

Data Mining:



— Chapter 1 —
— Introduction —



Coverage (Chapters 1-7 of This Book)

- The book will be divided into courses:
 - Introduction to data warehousing and data mining
 - Data mining: Principles and algorithms
- Part 1 Coverage
 - Introduction
 - Data Preprocessing
 - Data Warehouse and OLAP Technology: An Introduction
 - Advanced Data Cube Technology and Data Generalization
 - Mining Frequent Patterns, Association and Correlations
 - Classification and Prediction
 - Cluster Analysis



Part2 Coverage (Chapters 8-11 of This Book)

- Mining data streams, time-series, and sequence data
- Mining graphs, social networks and multi-relational data
- Mining object, spatial, multimedia, text and Web data
 - Mining complex data objects
 - Spatial and spatiotemporal data mining
 - Multimedia data mining
 - Text mining
 - Web mining
- Applications and trends of data mining
 - Mining business & biological data
 - Visual data mining
 - Data mining and society: Privacy-preserving data mining
- Additional (often current) themes could be added to the course

Chapter 1. Introduction

- Motivation: Why data mining?
- What is data mining?
- Data Mining: On what kind of data?
- Data mining functionality
- Classification of data mining systems
- Top-10 most popular data mining algorithms
- Major issues in data mining
- Overview of the course

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

Evolution of Sciences



- Before 1600, **empirical science**
- 1600-1950s, **theoretical science**
 - Each discipline has grown a *theoretical* component. Theoretical models often motivate experiments and generalize our understanding.
- 1950s-1990s, **computational science**
 - Over the last 50 years, most disciplines have grown a third, *computational* branch (e.g. empirical, theoretical, and computational ecology, or physics, or linguistics.)
 - Computational Science traditionally meant simulation. It grew out of our inability to find closed-form solutions for complex mathematical models.
- 1990-now, **data science**
 - The flood of data from new scientific instruments and simulations
 - The ability to economically store and manage petabytes of data online
 - The Internet and computing Grid that makes all these archives universally accessible
 - Scientific info. management, acquisition, organization, query, and visualization tasks scale almost linearly with data volumes. **Data mining** is a major new challenge!
- Jim Gray and Alex Szalay, *The World Wide Telescope: An Archetype for Online Science*, Comm. ACM, 45(11): 50-54, Nov. 2002

Evolution of Database Technology

- 1960s:
 - Data collection, database creation, IMS and network DBMS
- 1970s:
 - Relational data model, relational DBMS implementation
- 1980s:
 - RDBMS, advanced data models (extended-relational, OO, deductive, etc.)
 - Application-oriented DBMS (spatial, scientific, engineering, etc.)
- 1990s:
 - Data mining, data warehousing, multimedia databases, and Web databases
- 2000s
 - Stream data management and mining
 - Data mining and its applications
 - Web technology (XML, data integration) and global information systems



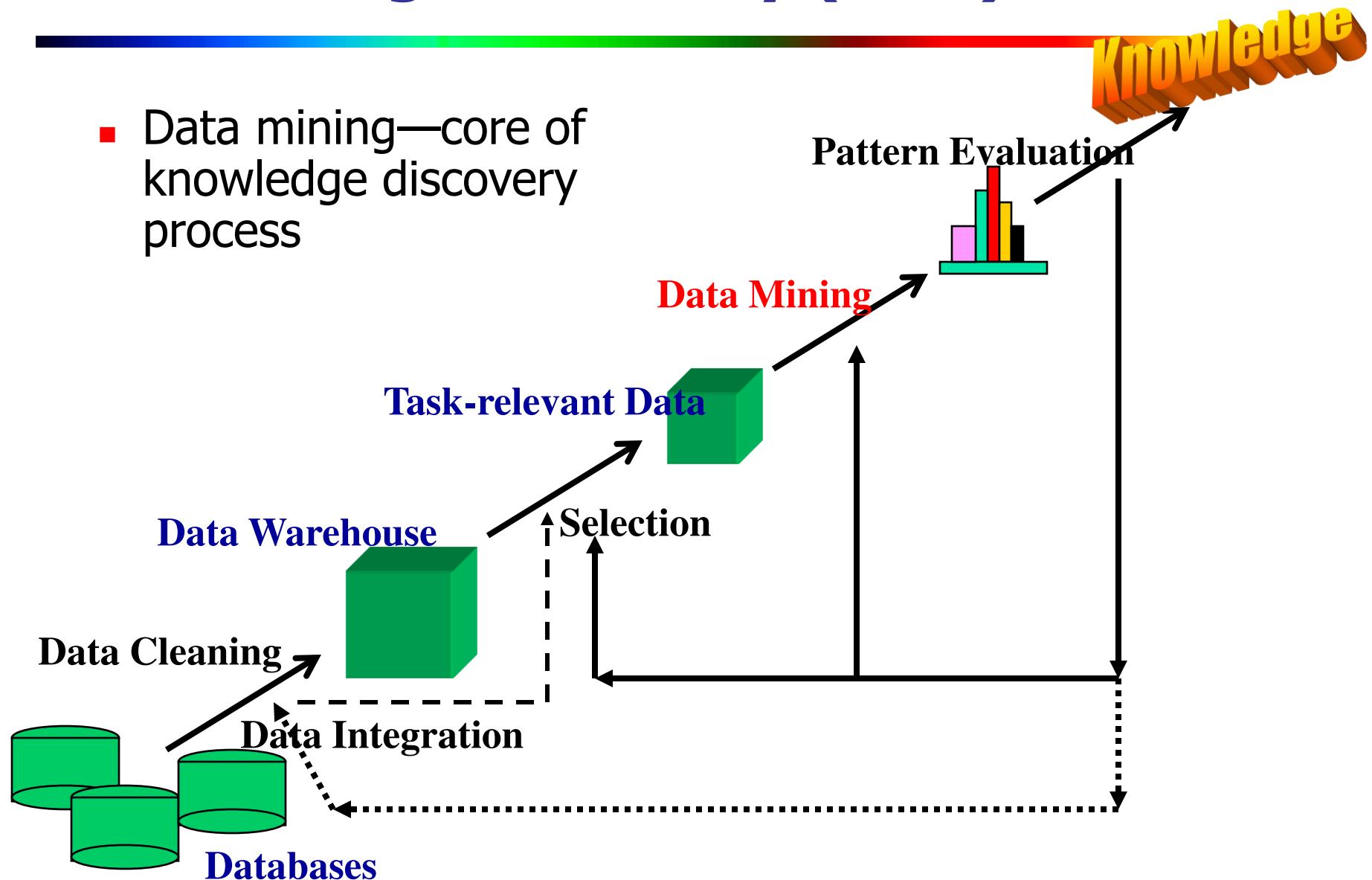
What Is Data Mining?

- 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
 - Data mining: a misnomer?
- Alternative names
 - Knowledge discovery (mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.
- Watch out: Is everything “data mining”?
 - Simple search and query processing
 - (Deductive) expert systems

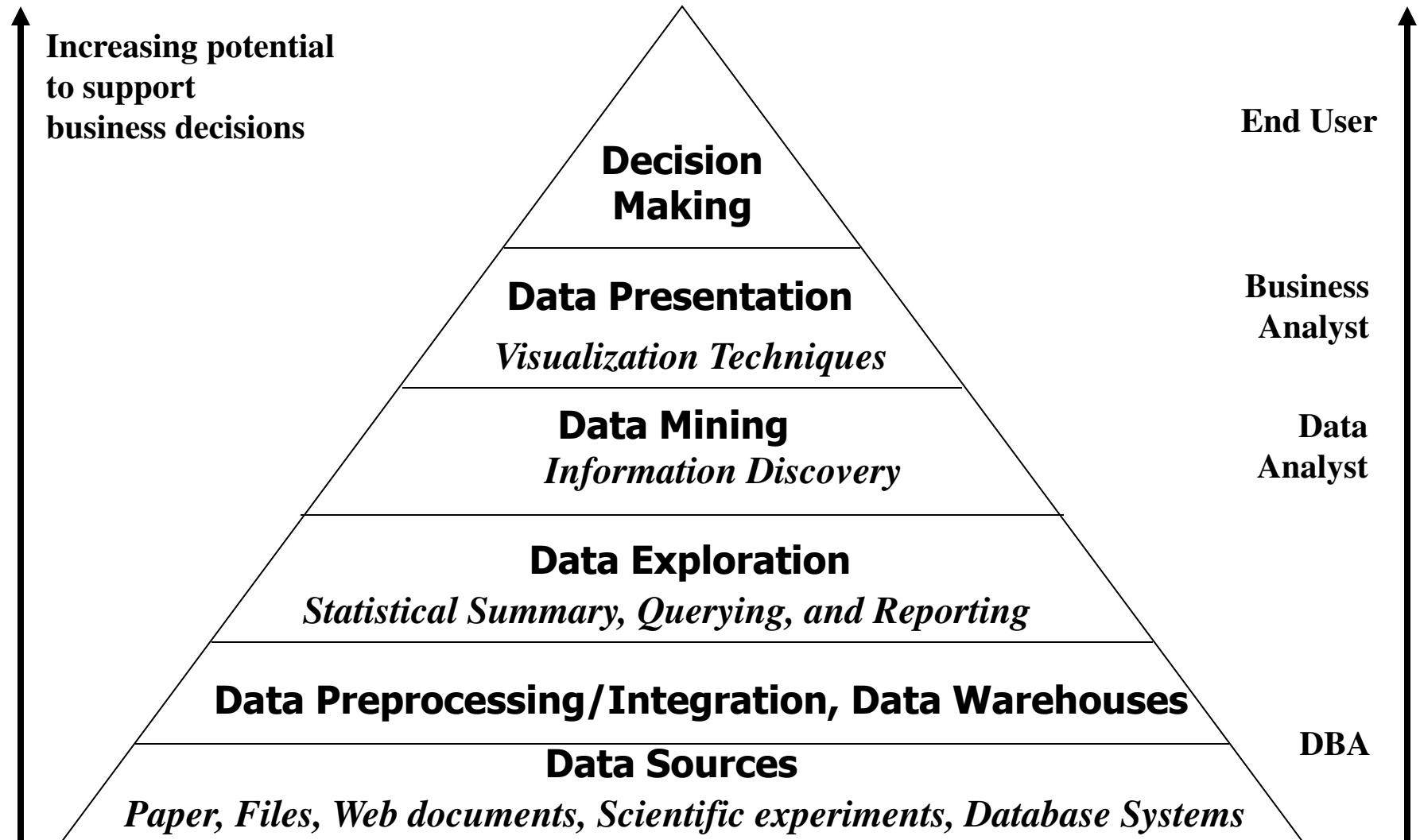


Knowledge Discovery (KDD) Process

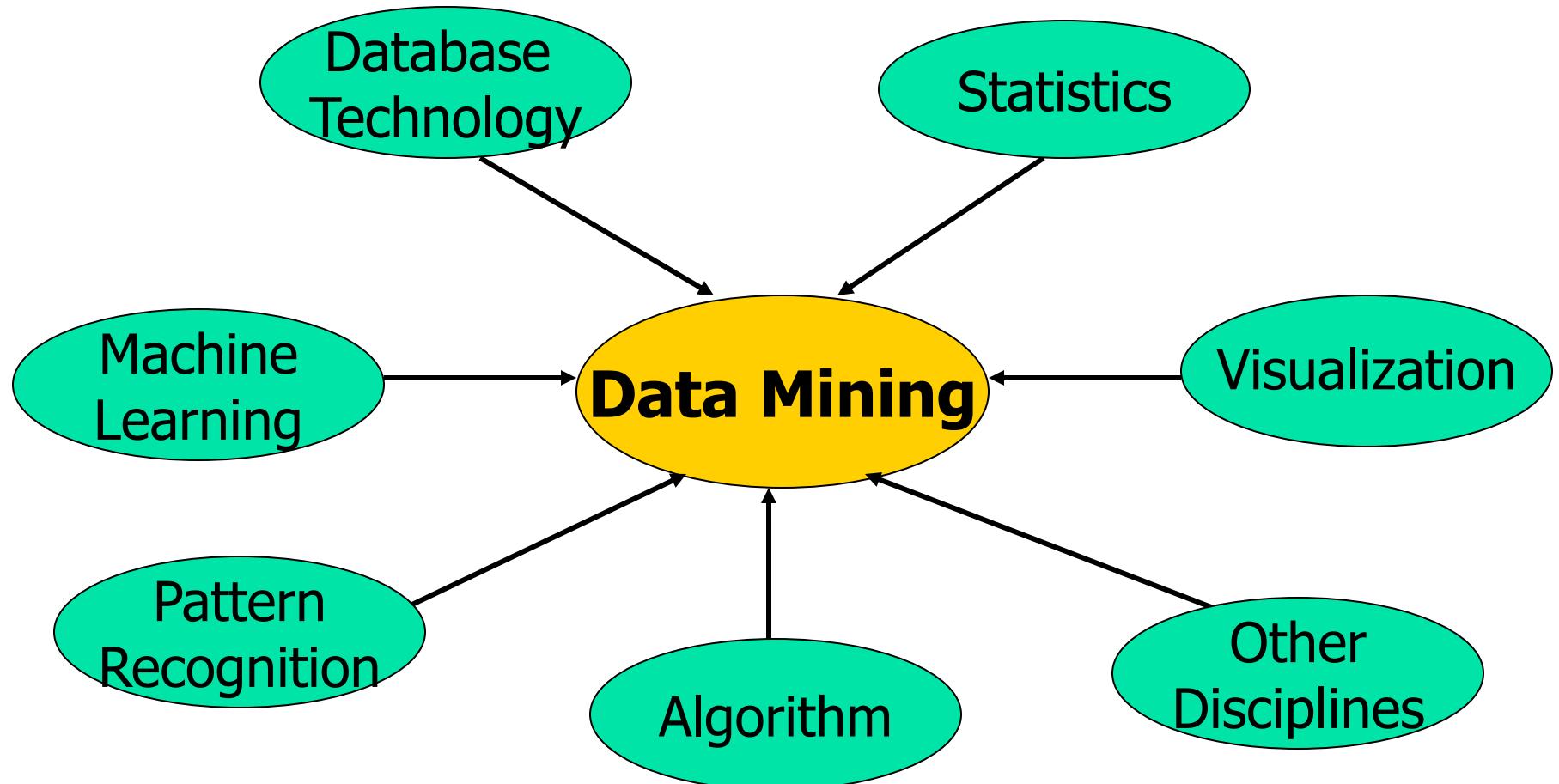
- Data mining—core of knowledge discovery process



Data Mining and Business Intelligence



Data Mining: Confluence of Multiple Disciplines



Why Not Traditional Data Analysis?

- 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 and legacy databases
 - Spatial, spatiotemporal, multimedia, text and Web data
 - Software programs, scientific simulations
- New and sophisticated applications

Multi-Dimensional View of Data Mining

- **Data to be mined**

- Relational, data warehouse, transactional, stream, object-oriented/relational, active, spatial, time-series, text, multi-media, heterogeneous, legacy, WWW

- **Knowledge to be mined**

- Characterization, discrimination, association, classification, clustering, trend/deviation, outlier analysis, etc.
- Multiple/integrated functions and mining at multiple levels

- **Techniques utilized**

- Database-oriented, data warehouse (OLAP), machine learning, statistics, visualization, etc.

- **Applications adapted**

- Retail, telecommunication, banking, fraud analysis, bio-data mining, stock market analysis, text mining, Web mining, etc.

Data Mining: Classification Schemes

- General functionality
 - Descriptive data mining
 - Predictive data mining
- Different views lead to different classifications
 - Data view: Kinds of data to be mined
 - Knowledge view: Kinds of knowledge to be discovered
 - Method view: Kinds of techniques utilized
 - Application view: Kinds of applications adapted

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, temporal data, sequence data (incl. bio-sequences)
 - Structure data, graphs, social networks and multi-linked data
 - Object-relational databases
 - Heterogeneous databases and legacy databases
 - Spatial data and spatiotemporal data
 - Multimedia database
 - Text databases
 - The World-Wide Web

Data Mining Functionalities

- Multidimensional concept description: Characterization and discrimination
 - Generalize, summarize, and contrast data characteristics, e.g., dry vs. wet regions
- Frequent patterns, association, correlation vs. causality
 - Diaper → Beer [0.5%, 75%] (Correlation or causality?)
- Classification and prediction
 - Construct models (functions) that 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 or missing numerical values

Data Mining Functionalities (2)

- Cluster analysis
 - Class label is unknown: Group data to form new classes, e.g., cluster houses to find distribution patterns
 - Maximizing intra-class similarity & minimizing interclass similarity
- Outlier analysis
 - Outlier: Data object that does not comply with the general behavior of the data
 - Noise or exception? Useful in fraud detection, rare events analysis
- Trend and evolution analysis
 - Trend and deviation: e.g., regression analysis
 - Sequential pattern mining: e.g., digital camera → large SD memory
 - Periodicity analysis
 - Similarity-based analysis
- Other pattern-directed or statistical analyses

Top-10 Most Popular DM Algorithms: 18 Identified Candidates (I)

- Classification
 - #1. C4.5: Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann., 1993.
 - #2. CART: L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, 1984.
 - #3. K Nearest Neighbours (kNN): Hastie, T. and Tibshirani, R. 1996. Discriminant Adaptive Nearest Neighbor Classification. TPAMI. 18(6)
 - #4. Naive Bayes Hand, D.J., Yu, K., 2001. Idiot's Bayes: Not So Stupid After All? Internat. Statist. Rev. 69, 385-398.
- Statistical Learning
 - #5. SVM: Vapnik, V. N. 1995. The Nature of Statistical Learning Theory. Springer-Verlag.
 - #6. EM: McLachlan, G. and Peel, D. (2000). Finite Mixture Models. J. Wiley, New York. Association Analysis
 - #7. Apriori: Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules. In VLDB '94.
 - #8. FP-Tree: Han, J., Pei, J., and Yin, Y. 2000. Mining frequent patterns without candidate generation. In SIGMOD '00.

The 18 Identified Candidates (II)

- Link Mining
 - #9. PageRank: Brin, S. and Page, L. 1998. The anatomy of a large-scale hypertextual Web search engine. In WWW-7, 1998.
 - #10. HITS: Kleinberg, J. M. 1998. Authoritative sources in a hyperlinked environment. SODA, 1998.
- Clustering
 - #11. K-Means: MacQueen, J. B., Some methods for classification and analysis of multivariate observations, in Proc. 5th Berkeley Symp. Mathematical Statistics and Probability, 1967.
 - #12. BIRCH: Zhang, T., Ramakrishnan, R., and Livny, M. 1996. BIRCH: an efficient data clustering method for very large databases. In SIGMOD '96.
- Bagging and Boosting
 - #13. AdaBoost: Freund, Y. and Schapire, R. E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55, 1 (Aug. 1997), 119-139.

The 18 Identified Candidates (III)

- Sequential Patterns
 - #14. GSP: Srikant, R. and Agrawal, R. 1996. Mining Sequential Patterns: Generalizations and Performance Improvements. In Proceedings of the 5th International Conference on Extending Database Technology, 1996.
 - #15. PrefixSpan: J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal and M-C. Hsu. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In ICDE '01.
- Integrated Mining
 - #16. CBA: Liu, B., Hsu, W. and Ma, Y. M. Integrating classification and association rule mining. KDD-98.
- Rough Sets
 - #17. Finding reduct: Zdzislaw Pawlak, Rough Sets: Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Norwell, MA, 1992
- Graph Mining
 - #18. gSpan: Yan, X. and Han, J. 2002. gSpan: Graph-Based Substructure Pattern Mining. In ICDM '02.

Top-10 Algorithm Finally Selected at ICDM'06

- #1: C4.5 (61 votes)
- #2: K-Means (60 votes)
- #3: SVM (58 votes)
- #4: Apriori (52 votes)
- #5: EM (48 votes)
- #6: PageRank (46 votes)
- #7: AdaBoost (45 votes)
- #7: kNN (45 votes)
- #7: Naive Bayes (45 votes)
- #10: CART (34 votes)

Major Issues in Data Mining

Mining methodology

- Mining different kinds of knowledge from diverse data types, e.g., bio, stream, Web
- Performance: efficiency, effectiveness, and scalability
- Pattern evaluation: the interestingness problem
- Incorporation of background knowledge
- Handling noise and incomplete data
- Parallel, distributed and incremental mining methods
- Integration of the discovered knowledge with existing one: knowledge fusion

User interaction

- Data mining query languages and ad-hoc mining
- Expression and visualization of data mining results
- Interactive mining of knowledge at multiple levels of abstraction

Applications and social impacts

- Domain-specific data mining & invisible data mining
- Protection of data security, integrity, and privacy

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**)
 - Conf. on Principles and practices of Knowledge Discovery and Data Mining (**PKDD**)
 - Pacific-Asia Conf. on Knowledge Discovery and Data Mining (**PAKDD**)
- Other related conferences
 - ACM SIGMOD
 - VLDB
 - (IEEE) ICDE
 - WWW, SIGIR
 - ICML, CVPR, NIPS
- 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.

Recommended Reference Books

- **S. Chakrabarti.** **Mining the Web: Statistical Analysis of Hypertext and Semi-Structured Data.** Morgan Kaufmann, 2002
- **R. O. Duda, P. E. Hart, and D. G. Stork,** **Pattern Classification, 2ed., Wiley-Interscience,** 2000
- **T. Dasu and T. Johnson.** **Exploratory Data Mining and Data Cleaning.** John Wiley & Sons, 2003
- **U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy.** **Advances in Knowledge Discovery and Data Mining.** AAAI/MIT Press, 1996
- **U. Fayyad, G. Grinstein, and A. Wierse,** **Information Visualization in Data Mining and Knowledge Discovery,** Morgan Kaufmann, 2001
- **J. Han and M. Kamber.** **Data Mining: Concepts and Techniques.** Morgan Kaufmann, 2nd ed., 2006
- **D. J. Hand, H. Mannila, and P. Smyth,** **Principles of Data Mining,** MIT Press, 2001
- **T. Hastie, R. Tibshirani, and J. Friedman,** **The Elements of Statistical Learning: Data Mining, Inference, and Prediction,** Springer-Verlag, 2001
- **B. Liu,** **Web Data Mining,** Springer 2006.
- **T. M. Mitchell,** **Machine Learning,** McGraw Hill, 1997
- **G. Piatetsky-Shapiro and W. J. Frawley.** **Knowledge Discovery in Databases.** AAAI/MIT Press, 1991
- **P.-N. Tan, M. Steinbach and V. Kumar,** **Introduction to Data Mining,** Wiley, 2005
- **S. M. Weiss and N. Indurkhya,** **Predictive Data Mining,** Morgan Kaufmann, 1998
- **I. H. Witten and E. Frank,** **Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations,** Morgan Kaufmann, 2nd ed. 2005

Summary

- Data mining: Discovering interesting patterns from large amounts of data
- A natural evolution of database technology, in great demand, with wide applications
- A KDD process includes data cleaning, data integration, data selection, transformation, data mining, pattern evaluation, and knowledge presentation
- Mining can be performed in a variety of information repositories
- Data mining functionalities: characterization, discrimination, association, classification, clustering, outlier and trend analysis, etc.
- Data mining systems and architectures
- Major issues in data mining

Supplementary Lecture Slides

- Note: The slides following the end of chapter summary are supplementary slides that could be useful for supplementary readings or teaching
- These slides may have its corresponding text contents in the book chapters, but were omitted due to limited time in author's own course lecture
- The slides in other chapters have similar convention and treatment

Why Data Mining?—Potential Applications

- Data analysis and decision support
 - Market analysis and management
 - Target marketing, customer relationship management (CRM), market basket analysis, cross selling, market segmentation
 - Risk analysis and management
 - Forecasting, customer retention, improved underwriting, quality control, competitive analysis
 - Fraud detection and detection of unusual patterns (outliers)
- Other Applications
 - Text mining (news group, email, documents) and Web mining
 - Stream data mining
 - Bioinformatics and bio-data analysis

Ex. 1: Market Analysis and Management

- Where does the data come from?—Credit card transactions, loyalty cards, discount coupons, customer complaint calls, plus (public) lifestyle studies
- Target marketing
 - Find clusters of “model” customers who share the same characteristics: interest, income level, spending habits, etc.
 - Determine customer purchasing patterns over time
- Cross-market analysis—Find associations/co-relations between product sales, & predict based on such association
- Customer profiling—What types of customers buy what products (clustering or classification)
- Customer requirement analysis
 - Identify the best products for different groups of customers
 - Predict what factors will attract new customers
- Provision of summary information
 - Multidimensional summary reports
 - Statistical summary information (data central tendency and variation)

Ex. 2: Corporate Analysis & Risk Management

- Finance planning and asset evaluation
 - cash flow analysis and prediction
 - contingent claim analysis to evaluate assets
 - cross-sectional and time series analysis (financial-ratio, trend analysis, etc.)
- Resource planning
 - summarize and compare the resources and spending
- Competition
 - monitor competitors and market directions
 - group customers into classes and a class-based pricing procedure
 - set pricing strategy in a highly competitive market

Ex. 3: Fraud Detection & Mining Unusual Patterns

- Approaches: Clustering & model construction for frauds, outlier analysis
- Applications: Health care, retail, credit card service, telecomm.
 - Auto insurance: ring of collisions
 - Money laundering: suspicious monetary transactions
 - Medical insurance
 - Professional patients, ring of doctors, and ring of references
 - Unnecessary or correlated screening tests
 - Telecommunications: phone-call fraud
 - Phone call model: destination of the call, duration, time of day or week. Analyze patterns that deviate from an expected norm
 - Retail industry
 - Analysts estimate that 38% of retail shrink is due to dishonest employees
 - Anti-terrorism

KDD Process: Several Key Steps

- Learning the application domain
 - relevant prior knowledge and goals of application
- Creating a target data set: data selection
- Data cleaning and preprocessing: (may take 60% of effort!)
- Data reduction and transformation
 - Find useful features, dimensionality/variable reduction, invariant representation
- Choosing functions of data mining
 - summarization, classification, regression, association, clustering
- Choosing the mining algorithm(s)
- Data mining: search for patterns of interest
- Pattern evaluation and knowledge presentation
 - visualization, transformation, removing redundant patterns, etc.
- Use of discovered knowledge

Are All the “Discovered” Patterns Interesting?

- Data mining may generate thousands of patterns: Not all of them are interesting
 - Suggested approach: Human-centered, query-based, focused mining
- **Interestingness measures**
 - A pattern is **interesting** if it is easily understood by humans, valid on new or test data with some degree of certainty, potentially useful, novel, or validates some hypothesis that a user seeks to confirm
- **Objective vs. subjective interestingness measures**
 - **Objective**: based on **statistics and structures of patterns**, e.g., support, confidence, etc.
 - **Subjective**: based on **user’s belief** in the data, e.g., unexpectedness, novelty, actionability, etc.

Find All and Only Interesting Patterns?

- Find all the interesting patterns: Completeness
 - Can a data mining system find all the interesting patterns? Do we need to find all of the interesting patterns?
 - Heuristic vs. exhaustive search
 - Association vs. classification vs. clustering
- Search for only interesting patterns: An optimization problem
 - Can a data mining system find only the interesting patterns?
 - Approaches
 - First generate all the patterns and then filter out the uninteresting ones
 - Generate only the interesting patterns—mining query optimization

Other Pattern Mining Issues

- Precise patterns vs. approximate patterns
 - Association and correlation mining: possible find sets of precise patterns
 - But approximate patterns can be more compact and sufficient
 - How to find high quality approximate patterns??
 - Gene sequence mining: approximate patterns are inherent
 - How to derive efficient approximate pattern mining algorithms??
- Constrained vs. non-constrained patterns
 - Why constraint-based mining?
 - What are the possible kinds of constraints? How to push constraints into the mining process?

A Few Announcements

- A new section CS412ADD: CRN 48711 and its rules/arrangements
- 4th Unit for I2CS students
 - Survey report for mining new types of data
- 4th Unit for in-campus students
 - High quality implementation of one selected (to be discussed with TA/Instructor) data mining algorithm in the textbook
 - Or, a research report if you plan to devote your future research thesis on data mining

Why Data Mining Query Language?

- Automated vs. query-driven?
 - Finding all the patterns autonomously in a database?—unrealistic because the patterns could be too many but uninteresting
- Data mining should be an interactive process
 - User directs what to be mined
- Users must be provided with a set of **primitives** to be used to communicate with the data mining system
- Incorporating these primitives in a **data mining query language**
 - More flexible user interaction
 - Foundation for design of graphical user interface
 - Standardization of data mining industry and practice

Primitives that Define a Data Mining Task

- Task-relevant data
 - Database or data warehouse name
 - Database tables or data warehouse cubes
 - Condition for data selection
 - Relevant attributes or dimensions
 - Data grouping criteria
- Type of knowledge to be mined
 - Characterization, discrimination, association, classification, prediction, clustering, outlier analysis, other data mining tasks
- Background knowledge
- Pattern interestingness measurements
- Visualization/presentation of discovered patterns

Primitive 3: Background Knowledge

- A typical kind of background knowledge: Concept hierarchies
- Schema hierarchy
 - E.g., street < city < province_or_state < country
- Set-grouping hierarchy
 - E.g., {20-39} = young, {40-59} = middle_aged
- Operation-derived hierarchy
 - email address: hagonzal@cs.uiuc.edu
login-name < department < university < country
- Rule-based hierarchy
 - low_profit_margin (X) <= price(X, P₁) and cost (X, P₂) and (P₁ - P₂) < \$50

Primitive 4: Pattern Interestingness Measure

- Simplicity
 - e.g., (association) rule length, (decision) tree size
- Certainty
 - e.g., confidence, $P(A|B) = \#(A \text{ and } B) / \#(B)$, classification reliability or accuracy, certainty factor, rule strength, rule quality, discriminating weight, etc.
- Utility
 - potential usefulness, e.g., support (association), noise threshold (description)
- Novelty
 - not previously known, surprising (used to remove redundant rules, e.g., Illinois vs. Champaign rule implication support ratio)

Primitive 5: Presentation of Discovered Patterns

- Different backgrounds/usages may require **different forms of representation**
 - E.g., rules, tables, crosstabs, pie/bar chart, etc.
- **Concept hierarchy** is also important
 - Discovered knowledge might be more understandable when represented at **high level of abstraction**
 - Interactive **drill up/down, pivoting, slicing and dicing** provide different perspectives to data
- Different kinds of **knowledge** require different representation: association, classification, clustering, etc.

DMQL—A Data Mining Query Language

- Motivation
 - A DMQL can provide the ability to support ad-hoc and interactive data mining
 - By providing a standardized language like SQL
 - Hope to achieve a similar effect like that SQL has on relational database
 - Foundation for system development and evolution
 - Facilitate information exchange, technology transfer, commercialization and wide acceptance
- Design
 - DMQL is designed with the primitives described earlier

An Example Query in DMQL

Example 1.11 Mining classification rules. Suppose, as a marketing manager of *AllElectronics*, you would like to classify customers based on their buying patterns. You are especially interested in those customers whose salary is no less than \$40,000, and who have bought more than \$1,000 worth of items, each of which is priced at no less than \$100. In particular, you are interested in the customer's age, income, the types of items purchased, the purchase location, and where the items were made. You would like to view the resulting classification in the form of rules. This data mining query is expressed in DMQL³ as follows, where each line of the query has been enumerated to aid in our discussion.

```
use database AllElectronics_db
use hierarchy location_hierarchy for T.branch, age_hierarchy for C.age
mine classification as promising_customers
in relevance to C.age, C.income, I.type, I.place_made, T.branch
from customer C, item I, transaction T
where I.item_ID = T.item_ID and C.cust_ID = T.cust_ID
      and C.income ≥ 40,000 and I.price ≥ 100
group by T.cust_ID
having sum(I.price) ≥ 1,000
display as rules
```

Other Data Mining Languages & Standardization Efforts

- Association rule language specifications
 - MSQL (Imielinski & Virmani'99)
 - MineRule (Meo Psaila and Ceri'96)
 - Query flocks based on Datalog syntax (Tsur et al'98)
- OLEDB for DM (Microsoft'2000) and recently DMX (Microsoft SQLServer 2005)
 - Based on OLE, OLE DB, OLE DB for OLAP, C#
 - Integrating DBMS, data warehouse and data mining
- DMML (Data Mining Mark-up Language) by DMG (www.dmg.org)
 - Providing a platform and process structure for effective data mining
 - Emphasizing on deploying data mining technology to solve business problems

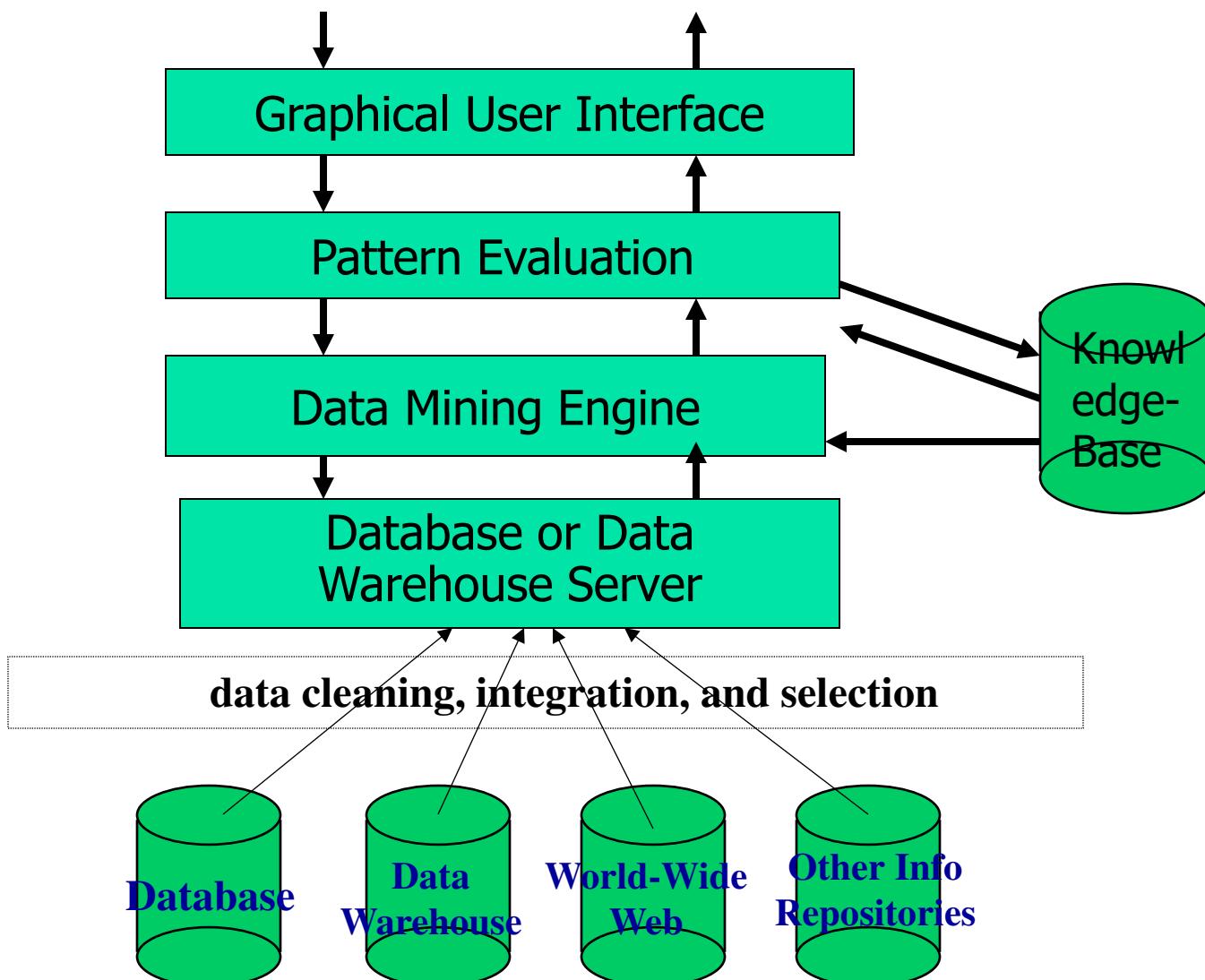
Integration of Data Mining and Data Warehousing

- **Data mining systems, DBMS, Data warehouse systems coupling**
 - No coupling, loose-coupling, semi-tight-coupling, tight-coupling
- **On-line analytical mining data**
 - integration of mining and OLAP technologies
- **Interactive mining multi-level knowledge**
 - Necessity of mining knowledge and patterns at different levels of abstraction by drilling/rolling, pivoting, slicing/dicing, etc.
- **Integration of multiple mining functions**
 - Characterized classification, first clustering and then association

Coupling Data Mining with DB/DW Systems

- No coupling—flat file processing, not recommended
- Loose coupling
 - Fetching data from DB/DW
- Semi-tight coupling—enhanced DM performance
 - Provide efficient implement a few data mining primitives in a DB/DW system, e.g., sorting, indexing, aggregation, histogram analysis, multiway join, precomputation of some stat functions
- Tight coupling—A uniform information processing environment
 - DM is smoothly integrated into a DB/DW system, mining query is optimized based on mining query, indexing, query processing methods, etc.

Architecture: Typical Data Mining System



Data Mining:

— Chapter 2 —

Chapter 2: Data Preprocessing

- Why preprocess the data?
- Descriptive data summarization
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Why Data Preprocessing?

- Data in the real world is dirty
 - **incomplete**: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - e.g., occupation=" "
 - **noisy**: containing errors or outliers
 - e.g., Salary="-10"
 - **inconsistent**: containing discrepancies in codes or names
 - e.g., Age="42" Birthday="03/07/1997"
 - e.g., Was rating "1,2,3", now rating "A, B, C"
 - e.g., discrepancy between duplicate records

Why Is Data Dirty?

- Incomplete data may come from
 - “Not applicable” data value when collected
 - Different considerations between the time when the data was collected and when it is analyzed.
 - Human/hardware/software problems
- Noisy data (incorrect values) may come from
 - Faulty data collection instruments
 - Human or computer error at data entry
 - Errors in data transmission
- Inconsistent data may come from
 - Different data sources
 - Functional dependency violation (e.g., modify some linked data)
- Duplicate records also need data cleaning

Why Is Data Preprocessing Important?

- No quality data, no quality mining results!
 - Quality decisions must be based on quality data
 - e.g., duplicate or missing data may cause incorrect or even misleading statistics.
 - Data warehouse needs consistent integration of quality data
- Data extraction, cleaning, and transformation comprises the majority of the work of building a data warehouse

Multi-Dimensional Measure of Data Quality

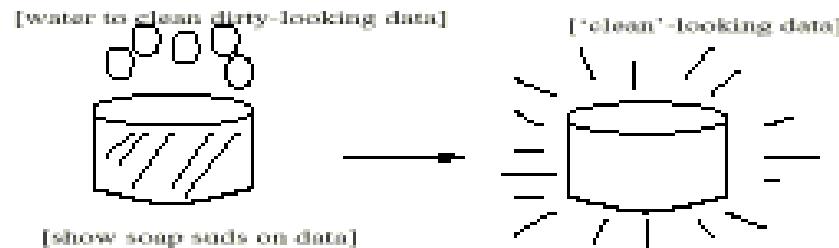
- A well-accepted multidimensional view:
 - Accuracy
 - Completeness
 - Consistency
 - Timeliness
 - Believability
 - Value added
 - Interpretability
 - Accessibility
- Broad categories:
 - Intrinsic, contextual, representational, and accessibility

Major Tasks in Data Preprocessing

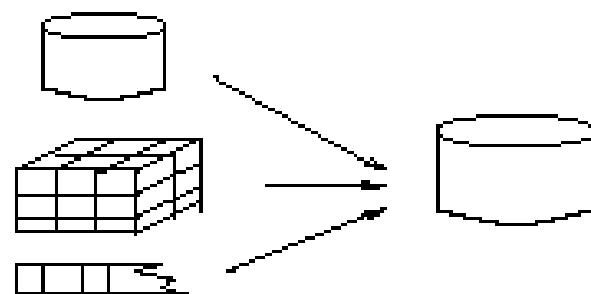
- Data cleaning
 - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
 - Integration of multiple databases, data cubes, or files
- Data transformation
 - Normalization and aggregation
- Data reduction
 - Obtains reduced representation in volume but produces the same or similar analytical results
- Data discretization
 - Part of data reduction but with particular importance, especially for numerical data

Forms of Data Preprocessing

Data Cleaning



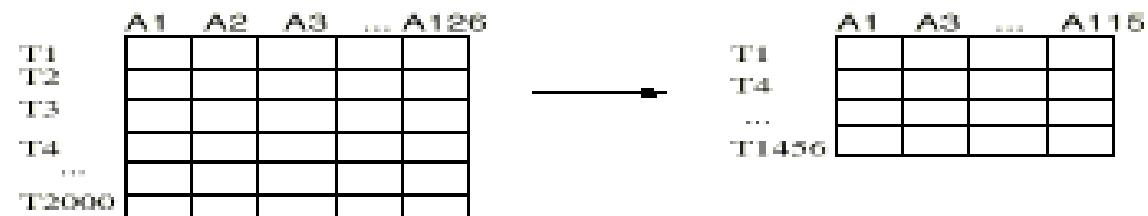
Data Integration



Data Transformation

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

Data Reduction



Chapter 2: Data Preprocessing

- Why preprocess the data?
- Descriptive data summarization
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Mining Data Descriptive Characteristics

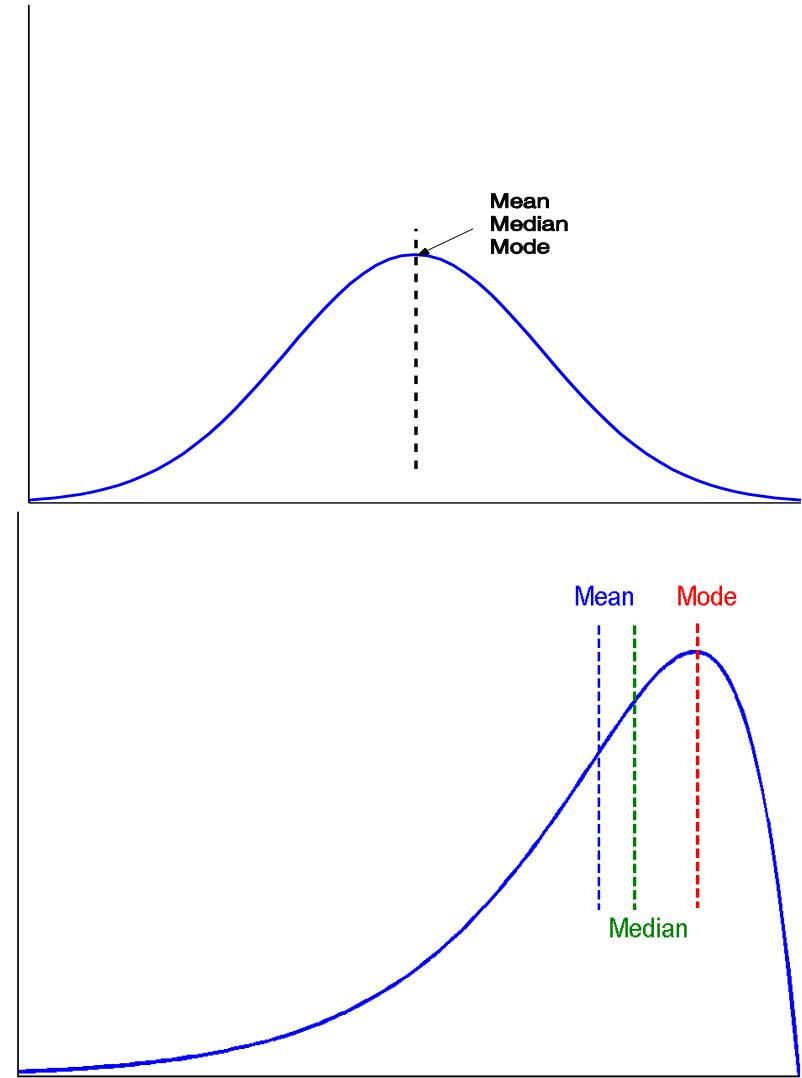
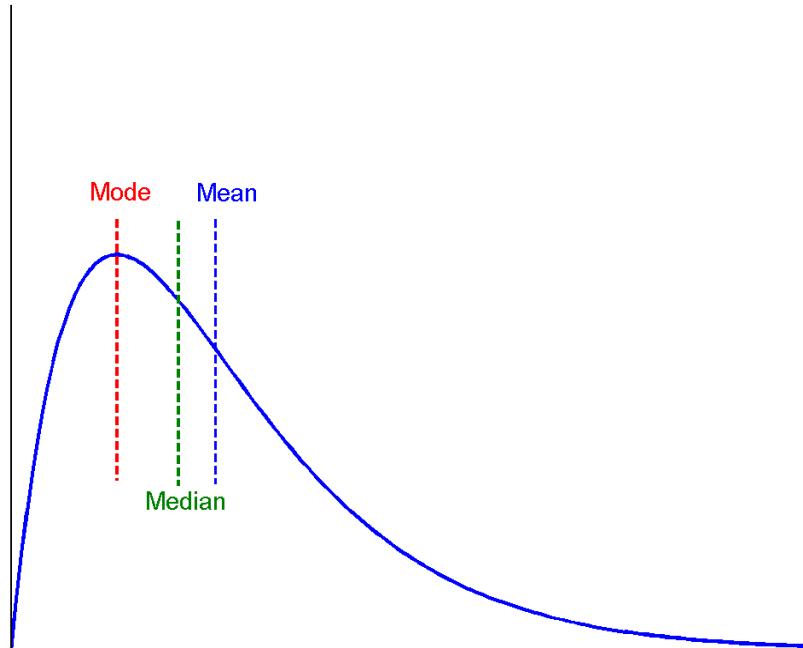
- Motivation
 - To better understand the data: central tendency, variation and spread
- Data dispersion characteristics
 - median, max, min, quantiles, outliers, variance, etc.
- Numerical dimensions correspond to sorted intervals
 - Data dispersion: analyzed with multiple granularities of precision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

Measuring the Central Tendency

- Mean (algebraic measure) (sample vs. population): $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ $\mu = \frac{\sum x}{N}$
 - Weighted arithmetic mean:
 - Trimmed mean: chopping extreme values
- Median: A holistic measure
 - Middle value if odd number of values, or average of the middle two values otherwise
 - Estimated by interpolation (for *grouped data*):
$$\text{median} = L_1 + \left(\frac{n/2 - (\sum f)l}{f_{\text{median}}} \right) c$$
- Mode
 - Value that occurs most frequently in the data
 - Unimodal, bimodal, trimodal
 - Empirical formula:
$$\text{mean} - \text{mode} = 3 \times (\text{mean} - \text{median})$$

Symmetric vs. Skewed Data

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



Measuring the Dispersion of Data

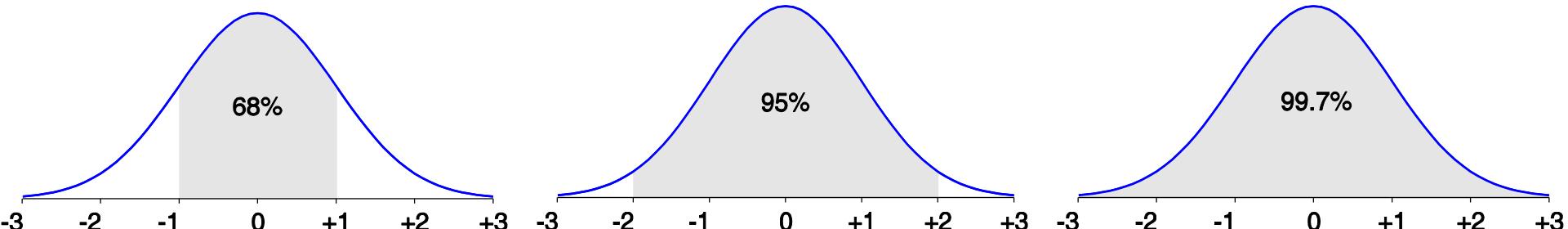
- Quartiles, outliers and boxplots
 - **Quartiles**: Q_1 (25^{th} percentile), Q_3 (75^{th} percentile)
 - **Inter-quartile range**: $\text{IQR} = Q_3 - Q_1$
 - **Five number summary**: min, Q_1 , M, Q_3 , max
 - **Boxplot**: ends of the box are the quartiles, median is marked, whiskers, and plot outlier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

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

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

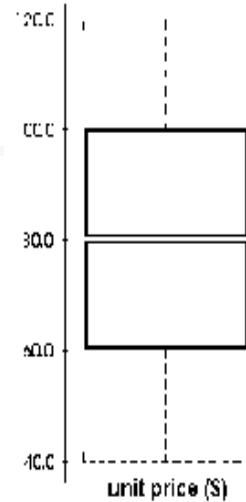
Properties of Normal Distribution Curve

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

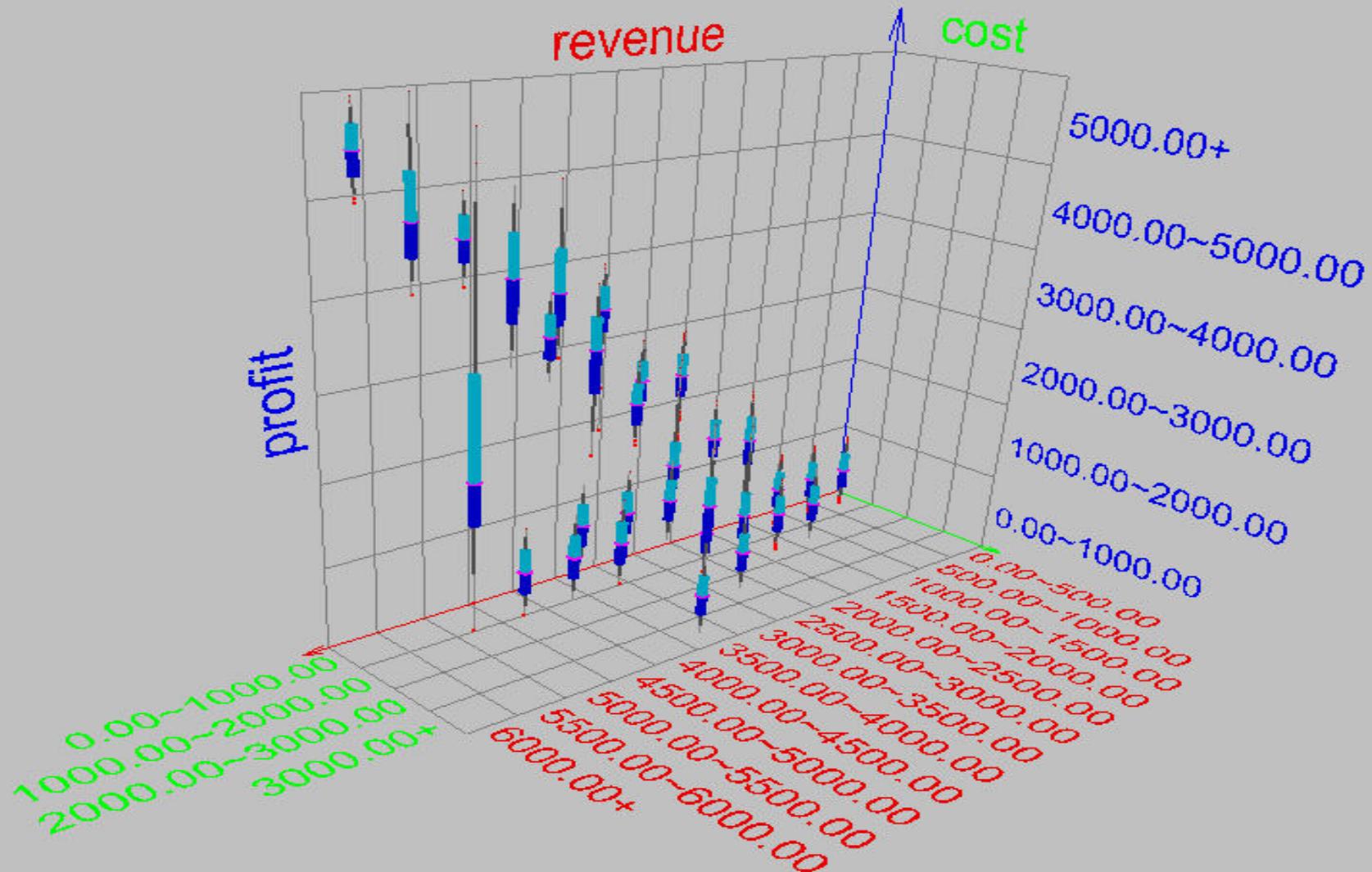


Boxplot Analysis

- Five-number summary of a distribution:
Minimum, Q1, M, 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 IRQ
 - The median is marked by a line within the box
 - Whiskers: two lines outside the box extend to Minimum and Maximum

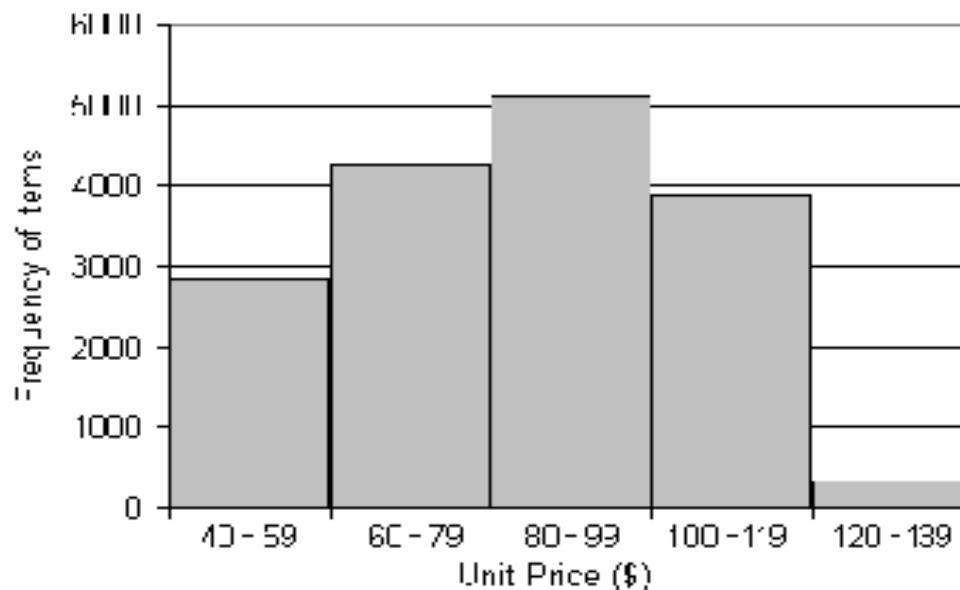


Visualization of Data Dispersion: Boxplot Analysis



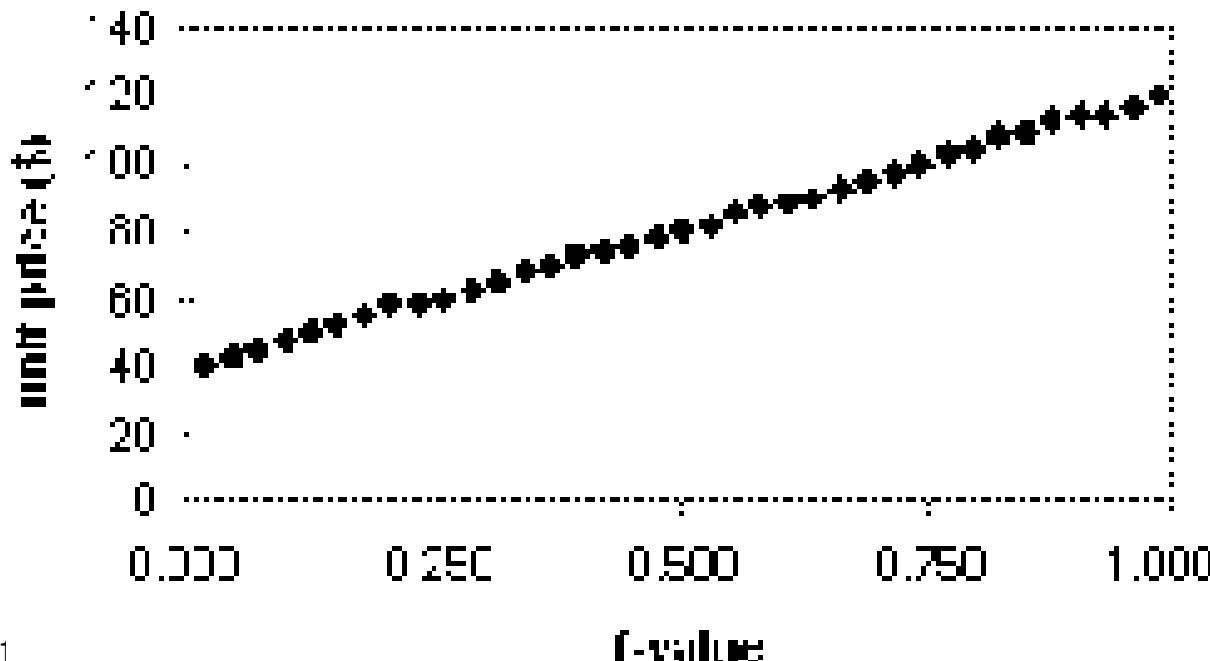
Histogram Analysis

- Graph displays of basic statistical class descriptions
 - Frequency histograms
 - A univariate graphical method
 - Consists of a set of rectangles that reflect the counts or frequencies of the classes present in the given data



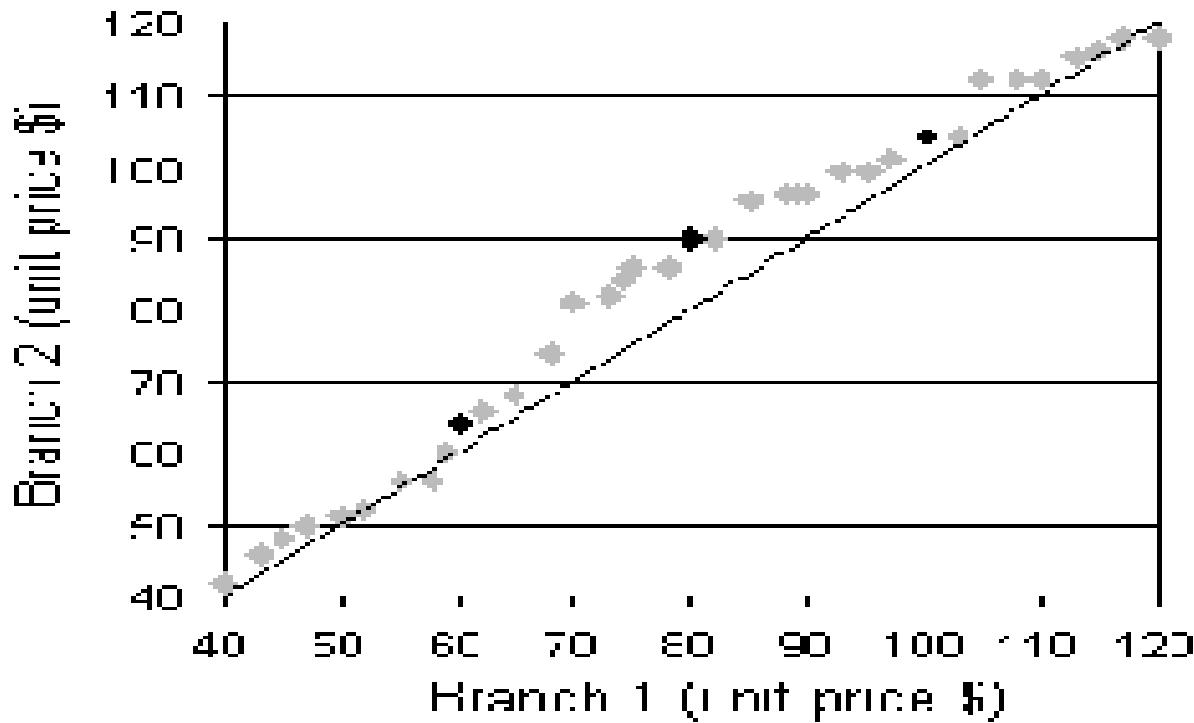
Quantile Plot

- 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



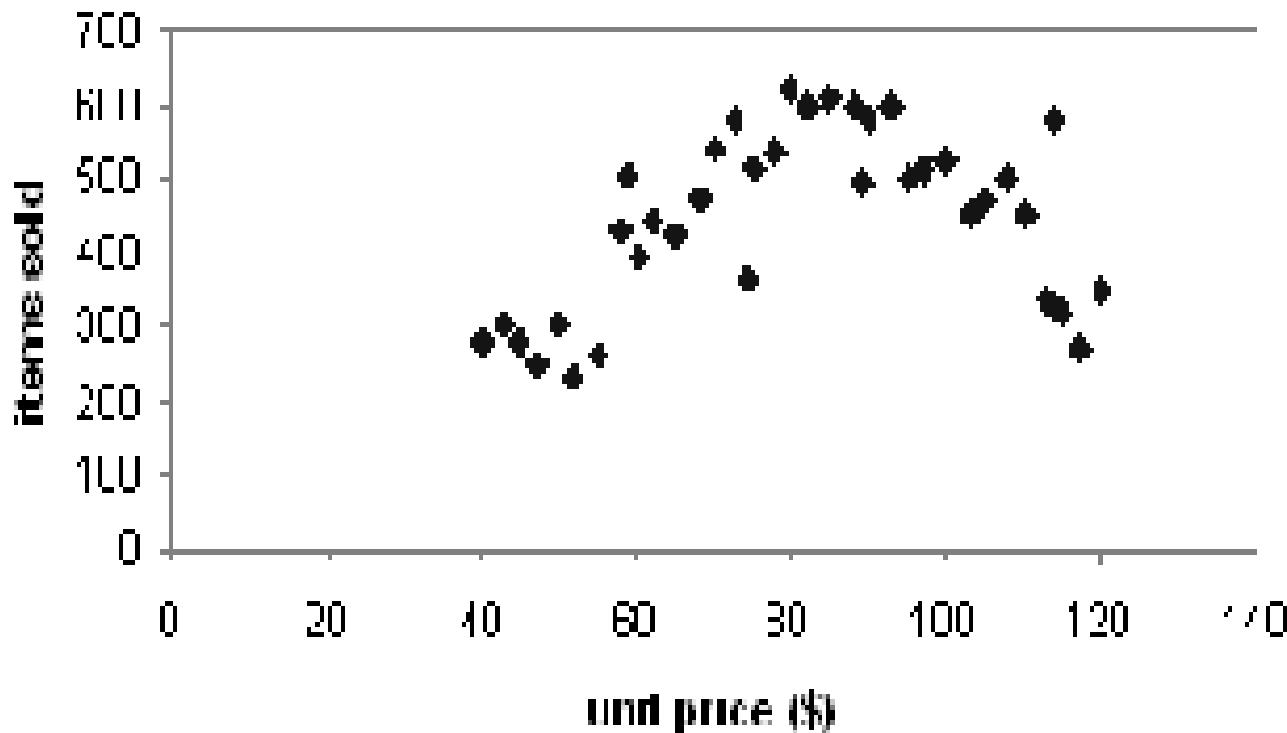
Quantile-Quantile (Q-Q) Plot

- Graphs the quantiles of one univariate distribution against the corresponding quantiles of another
- Allows the user to view whether there is a shift in going from one distribution to another



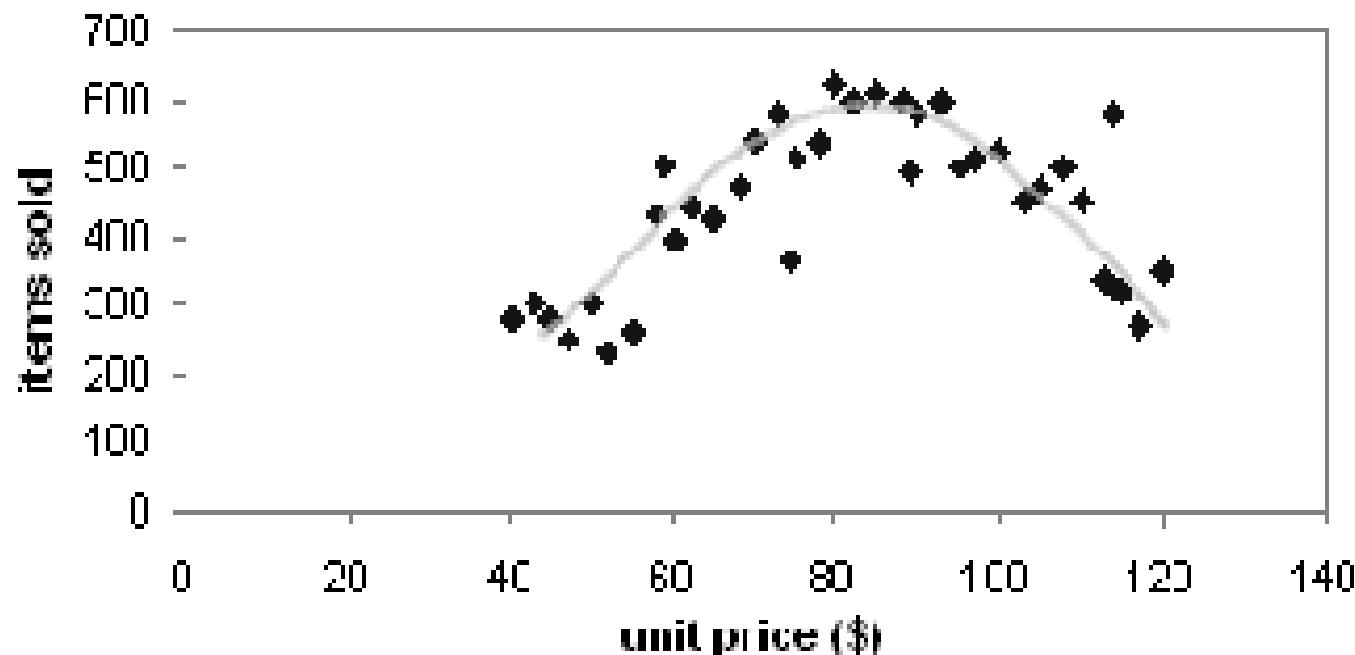
Scatter plot

- 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

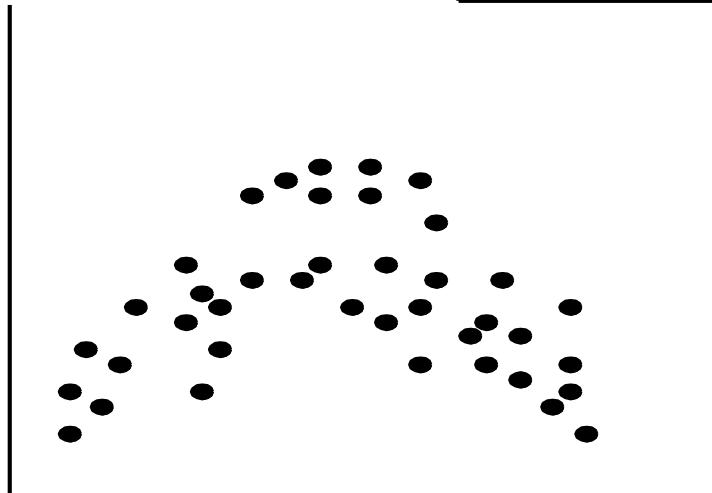
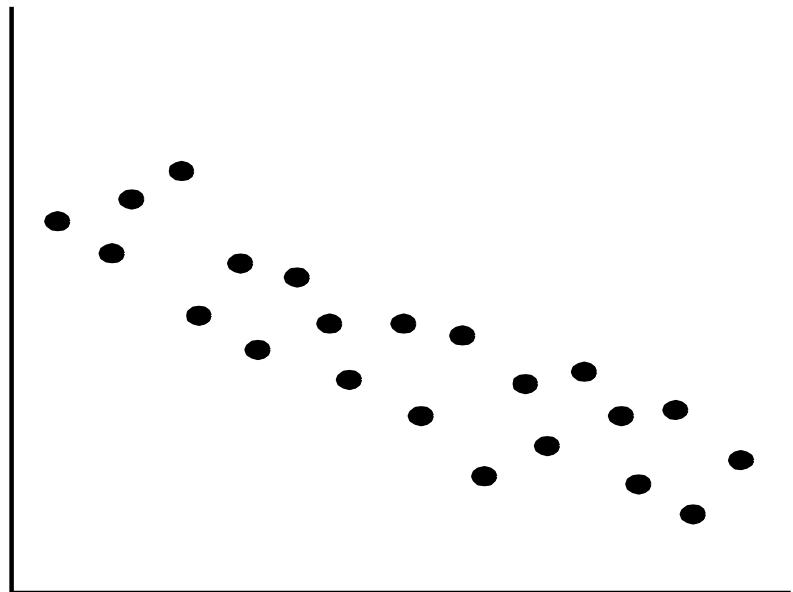
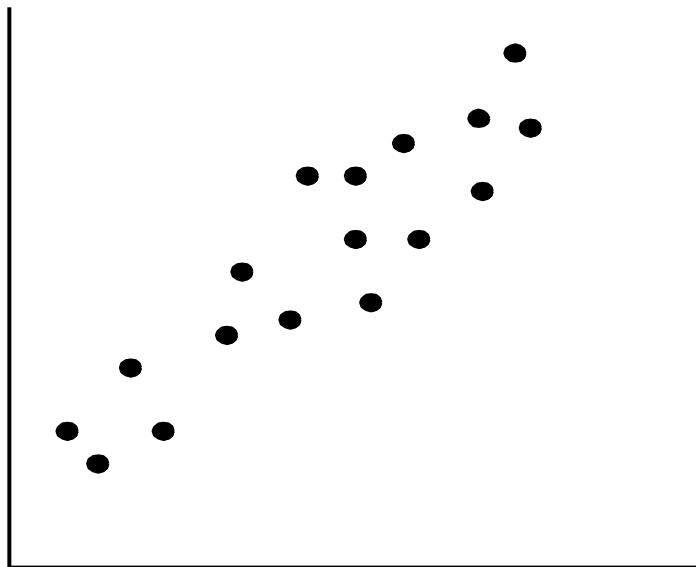


Loess Curve

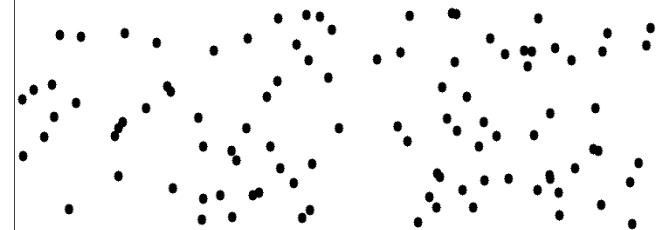
- Adds a smooth curve to a scatter plot in order to provide better perception of the pattern of dependence
- Loess curve is fitted by setting two parameters: a smoothing parameter, and the degree of the polynomials that are fitted by the regression



Positively and Negatively Correlated Data



Not Correlated Data



Graphic Displays of Basic Statistical Descriptions

- Histogram: (shown before)
- Boxplot: (covered before)
- 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 univariate 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
- Loess (local regression) curve: add a smooth curve to a scatter plot to provide better perception of the pattern of dependence

Chapter 2: Data Preprocessing

- Why preprocess the data?
- Descriptive data summarization
- **Data cleaning**
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Data Cleaning

- Importance
 - “Data cleaning is one of the three biggest problems in data warehousing”—Ralph Kimball
 - “Data cleaning is the number one problem in data warehousing”—DCI survey
- Data cleaning tasks
 - Fill in missing values
 - Identify outliers and smooth out noisy data
 - Correct inconsistent data
 - Resolve redundancy caused by data integration

Missing Data

- Data is not always available
 - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
 - equipment malfunction
 - inconsistent with other recorded data and thus deleted
 - data not entered due to misunderstanding
 - certain data may not be considered important at the time of entry
 - not register history or changes of the data
- Missing data may need to be inferred.

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably).
- Fill in the missing value manually: tedious + infeasible?
- Fill in it automatically with
 - a global constant : e.g., “unknown”, a new class?!
 - the attribute mean
 - the attribute mean for all samples belonging to the same class: smarter
 - the most probable value: inference-based such as Bayesian formula or decision tree

Noisy Data

- Noise: random error or variance in a measured variable
- Incorrect attribute values may due to
 - faulty data collection instruments
 - data entry problems
 - data transmission problems
 - technology limitation
 - inconsistency in naming convention
- Other data problems which requires data cleaning
 - duplicate records
 - incomplete data
 - inconsistent data

How to Handle Noisy Data?

- Binning
 - first sort data and partition into (equal-frequency) bins
 - then one can **smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.**
- Regression
 - smooth by fitting the data into regression functions
- Clustering
 - detect and remove outliers
- Combined computer and human inspection
 - detect suspicious values and check by human (e.g., deal with possible outliers)

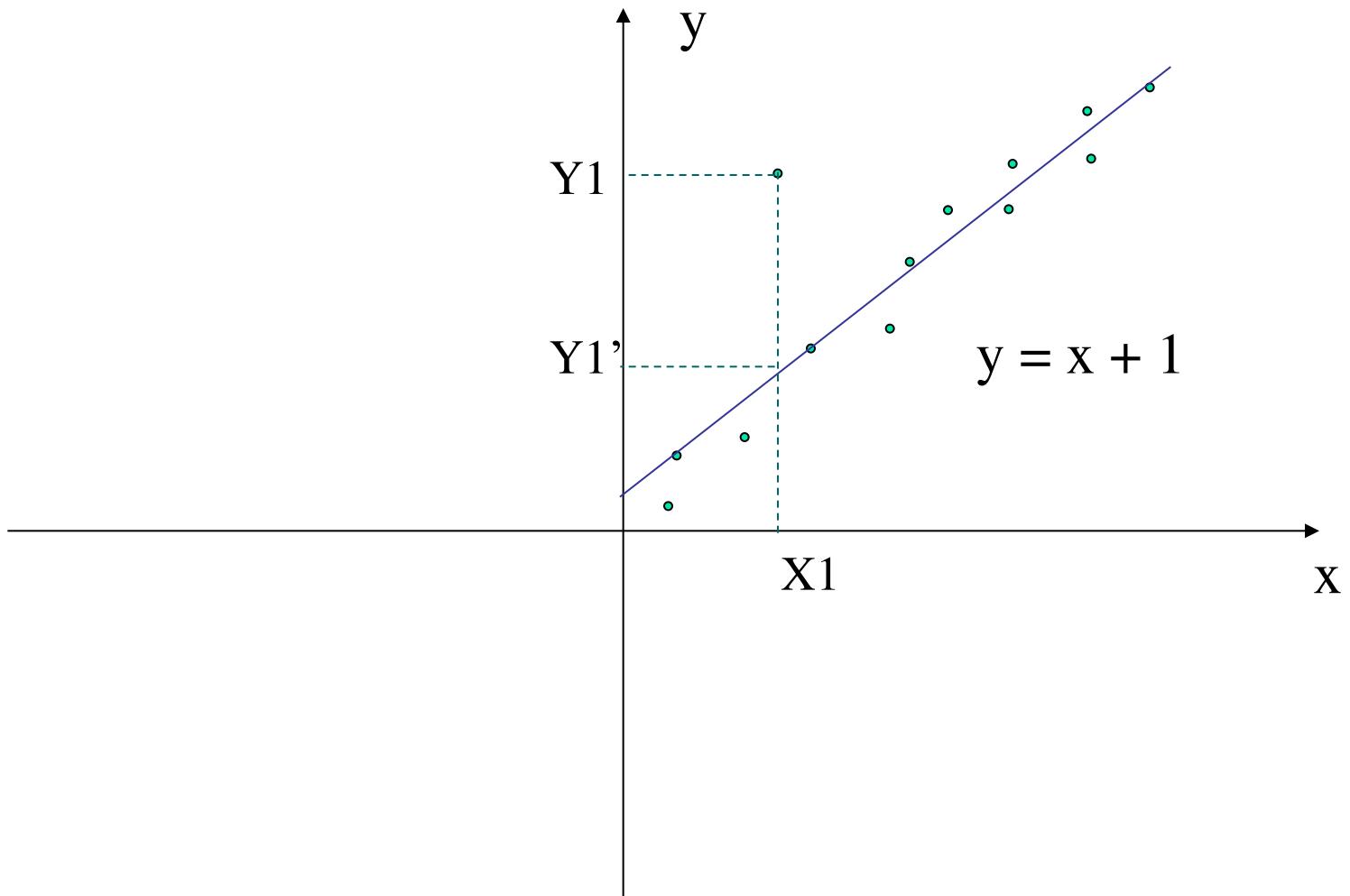
Simple Discretization Methods: Binning

- **Equal-width** (distance) partitioning
 - Divides the range into N intervals of equal size: uniform grid
 - if A and B are the lowest and highest values of the attribute, the width of intervals will be: $W = (B - A)/N$.
 - The most straightforward, but outliers may dominate presentation
 - Skewed data is not handled well
- **Equal-depth** (frequency) partitioning
 - Divides the range into N intervals, each containing approximately same number of samples
 - Good data scaling
 - Managing categorical attributes can be tricky

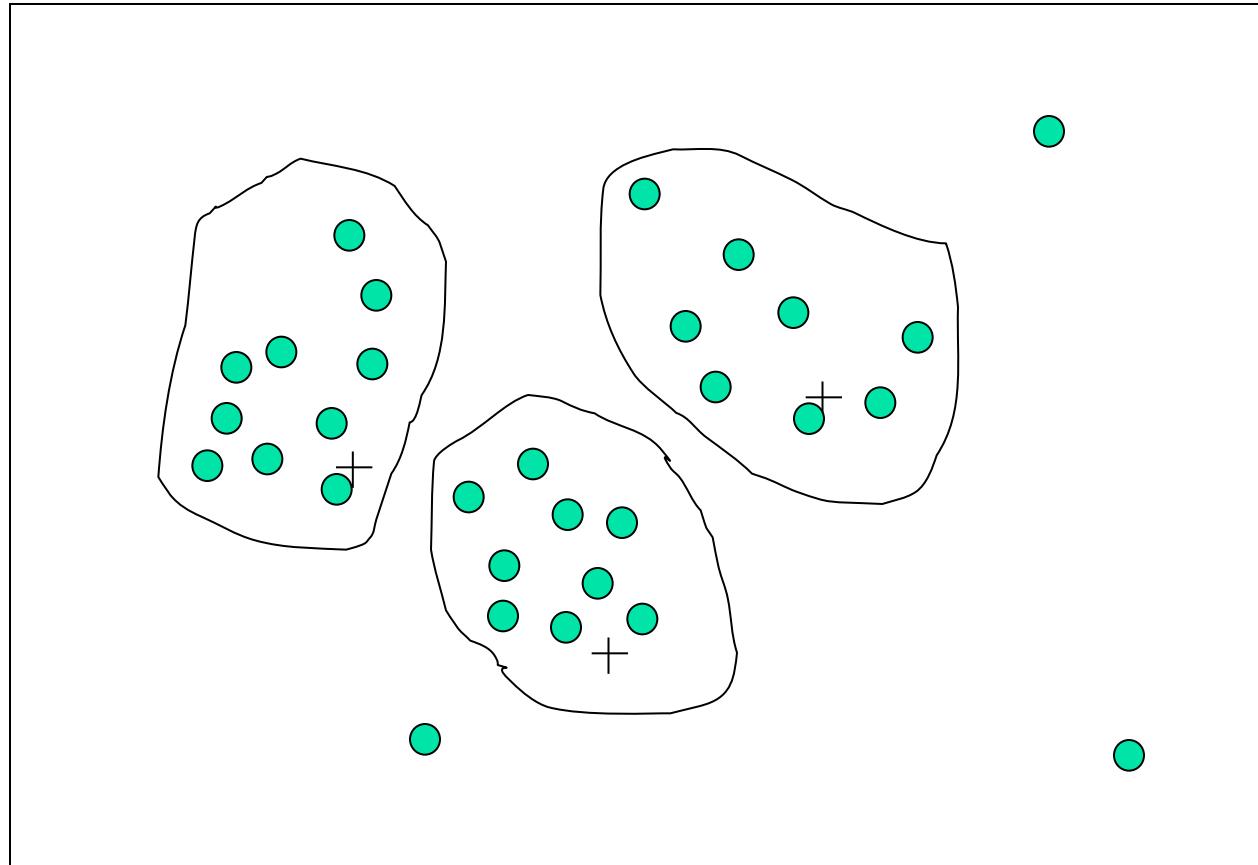
Binning Methods for Data Smoothing

- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34
 - * Partition into equal-frequency (equi-depth) bins:
 - Bin 1: 4, 8, 9, 15
 - Bin 2: 21, 21, 24, 25
 - Bin 3: 26, 28, 29, 34
 - * Smoothing by bin means:
 - Bin 1: 9, 9, 9, 9
 - Bin 2: 23, 23, 23, 23
 - Bin 3: 29, 29, 29, 29
 - * Smoothing by bin boundaries:
 - Bin 1: 4, 4, 4, 15
 - Bin 2: 21, 21, 25, 25
 - Bin 3: 26, 26, 26, 34

Regression



Cluster Analysis



Chapter 2: Data Preprocessing

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Data Integration

- Data integration:
 - Combines data from multiple sources into a coherent store
- Schema integration: e.g., A.cust-id \equiv B.cust-#
 - Integrate metadata from different sources
- Entity identification problem:
 - Identify real world entities from multiple data sources, e.g., Bill Clinton = William Clinton
- Detecting and resolving data value conflicts
 - For the same real world entity, attribute values from different sources are different
 - Possible reasons: different representations, different scales, e.g., metric vs. British units

Handling Redundancy in Data Integration

- Redundant data occur often when integration of multiple databases
 - *Object identification:* The same attribute or object may have different names in different databases
 - *Derivable data:* One attribute may be a “derived” attribute in another table, e.g., annual revenue
- Redundant attributes may be able to be detected by *correlation analysis*
- Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality

Correlation Analysis (Numerical Data)

- Correlation coefficient (also called Pearson's product moment coefficient)

$$r_{A,B} = \frac{\sum (A - \bar{A})(B - \bar{B})}{(n-1)\sigma_A\sigma_B} = \frac{\sum (AB) - n\bar{A}\bar{B}}{(n-1)\sigma_A\sigma_B}$$

where n is the number of tuples, \bar{A} and \bar{B} are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\Sigma(AB)$ is the sum of the AB cross-product.

- If $r_{A,B} > 0$, A and B are positively correlated (A's values increase as B's). The higher, the stronger correlation.
- $r_{A,B} = 0$: independent; $r_{A,B} < 0$: negatively correlated

Correlation Analysis (Categorical Data)

- χ^2 (chi-square) test

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- The larger the χ^2 value, the more likely the variables are related
- The cells that contribute the most to the χ^2 value are those whose actual count is very different from the expected count
- Correlation does not imply causality
 - # of hospitals and # of car-theft in a city are correlated
 - Both are causally linked to the third variable: population

Chi-Square Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum(col.)	300	1200	1500

- χ^2 (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

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

- It shows that like_science_fiction and play_chess are correlated in the group

Data Transformation

- Smoothing: remove noise from data
- Aggregation: summarization, data cube construction
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
 - min-max normalization
 - z-score normalization
 - normalization by decimal scaling
- Attribute/feature construction
 - New attributes constructed from the given ones

Data Transformation: Normalization

- Min-max normalization: to $[new_min_A, new_max_A]$

$$v' = \frac{v - min_A}{max_A - min_A} (new_max_A - new_min_A) + new_min_A$$

- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73,600 - 12,000}{98,000 - 12,000}(1.0 - 0) + 0 = 0.716$
- Z-score normalization (μ : mean, σ : standard deviation):

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

- Ex. Let $\mu = 54,000$, $\sigma = 16,000$. Then $\frac{73,600 - 54,000}{16,000} = 1.225$
- Normalization by decimal scaling

$$v' = \frac{v}{10^j} \quad \text{Where } j \text{ is the smallest integer such that } \text{Max}(|v'|) < 1$$

Chapter 2: Data Preprocessing

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Data Reduction Strategies

- Why data reduction?
 - A database/data warehouse may store terabytes of data
 - Complex data analysis/mining may take a very long time to run on the complete data set
- Data reduction
 - Obtain a reduced representation of the data set that is much smaller in volume but yet produce the same (or almost the same) analytical results
- Data reduction strategies
 - Data cube aggregation:
 - Dimensionality reduction — e.g., remove unimportant attributes
 - Data Compression
 - Numerosity reduction — e.g., fit data into models
 - Discretization and concept hierarchy generation

Data Cube Aggregation

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

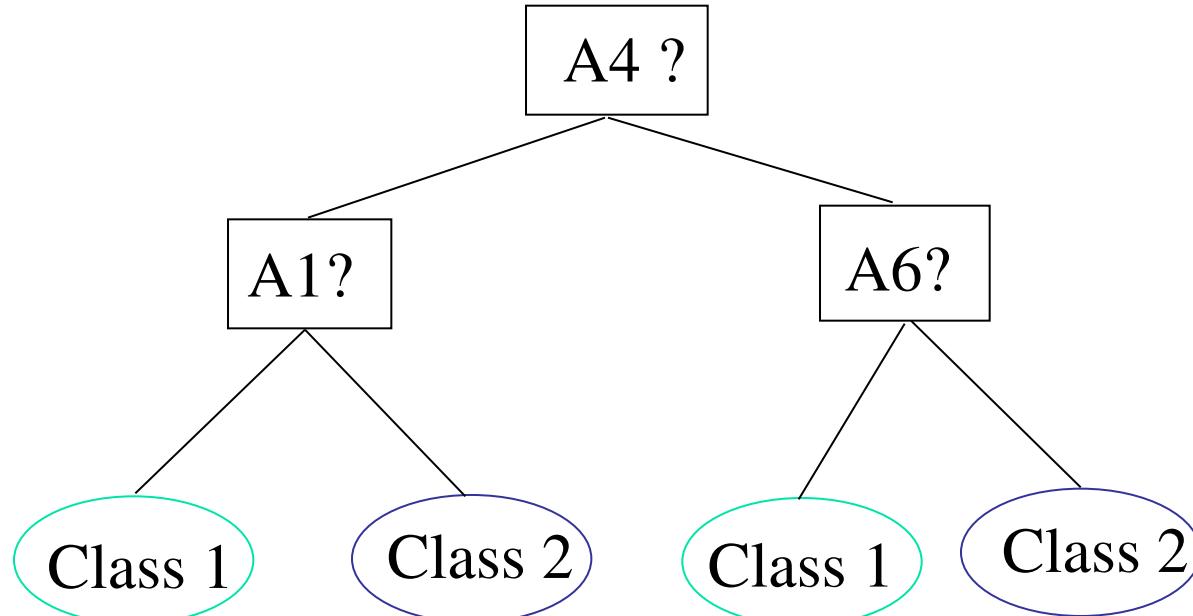
Attribute Subset Selection

- Feature selection (i.e., attribute subset selection):
 - Select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
 - reduce # of patterns in the patterns, easier to understand
- Heuristic methods (due to exponential # of choices):
 - Step-wise forward selection
 - Step-wise backward elimination
 - Combining forward selection and backward elimination
 - Decision-tree induction

Example of Decision Tree Induction

Initial attribute set:

{A1, A2, A3, A4, A5, A6}



-----> Reduced attribute set: {A1, A4, A6}

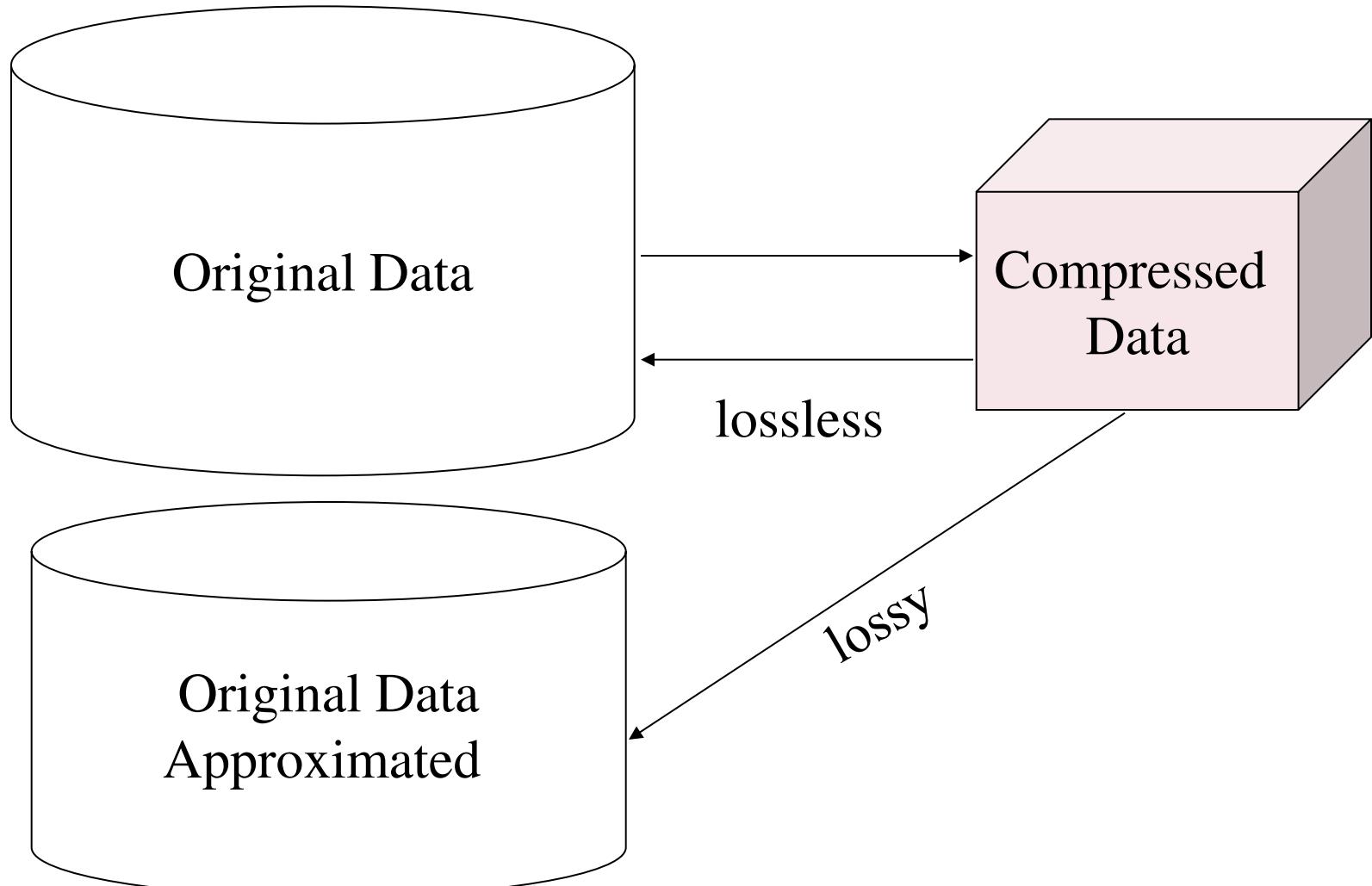
Heuristic Feature Selection Methods

- There are 2^d possible sub-features of d features
- Several heuristic feature selection methods:
 - Best single features under the feature independence assumption: choose by significance tests
 - Best step-wise feature selection:
 - The best single-feature is picked first
 - Then next best feature condition to the first, ...
 - Step-wise feature elimination:
 - Repeatedly eliminate the worst feature
 - Best combined feature selection and elimination
 - Optimal branch and bound:
 - Use feature elimination and backtracking

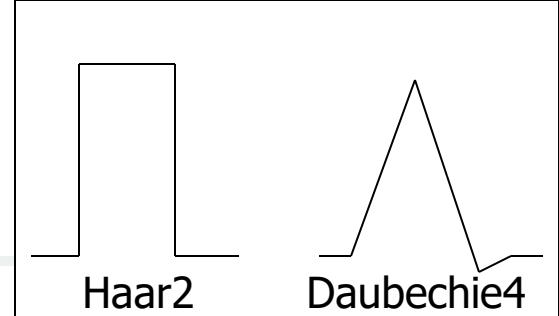
Data Compression

- String compression
 - There are extensive theories and well-tuned algorithms
 - Typically lossless
 - But only limited manipulation is possible without expansion
- Audio/video compression
 - Typically lossy compression, with progressive refinement
 - Sometimes small fragments of signal can be reconstructed without reconstructing the whole
- Time sequence is not audio
 - Typically short and vary slowly with time

Data Compression

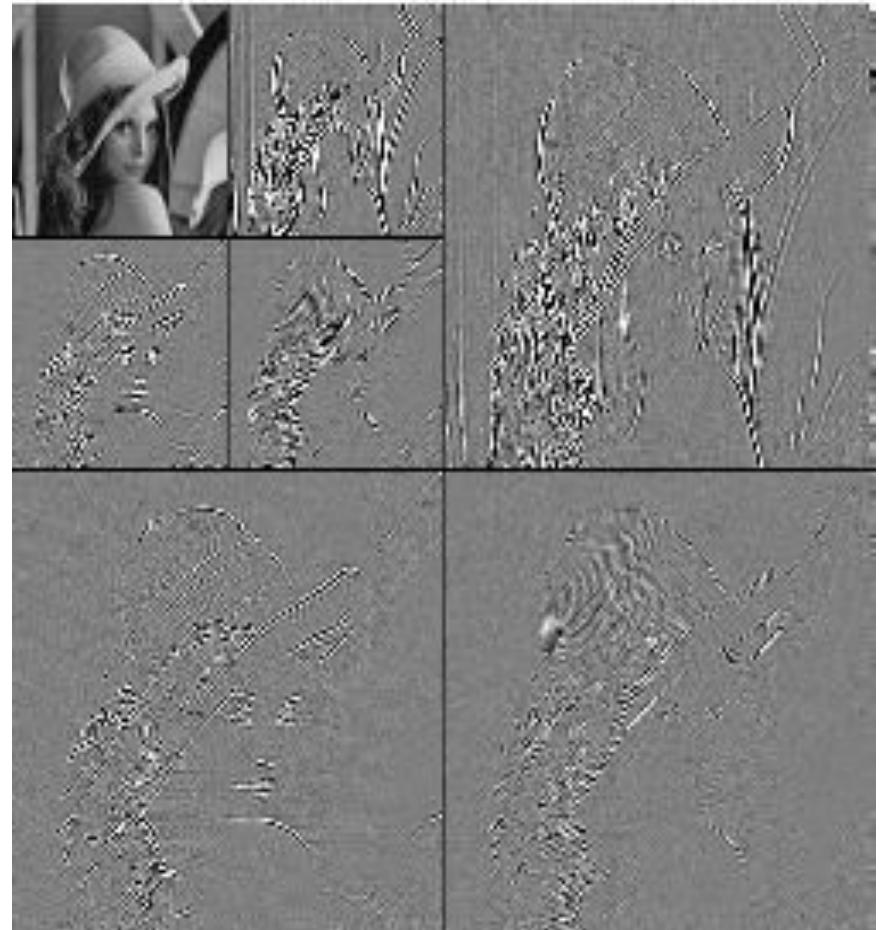
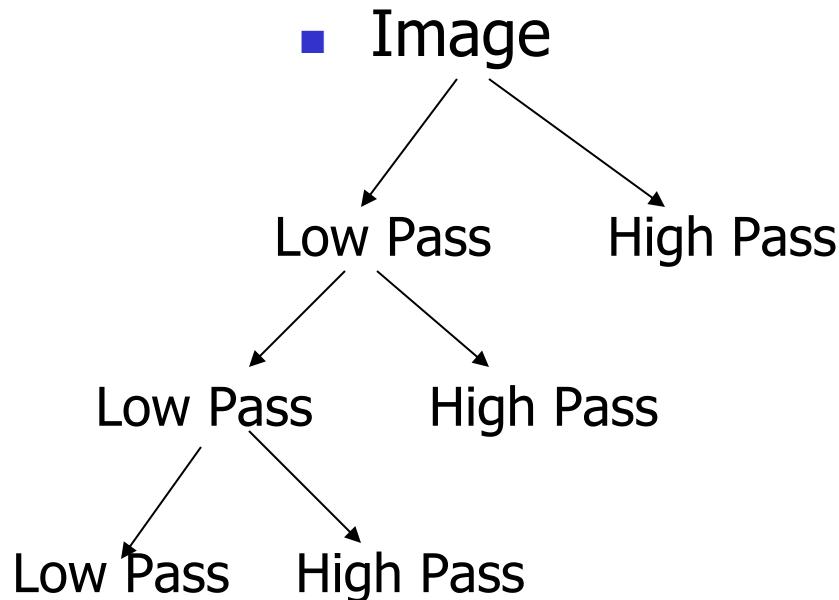


Dimensionality Reduction: Wavelet Transformation



- Discrete wavelet transform (DWT): linear signal processing, multi-resolutonal analysis
- Compressed approximation: store only a small fraction of the strongest of the wavelet coefficients
- Similar to discrete Fourier transform (DFT), but better lossy compression, localized in space
- Method:
 - Length, L , must be an integer power of 2 (padding with 0's, when necessary)
 - Each transform has 2 functions: smoothing, difference
 - Applies to pairs of data, resulting in two set of data of length $L/2$
 - Applies two functions recursively, until reaches the desired length

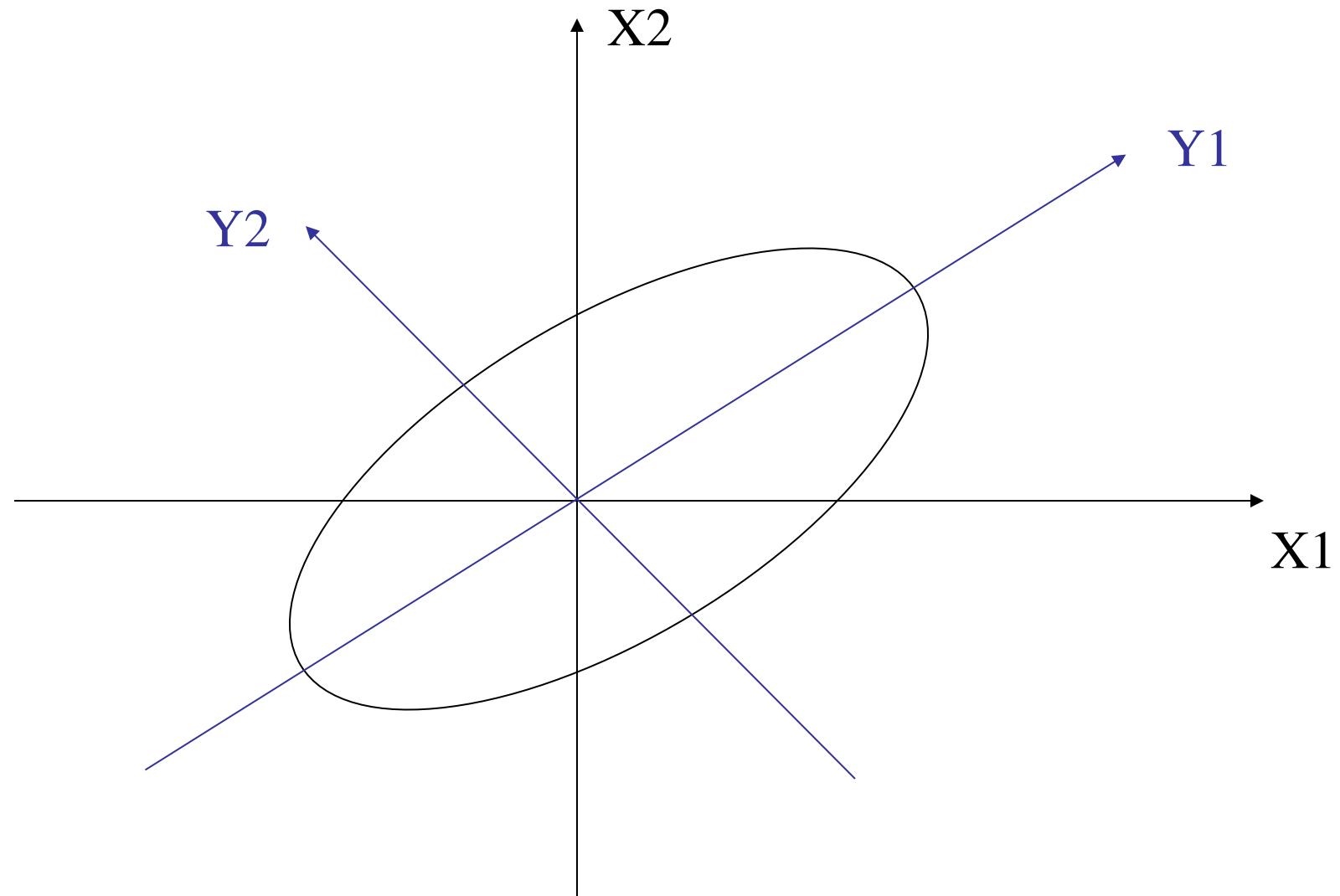
DWT for Image Compression



Dimensionality Reduction: Principal Component Analysis (PCA)

- Given N data vectors from n -dimensions, find $k \leq n$ orthogonal vectors (*principal components*) that can be best used to represent data
- Steps
 - Normalize input data: Each attribute falls within the same range
 - Compute k orthonormal (unit) vectors, i.e., *principal components*
 - Each input data (vector) is a linear combination of the k principal component vectors
 - The principal components are sorted in order of decreasing “significance” or strength
 - Since the components are sorted, the size of the data can be reduced by eliminating the weak components, i.e., those with low variance. (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data
- Works for numeric data only
- Used when the number of dimensions is large

Principal Component Analysis



Numerosity Reduction

- Reduce data volume by choosing alternative, smaller forms of data representation
- Parametric methods
 - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
 - Example: Log-linear models—obtain value at a point in m-D space as the product on appropriate marginal subspaces
- Non-parametric methods
 - Do not assume models
 - Major families: histograms, clustering, sampling

Data Reduction Method (1): Regression and Log-Linear Models

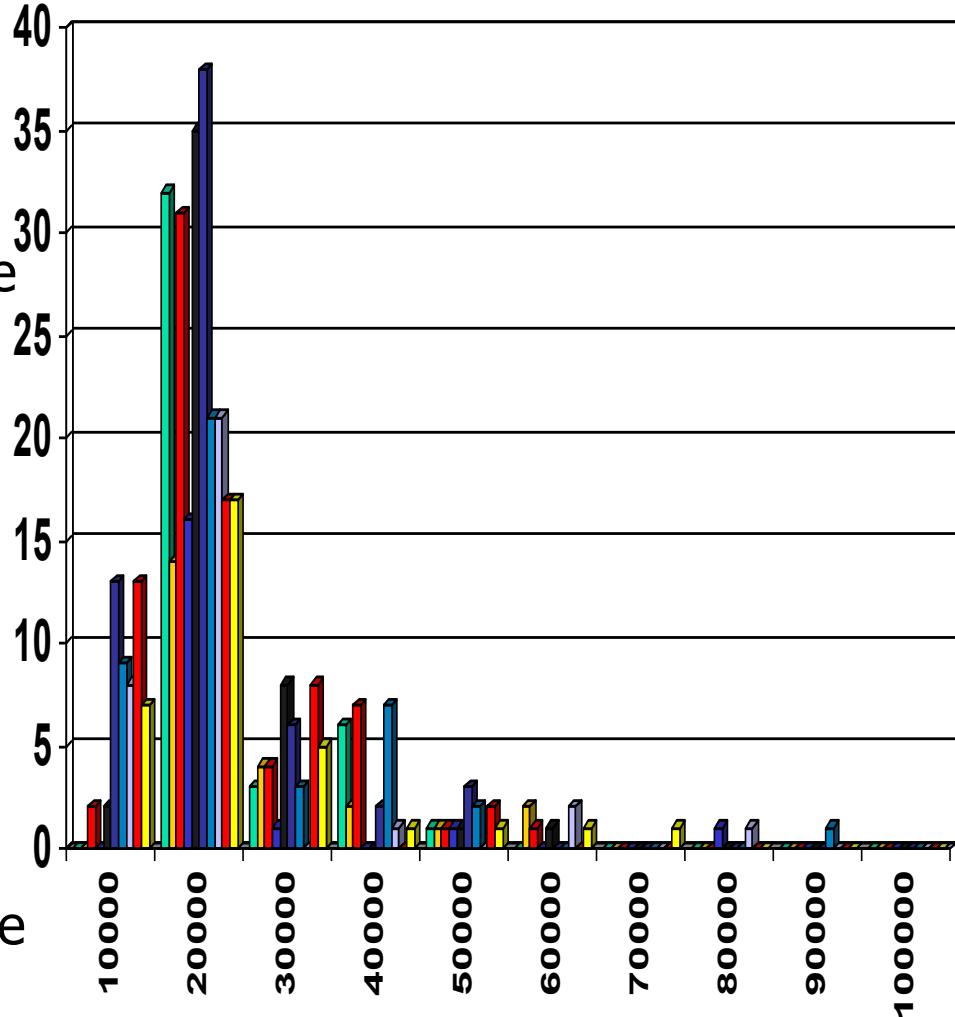
- Linear regression: Data are modeled to fit a straight line
 - Often uses the least-square method to fit the line
- Multiple regression: allows a response variable Y to be modeled as a linear function of multidimensional feature vector
- Log-linear model: approximates discrete multidimensional probability distributions

Regress Analysis and Log-Linear Models

- Linear regression: $Y = wX + b$
 - Two regression coefficients, w and b , specify the line and are to be estimated by using the data at hand
 - Using the least squares criterion to the known values of $Y_1, Y_2, \dots, X_1, X_2, \dots$
- Multiple regression: $Y = b_0 + b_1 X_1 + b_2 X_2$.
 - Many nonlinear functions can be transformed into the above
- Log-linear models:
 - The multi-way table of joint probabilities is approximated by a product of lower-order tables
 - Probability: $p(a, b, c, d) = \alpha_{ab} \beta_{ac} \gamma_{ad} \delta_{bcd}$

Data Reduction Method (2): Histograms

- Divide data into buckets and store average (sum) for each bucket
- Partitioning rules:
 - Equal-width: equal bucket range
 - Equal-frequency (or equal-depth)
 - V-optimal: with the least *histogram variance* (weighted sum of the original values that each bucket represents)
 - MaxDiff: set bucket boundary between each pair for pairs have the $\beta-1$ largest differences



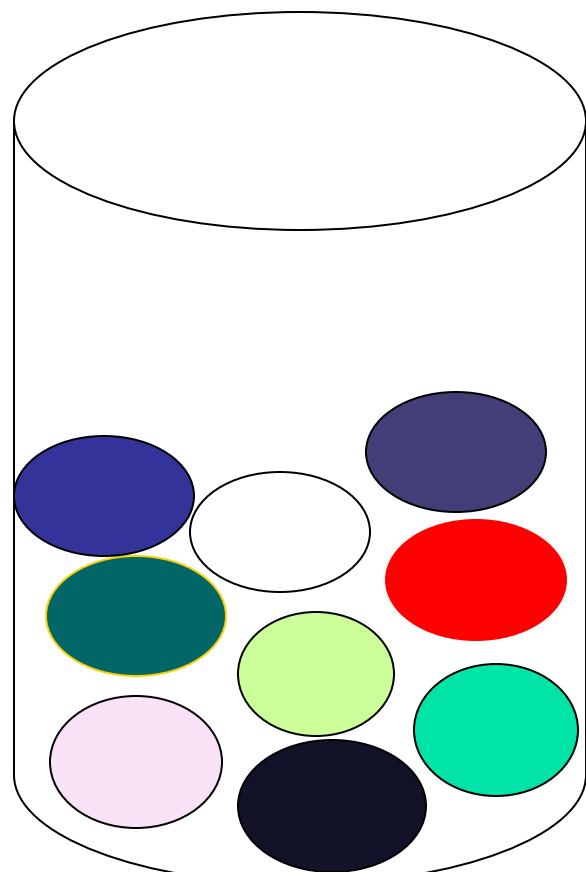
Data Reduction Method (3): Clustering

- Partition data set into clusters based on similarity, and store cluster representation (e.g., centroid and diameter) only
- Can be very effective if data is clustered but not if data is “smeared”
- Can have hierarchical clustering and be stored in multi-dimensional index tree structures
- There are many choices of clustering definitions and clustering algorithms
- Cluster analysis will be studied in depth in Chapter 7

Data Reduction Method (4): Sampling

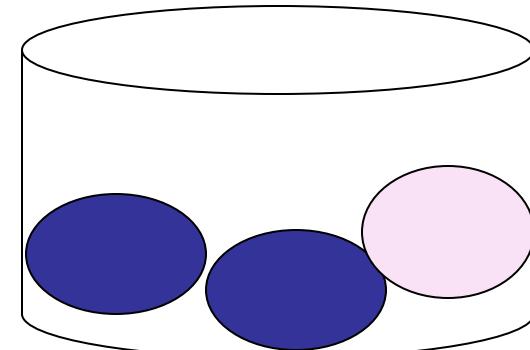
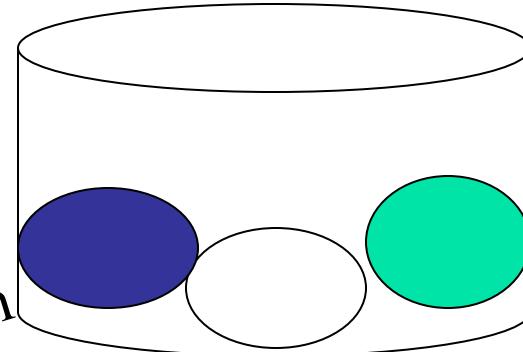
- Sampling: obtaining a small sample s to represent the whole data set N
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data
- Choose a **representative** subset of the data
 - Simple random sampling may have very poor performance in the presence of skew
- Develop adaptive sampling methods
 - Stratified sampling:
 - Approximate the percentage of each class (or subpopulation of interest) in the overall database
 - Used in conjunction with skewed data
- Note: Sampling may not reduce database I/Os (page at a time)

Sampling: with or without Replacement



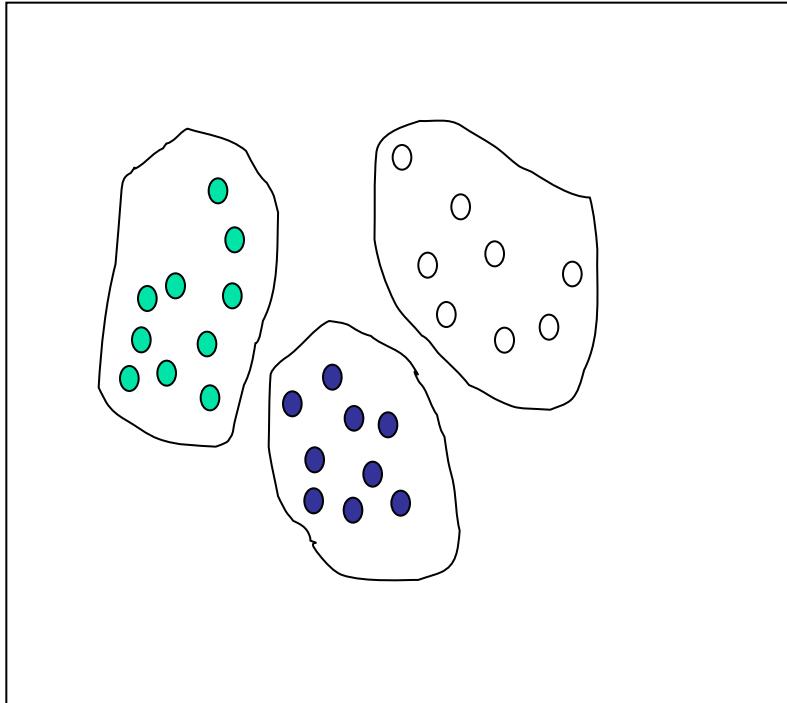
SRSWOR
(simple random
sample without
replacement)

SRSWR

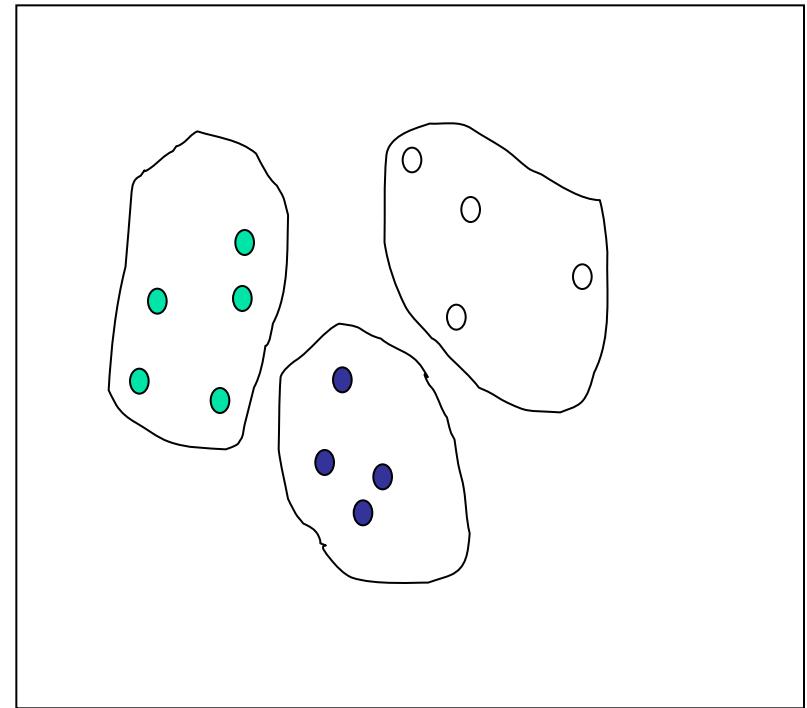


Sampling: Cluster or Stratified Sampling

Raw Data



Cluster/Stratified Sample



Chapter 2: Data Preprocessing

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Discretization

- Three types of attributes:
 - Nominal — values from an unordered set, e.g., color, profession
 - Ordinal — values from an ordered set, e.g., military or academic rank
 - Continuous — real numbers, e.g., integer or real numbers
- Discretization:
 - Divide the range of a continuous attribute into intervals
 - Some classification algorithms only accept categorical attributes.
 - Reduce data size by discretization
 - Prepare for further analysis

Discretization and Concept Hierarchy

- Discretization
 - Reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals
 - Interval labels can then be used to replace actual data values
 - Supervised vs. unsupervised
 - Split (top-down) vs. merge (bottom-up)
 - Discretization can be performed recursively on an attribute
- Concept hierarchy formation
 - Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for age) by higher level concepts (such as young, middle-aged, or senior)

Discretization and Concept Hierarchy Generation for Numeric Data

- Typical methods: All the methods can be applied recursively
 - Binning (covered above)
 - Top-down split, unsupervised,
 - Histogram analysis (covered above)
 - Top-down split, unsupervised
 - Clustering analysis (covered above)
 - Either top-down split or bottom-up merge, unsupervised
 - Entropy-based discretization: supervised, top-down split
 - Interval merging by χ^2 Analysis: unsupervised, bottom-up merge
 - Segmentation by natural partitioning: top-down split, unsupervised

Entropy-Based Discretization

- Given a set of samples S , if S is partitioned into two intervals S_1 and S_2 using boundary T , the information gain after partitioning is

$$I(S, T) = \frac{|S_1|}{|S|} \text{Entropy}(S_1) + \frac{|S_2|}{|S|} \text{Entropy}(S_2)$$

- Entropy is calculated based on class distribution of the samples in the set. Given m classes, the entropy of S_1 is

$$\text{Entropy}(S_1) = -\sum_{i=1}^m p_i \log_2(p_i)$$

where p_i is the probability of class i in S_1

- The boundary that minimizes the entropy function over all possible boundaries is selected as a binary discretization
- The process is recursively applied to partitions obtained until some stopping criterion is met
- Such a boundary may reduce data size and improve classification accuracy

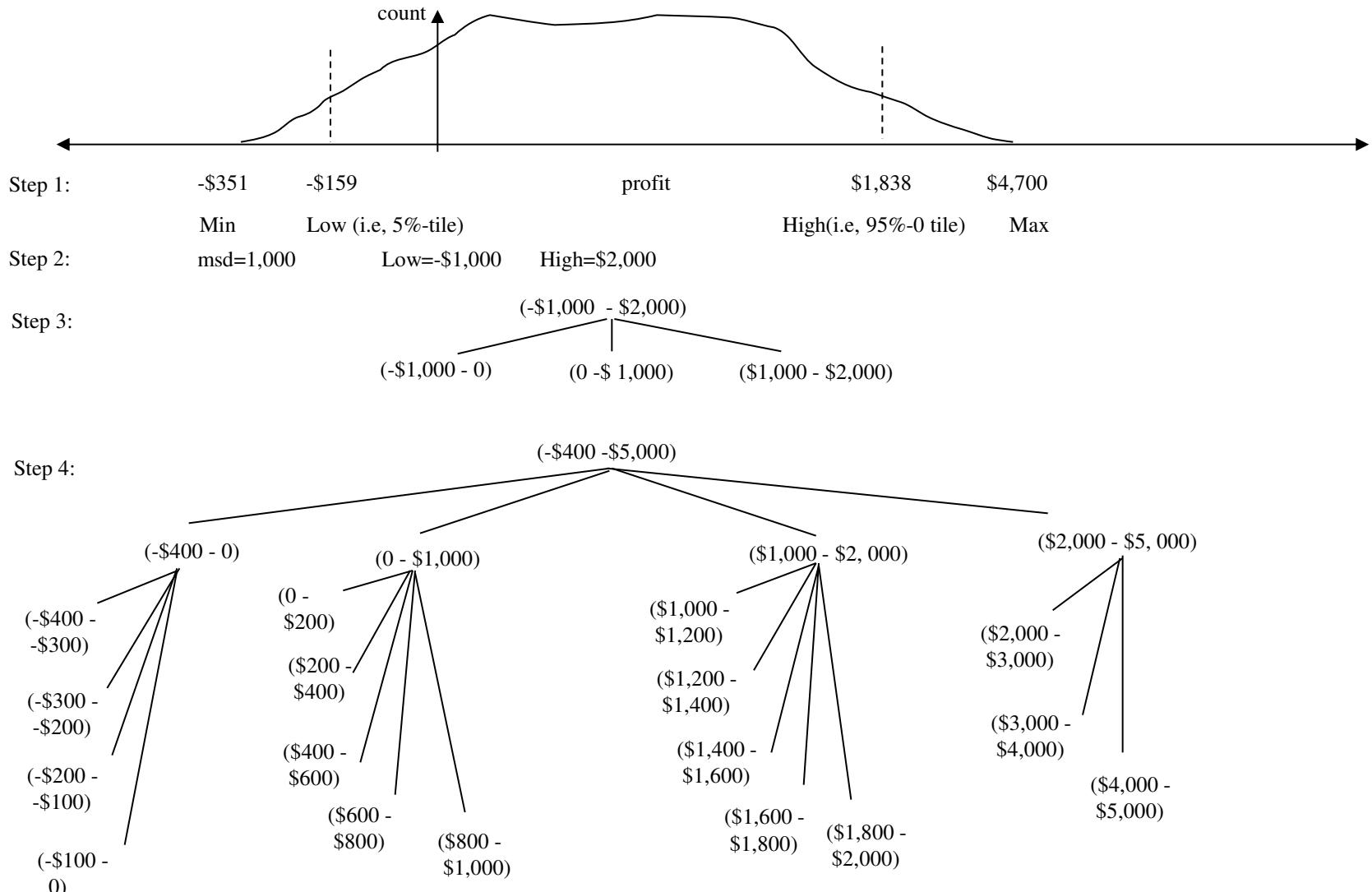
Interval Merge by χ^2 Analysis

- Merging-based (bottom-up) vs. splitting-based methods
- Merge: Find the best neighboring intervals and merge them to form larger intervals recursively
- ChiMerge [Kerber AAAI 1992, See also Liu et al. DMKD 2002]
 - Initially, each distinct value of a numerical attr. A is considered to be one interval
 - χ^2 tests are performed for every pair of adjacent intervals
 - Adjacent intervals with the least χ^2 values are merged together, since low χ^2 values for a pair indicate similar class distributions
 - This merge process proceeds recursively until a predefined stopping criterion is met (such as significance level, max-interval, max inconsistency, etc.)

Segmentation by Natural Partitioning

- A simply 3-4-5 rule can be used to segment numeric data into relatively uniform, “natural” intervals.
 - If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equi-width intervals
 - If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 intervals
 - If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 intervals

Example of 3-4-5 Rule



Concept Hierarchy Generation for Categorical Data

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

Automatic Concept Hierarchy Generation

- Some hierarchies can be automatically generated based on the analysis of the number of distinct values per attribute in the data set
 - The attribute with the most distinct values is placed at the lowest level of the hierarchy
 - Exceptions, e.g., weekday, month, quarter, year



Chapter 2: Data Preprocessing

- Why preprocess the data?
- Data cleaning
- Data integration and transformation
- Data reduction
- Discretization and concept hierarchy generation
- Summary

Summary

- Data preparation or preprocessing is a big issue for both data warehousing and data mining
- Descriptive data summarization is need for quality data preprocessing
- Data preparation includes
 - Data cleaning and data integration
 - Data reduction and feature selection
 - Discretization
- A lot a methods have been developed but data preprocessing still an active area of research

References

- D. P. Ballou and G. K. Tayi. Enhancing data quality in data warehouse environments. Communications of ACM, 42:73-78, 1999
- T. Dasu and T. Johnson. Exploratory Data Mining and Data Cleaning. John Wiley & Sons, 2003
- T. Dasu, T. Johnson, S. Muthukrishnan, V. Shkapenyuk. Mining Database Structure; Or, How to Build a Data Quality Browser. SIGMOD'02.
- H.V. Jagadish et al., Special Issue on Data Reduction Techniques. Bulletin of the Technical Committee on Data Engineering, 20(4), December 1997
- D. Pyle. Data Preparation for Data Mining. Morgan Kaufmann, 1999
- E. Rahm and H. H. Do. Data Cleaning: Problems and Current Approaches. *IEEE Bulletin of the Technical Committee on Data Engineering*. Vol.23, No.4
- V. Raman and J. Hellerstein. Potters Wheel: An Interactive Framework for Data Cleaning and Transformation, VLDB'2001
- T. Redman. Data Quality: Management and Technology. Bantam Books, 1992
- Y. Wand and R. Wang. Anchoring data quality dimensions ontological foundations. Communications of ACM, 39:86-95, 1996
- R. Wang, V. Storey, and C. Firth. A framework for analysis of data quality research. IEEE Trans. Knowledge and Data Engineering, 7:623-640, 1995

Data Mining: Concepts and Techniques

— Chapter 5 —

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map 
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map 
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

What Is Frequent Pattern Analysis?

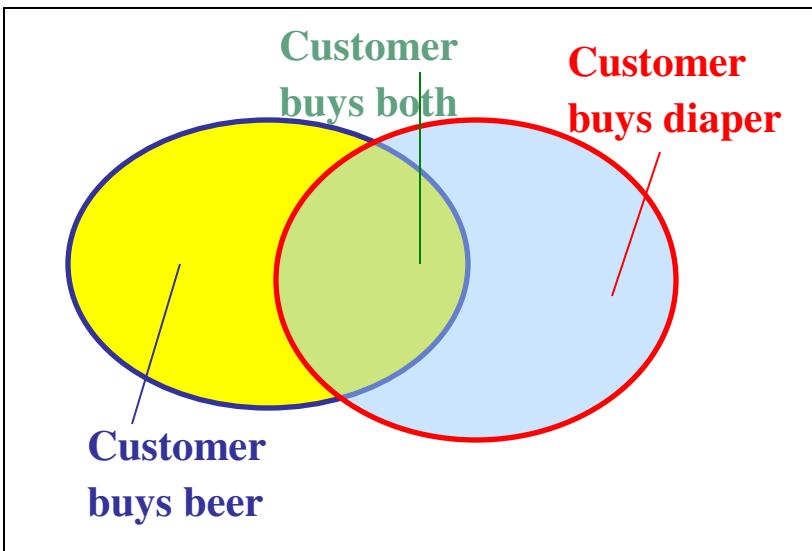
- **Frequent pattern**: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of **frequent itemsets** and **association rule mining**
- Motivation: Finding inherent regularities in data
 - What products were often purchased together?— Beer and diapers?!
 - What are the subsequent purchases after buying a PC?
 - What kinds of DNA are sensitive to this new drug?
 - Can we automatically classify web documents?
- Applications
 - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

Why Is Freq. Pattern Mining Important?

- Discloses an intrinsic and important property of data sets
- Forms the foundation for many essential data mining tasks
 - Association, correlation, and causality analysis
 - Sequential, structural (e.g., sub-graph) patterns
 - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
 - Classification: associative classification
 - Cluster analysis: frequent pattern-based clustering
 - Data warehousing: iceberg cube and cube-gradient
 - Semantic data compression: fascicles
 - Broad applications

Basic Concepts: Frequent Patterns and Association Rules

Transaction-id	Items bought
10	A, B, D
20	A, C, D
30	A, D, E
40	B, E, F
50	B, C, D, E, F



- Itemset $X = \{x_1, \dots, x_k\}$
- Find all the rules $X \rightarrow Y$ with minimum support and confidence
 - **support**, s , probability that a transaction contains $X \cup Y$
 - **confidence**, c , conditional probability that a transaction having X also contains Y

Let $sup_{min} = 50\%$, $conf_{min} = 50\%$

Freq. Pat.: $\{A:3, B:3, D:4, E:3, AD:3\}$

Association rules:

$A \rightarrow D$ (60%, 100%)

$D \rightarrow A$ (60%, 75%)

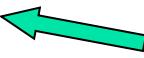
Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \dots, a_{100}\}$ contains $(_{100}^1) + (_{100}^2) + \dots + (_{100}^{100}) = 2^{100} - 1 = 1.27*10^{30}$ 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 ⊃ X, with the same support as X* (proposed by Pasquier, et al. @ ICDT'99)
- An itemset X is a **max-pattern** if X is frequent and there exists no frequent super-pattern Y ⊃ X (proposed by Bayardo @ SIGMOD'98)
- Closed pattern is a lossless compression of freq. patterns
 - Reducing the # of patterns and rules

Closed Patterns and Max-Patterns

- Exercise. $DB = \{\langle a_1, \dots, a_{100} \rangle, \langle a_1, \dots, a_{50} \rangle\}$
 - $\text{Min_sup} = 1$.
- What is the set of **closed itemset**?
 - $\langle a_1, \dots, a_{100} \rangle$: 1
 - $\langle a_1, \dots, a_{50} \rangle$: 2
- What is the set of **max-pattern**?
 - $\langle a_1, \dots, a_{100} \rangle$: 1
- What is the set of **all patterns**?
 - !!

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods 
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

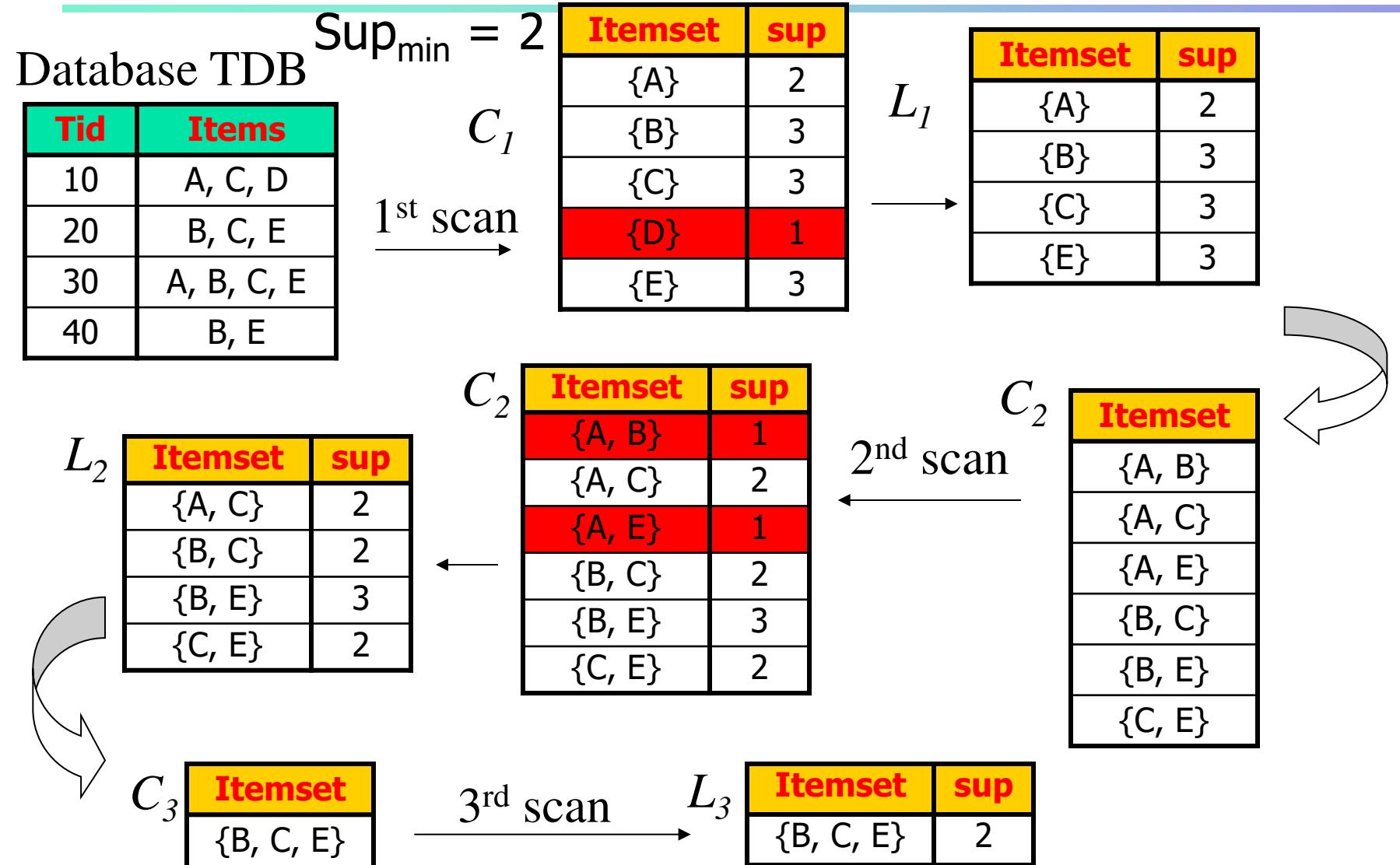
Scalable Methods for Mining Frequent Patterns

- The **downward closure** property of frequent patterns
 - Any subset of a frequent itemset must be frequent
 - If $\{\text{rice, Bread, Butter}\}$ is frequent, so is $\{\text{rice, Bread}\}$
 - i.e., every transaction having $\{\text{rice, Bread, Butter}\}$ also contains $\{\text{rice, Bread}\}$
- Scalable mining methods: Three major approaches
 - Apriori (Agrawal & Srikant@VLDB'94)
 - Freq. pattern growth (FPgrowth—Han, Pei & Yin @SIGMOD'00)
 - Vertical data format approach (Charm—Zaki & Hsiao @SDM'02)

Apriori: A Candidate Generation-and-Test Approach

- Apriori pruning principle: If there is **any** itemset which is infrequent, its superset should not be generated/tested!
(Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)
- Method:
 - Initially, scan DB once to get frequent 1-itemset
 - **Generate** length $(k+1)$ **candidate** itemsets from length k **frequent** itemsets
 - **Test** the candidates against DB
 - Terminate when no frequent or candidate set can be generated

The Apriori Algorithm—An Example



The Apriori Algorithm

- Pseudo-code:

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

$L_1 = \{\text{frequent items}\};$

for ($k = 1; L_k \neq \emptyset; k++$) **do begin**

C_{k+1} = candidates generated from L_k ;

for each transaction t in database do

 increment the count of all candidates in C_{k+1}

 that are contained in t

L_{k+1} = candidates in C_{k+1} with min_support

end

return $\cup_k L_k$;

Important Details of Apriori

- How to generate candidates?
 - Step 1: self-joining L_k
 - Step 2: pruning
- How to count supports of candidates?
- Example of Candidate-generation
 - $L_3 = \{abc, abd, acd, ace, bcd\}$
 - Self-joining: $L_3 * L_3$
 - $abcd$ from abc and abd
 - $acde$ from acd and ace
 - Pruning:
 - $acde$ is removed because ade is not in L_3
 - $C_4 = \{abcd\}$

How to Generate Candidates?

- Suppose the items in L_{k-1} are listed in an order
- Step 1: self-joining L_{k-1}
 - insert into C_k
 - select $p.item_1, p.item_2, \dots, p.item_{k-1}, q.item_{k-1}$
 - from $L_{k-1} p, L_{k-1} q$
 - where $p.item_1=q.item_1, \dots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$
- Step 2: pruning
 - forall **itemsets** c in C_k do
 - forall **($k-1$)-subsets** s of c do
 - if** (s is not in L_{k-1}) **then delete** c from C_k

Efficient Implementation of Apriori in SQL

- Hard to get good performance out of pure SQL (SQL-92) based approaches alone
- Make use of object-relational extensions like UDFs, BLOBs, Table functions etc.
 - Get orders of magnitude improvement
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. In SIGMOD'98

Challenges of Frequent Pattern Mining

- Challenges
 - Multiple scans of transaction database
 - Huge number of candidates
 - Tedium workload of support counting for candidates
- Improving Apriori: general ideas
 - Reduce passes of transaction database scans
 - Shrink number of candidates
 - Facilitate support counting of candidates

Partition: Scan Database Only Twice

- Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB
 - Scan 1: partition database and find local frequent patterns
 - Scan 2: consolidate global frequent patterns
- A. Savasere, E. Omiecinski, and S. Navathe. [An efficient algorithm for mining association in large databases](#). In *VLDB'95*

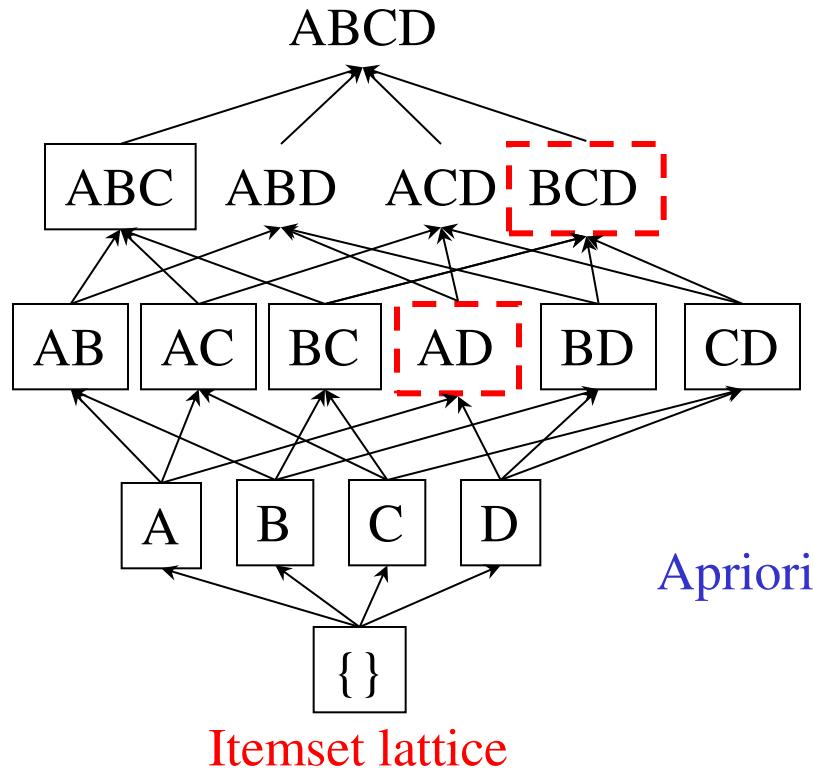
DHP: Reduce the Number of Candidates

- A k -itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
 - Candidates: a, b, c, d, e
 - Hash entries: {ab, ad, ae} {bd, be, de} ...
 - Frequent 1-itemset: a, b, d, e
 - ab is not a candidate 2-itemset if the sum of count of {ab, ad, ae} is below support threshold
- J. Park, M. Chen, and P. Yu. An effective hash-based algorithm for mining association rules. In *SIGMOD'95*

Sampling for Frequent Patterns

- Select a sample of original database, mine frequent patterns within sample using Apriori
- Scan database once to verify frequent itemsets found in sample, only *borders* of closure of frequent patterns are checked
 - Example: check *abcd* instead of *ab*, *ac*, ..., etc.
- Scan database again to find missed frequent patterns
- H. Toivonen. Sampling large databases for association rules. In *VLDB'96*

DIC: Reduce Number of Scans

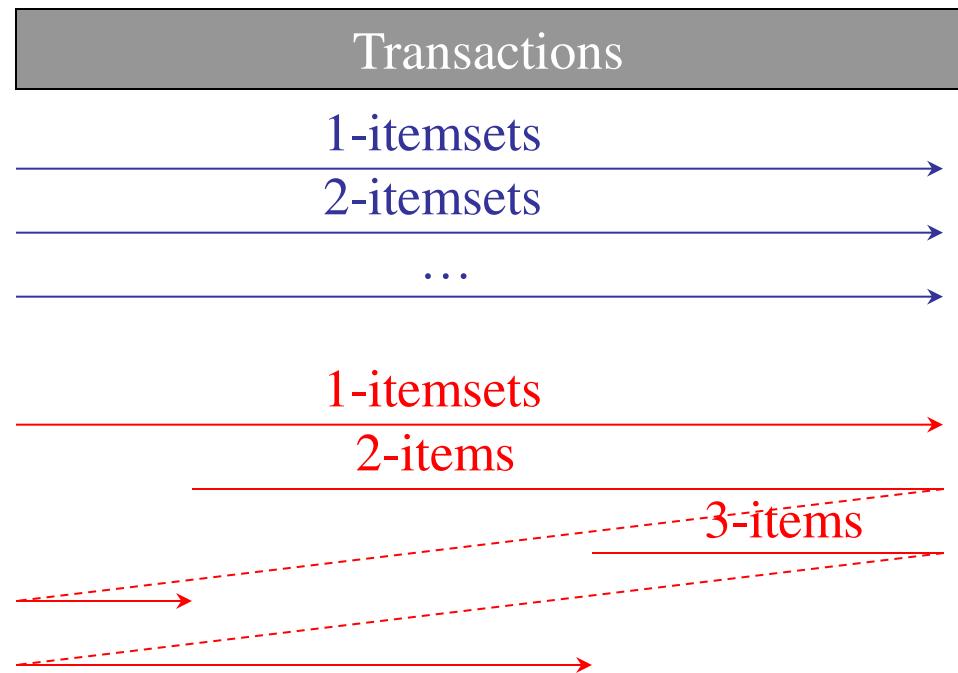


S. Brin R. Motwani, J. Ullman,
and S. Tsur. [Dynamic itemset
counting and implication rules for
market basket data](#). In

SIGMOD'97

August 27, 2021

- Once both A and D are determined frequent, the counting of AD begins
- Once all length-2 subsets of BCD are determined frequent, the counting of BCD begins



Bottleneck of Frequent-pattern Mining

- Multiple database scans are **costly**
- Mining long patterns needs many passes of scanning and generates lots of candidates
 - To find frequent itemset $i_1 i_2 \dots i_{100}$
 - # of scans: **100**
 - # of Candidates: $(_{100}^1) + (_{100}^2) + \dots + (_{100}^{100}) = 2^{100} - 1 = 1.27 * 10^{30}$!
- Bottleneck: candidate-generation-and-test
- Can we avoid candidate generation?

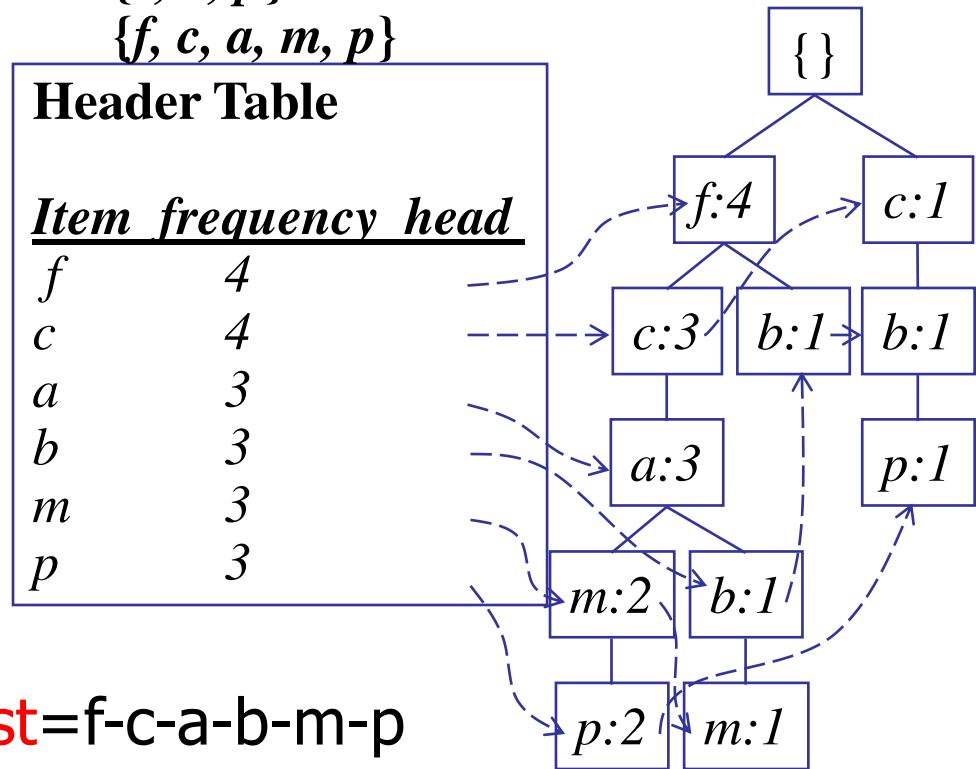
Mining Frequent Patterns Without Candidate Generation

- Grow long patterns from short ones using local frequent items
 - “abc” is a frequent pattern
 - Get all transactions having “abc”: DB|abc
 - “d” is a local frequent item in DB|abc → abcd is a frequent pattern

Construct FP-tree from a Transaction Database

<i>TID</i>	<i>Items bought</i>	<i>(ordered) frequent items</i>	<i>min_support = 3</i>
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}	
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}	
300	{b, f, h, j, o, w}	{f, b}	
400	{b, c, k, s, p}	{c, b, p}	
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}	

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



Benefits of the FP-tree Structure

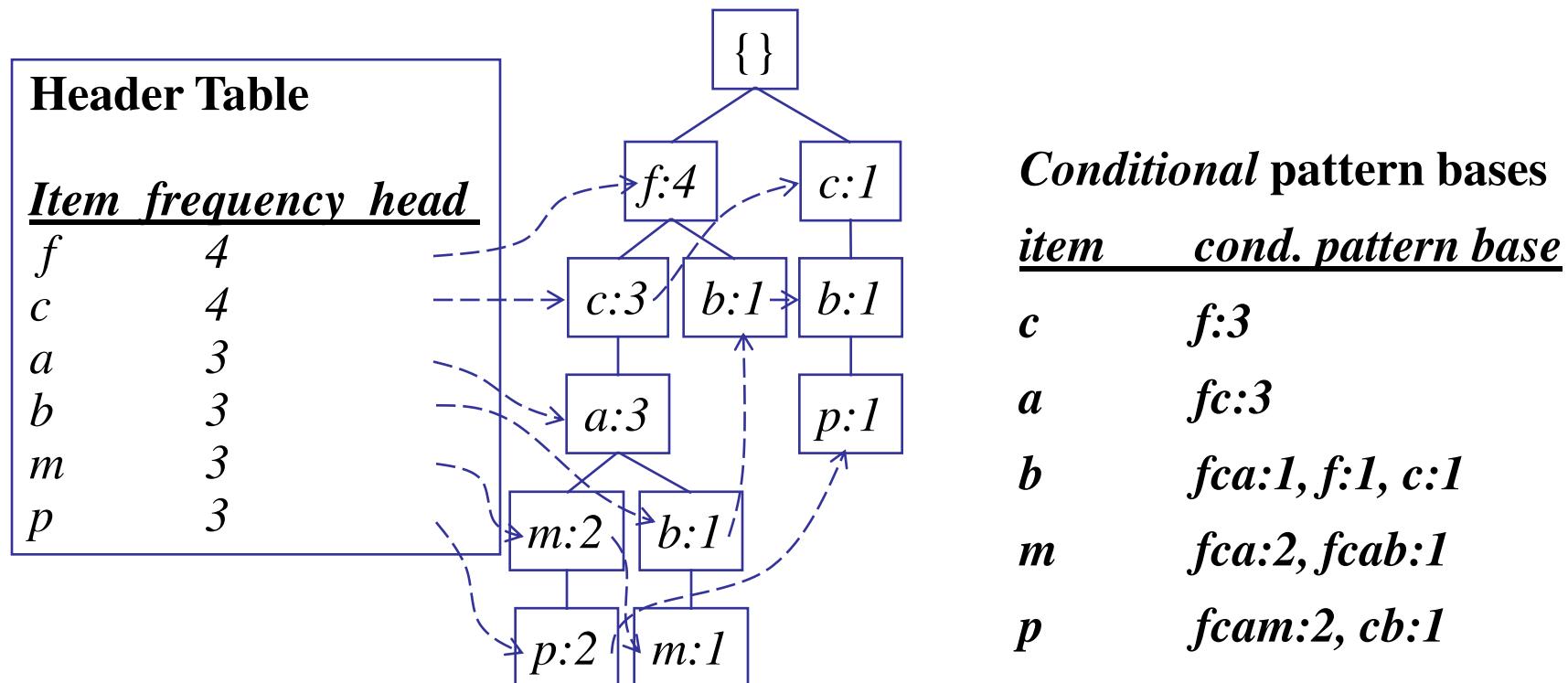
- Completeness
 - Preserve complete information for frequent pattern mining
 - Never break a long pattern of any transaction
- Compactness
 - Reduce irrelevant info—infrequent items are gone
 - Items in frequency descending order: the more frequently occurring, the more likely to be shared
 - Never be larger than the original database (not count node-links and the *count* field)
 - For Connect-4 DB, compression ratio could be over 100

Partition Patterns and Databases

- Frequent patterns can be partitioned into subsets according to f-list
 - F-list=f-c-a-b-m-p
 - Patterns containing p
 - Patterns having m but no p
 - ...
 - Patterns having c but no a nor b, m, p
 - Pattern f
- Completeness and non-redundancy

Find Patterns Having P From P-conditional Database

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



Mining Frequent Patterns With FP-trees

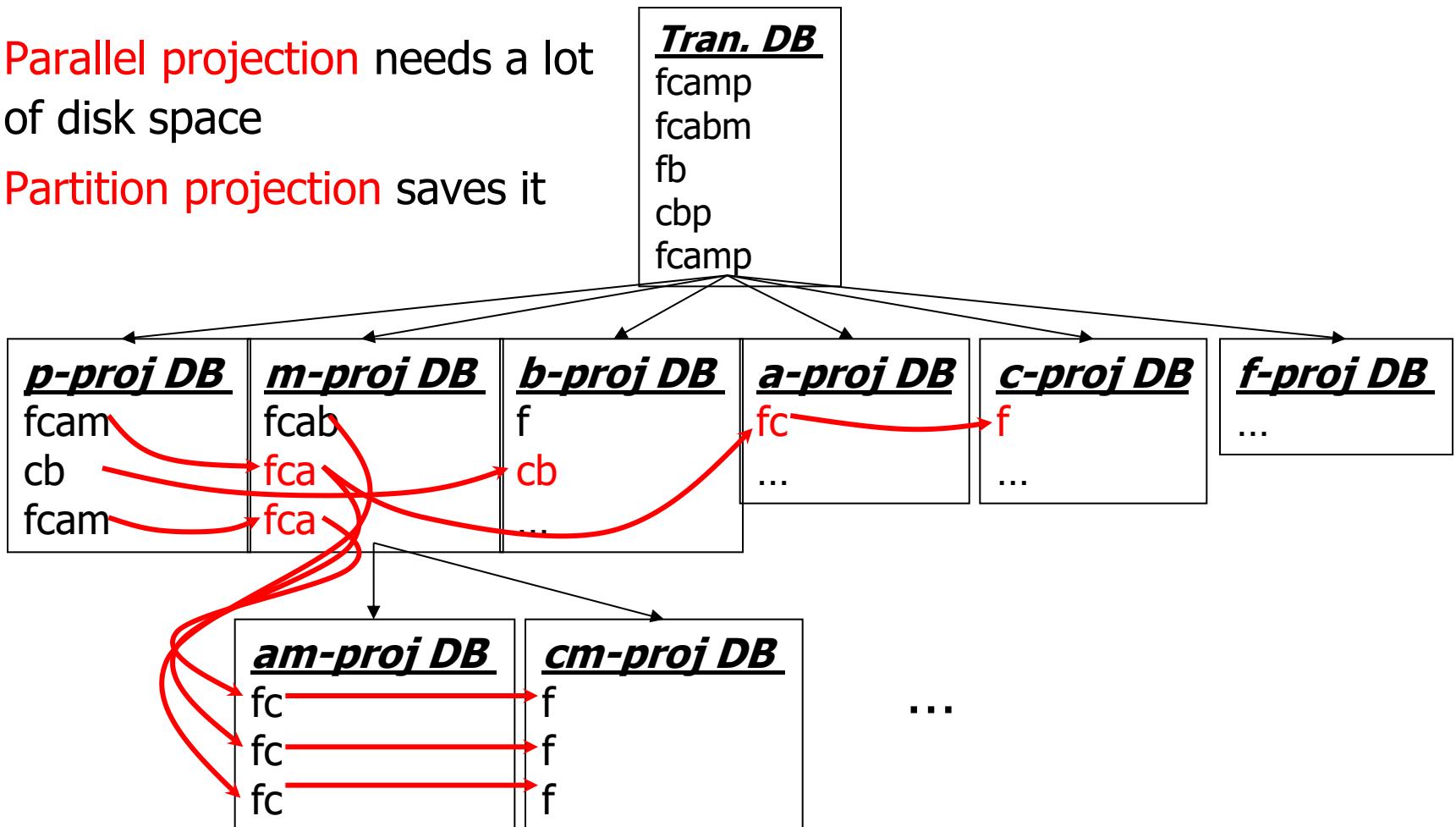
- Idea: Frequent pattern growth
 - Recursively grow frequent patterns by pattern and database partition
- Method
 - For each frequent item, construct its conditional pattern-base, and then its conditional FP-tree
 - Repeat the process on each newly created conditional FP-tree
 - Until the resulting FP-tree is empty, or it contains only one path—single path will generate all the combinations of its sub-paths, each of which is a frequent pattern

Scaling FP-growth by DB Projection

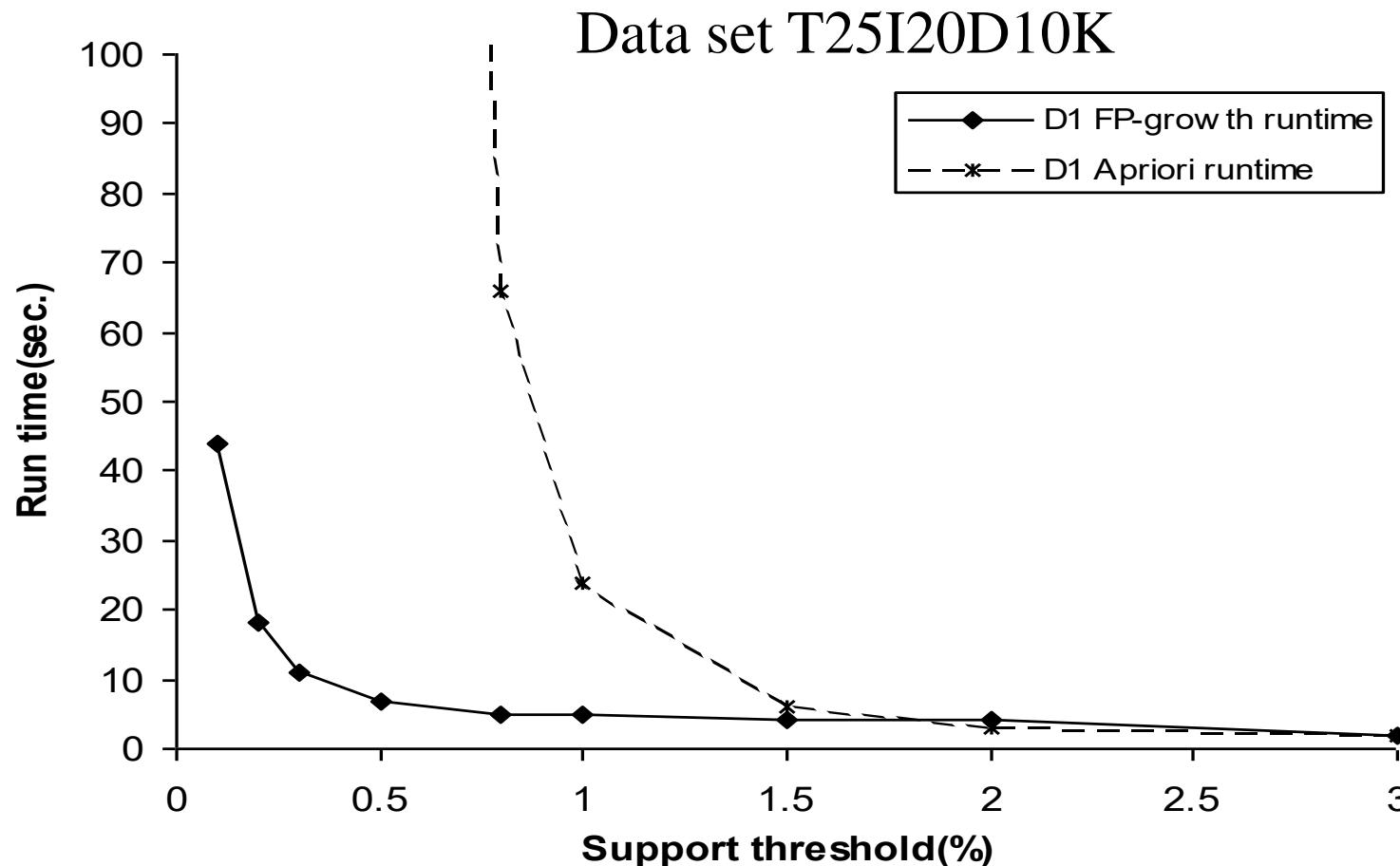
- FP-tree cannot fit in memory?—DB projection
- First partition a database into a set of projected DBs
- Then construct and mine FP-tree for each projected DB
- **Parallel projection** vs. **Partition projection** techniques
 - Parallel projection is space costly

Partition-based Projection

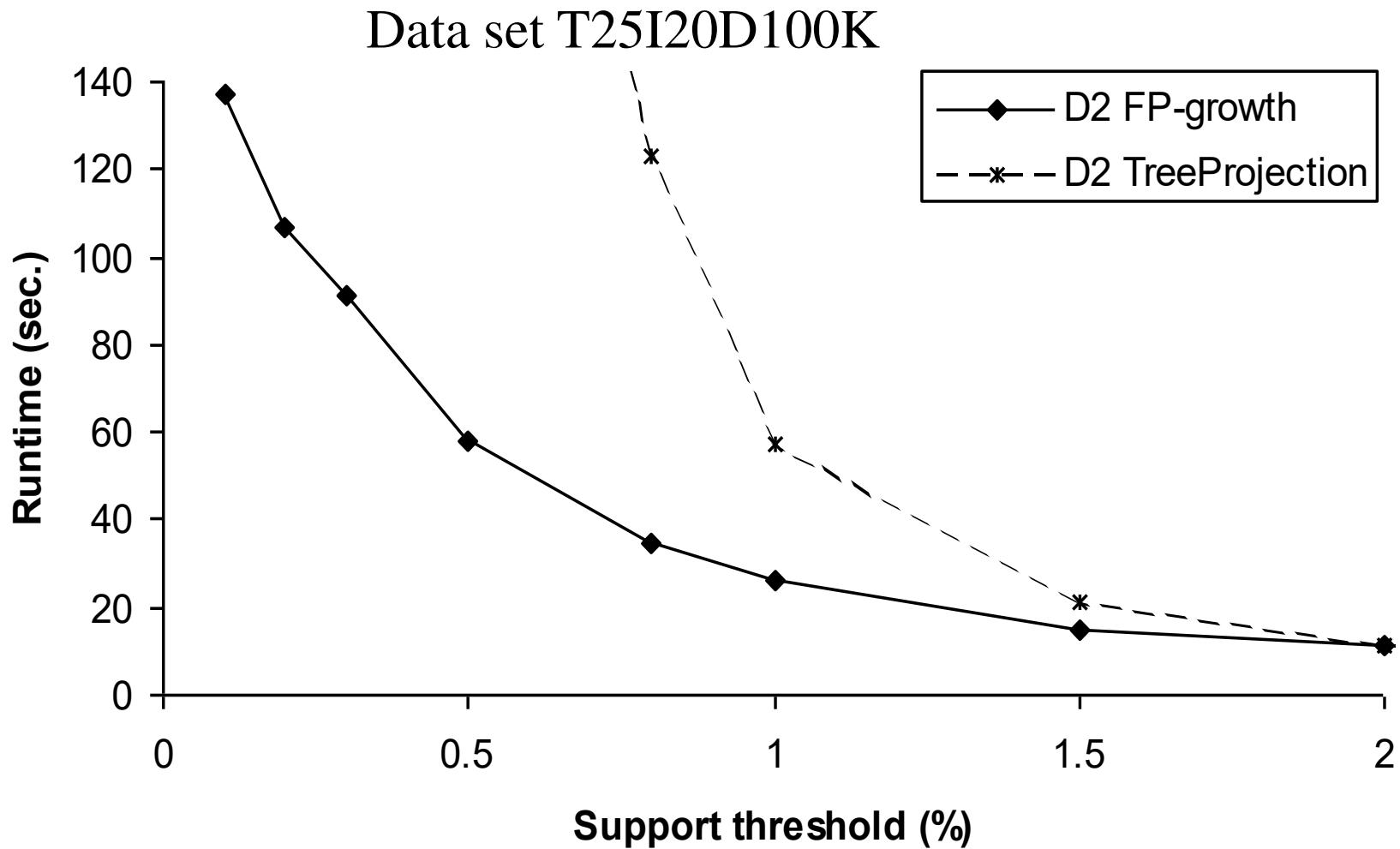
- Parallel projection needs a lot of disk space
- Partition projection saves it



FP-Growth vs. Apriori: Scalability With the Support Threshold



FP-Growth vs. Tree-Projection: Scalability with the Support Threshold



Why Is FP-Growth the Winner?

- Divide-and-conquer:
 - decompose both the mining task and DB according to the frequent patterns obtained so far
 - leads to focused search of smaller databases
- Other factors
 - no candidate generation, no candidate test
 - compressed database: FP-tree structure
 - no repeated scan of entire database
 - basic ops—counting local freq items and building sub FP-tree, no pattern search and matching

Implications of the Methodology

- Mining closed frequent itemsets and max-patterns
 - **CLOSET** (DMKD'00)
- Mining sequential patterns
 - **FreeSpan** (KDD'00), **PrefixSpan** (ICDE'01)
- Constraint-based mining of frequent patterns
 - **Convertible constraints** (KDD'00, ICDE'01)
- Computing iceberg data cubes with complex measures
 - **H-tree and H-cubing algorithm** (SIGMOD'01)

MaxMiner: Mining Max-patterns

- 1st scan: find frequent items
 - A, B, C, D, E
 - 2nd scan: find support for
 - AB, AC, AD, AE, ABCDE
 - BC, BD, BE, BCDE
 - CD, CE, CDE, DE,
 - Since BCDE is a max-pattern, no need to check BCD, BDE, CDE in later scan
 - R. Bayardo. Efficiently mining long patterns from databases. In *SIGMOD'98*
- Potential max-patterns
- | Tid | Items |
|-----|-----------|
| 10 | A,B,C,D,E |
| 20 | B,C,D,E, |
| 30 | A,C,D,F |

Mining Frequent Closed Patterns: CLOSET

- Flist: list of all frequent items in support ascending order
 - Flist: d-a-f-e-c
- Divide search space
 - Patterns having d
 - Patterns having d but no a, etc.
- Find frequent closed pattern recursively
 - Every transaction having d also has cfa \rightarrow cfad is a frequent closed pattern
- J. Pei, J. Han & R. Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets", DMKD'00.

Min_sup=2	
TID	Items
10	a, c, d, e, f
20	a, b, e
30	c, e, f
40	a, c, d, f
50	c, e, f

CLOSET+: Mining Closed Itemsets by Pattern-Growth

- Itemset merging: if Y appears in every occurrence of X , then Y is merged with X
- Sub-itemset pruning: if $Y \supset X$, and $\text{sup}(X) = \text{sup}(Y)$, X and all of X 's descendants in the set enumeration tree can be pruned
- Hybrid tree projection
 - Bottom-up physical tree-projection
 - Top-down pseudo tree-projection
- Item skipping: if a local frequent item has the same support in several header tables at different levels, one can prune it from the header table at higher levels
- Efficient subset checking

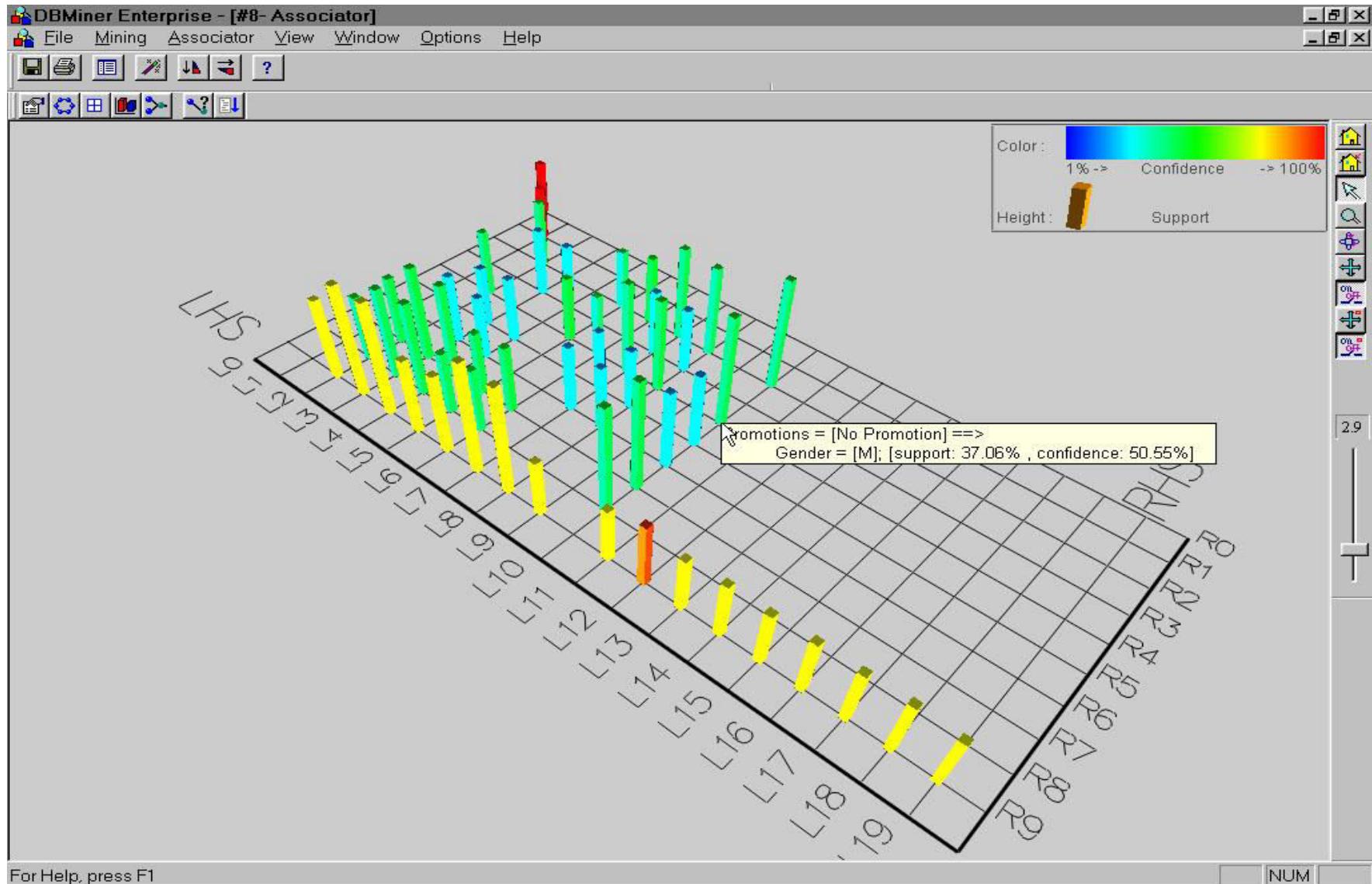
CHARM: Mining by Exploring Vertical Data Format

- Vertical format: $t(AB) = \{T_{11}, T_{25}, \dots\}$
 - tid-list: list of trans.-ids containing an itemset
- Deriving closed patterns based on vertical intersections
 - $t(X) = t(Y)$: X and Y always happen together
 - $t(X) \subset t(Y)$: transaction having X always has Y
- Using **diffset** to accelerate mining
 - Only keep track of differences of tids
 - $t(X) = \{T_1, T_2, T_3\}$, $t(XY) = \{T_1, T_3\}$
 - Diffset $(XY, X) = \{T_2\}$
- Eclat/MaxEclat (Zaki et al. @KDD'97), VIPER(P. Shenoy et al. @SIGMOD'00), CHARM (Zaki & Hsiao@SDM'02)

Further Improvements of Mining Methods

- AFOPT (Liu, et al. @ KDD'03)
 - A “push-right” method for mining condensed frequent pattern (CFP) tree
- Carpenter (Pan, et al. @ KDD'03)
 - Mine data sets with small rows but numerous columns
 - Construct a row-enumeration tree for efficient mining

Visualization of Association Rules: Plane Graph



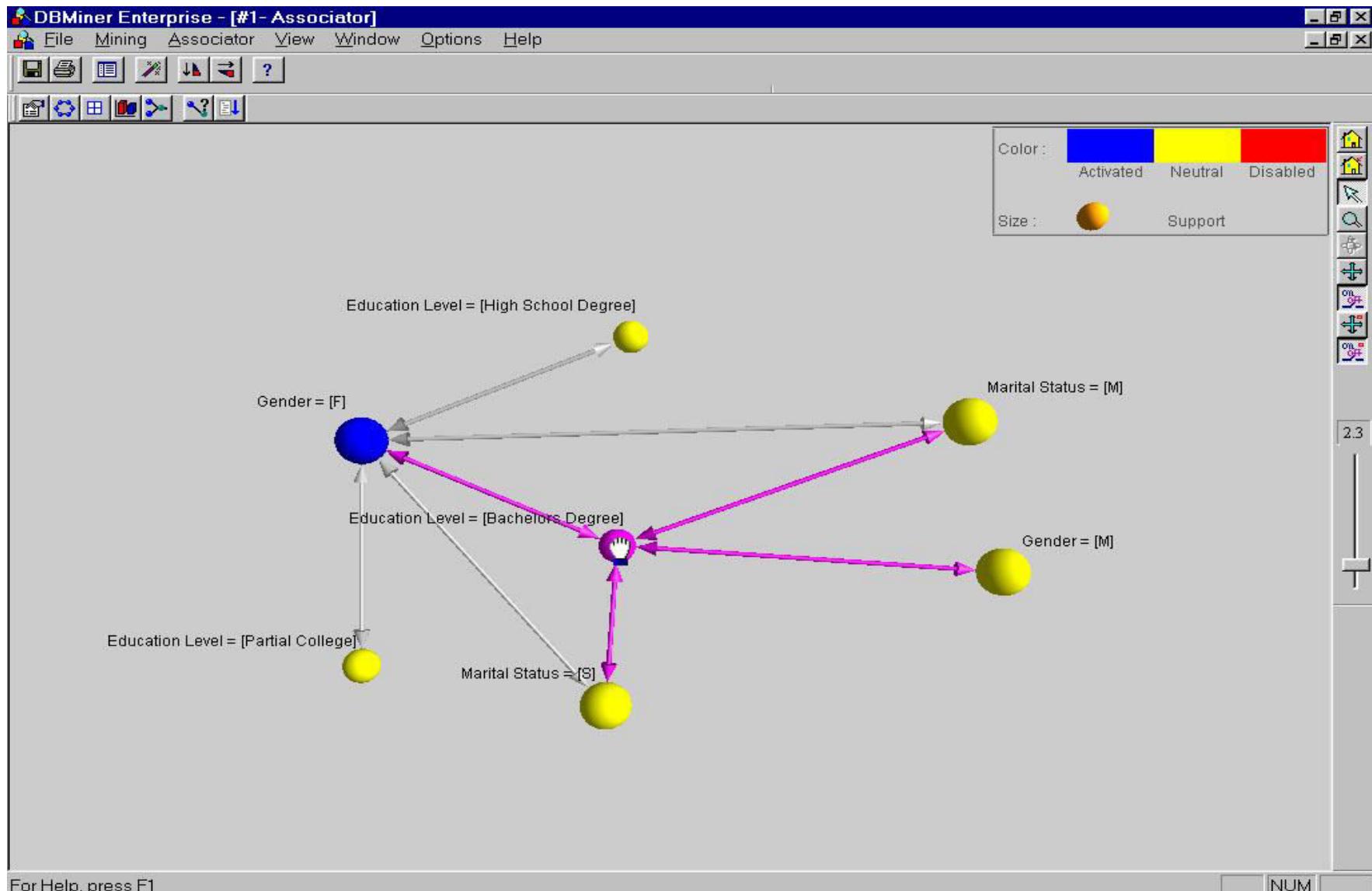
For Help, press F1

August 27, 2021

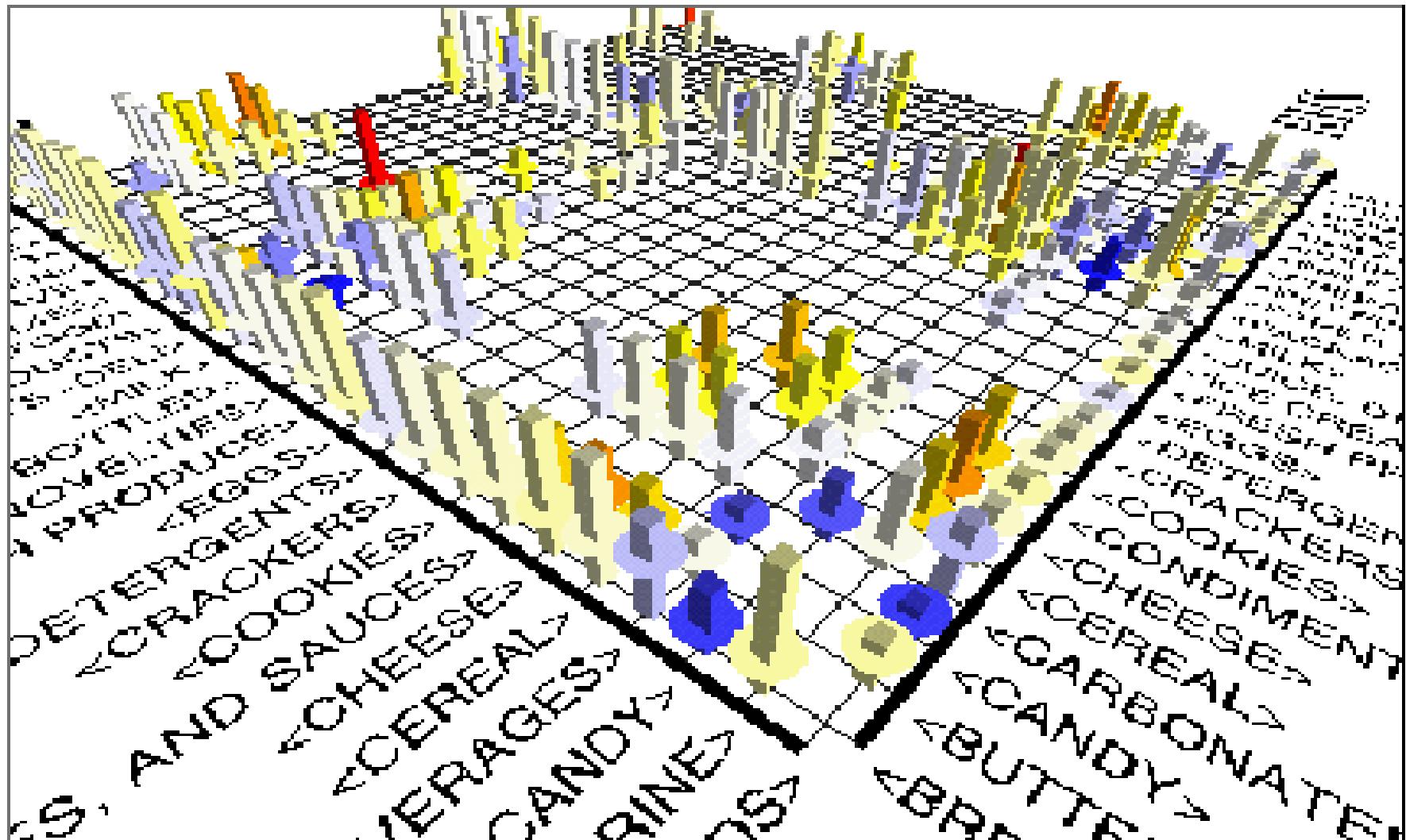
Data Mining: Concepts and Techniques

40

Visualization of Association Rules: Rule Graph



Visualization of Association Rules (SGI/MineSet 3.0)



Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary



Mining Various Kinds of Association Rules

- Mining multilevel association
- Mining multidimensional association
- Mining quantitative association
- Mining interesting correlation patterns

Mining Multiple-Level Association Rules

- Items often form hierarchies
- Flexible support settings
 - Items at the lower level are expected to have lower support
- Exploration of *shared* multi-level mining (Agrawal & Srikant@VLB'95, Han & Fu@VLDB'95)

uniform support

Level 1
min_sup = 5%

Milk
[support = 10 %]

Level 2
min_sup = 5%

2% Milk
[support = 6 %]

reduced support

Level 1
min_sup = 5%

Skim Milk
[support = 4 %]

Level 2
min_sup = 3%

Multi-level Association: Redundancy Filtering

- Some rules may be redundant due to “ancestor” relationships between items.
- Example
 - $\text{milk} \Rightarrow \text{wheat bread}$ [support = 8%, confidence = 70%]
 - $2\% \text{ milk} \Rightarrow \text{wheat bread}$ [support = 2%, confidence = 72%]
- We say the first rule is an ancestor of the second rule.
- A rule is redundant if its support is close to the “expected” value, based on the rule’s ancestor.

Mining Multi-Dimensional Association

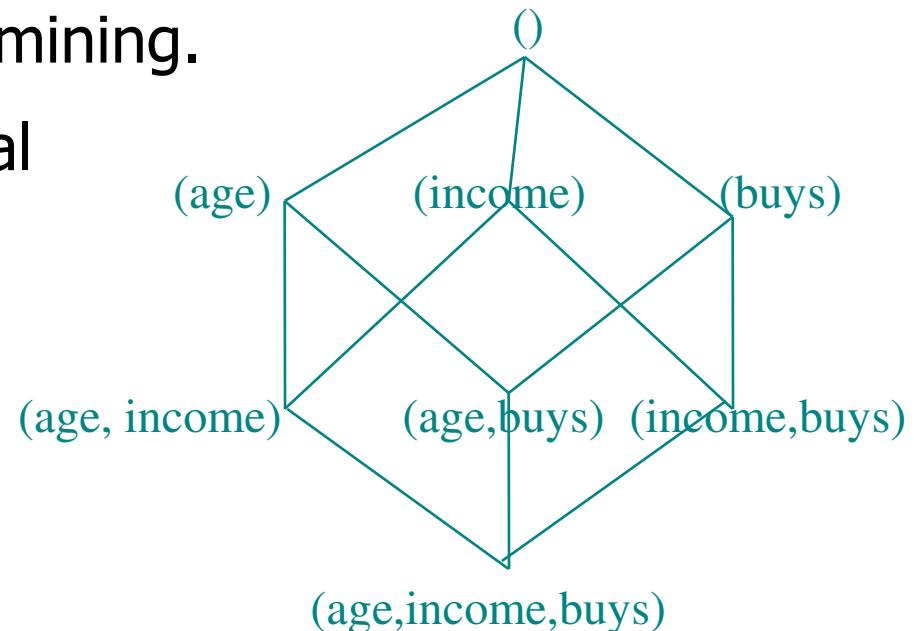
- Single-dimensional rules:
 $\text{buys}(X, \text{"milk"}) \Rightarrow \text{buys}(X, \text{"bread"})$
- Multi-dimensional rules: ≥ 2 dimensions or predicates
 - Inter-dimension assoc. rules (*no repeated predicates*)
 $\text{age}(X, \text{"19-25"}) \wedge \text{occupation}(X, \text{"student"}) \Rightarrow \text{buys}(X, \text{"coke"})$
 - hybrid-dimension assoc. rules (*repeated predicates*)
 $\text{age}(X, \text{"19-25"}) \wedge \text{buys}(X, \text{"popcorn"}) \Rightarrow \text{buys}(X, \text{"coke"})$
- Categorical Attributes: finite number of possible values, no ordering among values—data cube approach
- Quantitative Attributes: numeric, implicit ordering among values—discretization, clustering, and gradient approaches

Mining Quantitative Associations

- Techniques can be categorized by how numerical attributes, such as **age** or **salary** are treated
1. Static discretization based on predefined concept hierarchies (data cube methods)
 2. Dynamic discretization based on data distribution (quantitative rules, e.g., Agrawal & Srikant@SIGMOD96)
 3. Clustering: Distance-based association (e.g., Yang & Miller@SIGMOD97)
 - one dimensional clustering then association
 4. Deviation: (such as Aumann and Lindell@KDD99)
Sex = female => Wage: mean=\$7/hr (overall mean = \$9)

Static Discretization of Quantitative Attributes

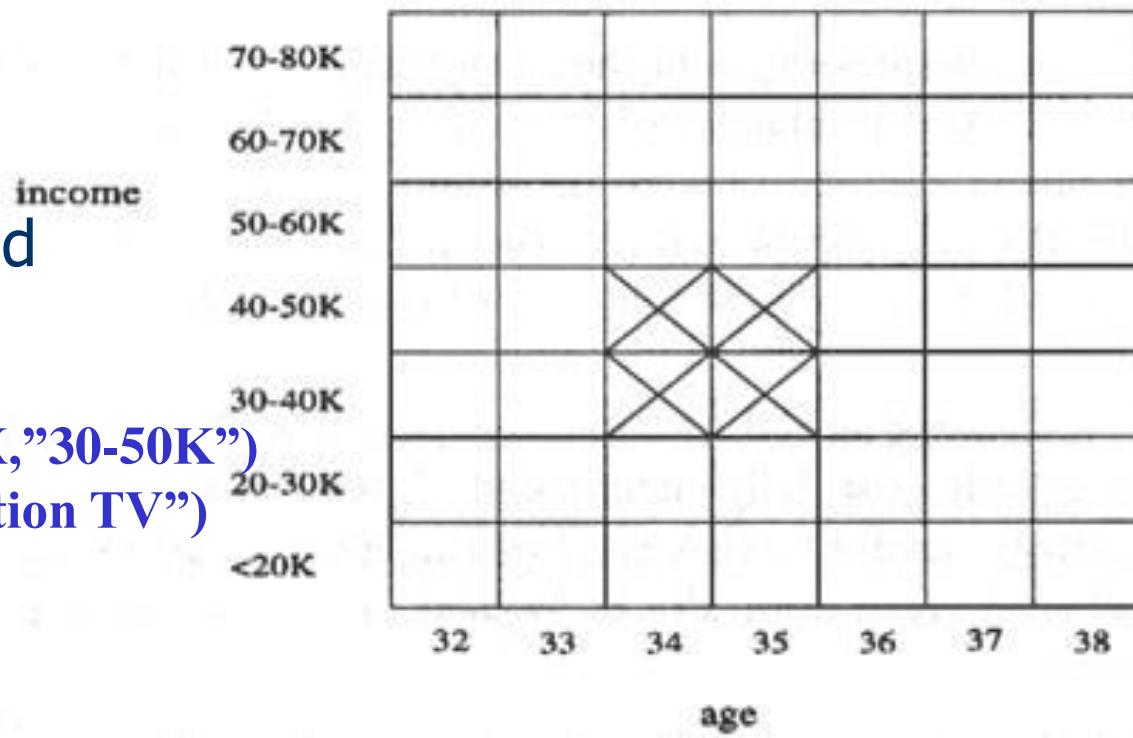
- Discretized prior to mining using concept hierarchy.
- Numeric values are replaced by ranges.
- In relational database, finding all frequent k -predicate sets will require k or $k+1$ table scans.
- Data cube is well suited for mining.
- The cells of an n -dimensional cuboid correspond to the predicate sets.
- Mining from data cubes can be much faster.



Quantitative Association Rules

- Proposed by Lent, Swami and Widom ICDE'97
- Numeric attributes are *dynamically* discretized
 - Such that the confidence or compactness of the rules mined is maximized
- 2-D quantitative association rules: $A_{\text{quan1}} \wedge A_{\text{quan2}} \Rightarrow A_{\text{cat}}$
- Cluster *adjacent* association rules to form general rules using a 2-D grid
- Example

$\text{age}(X, "34-35") \wedge \text{income}(X, "30-50K")$
 $\Rightarrow \text{buys}(X, \text{"high resolution TV"})$

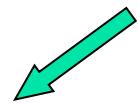


Mining Other Interesting Patterns

- Flexible support constraints (Wang et al. @ VLDB'02)
 - Some items (e.g., diamond) may occur rarely but are valuable
 - Customized sup_{\min} specification and application
- Top-K closed frequent patterns (Han, et al. @ ICDM'02)
 - Hard to specify sup_{\min} , but top-k with length_{\min} is more desirable
 - Dynamically raise sup_{\min} in FP-tree construction and mining, and select most promising path to mine

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary



Interestingness Measure: Correlations (Lift)

- $play\ basketball \Rightarrow eat\ cereal$ [40%, 66.7%] is misleading
 - The overall % of students eating cereal is 75% > 66.7%.
- $play\ basketball \Rightarrow not\ eat\ cereal$ [20%, 33.3%] is more accurate, although with lower support and confidence
- Measure of dependent/correlated events: lift

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

	Basketball	Not basketball	Sum (row)
Cereal	2000	1750	3750
Not cereal	1000	250	1250
Sum(col.)	3000	2000	5000

$$lift(B, C) = \frac{2000/5000}{3000/5000 * 3750/5000} = 0.89$$

$$lift(B, \neg C) = \frac{1000/5000}{3000/5000 * 1250/5000} = 1.33$$

Are *lift* and χ^2 Good Measures of Correlation?

- "*Buy walnuts \Rightarrow buy milk [1%, 80%]*" is misleading
 - if 85% of customers buy milk
- Support and confidence are not good to represent correlations
- So many interestingness measures? (Tan, Kumar, Srivastava @KDD'02)

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

$$all_conf = \frac{\sup(X)}{\max_item_sup(X)}$$

$$coh = \frac{\sup(X)}{|universe(X)|}$$

	Milk	No Milk	Sum (row)
Coffee	m, c	$\sim m, c$	c
No Coffee	$\sim m, \sim c$	$\sim m, c$	$\sim c$
Sum(col.)	m	$\sim m$	Σ

DB	m, c	$\sim m, c$	$m \sim c$	$\sim m \sim c$	lift	all-conf	coh	χ^2
A1	1000	100	100	10,000	9.26	0.91	0.83	9055
A2	100	1000	1000	100,000	8.44	0.09	0.05	670
A3	1000	100	10000	100,000	9.18	0.09	0.09	8172
A4	1000	1000	1000	1000	1	0.5	0.33	0

Which Measures Should Be Used?

- **lift** and χ^2 are not good measures for correlations in large transactional DBs
- **all-conf** or **coherence** could be good measures (Omiecinski@TKDE'03)
- Both **all-conf** and **coherence** have the downward closure property
- Efficient algorithms can be derived for mining (Lee et al. @ICDM'03sub)

symbol	measure	range	formula
ϕ	ϕ -coefficient	-1 … 1	$\frac{P(A,B) - P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
Q	Yule's Q	-1 … 1	$\frac{P(A,B)P(\bar{A},\bar{B}) - P(A,\bar{B})P(\bar{A},B)}{P(A,B)P(\bar{A},\bar{B}) + P(A,\bar{B})P(\bar{A},B)}$
Y	Yule's Y	-1 … 1	$\frac{\sqrt{P(A,B)P(\bar{A},\bar{B})} - \sqrt{P(A,\bar{B})P(\bar{A},B)}}{\sqrt{P(A,B)P(\bar{A},\bar{B})} + \sqrt{P(A,\bar{B})P(\bar{A},B)}}$
k	Cohen's	-1 … 1	$\frac{P(A,B) - P(A)P(B)}{1 - P(A)P(B) - P(\bar{A})P(\bar{B})}$
PS	Piatetsky-Shapiro's	-0.25 … 0.25	$P(A,B) - P(A)P(B)$
F	Certainty factor	-1 … 1	$\max\left(\frac{P(B A) - P(B)}{1 - P(B)}, \frac{P(A B) - P(A)}{1 - P(A)}\right)$
AV	added value	-0.5 … 1	$\max(P(B A) - P(B), P(A B) - P(A))$
K	Klosgen's Q	-0.33 … 0.38	$\sqrt{P(A,B) \max(P(B A) - P(B), P(A B) - P(A))}$
g	Goodman-kruskal's	0 … 1	$\frac{\sum_j \max_k P(A_j, B_k) + \sum_k \max_j P(A_j, B_k) - \max_j P(A_j) - \max_k P(B_k)}{2 - \max_j P(A_j) - \max_k P(B_k)}$
M	Mutual Information	0 … 1	$\frac{\sum_i \sum_j P(A_i, B_j) \log \frac{P(A_i, B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i) \log P(A_i) \log P(A_i), -\sum_i P(B_i) \log P(B_i) \log P(B_i))}$
J	J-Measure	0 … 1	$\max(P(A, B) \log \frac{P(B A)}{P(B)}, P(A\bar{B}) \log \frac{P(\bar{B} A)}{P(\bar{B})})$ $P(A, B) \log \frac{P(A B)}{P(A)} + P(\bar{A}B) \log \frac{P(\bar{A} B)}{P(\bar{A})}$
G	Gini index	0 … 1	$\max(P(A)[P(B A)^2 + P(\bar{B} A)^2] + P(\bar{A})[P(B \bar{A})^2 + P(\bar{B} \bar{A})^2] - P(B)^2 - P(\bar{B})^2,$ $P(B)[P(A B)^2 + P(\bar{A} B)^2] + P(\bar{B})[P(A \bar{B})^2 + P(\bar{A} \bar{B})^2] - P(A)^2 - P(\bar{A})^2)$
s	support	0 … 1	$P(A, B)$
c	confidence	0 … 1	$\max(P(B A), P(A B))$
L	Laplace	0 … 1	$\max\left(\frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2}\right)$
IS	Cosine	0 … 1	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
γ	coherence(Jaccard)	0 … 1	$\frac{P(A,B)}{P(A) + P(B) - P(A,B)}$
α	all_confidence	0 … 1	$\frac{P(A,B)}{\max(P(A), P(B))}$
o	odds ratio	0 … ∞	$\frac{P(A,B)P(\bar{A},\bar{B})}{P(\bar{A},B)P(A,\bar{B})}$
V	Conviction	0.5 … ∞	$\max\left(\frac{P(A)P(\bar{B})}{P(AB)}, \frac{P(B)P(\bar{A})}{P(B\bar{A})}\right)$
λ	lift	0 … ∞	$\frac{P(A,B)}{P(A)\bar{P}(B)}$
S	Collective strength	0 … ∞	$\frac{P(A,B) + P(\bar{A}\bar{B})}{P(A)P(B) + P(\bar{A})P(\bar{B})} \times \frac{1 - P(A)P(B) - P(\bar{A})P(\bar{B})}{1 - P(A,B) - P(\bar{A}\bar{B})}$
χ^2	χ^2	0 … ∞	$\sum_i \frac{(P(A_i) - E_i)^2}{E_i}$

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary



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
 - System optimization: explores such constraints for efficient mining—**constraint-based mining**

Constraints in Data Mining

- Knowledge type constraint:
 - classification, association, etc.
- Data constraint — using SQL-like queries
 - find product pairs sold together in stores in Chicago in Dec.'02
- 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: $\text{min_support} \geq 3\%$, $\text{min_confidence} \geq 60\%$

Constrained Mining vs. Constraint-Based Search

- Constrained mining vs. constraint-based search/reasoning
 - Both are aimed at reducing search space
 - Finding **all patterns** satisfying constraints vs. finding **some (or one) answer** in constraint-based search in AI
 - **Constraint-pushing** vs. **heuristic search**
 - It is an interesting research problem on how to integrate them
- Constrained mining vs. query processing in DBMS
 - Database query processing requires to find all
 - Constrained pattern mining shares a similar philosophy as pushing selections deeply in query processing

Anti-Monotonicity in Constraint Pushing

- Anti-monotonicity

- When an itemset S **violates** the constraint, so does any of its superset
- $\text{sum}(S.\text{Price}) \leq v$ is **anti-monotone**
- $\text{sum}(S.\text{Price}) \geq v$ is **not anti-monotone**

- Example. C: $\text{range}(S.\text{profit}) \leq 15$ is **anti-monotone**

- Itemset ab violates C
- So does every superset of ab

TDB (min_sup=2)

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Monotonicity for Constraint Pushing

TDB (min_sup=2)

- Monotonicity
 - When an itemset S **satisfies** the constraint, so does any of its superset
 - $\text{sum}(S.\text{Price}) \geq v$ is **monotone**
 - $\text{min}(S.\text{Price}) \leq v$ is **monotone**
- Example. C: $\text{range}(S.\text{profit}) \geq 15$
 - Itemset ab satisfies C
 - So does every superset of ab

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Succinctness

- Succinctness:
 - Given A_1 , the set of items satisfying a succinctness constraint C , then any set S satisfying C is based on A_1 , i.e., S contains a subset belonging to A_1
 - Idea: Without looking at the transaction database, whether an itemset S satisfies constraint C can be determined based on the selection of items
 - $\min(S.\text{Price}) \leq v$ is succinct
 - $\sum(S.\text{Price}) \geq v$ is not succinct
- Optimization: If C is succinct, C is pre-counting pushable

The Apriori Algorithm — Example

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Data Mining: Concepts and Techniques

Naïve Algorithm: Apriori + Constraint

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

Scan D

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_2

Scan D

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

itemset
{2 3 5}

Scan D

itemset	sup
{2 3 5}	2

Constraint:

Sum{S.price} < 5

The Constrained Apriori Algorithm: Push an Anti-monotone Constraint Deep

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

Scan D

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

L_2

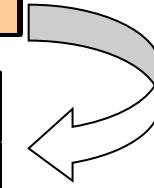
itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_2

Scan D

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

C_2



itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

itemset
{2 3 5}

Scan D

itemset	sup
{2 3 5}	2

Constraint:

Sum{S.price} < 5

The Constrained Apriori Algorithm: Push a Succinct Constraint Deep

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

C_1

Scan D

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

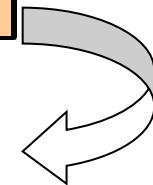
L_1

itemset	sup.
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

C_2

Scan D

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}



not immediately
to be used

itemset
{2 3 5}

Scan D

itemset	sup.
{2 3 5}	2

L_3

Constraint:

$\min\{S.\text{price}\} \leq 1$

Converting “Tough” Constraints

- Convert tough constraints into anti-monotone or monotone by properly ordering items
- Examine C: $\text{avg}(S.\text{profit}) \geq 25$
 - Order items in value-descending order
 - $\langle a, f, g, d, b, h, c, e \rangle$
 - If an itemset afb violates C
 - So does $afbh, afb^*$
 - It becomes **anti-monotone!**

TDB (min_sup=2)

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Strongly Convertible Constraints

- $\text{avg}(X) \geq 25$ is convertible anti-monotone w.r.t. item **value descending** order R : $\langle a, f, g, d, b, h, c, e \rangle$
 - If an itemset af violates a constraint C , so does every itemset with af as prefix, such as afd
- $\text{avg}(X) \geq 25$ is convertible monotone w.r.t. item **value ascending** order R^{-1} : $\langle e, c, h, b, d, g, f, a \rangle$
 - If an itemset d satisfies a constraint C , so does itemsets df and dfa , which having d as a prefix
- Thus, $\text{avg}(X) \geq 25$ is **strongly convertible**

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Can Apriori Handle Convertible Constraint?

- A convertible, not monotone nor anti-monotone nor succinct constraint cannot be pushed deep into the an Apriori mining algorithm
 - Within the level wise framework, no direct pruning based on the constraint can be made
 - Itemset df violates constraint C: $\text{avg}(X) \geq 25$
 - Since adf satisfies C, Apriori needs df to assemble adf, df cannot be pruned
- But it can be pushed into frequent-pattern growth framework!

Item	Value
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Mining With Convertible Constraints

- C: $\text{avg}(X) \geq 25$, $\text{min_sup}=2$
- List items in every transaction in value descending order R: $\langle a, f, g, d, b, h, c, e \rangle$
 - C is convertible anti-monotone w.r.t. R
- Scan TDB once
 - remove infrequent items
 - Item h is dropped
 - Itemsets a and f are good, ...
- Projection-based mining
 - Imposing an appropriate order on item projection
 - Many tough constraints can be converted into (anti)-monotone

Item	Value
a	40
f	30
g	20
d	10
b	0
h	-10
c	-20
e	-30

TDB ($\text{min_sup}=2$)

TID	Transaction
10	a, f, d, b, c
20	f, g, d, b, c
30	a, f, d, c, e
40	f, g, h, c, e

Handling Multiple Constraints

- Different constraints may require different or even conflicting item-ordering
- If there exists an order R s.t. both C_1 and C_2 are convertible w.r.t. R , then there is no conflict between the two convertible constraints
- If there exists conflict on order of items
 - Try to satisfy one constraint first
 - Then using the order for the other constraint to mine frequent itemsets in the corresponding projected database

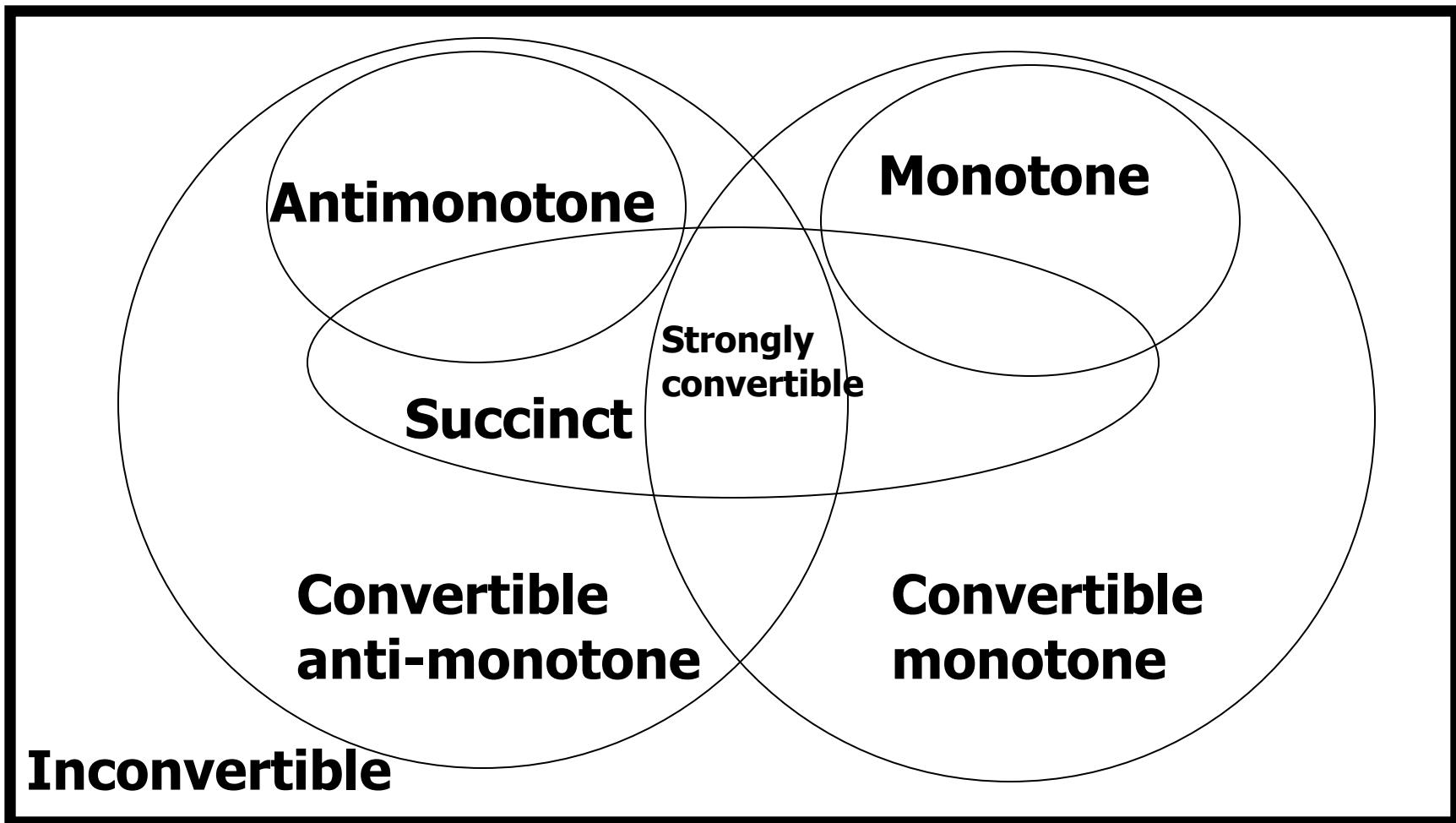
What Constraints Are Convertible?

Constraint	Convertible anti-monotone	Convertible monotone	Strongly convertible
$\text{avg}(S) \leq, \geq v$	Yes	Yes	Yes
$\text{median}(S) \leq, \geq v$	Yes	Yes	Yes
$\text{sum}(S) \leq v$ (items could be of any value, $v \geq 0$)	Yes	No	No
$\text{sum}(S) \leq v$ (items could be of any value, $v \leq 0$)	No	Yes	No
$\text{sum}(S) \geq v$ (items could be of any value, $v \geq 0$)	No	Yes	No
$\text{sum}(S) \geq v$ (items could be of any value, $v \leq 0$)	Yes	No	No
.....			

Constraint-Based Mining—A General Picture

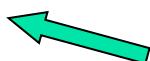
Constraint	Antimonotone	Monotone	Succinct
$v \in S$	no	yes	yes
$S \supseteq V$	no	yes	yes
$S \subseteq V$	yes	no	yes
$\min(S) \leq v$	no	yes	yes
$\min(S) \geq v$	yes	no	yes
$\max(S) \leq v$	yes	no	yes
$\max(S) \geq v$	no	yes	yes
$\text{count}(S) \leq v$	yes	no	weakly
$\text{count}(S) \geq v$	no	yes	weakly
$\sum(S) \leq v \ (\ a \in S, a \geq 0 \)$	yes	no	no
$\sum(S) \geq v \ (\ a \in S, a \geq 0 \)$	no	yes	no
$\text{range}(S) \leq v$	yes	no	no
$\text{range}(S) \geq v$	no	yes	no
$\text{avg}(S) \theta v, \theta \in \{=, \leq, \geq\}$	convertible	convertible	no
$\text{support}(S) \geq \xi$	yes	no	no
$\text{support}(S) \leq \xi$	no	yes	no

A Classification of Constraints



Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary



Frequent-Pattern Mining: Summary

- Frequent pattern mining—an important task in data mining
- Scalable frequent pattern mining methods
 - Apriori (Candidate generation & test)
 - Projection-based (FPgrowth, CLOSET+, ...)
 - Vertical format approach (CHARM, ...)
- Mining a variety of rules and interesting patterns
- Constraint-based mining
- Mining sequential and structured patterns
- Extensions and applications

Frequent-Pattern Mining: Research Problems

- Mining fault-tolerant frequent, sequential and structured patterns
 - Patterns allows limited faults (insertion, deletion, mutation)
- Mining truly interesting patterns
 - Surprising, novel, concise, ...
- Application exploration
 - E.g., DNA sequence analysis and bio-pattern classification
 - “Invisible” data mining

Ref: Basic Concepts of Frequent Pattern Mining

- (**Association Rules**) R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. SIGMOD'93.
- (**Max-pattern**) R. J. Bayardo. Efficiently mining long patterns from databases. SIGMOD'98.
- (**Closed-pattern**) N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. ICDT'99.
- (**Sequential pattern**) R. Agrawal and R. Srikant. Mining sequential patterns. ICDE'95

Ref: Apriori and Its Improvements

- R. Agrawal and R. Srikant. Fast algorithms for mining association rules. VLDB'94.
- H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. KDD'94.
- A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. VLDB'95.
- J. S. Park, M. S. Chen, and P. S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95.
- H. Toivonen. Sampling large databases for association rules. VLDB'96.
- S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. SIGMOD'97.
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98.

Ref: Depth-First, Projection-Based FP Mining

- R. Agarwal, C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. *J. Parallel and Distributed Computing*:02.
- J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *SIGMOD' 00*.
- J. Pei, J. Han, and R. Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. *DMKD'00*.
- J. Liu, Y. Pan, K. Wang, and J. Han. Mining Frequent Item Sets by Opportunistic Projection. *KDD'02*.
- J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining Top-K Frequent Closed Patterns without Minimum Support. *ICDM'02*.
- J. Wang, J. Han, and J. Pei. CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. *KDD'03*.
- G. Liu, H. Lu, W. Lou, J. X. Yu. On Computing, Storing and Querying Frequent Patterns. *KDD'03*.

Ref: Vertical Format and Row Enumeration Methods

- M. J. Zaki, S. Parthasarathy, M. Ogihsara, and W. Li. Parallel algorithm for discovery of association rules. DAMI:97.
- Zaki and Hsiao. CHARM: An Efficient Algorithm for Closed Itemset Mining, SDM'02.
- C. Bucila, J. Gehrke, D. Kifer, and W. White. DualMiner: A Dual-Pruning Algorithm for Itemsets with Constraints. KDD'02.
- F. Pan, G. Cong, A. K. H. Tung, J. Yang, and M. Zaki , CARPENTER: Finding Closed Patterns in Long Biological Datasets. KDD'03.

Ref: Mining Multi-Level and Quantitative Rules

- R. Srikant and R. Agrawal. Mining generalized association rules. VLDB'95.
- J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. VLDB'95.
- R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. SIGMOD'96.
- T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. SIGMOD'96.
- K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. KDD'97.
- R.J. Miller and Y. Yang. Association rules over interval data. SIGMOD'97.
- Y. Aumann and Y. Lindell. A Statistical Theory for Quantitative Association Rules KDD'99.

Ref: Mining Correlations and Interesting Rules

- M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM'94.
- S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. SIGMOD'97.
- C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. VLDB'98.
- P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. KDD'02.
- E. Omiecinski. Alternative Interest Measures for Mining Associations. TKDE'03.
- Y. K. Lee, W.Y. Kim, Y. D. Cai, and J. Han. CoMine: Efficient Mining of Correlated Patterns. ICDM'03.

Ref: Mining Other Kinds of Rules

- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96.
- B. Lent, A. Swami, and J. Widom. Clustering association rules. ICDE'97.
- A. Savasere, E. Omiecinski, and S. Navathe. Mining for strong negative associations in a large database of customer transactions. ICDE'98.
- D. Tsur, J. D. Ullman, S. Abitboul, C. Clifton, R. Motwani, and S. Nestorov. Query flocks: A generalization of association-rule mining. SIGMOD'98.
- F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Ratio rules: A new paradigm for fast, quantifiable data mining. VLDB'98.
- K. Wang, S. Zhou, J. Han. Profit Mining: From Patterns to Actions. EDBT'02.

Ref: Constraint-Based Pattern Mining

- R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. KDD'97.
- R. Ng, L.V.S. Lakshmanan, J. Han & A. Pang. Exploratory mining and pruning optimizations of constrained association rules. SIGMOD'98.
- M.N. Garofalakis, R. Rastogi, K. Shim: SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. VLDB'99.
- G. Grahe, L. Lakshmanan, and X. Wang. Efficient mining of constrained correlated sets. ICDE'00.
- J. Pei, J. Han, and L. V. S. Lakshmanan. Mining Frequent Itemsets with Convertible Constraints. ICDE'01.
- J. Pei, J. Han, and W. Wang, Mining Sequential Patterns with Constraints in Large Databases, CIKM'02.

Ref: Mining Sequential and Structured Patterns

- R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. EDBT'96.
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. DAMI:97.
- M. Zaki. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning:01.
- J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. ICDE'01.
- M. Kuramochi and G. Karypis. Frequent Subgraph Discovery. ICDM'01.
- X. Yan, J. Han, and R. Afshar. CloSpan: Mining Closed Sequential Patterns in Large Datasets. SDM'03.
- X. Yan and J. Han. CloseGraph: Mining Closed Frequent Graph Patterns. KDD'03.

Ref: Mining Spatial, Multimedia, and Web Data

- K. Koperski and J. Han, Discovery of Spatial Association Rules in Geographic Information Databases, SSD'95.
- O. R. Zaiane, M. Xin, J. Han, Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. ADL'98.
- O. R. Zaiane, J. Han, and H. Zhu, Mining Recurrent Items in Multimedia with Progressive Resolution Refinement. ICDE'00.
- D. Gunopulos and I. Tsoukatos. Efficient Mining of Spatiotemporal Patterns. SSTD'01.

Ref: Mining Frequent Patterns in Time-Series Data

- B. Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. ICDE'98.
- J. Han, G. Dong and Y. Yin, Efficient Mining of Partial Periodic Patterns in Time Series Database, ICDE'99.
- H. Lu, L. Feng, and J. Han. Beyond Intra-Transaction Association Analysis: Mining Multi-Dimensional Inter-Transaction Association Rules. TOIS:00.
- B.-K. Yi, N. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, and A. Biliris. Online Data Mining for Co-Evolving Time Sequences. ICDE'00.
- W. Wang, J. Yang, R. Muntz. TAR: Temporal Association Rules on Evolving Numerical Attributes. ICDE'01.
- J. Yang, W. Wang, P. S. Yu. Mining Asynchronous Periodic Patterns in Time Series Data. TKDE'03.

Ref: Iceberg Cube and Cube Computation

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96.
- Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multi-dimensional aggregates. SIGMOD'97.
- J. Gray, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. DAMI: 97.
- M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing iceberg queries efficiently. VLDB'98.
- S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of OLAP data cubes. EDBT'98.
- K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cubes. SIGMOD'99.

Ref: Iceberg Cube and Cube Exploration

- J. Han, J. Pei, G. Dong, and K. Wang, Computing Iceberg Data Cubes with Complex Measures. SIGMOD' 01.
- W. Wang, H. Lu, J. Feng, and J. X. Yu. Condensed Cube: An Effective Approach to Reducing Data Cube Size. ICDE'02.
- G. Dong, J. Han, J. Lam, J. Pei, and K. Wang. Mining Multi-Dimensional Constrained Gradients in Data Cubes. VLDB'01.
- T. Imielinski, L. Khachiyan, and A. Abdulghani. Cubegrades: Generalizing association rules. DAMI:02.
- L. V. S. Lakshmanan, J. Pei, and J. Han. Quotient Cube: How to Summarize the Semantics of a Data Cube. VLDB'02.
- D. Xin, J. Han, X. Li, B. W. Wah. Star-Cubing: Computing Iceberg Cubes by Top-Down and Bottom-Up Integration. VLDB'03.

Ref: FP for Classification and Clustering

- G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. KDD'99.
- B. Liu, W. Hsu, Y. Ma. Integrating Classification and Association Rule Mining. KDD'98.
- W. Li, J. Han, and J. Pei. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. ICDM'01.
- H. Wang, W. Wang, J. Yang, and P.S. Yu. Clustering by pattern similarity in large data sets. SIGMOD' 02.
- J. Yang and W. Wang. CLUSEQ: efficient and effective sequence clustering. ICDE'03.
- B. Fung, K. Wang, and M. Ester. Large Hierarchical Document Clustering Using Frequent Itemset. SDM'03.
- X. Yin and J. Han. CPAR: Classification based on Predictive Association Rules. SDM'03.

Ref: Stream and Privacy-Preserving FP Mining

- A. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke. Privacy Preserving Mining of Association Rules. KDD'02.
- J. Vaidya and C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. KDD'02.
- G. Manku and R. Motwani. Approximate Frequency Counts over Data Streams. VLDB'02.
- Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-Dimensional Regression Analysis of Time-Series Data Streams. VLDB'02.
- C. Giannella, J. Han, J. Pei, X. Yan and P. S. Yu. Mining Frequent Patterns in Data Streams at Multiple Time Granularities, Next Generation Data Mining:03.
- A. Evfimievski, J. Gehrke, and R. Srikant. Limiting Privacy Breaches in Privacy Preserving Data Mining. PODS'03.

Ref: Other Freq. Pattern Mining Applications

- Y. Huhtala, J. Kärkkäinen, P. Porkka, H. Toivonen. Efficient Discovery of Functional and Approximate Dependencies Using Partitions. ICDE'98.
- H. V. Jagadish, J. Madar, and R. Ng. Semantic Compression and Pattern Extraction with Fascicles. VLDB'99.
- T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining Database Structure; or How to Build a Data Quality Browser. SIGMOD'02.



Data Mining: Concepts and Techniques

— Chapter 6 —

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

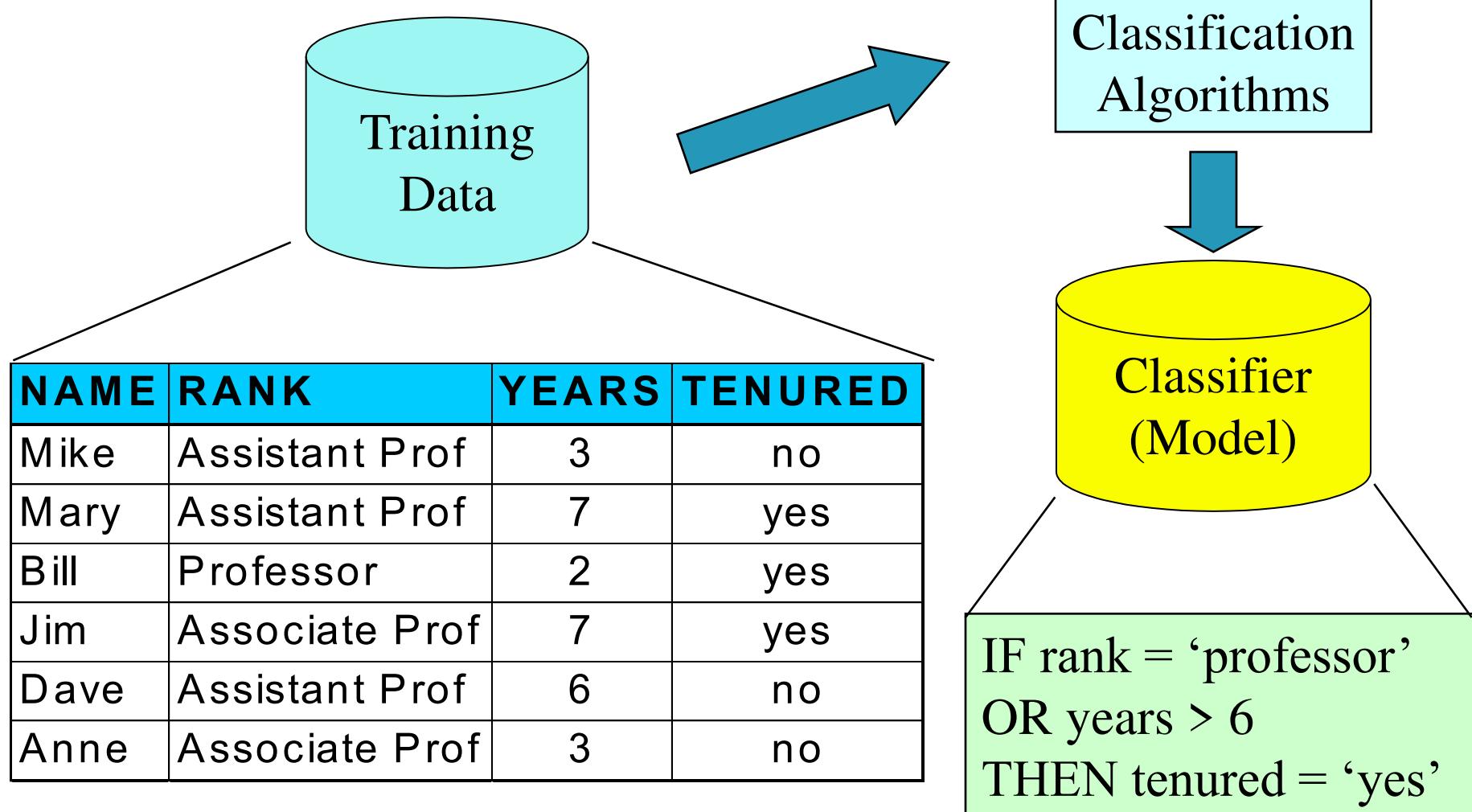
Classification vs. Prediction

- Classification
 - predicts categorical class labels (discrete or nominal)
 - classifies data (constructs a model) based on the training set and the values (**class labels**) in a classifying attribute and uses it in classifying new data
- Prediction
 - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical applications
 - Credit approval
 - Target marketing
 - Medical diagnosis
 - Fraud detection

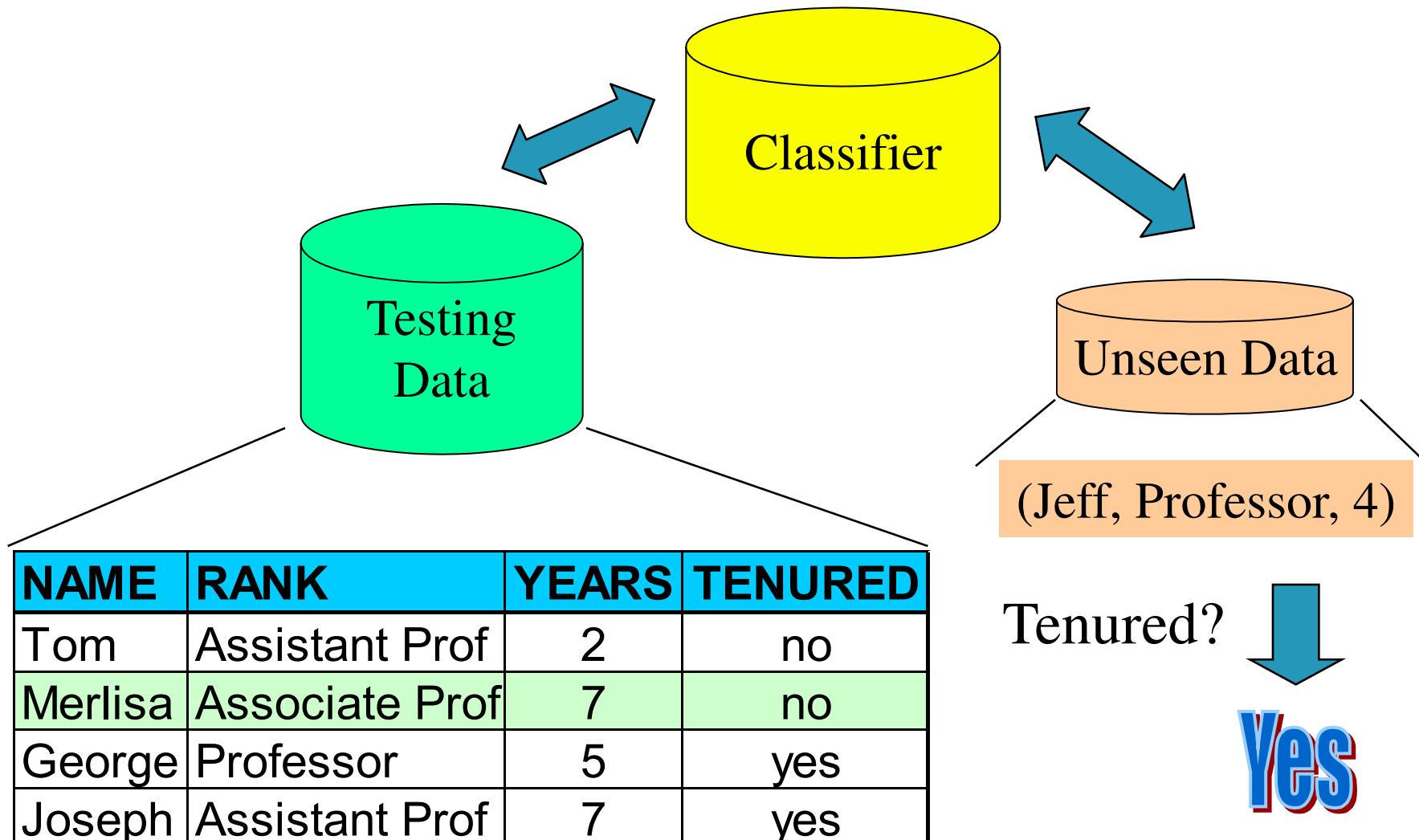
Classification—A Two-Step Process

- **Model construction:** describing a set of predetermined classes
 - Each tuple/sample is assumed to belong to a predefined class, as determined by the **class label attribute**
 - The set of tuples used for model construction is **training set**
 - The model is represented as classification rules, decision trees, or mathematical formulae
- **Model usage:** for classifying future or unknown objects
 - **Estimate accuracy** of the model
 - The known label of test sample is compared with the classified result from the model
 - Accuracy rate is the percentage of test set samples that are correctly classified by the model
 - Test set is independent of training set, otherwise over-fitting will occur
 - If the accuracy is acceptable, use the model to **classify data** tuples whose class labels are not known

Process (1): Model Construction



Process (2): Using the Model in Prediction



Supervised vs. Unsupervised Learning

- Supervised learning (classification)
 - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
 - New data is classified based on the training set
- Unsupervised learning (clustering)
 - The class labels of training data is unknown
 - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
 - Issues regarding classification and prediction
 - Classification by decision tree induction
 - Bayesian classification
 - Rule-based classification
 - Classification by back propagation
 - Support Vector Machines (SVM)
 - Associative classification
 - Lazy learners (or learning from your neighbors)
 - Other classification methods
 - Prediction
 - Accuracy and error measures
 - Ensemble methods
 - Model selection
 - Summary
- 

Issues: Data Preparation

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

Issues: Evaluating Classification Methods

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

Chapter 6. Classification and Prediction

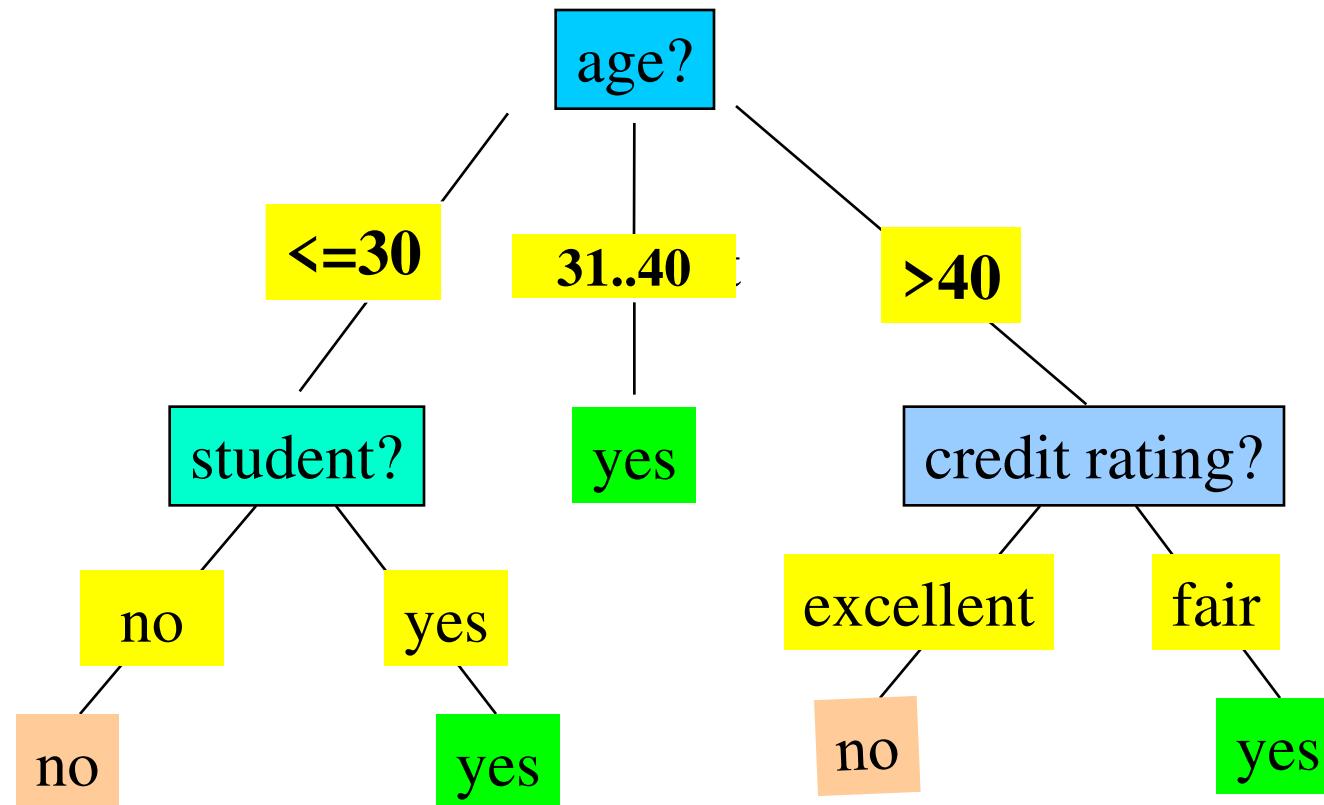
- What is classification? What is prediction?
 - Issues regarding classification and prediction
 - Classification by decision tree induction
 - Bayesian classification
 - Rule-based classification
 - Classification by back propagation
- 
- Support Vector Machines (SVM)
 - Associative classification
 - Lazy learners (or learning from your neighbors)
 - Other classification methods
 - Prediction
 - Accuracy and error measures
 - Ensemble methods
 - Model selection
 - Summary

Decision Tree Induction: Training Dataset

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

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

Output: A Decision Tree for “buys_computer”



Algorithm for Decision Tree Induction

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

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

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

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

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

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

- Information gained by branching on attribute A

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

Attribute Selection: Information Gain

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

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

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

$$+ \frac{5}{14} I(3,2) = 0.694$$

age	p_i	n_i	$I(p_i, n_i)$
≤ 30	2	3	0.971
31...40	4	0	0
>40	3	2	0.971

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

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

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit_rating) = 0.048$$

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
≤ 30	medium	no	fair	no
≤ 30	low	yes	fair	yes
>40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

Computing Information-Gain for Continuous-Value Attributes

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

Gain Ratio for Attribute Selection (C4.5)

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

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

- GainRatio(A) = Gain(A)/SplitInfo(A)
- Ex. $SplitInfo_A(D) = - \frac{4}{14} \times \log_2 \left(\frac{4}{14} \right) - \frac{6}{14} \times \log_2 \left(\frac{6}{14} \right) - \frac{4}{14} \times \log_2 \left(\frac{4}{14} \right) = 0.926$
 - $gain_ratio(income) = 0.029/0.926 = 0.031$
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini index (CART, IBM IntelligentMiner)

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

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

where p_j is the relative frequency of class j in D

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

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

- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

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

Gini index (CART, IBM IntelligentMiner)

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

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

- Suppose the attribute income partitions D into 10 in D_1 : {low, medium} and 4 in D_2

$$\begin{aligned} gini_{income \in \{low, medium\}}(D) &= \left(\frac{10}{14} \right) Gini(D_1) + \left(\frac{4}{14} \right) Gini(D_2) \\ &= \frac{10}{14} \left(1 - \left(\frac{6}{10} \right)^2 - \left(\frac{4}{10} \right)^2 \right) + \frac{4}{14} \left(1 - \left(\frac{1}{4} \right)^2 - \left(\frac{3}{4} \right)^2 \right) \\ &= 0.450 \\ &= Gini_{income \in \{high\}}(D) \end{aligned}$$

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

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

Comparing Attribute Selection Measures

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

Overfitting and Tree Pruning

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

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
 - Issues regarding classification and prediction
 - Classification by decision tree induction
 - Bayesian classification
 - Rule-based classification
 - Classification by back propagation
- 
- Support Vector Machines (SVM)
 - Associative classification
 - Lazy learners (or learning from your neighbors)
 - Other classification methods
 - Prediction
 - Accuracy and error measures
 - Ensemble methods
 - Model selection
 - Summary

Bayesian Classification: Why?

- A statistical classifier: performs *probabilistic prediction*, i.e., predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayesian Theorem: Basics

- Let \mathbf{X} be a data sample ("*evidence*"): class label is unknown
- Let H be a *hypothesis* that X belongs to class C
- Classification is to determine $P(H|\mathbf{X})$, the probability that the hypothesis holds given the observed data sample \mathbf{X}
- $P(H)$ (*prior probability*), the initial probability
 - E.g., \mathbf{X} will buy computer, regardless of age, income, ...
- $P(\mathbf{X})$: probability that sample data is observed
- $P(\mathbf{X}|H)$ (*posteriori probability*), the probability of observing the sample \mathbf{X} , given that the hypothesis holds
 - E.g., Given that \mathbf{X} will buy computer, the prob. that X is 31..40, medium income

Bayesian Theorem

- Given training data \mathbf{X} , *posteriori probability of a hypothesis H*, $P(H|\mathbf{X})$, follows the Bayes theorem

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

- Informally, this can be written as
posteriori = likelihood x prior/evidence
- Predicts \mathbf{X} belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|\mathbf{X})$ for all the k classes
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Towards Naïve Bayesian Classifier

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n -D attribute vector $\mathbf{X} = (x_1, x_2, \dots, x_n)$
- Suppose there are m classes C_1, C_2, \dots, C_m .
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

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

- Since $P(X)$ is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Derivation of Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

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

- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k | C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$ (# of tuples of C_i in D)
- If A_k is continuous-valued, $P(x_k | C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

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

and $P(x_k | C_i)$ is

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

Naïve Bayesian Classifier: Training Dataset

Class:

C1:`buys_computer = 'yes'`

C2:`buys_computer = 'no'`

Data sample

X = (age ≤ 30 ,
Income = medium,
Student = yes
Credit_rating = Fair)

	age	income	student	credit_rating	buys_computer
	≤ 30	high	no	fair	no
	≤ 30	high	no	excellent	no
	31...40	high	no	fair	yes
	> 40	medium	no	fair	yes
	> 40	low	yes	fair	yes
	> 40	low	yes	excellent	no
	31...40	low	yes	excellent	yes
	≤ 30	medium	no	fair	no
	≤ 30	low	yes	fair	yes
	> 40	medium	yes	fair	yes
	≤ 30	medium	yes	excellent	yes
	31...40	medium	no	excellent	yes
	31...40	high	yes	fair	yes
	> 40	medium	no	excellent	no

Naïve Bayesian Classifier: An Example

- $P(C_i)$:
 $P(\text{buys_computer} = \text{"yes"}) = 9/14 = 0.643$
 $P(\text{buys_computer} = \text{"no"}) = 5/14 = 0.357$
- Compute $P(X|C_i)$ for each class
 $P(\text{age} = \text{"<=30"} | \text{buys_computer} = \text{"yes"}) = 2/9 = 0.222$
 $P(\text{age} = \text{"<= 30"} | \text{buys_computer} = \text{"no"}) = 3/5 = 0.6$
 $P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"yes"}) = 4/9 = 0.444$
 $P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$
 $P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$
 $P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"no"}) = 1/5 = 0.2$
 $P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$
 $P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$$\mathbf{P(X|C_i)} : P(X|\text{buys_computer} = \text{"yes"}) = 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$$
$$P(X|\text{buys_computer} = \text{"no"}) = 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$$

$$\mathbf{P(X|C_i)*P(C_i)} : P(X|\text{buys_computer} = \text{"yes"}) * P(\text{buys_computer} = \text{"yes"}) = 0.028$$
$$P(X|\text{buys_computer} = \text{"no"}) * P(\text{buys_computer} = \text{"no"}) = 0.007$$

Therefore, X belongs to class ("buys_computer = yes")

Avoiding the 0-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be non-zero. Otherwise, the predicted prob. will be zero

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

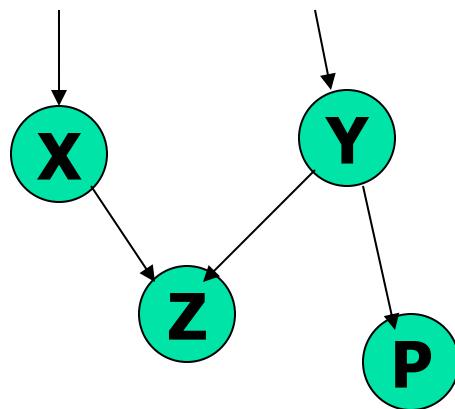
- Ex. Suppose a dataset with 1000 tuples, income=low (0), income=medium (990), and income = high (10),
- Use Laplacian correction (or Laplacian estimator)
 - Adding 1 to each case
 - Prob(income = low) = 1/1003
 - Prob(income = medium) = 991/1003
 - Prob(income = high) = 11/1003
 - The “corrected” prob. estimates are close to their “uncorrected” counterparts

Naïve Bayesian Classifier: Comments

- Advantages
 - Easy to implement
 - Good results obtained in most of the cases
- Disadvantages
 - Assumption: class conditional independence, therefore loss of accuracy
 - Practically, dependencies exist among variables
 - E.g., hospitals: patients: Profile: age, family history, etc.
 - Symptoms: fever, cough etc., Disease: lung cancer, diabetes, etc.
 - Dependencies among these cannot be modeled by Naïve Bayesian Classifier
- How to deal with these dependencies?
 - Bayesian Belief Networks

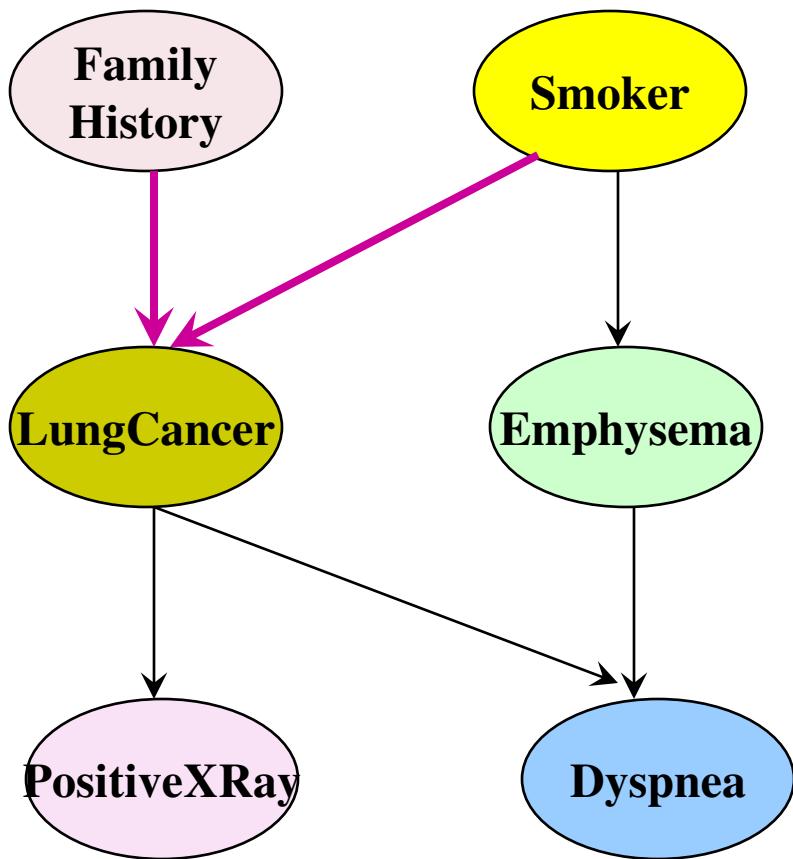
Bayesian Belief Networks

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



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

Bayesian Belief Network: An Example



The **conditional probability table (CPT)** for variable LungCancer:

	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

CPT shows the conditional probability for each possible combination of its parents

Derivation of the probability of a particular combination of values of \mathbf{X} , from CPT:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | Parents(Y_i))$$

Bayesian Belief Networks

Training Bayesian Networks

- Several scenarios:
 - Given both the network structure and all variables observable: *learn only the CPTs*
 - Network structure known, some hidden variables: *gradient descent* (greedy hill-climbing) method, analogous to neural network learning
 - Network structure unknown, all variables observable: search through the model space to *reconstruct network topology*
 - Unknown structure, all hidden variables: No good algorithms known for this purpose
- Ref. D. Heckerman: Bayesian networks for data mining

Chapter 6. Classification and Prediction

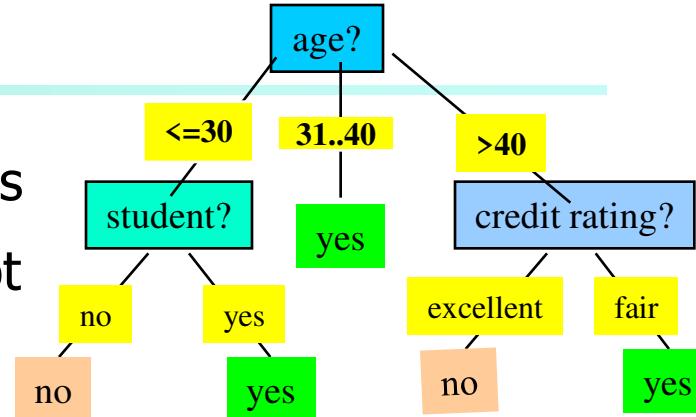
- What is classification? What is prediction?
 - Issues regarding classification and prediction
 - Classification by decision tree induction
 - Bayesian classification
 - Rule-based classification
 - Classification by back propagation
- 
- Support Vector Machines (SVM)
 - Associative classification
 - Lazy learners (or learning from your neighbors)
 - Other classification methods
 - Prediction
 - Accuracy and error measures
 - Ensemble methods
 - Model selection
 - Summary

Using IF-THEN Rules for Classification

- Represent the knowledge in the form of **IF-THEN** rules
 - R: IF *age* = youth AND *student* = yes THEN *buys_computer* = yes
 - Rule antecedent/precondition vs. rule consequent
- Assessment of a rule: *coverage* and *accuracy*
 - $n_{\text{covers}} = \# \text{ of tuples covered by } R$
 - $n_{\text{correct}} = \# \text{ of tuples correctly classified by } R$
$$\text{coverage}(R) = n_{\text{covers}} / |D| \quad /* D: training data set */$$
$$\text{accuracy}(R) = n_{\text{correct}} / n_{\text{covers}}$$
- If more than one rule is triggered, need **conflict resolution**
 - Size ordering: assign the highest priority to the triggering rules that has the “toughest” requirement (i.e., with the *most attribute test*)
 - Class-based ordering: decreasing order of *prevalence or misclassification cost per class*
 - Rule-based ordering (**decision list**): rules are organized into one long priority list, according to some measure of rule quality or by experts

Rule Extraction from a Decision Tree

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



IF <i>age</i> = young AND <i>student</i> = <i>no</i>	THEN <i>buys_computer</i> = <i>no</i>
IF <i>age</i> = young AND <i>student</i> = <i>yes</i>	THEN <i>buys_computer</i> = <i>yes</i>
IF <i>age</i> = mid-age	THEN <i>buys_computer</i> = <i>yes</i>
IF <i>age</i> = old AND <i>credit_rating</i> = <i>excellent</i>	THEN <i>buys_computer</i> = <i>yes</i>
IF <i>age</i> = young AND <i>credit_rating</i> = <i>fair</i>	THEN <i>buys_computer</i> = <i>no</i>

Chapter 6. Classification and Prediction

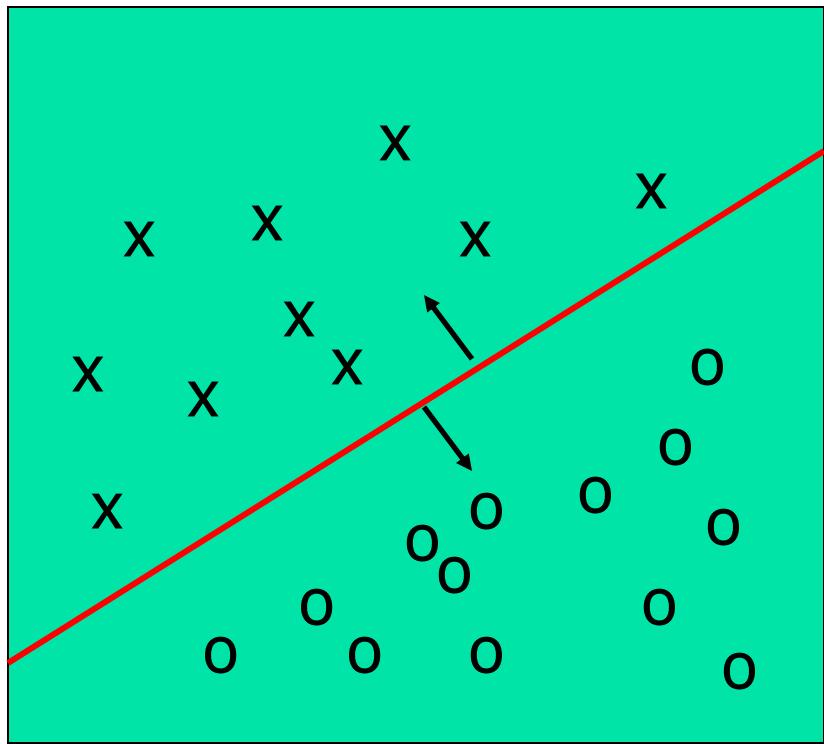
- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary



Classification: A Mathematical Mapping

- Classification:
 - predicts categorical class labels
- E.g., Personal homepage classification
 - $x_i = (x_1, x_2, x_3, \dots)$, $y_i = +1$ or -1
 - x_1 : # of a word “homepage”
 - x_2 : # of a word “welcome”
- Mathematically
 - $x \in X = \Re^n$, $y \in Y = \{+1, -1\}$
 - We want a function $f: X \rightarrow Y$

Linear Classification

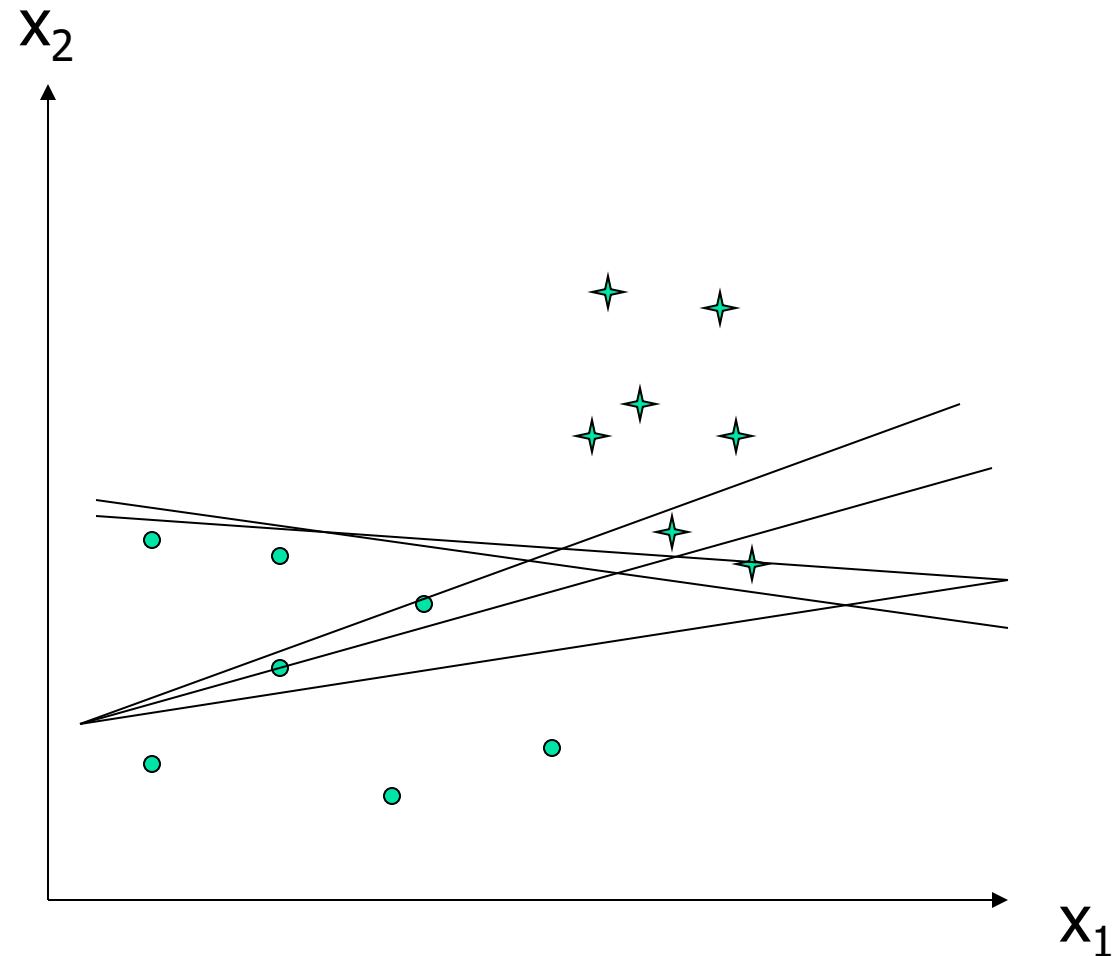


- Binary Classification problem
- The data above the red line belongs to class 'x'
- The data below red line belongs to class 'o'
- Examples: SVM, Perceptron, Probabilistic Classifiers

Discriminative Classifiers

- Advantages
 - prediction accuracy is generally high
 - As compared to Bayesian methods – in general
 - robust, works when training examples contain errors
 - fast evaluation of the learned target function
 - Bayesian networks are normally slow
- Criticism
 - long training time
 - difficult to understand the learned function (weights)
 - Bayesian networks can be used easily for pattern discovery
 - not easy to incorporate domain knowledge
 - Easy in the form of priors on the data or distributions

Perceptron & Winnow



- Vector: x, w

- Scalar: x, y, w

Input: $\{(x_1, y_1), \dots\}$

Output: classification function $f(x)$

$$f(x_i) > 0 \text{ for } y_i = +1$$

$$f(x_i) < 0 \text{ for } y_i = -1$$

$$f(x) \Rightarrow wx + b = 0$$

$$\text{or } w_1x_1 + w_2x_2 + b = 0$$

- Perceptron: update W additively
- Winnow: update W multiplicatively

Classification by Backpropagation

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

Neural Network as a Classifier

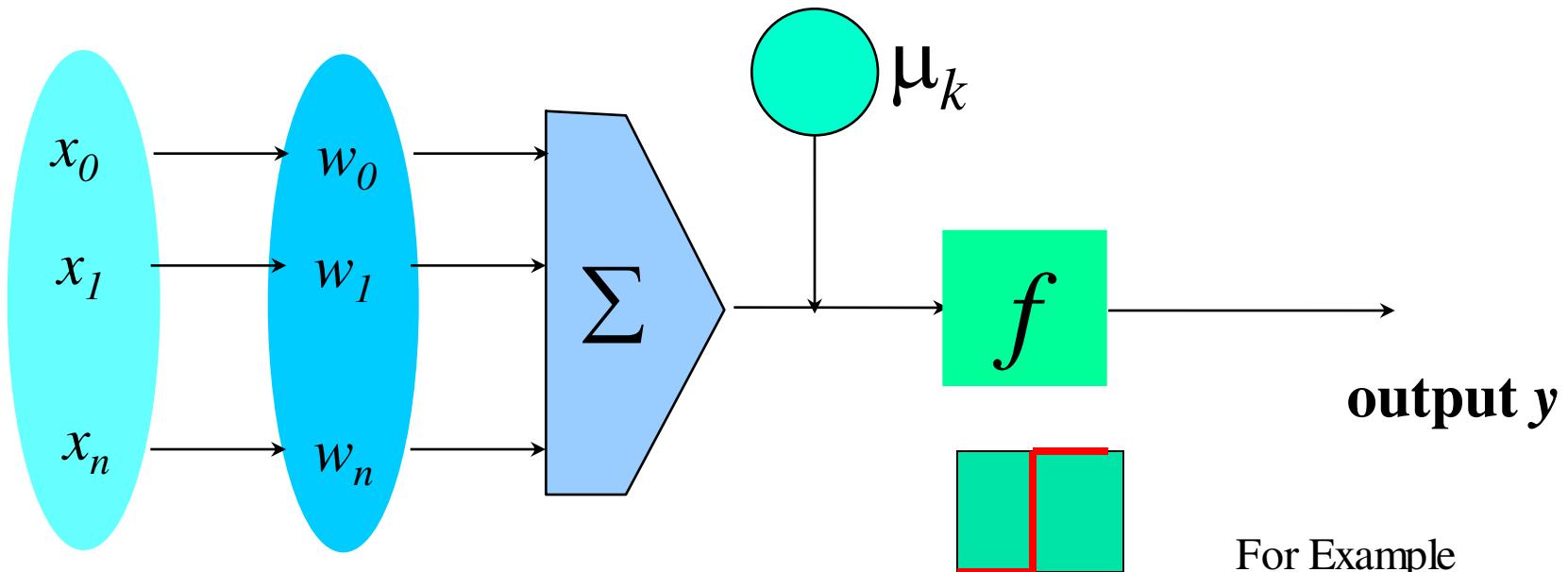
■ Weakness

- Long training time
- Require a number of parameters typically best determined empirically, e.g., the network topology or ``structure."
- Poor interpretability: Difficult to interpret the symbolic meaning behind the learned weights and of ``hidden units" in the network

■ Strength

- High tolerance to noisy data
- Ability to classify untrained patterns
- Well-suited for continuous-valued inputs and outputs
- Successful on a wide array of real-world data
- Algorithms are inherently parallel
- Techniques have recently been developed for the extraction of rules from trained neural networks

A Neuron (= a perceptron)



Input vector \mathbf{x} **weight vector \mathbf{w}** **weighted sum** **Activation function**

For Example

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

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

A Multi-Layer Feed-Forward Neural Network

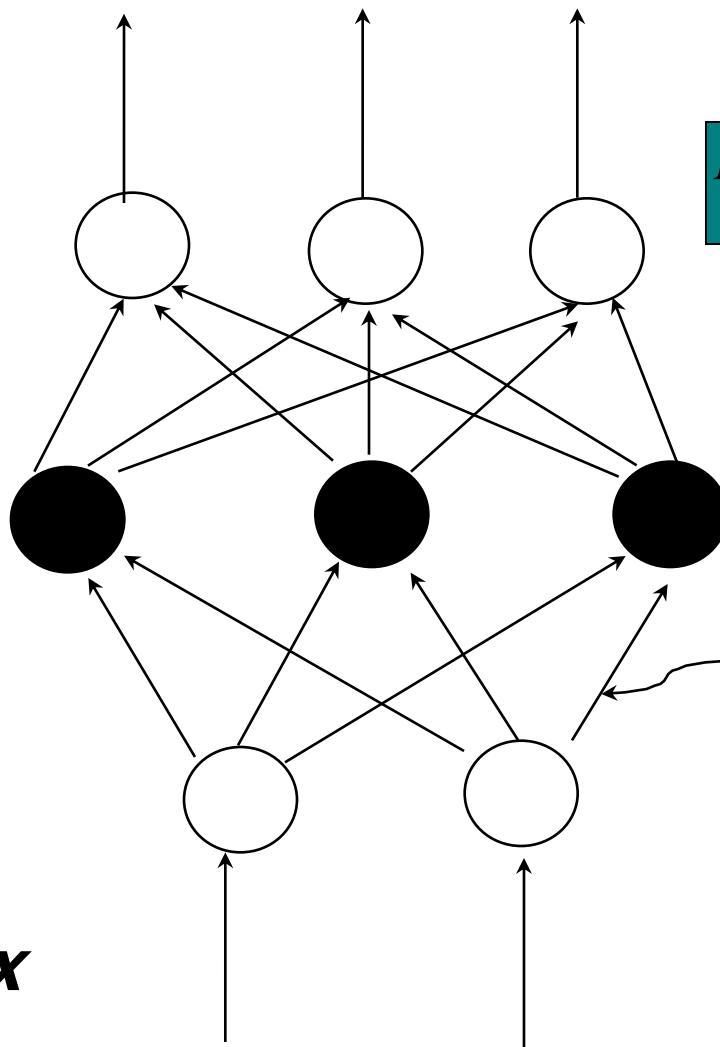
Output vector

Output layer

Hidden layer

Input layer

Input vector: X



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

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

$$w_{ij} = w_{ij} + (l) Err_j O_i$$

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

w_{ij}

$$O_j = \frac{1}{1+e^{-I_j}}$$

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

How A Multi-Layer Neural Network Works?

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

Defining a Network Topology

- First decide the **network topology**: # of units in the *input layer*, # of *hidden layers* (if > 1), # of units in *each hidden layer*, and # of units in the *output layer*
- Normalizing the input values for each attribute measured in the training tuples to [0.0—1.0]
- One **input** unit per domain value, each initialized to 0
- **Output**, if for classification and more than two classes, one output unit per class is used
- Once a network has been trained and its accuracy is **unacceptable**, repeat the training process with a *different network topology* or a *different set of initial weights*

Backpropagation

- Iteratively process a set of training tuples & compare the network's prediction with the actual known target value
- For each training tuple, the weights are modified to **minimize the mean squared error** between the network's prediction and the actual target value
- Modifications are made in the "**backwards**" direction: from the output layer, through each hidden layer down to the first hidden layer, hence "**backpropagation**"
- Steps
 - Initialize weights (to small random #'s) and biases in the network
 - Propagate the inputs forward (by applying activation function)
 - Backpropagate the error (by updating weights and biases)
 - Terminating condition (when error is very small, etc.)

Backpropagation and Interpretability

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

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM) 
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

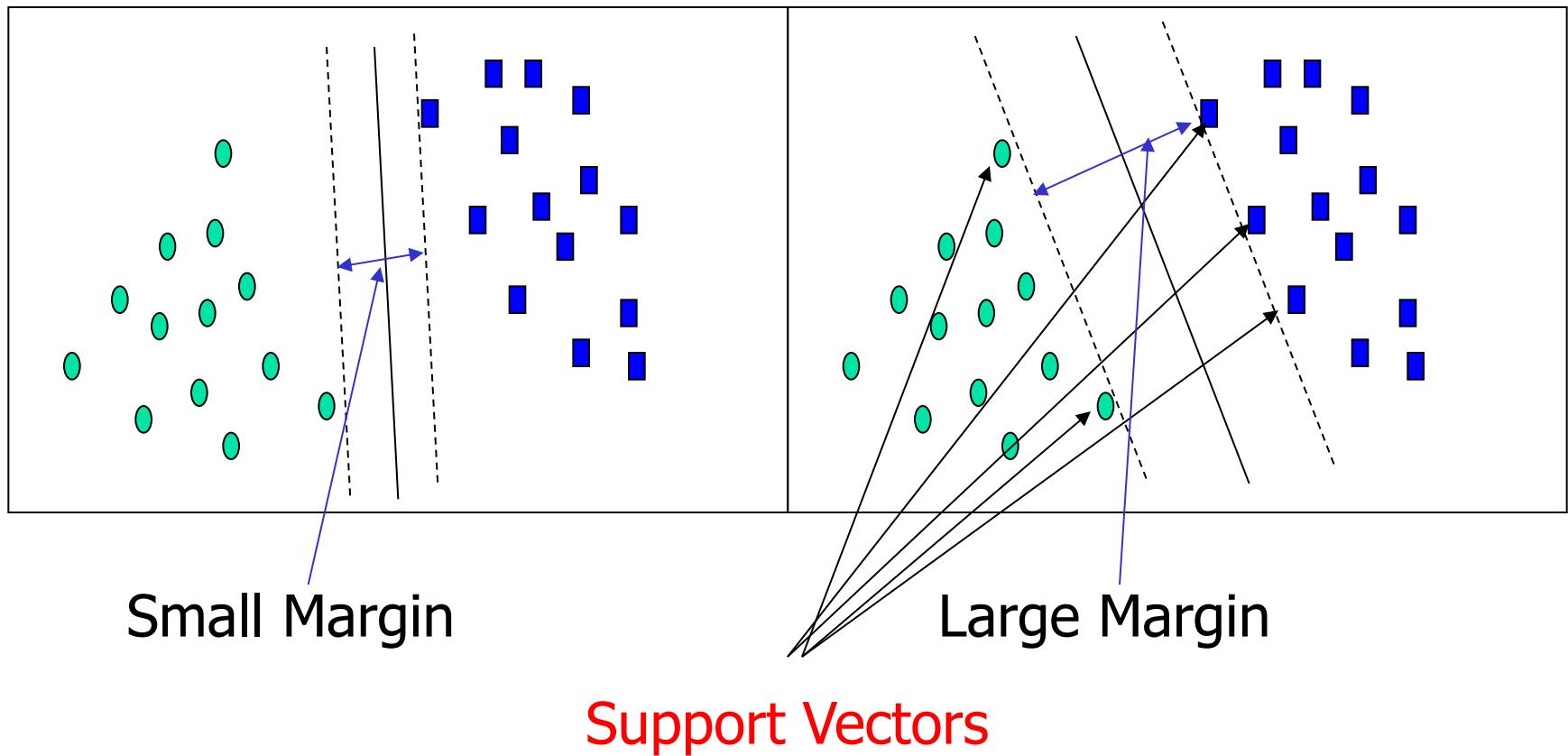
SVM—Support Vector Machines

- A new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using support vectors (“essential” training tuples) and margins (defined by the support vectors)

SVM—History and Applications

- Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s
- Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Used both for classification and prediction
- Applications:
 - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests

SVM—General Philosophy



SVM—Margins and Support Vectors

A_2

small margin

- class 1, $y = +1$ (*buys_computer* = "yes")
- class 2, $y = -1$ (*buys_computer* = "no")

A_1

A_2

large margin

- class 1,
- class 2,

A_1

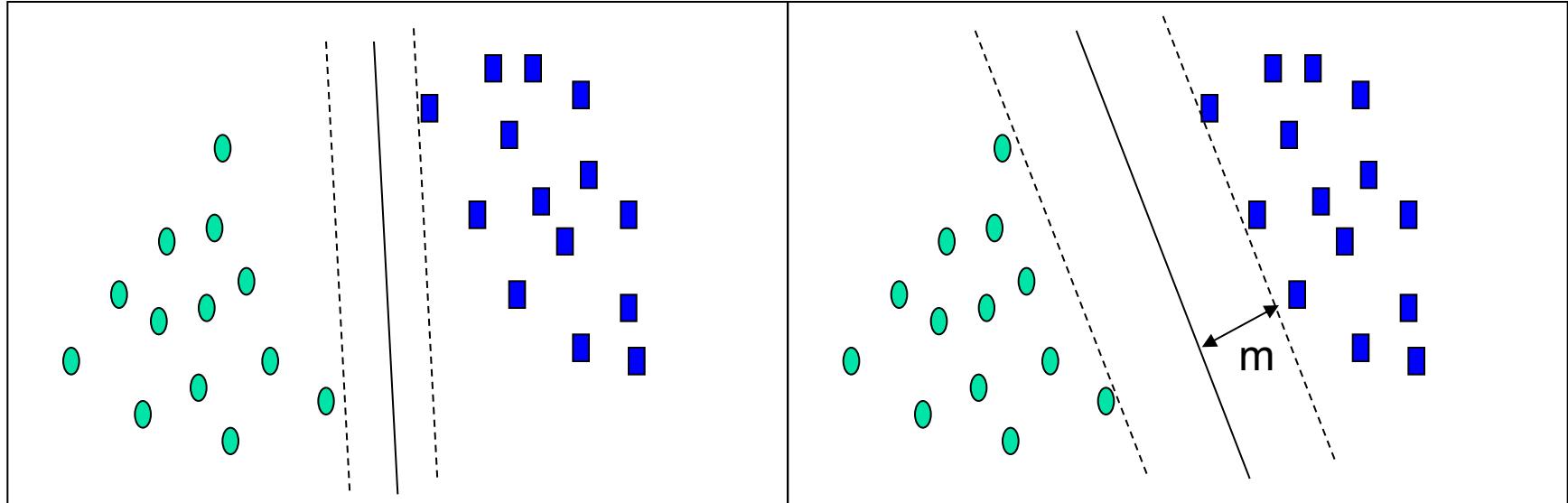
A_2

large margin

- c
- c

A_1

SVM—When Data Is Linearly Separable



Let data D be $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_{|D|}, y_{|D|})$, where \mathbf{X}_i is the set of training tuples associated with the class labels y_i

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane (MMH)**

SVM—Linearly Separable

- A separating hyperplane can be written as

$$\mathbf{W} \bullet \mathbf{X} + b = 0$$

where $\mathbf{W}=\{w_1, w_2, \dots, w_n\}$ is a weight vector and b a scalar (bias)

- For 2-D it can be written as

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

- The hyperplane defining the sides of the margin:

$$H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1 \quad \text{for } y_i = +1, \text{ and}$$

$$H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1 \quad \text{for } y_i = -1$$

- Any training tuples that fall on hyperplanes H_1 or H_2 (i.e., the sides defining the margin) are **support vectors**
- This becomes a **constrained (convex) quadratic optimization** problem: Quadratic objective function and linear constraints → *Quadratic Programming (QP)* → Lagrangian multipliers

Why Is SVM Effective on High Dimensional Data?

- The complexity of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data
- The support vectors are the essential or critical training examples — they lie closest to the decision boundary (MMH)
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

SVM—Linearly Inseparable

- Transform the original input data into a higher dimensional space

Example 6.8 Nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $\mathbf{X} = (x_1, x_2, x_3)$ is mapped into a 6D space \mathbf{Z} using the mappings $\phi_1(\mathbf{X}) = x_1, \phi_2(\mathbf{X}) = x_2, \phi_3(\mathbf{X}) = x_3, \phi_4(\mathbf{X}) = (x_1)^2, \phi_5(\mathbf{X}) = x_1x_2$, and $\phi_6(\mathbf{X}) = x_1x_3$. A decision hyperplane in the new space is $d(\mathbf{Z}) = \mathbf{W}\mathbf{Z} + b$, where \mathbf{W} and \mathbf{Z} are vectors. This is linear. We solve for \mathbf{W} and b and then substitute back so that we see that the linear decision hyperplane in the new (\mathbf{Z}) space corresponds to a nonlinear second order polynomial in the original 3-D input space,

$$\begin{aligned} d(\mathbf{Z}) &= w_1z_1 + w_2z_2 + w_3z_3 + w_4(z_1)^2 + w_5z_4 + w_6z_5 + b \\ &= w_1x_1 + w_2x_2 + w_3x_3 + w_4(x_1)^2 + w_5x_1x_2 + w_6x_1x_3 + b \end{aligned}$$

- Search for a linear separating hyperplane in the new space

SVM—Kernel functions

- Instead of computing the dot product on the transformed data tuples, it is mathematically equivalent to instead applying a kernel function $K(\mathbf{X}_i, \mathbf{X}_j)$ to the original data, i.e., $K(\mathbf{X}_i, \mathbf{X}_j) = \Phi(\mathbf{X}_i) \cdot \Phi(\mathbf{X}_j)$
- Typical Kernel Functions

$$\text{Polynomial kernel of degree } h : K(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i \cdot \mathbf{X}_j + 1)^h$$

$$\text{Gaussian radial basis function kernel} : K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\|\mathbf{X}_i - \mathbf{X}_j\|^2 / 2\sigma^2}$$

$$\text{Sigmoid kernel} : K(\mathbf{X}_i, \mathbf{X}_j) = \tanh(\kappa \mathbf{X}_i \cdot \mathbf{X}_j - \delta)$$

- SVM can also be used for classifying multiple (> 2) classes and for regression analysis (with additional user parameters)

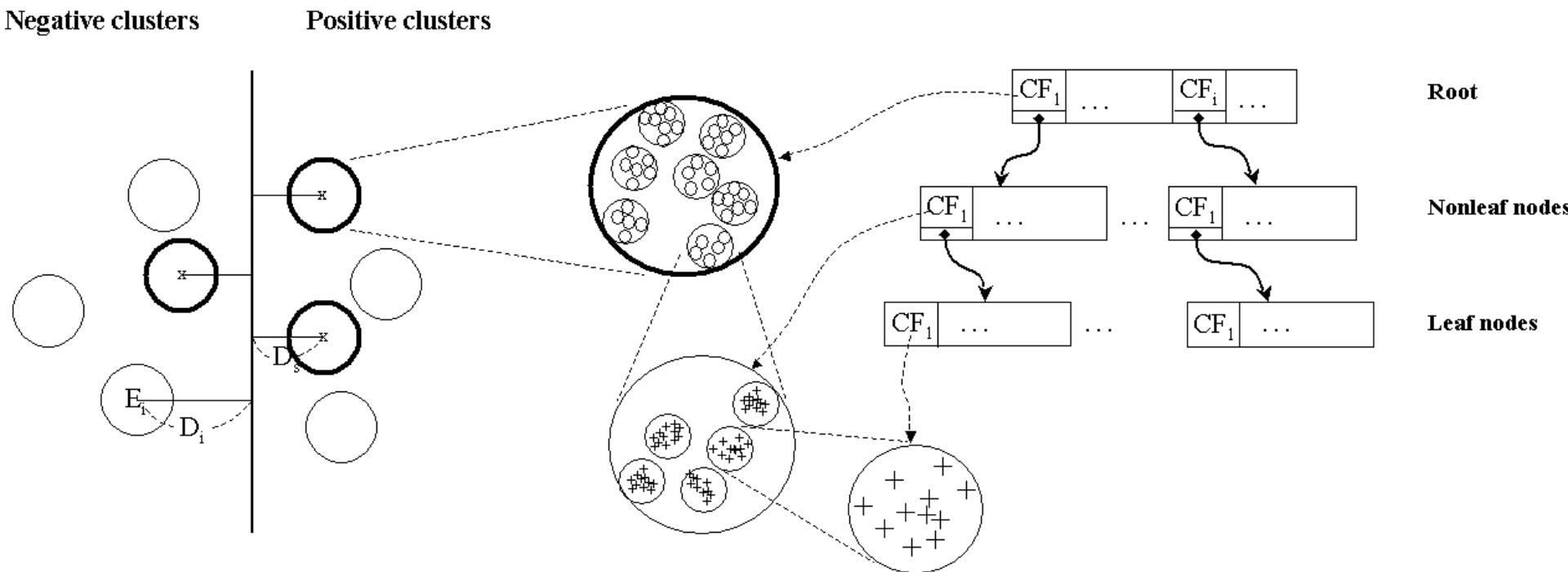
Scaling SVM by Hierarchical Micro-Clustering

- SVM is not scalable to the number of data objects in terms of training time and memory usage
- “Classifying Large Datasets Using SVMs with Hierarchical Clusters Problem” by Hwanjo Yu, Jiong Yang, Jiawei Han, KDD’03
- CB-SVM (Clustering-Based SVM)
 - Given limited amount of system resources (e.g., memory), maximize the SVM performance in terms of accuracy and the training speed
 - Use micro-clustering to effectively reduce the number of points to be considered
 - At deriving support vectors, de-cluster micro-clusters near “candidate vector” to ensure high classification accuracy

CB-SVM: Clustering-Based SVM

- Training data sets may not even fit in memory
- Read the data set once (minimizing disk access)
 - Construct a statistical summary of the data (i.e., hierarchical clusters) given a limited amount of memory
 - The statistical summary maximizes the benefit of learning SVM
- The summary plays a role in indexing SVMs
- Essence of Micro-clustering (Hierarchical indexing structure)
 - Use micro-cluster hierarchical indexing structure
 - provide finer samples closer to the boundary and coarser samples farther from the boundary
 - Selective de-clustering to ensure high accuracy

CF-Tree: Hierarchical Micro-cluster

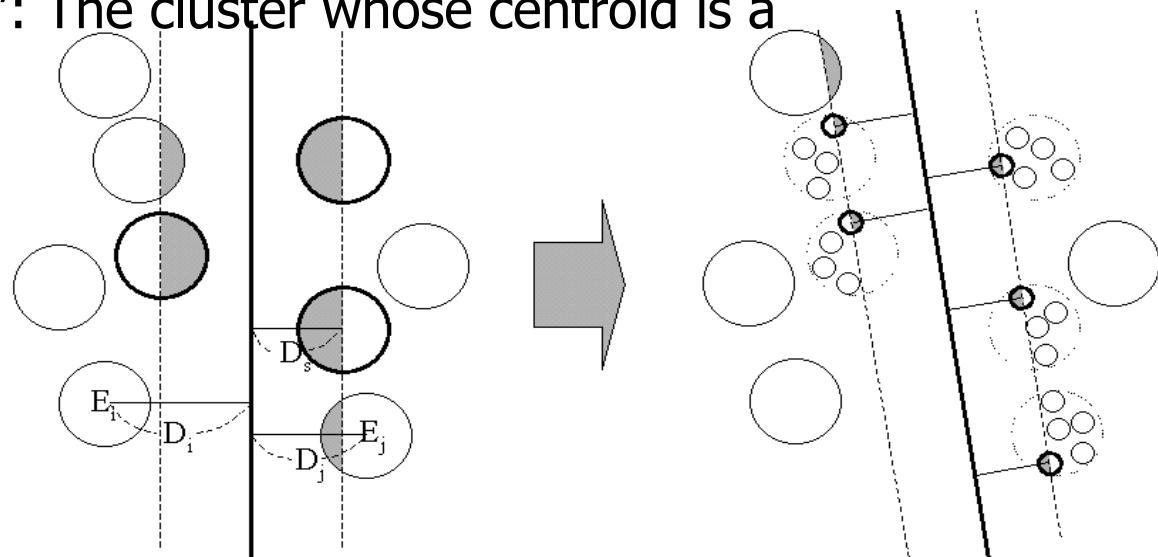


CB-SVM Algorithm: Outline

- Construct two CF-trees from positive and negative data sets independently
 - Need one scan of the data set
- Train an SVM from the centroids of the root entries
- De-cluster the entries near the boundary into the next level
 - The children entries de-clustered from the parent entries are accumulated into the training set with the non-declustered parent entries
- Train an SVM again from the centroids of the entries in the training set
- Repeat until nothing is accumulated

Selective Declustering

- CF tree is a suitable base structure for selective declustering
- De-cluster only the cluster E_i such that
 - $D_i - R_i < D_s$, where D_i is the distance from the boundary to the center point of E_i and R_i is the radius of E_i
 - Decluster only the cluster whose subclusters have possibilities to be the support cluster of the boundary
 - “Support cluster”: The cluster whose centroid is a support vector



Experiment on Synthetic Dataset

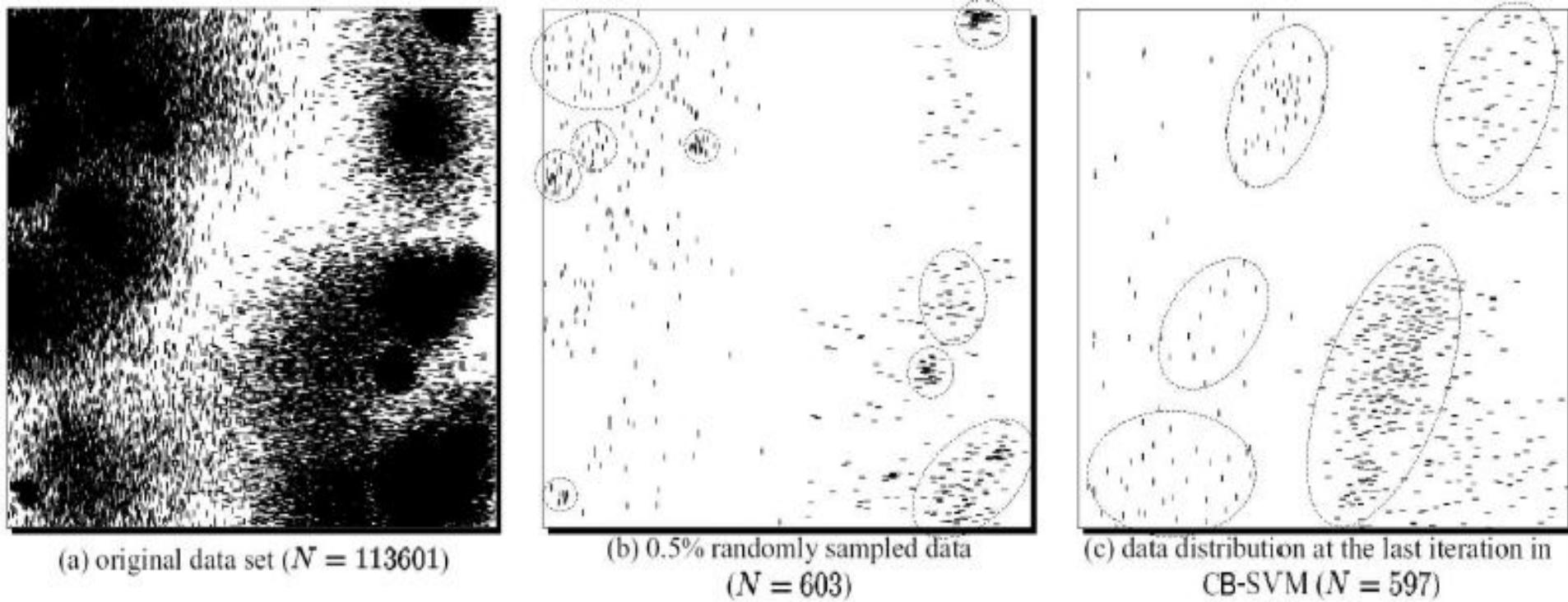


Figure 6: Synthetic data set in a two-dimensional space. '|': positive data; '—': negative data

Experiment on a Large Data Set

S-Rate	# of data	# of errors	T-Time	S-Time
0.0001%	23	6425	0.000114	822.97
0.001%	226	2413	0.000972	825.40
0.01%	2333	1132	0.03	828.61
0.1%	23273	1012	6.287	835.87
1%	230380	1015	1192.793	838.92
5%	1151714	1020	20705.4	842.92
ASVM	307	865		54872.213
CB-SVM	2893	876	1.639	2528.213

Table 4: Performance results on the very large data set (# of training data = 23066169, # of testing data = 233890). S-Rate: sampling rate; T-Time: training time; S-Time: sampling time; ASVM: selective sampling

SVM vs. Neural Network

- SVM
 - Relatively new concept
 - Deterministic algorithm
 - Nice Generalization properties
 - Hard to learn – learned in batch mode using quadratic programming techniques
 - Using kernels can learn very complex functions
- Neural Network
 - Relatively old
 - Nondeterministic algorithm
 - Generalizes well but doesn't have strong mathematical foundation
 - Can easily be learned in incremental fashion
 - To learn complex functions—use multilayer perceptron (not that trivial)

SVM Related Links

- SVM Website
 - <http://www.kernel-machines.org/>
- Representative implementations
 - LIBSVM: an efficient implementation of SVM, multi-class classifications, nu-SVM, one-class SVM, including also various interfaces with java, python, etc.
 - SVM-light: simpler but performance is not better than LIBSVM, support only binary classification and only C language
 - SVM-torch: another recent implementation also written in C.

SVM—Introduction Literature

- “Statistical Learning Theory” by Vapnik: extremely hard to understand, containing many errors too.
- C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
 - Better than the Vapnik’s book, but still written too hard for introduction, and the examples are so not-intuitive
- The book “An Introduction to Support Vector Machines” by N. Cristianini and J. Shawe-Taylor
 - Also written hard for introduction, but the explanation about the mercer’s theorem is better than above literatures
- The neural network book by Haykins
 - Contains one nice chapter of SVM introduction

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary



Associative Classification

- Associative classification
 - Association rules are generated and analyzed for use in classification
 - Search for strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels
 - Classification: Based on evaluating a set of rules in the form of
$$P_1 \wedge p_2 \dots \wedge p_l \rightarrow "A_{\text{class}} = C" \text{ (conf, sup)}$$
- Why effective?
 - It explores highly confident associations among multiple attributes and may overcome some constraints introduced by decision-tree induction, which considers only one attribute at a time
 - In many studies, associative classification has been found to be more accurate than some traditional classification methods, such as C4.5

Typical Associative Classification Methods

- CBA (Classification By Association: Liu, Hsu & Ma, KDD'98)
 - Mine association possible rules in the form of
 - Cond-set (a set of attribute-value pairs) → class label
 - Build classifier: Organize rules according to decreasing precedence based on confidence and then support
- CMAR (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
 - Classification: Statistical analysis on multiple rules
- CPAR (Classification based on Predictive Association Rules: Yin & Han, SDM'03)
 - Generation of predictive rules (FOIL-like analysis)
 - High efficiency, accuracy similar to CMAR
- RCBT (Mining top- k covering rule groups for gene expression data, Cong et al. SIGMOD'05)
 - Explore high-dimensional classification, using top- k rule groups
 - Achieve high classification accuracy and high run-time efficiency

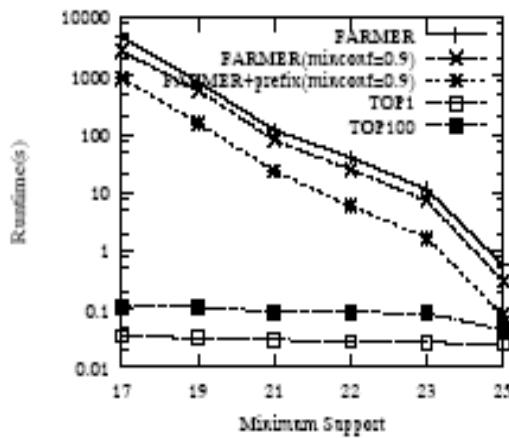
A Closer Look at CMAR

- CMAR (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
- Efficiency: Uses an enhanced FP-tree that maintains the distribution of class labels among tuples satisfying each frequent itemset
- Rule pruning whenever a rule is inserted into the tree
 - Given two rules, R_1 and R_2 , if the antecedent of R_1 is more general than that of R_2 and $\text{conf}(R_1) \geq \text{conf}(R_2)$, then R_2 is pruned
 - Prunes rules for which the rule antecedent and class are not positively correlated, based on a χ^2 test of statistical significance
- Classification based on generated/pruned rules
 - If only one rule satisfies tuple X, assign the class label of the rule
 - If a rule set S satisfies X, CMAR
 - divides S into groups according to class labels
 - uses a weighted χ^2 measure to find the strongest group of rules, based on the statistical correlation of rules within a group
 - assigns X the class label of the strongest group

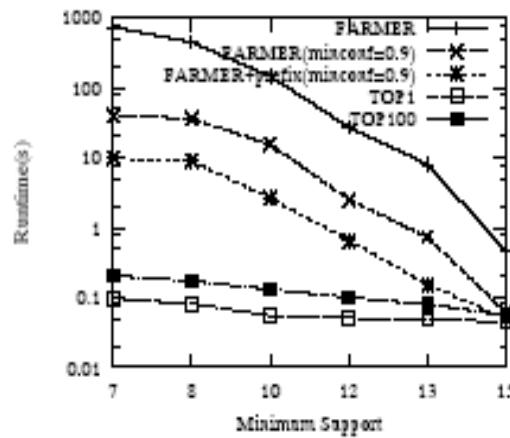
Associative Classification May Achieve High Accuracy and Efficiency (Cong et al. SIGMOD05)

Dataset	RCBT	CBA	IRG Classifier	C4.5 family			SVM
				single tree	bagging	boosting	
AML/ALL (ALL)	91.18%	91.18%	64.71%	91.18%	91.18%	91.18%	97.06%
Lung Cancer(LC)	97.99%	81.88%	89.93%	81.88%	96.64%	81.88%	96.64%
Ovarian Cancer(OC)	97.67%	93.02%	-	97.67%	97.67%	97.67%	97.67%
Prostate Cancer(PC)	97.06%	82.35%	88.24%	26.47%	26.47%	26.47%	79.41%
Average Accuracy	95.98%	87.11%	80.96%	74.3%	77.99%	74.3%	92.70%

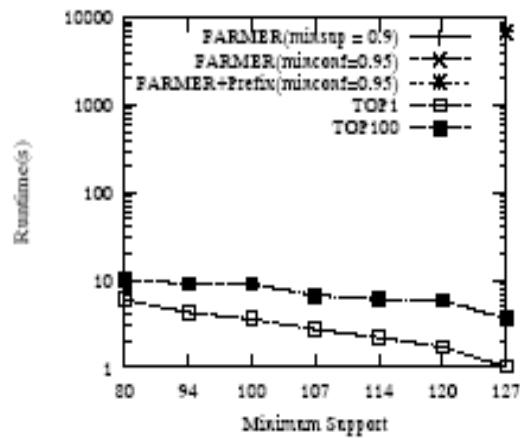
Table 2: Classification Results



(a) ALL-AML leukemia



(b) Lung Cancer



(c) Ovarian Cancer

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors) 
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

Lazy vs. Eager Learning

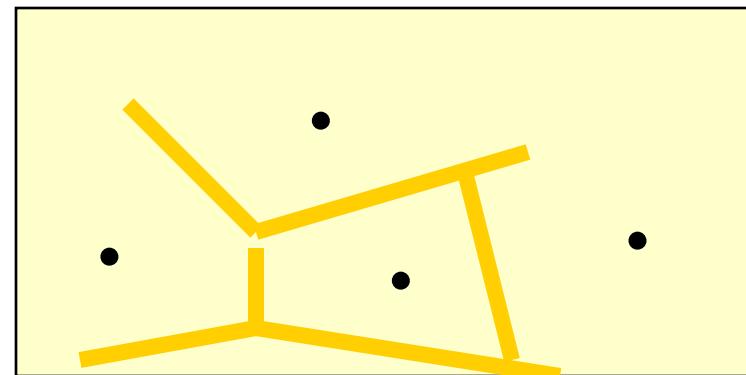
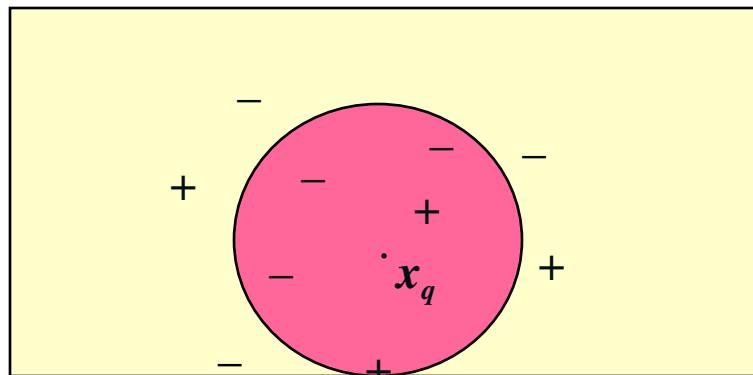
- Lazy vs. eager learning
 - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
 - Eager learning (the above discussed methods): Given a set of training set, constructs a classification model before receiving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

Lazy Learner: Instance-Based Methods

- Instance-based learning:
 - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Typical approaches
 - k -nearest neighbor approach
 - Instances represented as points in a Euclidean space.
 - Locally weighted regression
 - Constructs local approximation
 - Case-based reasoning
 - Uses symbolic representations and knowledge-based inference

The k -Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, $\text{dist}(\mathbf{X}_1, \mathbf{X}_2)$
- Target function could be discrete- or real- valued
- For discrete-valued, k -NN returns the most common value among the k training examples nearest to x_q
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples



Discussion on the k -NN Algorithm

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

Case-Based Reasoning (CBR)

- CBR: Uses a database of problem solutions to solve new problems
- Store symbolic description (tuples or cases)—not points in a Euclidean space
- Applications: Customer-service (product-related diagnosis), legal ruling
- Methodology
 - Instances represented by rich symbolic descriptions (e.g., function graphs)
 - Search for similar cases, multiple retrieved cases may be combined
 - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- Challenges
 - Find a good similarity metric
 - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

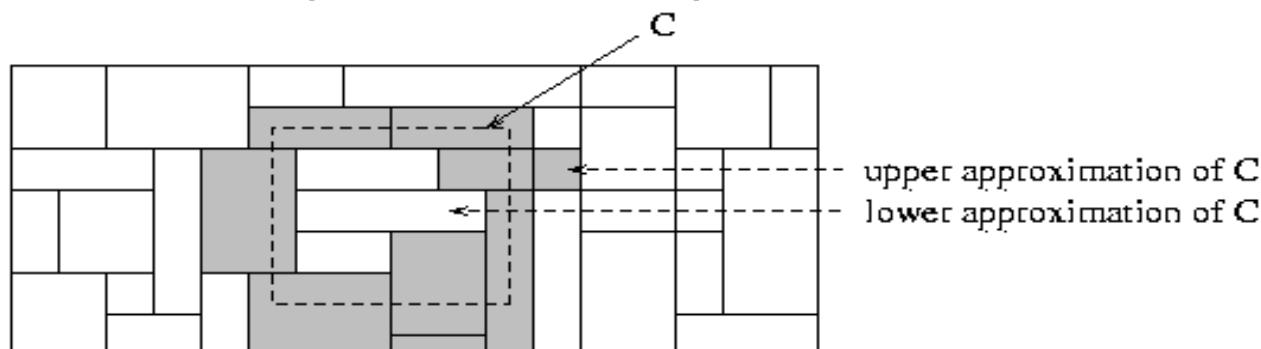


Genetic Algorithms (GA)

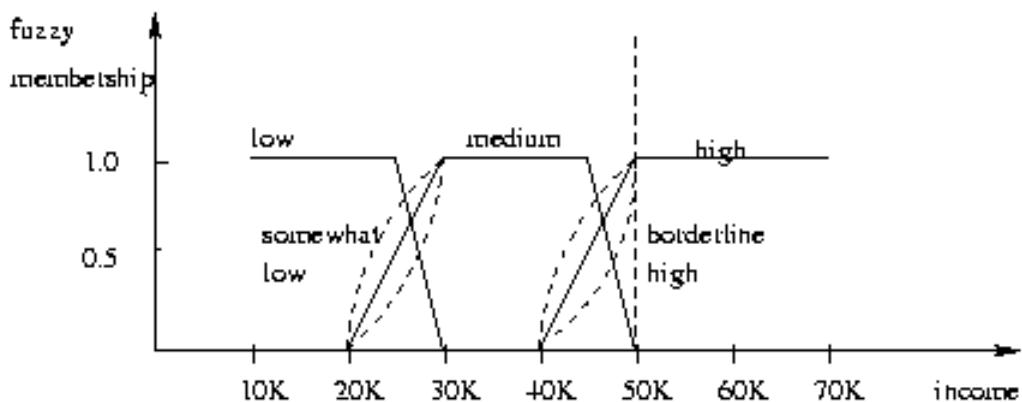
- Genetic Algorithm: based on an analogy to biological evolution
- An initial **population** is created consisting of randomly generated rules
 - Each rule is represented by a string of bits
 - E.g., if A_1 and $\neg A_2$ then C_2 can be encoded as 100
 - If an attribute has $k > 2$ values, k bits can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

Rough Set Approach

- Rough sets are used to **approximately or “roughly” define equivalent classes**
- A rough set for a given class C is approximated by two sets: a **lower approximation** (certain to be in C) and an **upper approximation** (cannot be described as not belonging to C)
- Finding the minimal subsets (**reducts**) of attributes for feature reduction is NP-hard but a **discernibility matrix** (which stores the differences between attribute values for each pair of data tuples) is used to reduce the computation intensity



Fuzzy Set Approaches



- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership (such as using **fuzzy membership graph**)
- Attribute values are converted to fuzzy values
 - e.g., income is mapped into the discrete categories {low, medium, high} with fuzzy values calculated
- For a given new sample, more than one fuzzy value may apply
- Each applicable rule contributes a vote for membership in the categories
- Typically, the truth values for each predicted category are summed, and these sums are combined

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction 
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

What Is Prediction?

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

Linear Regression

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

$$y = w_0 + w_1 x$$

where w_0 (y -intercept) and w_1 (slope) are regression coefficients

- Method of least squares: estimates the best-fitting straight line

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

- Multiple linear regression: involves more than one predictor variable
 - Training data is of the form $(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \dots, (\mathbf{X}_{|D|}, y_{|D|})$
 - Ex. For 2-D data, we may have: $y = w_0 + w_1 x_1 + w_2 x_2$
 - Solvable by extension of least square method or using SAS, S-Plus
 - Many nonlinear functions can be transformed into the above

Nonlinear Regression

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

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

convertible to linear with new variables: $x_2 = x^2$, $x_3 = x^3$

$$y = w_0 + w_1 x + w_2 x_2 + w_3 x_3$$

- Other functions, such as power function, can also be transformed to linear model
- Some models are intractable nonlinear (e.g., sum of exponential terms)
 - possible to obtain least square estimates through extensive calculation on more complex formulae

Other Regression-Based Models

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

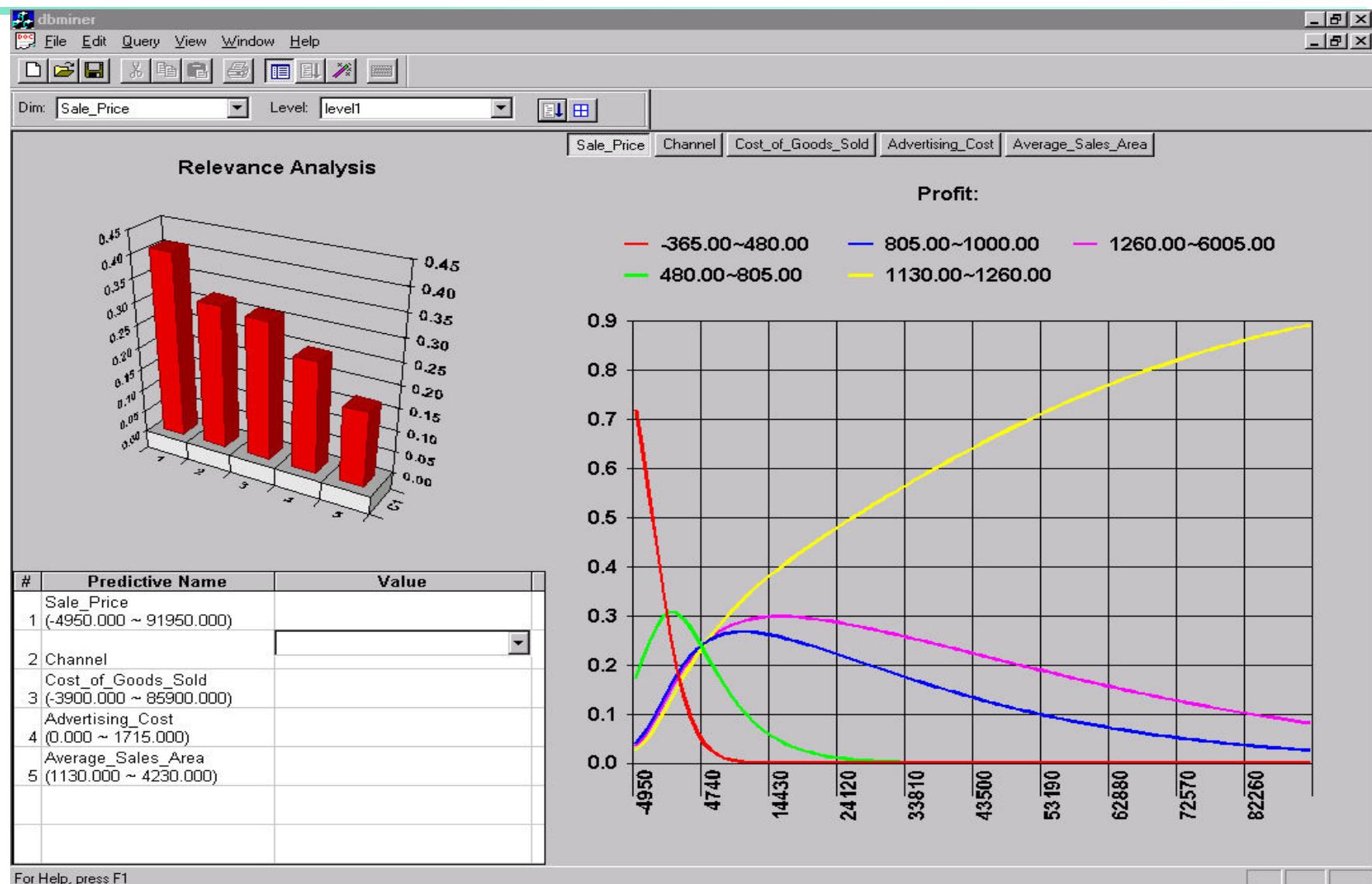
Regression Trees and Model Trees

- Regression tree: proposed in CART system (Breiman et al. 1984)
 - CART: Classification And Regression Trees
 - Each leaf stores a *continuous-valued prediction*
 - It is the *average value of the predicted attribute* for the training tuples that reach the leaf
- Model tree: proposed by Quinlan (1992)
 - Each leaf holds a regression model—a multivariate linear equation for the predicted attribute
 - A more general case than regression tree
- Regression and model trees tend to be more accurate than linear regression when the data are not represented well by a simple linear model

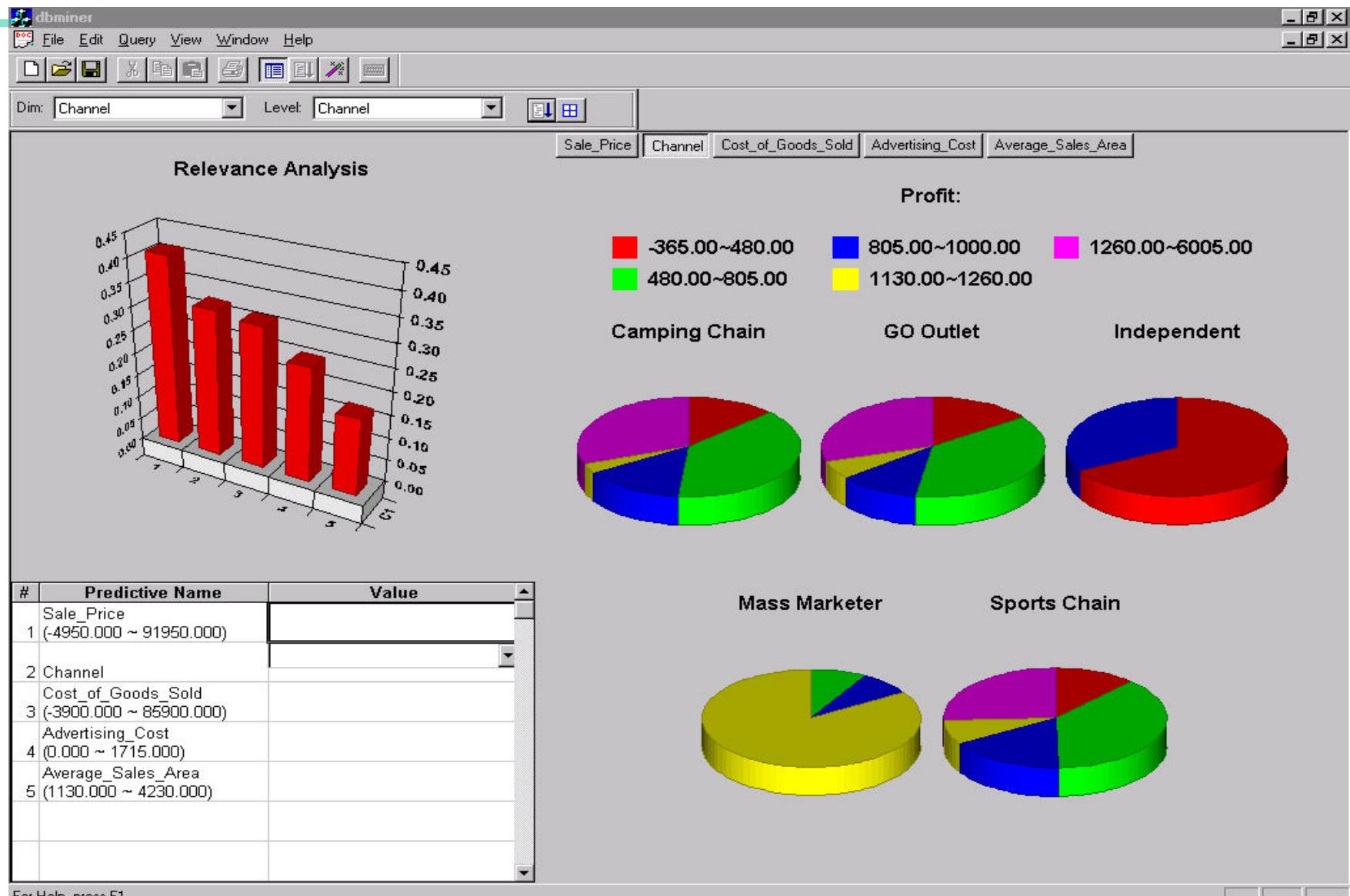
Predictive Modeling in Multidimensional Databases

- Predictive modeling: Predict data values or construct generalized linear models based on the database data
- One can only predict value ranges or category distributions
- Method outline:
 - Minimal generalization
 - Attribute relevance analysis
 - Generalized linear model construction
 - Prediction
- Determine the major factors which influence the prediction
 - Data relevance analysis: uncertainty measurement, entropy analysis, expert judgement, etc.
- Multi-level prediction: drill-down and roll-up analysis

Prediction: Numerical Data



Prediction: Categorical Data



Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary



	C_1	C_2
C_1	True positive	False negative
C_2	False positive	True negative

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

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

$\text{sensitivity} = t\text{-pos}/\text{pos}$	/* true positive recognition rate */
$\text{specificity} = t\text{-neg}/\text{neg}$	/* true negative recognition rate */
$\text{precision} = t\text{-pos}/(t\text{-pos} + f\text{-pos})$	

$\text{accuracy} = \text{sensitivity} * \text{pos}/(\text{pos} + \text{neg}) + \text{specificity} * \text{neg}/(\text{pos} + \text{neg})$

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

Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- **Loss function:** measures the error betw. y_i and the predicted value y'_i
 - Absolute error: $|y_i - y'_i|$
 - Squared error: $(y_i - y'_i)^2$
- Test error (generalization error): the average loss over the test set
 - Mean absolute error: $\frac{\sum_{i=1}^d |y_i - y'_i|}{d}$ Mean squared error: $\frac{\sum_{i=1}^d (y_i - y'_i)^2}{d}$
 - Relative absolute error: $\frac{\sum_{i=1}^d |y_i - y'_i|}{\sum_{i=1}^d |y_i - \bar{y}|}$ Relative squared error: $\frac{\sum_{i=1}^d (y_i - y'_i)^2}{\sum_{i=1}^d (y_i - \bar{y})^2}$

The mean squared-error exaggerates the presence of outliers

Popularly use (square) root mean-square error, similarly, root relative squared error

Evaluating the Accuracy of a Classifier or Predictor (I)

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

Evaluating the Accuracy of a Classifier or Predictor (II)

- Bootstrap
 - Works well with small data sets
 - Samples the given training tuples uniformly *with replacement*
 - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is **.632 bootstrap**
 - Suppose we are given a data set of d tuples. The data set is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data will end up in the bootstrap, and the remaining 36.8% will form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$)
 - Repeat the sampling procedure k times, overall accuracy of the model:

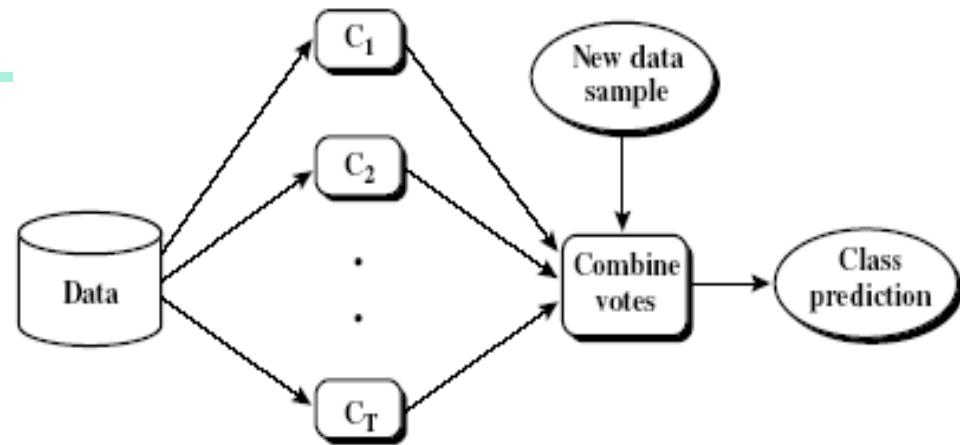
$$acc(M) = \sum_{i=1}^k (0.632 \times acc(M_i)_{test_set} + 0.368 \times acc(M_i)_{train_set})$$

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary



Ensemble Methods: Increasing the Accuracy



- Ensemble methods
 - Use a combination of models to increase accuracy
 - Combine a series of k learned models, M_1, M_2, \dots, M_k , with the aim of creating an improved model M^*
- Popular ensemble methods
 - Bagging: averaging the prediction over a collection of classifiers
 - Boosting: weighted vote with a collection of classifiers
 - Ensemble: combining a set of heterogeneous classifiers

Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
 - Given a set D of d tuples, at each iteration i , a training set D_i of d tuples is sampled with replacement from D (i.e., bootstrap)
 - A classifier model M_i is learned for each training set D_i
- Classification: classify an unknown sample \mathbf{X}
 - Each classifier M_i returns its class prediction
 - The bagged classifier M^* counts the votes and assigns the class with the most votes to \mathbf{X}
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
 - Often significant better than a single classifier derived from D
 - For noise data: not considerably worse, more robust
 - Proved improved accuracy in prediction

Boosting

- Analogy: Consult several doctors, based on a combination of weighted diagnoses—weight assigned based on the previous diagnosis accuracy
- How boosting works?
 - Weights are assigned to each training tuple
 - A series of k classifiers is iteratively learned
 - After a classifier M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1} , to pay more attention to the training tuples that were misclassified by M_i
 - The final M^* combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy
- The boosting algorithm can be extended for the prediction of continuous values
- Comparing with bagging: boosting tends to achieve greater accuracy, but it also risks overfitting the model to misclassified data

Adaboost (Freund and Schapire, 1997)

- Given a set of d class-labeled tuples, $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_d, y_d)$
- Initially, all the weights of tuples are set the same ($1/d$)
- Generate k classifiers in k rounds. At round i ,
 - Tuples from D are sampled (with replacement) to form a training set D_i of the same size
 - Each tuple's chance of being selected is based on its weight
 - A classification model M_i is derived from D_i
 - Its error rate is calculated using D_i as a test set
 - If a tuple is misclassified, its weight is increased, o.w. it is decreased
- Error rate: $\text{err}(\mathbf{X}_j)$ is the misclassification error of tuple \mathbf{X}_j . Classifier M_i error rate is the sum of the weights of the misclassified tuples:

$$\text{error}(M_i) = \sum_j^d w_j \times \text{err}(\mathbf{X}_j)$$

- The weight of classifier M_i 's vote is $\log \frac{1 - \text{error}(M_i)}{\text{error}(M_i)}$

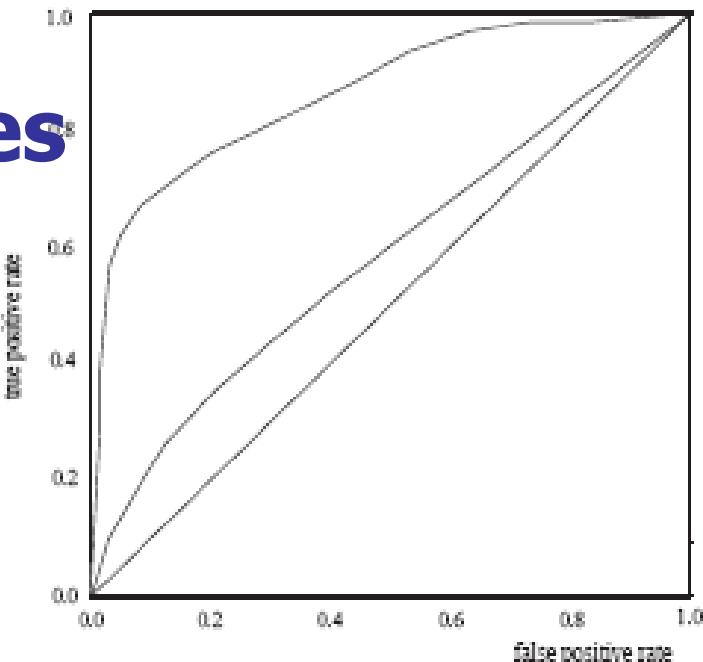
Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary



Model Selection: ROC Curves

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



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

Chapter 6. Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian classification
- Rule-based classification
- Classification by back propagation
- Support Vector Machines (SVM)
- Associative classification
- Lazy learners (or learning from your neighbors)
- Other classification methods
- Prediction
- Accuracy and error measures
- Ensemble methods
- Model selection
- Summary

Summary (I)

- Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends.
- Effective and scalable methods have been developed for decision trees induction, Naive Bayesian classification, Bayesian belief network, rule-based classifier, Backpropagation, Support Vector Machine (SVM), associative classification, nearest neighbor classifiers, and case-based reasoning, and other classification methods such as genetic algorithms, rough set and fuzzy set approaches.
- Linear, nonlinear, and generalized linear models of regression can be used for prediction. Many nonlinear problems can be converted to linear problems by performing transformations on the predictor variables. Regression trees and model trees are also used for prediction.

Summary (II)

- **Stratified k-fold cross-validation** is a recommended method for accuracy estimation. **Bagging** and **boosting** can be used to increase overall accuracy by learning and combining a series of individual models.
- **Significance tests** and **ROC curves** are useful for model selection
- There have been numerous **comparisons of the different classification and prediction methods**, and the matter remains a research topic
- No single method has been found to be superior over all others for all data sets
- Issues such as accuracy, training time, robustness, interpretability, and scalability must be considered and can involve trade-offs, further complicating the quest for an overall superior method

References (1)

- C. Apte and S. Weiss. **Data mining with decision trees and decision rules**. Future Generation Computer Systems, 13, 1997.
- C. M. Bishop, **Neural Networks for Pattern Recognition**. Oxford University Press, 1995.
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. **Classification and Regression Trees**. Wadsworth International Group, 1984.
- C. J. C. Burges. **A Tutorial on Support Vector Machines for Pattern Recognition**. *Data Mining and Knowledge Discovery*, 2(2): 121-168, 1998.
- P. K. Chan and S. J. Stolfo. **Learning arbiter and combiner trees from partitioned data for scaling machine learning**. KDD'95.
- W. Cohen. **Fast effective rule induction**. ICML'95.
- G. Cong, K.-L. Tan, A. K. H. Tung, and X. Xu. **Mining top-k covering rule groups for gene expression data**. SIGMOD'05.
- A. J. Dobson. **An Introduction to Generalized Linear Models**. Chapman and Hall, 1990.
- G. Dong and J. Li. **Efficient mining of emerging patterns: Discovering trends and differences**. KDD'99.

References (2)

- R. O. Duda, P. E. Hart, and D. G. Stork. **Pattern Classification**, 2ed. John Wiley and Sons, 2001
- U. M. Fayyad. **Branching on attribute values in decision tree generation**. AAAI'94.
- Y. Freund and R. E. Schapire. **A decision-theoretic generalization of on-line learning and an application to boosting**. J. Computer and System Sciences, 1997.
- J. Gehrke, R. Ramakrishnan, and V. Ganti. **Rainforest: A framework for fast decision tree construction of large datasets**. VLDB'98.
- J. Gehrke, V. Gant, R. Ramakrishnan, and W.-Y. Loh, **BOAT -- Optimistic Decision Tree Construction**. SIGMOD'99.
- T. Hastie, R. Tibshirani, and J. Friedman. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. Springer-Verlag, 2001.
- D. Heckerman, D. Geiger, and D. M. Chickering. **Learning Bayesian networks: The combination of knowledge and statistical data**. Machine Learning, 1995.
- M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. **Generalization and decision tree induction: Efficient classification in data mining**. RIDE'97.
- B. Liu, W. Hsu, and Y. Ma. **Integrating Classification and Association Rule**. KDD'98.
- W. Li, J. Han, and J. Pei, **CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules**, ICDM'01.

References (3)

- T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. **A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms.** Machine Learning, 2000.
- J. Magidson. **The Chaid approach to segmentation modeling: Chi-squared automatic interaction detection.** In R. P. Bagozzi, editor, Advanced Methods of Marketing Research, Blackwell Business, 1994.
- M. Mehta, R. Agrawal, and J. Rissanen. **SLIQ : A fast scalable classifier for data mining.** EDBT'96.
- T. M. Mitchell. **Machine Learning.** McGraw Hill, 1997.
- S. K. Murthy, **Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey**, Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan. **Induction of decision trees.** *Machine Learning*, 1:81-106, 1986.
- J. R. Quinlan and R. M. Cameron-Jones. **FOIL: A midterm report.** ECML'93.
- J. R. Quinlan. **C4.5: Programs for Machine Learning.** Morgan Kaufmann, 1993.
- J. R. Quinlan. **Bagging, boosting, and c4.5.** AAAI'96.

References (4)

- R. Rastogi and K. Shim. **Public: A decision tree classifier that integrates building and pruning.** VLDB'98.
- J. Shafer, R. Agrawal, and M. Mehta. **SPRINT : A scalable parallel classifier for data mining.** VLDB'96.
- J. W. Shavlik and T. G. Dietterich. **Readings in Machine Learning.** Morgan Kaufmann, 1990.
- P. Tan, M. Steinbach, and V. Kumar. **Introduction to Data Mining.** Addison Wesley, 2005.
- S. M. Weiss and C. A. Kulikowski. **Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems.** Morgan Kaufman, 1991.
- S. M. Weiss and N. Indurkhya. **Predictive Data Mining.** Morgan Kaufmann, 1997.
- I. H. Witten and E. Frank. **Data Mining: Practical Machine Learning Tools and Techniques**, 2ed. Morgan Kaufmann, 2005.
- X. Yin and J. Han. **CPAR: Classification based on predictive association rules.** SDM'03
- H. Yu, J. Yang, and J. Han. **Classifying large data sets using SVM with hierarchical clusters.** KDD'03.



Data Mining: Concepts and Techniques

— Chapter 7 —

Jiawei Han

Department of Computer Science
University of Illinois at Urbana-Champaign

www.cs.uiuc.edu/~hanj

©2006 Jiawei Han and Micheline Kamber, All rights reserved



Chapter 7. Cluster Analysis

1. What is Cluster Analysis? 
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary

What is Cluster Analysis?

- Cluster: a collection of data objects
 - Similar to one another within the same cluster
 - Dissimilar to the objects in other clusters
- Cluster analysis
 - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- **Unsupervised learning**: no predefined classes
- Typical applications
 - As a **stand-alone tool** to get insight into data distribution
 - As a **preprocessing step** for other algorithms

Clustering: Rich Applications and Multidisciplinary Efforts

- Pattern Recognition
- Spatial Data Analysis
 - Create thematic maps in GIS by clustering feature spaces
 - Detect spatial clusters or for other spatial mining tasks
- Image Processing
- Economic Science (especially market research)
- WWW
 - Document classification
 - Cluster Weblog data to discover groups of similar access patterns

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earthquake studies: Observed earth quake epicenters should be clustered along continent faults

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters with
 - high intra-class similarity
 - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

Measure the Quality of Clustering

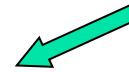
- **Dissimilarity/Similarity metric:** Similarity is expressed in terms of a distance function, typically metric: $d(i, j)$
- There is a separate “quality” function that measures the “goodness” of a cluster.
- The definitions of **distance functions** are usually very different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables.
- Weights should be associated with different variables based on applications and data semantics.
- It is hard to define “similar enough” or “good enough”
 - the answer is typically highly subjective.

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Ability to handle dynamic data
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



Data Structures

- Data matrix
 - (two modes)

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

- Dissimilarity matrix
 - (one mode)

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

Type of data in clustering analysis

- Interval-scaled variables
- Binary variables
- Nominal, ordinal, and ratio variables
- Variables of mixed types

Interval-valued variables

- Standardize data

- Calculate the mean absolute deviation:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \dots + |x_{nf} - m_f|)$$

where $m_f = \frac{1}{n}(x_{1f} + x_{2f} + \dots + x_{nf})$.

- Calculate the standardized measurement (*z-score*)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is more robust than using standard deviation

Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- Some popular ones include: *Minkowski distance*:

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

where $i = (x_{i1}, x_{i2}, \dots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \dots, x_{jp})$ are two p -dimensional data objects, and q is a positive integer

- If $q = 1$, d is Manhattan distance

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

Similarity and Dissimilarity Between Objects (Cont.)

- If $q = 2$, d is Euclidean distance:

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

- Properties

- $d(i, j) \geq 0$
- $d(i, i) = 0$
- $d(i, j) = d(j, i)$
- $d(i, j) \leq d(i, k) + d(k, j)$

- Also, one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures

Binary Variables

- A contingency table for binary data

		Object <i>j</i>		
		1	0	<i>sum</i>
Object <i>i</i>	1	<i>a</i>	<i>b</i>	<i>a+b</i>
	0	<i>c</i>	<i>d</i>	<i>c+d</i>
	<i>sum</i>	<i>a+c</i>	<i>b+d</i>	<i>p</i>

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

$$d(i, j) = \frac{b + c}{a + b + c + d}$$

$$d(i, j) = \frac{b + c}{a + b + c}$$

$$\text{sim}_{\text{Jaccard}}(i, j) = \frac{a}{a + b + c}$$

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 set to 1, and the value N be set to 0

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

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

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

Nominal Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- 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 variables
 - creating a new binary variable for each of the M nominal states

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 $r_{if} \in \{1, \dots, M_f\}$
 - 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

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale, such as Ae^{Bt} or Ae^{-Bt}
- Methods:
 - treat them like interval-scaled variables—*not a good choice!* (why?—the scale can be distorted)
 - apply logarithmic transformation
$$y_{if} = \log(x_{if})$$
 - treat them as continuous ordinal data treat their rank as interval-scaled

Variables of Mixed Types

- A database may contain all the six types of variables
 - symmetric binary, asymmetric binary, nominal, ordinal, interval and ratio
- One may use a weighted formula to combine their effects

$$d(i, j) = \frac{\sum_{f=1}^P \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^P \delta_{ij}^{(f)}}$$

- f is binary or nominal:
 $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, or $d_{ij}^{(f)} = 1$ otherwise
- f is interval-based: use the normalized distance
- f is ordinal or ratio-scaled
 - compute ranks r_{if} and
 - and treat z_{if} as interval-scaled

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

Vector Objects

- Vector objects: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine measure

$$s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{|\vec{X}| |\vec{Y}|},$$

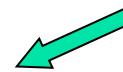
\vec{X}^t is a transposition of vector \vec{X} , $|\vec{X}|$ is the Euclidean norm of vector \vec{X} ,

- A variant: Tanimoto coefficient

$$s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{\vec{X}^t \cdot \vec{X} + \vec{Y}^t \cdot \vec{Y} - \vec{X}^t \cdot \vec{Y}},$$

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



Major Clustering Approaches (I)

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

Major Clustering Approaches (II)

- Grid-based approach:
 - based on a multiple-level granularity structure
 - Typical methods: STING, WaveCluster, CLIQUE
- Model-based:
 - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
 - Typical methods: EM, SOM, COBWEB
- Frequent pattern-based:
 - Based on the analysis of frequent patterns
 - Typical methods: pCluster
- User-guided or constraint-based:
 - Clustering by considering user-specified or application-specific constraints
 - Typical methods: COD (obstacles), constrained clustering

Typical Alternatives to Calculate the Distance between Clusters

- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $\text{dis}(K_i, K_j) = \min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $\text{dis}(K_i, K_j) = \max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $\text{dis}(K_i, K_j) = \text{avg}(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e., $\text{dis}(K_i, K_j) = \text{dis}(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e., $\text{dis}(K_i, K_j) = \text{dis}(M_i, M_j)$
 - Medoid: one chosen, centrally located object in the cluster

Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the “middle” of a cluster

$$C_m = \frac{\sum_{i=1}^N (t_{ip})}{N}$$

- Radius: square root of average distance from any point of the cluster to its centroid

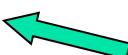
$$R_m = \sqrt{\frac{\sum_{i=1}^N (t_{ip} - c_m)^2}{N}}$$

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

$$D_m = \sqrt{\frac{\sum_{i=1}^N \sum_{j=1}^N (t_{ip} - t_{iq})^2}{N(N-1)}}$$

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database \mathcal{D} of n objects into a set of k clusters, s.t., min sum of squared distance

$$\sum_{m=1}^k \sum_{t_{mi} \in K_m} (C_m - t_{mi})^2$$

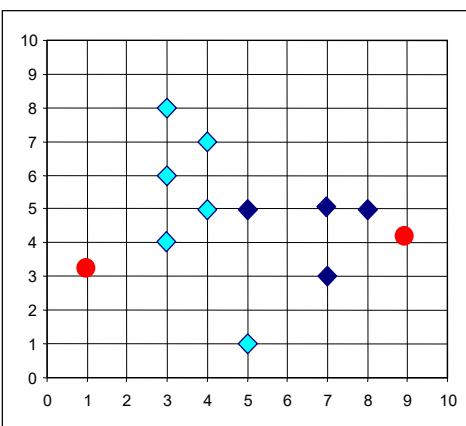
- Given a k , find a partition of k *clusters* that optimizes the chosen partitioning criterion
 - Global optimal: exhaustively enumerate all partitions
 - Heuristic methods: *k-means* and *k-medoids* algorithms
 - *k-means* (MacQueen'67): Each cluster is represented by the center of the cluster
 - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

The *K-Means* Clustering Method

- Given k , the *k-means* algorithm is implemented in four steps:
 - Partition objects into k nonempty subsets
 - Compute seed points as the centroids of the clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
 - Assign each object to the cluster with the nearest seed point
 - Go back to Step 2, stop when no more new assignment

The *K*-Means Clustering Method

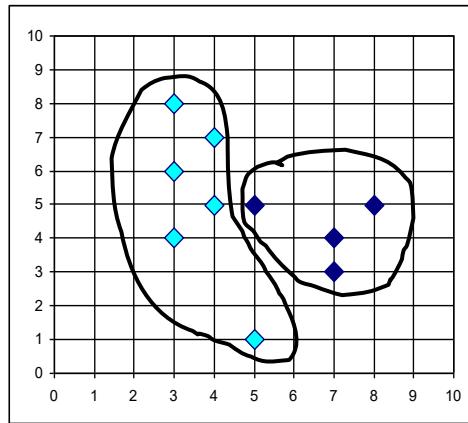
■ Example



K=2

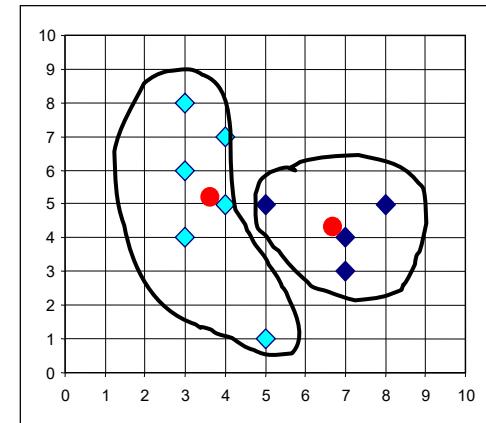
Arbitrarily choose K object as initial cluster center

Assign each objects to most similar center



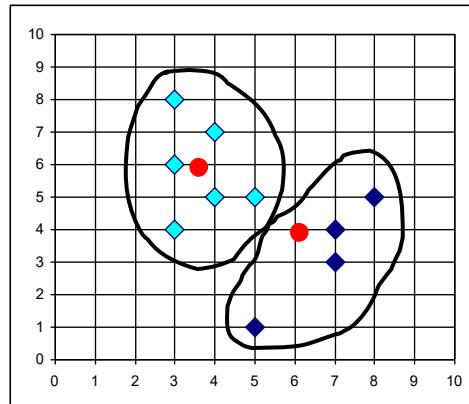
reassign

Update the cluster means



reassign

Update the cluster means



Comments on the *K-Means* Method

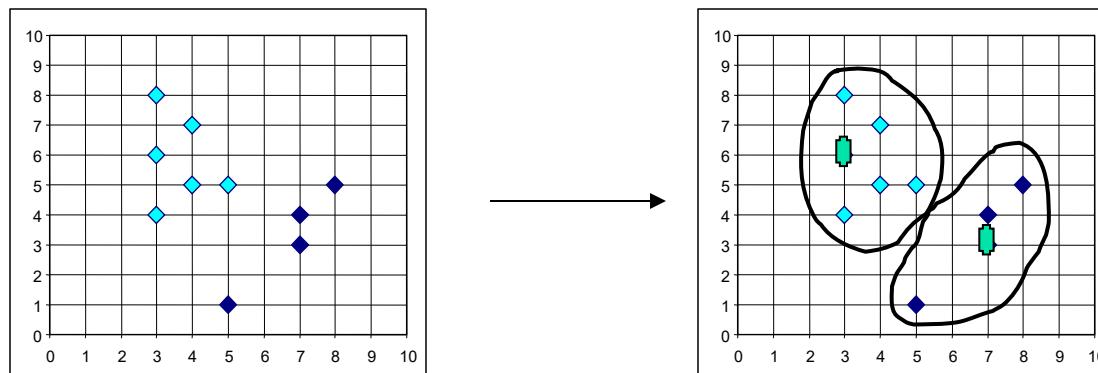
- Strength: *Relatively efficient*: $\mathcal{O}(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
 - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Comment: Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness
 - Applicable only when *mean* is defined, then what about categorical data?
 - Need to specify k , the *number* of clusters, in advance
 - Unable to handle noisy data and *outliers*
 - Not suitable to discover clusters with *non-convex shapes*

Variations of the *K-Means* Method

- A few variants of the *k-means* which differ in
 - Selection of the initial k means
 - Dissimilarity calculations
 - Strategies to calculate cluster means
- Handling categorical data: *k-modes* (Huang'98)
 - Replacing means of clusters with modes
 - Using new dissimilarity measures to deal with categorical objects
 - Using a frequency-based method to update modes of clusters
 - A mixture of categorical and numerical data: *k-prototype* method

What Is the Problem of the K-Means Method?

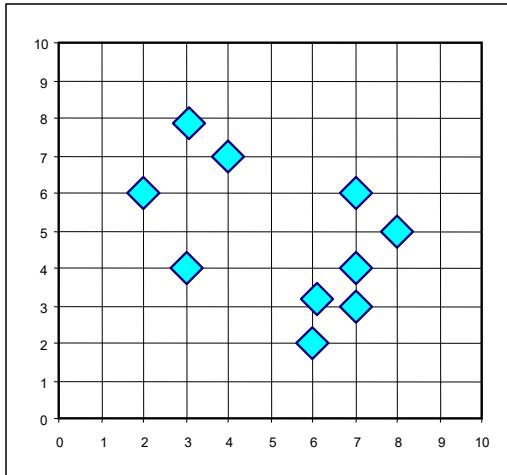
- The k-means algorithm is sensitive to outliers !
 - Since an object with an extremely large value may substantially distort the distribution of the data.
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.



The *K-Medoids* Clustering Method

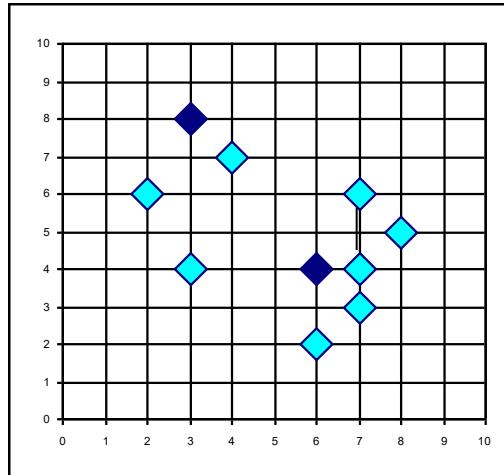
- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
 - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
 - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)

A Typical K-Medoids Algorithm (PAM)

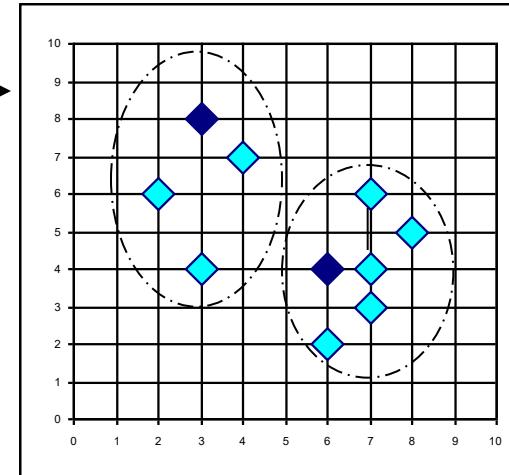


K=2

Arbitrary
choose k
object as
initial
medoids



Assign
each
remainin
g object to
nearest
medoids

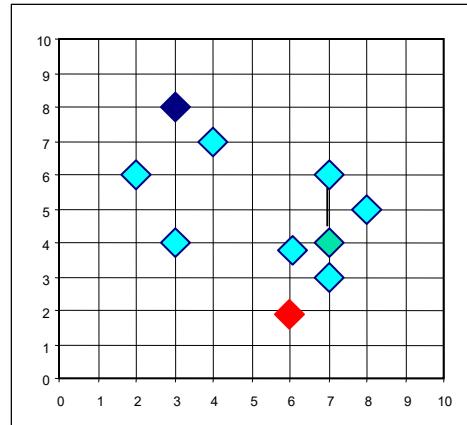


Total Cost = 20

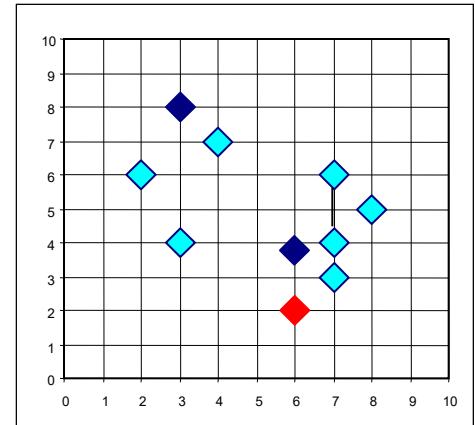
Randomly select a
nonmedoid object, O_{random}

Do loop
Until no
change

Swapping O
and O_{random}
If quality is
improved.



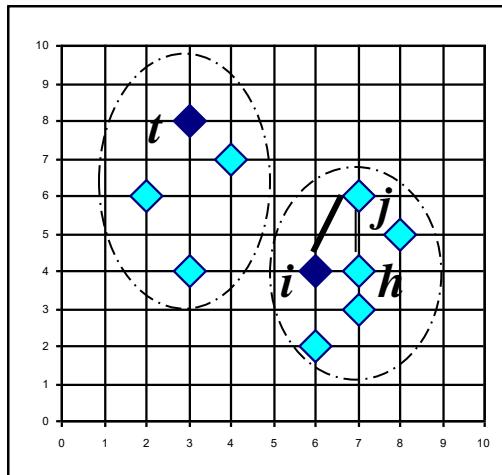
Compute
total cost of
swapping



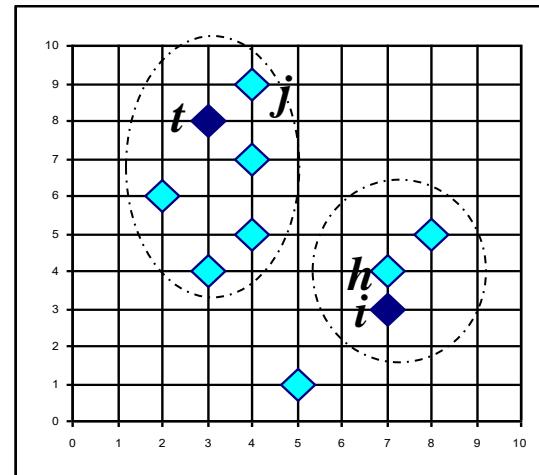
PAM (Partitioning Around Medoids) (1987)

- PAM (Kaufman and Rousseeuw, 1987), built in Splus
- Use real object to represent the cluster
 - Select k representative objects arbitrarily
 - For each pair of non-selected object h and selected object i , calculate the total swapping cost TC_{ih}
 - For each pair of i and h ,
 - If $TC_{ih} < 0$, i is replaced by h
 - Then assign each non-selected object to the most similar representative object
 - repeat steps 2-3 until there is no change

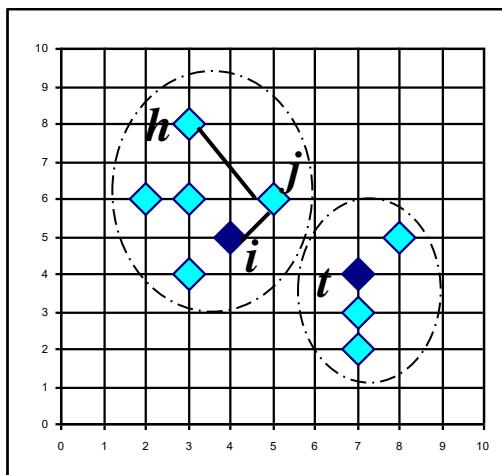
PAM Clustering: Total swapping cost $TC_{jh} = \sum_j C_{jih}$



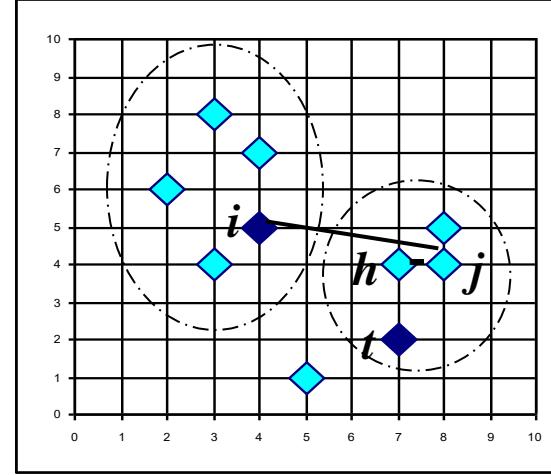
$$C_{jih} = d(j, h) - d(j, i)$$



$$C_{jih} = 0$$



$$C_{jih} = d(j, t) - d(j, i)$$



$$C_{jih} = d(j, h) - d(j, t)$$

What Is the Problem with PAM?

- Pam is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- Pam works efficiently for small data sets but does not **scale well** for large data sets.
 - $O(k(n-k)^2)$ for each iteration

where n is # of data,k is # of clusters

→ Sampling based method,

CLARA(Clustering LARge Applications)

CLARA (Clustering Large Applications) (1990)

- *CLARA* (Kaufmann and Rousseeuw in 1990)
 - Built in statistical analysis packages, such as S+
- It draws *multiple samples* of the data set, applies *PAM* on each sample, and gives the best clustering as the output
- Strength: deals with larger data sets than *PAM*
- Weakness:
 - Efficiency depends on the sample size
 - A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased

CLARANS ("Randomized" CLARA) (1994)

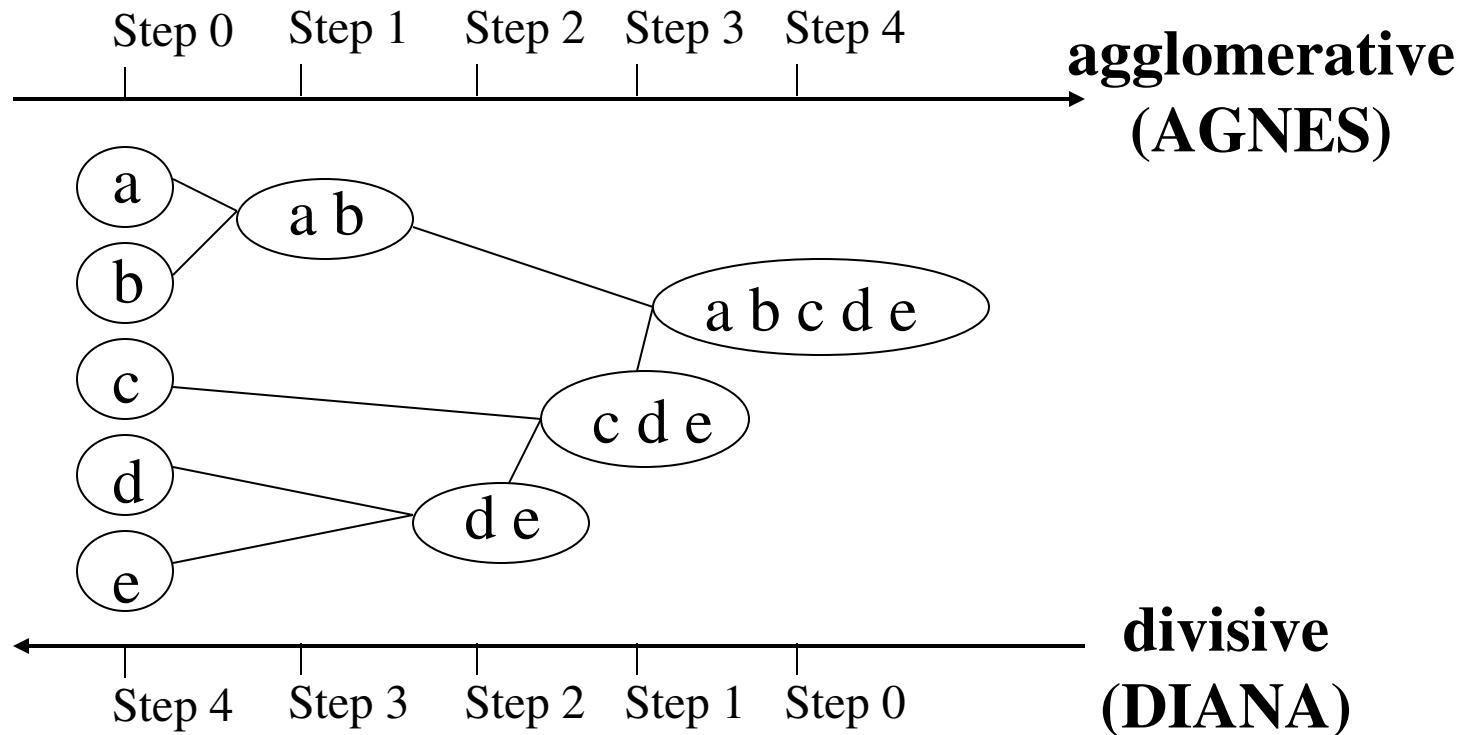
- *CLARANS* (A Clustering Algorithm based on Randomized Search) (Ng and Han'94)
- CLARANS draws sample of neighbors dynamically
- The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of k medoids
- If the local optimum is found, *CLARANS* starts with new randomly selected node in search for a new local optimum
- It is more efficient and scalable than both *PAM* and *CLARA*
- Focusing techniques and spatial access structures may further improve its performance (Ester et al.'95)

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary

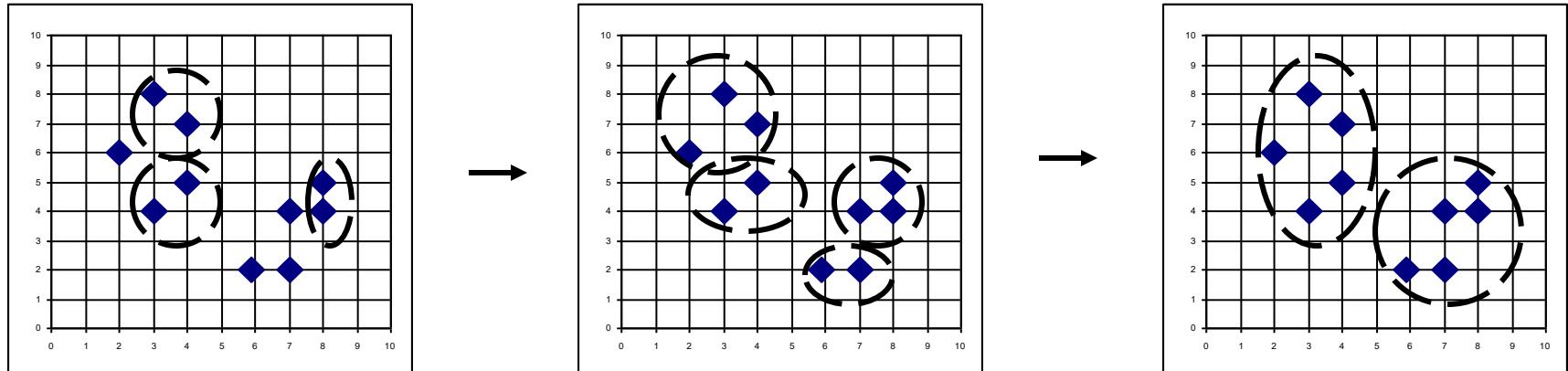
Hierarchical Clustering

- Use distance matrix as clustering criteria. This method does not require the number of clusters k as an input, but needs a termination condition



AGNES (Agglomerative Nesting)

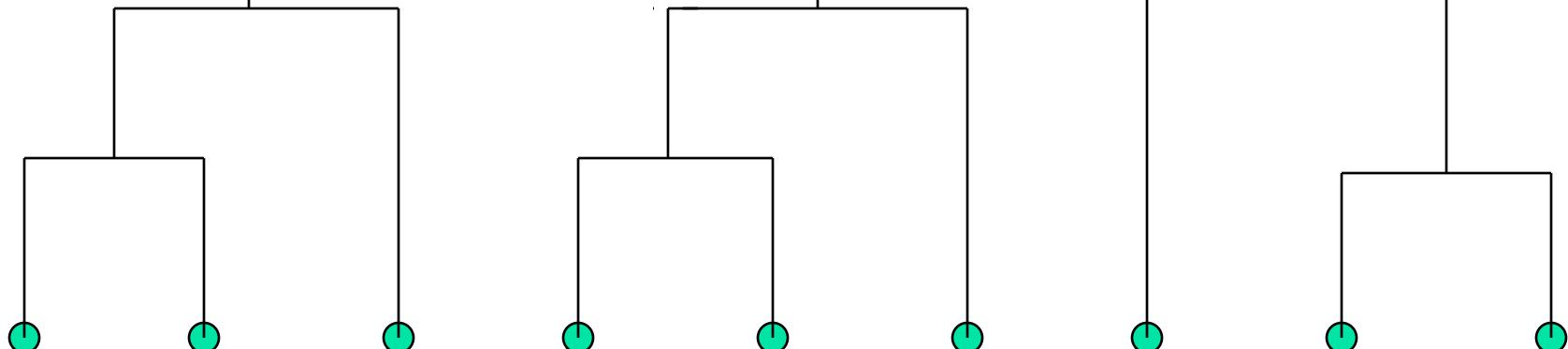
- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster



Dendrogram: Shows How the Clusters are Merged

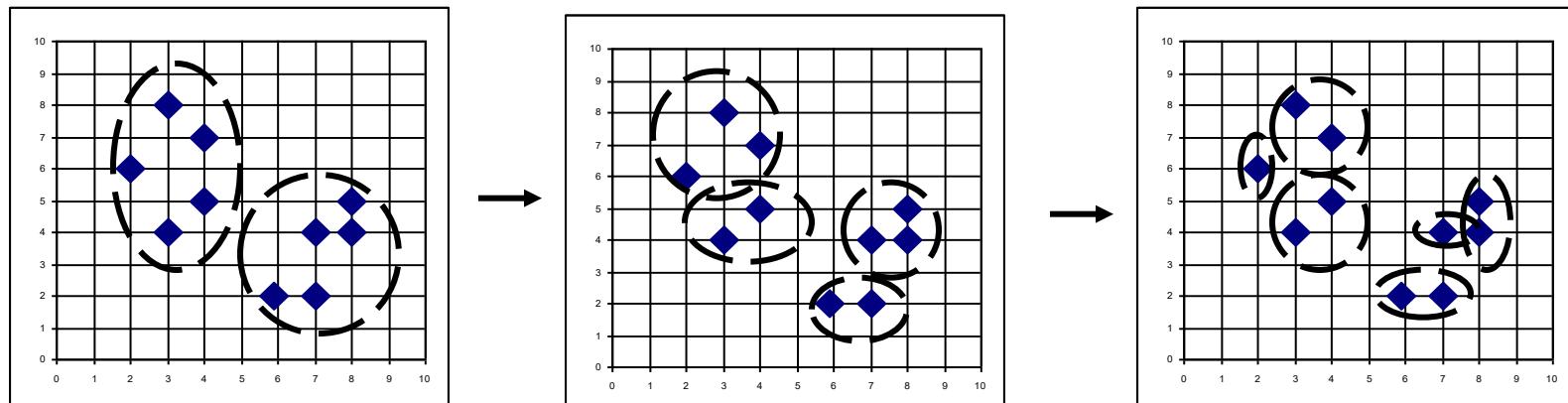
Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram.

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



DIANA (Divisive Analysis)

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



Recent Hierarchical Clustering Methods

- Major weakness of agglomerative clustering methods
 - do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
 - can never undo what was done previously
- Integration of hierarchical with distance-based clustering
 - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
 - ROCK (1999): clustering categorical data by neighbor and link analysis
 - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH (1996)

- Birch: Balanced Iterative Reducing and Clustering using Hierarchies (Zhang, Ramakrishnan & Livny, SIGMOD'96)
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
 - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
 - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness*: handles only numeric data, and sensitive to the order of the data record.

Clustering Feature Vector in BIRCH

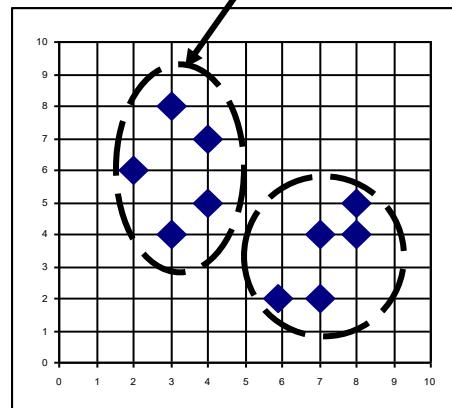
Clustering Feature: $CF = (N, \overrightarrow{LS}, SS)$

N: Number of data points

LS: $\sum_{i=1}^N X_i$

SS: $\sum_{i=1}^N X_i^2$

$$CF = (5, (16,30),(54,190))$$



(3,4)

(2,6)

(4,5)

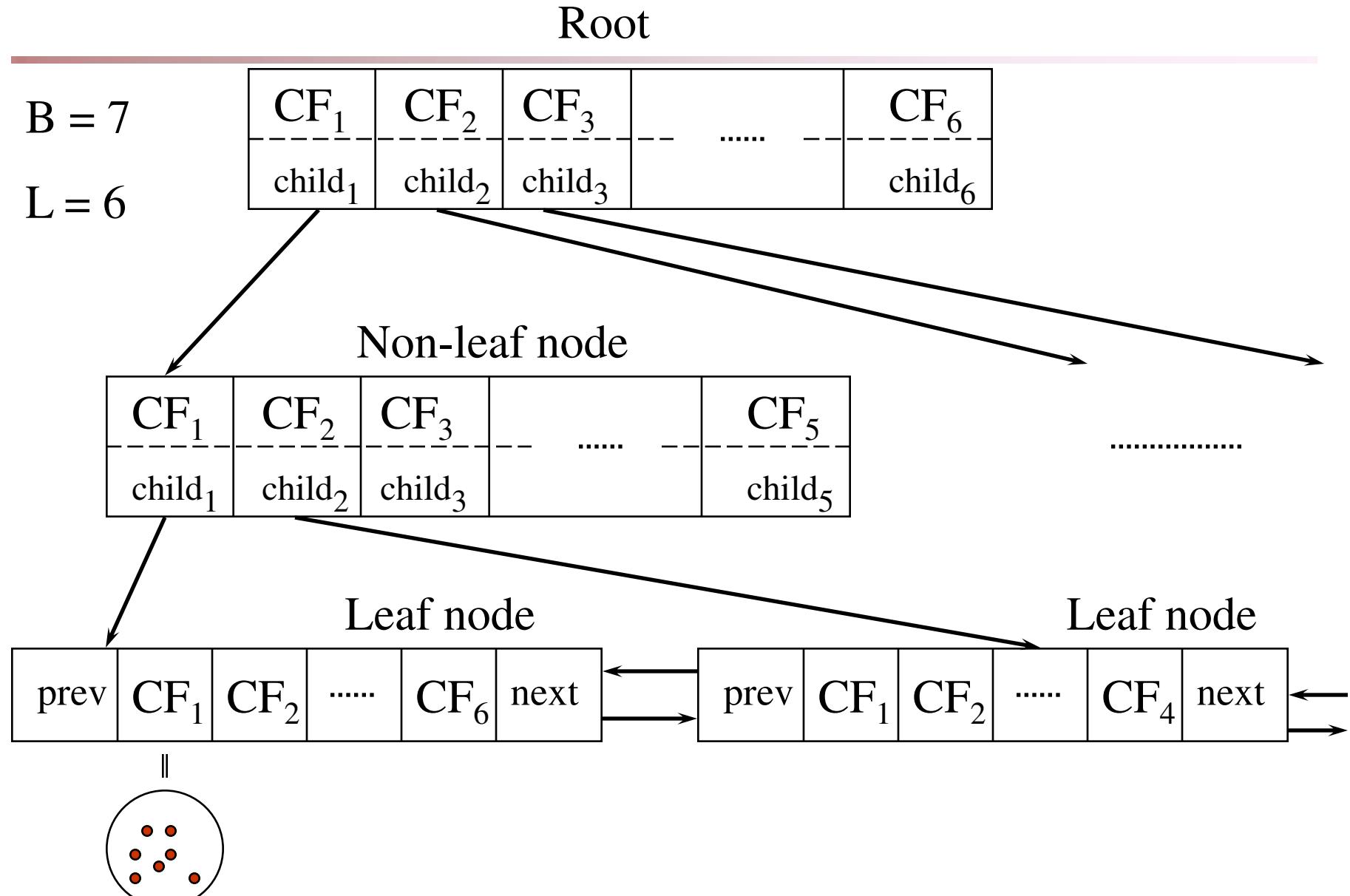
(4,7)

(3,8)

CF-Tree in BIRCH

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

The CF Tree Structure



Clustering Categorical Data: The ROCK Algorithm

- ROCK: RObust Clustering using linKs
 - S. Guha, R. Rastogi & K. Shim, ICDE'99
- Major ideas
 - Use links to measure similarity/proximity
 - Not distance-based
 - Computational complexity: $O(n^2 + nm_m m_a + n^2 \log n)$
- Algorithm: sampling-based clustering
 - Draw random sample
 - Cluster with links
 - Label data in disk
- Experiments
 - Congressional voting, mushroom data

Similarity Measure in ROCK

- Traditional measures for categorical data may not work well, e.g., Jaccard coefficient
- Example: Two groups (clusters) of transactions
 - C₁. <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
 - C₂. <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Jaccard co-efficient may lead to wrong clustering result
 - C₁: 0.2 ({a, b, c}, {b, d, e}) to 0.5 ({a, b, c}, {a, b, d})
 - C₁ & C₂: could be as high as 0.5 ({a, b, c}, {a, b, f})
- Jaccard co-efficient-based similarity function:
 - Ex. Let T₁ = {a, b, c}, T₂ = {c, d, e}

$$Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}$$

$$Sim(T_1, T_2) = \frac{|\{c\}|}{|\{a, b, c, d, e\}|} = \frac{1}{5} = 0.2$$

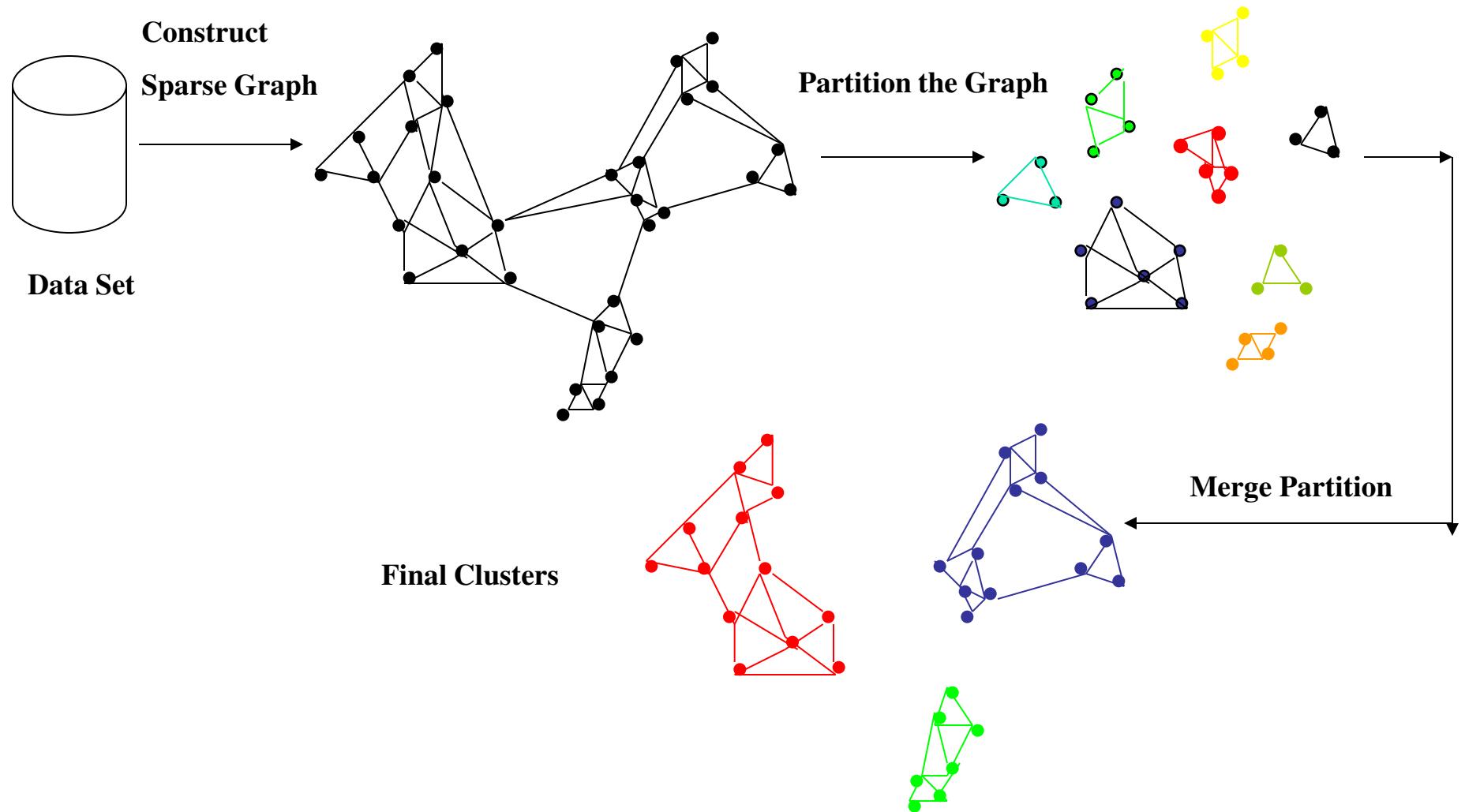
Link Measure in ROCK

- Links: # of common neighbors
 - $C_1 <a, b, c, d, e>$: $\{\underline{a}, \underline{b}, \underline{c}\}$, $\{\underline{a}, \underline{b}, \underline{d}\}$, $\{\underline{a}, \underline{b}, \underline{e}\}$, $\{\underline{a}, \underline{c}, \underline{d}\}$, $\{\underline{a}, \underline{c}, \underline{e}\}$, $\{\underline{a}, \underline{d}, \underline{e}\}$, $\{\underline{b}, \underline{c}, \underline{d}\}$, $\{\underline{b}, \underline{c}, \underline{e}\}$, $\{\underline{b}, \underline{d}, \underline{e}\}$, $\{\underline{c}, \underline{d}, \underline{e}\}$
 - $C_2 <a, b, f, g>$: $\{\underline{a}, \underline{b}, \underline{f}\}$, $\{\underline{a}, \underline{b}, \underline{g}\}$, $\{\underline{a}, \underline{f}, \underline{g}\}$, $\{\underline{b}, \underline{f}, \underline{g}\}$
- Let $T_1 = \{a, b, c\}$, $T_2 = \{c, d, e\}$, $T_3 = \{a, b, f\}$
 - $\text{link}(T_1, T_2) = 4$, since they have 4 common neighbors
 - $\{a, c, d\}$, $\{a, c, e\}$, $\{b, c, d\}$, $\{b, c, e\}$
 - $\text{link}(T_1, T_3) = 3$, since they have 3 common neighbors
 - $\{a, b, d\}$, $\{a, b, e\}$, $\{a, b, g\}$
- Thus link is a better measure than Jaccard coefficient

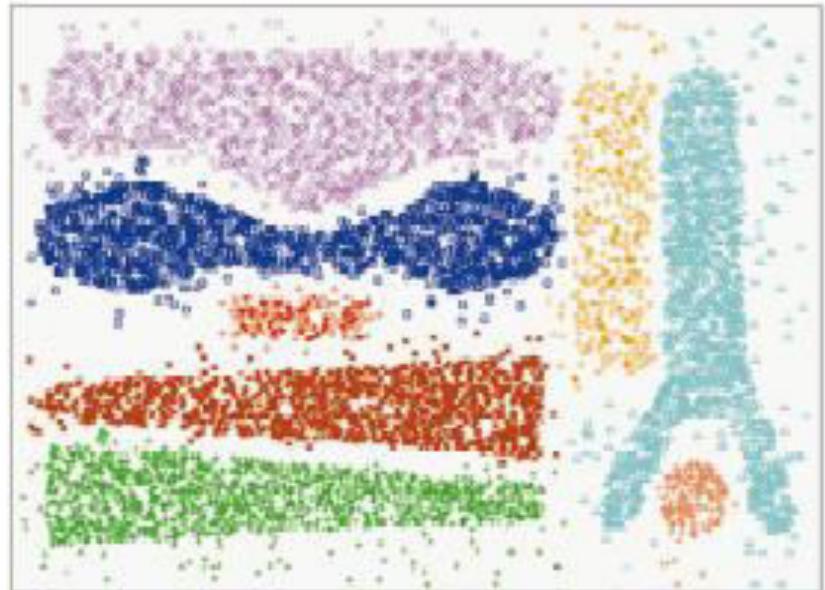
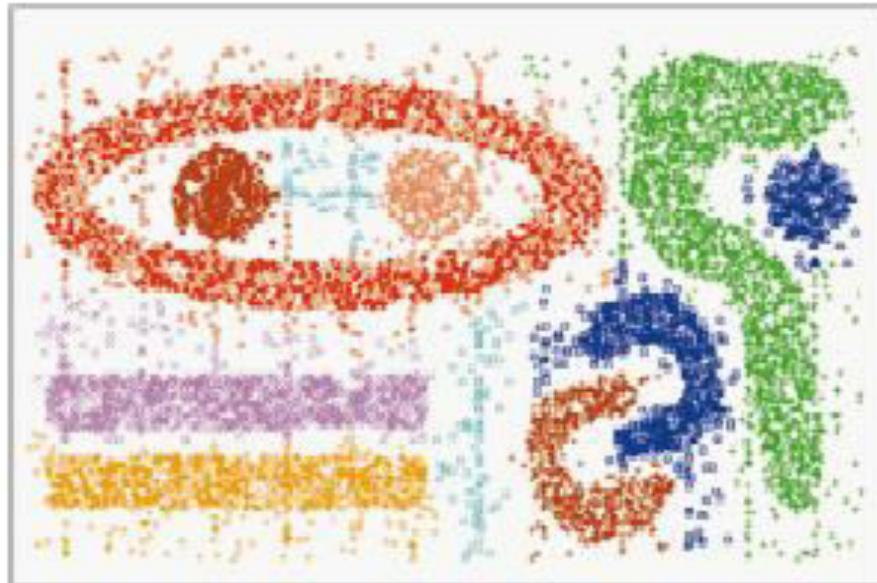
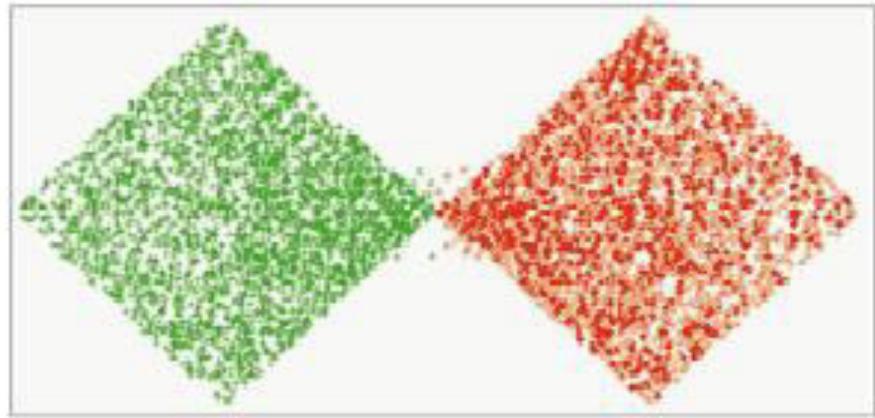
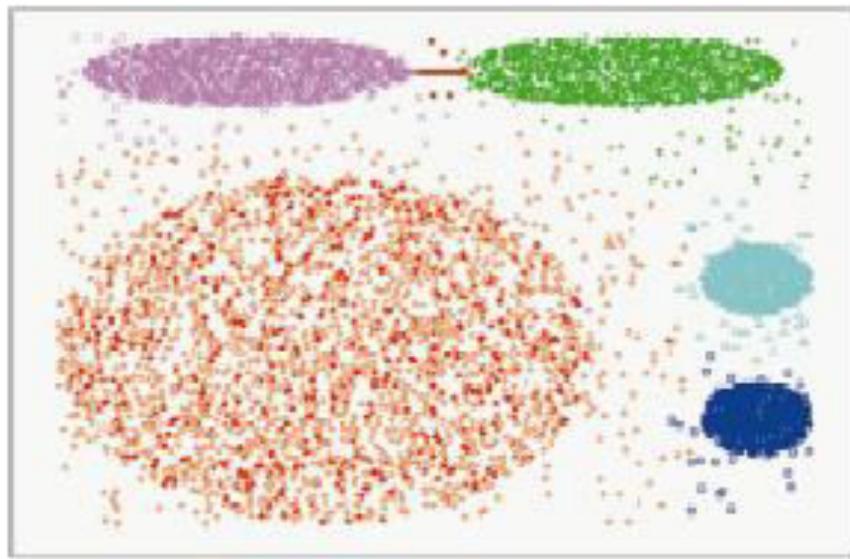
CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- CHAMELEON: by G. Karypis, E.H. Han, and V. Kumar'99
- Measures the similarity based on a dynamic model
 - Two clusters are merged only if the *interconnectivity* and *closeness* (*proximity*) between two clusters are high *relative to* the internal interconnectivity of the clusters and closeness of items within the clusters
 - **Cure** ignores information about **interconnectivity** of the objects, **Rock** ignores information about the **closeness** of two clusters
- A two-phase algorithm
 1. Use a graph partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
 2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

Overall Framework of CHAMELEON



CHAMELEON (Clustering Complex Objects)



Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods 
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary

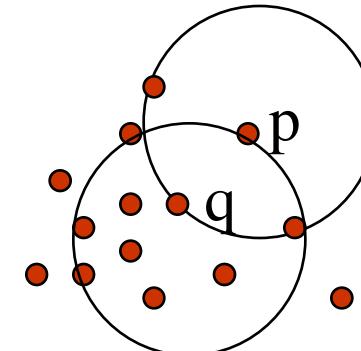
Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
 - Discover clusters of arbitrary shape
 - Handle noise
 - One scan
 - Need density parameters as termination condition
- Several interesting studies:
 - DBSCAN: Ester, et al. (KDD'96)
 - OPTICS: Ankerst, et al (SIGMOD'99).
 - DENCLUE: Hinneburg & D. Keim (KDD'98)
 - CLIQUE: Agrawal, et al. (SIGMOD'98) (more grid-based)

Density-Based Clustering: Basic Concepts

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

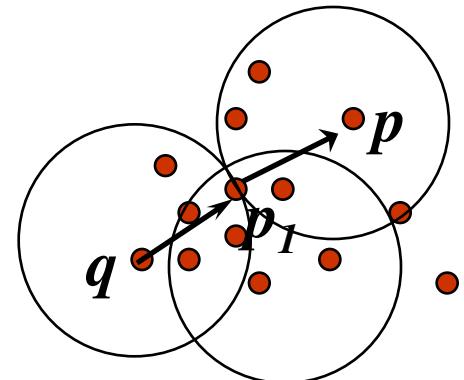
$$|N_{Eps}(q)| \geq MinPts$$



Density-Reachable and Density-Connected

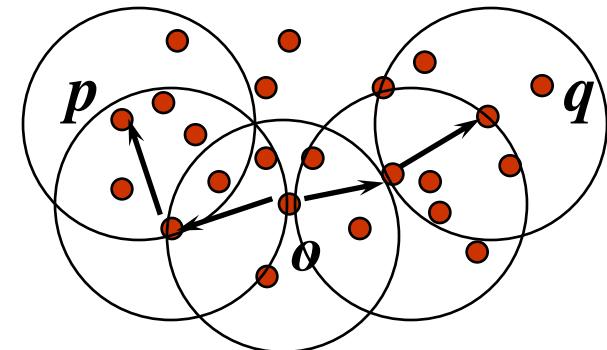
- Density-reachable:

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



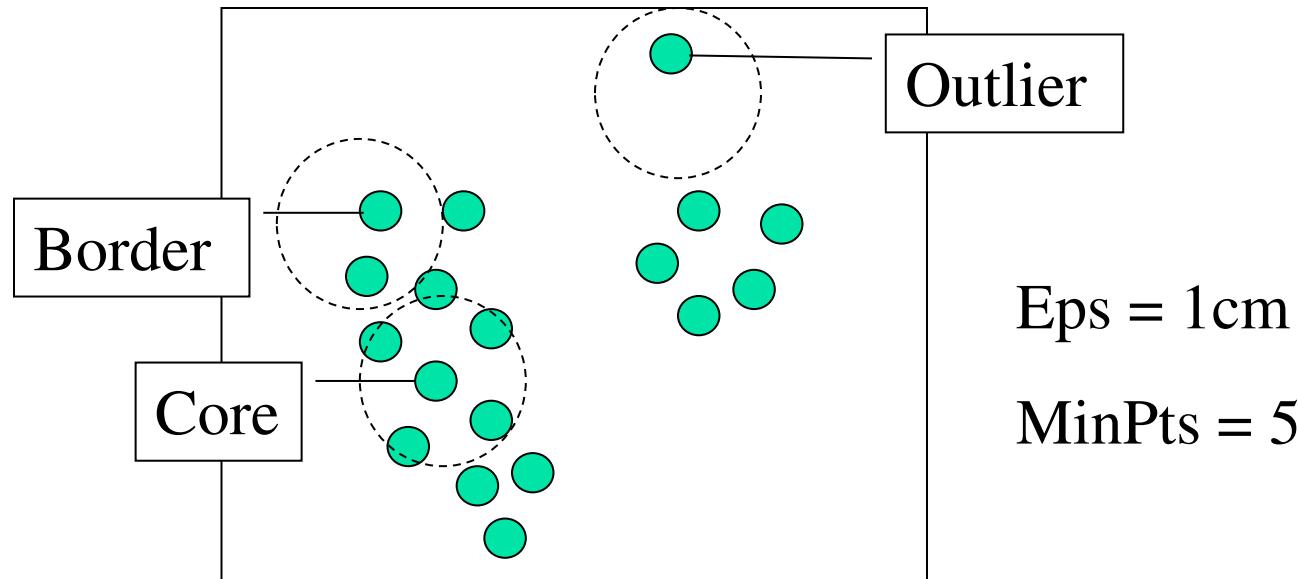
- Density-connected

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



DBSCAN: Density Based Spatial Clustering of Applications with Noise

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



DBSCAN: The Algorithm

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

DBSCAN: Sensitive to Parameters

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

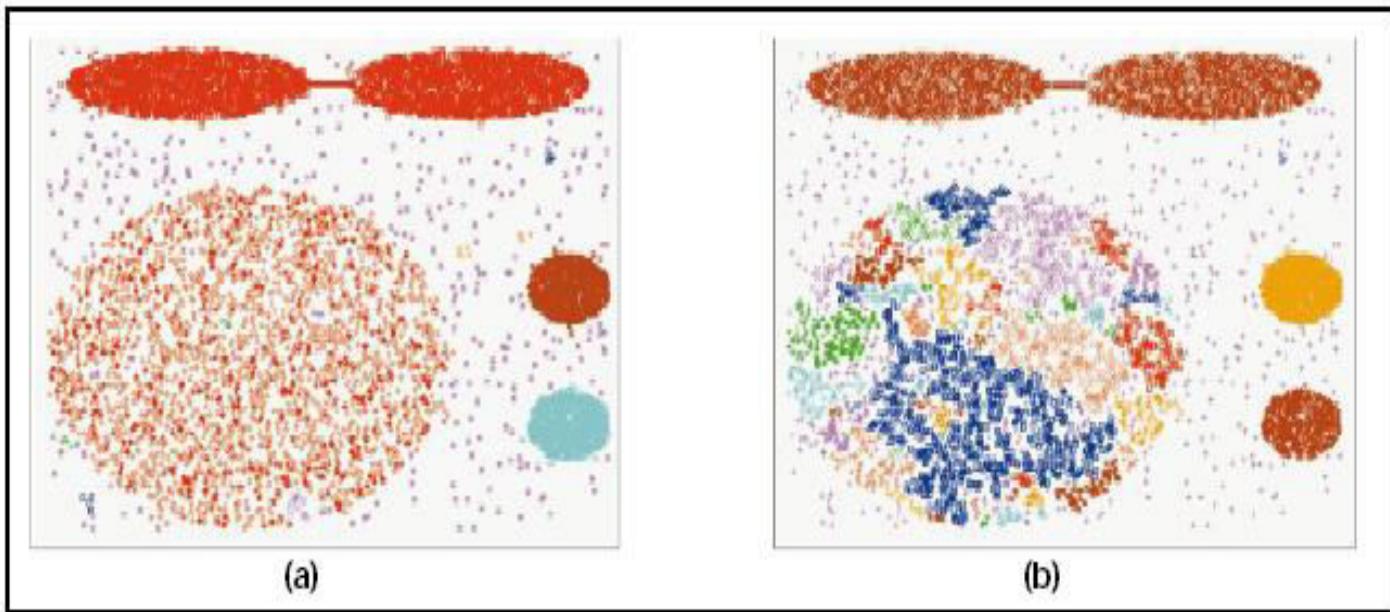
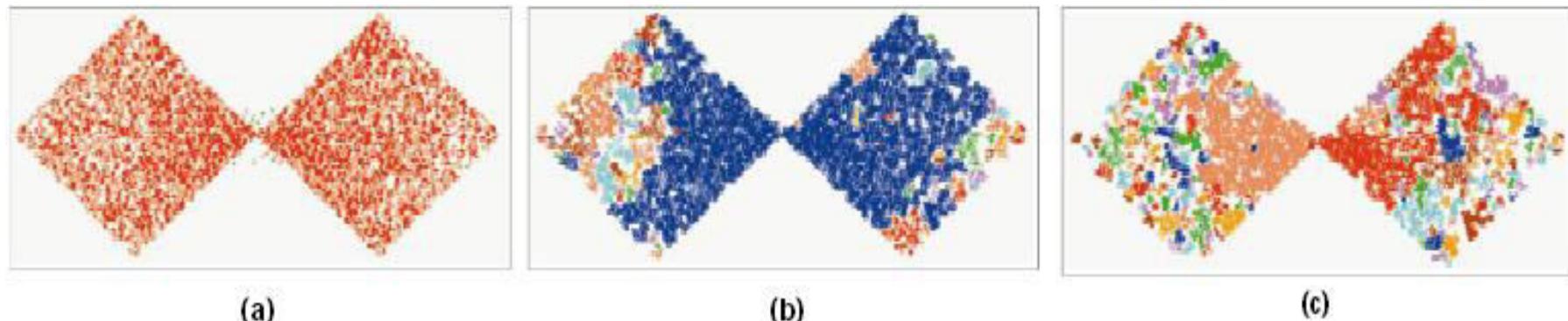
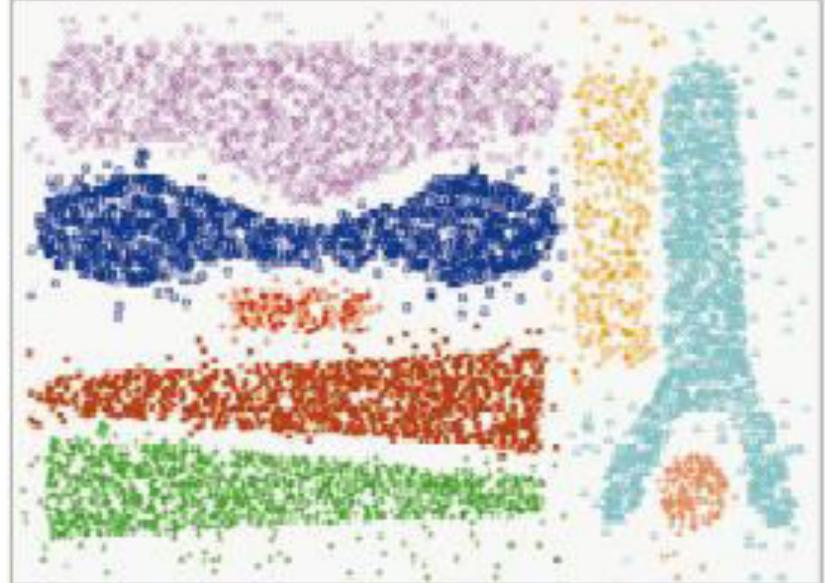
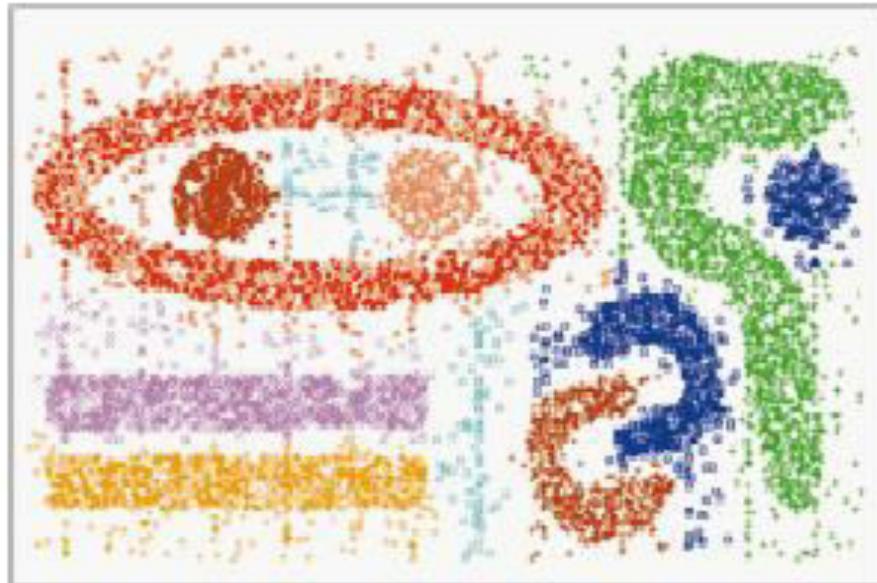
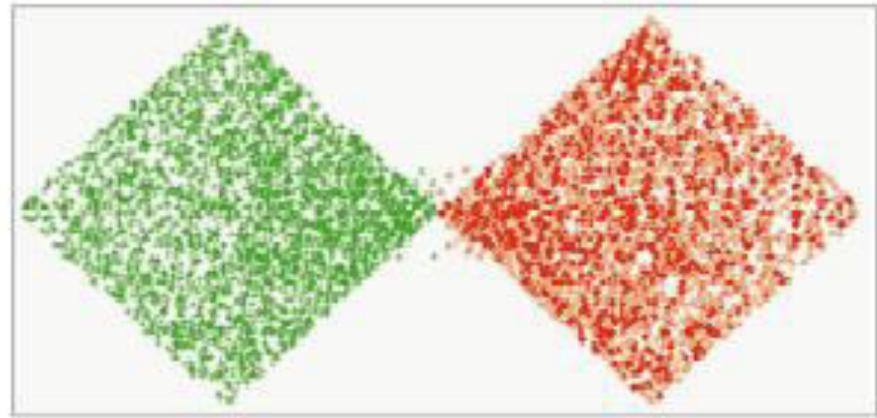
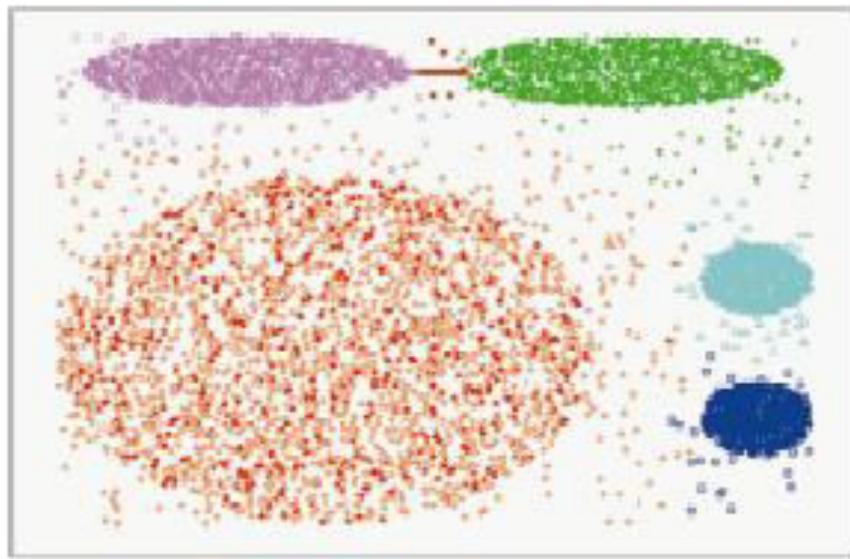


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



CHAMELEON (Clustering Complex Objects)



OPTICS: A Cluster-Ordering Method (1999)

- OPTICS: Ordering Points To Identify the Clustering Structure
 - Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)
 - Produces a special order of the database wrt its density-based clustering structure
 - This cluster-ordering contains info equiv to the density-based clusterings corresponding to a broad range of parameter settings
 - Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
 - Can be represented graphically or using visualization techniques

OPTICS: Some Extension from DBSCAN

- Index-based:
 - $k = \text{number of dimensions}$
 - $N = 20$
 - $p = 75\%$
 - $M = N(1-p) = 5$
- Complexity: $O(kN^2)$

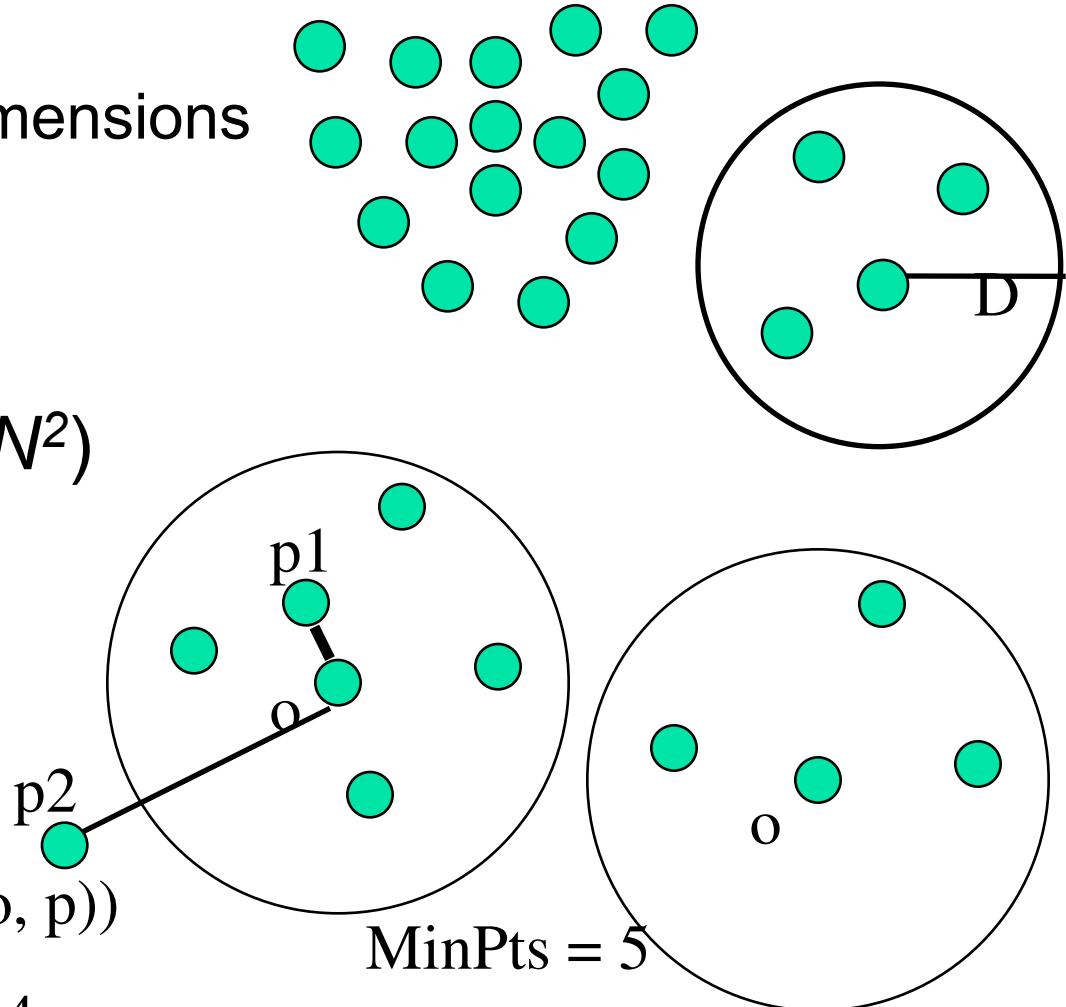
- Core Distance

- Reachability Distance

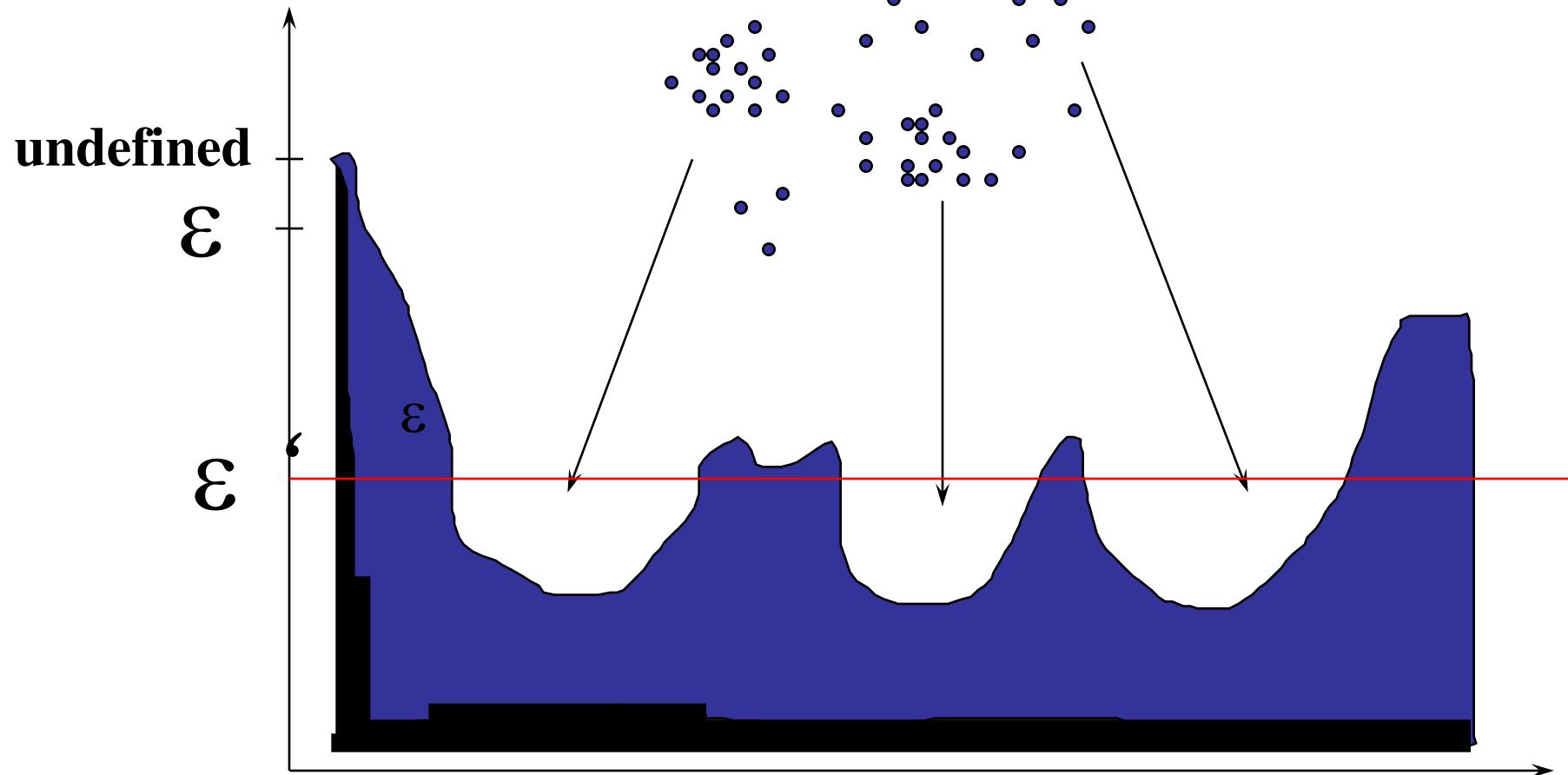
Max (core-distance (o), $d(o, p)$)

$$r(p_1, o) = 2.8\text{cm. } r(p_2, o) = 4\text{cm}$$

$$\varepsilon = 3 \text{ cm}$$

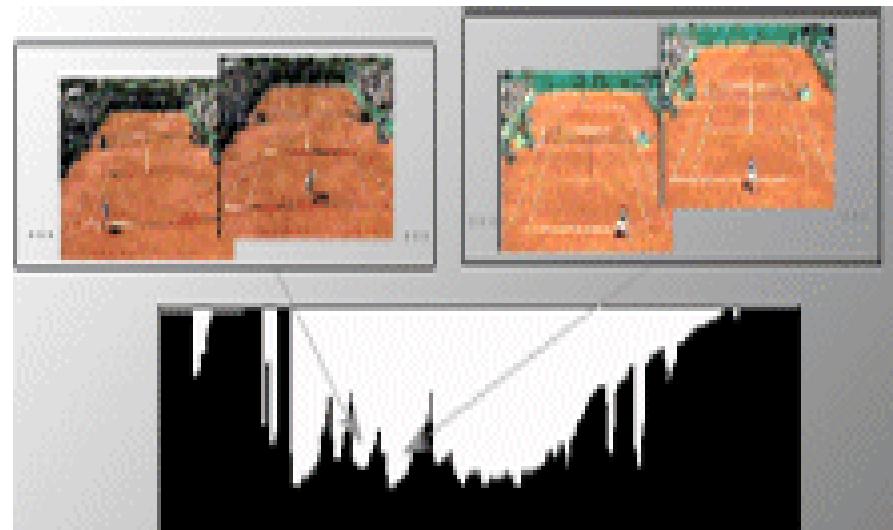
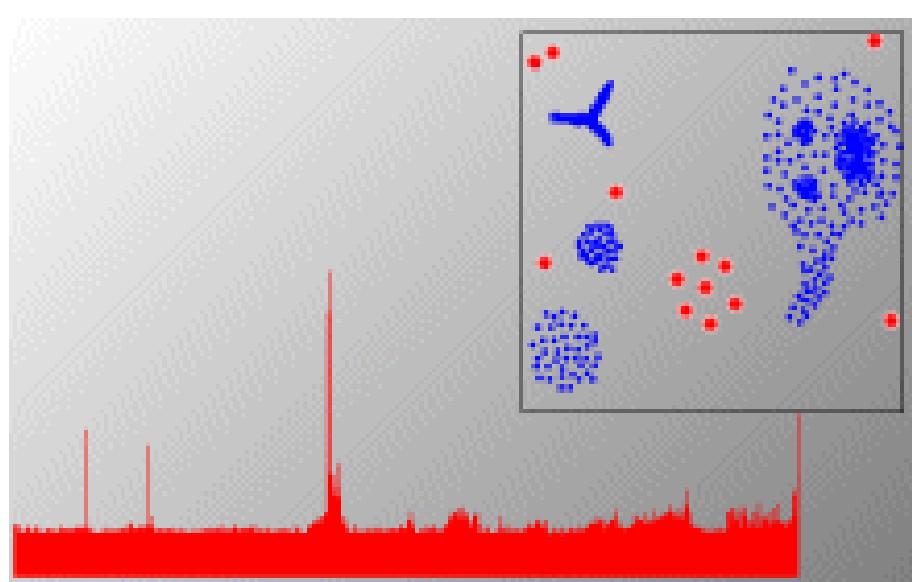
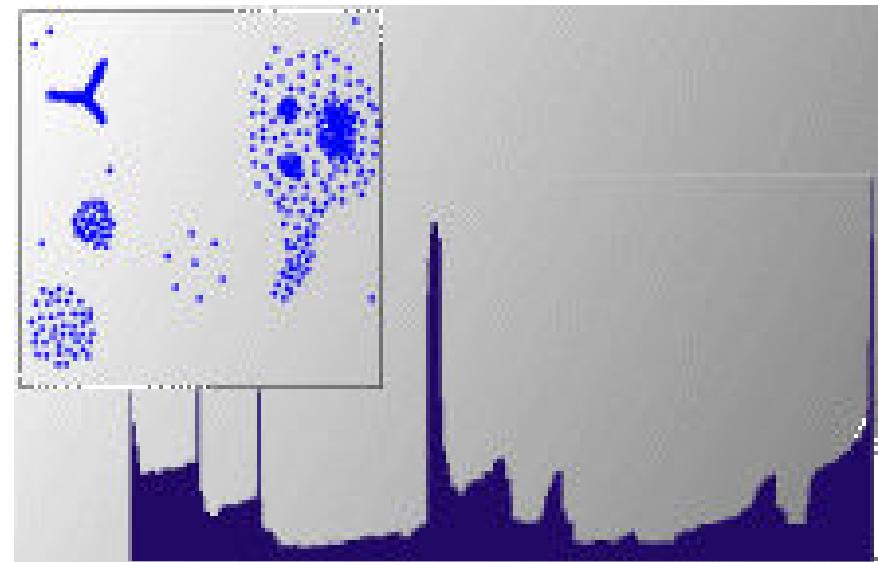
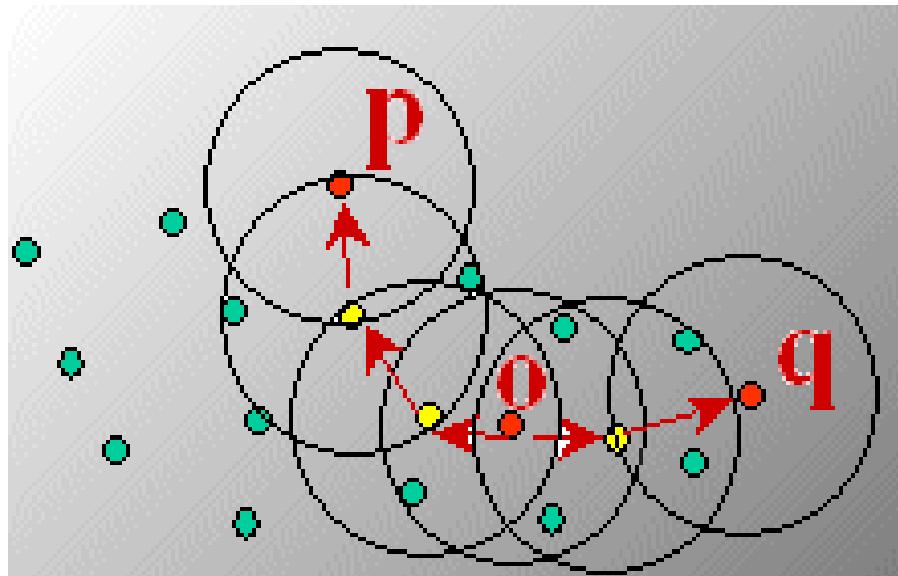


Reachability -distance



Cluster-order
of the objects

Density-Based Clustering: OPTICS & Its Applications



c

DENCLUE: Using Statistical Density Functions

- DENsity-based CLUstEring by Hinneburg & Keim (KDD'98)
- Using statistical density functions:

$$f_{Gaussian}(x, y) = e^{-\frac{d(x, y)^2}{2\sigma^2}}$$

$$f_{Gaussian}^D(x) = \sum_{i=1}^N e^{-\frac{d(x, x_i)^2}{2\sigma^2}}$$

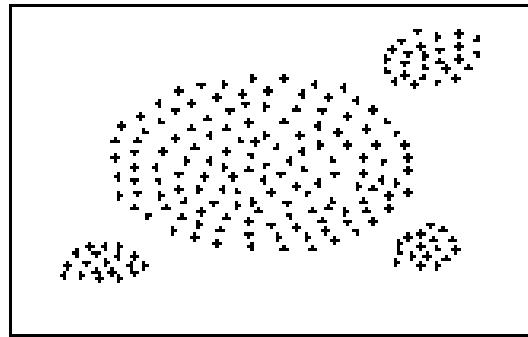
- Major features

- Solid mathematical foundation
- Good for data sets with large amounts of noise
- Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets
- Significant faster than existing algorithm (e.g., DBSCAN)
- But needs a large number of parameters

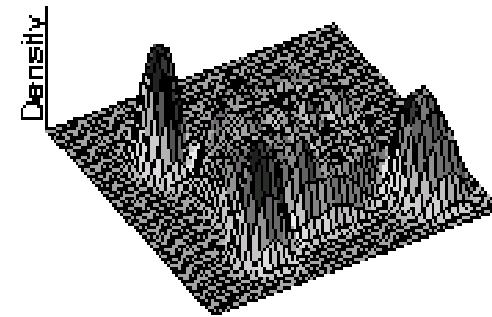
Denclue: Technical Essence

- Uses grid cells but only keeps information about grid cells that do actually contain data points and manages these cells in a tree-based access structure
- Influence function: describes the impact of a data point within its neighborhood
- Overall density of the data space can be calculated as the sum of the influence function of all data points
- Clusters can be determined mathematically by identifying density attractors
- Density attractors are local maximal of the overall density function

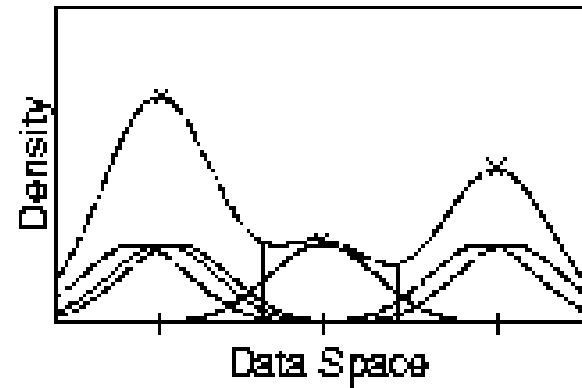
Density Attractor



(a) Data Set



(c) Gaussian



Center-Defined and Arbitrary

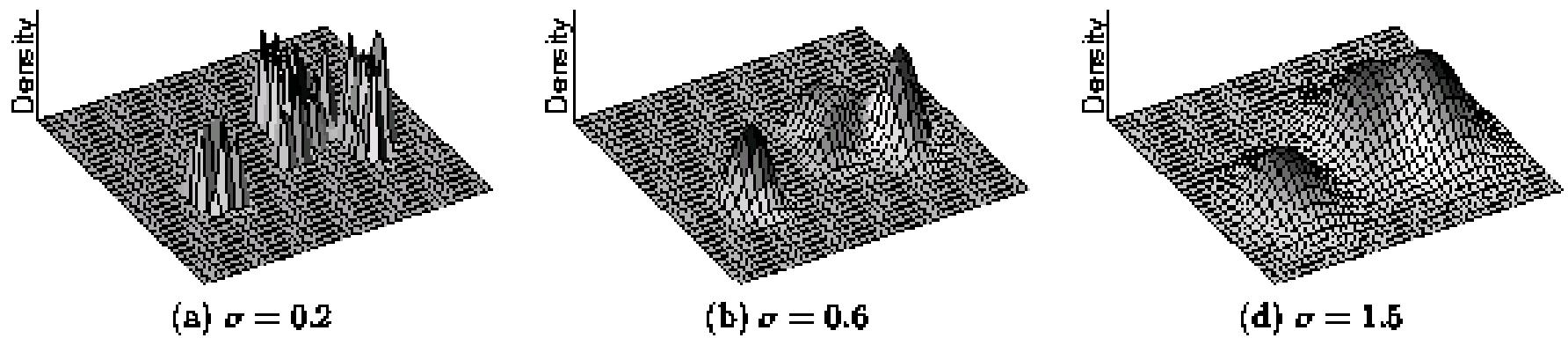


Figure 3: Example of Center-Defined Clusters for different σ

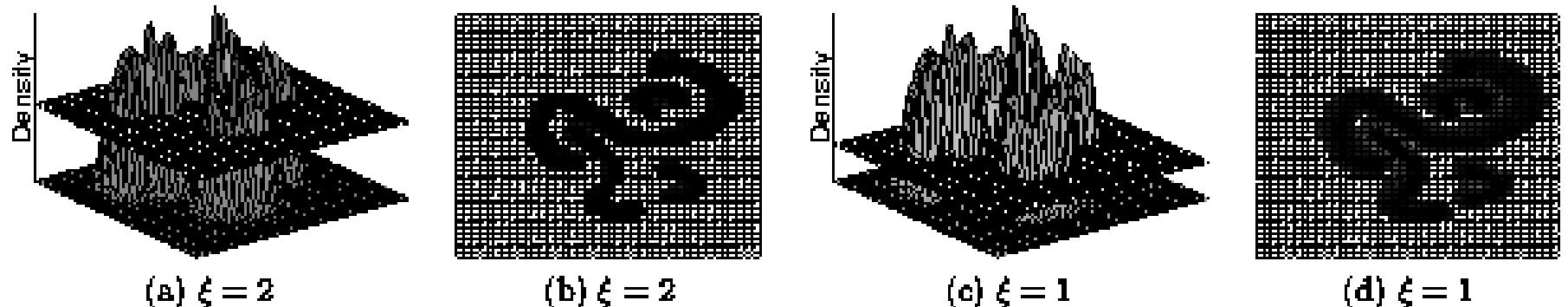


Figure 4: Example of Arbitrary-Shape Clusters for different ξ

Chapter 7. Cluster Analysis

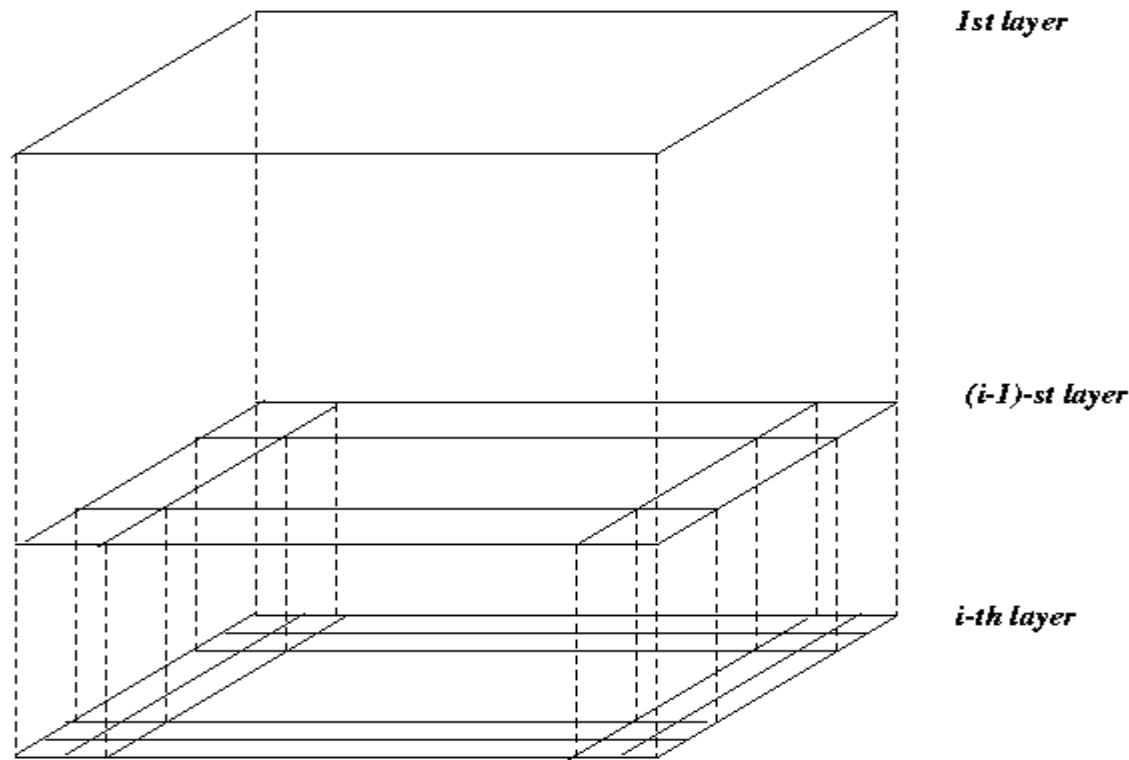
1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary

Grid-Based Clustering Method

- Using multi-resolution grid data structure
- Several interesting methods
 - **STING** (a STatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
 - **WaveCluster** by Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
 - A multi-resolution clustering approach using wavelet method
 - **CLIQUE**: Agrawal, et al. (SIGMOD'98)
 - On high-dimensional data (thus put in the section of clustering high-dimensional data)

STING: A Statistical Information Grid Approach

- Wang, Yang and Muntz (VLDB'97)
- The spatial area area is divided into rectangular cells
- There are several levels of cells corresponding to different levels of resolution



The STING Clustering Method

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
 - *count, mean, s, min, max*
 - type of distribution—normal, *uniform*, etc.
- Use a top-down approach to answer spatial data queries
- Start from a pre-selected layer—typically with a small number of cells
- For each cell in the current level compute the confidence interval

Comments on STING

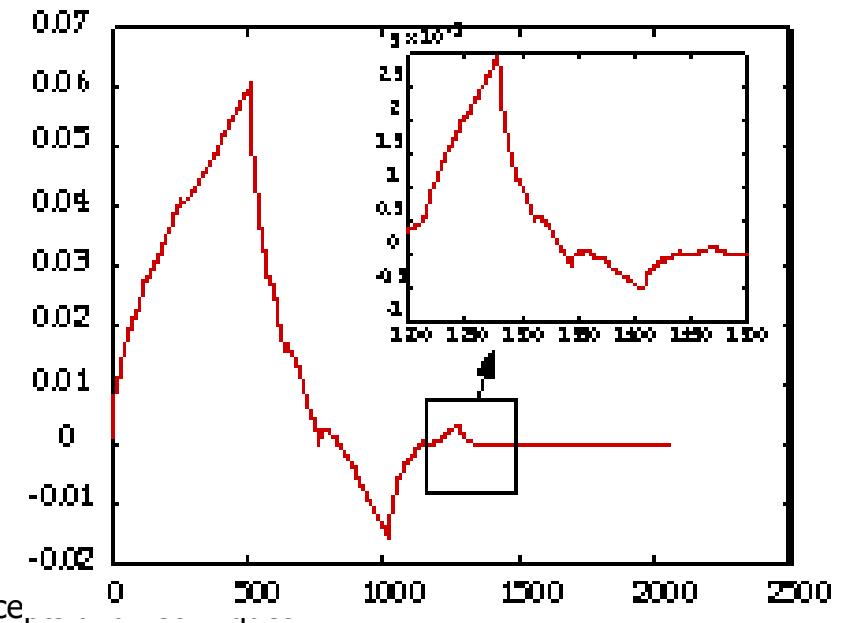
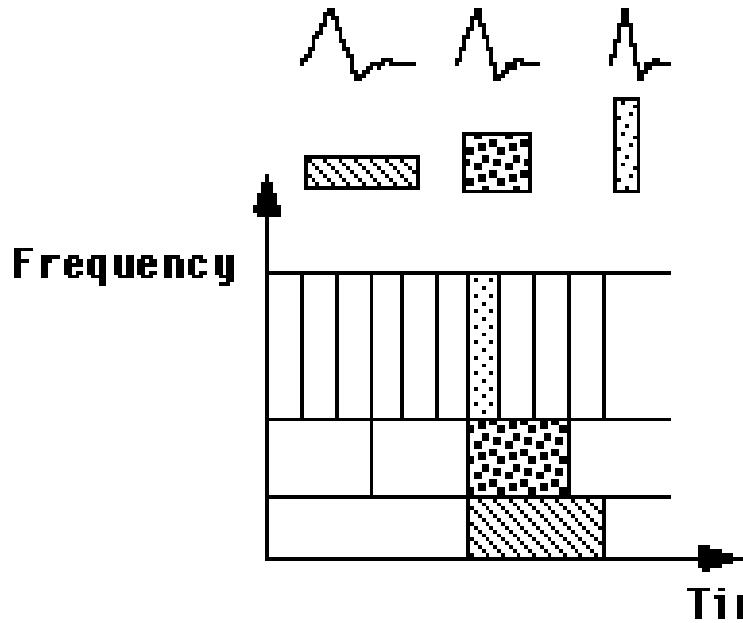
- Remove the irrelevant cells from further consideration
- When finish examining the current layer, proceed to the next lower level
- Repeat this process until the bottom layer is reached
- Advantages:
 - Query-independent, easy to parallelize, incremental update
 - $O(K)$, where K is the number of grid cells at the lowest level
- Disadvantages:
 - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

WaveCluster: Clustering by Wavelet Analysis (1998)

- Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
- A multi-resolution clustering approach which applies wavelet transform to the feature space
- How to apply wavelet transform to find clusters
 - Summarizes the data by imposing a multidimensional grid structure onto data space
 - These multidimensional spatial data objects are represented in a n-dimensional feature space
 - Apply wavelet transform on feature space to find the dense regions in the feature space
 - Apply wavelet transform multiple times which result in clusters at different scales from fine to coarse

Wavelet Transform

- Wavelet transform: A signal processing technique that decomposes a signal into different frequency sub-band (can be applied to n-dimensional signals)
- Data are transformed to preserve relative distance between objects at different levels of resolution
- Allows natural clusters to become more distinguishable



The WaveCluster Algorithm

- Input parameters
 - # of grid cells for each dimension
 - the wavelet, and the # of applications of wavelet transform
- Why is wavelet transformation useful for clustering?
 - Use hat-shape filters to emphasize region where points cluster, but simultaneously suppress weaker information in their boundary
 - Effective removal of outliers, multi-resolution, cost effective
- Major features:
 - Complexity $O(N)$
 - Detect arbitrary shaped clusters at different scales
 - Not sensitive to noise, not sensitive to input order
 - Only applicable to low dimensional data
- Both grid-based and density-based

Quantization & Transformation

- First, quantize data into m-D grid structure, then wavelet transform
 - a) scale 1: high resolution
 - b) scale 2: medium resolution
 - c) scale 3: low resolution

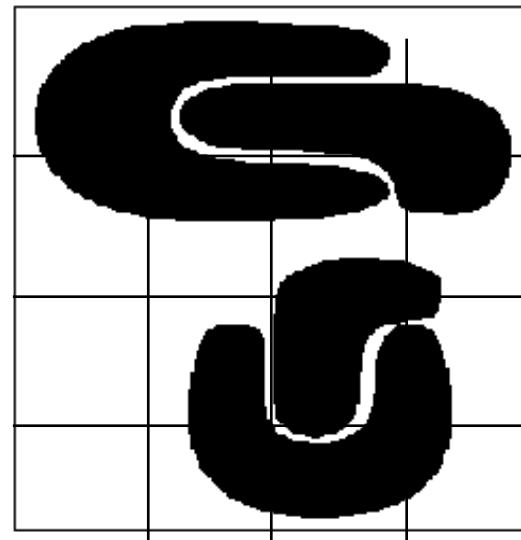
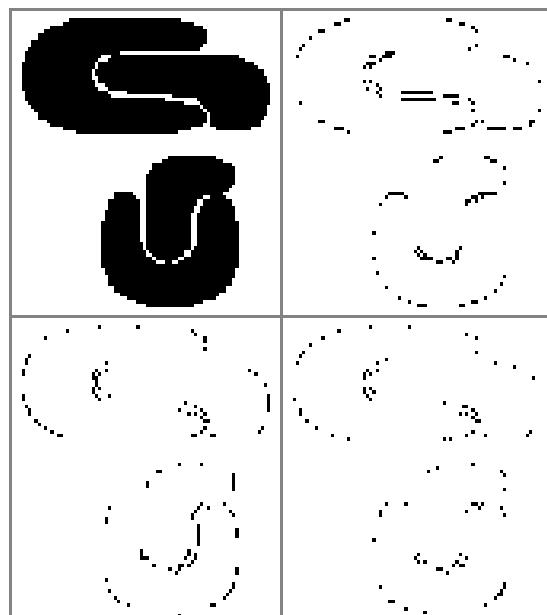


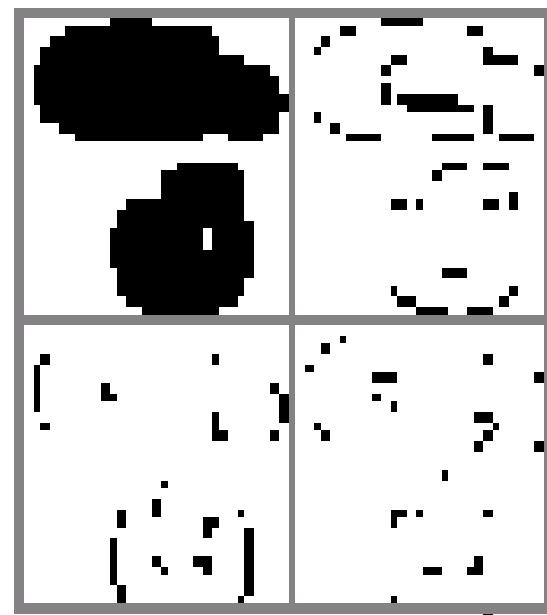
Figure 1: A sample 2-dimensional feature space.



a)



b)



c)

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary

Model-Based Clustering

- What is model-based clustering?
 - Attempt to optimize the fit between the given data and some mathematical model
 - Based on the assumption: Data are generated by a mixture of underlying probability distribution
- Typical methods
 - Statistical approach
 - EM (Expectation maximization), AutoClass
 - Machine learning approach
 - COBWEB, CLASSIT
 - Neural network approach
 - SOM (Self-Organizing Feature Map)

EM — Expectation Maximization

- EM — A popular iterative refinement algorithm
- An extension to k-means
 - Assign each object to a cluster according to a weight (prob. distribution)
 - New means are computed based on weighted measures
- General idea
 - Starts with an initial estimate of the parameter vector
 - Iteratively rescores the patterns against the mixture density produced by the parameter vector
 - The rescored patterns are used to update the parameter updates
 - Patterns belonging to the same cluster, if they are placed by their scores in a particular component
- Algorithm converges fast but may not be in global optima

The EM (Expectation Maximization) Algorithm

- Initially, randomly assign k cluster centers
- Iteratively refine the clusters based on two steps
 - Expectation step: assign each data point X_i to cluster C_i with the following probability

$$P(X_i \in C_k) = p(C_k | X_i) = \frac{p(C_k)p(X_i | C_k)}{p(X_i)},$$

- Maximization step:
 - Estimation of model parameters

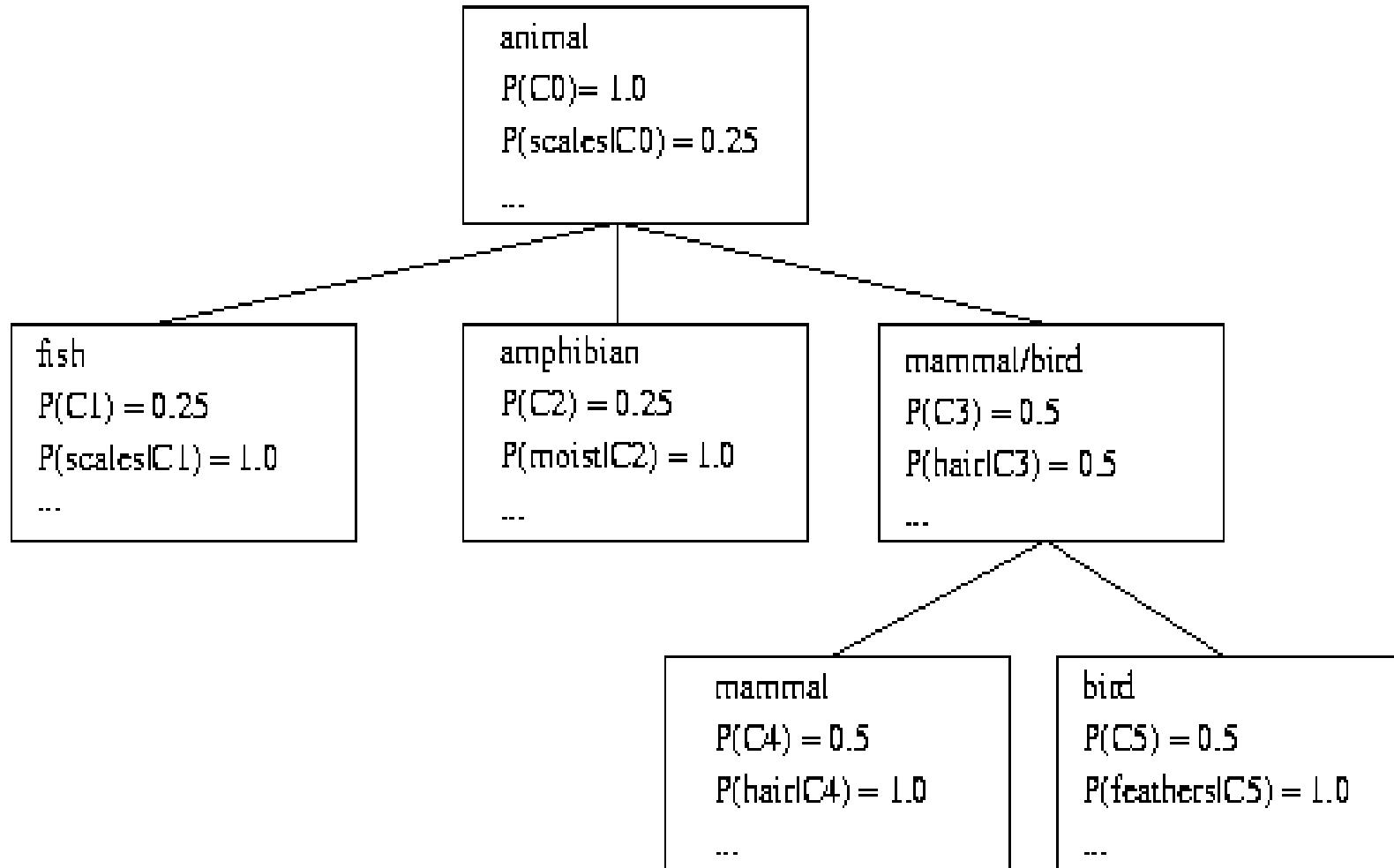
$$m_k = \frac{1}{N} \sum_{i=1}^N \frac{X_i P(X_i \in C_k)}{\sum_j P(X_i \in C_j)}.$$

Conceptual Clustering

- Conceptual clustering
 - A form of clustering in machine learning
 - Produces a classification scheme for a set of unlabeled objects
 - Finds characteristic description for each concept (class)
- COBWEB (Fisher'87)
 - A popular a simple method of incremental conceptual learning
 - Creates a hierarchical clustering in the form of a **classification tree**
 - Each node refers to a concept and contains a probabilistic description of that concept

COBWEB Clustering Method

A classification tree



More on Conceptual Clustering

- Limitations of COBWEB
 - The assumption that the attributes are independent of each other is often too strong because correlation may exist
 - Not suitable for clustering large database data – skewed tree and expensive probability distributions
- CLASSIT
 - an extension of COBWEB for incremental clustering of continuous data
 - suffers similar problems as COBWEB
- AutoClass (Cheeseman and Stutz, 1996)
 - Uses Bayesian statistical analysis to estimate the number of clusters
 - Popular in industry

Neural Network Approach

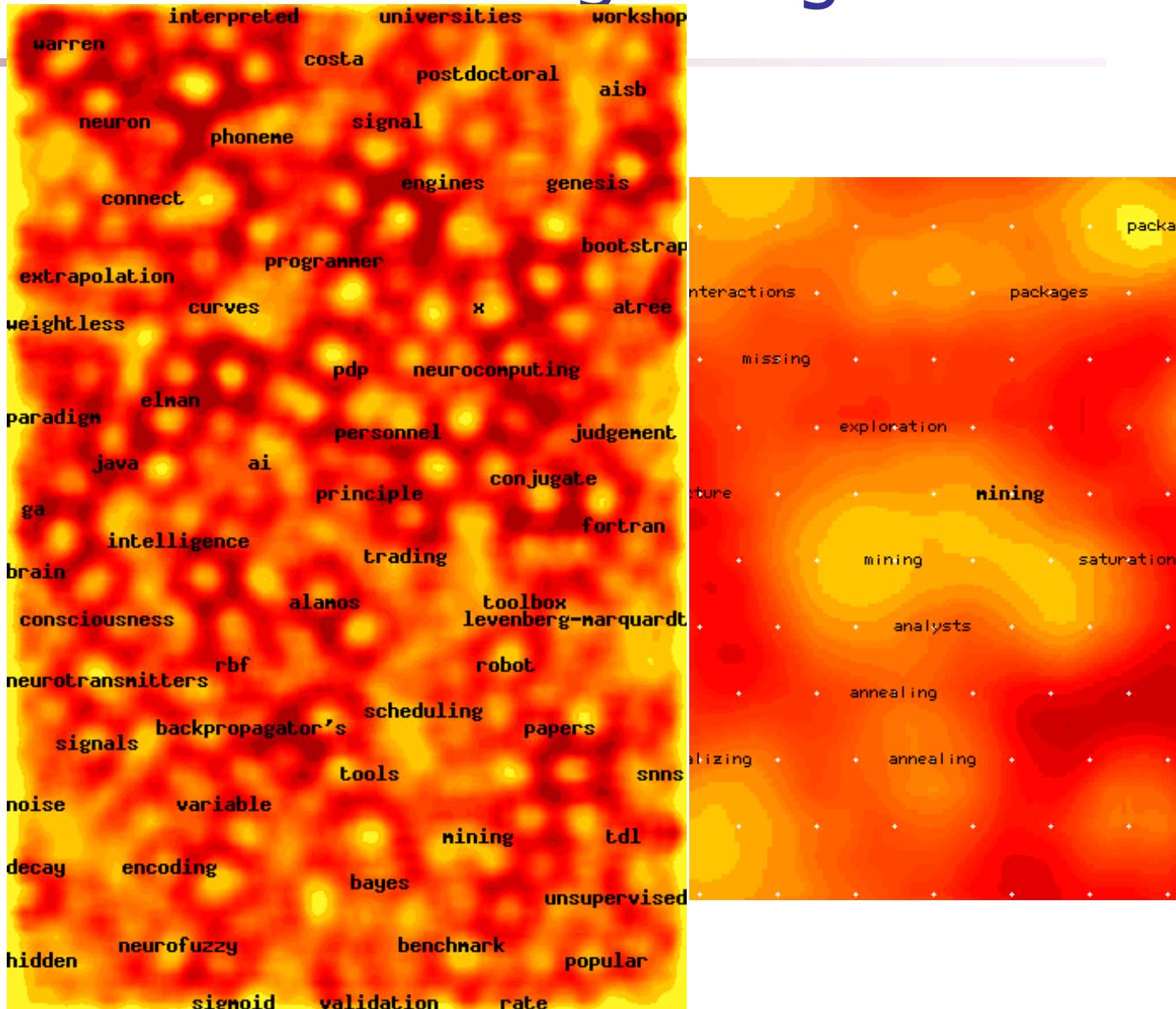
- Neural network approaches
 - Represent each cluster as an exemplar, acting as a “prototype” of the cluster
 - New objects are distributed to the cluster whose exemplar is the most similar according to some distance measure
- Typical methods
 - SOM (Soft-Organizing feature Map)
 - Competitive learning
 - Involves a hierarchical architecture of several units (neurons)
 - Neurons compete in a “winner-takes-all” fashion for the object currently being presented

Self-Organizing Feature Map (SOM)

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

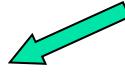
Web Document Clustering Using SOM

- The result of SOM clustering of 12088 Web articles
- The picture on the right: drilling down on the keyword “mining”
- Based on websom.hut.fi Web page



Chapter 6. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



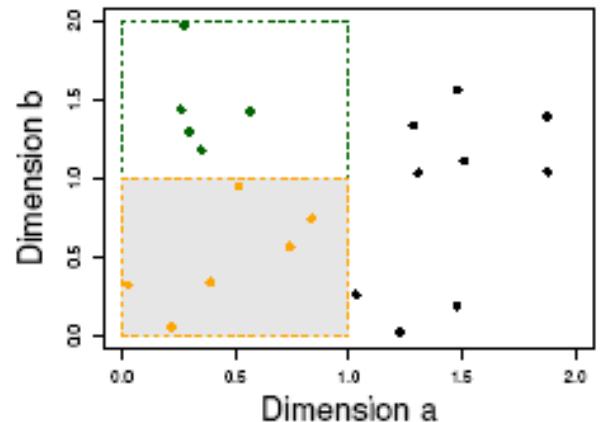
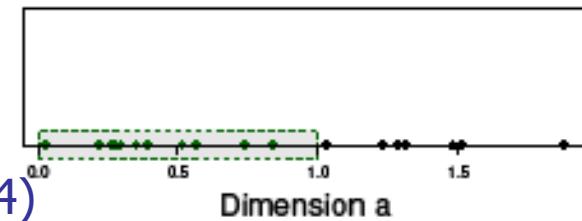
Clustering High-Dimensional Data

- Clustering high-dimensional data
 - Many applications: text documents, DNA micro-array data
 - Major challenges:
 - Many irrelevant dimensions may mask clusters
 - Distance measure becomes meaningless—due to equi-distance
 - Clusters may exist only in some subspaces
- Methods
 - Feature transformation: only effective if most dimensions are relevant
 - PCA & SVD useful only when features are highly correlated/redundant
 - Feature selection: wrapper or filter approaches
 - useful to find a subspace where the data have nice clusters
 - Subspace-clustering: find clusters in all the possible subspaces
 - CLIQUE, ProClus, and frequent pattern-based clustering

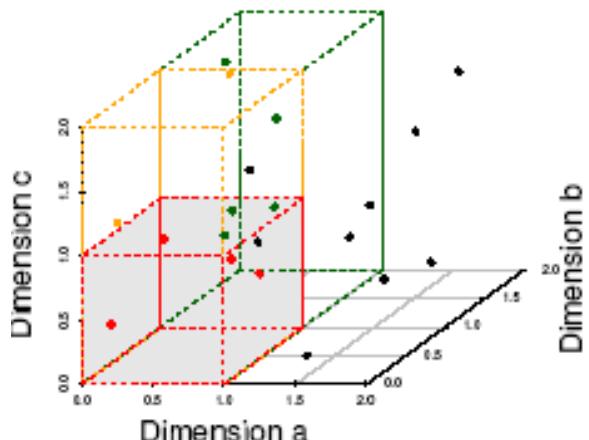
The Curse of Dimensionality

(graphs adapted from Parsons et al. KDD Explorations 2004)

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



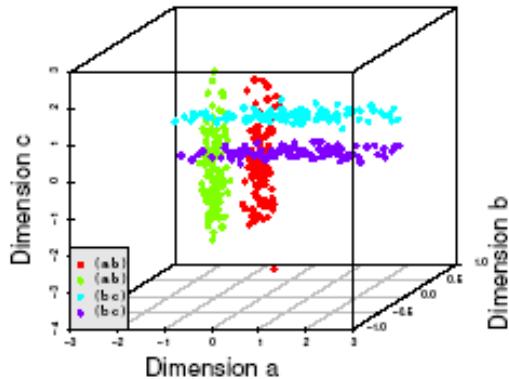
(b) 6 Objects in One Unit Bin



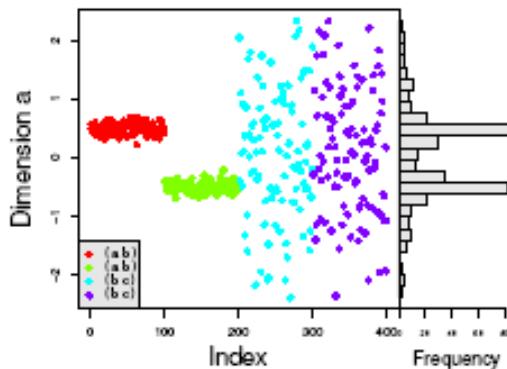
(c) 4 Objects in One Unit Bin

Why Subspace Clustering?

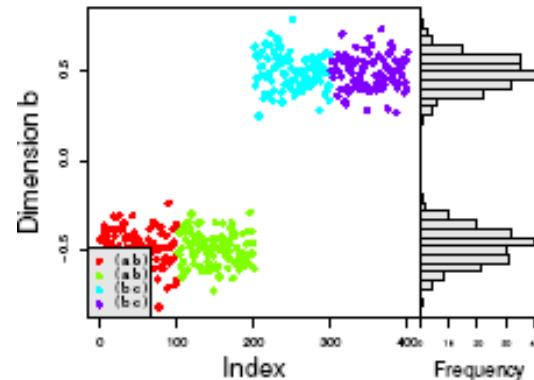
(adapted from Parsons et al. SIGKDD Explorations 2004)



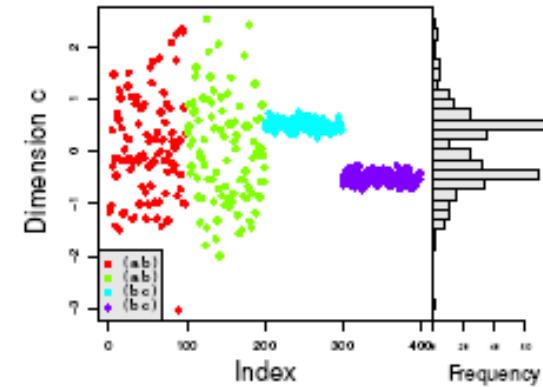
- Clusters may exist only in some subspaces
- Subspace-clustering: find clusters in all the subspaces



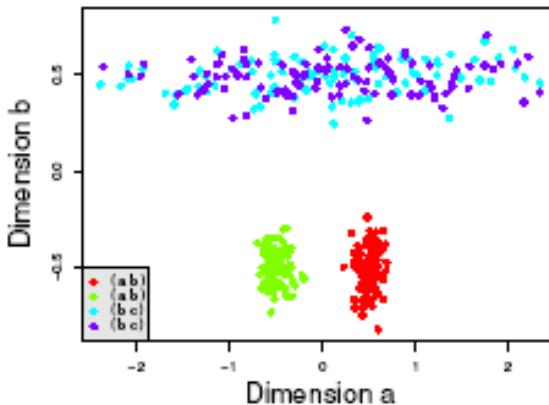
(a) Dimension a



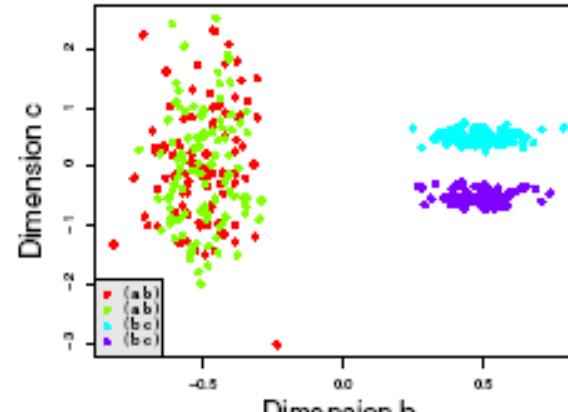
(b) Dimension b



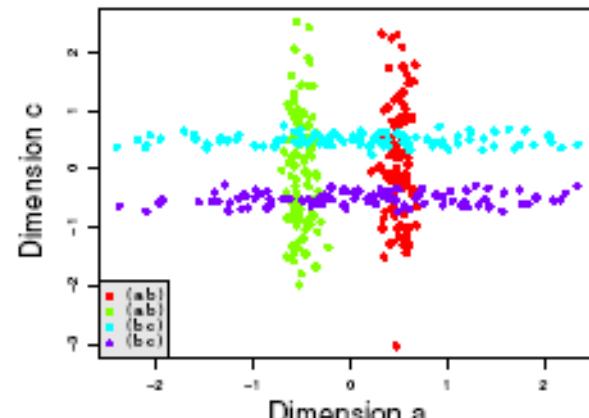
(c) Dimension c



(a) Dims a & b



(b) Dims b & c



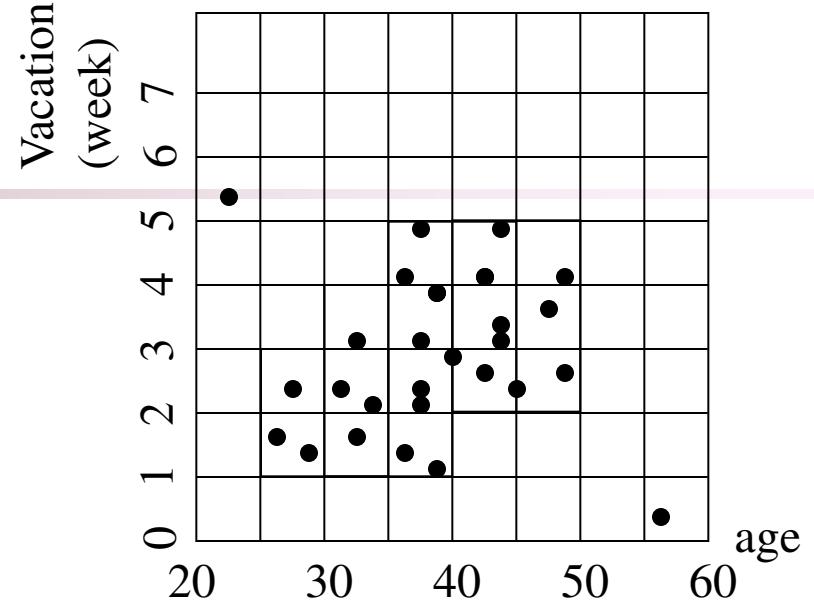
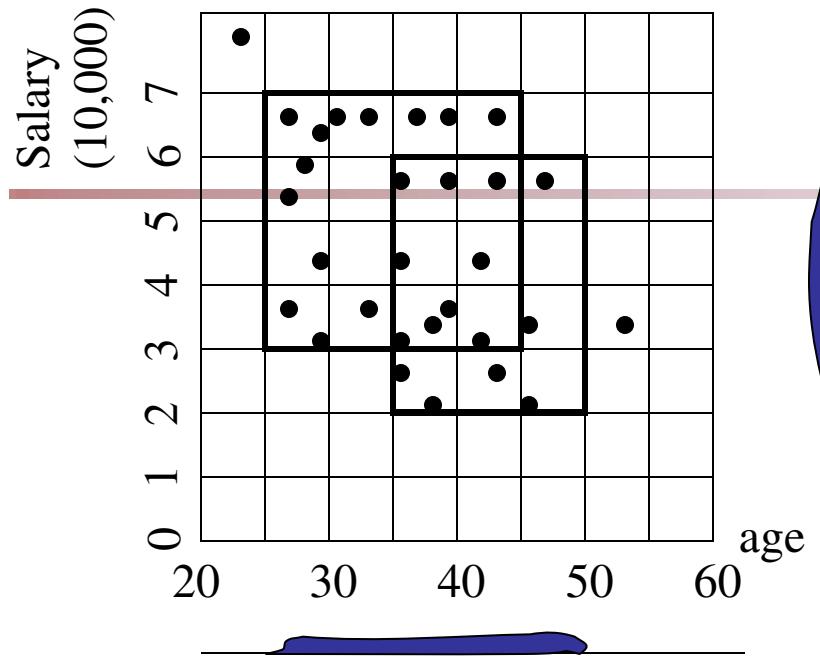
(c) Dims a & c

CLIQUE (Clustering In QUEst)

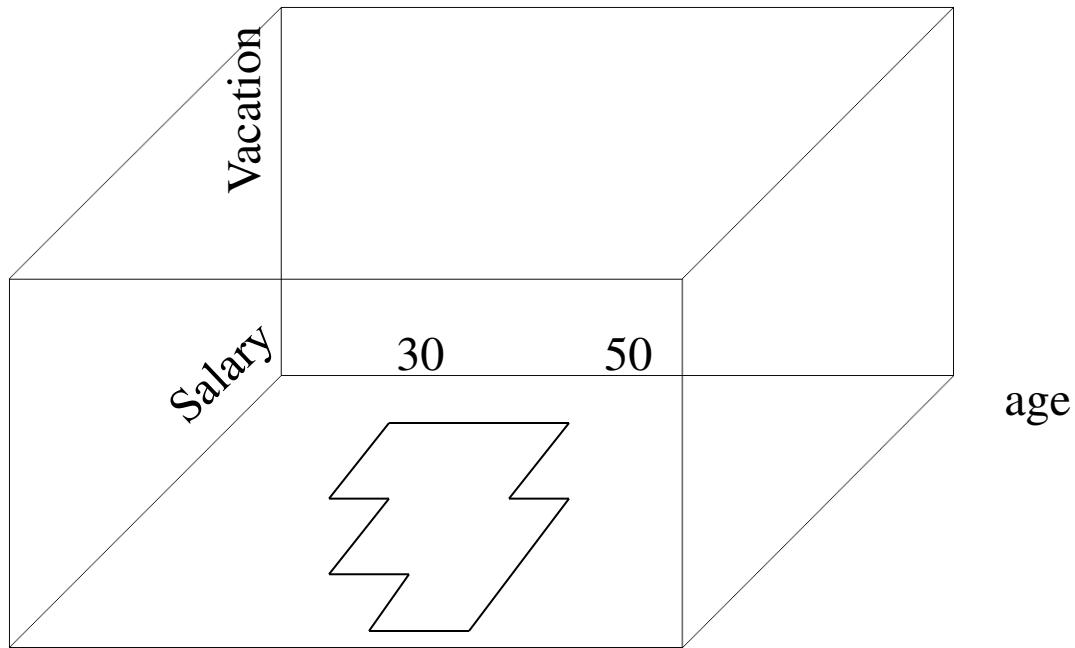
- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98)
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space
- CLIQUE can be considered as both density-based and grid-based
 - It partitions each dimension into the same number of equal length interval
 - It partitions an m -dimensional data space into non-overlapping rectangular units
 - A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter
 - A cluster is a maximal set of connected dense units within a subspace

CLIQUE: The Major Steps

- Partition the data space and find the number of points that lie inside each cell of the partition.
- Identify the subspaces that contain clusters using the Apriori principle
- Identify clusters
 - Determine dense units in all subspaces of interests
 - Determine connected dense units in all subspaces of interests.
- Generate minimal description for the clusters
 - Determine maximal regions that cover a cluster of connected dense units for each cluster
 - Determination of minimal cover for each cluster



$\tau = 3$



Strength and Weakness of *CLIQUE*

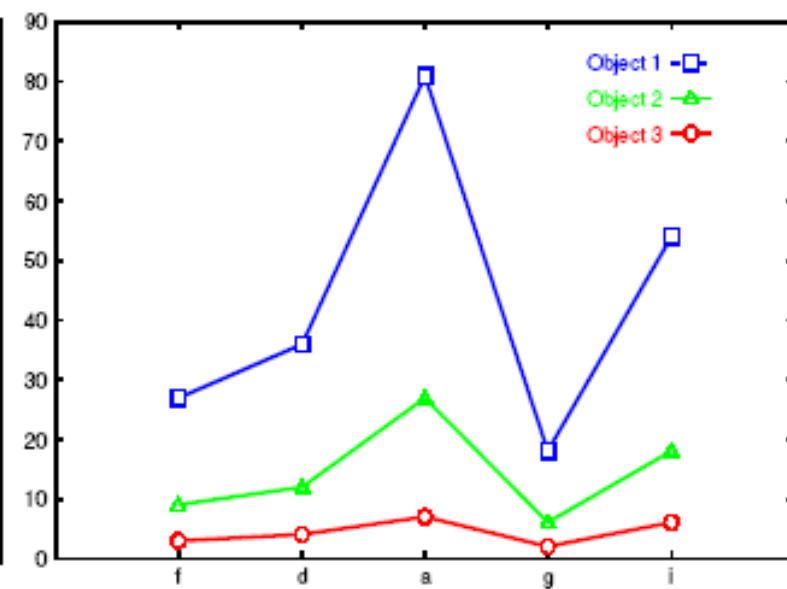
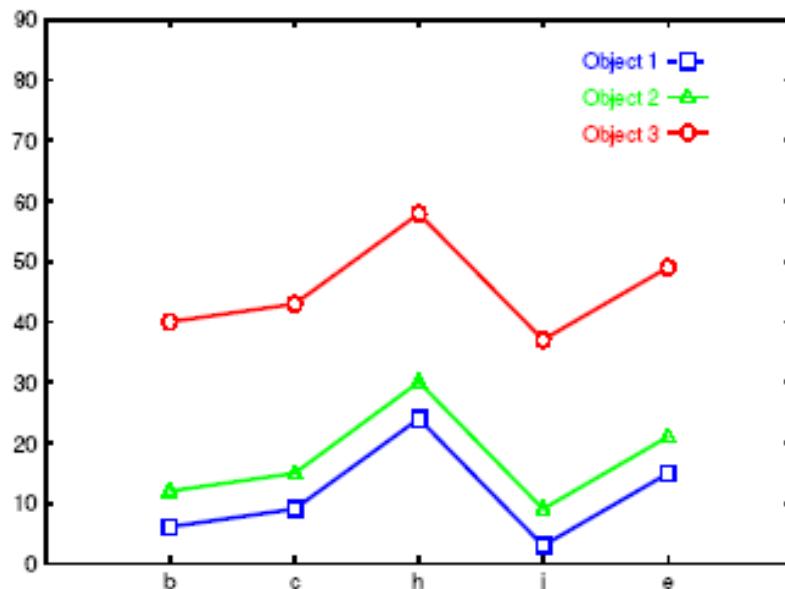
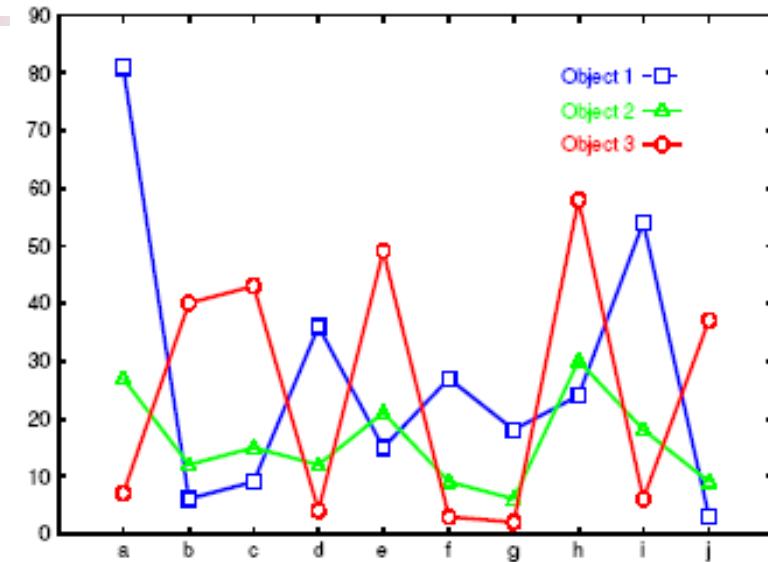
- Strength
 - automatically finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
 - *insensitive* to the order of records in input and does not presume some canonical data distribution
 - scales *linearly* with the size of input and has good scalability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

Frequent Pattern-Based Approach

- Clustering high-dimensional space (e.g., clustering text documents, microarray data)
 - Projected subspace-clustering: which dimensions to be projected on?
 - CLIQUE, ProClus
 - Feature extraction: costly and may not be effective?
 - Using frequent patterns as “features”
 - “Frequent” are inherent features
 - Mining freq. patterns may not be so expensive
- Typical methods
 - Frequent-term-based document clustering
 - Clustering by pattern similarity in micro-array data (pClustering)

Clustering by Pattern Similarity (p -Clustering)

- Right: The micro-array “raw” data shows 3 genes and their values in a multi-dimensional space
 - Difficult to find their patterns
- Bottom: Some subsets of dimensions form nice **shift** and **scaling** patterns



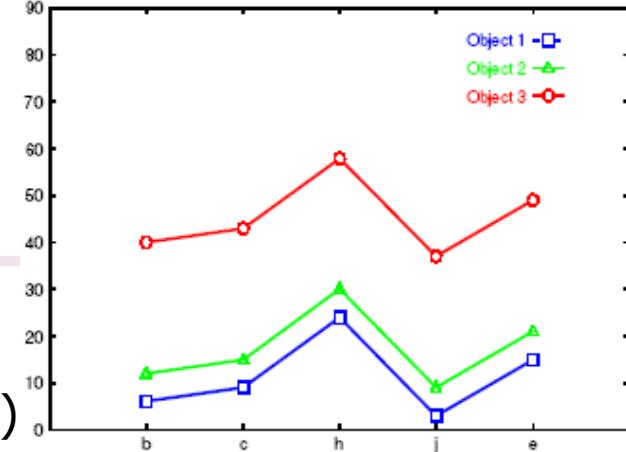
Why p -Clustering?

- Microarray data analysis may need to
 - Clustering on thousands of dimensions (attributes)
 - Discovery of both **shift** and **scaling** patterns
- Clustering with Euclidean distance measure? — cannot find shift patterns
- Clustering on derived attribute $A_{ij} = a_i - a_j$? — introduces **N(N-1)** dimensions
- Bi-cluster using transformed mean-squared residue score matrix (I, J)

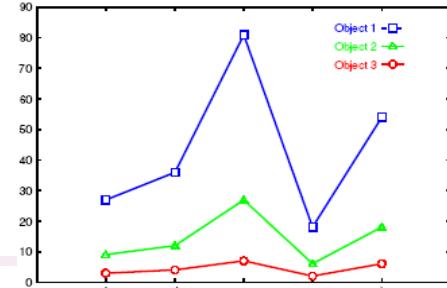
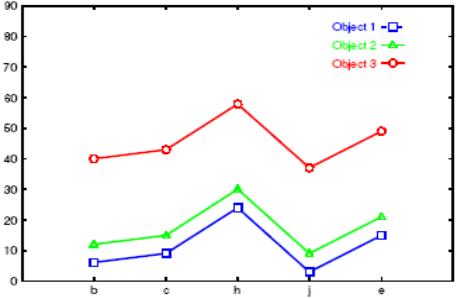
$$H(IJ) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (d_{ij} - d_{iJ} - d_{IJ} + d_{IJ})^2$$

- Where $d_{ij} = \frac{1}{|J|} \sum_{j \in J} d_{ij}$ $d_{iJ} = \frac{1}{|I|} \sum_{i \in I} d_{ij}$ $d_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} d_{ij}$
- A submatrix is a δ -cluster if $H(I, J) \leq \delta$ for some $\delta > 0$

- Problems with bi-cluster
 - No downward closure property,
 - Due to averaging, it may contain outliers but still within δ -threshold



p -Clustering: Clustering by Pattern Similarity



- Given object x, y in O and features a, b in T , p Cluster is a 2 by 2 matrix

$$pScore\left(\begin{bmatrix} d_{xa} & d_{xb} \\ d_{ya} & d_{yb} \end{bmatrix}\right) = |(d_{xa} - d_{xb}) - (d_{ya} - d_{yb})|$$

- A pair (O, T) is in δ - p Cluster if for any 2 by 2 matrix X in (O, T) , $pScore(X) \leq \delta$ for some $\delta > 0$
- Properties of δ - p Cluster
 - Downward closure
 - Clusters are more homogeneous than bi-cluster (thus the name: pair-wise Cluster)
- Pattern-growth algorithm has been developed for efficient mining
- For scaling patterns, one can observe, taking logarithmic on $\frac{d_{xa}}{d_{xb}} / \frac{d_{ya}}{d_{yb}} < \delta$ will lead to the p Score form

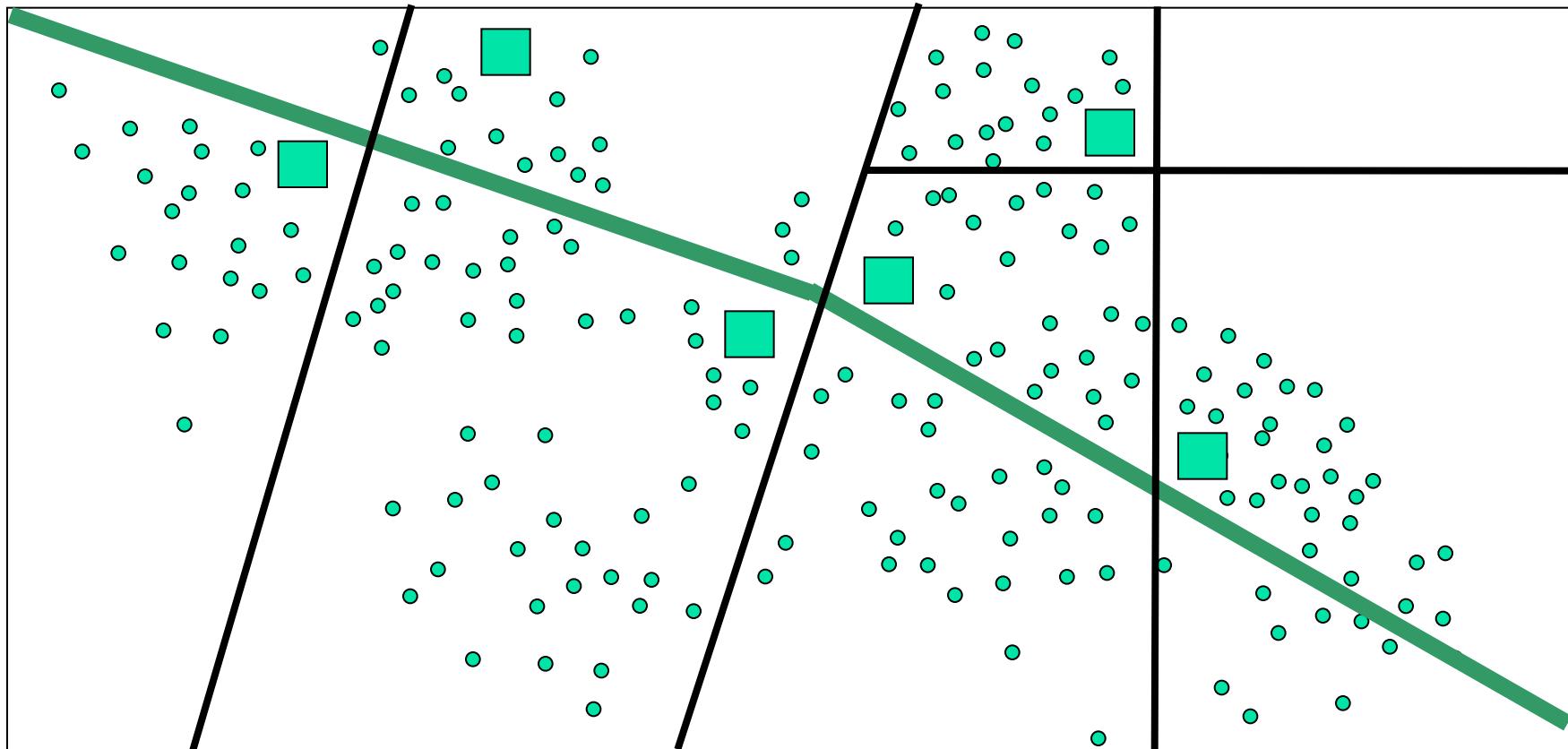
Chapter 6. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



Why Constraint-Based Cluster Analysis?

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

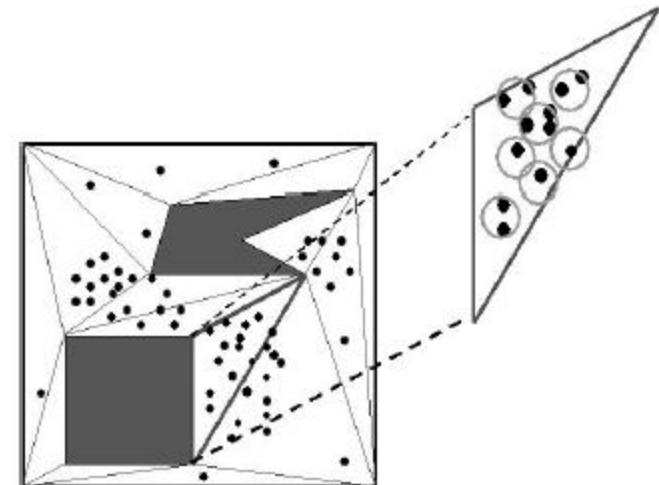
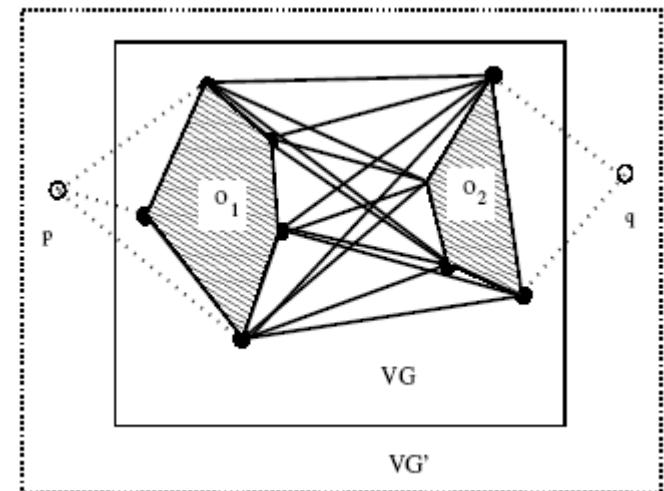


A Classification of Constraints in Cluster Analysis

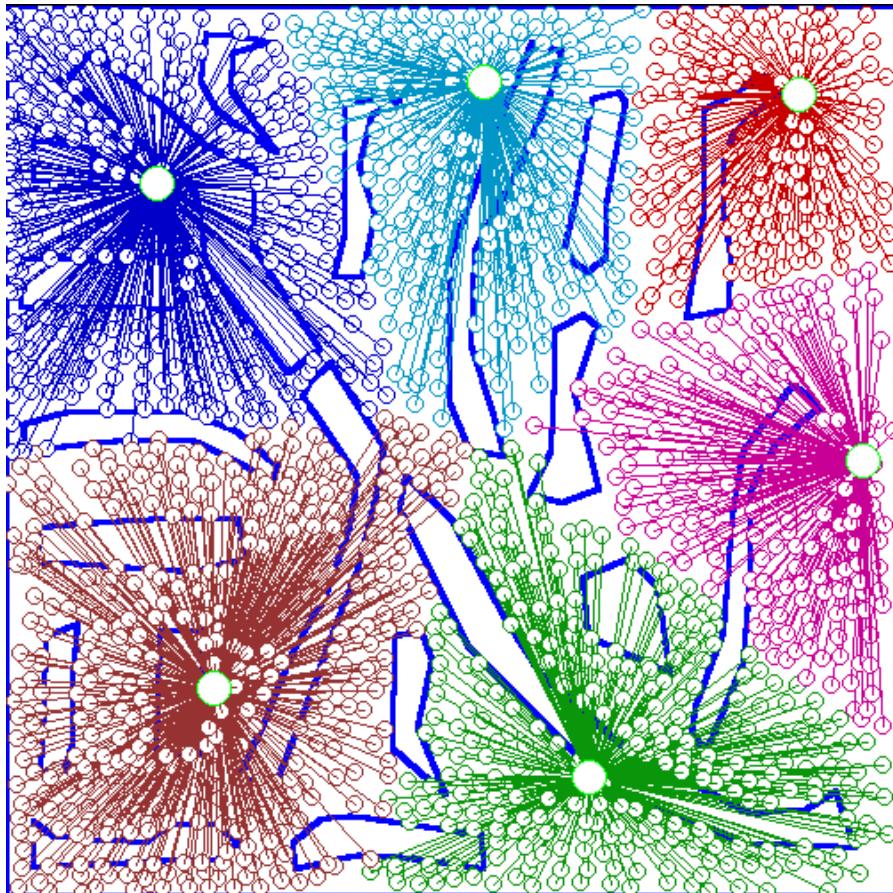
- Clustering in applications: desirable to have user-guided (i.e., constrained) cluster analysis
- Different constraints in cluster analysis:
 - Constraints on individual objects (do selection first)
 - Cluster on houses worth over \$300K
 - Constraints on distance or similarity functions
 - Weighted functions, obstacles (e.g., rivers, lakes)
 - Constraints on the selection of clustering parameters
 - # of clusters, MinPts, etc.
 - User-specified constraints
 - Contain at least 500 valued customers and 5000 ordinary ones
 - Semi-supervised: giving small training sets as “constraints” or hints

Clustering With Obstacle Objects

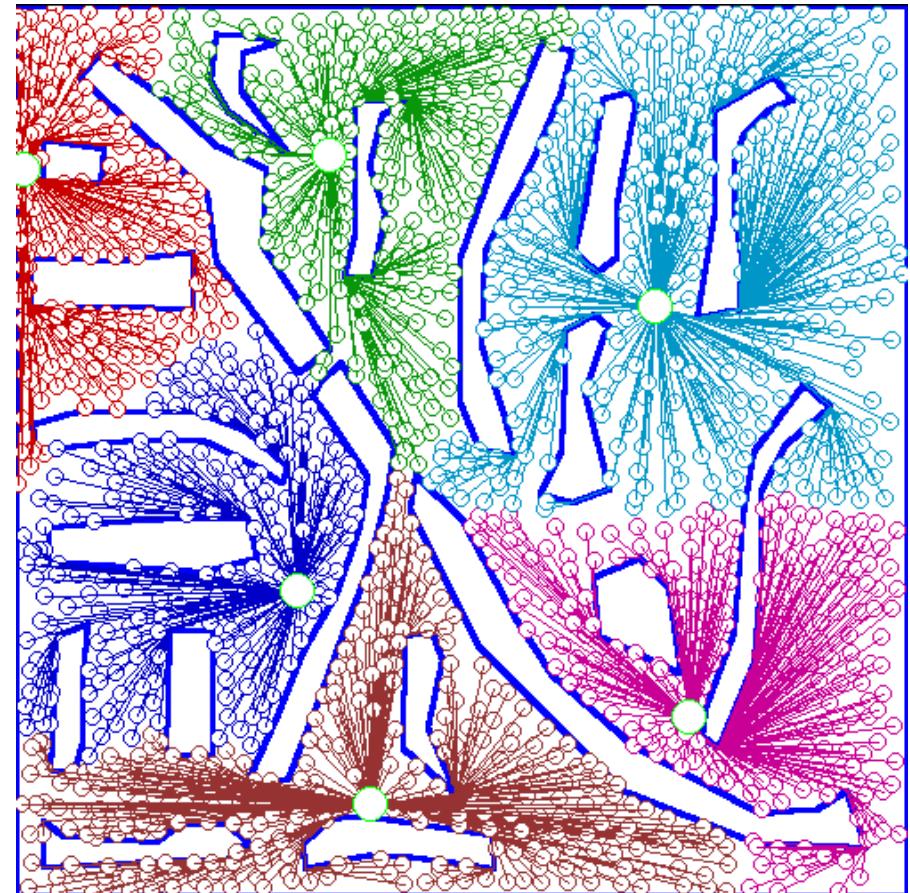
- K-medoids is more preferable since k-means may locate the ATM center in the middle of a lake
- Visibility graph and shortest path
- Triangulation and micro-clustering
- Two kinds of join indices (shortest-paths) worth pre-computation
 - VV index: indices for any pair of obstacle vertices
 - MV index: indices for any pair of micro-cluster and obstacle indices



An Example: Clustering With Obstacle Objects



Not Taking obstacles into account



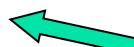
Taking obstacles into account

Clustering with User-Specified Constraints

- Example: Locating k delivery centers, each serving at least m valued customers and n ordinary ones
- Proposed approach
 - Find an initial “solution” by partitioning the data set into k groups and satisfying user-constraints
 - Iteratively refine the solution by micro-clustering relocation (e.g., moving δ μ -clusters from cluster C_i to C_j) and “deadlock” handling (break the microclusters when necessary)
 - Efficiency is improved by micro-clustering
- How to handle more complicated constraints?
 - E.g., having approximately same number of valued customers in each cluster?! — Can you solve it?

Chapter 7. Cluster Analysis

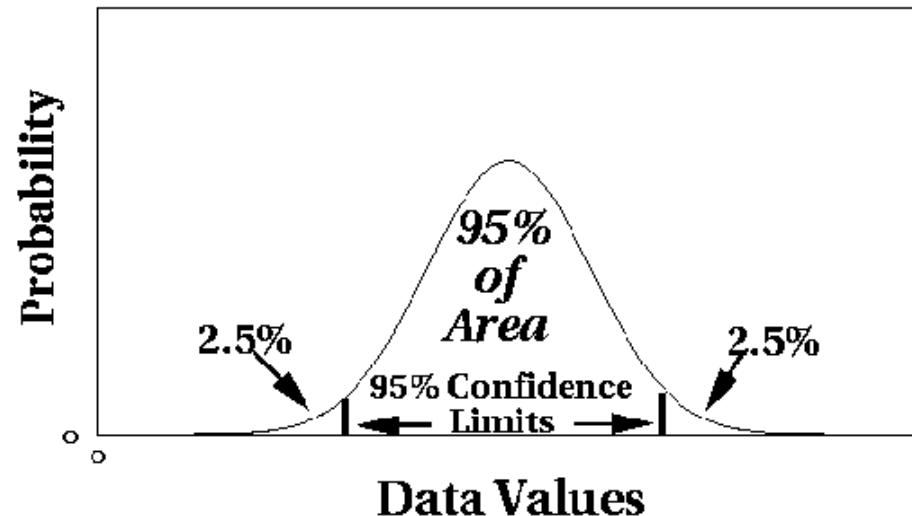
1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



What Is Outlier Discovery?

- What are outliers?
 - The set of objects are considerably dissimilar from the remainder of the data
 - Example: Sports: Michael Jordon, Wayne Gretzky, ...
- Problem: Define and find outliers in large data sets
- Applications:
 - Credit card fraud detection
 - Telecom fraud detection
 - Customer segmentation
 - Medical analysis

Outlier Discovery: Statistical Approaches



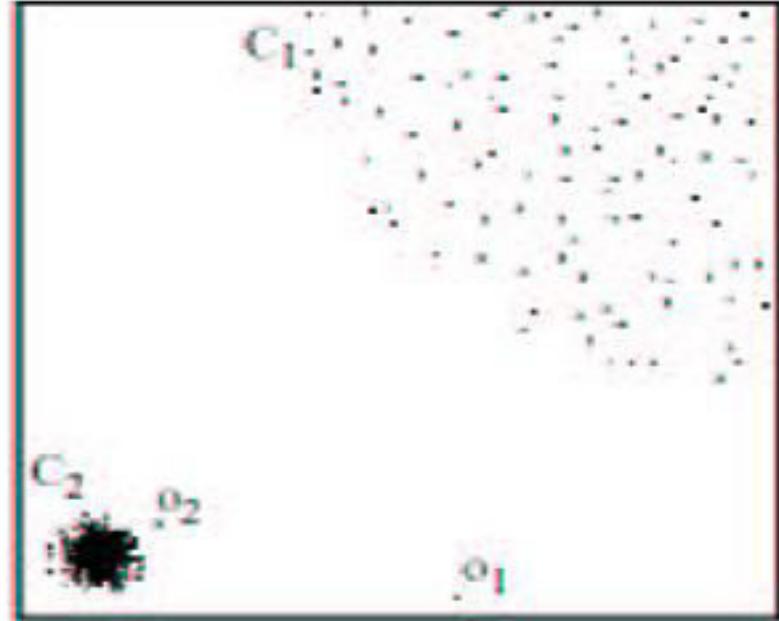
- Assume a model underlying distribution that generates data set (e.g. normal distribution)
- Use discordancy tests depending on
 - data distribution
 - distribution parameter (e.g., mean, variance)
 - number of expected outliers
- Drawbacks
 - most tests are for single attribute
 - In many cases, data distribution may not be known

Outlier Discovery: Distance-Based Approach

- Introduced to counter the main limitations imposed by statistical methods
 - We need multi-dimensional analysis without knowing data distribution
- Distance-based outlier: A $DB(p, D)$ -outlier is an object O in a dataset T such that at least a fraction p of the objects in T lies at a distance greater than D from O
- Algorithms for mining distance-based outliers
 - Index-based algorithm
 - Nested-loop algorithm
 - Cell-based algorithm

Density-Based Local Outlier Detection

- Distance-based outlier detection is based on global distance distribution
- It encounters difficulties to identify outliers if data is not uniformly distributed
- Ex. C_1 contains 400 loosely distributed points, C_2 has 100 tightly condensed points, 2 outlier points o_1, o_2
- Distance-based method cannot identify o_2 as an outlier
- Need the concept of local outlier



- Local outlier factor (LOF)
 - Assume outlier is not crisp
 - Each point has a LOF

Outlier Discovery: Deviation-Based Approach

- Identifies outliers by examining the main characteristics of objects in a group
- Objects that “deviate” from this description are considered outliers
- Sequential exception technique
 - simulates the way in which humans can distinguish unusual objects from among a series of supposedly like objects
- OLAP data cube technique
 - uses data cubes to identify regions of anomalies in large multidimensional data

Chapter 7. Cluster Analysis

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Outlier Analysis
12. Summary



Summary

- Cluster analysis groups objects based on their similarity and has wide applications
- Measure of similarity can be computed for various types of data
- Clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- Outlier detection and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches
- There are still lots of research issues on cluster analysis

Problems and Challenges

- Considerable progress has been made in scalable clustering methods
 - Partitioning: k-means, k-medoids, CLARANS
 - Hierarchical: BIRCH, ROCK, CHAMELEON
 - Density-based: DBSCAN, OPTICS, DenClue
 - Grid-based: STING, WaveCluster, CLIQUE
 - Model-based: EM, Cobweb, SOM
 - Frequent pattern-based: pCluster
 - Constraint-based: COD, constrained-clustering
- Current clustering techniques do not address all the requirements adequately, still an active area of research

References (1)

- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. SIGMOD'98
- M. R. Anderberg. Cluster Analysis for Applications. Academic Press, 1973.
- M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure, SIGMOD'99.
- P. Arabie, L. J. Hubert, and G. De Soete. Clustering and Classification. World Scientific, 1996
- Beil F., Ester M., Xu X.: "[Frequent Term-Based Text Clustering](#)", KDD'02
- M. M. Breunig, H.-P. Kriegel, R. Ng, J. Sander. LOF: Identifying Density-Based Local Outliers. SIGMOD 2000.
- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. KDD'96.
- M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. SSD'95.
- D. Fisher. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2:139-172, 1987.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. VLDB'98.

References (2)

- V. Ganti, J. Gehrke, R. Ramakrishnan. CACTUS Clustering Categorical Data Using Summaries. *KDD'99*.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. In Proc. VLDB'98.
- S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. *SIGMOD'98*.
- S. Guha, R. Rastogi, and K. Shim. [ROCK: A robust clustering algorithm for categorical attributes](#). In *ICDE'99*, pp. 512-521, Sydney, Australia, March 1999.
- A. Hinneburg, D.I A. Keim: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. *KDD'98*.
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Printice Hall, 1988.
- G. Karypis, E.-H. Han, and V. Kumar. [CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling](#). *COMPUTER*, 32(8): 68-75, 1999.
- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.
- E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. *VLDB'98*.
- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- P. Michaud. Clustering techniques. *Future Generation Computer systems*, 13, 1997.
- R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. *VLDB'94*.

References (3)

- *L. Parsons, E. Haque and H. Liu, [Subspace Clustering for High Dimensional Data: A Review](#) , SIGKDD Explorations, 6(1), June 2004*
- E. Schikuta. Grid clustering: An efficient hierarchical clustering method for very large data sets. Proc. 1996 Int. Conf. on Pattern Recognition,,.
- G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.
- A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng. [Constraint-Based Clustering in Large Databases](#), ICDT'01.
- A. K. H. Tung, J. Hou, and J. Han. [Spatial Clustering in the Presence of Obstacles](#) , ICDE'01
- H. Wang, W. Wang, J. Yang, and P.S. Yu. [Clustering by pattern similarity in large data sets](#), SIGMOD'02.
- W. Wang, Yang, R. Muntz, STING: A Statistical Information grid Approach to Spatial Data Mining, VLDB'97.
- T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH : an efficient data clustering method for very large databases. SIGMOD'96.

