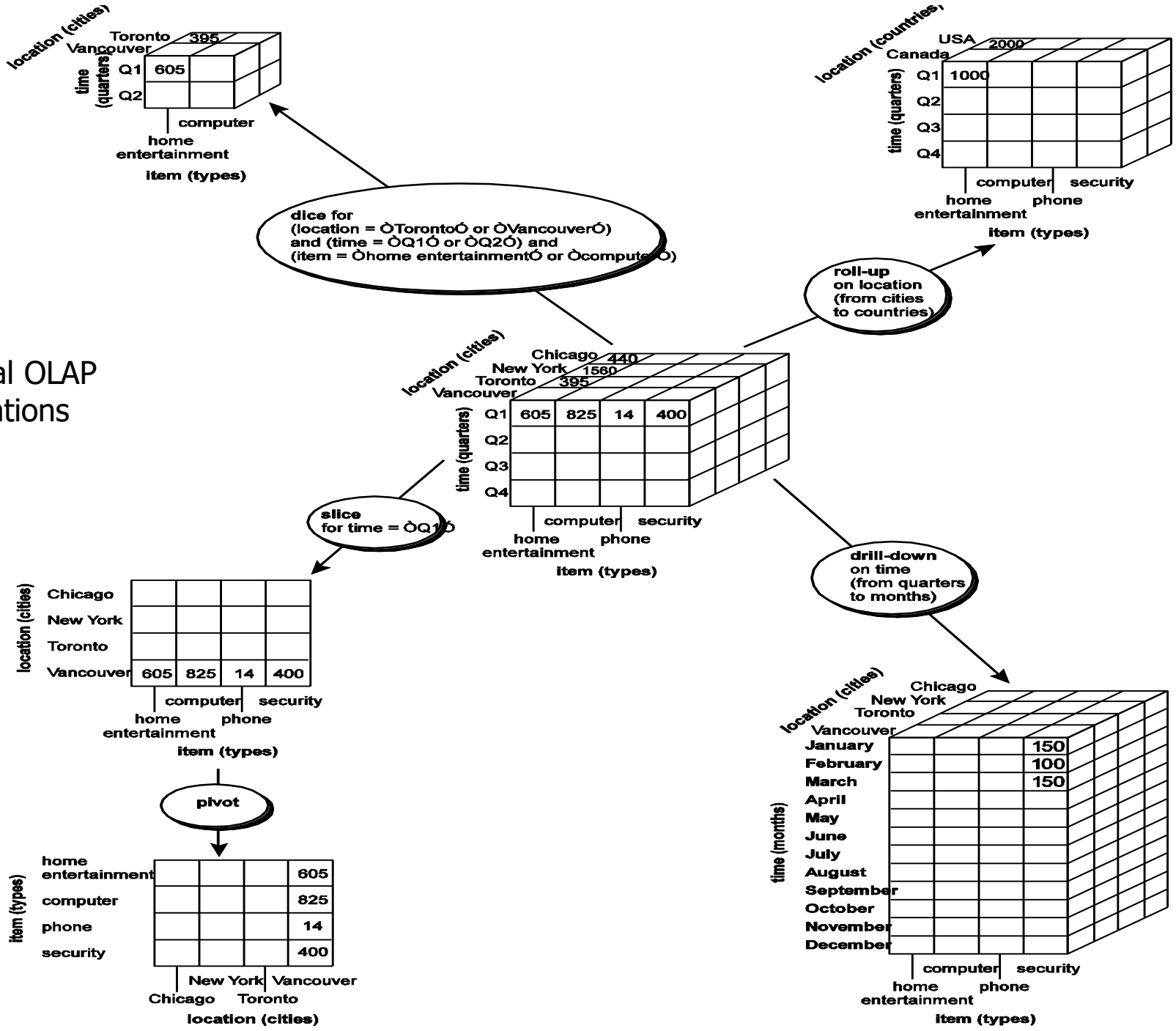
3-D (*base*) cuboid

Typical OLAP Operations

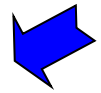
- Roll up (drill-up): summarize data
  - *by climbing up hierarchy or by dimension reduction*
- Drill down (roll down): reverse of roll-up
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- Slice and dice: *project and select*

- Pivot (rotate):
  - *reorient the cube, visualization, 3D to series of 2D planes*

Typical OLAP Operations



Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage 
- Data Warehouse Implementation
- Data Generalization by Attribute-Oriented Induction
- Summary

Design of Data Warehouse: A Business Analysis Framework

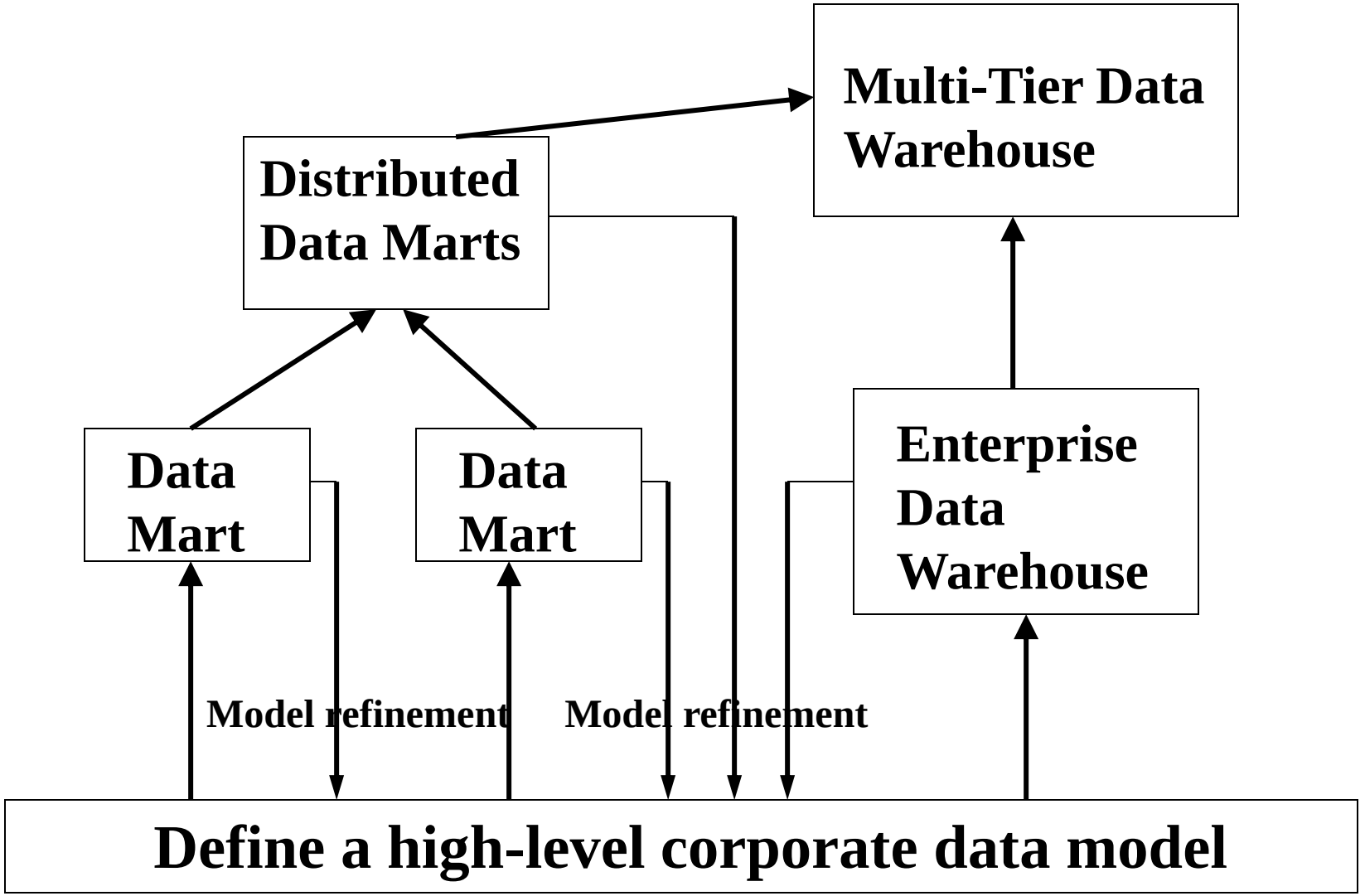
- Four views regarding the design of a data warehouse
  - Top-down view
    - allows selection of the relevant information necessary for the data warehouse
  - Data source view
    - exposes the information being captured, stored, and managed by operational systems
  - Data warehouse view
    - consists of fact tables and dimension tables
  - Business query view
    - sees the perspectives of data in the warehouse from the view of end-user

Data Warehouse Design Process

- **Top-down, bottom-up approaches or a combination** of both
  - Top-down: Starts with overall design and planning (mature)
  - Bottom-up: Starts with experiments and prototypes (rapid)

- **Typical data warehouse design process**
  - Choose a business process to model, e.g., orders, invoices, etc.
  - Choose the *grain* (*atomic level of data*) of the business process
  - Choose the dimensions that will apply to each fact table record
  - Choose the measure that will populate each fact table record

Data Warehouse Development: A Recommended Approach



Data Warehouse Usage

- Three kinds of data warehouse applications
  - Information processing
    - supports querying, basic statistical analysis, and reporting using crosstabs, tables, charts and graphs
  - Analytical processing
    - multidimensional analysis of data warehouse data
    - supports basic OLAP operations, slice-dice, drilling, pivoting
  - Data mining
    - knowledge discovery from hidden patterns
    - supports associations, constructing analytical models, performing classification and prediction, and presenting the mining results using visualization tools

From On-Line Analytical Processing (OLAP) to On Line Analytical Mining (OLAM)

- Why online analytical mining?
  - High quality of data in data warehouses
    - DW contains integrated, consistent, cleaned data
  - Available information processing structure surrounding data warehouses
    - Web accessing, service facilities, reporting and OLAP tools
  - OLAP-based exploratory data analysis
    - Mining with drilling, dicing, pivoting, etc.
  - On-line selection of data mining functions
    - Integration and swapping of multiple mining functions, algorithms, and tasks

Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Data Warehouse Design and Usage
- Data Warehouse Implementation 
- Data Generalization by Attribute-Oriented Induction
- Summary

Efficient Data Cube Computation

- Data cube can be viewed as a lattice of cuboids
  - The bottom-most cuboid is the base cuboid
  - The top-most cuboid (apex) contains only one cell
  - How many cuboids in an n-dimensional cube with L levels? $$T = \prod_{i=1}^{n} (L_i + 1)$$
- Materialization of data cube
  - Materialize every (cuboid) (**full materialization**), none (**no materialization**), or some (**partial materialization**)
  - Selection of which cuboids to materialize
    - Based on size, sharing, access frequency, etc.

The "Compute Cube" Operator

- Cube definition and computation in DMQL

  define cube sales [item, city, year]: sum (sales_in_dollars)

  compute cube sales

- Transform it into a SQL-like language (with a new operator cube by, introduced by Gray et al.'96)

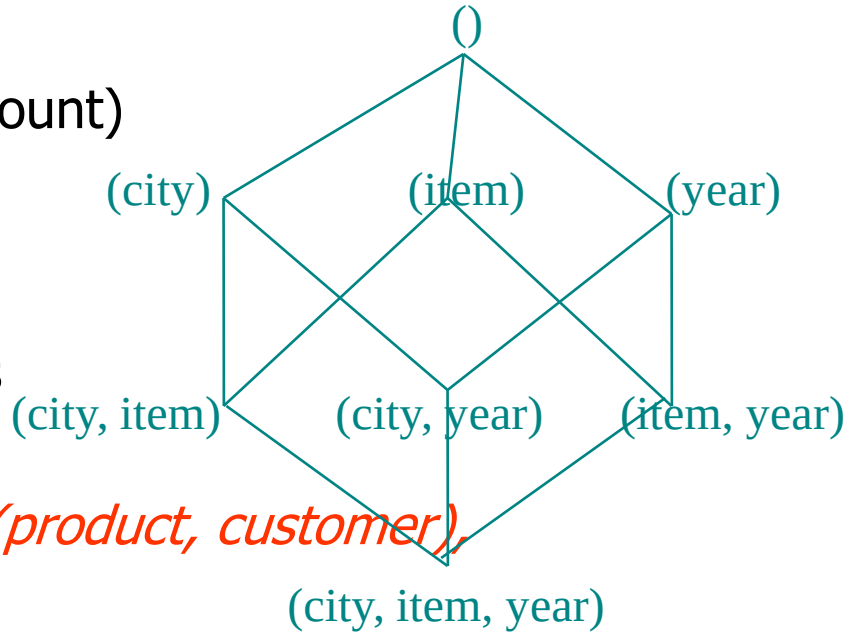
  SELECT item, city, year, SUM (amount)

  FROM SALES

  CUBE BY item, city, year

- Need compute the following Group-Bys

  (date, product, customer),
  (date,product),(date, customer), (product, customer),
  (date), (product), (customer)
  ()



OLAP Server Architectures

- Relational OLAP (ROLAP)
  - Use relational or extended-relational DBMS to store and manage warehouse data and OLAP middle ware
  - Include optimization of DBMS backend, implementation of aggregation navigation logic, and additional tools and services
  - Greater scalability
- Multidimensional OLAP (MOLAP)
  - Sparse array-based multidimensional storage engine
  - Fast indexing to pre-computed summarized data
- Hybrid OLAP (HOLAP) (e.g., Microsoft SQLServer)
  - Flexibility, e.g., low level: relational, high-level: array
- Specialized SQL servers (e.g., Redbricks)
  - Specialized support for SQL queries over star/snowflake schemas

Mining Multi-Dimensional Association

- Single-dimensional rules:

  buys(X, "milk") $\Rightarrow$ buys(X, "bread")

- Multi-dimensional rules: $\geq$ 2 dimensions or predicates
  - Inter-dimension assoc. rules (*no repeated predicates*)

    age(X,"19-25") $\wedge$ occupation(X,"student") $\Rightarrow$ buys(X, "coke")
  - hybrid-dimension assoc. rules (*repeated predicates*)

    age(X,"19-25") $\wedge$ buys(X, "popcorn") $\Rightarrow$ buys(X, "coke")

Constraint-based (Query-Directed) Mining

- Finding all the patterns in a database autonomously? — unrealistic!
  - The patterns could be too many but not focused!
- Data mining should be an interactive process
  - User directs what to be mined using a data mining query language (or a graphical user interface)
- Constraint-based mining
  - User flexibility: provides constraints on what to be mined

Constraints in Data Mining

- Knowledge type constraint:
  - classification, association, etc.
- Data constraint
  - find product pairs sold together in stores in Chicago this year
- Dimension/level constraint
  - in relevance to region, price, brand, customer category
- Rule (or pattern) constraint
  - small sales (price < $10) triggers big sales (sum > $200)
- Interestingness constraint
  - strong rules: min_support ≥ 3%, min_confidence ≥ 60%

Meta-Rule Guided Mining

- Meta-rule can be in the rule form with partially instantiated predicates and constants

  $P_1(X, Y) \wedge P_2(X, W) \Rightarrow buys(X, \text{“iPad”})$

- The resulting rule derived can be

  $age(X, \text{“15-25”}) \wedge profession(X, \text{“student”}) \Rightarrow buys(X, \text{“iPad”})$

- In general, it can be in the form of

  $P_1 \wedge P_2 \wedge \ldots \wedge P_l \Rightarrow Q_1 \wedge Q_2 \wedge \ldots \wedge Q_r$

Challenges

- A major challenge in mining frequent itemsets from a large data set is the fact that such mining often generates a huge number of itemsets satisfying the minimum support threshold, especially when *minsup is* set low.
- This is because if an itemset is frequent, each of its subsets is frequent as well.
- A long itemset will contain a combinatorial number of shorter, frequent sub-itemsets.

Closed Patterns and Max-Patterns

- Example : A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \ldots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{1}{1}\binom{0}{0}\binom{0}{0} = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!
- Solution: *Mine closed patterns and max-patterns instead*
- An itemset X is closed if X is *frequent* and there exists *no super-pattern* Y ↄ X, *with the same support* as X
- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ↄ X
- Closed pattern is a lossless compression of freq. patterns

Closed Patterns and Max-Patterns

- Exercise. DB = {<a_1, …, a_{100}>, < a_1, …, a_{50}>}
  - Min_sup = 1.
- What is the set of closed itemset?
  - <a_1, …, a_{100}>: 1
  - < a_1, …, a_{50}>: 2
- What is the set of max-pattern?
  - <a_1, …, a_{100}>: 1

Colossal itemset

- The result of frequent closed itemset mining algorithms includes small and mid-sized itemsets, which does not enclose valuable and complete information in many applications.
- In application dealing with high dimensional datasets such as bioinformatics (Micro array analysis, biological sequence analysis) , association rule mining gives greater importance to the large sized itemsets called as colossal Itemsets
- An itemset X is called frequent colossal closed itemset if and only if it is frequent closed and card(X) ≥ mincard, where mincard is user specified least cardinality threshold

Table 1

Tid	features
1	$f_1, f_2, f_4, f_6, f_{10}$
2	f_1, f_2, f_4, f_7, f_8
3	f_2, f_4, f_7, f_8
4	$f_1, f_2, f_6, f_8, f_9, f_{10}$
5	$f_1, f_3, f_4, f_7, f_8, f_{10}$
6	f_2, f_4, f_9
7	f_5, f_7
8	f_5, f_{11}

- In Table 1, the itemset $X = \{f_2, f_4, f_7, f_8\}$, is frequent colossal closed itemset with minimum support threshold set to 2 and minimum cardinality threshold set to 4, $sup(X) \geq 2$ and $card(X) \geq 4$.

Colossal Patterns: A Motivating Example

Let's make a set of 40 transactions

T1 = 1 2 3 4 ..... 39 40
T2 = 1 2 3 4 ..... 39 40
: .
: .
: .
: .
T40=1 2 3 4 ..... 39 40

Then delete the items on the diagonal

T_1 = 2 3 4 ..... 39 40
T_2 = 1 3 4 ..... 39 40
: .
: .
: .
: .
T_{40}=1 2 3 4 39

A Show of Colossal Pattern Mining!

T_1 = 2 3 4 ..... 39 40
T_2 = 1 3 4 ..... 39 40
: .
: .
: .
: .
T_{40}=1 2 3 4 39

T_{41}= 41 42 43 79
T_{42}= 41 42 43 79
: .
: .
T_{60}= 41 42 43 ... 79

Let the min-support threshold σ= 20

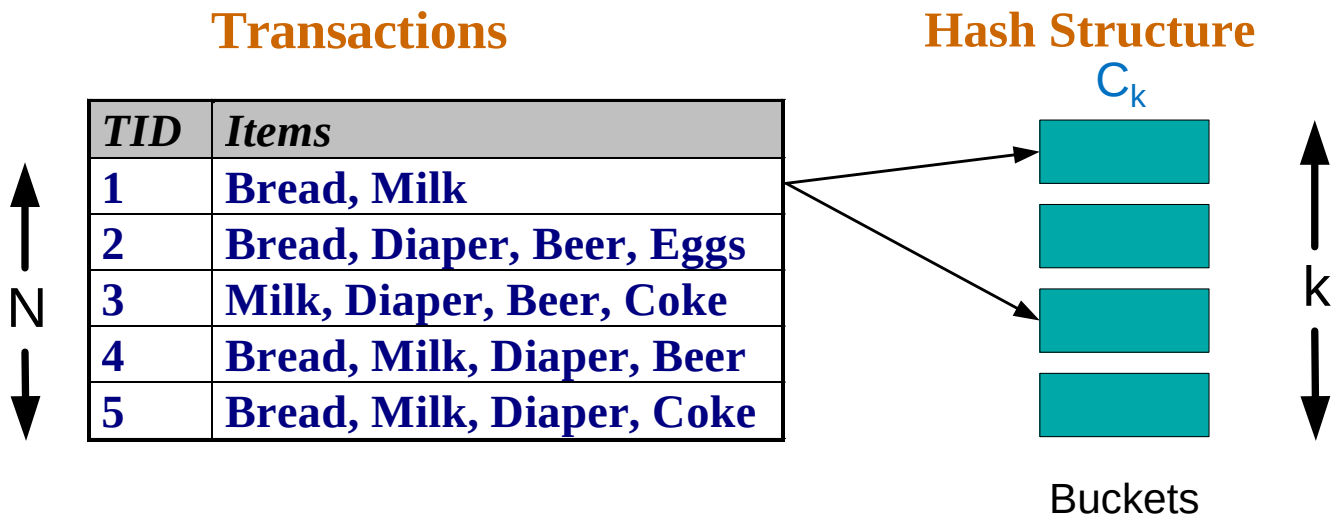
Then there are $\binom{40}{20}$ closed/maximal frequent patterns of size 20

However, there is only one colossal pattern with size greater than 20,

α= {41,42,…,79} of size 39

Computing Frequent Itemsets

- Given the set of candidate itemsets C_k, we need to compute the support and find the frequent itemsets L_k.
- Scan the data, and use a hash structure to keep a counter for each candidate itemset that appears in the data



A simple hash structure

- Create a dictionary (hash table) that stores the candidate itemsets as keys, and the number of appearances as the value.
- Increment the counter for each itemset that you see in the

Example

Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8},

{1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5},

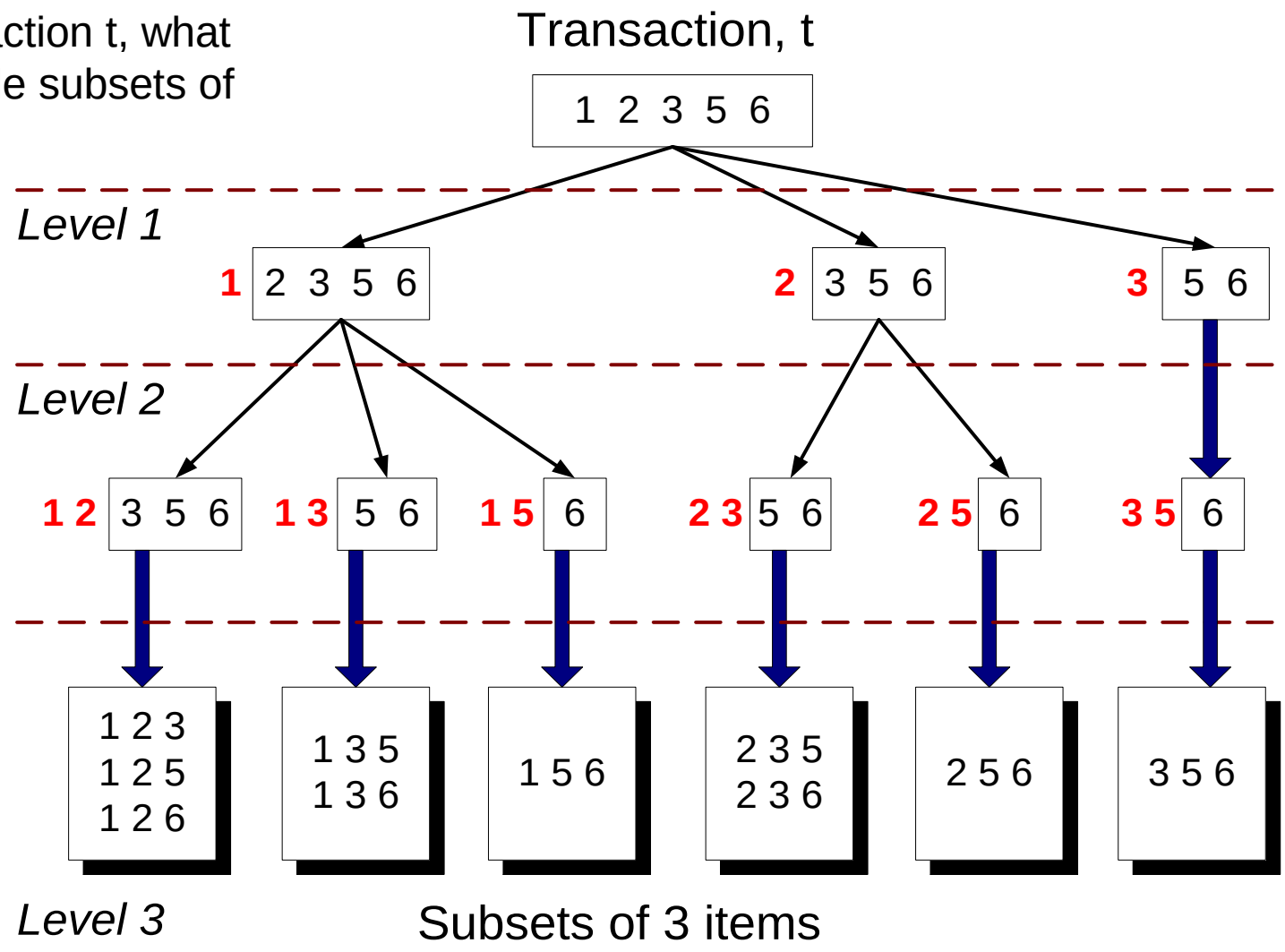
{3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

Hash table stores the counts of the candidate itemsets as they have been computed so far

Key	Value
{3 6 7}	0
{3 4 5}	1
{1 3 6}	3
{1 4 5}	5
{2 3 4}	2
{1 5 9}	1
{3 6 8}	0
{4 5 7}	2
{6 8 9}	0
{5 6 7}	3
{1 2 4}	8
{3 5 7}	1
{1 2 5}	0
{3 5 6}	1
{4 5 8}	0

Subset Generation

Given a transaction t, what are the possible subsets of size 3?



Recursion!

Example

Tuple {1,2,3,5,6} generates the following itemsets of length 3:

{1 2 3}, {1 2 5}, {1 2 6}, {1 3 5}, {1 3 6},

{1 5 6}, {2 3 5}, {2 3 6}, {3 5 6},

Increment the counters for the itemsets in the dictionary

Key	Value
{3 6 7}	0
{3 4 5}	1
{1 3 6}	3
{1 4 5}	5
{2 3 4}	2
{1 5 9}	1
{3 6 8}	0
{4 5 7}	2
{6 8 9}	0
{5 6 7}	3
{1 2 4}	8
{3 5 7}	1
{1 2 5}	0
{3 5 6}	1
{4 5 8}	0

Example

Tuple {1,2,3,5,6} generates the following itemsets of length 3:

{1 2 3}, {1 2 5}, {1 2 6}, {1 3 5}, {1 3 6},

{1 5 6}, {2 3 5}, {2 3 6}, {3 5 6},

Increment the counters for the itemsets in the dictionary

Key	Value
{3 6 7}	0
{3 4 5}	1
{1 3 6}	4
{1 4 5}	5
{2 3 4}	2
{1 5 9}	1
{3 6 8}	0
{4 5 7}	2
{6 8 9}	0
{5 6 7}	3
{1 2 4}	8
{3 5 7}	1
{1 2 5}	1
{3 5 6}	2
{4 5 8}	0

The Hash Tree Structure

Suppose you have the same 15 candidate itemsets of length 3:

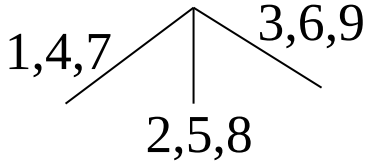
{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4},

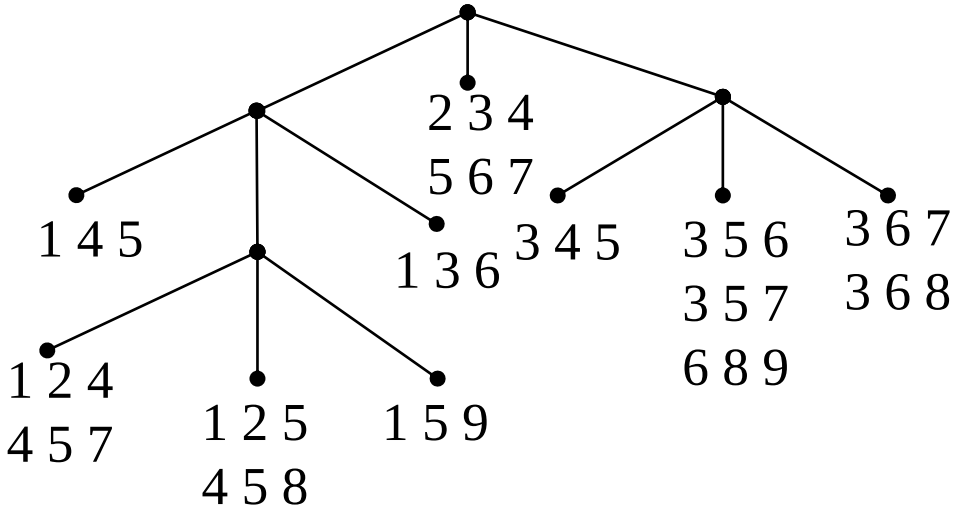
{5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

You need:

- Hash function
- Leafs: Store the itemsets

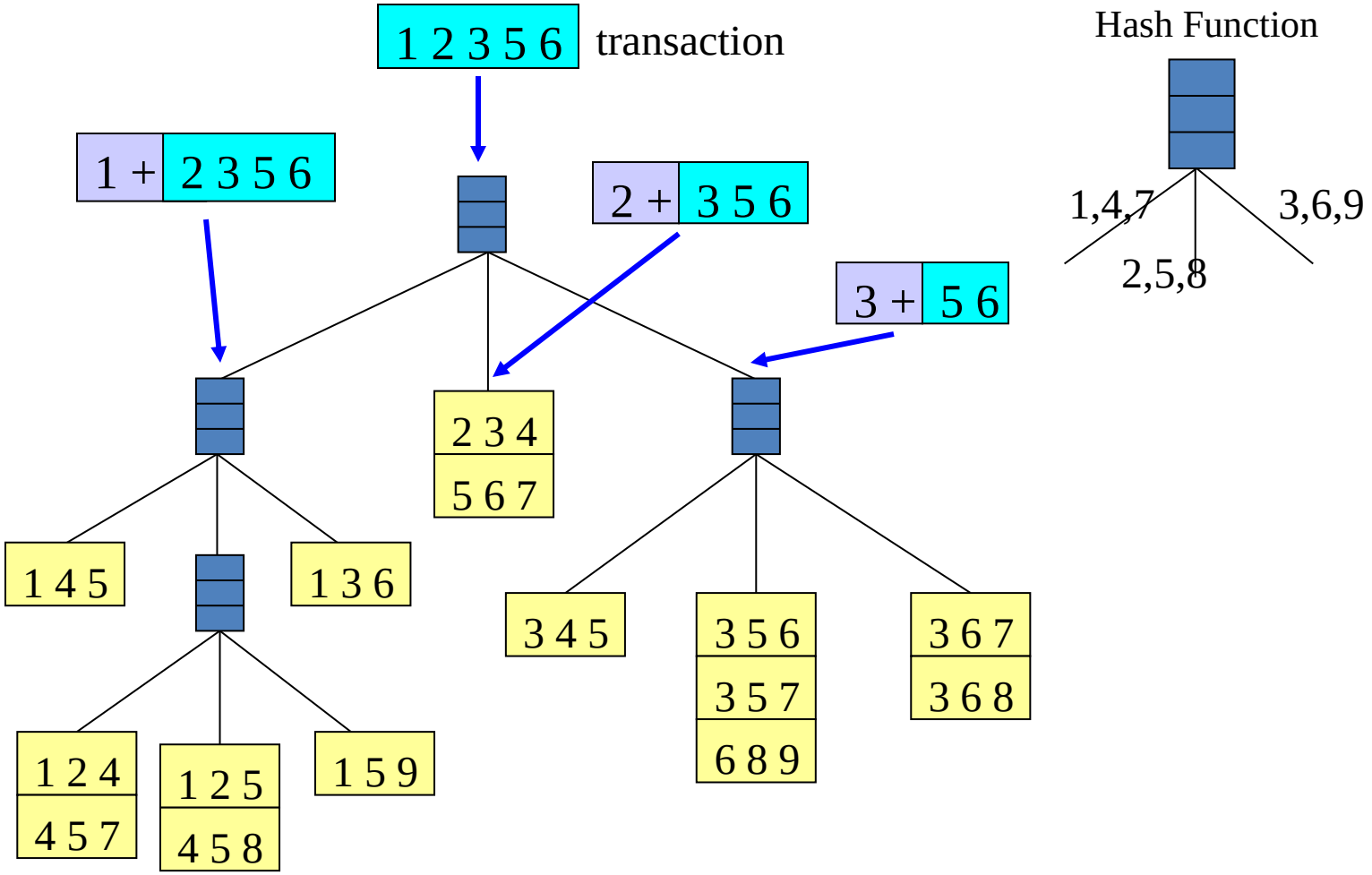
Hash function = x mod 3



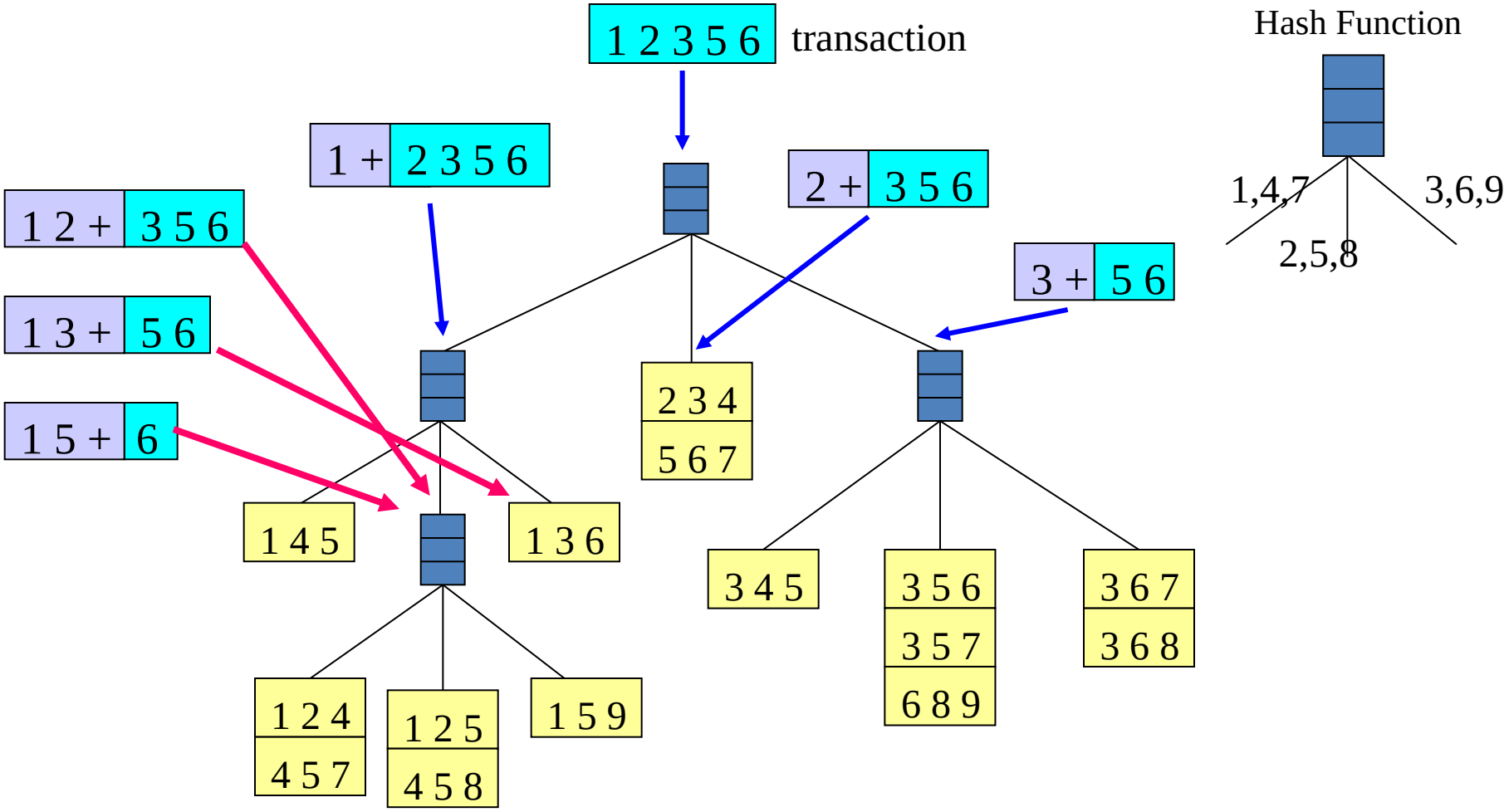


At the i-th level we hash on the i-th item

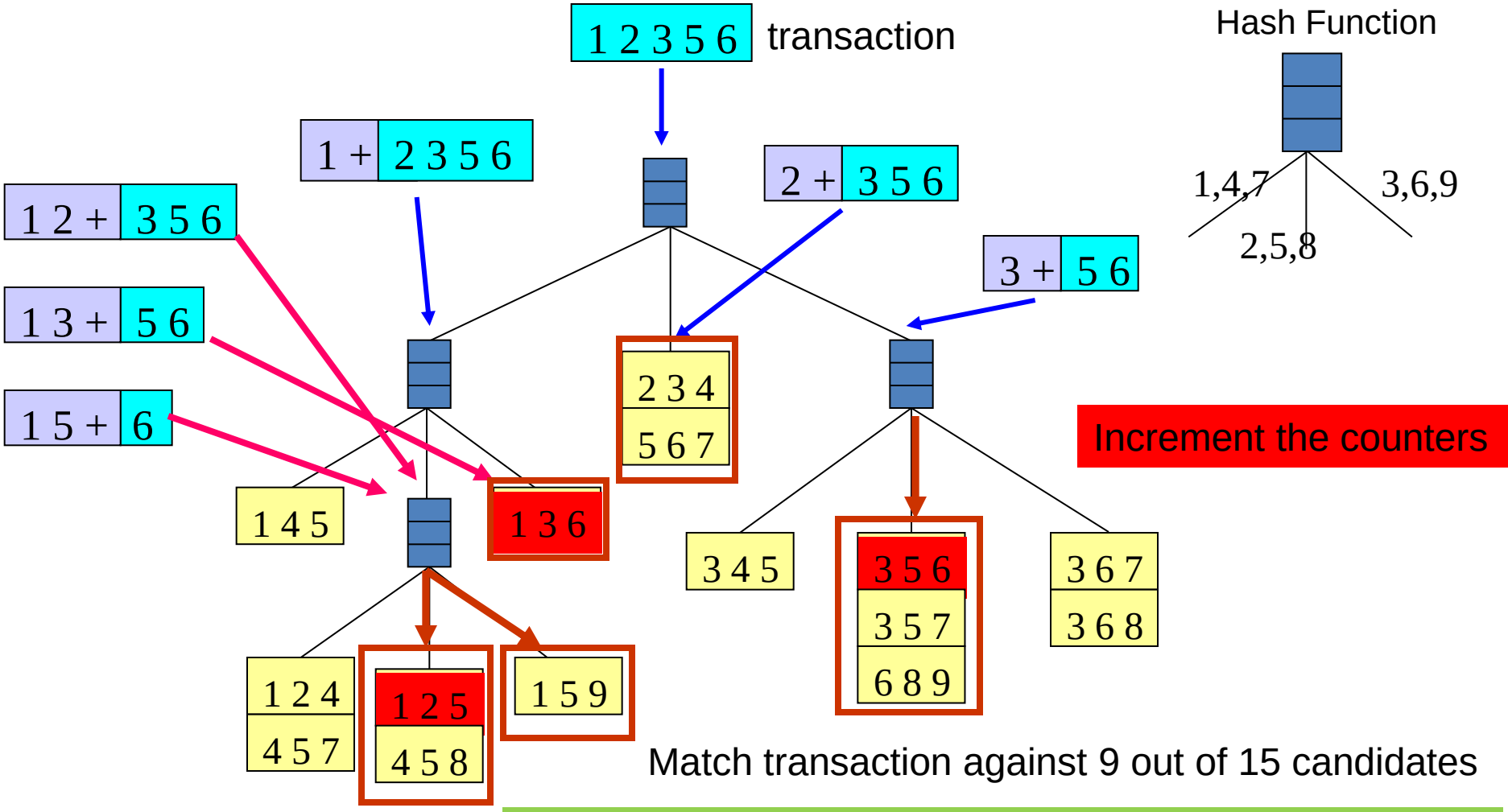
Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



Hash-tree enables to enumerate itemsets in transaction and match them against candidates

Data Objects

- Data sets are made up of data objects.
- A **data object** represents an entity.
- Examples:
  - sales database: customers, store items, sales
  - medical database: patients, treatments
  - university database: students, professors, courses
- Also called *samples , examples, instances, data points, objects, tuples*.
- Data objects are described by **attributes**.
- Database rows -> data objects; columns ->attributes.

Attributes

- **Attribute (**or **dimensions, features, variables**): a data field, representing a characteristic or feature of a data object**.**
  - *E.g., customer _ID, name, address*
- Types:
  - Nominal
  - Binary
  - Numeric: quantitative
    - Interval-scaled
    - Ratio-scaled

Attribute Types

- **Nominal:** categories, states, or "names of things"
  - *Hair_color* = {*auburn, black, blond, brown, grey, red, white*}
  - marital status, occupation, ID numbers, zip codes
- **Binary**
  - Nominal attribute with only 2 states (0 and 1)
  - Symmetric binary: both outcomes equally important
    - e.g., gender
  - Asymmetric binary: outcomes not equally important.
    - e.g., medical test (positive vs. negative)
    - Convention: assign 1 to most important outcome (e.g., HIV positive)
- **Ordinal**
  - Values have a meaningful order (ranking) but magnitude between successive values is not known.
  - *Size* = {*small, medium, large*}, grades, army rankings

Numeric Attribute Types

- Quantity (integer or real-valued)
- **Interval**
  - Measured on a scale of **equal-sized units**
  - Values have order
    - E.g., *temperature in C˚or F˚, calendar dates*
- **Ratio**
  - Inherent **zero-point**
  - We can speak of values as being an order of magnitude larger than the unit of measurement (10 K˚ is twice as high as 5 K˚).
    - e.g., *temperature in Kelvin, length, counts*

Discrete vs. Continuous Attributes

- **Discrete Attribute**
  - Has only a finite or countably infinite set of values
    - E.g., zip codes, profession, or the set of words in a collection of documents
  - Sometimes, represented as integer variables
  - Note: Binary attributes are a special case of discrete attributes
- **Continuous Attribute**
  - Has real numbers as attribute values
    - E.g., temperature, height, or weight
  - Practically, real values can only be measured and represented using a finite number of digits
  - Continuous attributes are typically represented as floating-point variables

Similarity and Dissimilarity

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

Data Matrix and Dissimilarity Matrix

- Data matrix
  - n data points with p dimensions

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

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

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

Proximity Measure for Nominal Attributes

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

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

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

Proximity Measure for Binary Attributes

- A contingency table for binary data

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

- Distance measure for symmetric binary variables:

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

- Distance measure for asymmetric binary variables:

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

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

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

Dissimilarity between Binary Variables

- Example

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

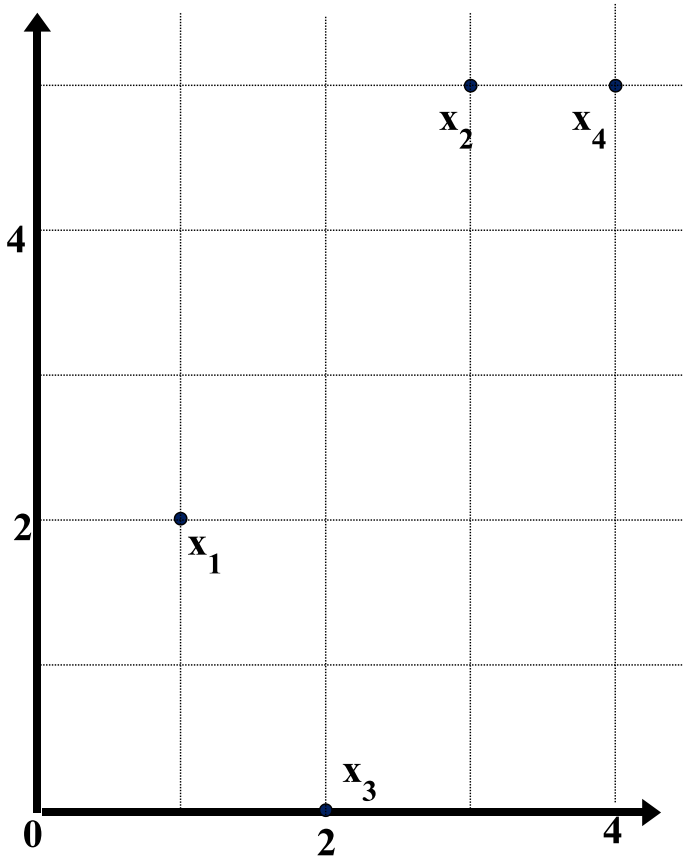
  - Gender is a symmetric attribute
  - The remaining attributes are asymmetric binary
  - Let the values Y and P be 1, and the value N 0

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

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

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

Example:
Data Matrix and Dissimilarity Matrix



Data Matrix

point	attribute1	attribute2
x1	1	2
x2	3	5
x3	2	0
x4	4	5

Dissimilarity Matrix

(with Euclidean Distance)

L2	x1	x2	x3	x4
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

Distance on Numeric Data: Minkowski Distance

- *Minkowski distance*: A popular distance measure

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

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and h is the order (the distance so defined is also called L-h norm)

Special Cases of Minkowski Distance

- h = 1: Manhattan (city block, L_1 norm) distance
  - E.g., the Hamming distance: the number of bits that are different between two binary vectors

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + \ldots + |x_{i_p} - x_{j_p}|$$

- h = 2: (L_2 norm) Euclidean distance

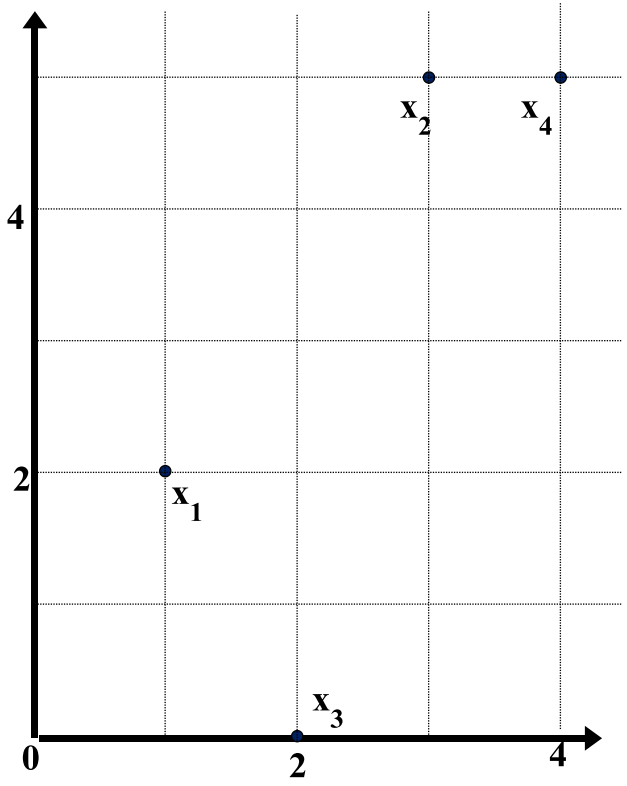
$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + \ldots + |x_{i_p} - x_{j_p}|^2)}$$

- $h \to \infty$. "supremum" (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component (attribute) of the vectors

$$d(i,j) = \lim_{h \to \infty} \left(\sum_{f=1}^{p} |x_{if} - x_{jf}|^h \right)^{\frac{1}{h}} = \max_{f}^{p} |x_{if} - x_{jf}|$$

Example: Minkowski Distance

point	attribute 1	attribute 2
x1	1	2
x2	3	5
x3	2	0
x4	4	5



Dissimilarity Matrices

Manhattan (L_1)

L	x1	x2	x3	x4
x1	0			
x2	5	0		
x3	3	6	0	
x4	6	1	7	0

Euclidean (L_2)

L2	x1	x2	x3	x4
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

Supremum

L_∞	x1	x2	x3	x4
x1	0			
x2	3	0		
x3	2	5	0	
x4	3	1	5	0

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{if} by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - map the range of each variable onto [0, 1] by replacing i-th object in the f-th variable by

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

  - compute the dissimilarity using methods for interval-scaled variables

Cosine Similarity

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

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

- Other vector objects: gene features in micro-arrays, …
- Applications: information retrieval, biologic taxonomy, gene feature mapping, ...
- Cosine measure: If d_1 and d_2 are two vectors (e.g., term-frequency vectors), then

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2|| ,$$

where $\bullet$ indicates vector dot product, $||d||$: the length of vector d

Example: Cosine Similarity

- $\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||$,
  where $\bullet$ indicates vector dot product, $||d||$: the length of vector d

- Ex: Find the **similarity** between documents 1 and 2.

$d_1 =$ (5, 0, 3, 0, 2, 0, 0, 2, 0, 0)
$d_2 =$ (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)

$d_1 \bullet d_2 = 5*3+0*0+3*2+0*0+2*1+0*1+0*1+2*1+0*0+0*1 = 25$
$||d_1|| = (5*5+0*0+3*3+0*0+2*2+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$
$||d_2|| = (3*3+0*0+2*2+0*0+1*1+1*1+0*0+1*1+0*0+1*1)^{0.5} = (17)^{0.5} = 4.12$
$\cos(d_1, d_2) = 0.94$

Basic Statistical Descriptions of Data

- Motivation
  - To better understand the data: central tendency, variation and spread
- Data dispersion characteristics
  - median, max, min, quantiles, outliers, variance, etc.
- Numerical dimensions correspond to sorted intervals
  - Data dispersion: analyzed with multiple granularities of precision
  - Boxplot or quantile analysis on sorted intervals

Measuring the Central Tendency

- Mean (algebraic measure) (sample vs. population):

  $$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \mu = \frac{\sum x}{N}$$

  Note: *n* is sample size and *N* is population size.

  - Weighted arithmetic mean:

  $$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

- Median:
  - Middle value if odd number of values, or average of the middle two values otherwise
  - Estimated by interpolation (for *grouped data*):

  $$median = L_1 + (\frac{n/2 - (\sum freq)l}{freq_{median}})width$$

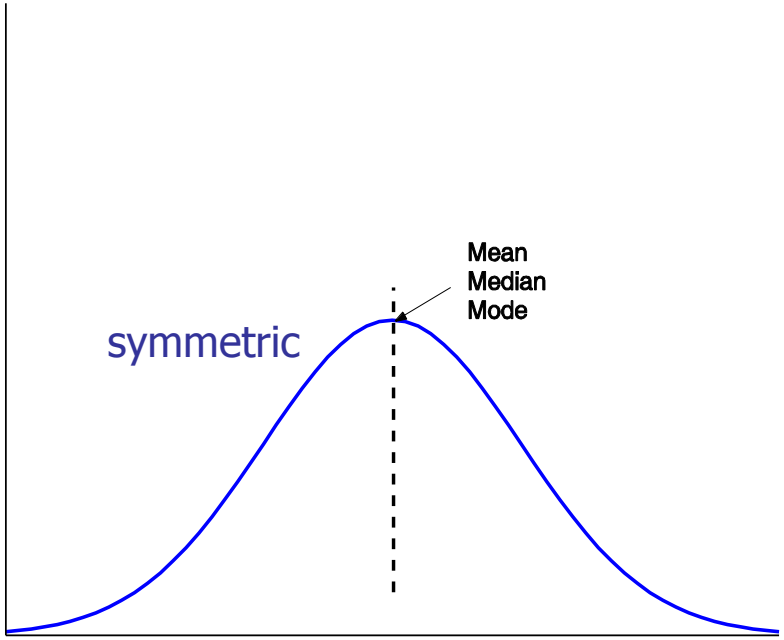
age	*frequency*
1–5	200
6–15	450
16–20	300
21–50	1500
51–80	700
81–110	44

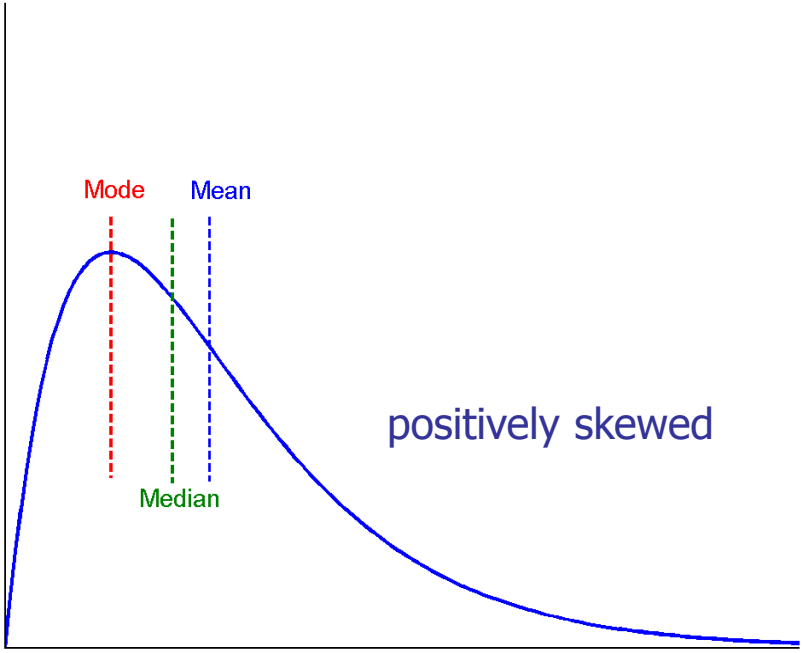
- Mode
  - Value that occurs most frequently in the data
  - Unimodal, bimodal, trimodal
  - Empirical formula:

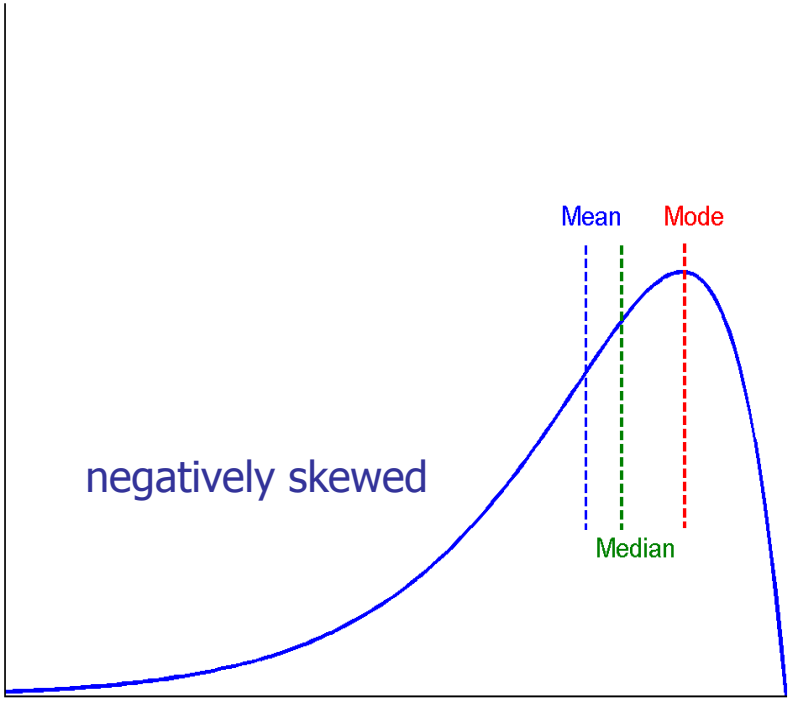
  $$mean - mode = 3 \times (mean - median)$$

Symmetric vs. Skewed Da

- Median, mean and mode of symmetric, positively and negatively skewed data







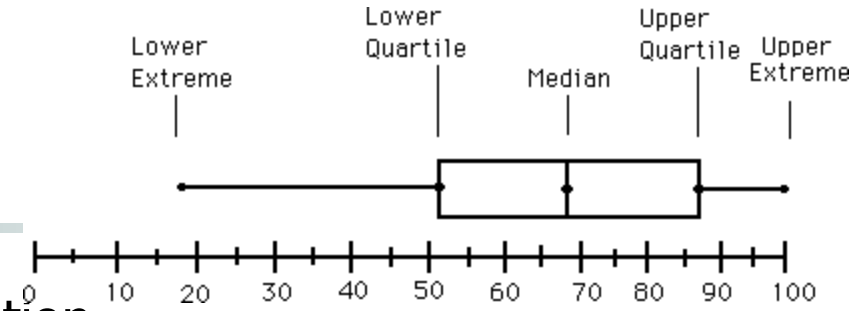
Measuring the Dispersion of Data

- Quartiles, outliers and boxplots
  - **Quartiles**: Q_1 (25th percentile), Q_3 (75th percentile)
  - **Inter-quartile range**: IQR = $Q_3 - Q_1$
  - **Five number summary**: min, Q_1, median, Q_3, max
  - **Boxplot**: ends of the box are the quartiles; median is marked; add whiskers, and plot outliers individually
  - **Outlier**: usually, a value higher/lower than 1.5 x IQR
- Variance and standard deviation (*sample: s, population: σ)*
  - **Variance**: (algebraic, scalable computation)

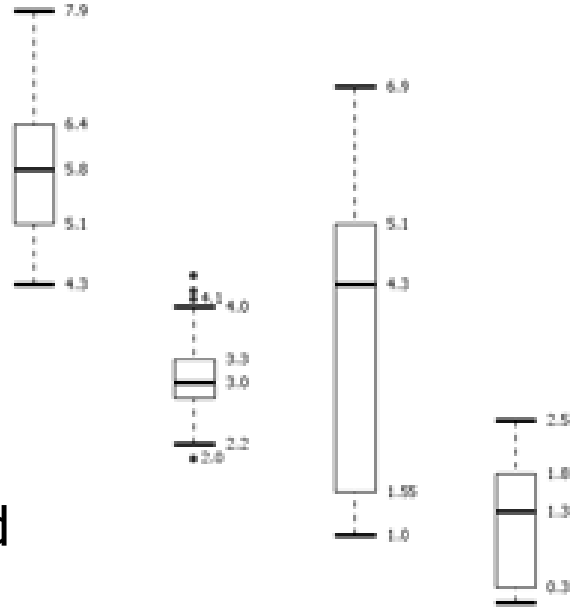
$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n-1}[\sum_{i=1}^{n}x_i^2 - \frac{1}{n}(\sum_{i=1}^{n}x_i)^2] \qquad \sigma^2 = \frac{1}{N}\sum_{i=1}^{n}(x_i - \mu)^2 = \frac{1}{N}\sum_{i=1}^{n}x_i^2 - \mu^2$$

  - **Standard deviation** *s (or σ)* is the square root of variance s^2 *(or* σ^2*)*

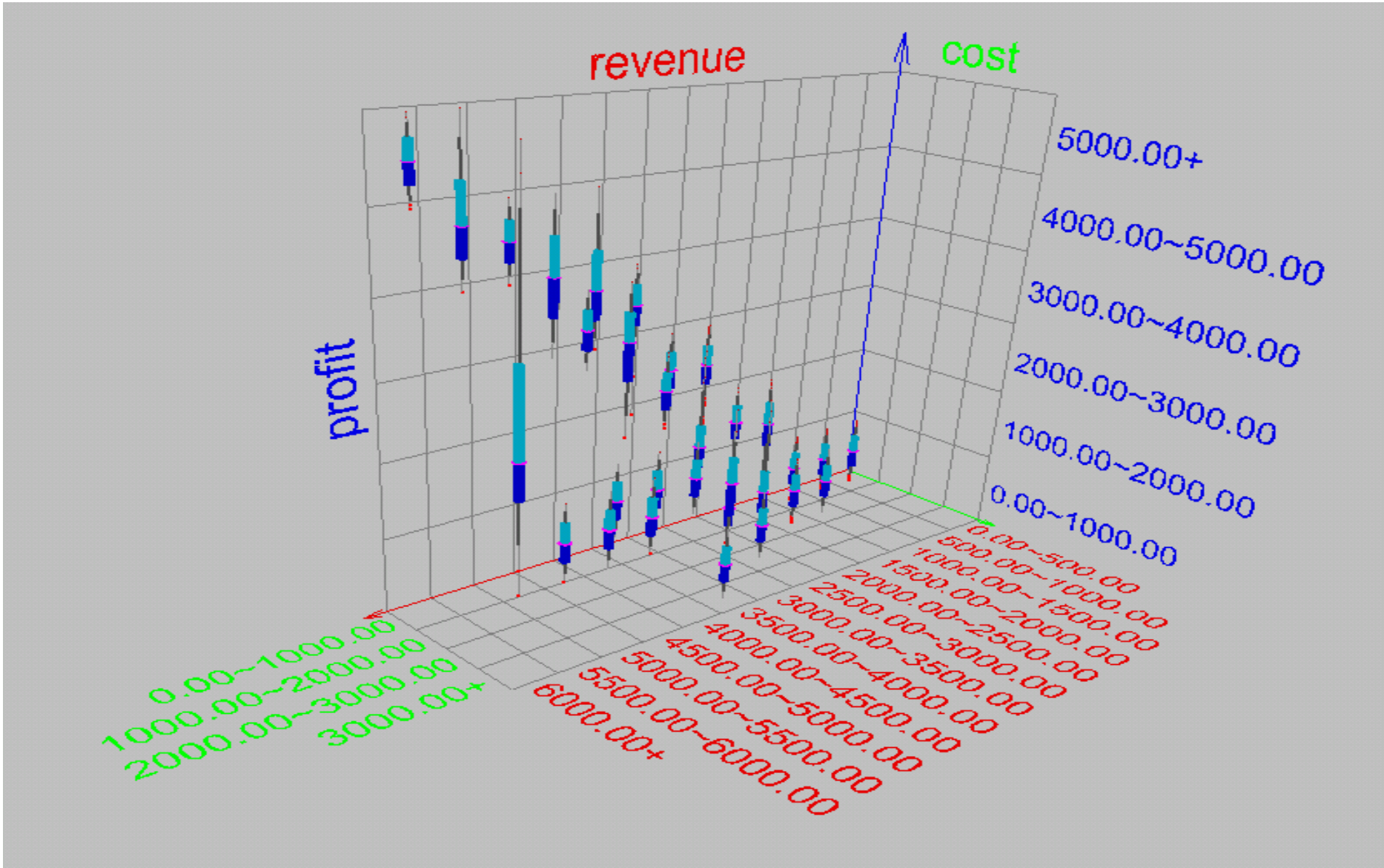
Boxplot Analysis



- **Five-number summary** of a distribution
  - Minimum, Q1, Median, Q3, Maximum
- **Boxplot**
  - Data is represented with a box
  - The ends of the box are at the first and third quartiles, i.e., the height of the box is IQR
  - The median is marked by a line within the box
  - Whiskers: two lines outside the box extended to Minimum and Maximum
  - Outliers: points beyond a specified outlier threshold, plotted individually

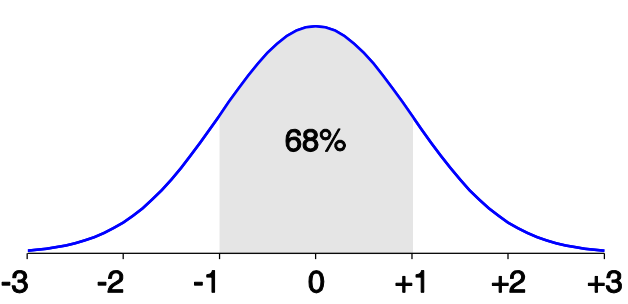


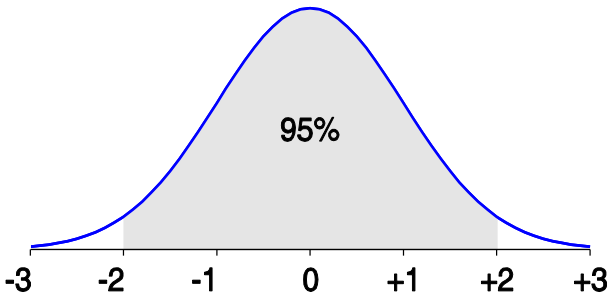
Visualization of Data Dispersion: 3-D Boxplots

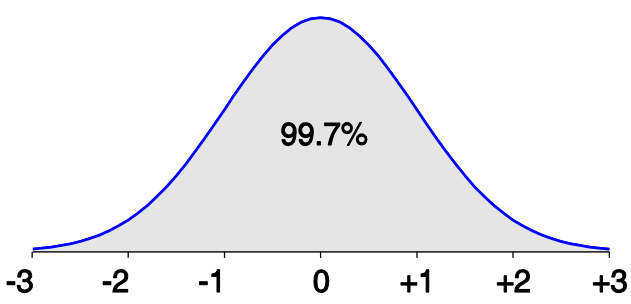


Properties of Normal Distribution Curve

- The normal (distribution) curve
  - From μ–σ to μ+σ: contains about 68% of the measurements (μ: mean, σ: standard deviation)
  - From μ–2σ to μ+2σ: contains about 95% of it
  - From μ–3σ to μ+3σ: contains about 99.7% of it





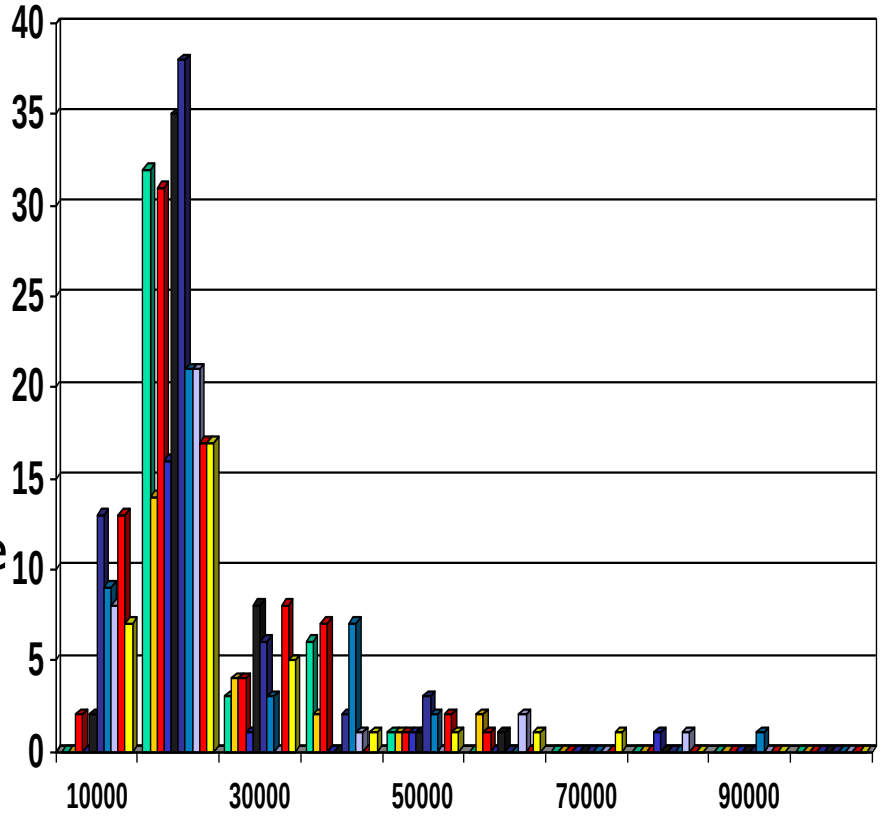


Graphic Displays of Basic Statistical Descriptions

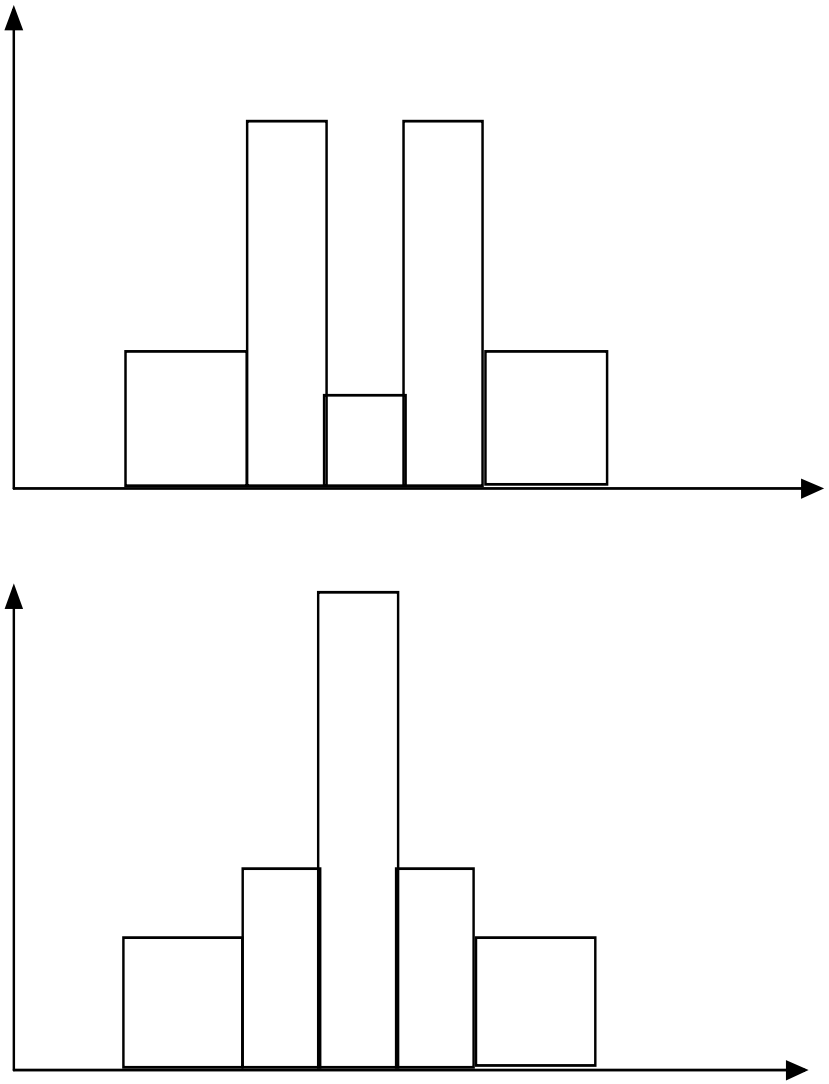
- **Boxplot**: graphic display of five-number summary
- **Histogram**: x-axis are values, y-axis repres. frequencies
- **Quantile plot**: each value x_i is paired with f_i indicating that approximately 100 f_i% of data are $\leq x_i$
- **Quantile-quantile (q-q) plot**: graphs the quantiles of one univariant distribution against the corresponding quantiles of another
- **Scatter plot**: each pair of values is a pair of coordinates and plotted as points in the plane

Histogram Analysis

- Histogram: Graph display of tabulated frequencies, shown as bars
- It shows what proportion of cases fall into each of several categories
- Differs from a bar chart in that it is the *area* of the bar that denotes the value, not the height as in bar charts, a crucial distinction when the categories are not of uniform width
- The categories are usually specified as non-overlapping intervals of some variable. The categories (bars) must be adjacent



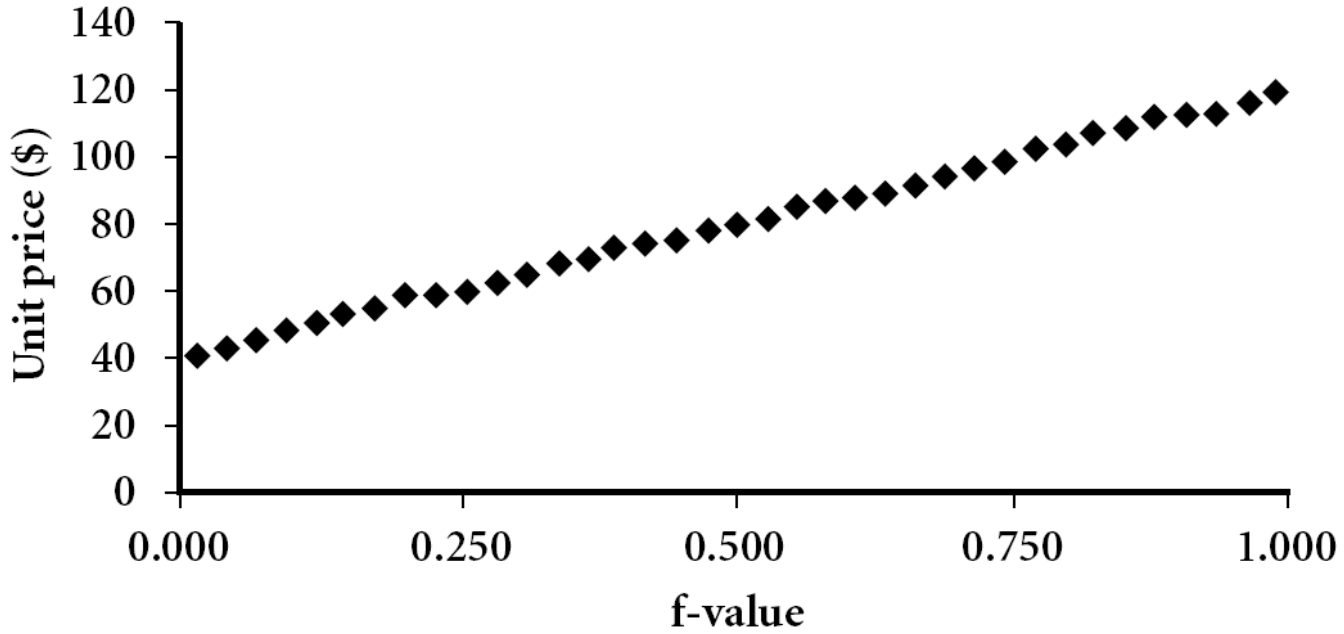
Histograms Often Tell More than Boxplots



- The two histograms shown in the left may have the same boxplot representation
  - The same values for: min, Q1, median, Q3, max
- But they have rather different data distributions

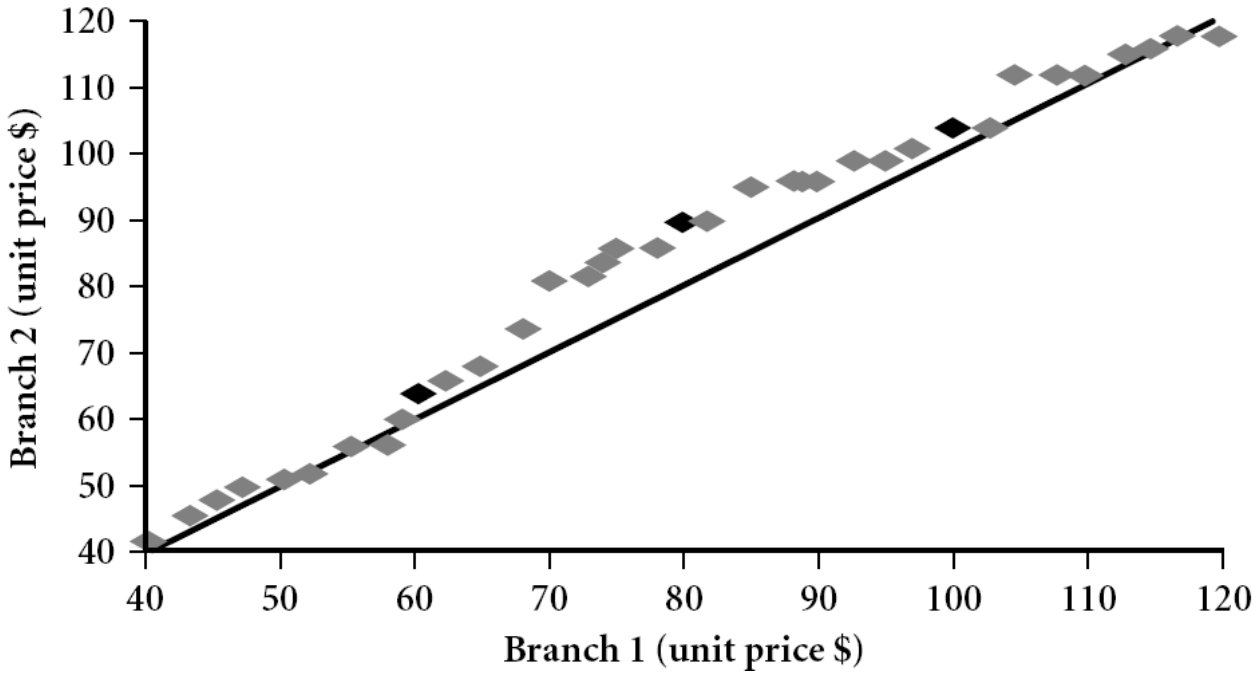
Quantile Plot

- Displays all of the data (allowing the user to assess both the overall behavior and unusual occurrences)
- Plots **quantile** information
  - For a data x_i data sorted in increasing order, f_i indicates that approximately 100 f_i% of the data are below or equal to the value x_i



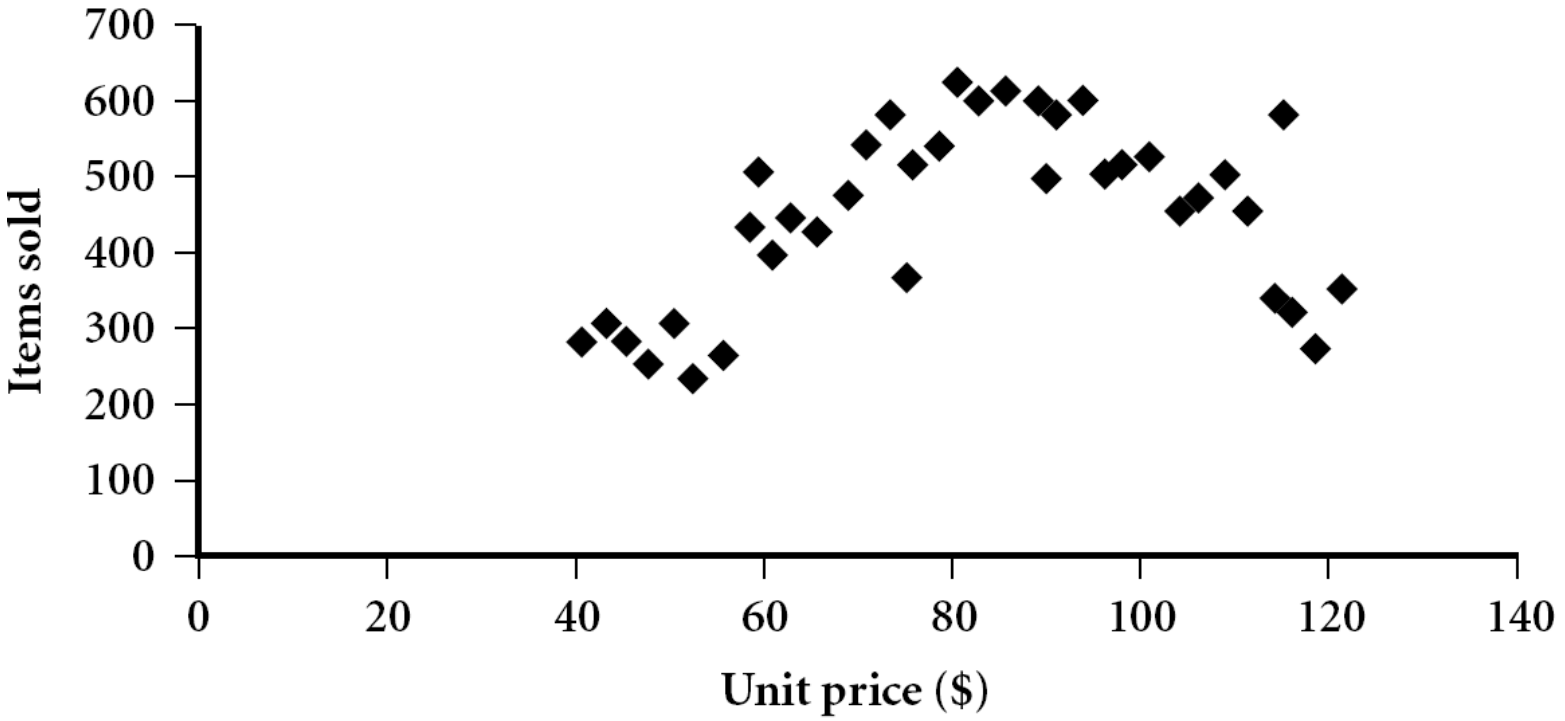
Quantile-Quantile (Q-Q) Plot

- Graphs the quantiles of one univariate distribution against the corresponding quantiles of another
- View: Is there is a shift in going from one distribution to another?
- Example shows unit price of items sold at Branch 1 vs. Branch 2 for each quantile. Unit prices of items sold at Branch 1 tend to be lower than those at Branch 2.

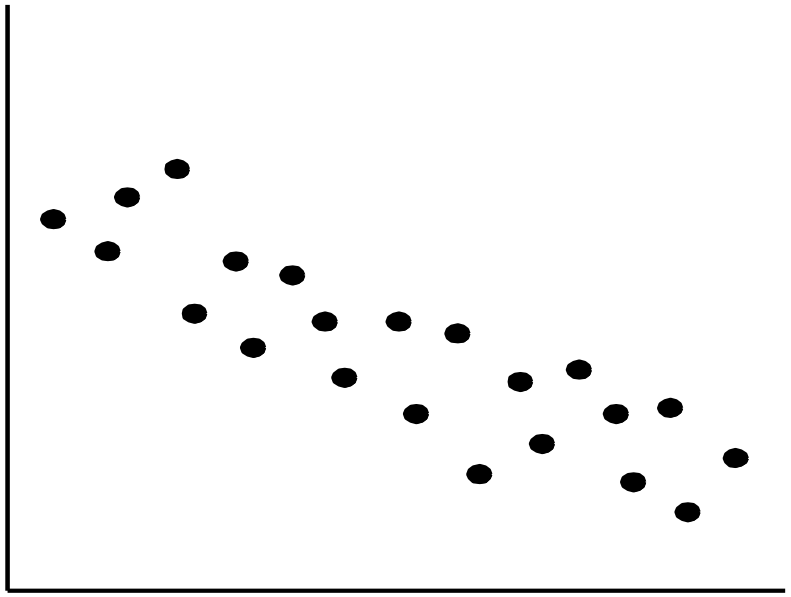


Scatter plot

- Provides a first look at bivariate data to see clusters of points, outliers, etc
- Each pair of values is treated as a pair of coordinates and plotted as points in the plane

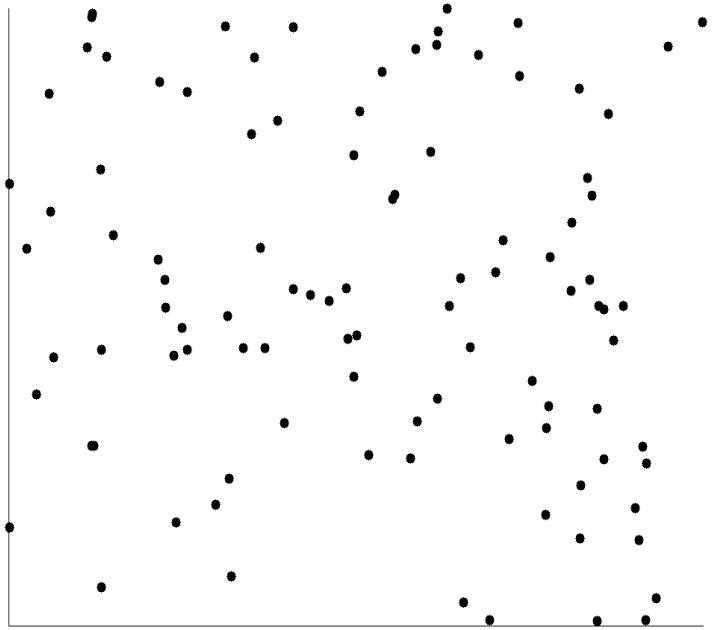


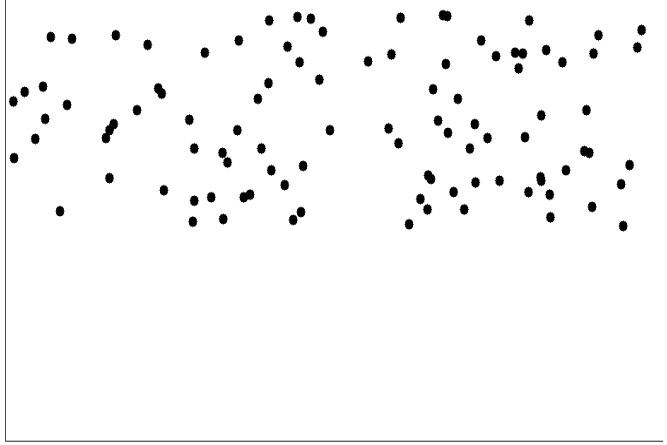
Positively and Negatively Correlated Data

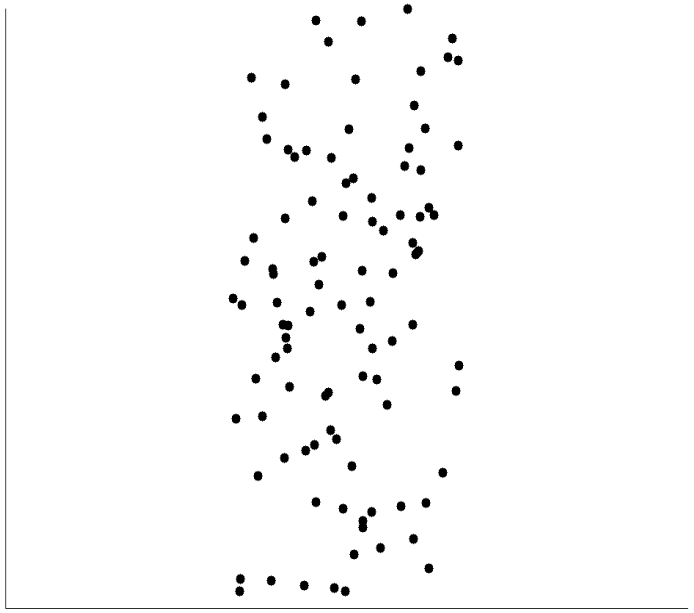


- The left half fragment is positively correlated
- The right half is negative correlated

Uncorrelated Data







DWDM ..

Data integration: Redundancy issues: An attr might be redundant if it is derived from another attribute

r_{AB} → Correlation measure b/w attr. A & B.

$$r_{AB} = \sum_{i=1}^{n} \frac{(A_i - \bar{A})(B_i - \bar{B})}{n \sigma_A \sigma_B} \quad \text{(for numeric data)}$$

$\bar{A}$ → Mean σ_A → Std. dev. n → No. of tuples

If $r_{AB} > 0$ → A, B are +vely dependent $A \not\perp B$
$r_{AB} = 0$ → No correlation $A \perp B$
$r_{AB} < 0$ → A, B are -vely correlated $A \not\perp B$

For categoric data:
Chi Sq. (χ^2)
(Pearson correlation)

Eg:

	M	F
Magazine	250	200
Newspaper	50	1000

Let A have C distinct values $a_1, a_2 \ldots a_c$
B have r distinct values $b_1, b_2 \ldots b_r$

For (A_i, B_j)

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

O_{ij} → Actual count (Observed freq)
E_{ij} → Expected freq.

$$E_{ij} = \frac{\text{count}(A = a_i) \times \text{Count}(B = b_j)}{N}$$

$$\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{c}$$

	M	F	
Mag.	250 (90)	200	450
News	50	1000	1050
	300	1200	1500

$$E_{ij} = \frac{C_{Male} \times C_{Mag.}}{1500} = \frac{300 \times 450}{(1200+300)} = 90$$

male, mag.

$$E_{Male,News} = \frac{300 \times 1050}{1500} = 210$$

$$E_{F,Mag} = \frac{450 \times 1200}{1500} = 360$$

$$E_{F,N} = \frac{1200 \times 1050}{1500} = 840$$

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

$$= 4 \times \frac{(160)^2}{90} = 4 \times \frac{25600}{90}$$

From stat. table, for 2×2 matrix,

$\chi^2 = 10.828$ (r−1)(c−1) deg. of freedom

$= 10.828$

Calculated value (χ^2) ≥ Table value (χ^2)

↳ Correlated values

∴ Gender & Reading materials are correlated.

Feature selection

→ Choose optimal subset of features
  → Remove irrelevant features.
  → Inc. pred. accuracy
  → Improve learning efficiency → Reduce storage space
                                  ↳ " computation cost

Searching best subset of features
Criteria for evaluating different ~~one~~ subsets.

Selection of best subset:
Heuristic methods:
→ stepwise forward selection
  {A1, A2 . . . . An}
  Start with {} add best attr {A best}, add next best {A best+1} . . .

→ Stepwise backward elimination:
Start with all, keep removing worst attr.

→ Combination of forward selection & backward elimination: At each step, select a best attr. & eliminate a worst attr.

Criteria for best attr. split:
Gain measure, gini measure
Attributes that do not occur on generating Decision tree are considered irrelevant.

(Wrapper based)

Genetic algorithm: → Maximise accuracy measure
Init. chromosome.
  10110101
  ↓ ↓ absent
  feature present

Evaluate chromosom with fitness function
↓
fitness fn


Selection, crossover, mutation
↓
Genetic operations
Check stopping criteria.
↳ Max iter,
↳ Converged test fn
subset of

Criteria to evaluate features:

① Entropy measure $= \sum_{i=1}^{c} P(C_i) \log_2(C_i)$

c → no. of class label.

Information gain

$$I_f(A) = -\sum_{j=1}^{p} \frac{|D_j|}{|D|} I(D_j)$$

$|D|$ → no. of types in partition j.
$|D_j|$ → No. of tuply in D_j

Distance measure:

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

↳ Depending on p, can be manhattan dist, euclidian dist, etc

Dependency measure / correlation measure.

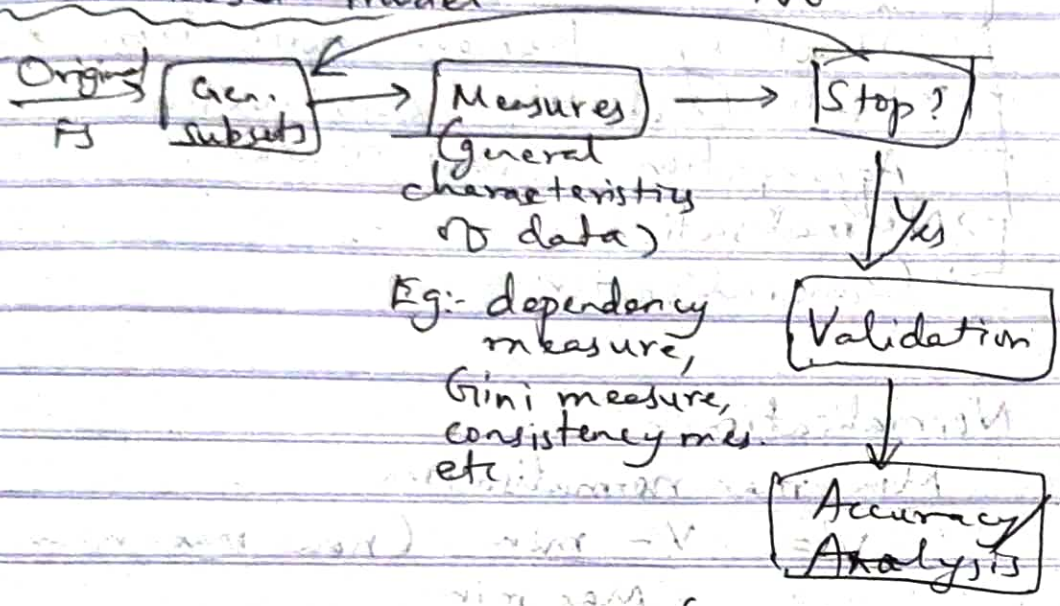
$$r(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n \sigma_A \sigma_B}$$

Consistency measure:
If input feature values are same
↓
Same class label (o/p)

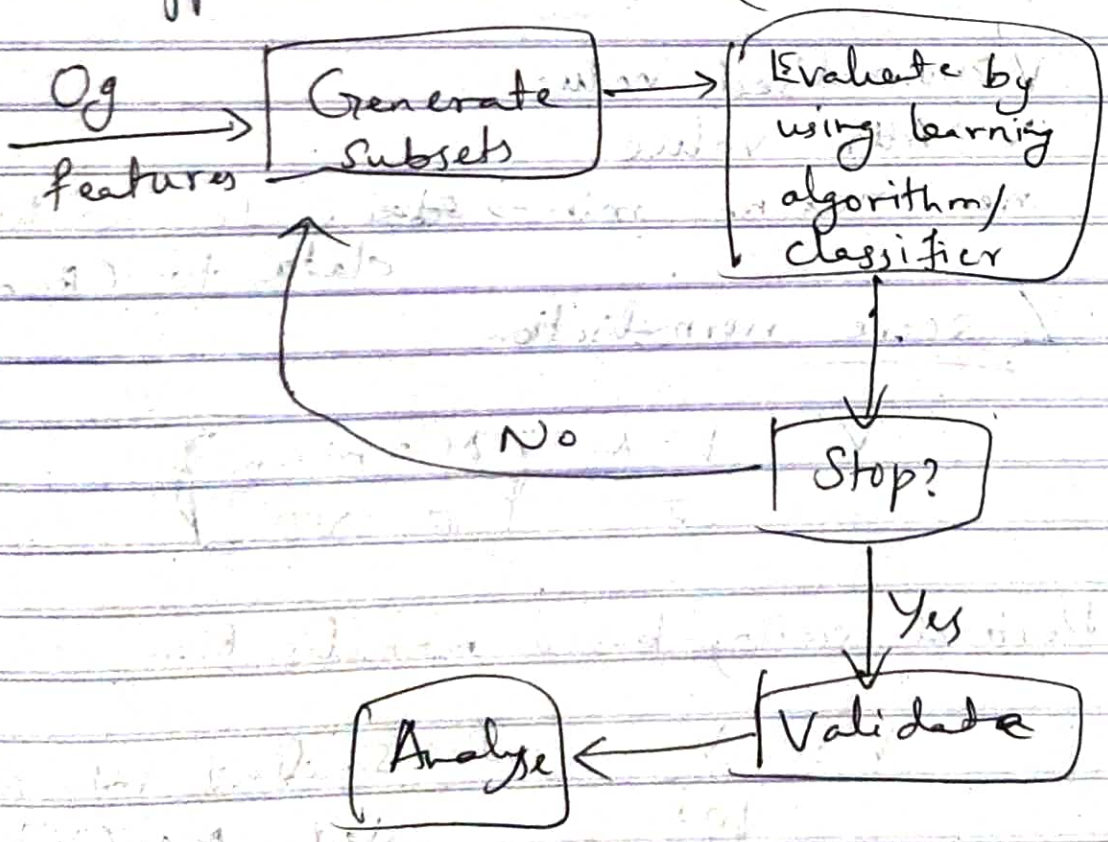
Accuracy measure:
Evaluate all possible subsets from accuracy method subset with highest acc. is considered.

Models of FS:

Filter based model (Faster)



Wrapper based model: (Slower)



Usually better results, longer computation energy & time.

Data transformation.

A fn that maps the entire data set of values of a given attribute to a new set.

Methods

- Smoothing → Remove noise from data
- Feature ~~selection~~ construction
- Aggregation → Summarisation
- Normalisation
- Generalisation

Normalisation :-

Min max normalisation

$$V' = \frac{V - \min}{\text{Max} - \min} (\text{new max} - \text{new min}) + \text{new min}$$

V' → normalised value

V → Attr. value

new max, new min → ~~start~~ scales to fit data to (Range).

Z score normalisation

$$V' = \frac{V - \mu}{\sigma}$$

μ → mean

σ → SD

Decimal scaling based normalisation.

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

j → Smallest int. such that $\text{Max}(V') < 1$

Ex:- 986 → 9.17. j is 1000 → 3

0.917 → 0.986

Principal Component Analysis

Dimensionality reduction:

Steps:

- Collect data
- Subtract mean from data
- Calc covariance matrix
- Calc eigenvalues & eigenvectors of the covariance matrix
- Order eigenvectors by eigenvalues (high to low)
- Select 'p' eigenvectors
- Derive new data.

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9
~~$x - \bar{x}$~~	~~$y - \bar{y}$~~

$\bar{x}$ = 1.81 $\bar{y}$ = 1.91

$x - \bar{x}$ = 0.69, −1.31, 0.39, 0.09, 1.29, 0.49, 0.19, −0.81, −0.31, −0.71

$y - \bar{y}$ = 0.49, −1.21, 0.99, 0.29, 1.09, 0.79, −0.31, −0.81, −0.31, −1.01

(3) Covariance matrix (2×2)

$$cov(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{(n-1)}$$

$$= \begin{bmatrix} cov(x,x) & cov(x,y) \\ cov(y,x) & cov(y,y) \end{bmatrix}$$

$$= \begin{bmatrix} 0.616155 & 0.615444 \\ 0.615444 & 0.71655 \end{bmatrix}$$

Eigenvalues = $\begin{pmatrix} 0.049083 \\ 1.284602771 \end{pmatrix}$

Eigen vectors = $\begin{pmatrix} -0.7351 & -0.6778 \\ 0.6778 & -0.7351 \end{pmatrix}$

Order the vectors:

$\begin{bmatrix} -0.6778 & -0.7357 \\ -0.7351 & 0.6778 \end{bmatrix}$

↓ Select

Derive new data

$(-0.6778 \times 0.69) + (-0.7351 \times 0.49)$ so on

Modified $\underset{(P(1))}{X}$:- $-0.827, 1.7, -0.992, -0.274, -1.6, 0.91, 0.099, 1.14, 0.43, 1.22$

Clustering → Partition based (Simple but sensitive to noise) Every obj ∈ 1 clu
→ Heirarchical based (No need to kn; $O(n^2)$)
(agglomerative & divisive) [dendogram]

$O(n^{k+t})$ → clusters, → iterations

n → no. of data points

FCM :- Fuzzy C means; Object ∈ more than 1 clust with membership matrix

Density based clustering

↳ No. of points in given ~~no. of~~ radius

DBScan (Density based) :-

P is a core point if with ε radius, there are given min. number of points around it

A point is border point if a point is not core point but lies within the neighbourhood of a core point

A noise point is a point that is neither a core nor a border point. These points are to be discarded.

Adv: Not sensitive to noise; Any shape of cluster is possible

Disadv:- Picking parameter values (ε & min val.)

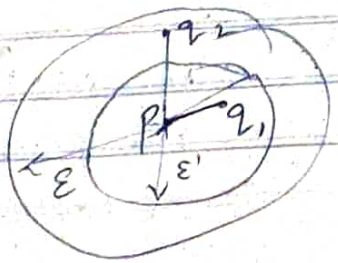
OPTICS: Ordering Points To Identify Clustering Structure

Need to identify core dist, reachability dist along with core, border point

In fig 2, $\varepsilon' < \varepsilon$ & point is core pt even with ε'.

Core dist. of an objective 'p' is the smallest ε' value that makes 'p' as a core point

Reachability dist of an object 'q' w.r.t another object 'p' is the greater value of the core dist of 'p' & euclidian dist of (p↔q).



Reachability dist$(p, q_1) = \varepsilon'$

Reachability $(p, q_2) = \max(\varepsilon', p_{q_2})$

= euclidian p, q_2

SRIRAM

Optics → It stores core dist & reachability dist. for each obj.

→ Maintains a list called seedlist → objects are sorted by reach. dist from their resp. closest core points.

? ? ?

CLIQUE (Clustering in Quest)

A dimension growth subspace clustering (first of its kind)

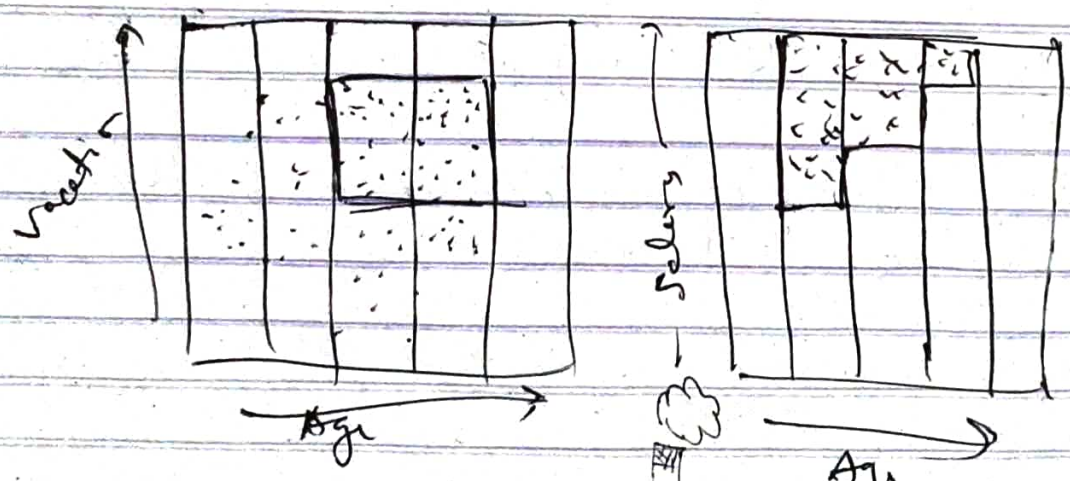
Starts with a single dimension, divides this dim into grids.

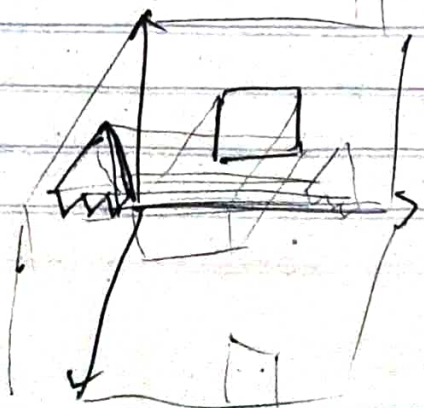
Identifies dense units in grids (no. of min pts)

The subspace representing these dense units are intersected to form candidate search for higher dim.

Get maximal & minimal region

Highly dense units — Low dense region



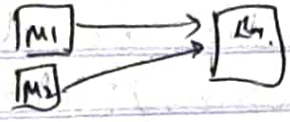


Grid size??
& min points to be chosen

FREERAM SHEERAM

Classifiers
- Naive bayes
- Decision tree
- KNN
- Artif. Neural Net
  - Backprop.
  - Perceptron

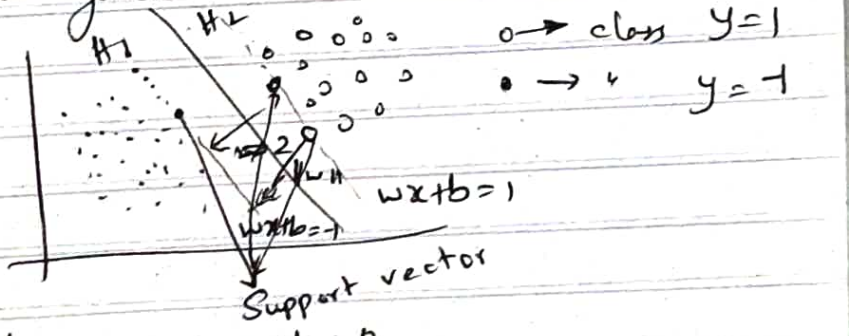
Ensemble classifiers
- Set of classifiers
- (Learning is based on comb. of multiple classifiers)



Bagging → Each classifier returns class pred. for X which counts as a vote. Democracy

Boosting → Assign wt to each classifier based on how well it performs.
Sum weights of each classifier that assigned class C. Class with highest wt. wins.

SVM (Support Vector Machine)
→ Max. margin hyperplane (MMH) that separates the given classes.



Any hyperplane $wx+b=0$
w is normal vector to the hyperplane $w_1, w_2, \ldots w_n$
n is no. of attrs
x is the training tuple
b is the bias

Need to maximise dist. b/w H_1, H_2
$dist(H_1, H_2) = \frac{2}{\|w\|}$

$$d(x_t) = \sum_{i=1}^{l} y_i \alpha_i x_i x_t + b_0$$

$y_i \rightarrow$ Class label for sup vector x_i

$l \rightarrow$ No. of support vector

$x_t \Rightarrow$ test tuple

$\alpha_i, b_0 \rightarrow$ Numeric parameters defined by user

$d(x_i) > 0 \rightarrow$ +ve class

$d(x_i) < 0 \rightarrow$ -ve class

SVM can be non-linearly separable data

Kernel functions → Maps data to higher dim feature space where the data points are linearly separable

8/3/22

Data Preprocessing

→ Y ?

real-world data might be incomplete, noisy, inconsistant.

eg: DOB — year / birth.

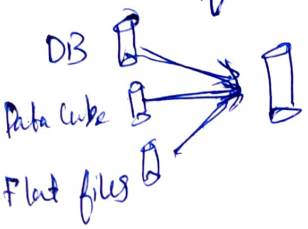
incomplete → Missing values.

noisy → outliers, errors.

① Data cleaning :- To fill the missing values & to smooth the data.

② Data integration.

data from diff sources are integrated.



③ Data Transformation.

* -2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Normalisation.

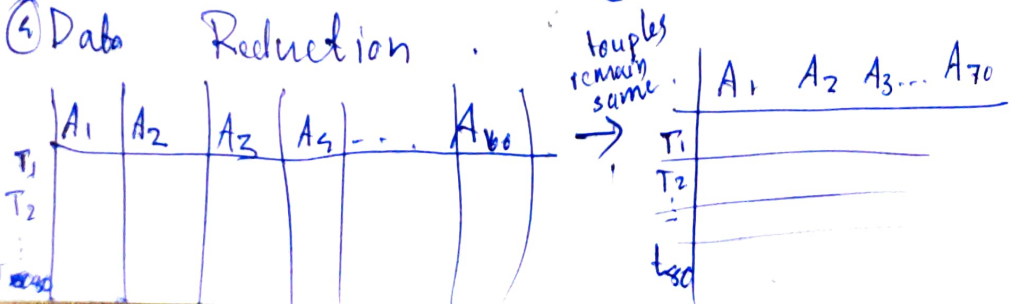
* Aggregation

monthly sales data → yearly sales data.

* generalization → Mapping from lower level concepts to high level

street → city.

④ Data Reduction.



⇒ FS, PCA
FS → feature selection
PCA → principal comp Analysis

① Data Cleaning

i) – fill the missing values.
ii) – smooth the data
– ...

* ① ignore the tuple.

if % of missing value is high, ~~loss of~~ ~~data will be lost~~.

if % of missing value is less (eg: 2 %?) we might loose imp information.

ignore → because only few values r present, so including them also wont have much affect.

cannot ignore.

② Fill the missing values manualy.

for smaller datasets & for fewer values its feasible.

③ Use some global constant to fill missing values.

eg: unknown, ∞

Drawbacks

sometimes mining alg may consider this pattern as intresting patterns, but will not be useful for users at all.

– not a good method.

✓ ④ Use attribute mean value.
– better method compared to others

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

↳ DT, Bayesian ~~fo~~ inferance.

Based on highest gain.

ii) Smooth the data.

→ Binning — methods are going to smooth the sorted data.

eg: price

4, 8, 15, 21, 21, 24, 25, 28, 34

→ * use equal frequency binning

same no of samples in each bin.

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

* smoothing by bin mean.

Bin 1 → 4, 8, 15 → 9, 9, 9

Bin 2 → 21, 21, 24 → 22, 22, 22

Bin 3 → 25, 28, 34 → 29, 29, 29

* smooth by bin boundary

Bin 1 → 4, 8, 15 → 4, 4, 15

Bin 2 → 21, 21, 24 → 21, 21, 24

Bin 3 → 25, 28, 34 → 25, 25, 34

increase each non boundary values to the closest boundary value.

* smooth by median.

change every value to median value.

→ equal width binning.

bin size = $\frac{\text{max} - \text{min}}{\text{no of elements}}$

- divide the range into N intervals of width w
- interval = $\frac{\text{Max} - \text{min}}{N}$

→ Boundaries -

min+w, min+2w ... min+nw

Data

eg: 5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 222, 230

Width: $\frac{230 - 5}{3} = 75$

(N=3)

Bin 1 = 5, 10, 11, 13, 15, 35, 50, 55, 72 [5 & 75]

Bin 2 = 92 (75 & 150)

Bin 3 = 204, 222, 230 (150 & 225)

3/3/22

~~Data Preprocessing~~

② Data Integration

→ check for redundancy in the attributes

→ An attribute may be redundant if it is derived from another attribute.

A, B → dependent attribute

* time of training ↑
* model won't be efficient

* remove redundant attributes, keep only independent ones

correlation ~~A~~B $r_{A,B} = \sum_{i=1}^{n} \frac{(A_i - \bar{A})(B_i - \bar{B})}{n \sigma_A \sigma_B}$

$\bar{A}, \bar{B}$ → mean values of A & B respectively.

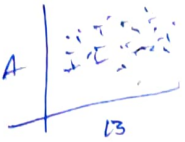
σ_A σ_B → std deviation of A & B

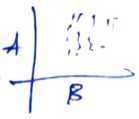
n → no of tuples

$r_{A,B}$ → +ve Dependent

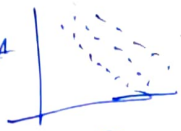
+ vely correlated

choose these attributes $r_{A,B}$ → 0 no correlation independent





$r_{A,B}$ → -ve dependent



-ve correlation

⇒ only for numerical data + this method can be used

⇒ Categorical Attributes

Chi square Measure (χ^2)

(Pearson correlation measure)

gender

Prof reading M F

magazine also 800

newspaper so 1000

→ A has c distinct values $a_1, a_2 \ldots a_c$

→ B has r " " $b_1, b_2 \ldots b_r$

Let (A_i, B_j) denote the event that A take value a_i & B take value b_j

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

O_{ij} → actual count → observed freq

e_{ij} → expected frequency

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

N → no of tuples

	M	F	T
M	250	200	450
NM	50	1000	1050
T	300	1200	

$$C_{11} = \frac{\text{count (male)} \times \text{count (magazine)}}{N}$$

$$= \frac{300 \times 450}{1500} = 90 \longrightarrow e_{11}$$

$C_{12} = 1200$

$$C_{(male, news)} = \frac{300 \times 1050}{1500} = 210$$

$$C_{(F, mag)} = \frac{1200 \times 450}{1500} = 360$$

$$C_{(F, new\ P)} = \frac{1200 \times 1050}{1500} = 840$$

$$X^2 = \frac{(250 - 90)^2}{90} + \frac{(50 - 210)^2}{210}$$

$$+ \frac{(200 - 360)^2}{360} + \frac{(1000 - 840)^2}{840}$$

$$= \frac{160^2}{90} + \frac{160^2}{210} + \frac{160^2}{360} + \frac{160^2}{840}$$

$$= 567.93$$

statistics 2 x 2 matrix = 10.828

$(r-1)(c-1)$ degrees of Freedom.

$1 \times 1 = 1$

→ calculated $(x^2) \geq$ Actual X^2 value

From stastics, → both are correlated.

$507.93 \geq 10.828$

Hence, Gender & reading mat are correlated.

→ if there is no deviation between observed & expected freq, χ^2 value will be less.

15/3/22.

Feature Selection

→ FS is a process that chooses an optimal subset of features from the original set.

Why FS?

→ To remove irrelevant features.

→ To increase the predictive accuracy.

→ To improve the learning efficiency – reduce the storage space – reduce the computational cost.

Criterias for feature Selection.

① searching for best subset of features.

② Criteria for evaluating diff subsets.

① ⟹ Heuristic Methods –

(i) Stepwise forward Selection.

$\{A_1, A_2, A_3, A_4, A_5, A_6\}$ [all attributes given in DS]

$\{\}$ start with empty set.

$\{A_1\}$ — best attribute

$\{A_1, A_4\}$ → add see best attribute.

⋮ continue the process

$\{A_1, A_4, A_5\}$ — reduced set

→ every step add the best attribute from the given set.

(ii) Stepwise backward elimination

$\{A_1\ A_2\ A_3\ A_4\ A_5\ A_6\}$ — original set.

|

every step remove one worst attribute.

$\{A_1, A_3, A_4, A_5, A_6\}$

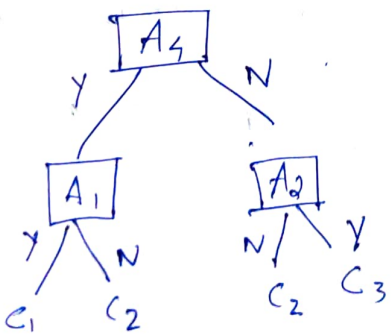
$\{A_1, A_4, A_6\}$ ← reduced set.

(iii) Combination of forward selection and Backward elimination.

→ each step select one best attribute and eliminate worst attribute.

(iv) Decision Tree.

→ gain measure or gini measure to find the best split attribute.



splitting attributes - best attributes.

← class label.

→ The attributes that does not occur in the tree are considered as irrelevant attributes.

DST → classification prob as well as feature selection

v) GA (Genetic Alg)

→ ① Initialise the cromosomes

1 0 1 1 0 1 1 0 — length of cromosome is equal to the no of features in the DS.

1 — feature present
0 — absence of feature

② → evaluate the chromosone by using fitness function.

Accuracy measure — fitness function maximization.

③ → selection, crossover, mutation

selection — select best chromosomes using fitness fn

crossover — exchange genes

mutation — flip one bit.

④ → ~~stop~~ check the stopping criteria —

max no of iterations or convergence of fitness function

② ⇒ Criteria to evaluate the Subsets of Feature

i) Entropy $-\sum_{i=1}^{c} P(C_i) \log_2(C_i)$

c — no of class labels.

$p(c_i)$ → prob of class c_i.

(ii) Gain

$$Info(A) = -\sum_{i=1}^{P} \frac{|D_i|}{|D|} I(D_j)$$

$|D_j|$ – no of tuples in the partition j

$|D|$ – total no of tuples in DS

$I(D_j)$ –

– If a particular feat gives the highest gain, we can keep the feature else eliminate it.

Attribute should give least gain?

(iii) Distance Measure

$$\sum_{i=1}^{n} \left(|x_i - y_i|^p \right)^{1/p}$$

(not for supremum dist)

$P = 1$ → manhatten distance.

$P = 2$ → euclidian dist.

similar features → same class label.

↓

keep the features (relevant).

else no consistency of class label – irrelevant features.

(3) Dependency Measures or Correlation measure

$$r(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n\,\sigma_A\,\sigma_B}$$

$r(x,y) = 0 \leftarrow$ independent.

eliminate the features:
- $r(x,y) = +ve \leftarrow$ x & y are +vely correlated.
- $r(x,y) = -ve$ x & y are -vely correlated.

(4) consistancy measure:

I/P feature values are same $\Rightarrow$ same class label (O/P).

(with dist measure check if features are similar?)

→ keep the features if they are consistant

(~~same~~ similar feature values → same class labels.

(5) Accuracy measure.

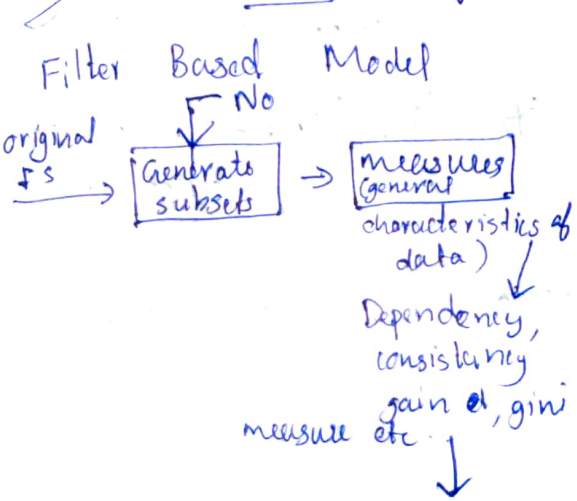
— eval ~~diff~~ all subsets using accuracy measure.

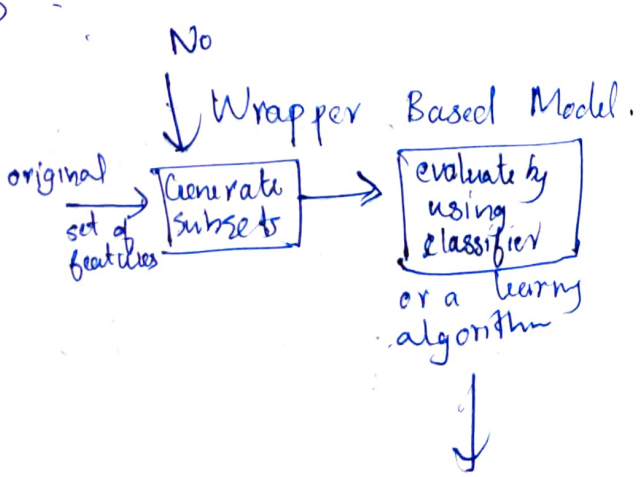
keep the ~~one~~ subset with highest accuracy.

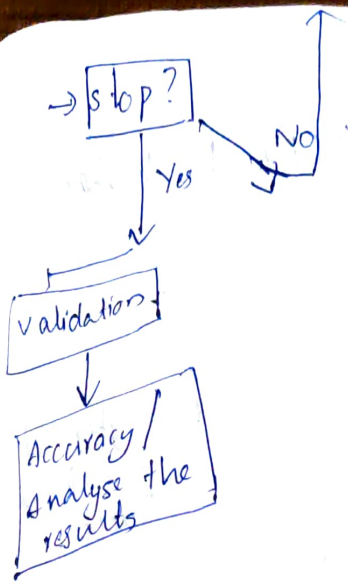
16/3/22

Models of F.S.

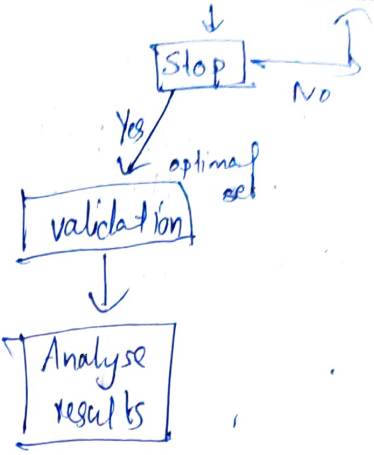
Filter Based Model







* Faster
no learning algorithm present



* slower training time is required
* computationally expensive learning algorithm.
* higher accuracy / good results, because already all features are evaluated.

→ GA-FS :- wrapper based model.
every chromosome is evaluated @ by classifier.
Fitness fn — is the accuracy got by using a learning algorithm.

Data Transformation

→ A fn that maps entire set of values of a given attribute to a new set.

Methods:
- smoothing - remove noise from data.
- feature construction - new features constructed from the given ones.
- aggregation - summarization
  eg: sales data aggregation
- Normalization (0 to 1)

- generalisation — concept hierarchy
  - lower level primitives are mapped to higher level primitives
  - street → city → country
- concept hierarchy (lower to higher / higher to lower) we can mine at diff levels — lower / higher / mid.

Normalization

- min max normalization.

$$v' = \frac{v - min}{max - min} (new\ Max_A - new\ min) + new\ min$$

v' → normalised value.

v → attribute value.

max, min — of the attribute

new max, new min — scale to which we want to convert the original value.

eg: Income $12000 to $98,000 normalise in range (0-1)

value — $73,000

$$v' = \frac{73,000 - 12,000}{98,000 - 12,000} (1-0) + 0 = 0.716$$

- Z-score Normalisation.

$$v' = \frac{v - \mu}{\sigma} \quad (\mu - mean,\ \sigma - S.D)$$

$\mu = 54,000 \quad \sigma = 16,000$

$v' = \frac{73,000 - 54,000}{16000}$

$= 1.22$

Decimal Scaling Based Normalization

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

j — smallest integer such that $\max |v'| < 1$

Attribute range from -986 to 917.

Max abs value is 986.

$j = 3$.

norm value in the raze of :

-0.986 to 0.917 → normalised value.

21/5/22

Principal Comp Analysis

↳ dimension reduction

↳ n of dim, select only, p no of dimension

$p < n$

without loosing any info from the given data.

Steps
1. Collect Data.
2. Substract mean from data.
3. Calculate covariance matrix.
4. Calculate eigen values & eigen vectors of the co-variance matrix.
5. Order them eigen vectors by eigen values (highest to lowest) least significant component.

feature vector : see principal component.

eigen vec 1, eigvec 2, eigvec 3 ---. eigVec n

PC1

choose only 'p' no of eigen vectors, where $p < n$.

('n' is given no of dimensions)

'p' → reduced dimension.

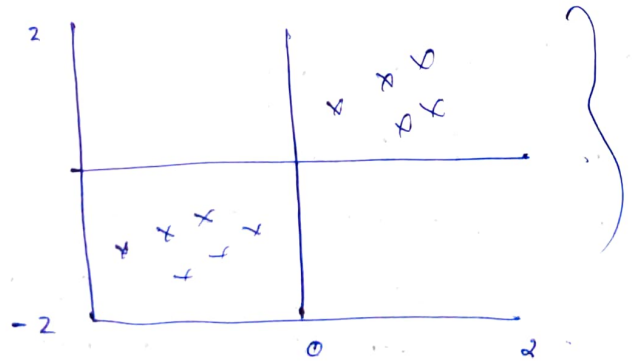
(5) Derive new data.

[If initial data is correlated, each data will be uncorrelated].

eg:

x	y	Subtract mean $x - \bar{x}$	$y - \bar{y}$
2.5	2.4	0.69	0.49
0.5	0.7	-1.31	-1.21
2.2	2.9	0.39	0.99
1.9	2.2	0.09	0.29
3.1	3.0	1.29	1.09
2.3	2.7	0.49	0.79
2	1.6	0.19	-0.31
1	1.1	-0.81	-0.81
1.5	1.6	-0.31	-0.31
1.1	0.9	-0.71	-1.01

$\bar{x} = 1.81$ $\bar{y} = 1.91$



x & y are correlated

(3) Find covariance matrix. (2 × 2)

$$cov(x, y) = \sum_{i=1}^{n} \frac{(x_i - \bar{x})(\bar{y}_i - \bar{y})}{(n-1)}$$

$$C = \begin{bmatrix} cov(x,x) & cov(x,y) \\ cov(y,x) & cov(y,y) \end{bmatrix}$$

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

$$C = \begin{bmatrix} 0.616155 & 0.615544 \\ 0.615544 & 0.716555 \end{bmatrix}$$

Find eigen vectors & eigen values for C.

↓

eg:

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

$\mu \vec{\eta} = v \vec{\eta}$.

$\mu \rightarrow n \times n$ matrix.

$\vec{\eta} \rightarrow$ non zero $n \times 1$ vector.

$v \rightarrow$ eigen value.

$\eta \rightarrow$ eigen vector.

① eigen values = $\begin{pmatrix} 0.0490833 \\ 1.2840277 1 \end{pmatrix}$.

(n — eigen vectors & n eigen values)
↳ dimension of data.

eigen vectors = $\begin{bmatrix} -0.7351 & -0.6778 \\ 0.6778 & -0.7351 \end{bmatrix}$

② arrange eigen vectors based on eigen values

$$\begin{bmatrix} -0.6778 & -0.7351 \\ -0.7351 & 0.6778 \end{bmatrix}.$$

PC1 PC2

(8) Derive new Data (P = 1)

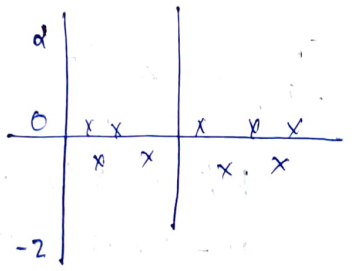
PC1 : -0.6778 , -0.7351

$-0.6778 \times 0.69 + (-0.7351) \times 0.49 = -0.827$

$-0.6778 \times (-1.31) + (-0.7351) \times (-1.21) = 1.7$

modified x = -0.827, 1.7, -0.992, -0.274, -1.6, 0.91, 0.099, 1.14, 0.43, 1.22

Modified Y (PC2) = -0.17, 0.14, 0.38, 0.13, 0.20, 0.17, -0.34, -0.04, 0.01, -0.16



→ now the dimensions are independent.

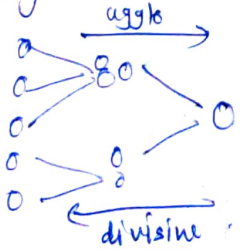
Best stg components of data, without loosing any information.

23/3/22

Clustering

→ Group the similar objects.
→ unsupervised (class label x)

Partition Based	hierarchical based
- k-mean, k medoid k – no of clusters - k should be known - every obj belongs to only one cluster. - choose k no of cluster center (k) - Find the dist of every pt with given cluster center.	→ agglomerative and divisive. * every obj in a single cluster. * merge * till every obj in a single cluster.



- assign the pt to nearest centroid
- repeat until no change in the centroids.

Adv
easy / simple.

Dis
very sensitive to noise

→ $O(n\ k\ t)$ — t: no of iterations; k: no of clusters; n: no of data pts

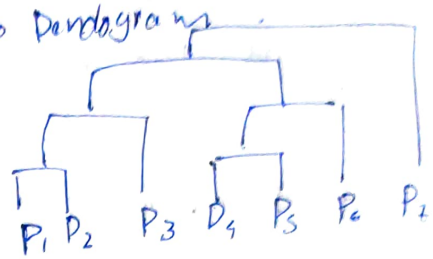
→ hard clustering.

* FCM (soft clustering).
→ obj can belong to more than one cluster with membership matrix.

→ k value not required

→ $O(n^2)$ — n: no of data pts.

→ Dendrogram.



Disadv
for high dimensional data, height of dendogram is going to increase.

→ cut the tree at some optimal level to get optimum no of clusters (using some optimisation techniques).

Density Based.

Density → no of pts with given radius (ε) of cluster

→ identify the dense and sparse region.

→ DBSCAN and OPTICS.

→ min pts & ε.

* core pts at ε rad, has min pts

p is core pt, if it covers min pts (given) within ε radius.

→ core pt, border pt & noise pt } entire dataset will be divided into 3.

border pt: its not core pt, but lies within the neighbourhood of core pt.

noise pt: not core pt nor border pt. DISCARD pt.

→ If a pt is a ~~be~~ core pt, make a cluster
if any border pt, add it to the nearest core pt.
discards the noise pt.

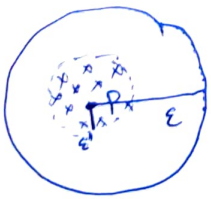
Adv
→ not sensitive to noise.
→ any shape of cluster is possible.

Dis
→ have to choose ε & min pts - diff for high dimensional data.
(Optimisation techniques should be used)

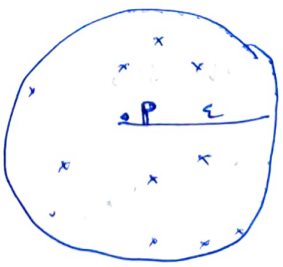
OPTICS (Ordering Points To Identify Clustering Strategy).

— need to identify
- core distance
- reachability distance.

— min pts = 5
ε = 6mm.



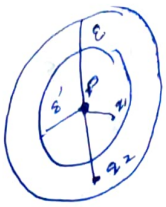
$\varepsilon' < \varepsilon$.
within ε' dist, 'P' is a core pt.
ε' → core distance.



ε is used by p to become core pt.

→ core distance of an obj 'p' is the smallest 'ε' value that makes p as core pt.
→ The reachability dist of an obj q wrt another object 'p' is the greater ~~of the~~ value of the core dist of

p and eucledian dist between p & q



p → core pt with core dist ε'

Reach (P, q1) — ε'

Reach (P, q2) — eucledian dist (P, q2)

Optics — It stores core dist & reachability dist for each obj.

- optics method mentains a list called seed list in which objects are sorted by reachability dist from their respective closest core objects.

28/3/22 Optics

Core point

→ core distance

→ reachability distance

→ optics computes an ordery of all objects in given db

→ It stores core dist and reachability dist of each data object

→ It maintains a seedlist

- objects are sorted by reachability distance from the respective closest core objects.

- points are prossed From smallest to higest reachability dist

Steps

① Begin with point P.

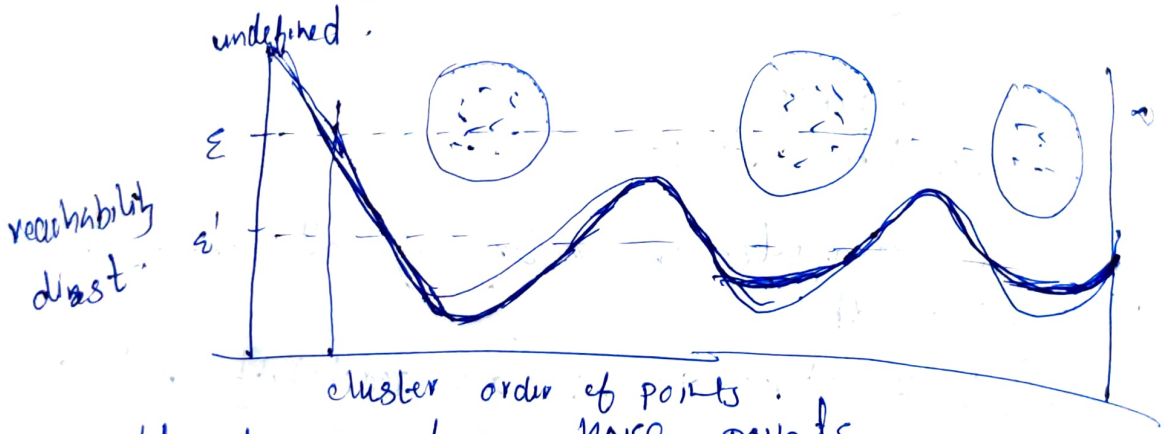
② within ε radius P retrieves all objs, determines core dist & reachability dist.

⑧ o/p p

- If p is core pt,
  - then For each obj q in the ε dist of p, update its reachability dist
  - insert q into list if its not processed

repeat until the i/p is fully consumed and list is empty.

- If p is not core pt
  - move on to the next object in the list or move on to the i/p ~~data~~ db if list is empty.

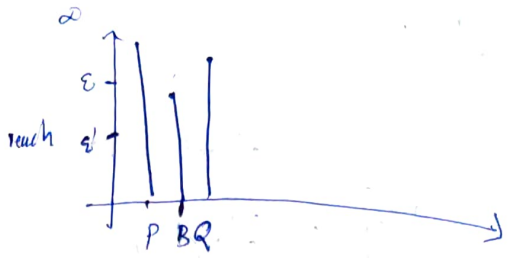


undefined → for noise points.

→ denser bumps, the pts will have very less reachability dist, & will be connected to a dense cluster

→ pts are processed on a specific order & this order selects an object & that is density reachable with lowest ε' value so that clusters with high density will be processed first

eg:



list: [B|∞ | C|40]

•B

[P | r | c]

① list starts from P

B|30 | C|40

② let density reachable pts of B be q, r, S

S|41 | q|39 | r|42 | C|40

→ only

q 39 | C|40 | S|41 | R 42

output pt : q

③ q is reachable to C & 2

X

C 40 | S 41 | R 42 | C 39 | 2 50

C 39 | S 41 | R 42 | 2 50

continued until list is empty or db is completely processed

→ pts processed in a manner that highly dense clusters processed first

→ clusters can be visualised after processing the pts

30/3/22

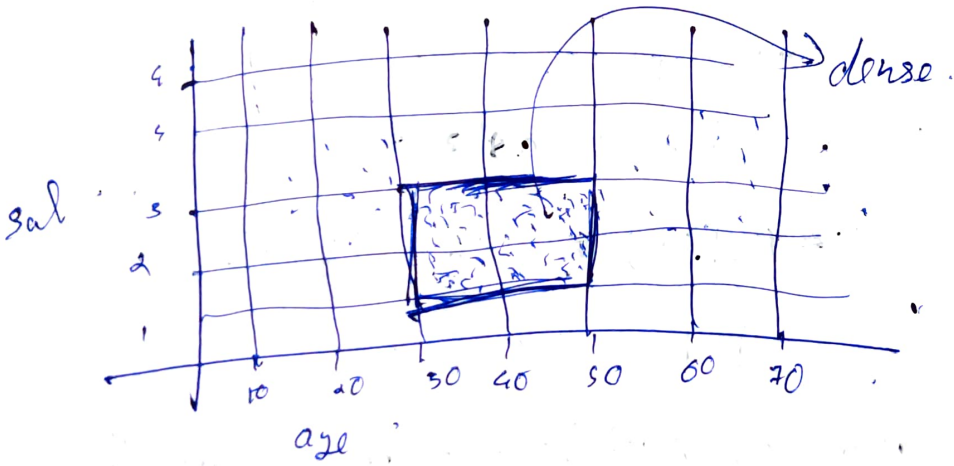
CLIQUE

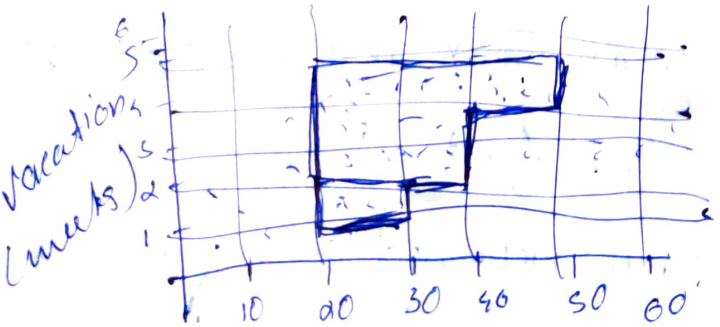
subspace clustering :

- First alg method for dimension growth of subspace clustery.
- It starts with a single dimension, dive
- Identifies dense units (no of mmpts).
- Subspace representing these dense units are intersected to form candidate search for higher dimension.

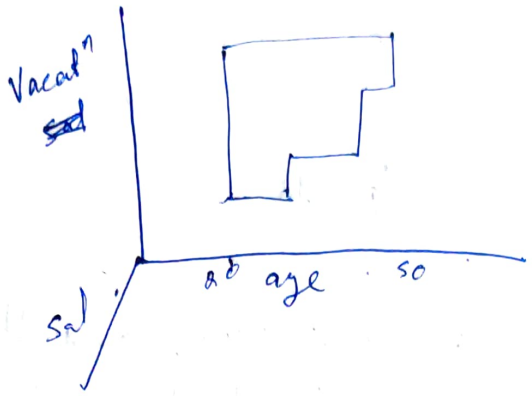
Maximal region - highly dense units.

Minimal region - low dense region.

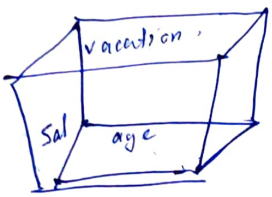




- intersecting (dimensions and dense regions)



- cube (we can view the data in the form of cube after intersection, at the end).



- density based clustering.

Drawback-

- decidiy grid size. } vary the parameters and find the optimal ones.
- Min pts.

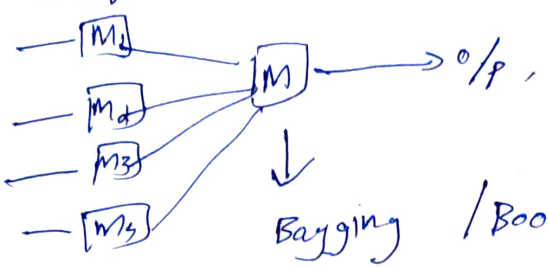
Classifiers-

- NB, DT, KNN, ANN.
  - DT: gain, gini
  - ANN: backpropgatn, perceptron model
- probability based classifier

Ensemble Classifiers

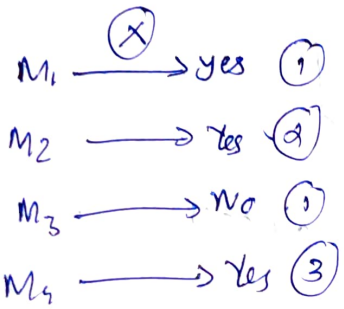
→ set of classifiers.
→ the learning is based on multiple classifiers; not on a single classifier.

single classifiers -



→ Bagging → each classifier returns the class prediction for 'x' which counts as a vote.

x is assigned to the class with most votes -



→ Boosting → assign wt to each classifier not based on how well the ~~class~~ classifier performed

→ sum the wts of each classifier that assigned class 'c' to 'x'.

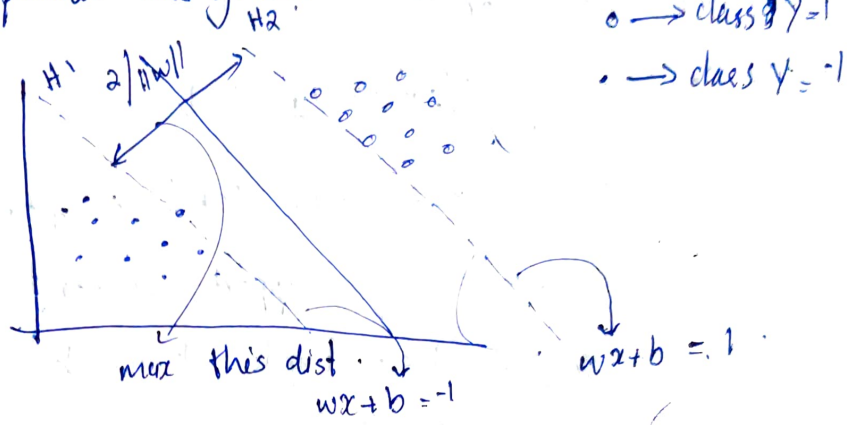
→ The class with highest sum is winner.

→ prior knowledge of classifiers are req. as weights should be assigned

[Eg: Naive bayes — should have independent attributes. If you have dependent attributes, assign less weight to Naive Bayes]

SVM (Support Vector Machine)

→ Find the maximum margin hyperplane (MMH) that seperates the given classes.



Any hyperplane can be written as $wx + b = 0$

w is normal vector to the hyperplane,

$w_1, w_2 \ldots w_n$, n is the no of attributes in the dataset.

x — is the training tuple

b — bias

→ Any $y=1$ that falls above hyperplane

$$wx + b > 0$$

→ any b's that falls below hyperplane.

$$wx + b < 0$$

→ By geometry,

dist betwn H_1 & H_2 = $2/\|w\|$.

Maximise the distance between H_1 & H_2.

- this will well seperate the classes.

→ Minimise w.

Support Vectors:

→ The pts that lie on H_1 & H_2 are called as support vectors.

Finding Class Labels:

$$d(x_1) = \sum_{i=1}^{l} y_i \alpha_i x_i x_t + b_0$$

y_i – is the class label of support vector x_i

l → no of support vectors.

x_t → test tuple to which class label has to be found.

α_i, b_0 → ~~and constant values~~ numeric parameters ~~etc~~ defined by user.

$d(x_t) > 0$ → +ve class.

$d(x_t) < 0$ → -ve class.

SVM – non linearly seperable data (can be used for)

- kernel functions are used ~~for non linear seperable~~ data vectors are mapped to high dimensional feature space, where these can be seperated in a linear fashion.

└ polynomial, sigmoid, radial -