

福昕PDF编辑器

• 永久 • 轻巧 • 自由

升级会员

批量购买



永久使用

无限制使用次数



极速轻巧

超低资源占用，告别卡顿慢



自由编辑

享受Word一样的编辑自由



扫一扫，关注公众号



CS 412 Intro. to Data Mining

Chapter 1. Introduction

Arindam Banerjee, Computer Science, UIUC, Fall 2021





Data and Information Systems (DAIS)

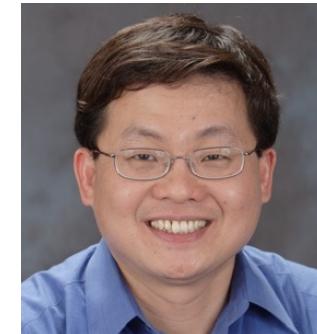
- Database Systems



Jiawei Han



Hanghang Tong



Kevin Chang

- Data Mining



Yongjoo Park

- Text Information Systems



Hari
Sundaram



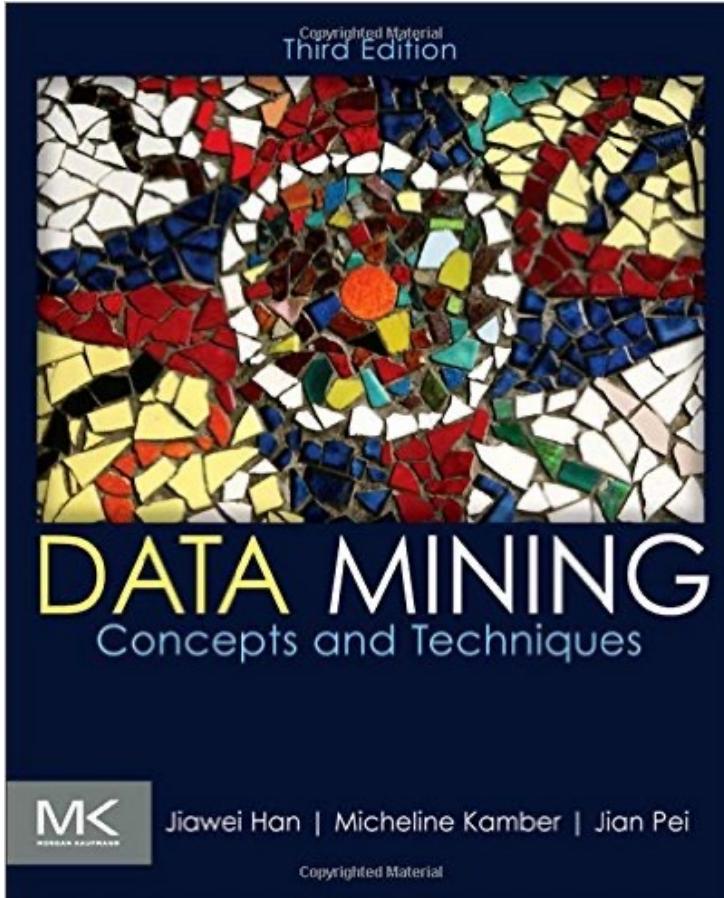
ChengXiang
Zhai

- Networks

Data and Information Systems (DAIS:) Course Structures at CS/UIUC

- Coverage: Database, data mining, text information systems, Web and bioinformatics
- Data mining
 - [Intro. to Data Mining \(CS 412\)](#)
 - [Data Mining Principles \(CS 512\)](#)
- Database Systems:
 - Database systems (CS 411)
 - Advanced Data Management (CS 511)
- Text information systems
 - Text Information System (CS 410)
 - Advanced Information Retrieval (CS 510)

CS 412. Course Page & Class Schedule



- Textbook
 - Jiawei Han, Micheline Kamber, Jian Pei, Hanghang Tong, *Data Mining: Concepts and Techniques* (4th ed), 2021
- Class Homepage: Fall, 2021
<https://canvas.illinois.edu/courses/13790>
- **Class Schedule: Tue, Thu 09:30 am-10:45 pm**
 - **3039 Campus Instructional Facility**
- Office hours: Tue, Thu 6:00-7:00 pm @zoom

CS 412. Fall 2021. Teaching Assistants



Yikun Ban



Dongqi Fu



Zhe Xu



Hang Zhang

- ❑ TA office hours:
 - ❑ Yikun: Tue, Thu 4 – 5 pm
 - ❑ Dongqi: Fri, 4 – 6 pm
 - ❑ Zhe: Mon, Wed 4 – 5 pm
 - ❑ Hang: Mon, Thu 11 am – 12 noon
- ❑ Wait list -- No wait list at this time, keep attending the class

CS 412. Course Work and Grading

- ❑ Assignments, Programming Assignments, and Exams
 - ❑ Written Assignments: 30% (three homework assignments expected)
 - ❑ Programming assignments: 30% (two programming assignments expected)
 - ❑ Midterm exam (take home): 20%
 - ❑ Final exam (take home): 20%
- ❑ For students taking 4th credit
 - ❑ For students registering 4 credits: 25%, overall scores to be scaled proportionally
 - ❑ Group project: 2-3 members
- ❑ Slides, Chapters, homeworks/exams, your scores & grades: **Canvas**
- ❑ Need help and/or discussions?
 - ❑ Canvas discussions
 - ❑ Slack

Key Dates

- Assignments
 - A1: Tue, Sept 7 out, Thu, **Sept 23** due (16 days)
 - A2: Thu, Sept 23 out, Mon, **Oct 11** due (18 days)
 - A3 (programming): Mon, Oct 11 out, Mon, **Nov 08** due (28 days)
 - A4 (programming): Mon, Nov 08 out, Fri, **Dec 03** due (25 days)
 - A5: Mon, Nov 15 out, Mon, **Dec 06** due (21 days)
- Take-Home Exams
 - Mid-term: Thu, **Oct 14**, 6 pm, 24 hours
 - Final: Fri, **Dec 10**, 6 pm, 24 hours

Please Mark Your Calendars

Start Early

4-Credit Projects

- ❑ Research project: Join a competition or propose your own project
 - ❑ Example: <https://www.kaggle.com/competitions>
- ❑ Important Dates
 - ❑ Project teams due: Mon, Sept 20
 - ❑ Project proposal due: Mon, Oct 04, 1 page + references
 - ❑ Mid-term report due: Wed, Nov 03, 4 pages + references
 - ❑ Final report due: Wed, Dec 08, 8 pages + references
- ❑ Reports need to be in 11-pt, single column, 1-inch margins

CS 412. Grades

- Following cutoffs represent what will likely be used to generate letter grades:

A+ $\geq 98\%$

A $\geq 94\% \text{ & } < 98\%$

A- $\geq 90\% \text{ & } < 94\%$

B+ $\geq 85\% \text{ & } < 90\%$

B $\geq 80\% \text{ & } < 85\%$

B- $\geq 77\% \text{ & } < 80\%$

C+ $\geq 74\% \text{ & } < 77\%$

C $\geq 70\% \text{ & } < 74\%$

C- $\geq 67\% \text{ & } < 70\%$

D $\geq 60\% \text{ & } < 67\%$

F $< 60\%$

- The above cutoffs are tentative and may be adjusted slightly; if there is any adjustment to the above cutoffs, we will NOT curve down your letter grades
- However, there will be no general curve-fitting in assigning the final grades

Chapter 1. Introduction

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

Why Data Mining?

- ❑ The Explosive Growth of Data: from terabytes to petabytes
 - ❑ Data collection and data availability
 - ❑ Automated data collection tools, database systems, Web, computerized society
 - ❑ Major sources of abundant data
 - ❑ Business: Web, e-commerce, transactions, stocks, transportation, ...
 - ❑ Science: Remote sensing, bioinformatics, climate science, agriculture, water, ...
 - ❑ Society and everyone: news, events, social networks, ...
- ❑ We are drowning in data, but starving for knowledge!
- ❑ “Necessity is the mother of invention”—Data mining—Automated analysis of massive data sets

Chapter 1. Introduction

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

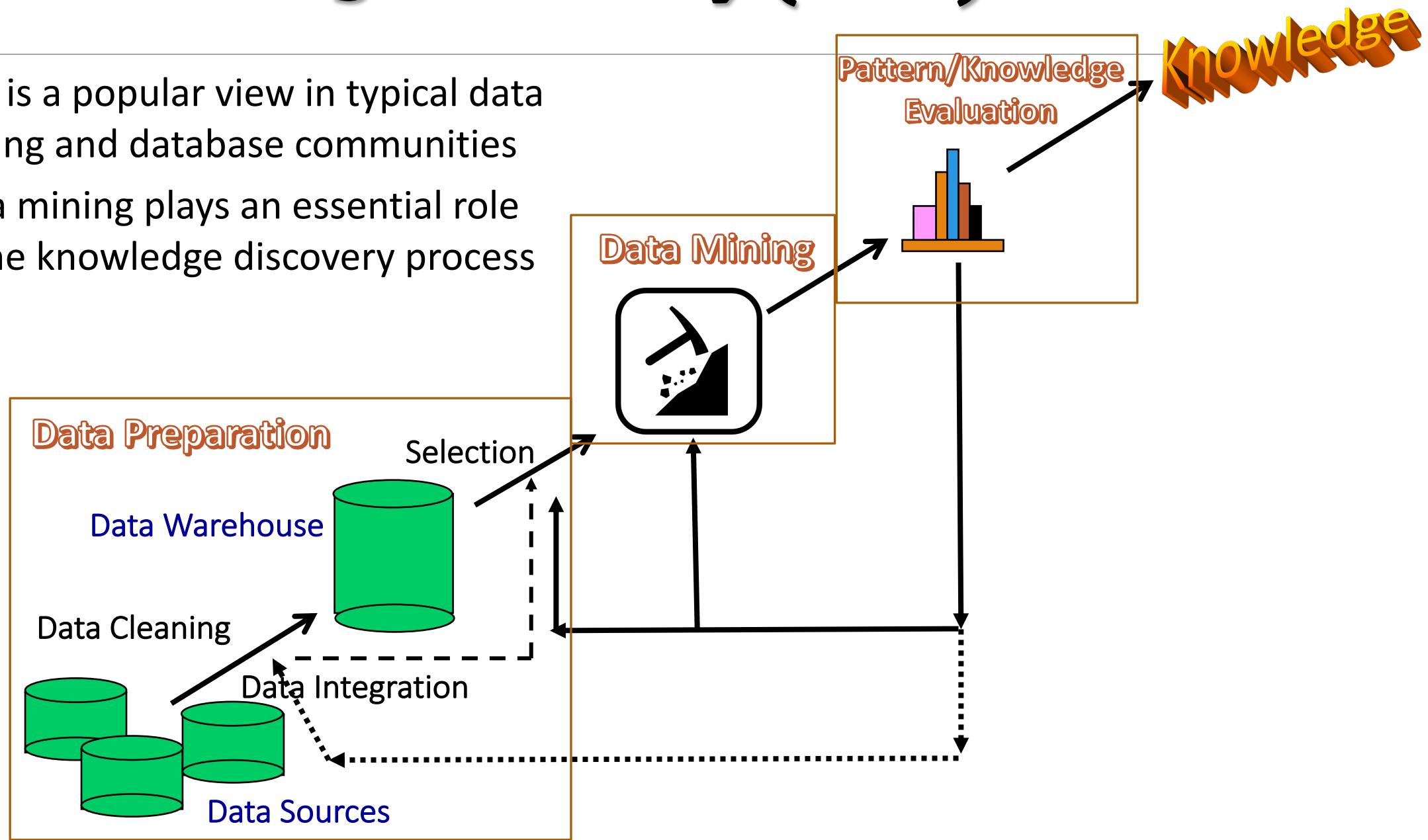
What Is Data Mining?

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

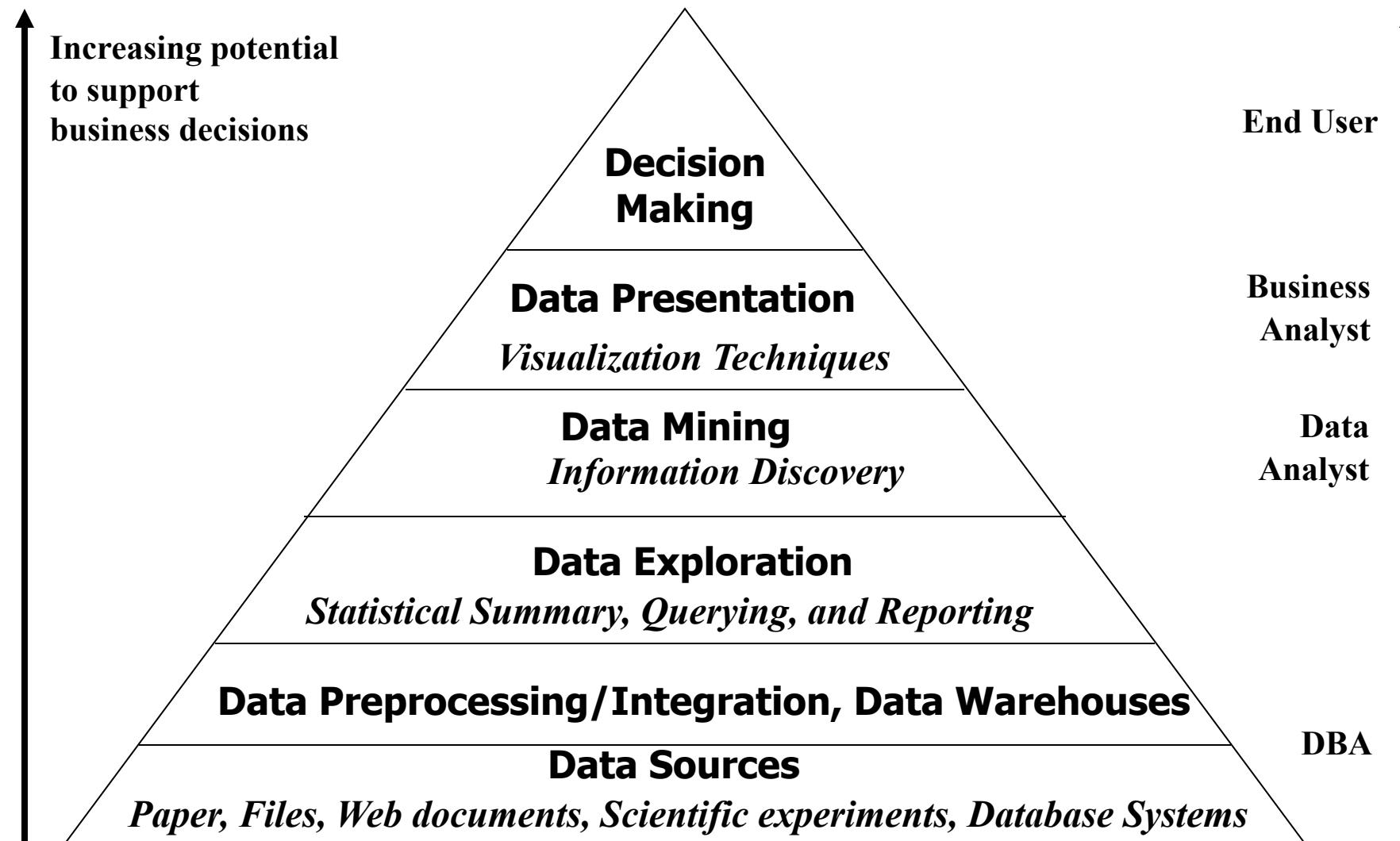
- This is a popular view in typical data mining and database communities
- Data mining plays an essential role in the knowledge discovery process



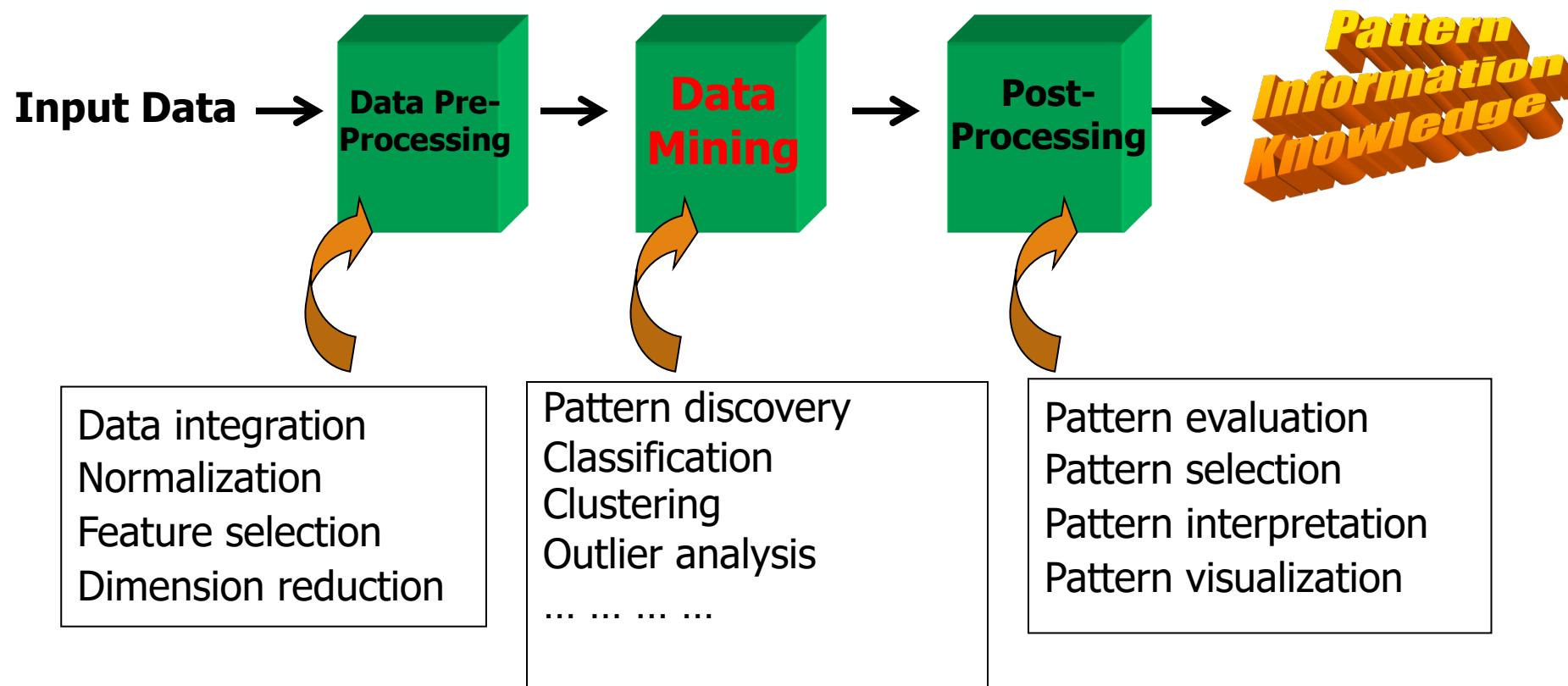
Example: A Web Mining Framework

- Web mining usually involves
 - Data cleaning
 - Data integration from multiple sources
 - Warehousing the data
 - Data cube construction
 - Data selection for data mining
 - Data mining
 - Presentation of the mining results
 - Patterns and knowledge to be used or stored into knowledge-base

Data Mining in Business Intelligence



KDD Process: A View from ML and Statistics



- This is a view from typical machine learning and statistics communities

Data Mining vs. Data Exploration

- Which view do you prefer?
 - KDD vs. ML/Stat. vs. Business Intelligence
 - Depending on the data, applications, and your focus

- Data Mining vs. Data Exploration
 - Business intelligence view
 - Warehouse, data cube, reporting but not much mining
 - Business objects vs. data mining tools
 - Supply chain example: mining vs. OLAP vs. presentation tools
 - Data presentation vs. data exploration

Chapter 1. Introduction

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



Multi-Dimensional View of Data Mining

Data to be mined

- Database data (extended-relational, object-oriented, heterogeneous), data warehouse, transactional data, stream, spatiotemporal, time-series, sequence, text and web, multi-media, graphs & social and information networks

Knowledge to be mined (or: Data mining functions)

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

Techniques utilized

- Data-intensive, data warehouse (OLAP), machine learning, statistics, pattern recognition, visualization, high-performance, etc.

Applications adapted

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

Chapter 1. Introduction

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

Data Mining: On What Kinds of Data?

- Database-oriented data sets and applications
 - Relational database, data warehouse, transactional database
 - Object-relational databases, Heterogeneous databases and legacy databases
- 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 information networks
 - Spatial data and spatiotemporal data
 - Multimedia database
 - Text databases
 - The World-Wide Web

Chapter 1. Introduction

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



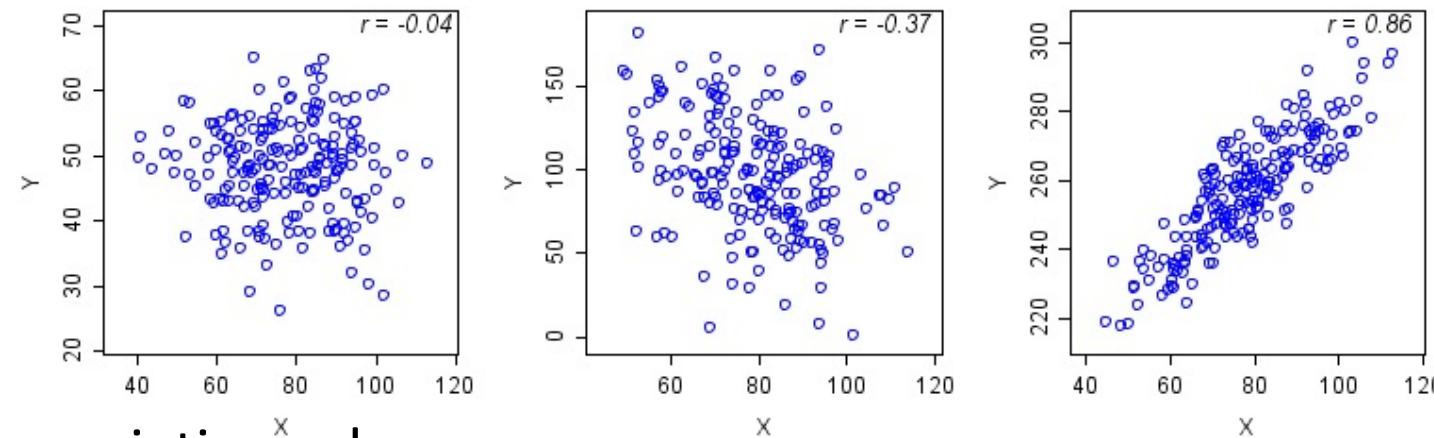
Data Mining Functions: (1) Generalization

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



Data Mining Functions: (2) Pattern Discovery

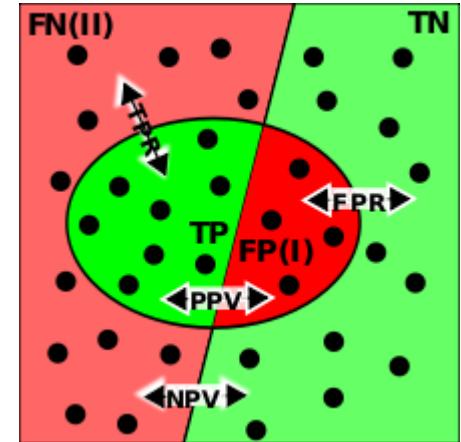
- Frequent patterns (or frequent itemsets)
 - What items are frequently purchased together in your Walmart?
- Association and Correlation Analysis



- A typical association rule
 - Diaper \rightarrow Beer [0.5%, 75%] (support, confidence)
 - Are strongly associated items also strongly correlated?
- How to mine such patterns and rules efficiently in large datasets?
- How to use such patterns for classification, clustering, and other applications?

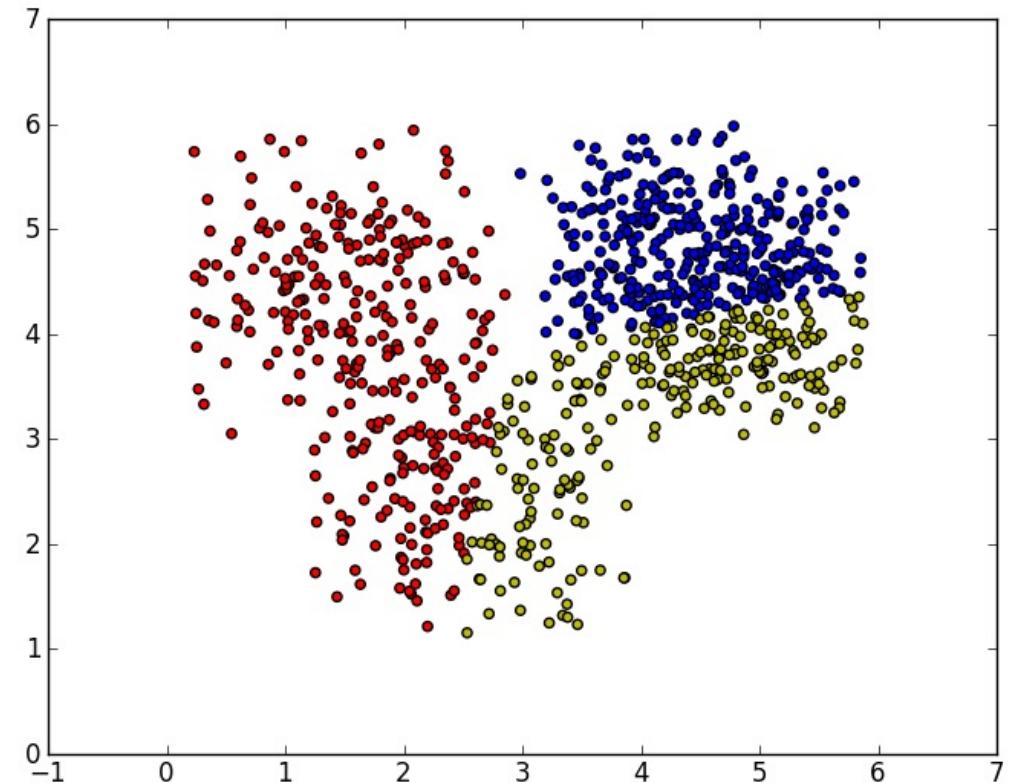
Data Mining Functions: (3) Classification

- ❑ Classification and label prediction
 - ❑ Construct models (functions) based on some training examples
 - ❑ Describe and distinguish classes or concepts for future prediction
 - ❑ Ex. 1. Classify countries based on (climate)
 - ❑ Ex. 2. Classify cars based on (gas mileage)
 - ❑ Predict some unknown class labels
- ❑ Typical methods
 - ❑ Decision trees, naïve Bayesian classification, support vector machines, neural networks, rule-based classification, pattern-based classification, logistic regression, ...
- ❑ Typical applications:
 - ❑ Credit card fraud detection, direct marketing, classifying stars, diseases, web-pages, ...



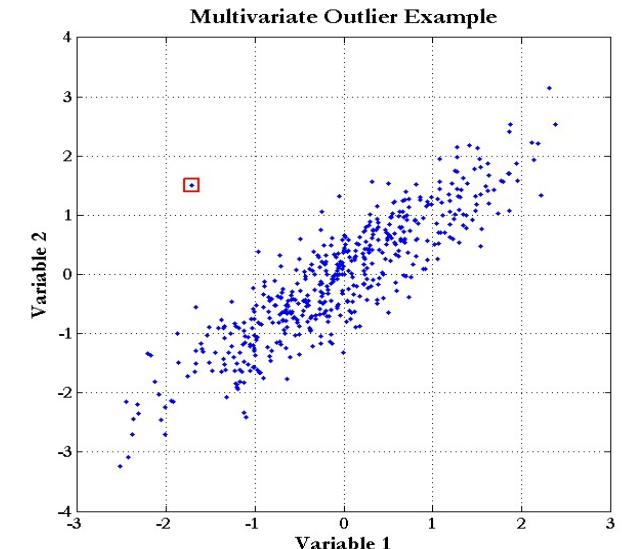
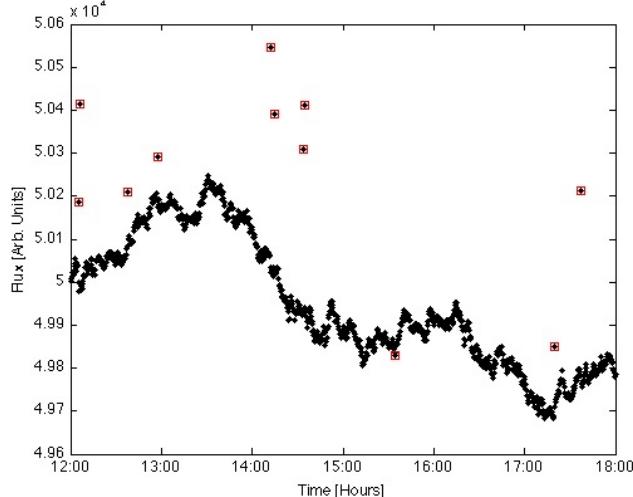
Data Mining Functions: (4) Cluster Analysis

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



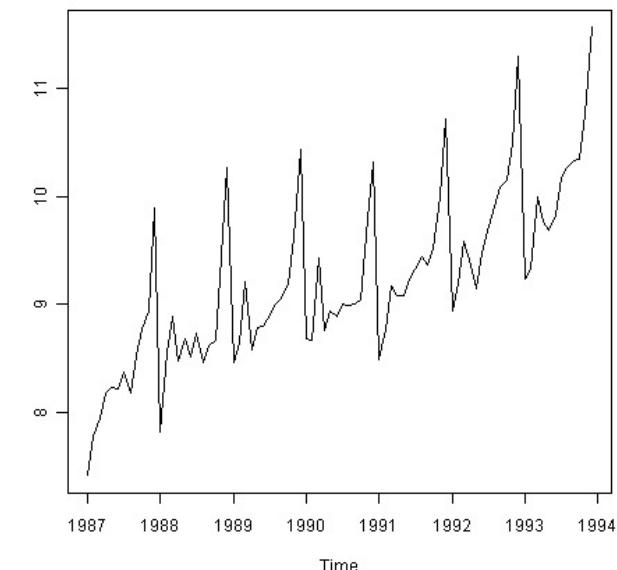
Data Mining Functions: (5) Outlier Analysis

- Outlier analysis
 - Outlier: A data object that does not comply with the general behavior of the data
 - Noise or exception?—One person's garbage could be another person's treasure
 - Methods: by product of clustering or regression analysis, ...
 - Useful in fraud detection, rare events analysis



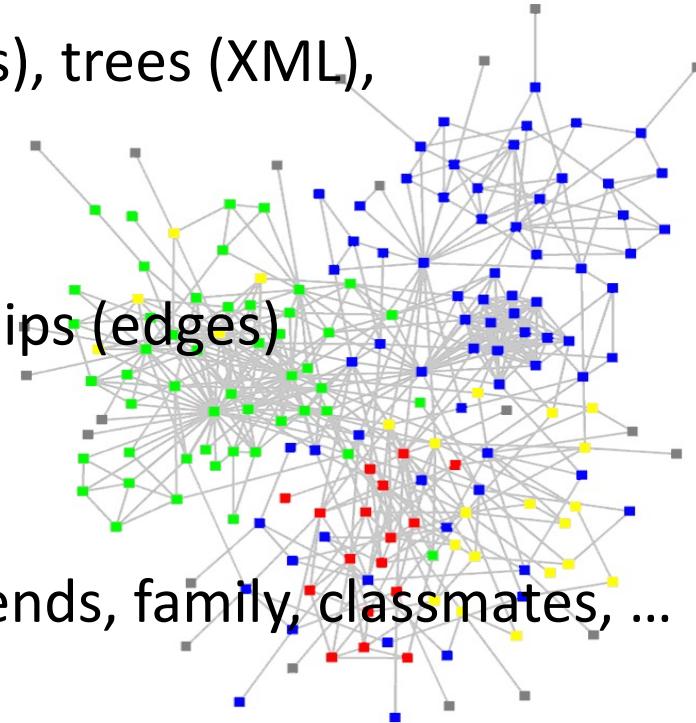
Data Mining Functions: (6) Time and Ordering: Sequential Pattern, Trend and Evolution Analysis

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



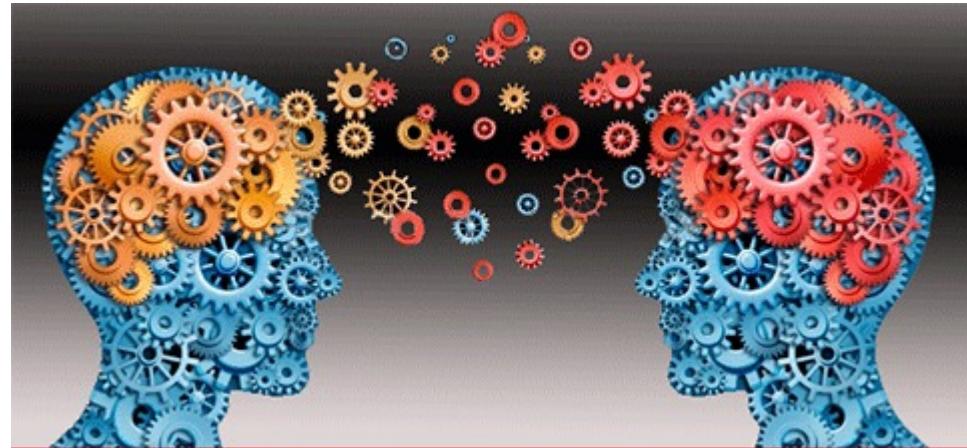
Data Mining Functions: (7) Structure and Network Analysis

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



Evaluation of Knowledge

- ❑ Are all mined knowledge interesting?
 - ❑ One can mine tremendous amount of “patterns”
 - ❑ Some may fit only certain dimension space (time, location, ...)
 - ❑ Some may not be representative, may be transient, ...
- ❑ Evaluation of mined knowledge → directly mine only interesting knowledge?
 - ❑ Descriptive vs. predictive
 - ❑ Coverage
 - ❑ Typicality vs. novelty
 - ❑ Accuracy
 - ❑ Timeliness
 - ❑ ...

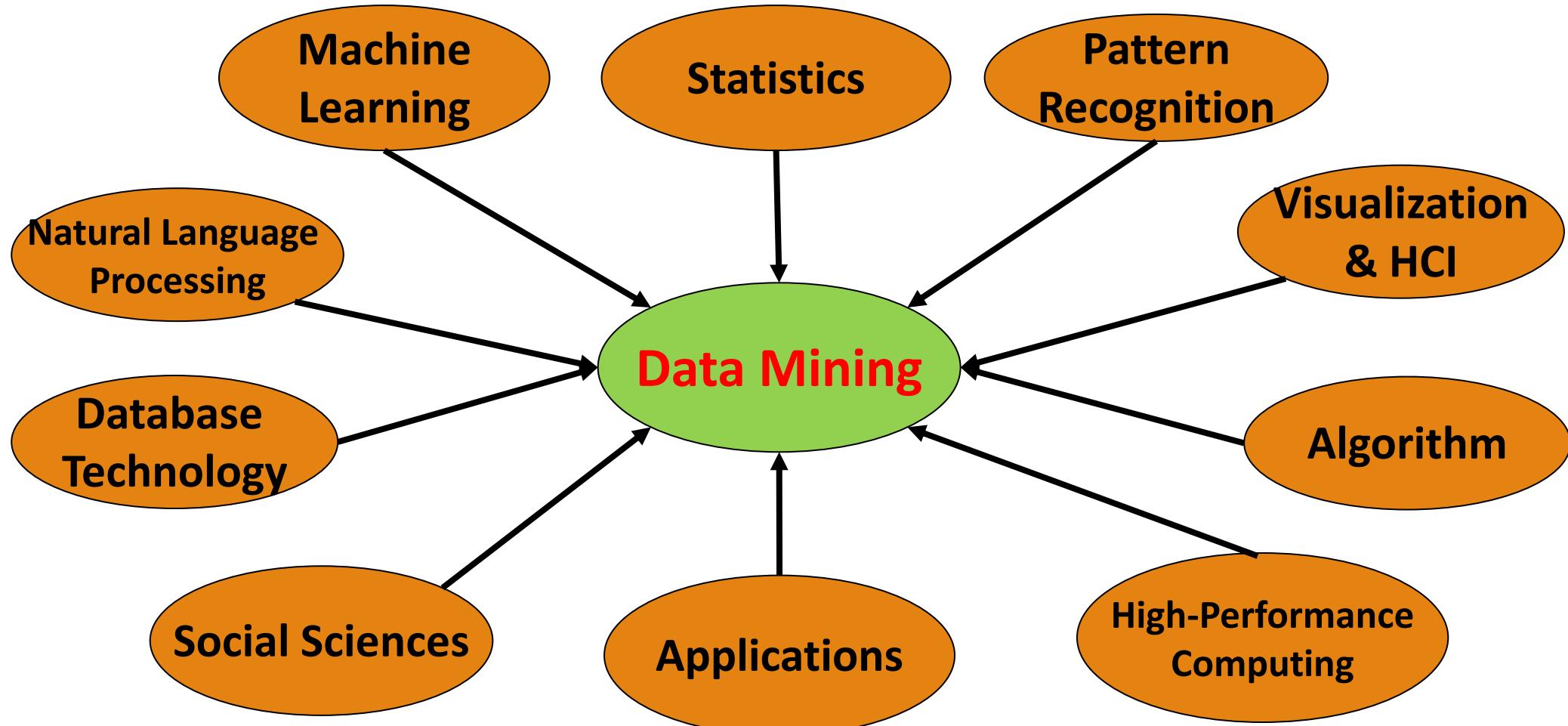


Chapter 1. Introduction

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



Data Mining: Confluence of Multiple Disciplines



Why Confluence of Multiple Disciplines?

- Tremendous amount of data
 - Algorithms must be scalable to handle big 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 and information networks
 - Spatial, spatiotemporal, multimedia, text and Web data
 - Software programs, scientific simulations
- New and sophisticated applications

Chapter 1. Introduction

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

Applications of Data Mining

- Web page analysis: classification, clustering, ranking
- Collaborative analysis & recommender systems
- Basket data analysis to targeted marketing
- Biological and medical data analysis
- Data mining and software engineering
- Data mining and text analysis
- Data mining and social and information network analysis
- Built-in (invisible data mining) functions in Google, MS, Yahoo!, Linked, Facebook, ...
- Major dedicated data mining systems/tools
- SAS, MS SQL-Server Analysis Manager, Oracle Data Mining Tools



Chapter 1. Introduction

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

Major Challenges in Data Mining (1)

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

Major Challenges in Data Mining (2)

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

Chapter 1. Introduction

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



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
 - SDM (2001), (IEEE) ICDM (2001), WSDM (2008), PKDD (1997), PAKDD (1997), etc.
- ACM Transactions on KDD (2007), IEEE Transactions on KDE (1989)

Conferences and Journals on Data Mining

- ❑ KDD Conferences
 - ❑ ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases and Data Mining ([KDD](#))
 - ❑ SIAM Data Mining Conf. ([SDM](#))
 - ❑ (IEEE) Int. Conf. on Data Mining ([ICDM](#))
 - ❑ European Conf. on Machine Learning and Principles and practices of Knowledge Discovery and Data Mining ([ECML-PKDD](#))
 - ❑ Pacific-Asia Conf. on Knowledge Discovery and Data Mining ([PAKDD](#))
 - ❑ Int. Conf. on Web Search and Data Mining ([WSDM](#))
- Other related conferences
 - DB conferences: ACM SIGMOD, VLDB, ICDE, EDBT, ICDT, ...
 - Web and IR conferences: WWW, SIGIR, WSDM
 - ML conferences: ICML, NeurIPS, AISTATS, COLT, UAI, ...
 - Vision/Language conferences: CVPR, ICCV, ACL, EMNLP, ...
 - Journals
 - Data Mining and Knowledge Discovery (DAMI or DMKD)
 - IEEE Trans. On Knowledge and Data Eng. (TKDE)
 - KDD Explorations
 - ACM Trans. on KDD

References? DBLP, Google, arXiv, CiteSeer

- Data mining and KDD (SIGKDD)
 - Conferences: ACM-SIGKDD, IEEE-ICDM, SIAM-DM, PKDD, PAKDD, etc.
 - Journal: Data Mining and Knowledge Discovery, KDD Explorations, ACM TKDD
- Database systems (SIGMOD)
 - 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: NeurIPS, ICML, ICLR, AAAI, AISTATS, IJCAI, COLT, CVPR, ICCV, UAI, 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.

Chapter 1. Introduction

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



Summary

- Data mining: Discovering interesting patterns and knowledge from massive amount of data
- A natural evolution of science and information 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 data
- Data mining functionalities: characterization, discrimination, association, classification, clustering, trend and outlier analysis, etc.
- Data mining technologies and applications
- Major issues in data mining

Recommended Reference Books

- Charu C. Aggarwal, Data Mining: The Textbook, Springer, 2015
- E. Alpaydin. Introduction to Machine Learning, 2nd ed., MIT Press, 2011
- R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2ed., Wiley-Interscience, 2000
- U. Fayyad, G. Grinstein, and A. Wierse, Information Visualization in Data Mining and Knowledge Discovery, Morgan Kaufmann, 2001
- J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd ed. , 2011
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., Springer, 2009
- T. M. Mitchell, Machine Learning, McGraw Hill, 1997
- P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, Wiley, 2005 (2nd ed. 2016)
- I. H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, 2nd ed. 2005
- Mohammed J. Zaki and Wagner Meira Jr., Data Mining and Analysis: Fundamental Concepts and Algorithms, Cambridge University Press, 2014





CS 412 Intro. to Data Mining

Chapter 2. Data and Measurements

Arindam Banerjee, Computer Science, UIUC, Fall 2021



Chapter 2. Getting to Know Your Data

- Data Objects and Attribute Types
- Basic Statistical Descriptions of Data
- Data Visualization
- Measuring Data Similarity and Correlation
- Summary



Types of Data Sets: (1) Record Data

- Relational records
 - Relational tables, highly structured
- Data matrix, e.g., numerical matrix, crosstabs

	China	England	France	Japan	USA	Total
Active Outdoors Crochet Glove		12.00	4.00	1.00	240.00	257.00
Active Outdoors Lycra Glove		10.00	6.00		323.00	339.00
InFlux Crochet Glove	3.00	6.00	8.00		132.00	149.00
InFlux Lycra Glove		2.00			143.00	145.00
Triumph Pro Helmet	3.00	1.00	7.00		333.00	344.00
Triumph Vertigo Helmet		3.00	22.00		474.00	499.00
Xtreme Adult Helmet	8.00	8.00	7.00	2.00	251.00	276.00
Xtreme Youth Helmet		1.00			76.00	77.00
Total	14.00	43.00	54.00	3.00	1,972.00	2,086.00

- Transaction data

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

- Document data: Term-frequency vector (matrix) of text documents

Person:

Pers_ID	Surname	First_Name	City
0	Miller	Paul	London
1	Ortega	Alvaro	Valencia
2	Huber	Urs	Zurich
3	Blanc	Gaston	Paris
4	Bertolini	Fabrizio	Rom

no relation

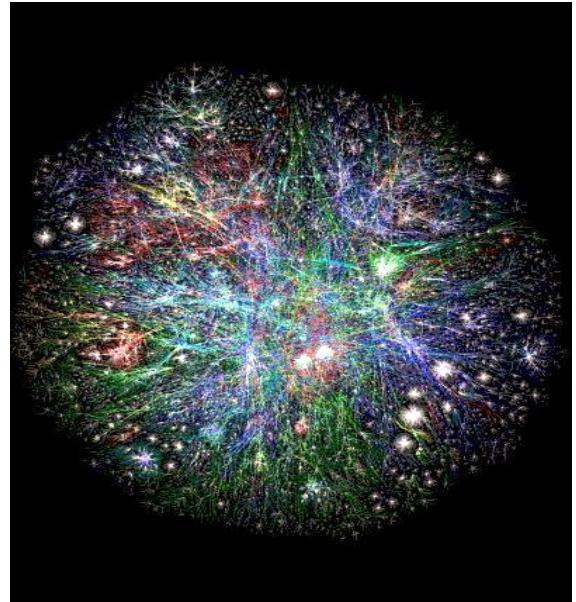
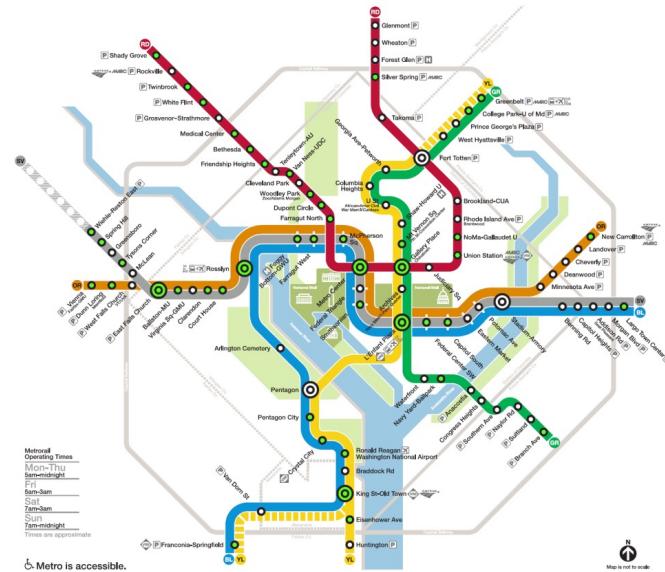
Car:

Car_ID	Model	Year	Value	Pers_ID
101	Bentley	1973	100000	0
102	Rolls Royce	1965	330000	0
103	Peugeot	1993	500	3
104	Ferrari	2005	150000	4
105	Renault	1998	2000	3
106	Renault	2001	7000	3
107	Smart	1999	2000	2

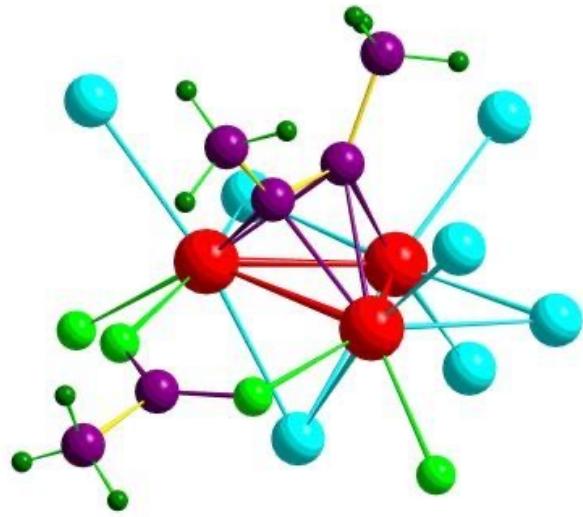
team	coach	y	pla	ball	score	game	n	wi	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2	
Document 2	0	7	0	2	1	0	0	3	0	0	
Document 3	0	1	0	0	1	2	2	0	3	0	

Types of Data Sets: (2) Graphs and Networks

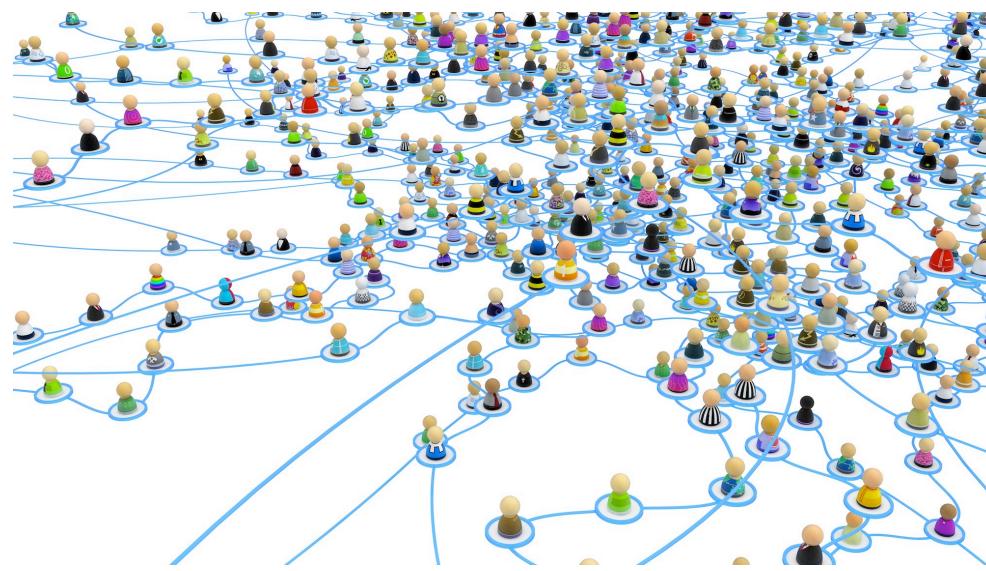
Transportation network



World Wide Web



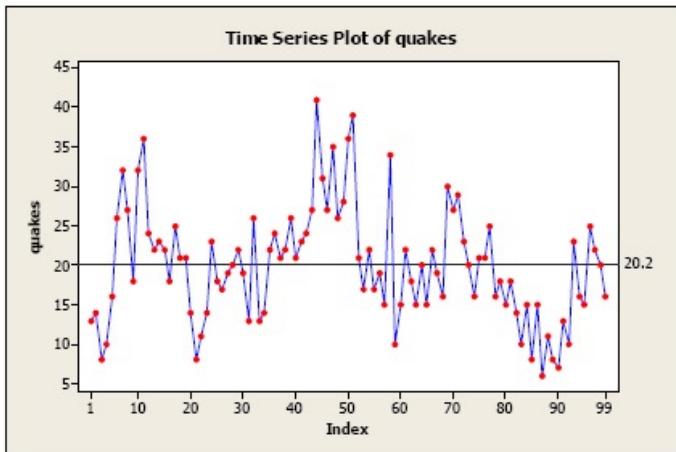
Molecular Structures



Social or information networks

Types of Data Sets: (3) Ordered Data

- Video data: sequence of images



- Temporal data: time-series



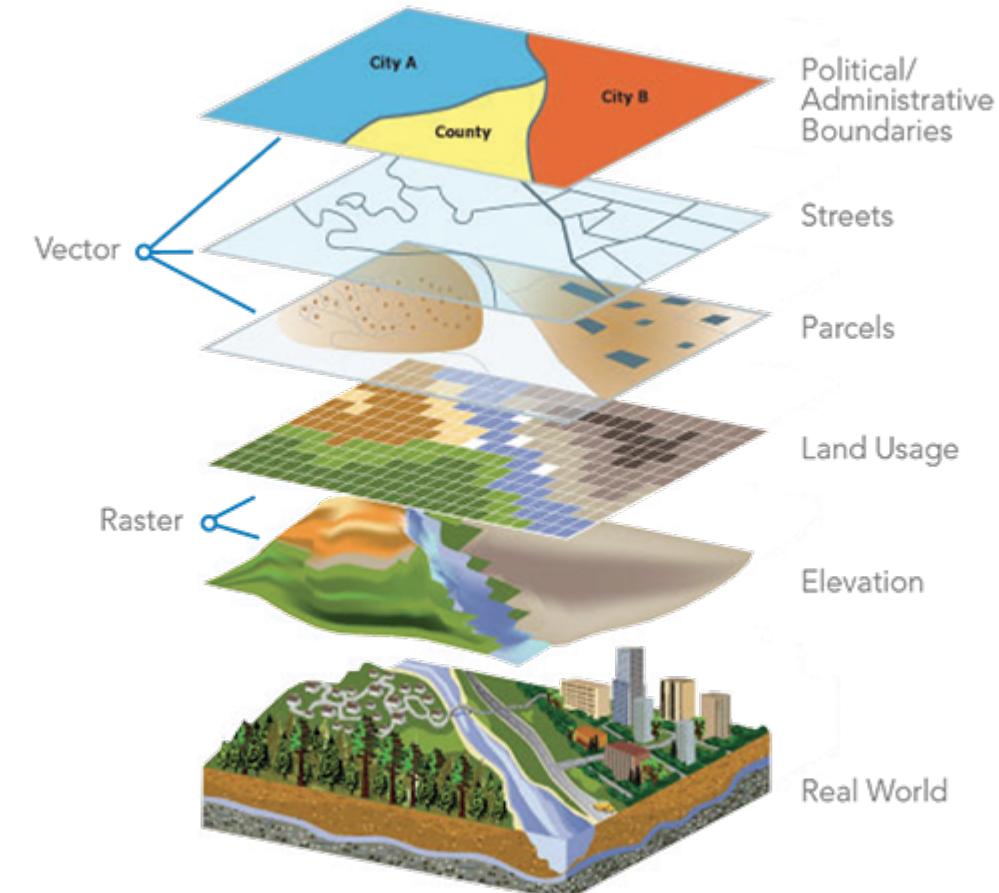
- Sequential Data: transaction sequences

- Genetic sequence data

Human	GTTTGAGG	-	ATGTTCAACAAATGCTCCTTCATTCCCTATTTACAGACCTGCCGCA
Chimpanzee	GTTTGAGG	-	-ATGTTCAATAATGCTGCTTCACTCCCTATTTACAGACCTGCCGCA
Macaque	GTTTGAGG	-	-ATGCTCAATAATGCTCCTTCATTCCCTCATTACAACCTGCCGCA
Human	GACAATTCTGCTAGCAGCCTTGTGCTATTATCTGTTTCTAAACCTTAGTAATTGAGTGT		
Chimpanzee	GACAATTCTGCTAGCAGCCTTGTGCTATTATCTGTTTCTAAACCTTAGTAATTGAGTGT		
Macaque	GACAATTCTGCTAGCAGCCTTGTGCTATTATCTGTTTCTAAACCTTAGTAATTGAGTGT		
Human	GATCTGGAGACTAA	-	CCTCTGAAATAAAATAAGCTGATTATTTATTTATTTCTCAAAACAA
Chimpanzee	GATCTGGAGACTAA	-	CCTCTGAAATAAAATAAGCTGATTATTTATTTATTTCTCAAAACAA
Macaque	TATCTGGAGACTAA	-	ACTCTGAAATAAAATAAGCTGATTATTTATTTATTTCTCAAAACAA
Human	CAGAACACGATTTAGCAAATTACTCTTAAGATAATTATTTACATTTCTATATTCTCTA		
Chimpanzee	CAGAACACGATTTAGCAAATTACTCTTAAGATAACTATTTACATTTCTATATTCTCTA		
Macaque	CAGAACATGATTTAGCAAATTACCTCTTAAGATAATTATTTGCACCTTCTATATTCTCTA		
Human	CCCTGAGTTGATGTTGAGCAATATGTCACCTTCATAAAGCCAGGTATACAC	-	-TTATG
Chimpanzee	CCCTGAGTTGATGTTGAGCCGATATGTCACCTTCATAAAGCCAGGTATACAC	-	-TTATG
Macaque	CCCTGAGTTGATGTTGAGCAATATGTCACCTCCACAAAGCCAGGTATATACATTACG		
Human	GACAGGTAAGTAAAAACATATTATTTATCTACGTTTGTCCAAGAATTAAATTTC	H I Y S T F L S K	
Chimpanzee	GACAGGTAAGTAAAAACATATTATTTATCTACGTTTGTCCAAGAATTAAATTTC		
Macaque	GACAGGTAAGTAAAAACATATTATTTATCTACGTTTGTCCAAGAATTAAATTTC		
Human	AACTGTTGCGCGTGTGGTAA	-	-TGTAAAACAAACTCAGTACAA
Chimpanzee	AACTGTTGCGCGTGTGGTAA	-	-TGTAAAACAAACTCAGTACAA
Macaque	AACTGTTGCGCGTGTGGTAA	-	-CTGTAAAACAAACTCAGTACG

Types of Data Sets: (4) Spatial, image and multimedia Data

- Spatial data: maps



- Image data:

- Video data:

Important Characteristics of Structured Data

- Dimensionality
 - Curse of dimensionality
- Sparsity
 - Only presence counts
- Resolution
 - Patterns depend on the scale
- Distribution
 - Centrality and dispersion

Data Objects

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

Attributes

- **Attribute (or dimensions, features, variables)**
 - A data field, representing a characteristic or feature of a data object.
 - *E.g., customer_ID, name, address*
- Types:
 - Nominal (e.g., red, blue)
 - Binary (e.g., {true, false})
 - Ordinal (e.g., {freshman, sophomore, junior, senior})
 - Numeric: quantitative
 - Interval-scaled: No true zero, can compute differences, means, etc.
 - Examples: Temp in °C or °F, calendar year
 - Ratio-scaled: True zero, ratio scaled, e.g., 10 is twice as much as 5
 - Examples: Temp in °K, years of experience, number of words

Attribute Types

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

Numeric Attribute Types

- Quantity (integer or real-valued)
- **Interval**
 - Measured on a scale of **equal-sized units**
 - Values have order
 - E.g., *temperature in C° or F°, calendar dates*
 - No true zero-point
- **Ratio**
 - Inherent **zero-point**
 - We can speak of values as being an order of magnitude larger than the unit of measurement (10 K° is twice as high as 5 K°).
 - e.g., *temperature in Kelvin, length, counts, monetary quantities*

Discrete vs. Continuous Attributes

□ Discrete Attribute

- Has only a finite or countably infinite set of values
 - E.g., zip codes, profession, or the set of words in a collection of documents
- Sometimes, represented as integer variables
- Note: Binary attributes are a special case of discrete attributes

□ Continuous Attribute

- Has real numbers as attribute values
 - E.g., temperature, height, or weight
- Practically, real values can only be measured and represented using a finite number of digits
- Continuous attributes are typically represented as floating-point variables

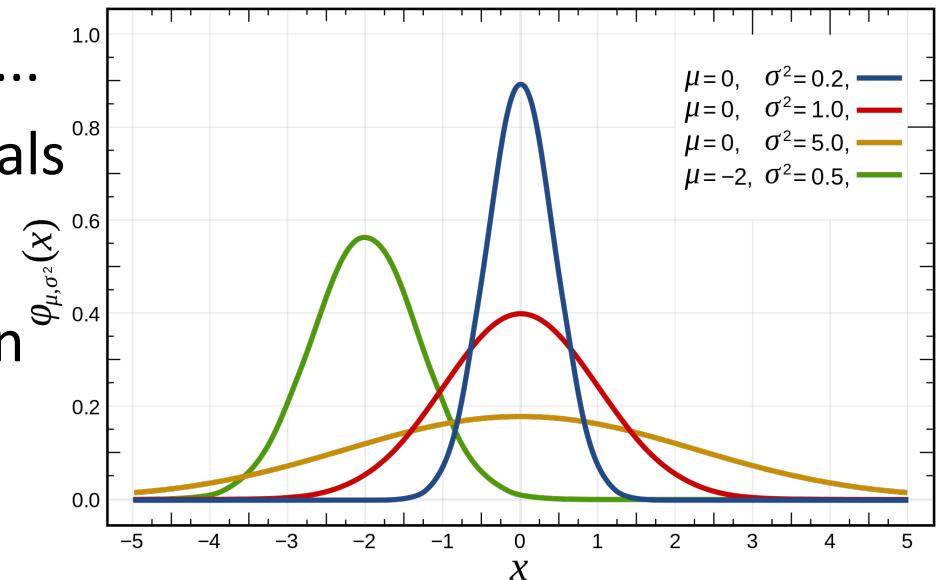
Chapter 2. Getting to Know Your Data

- Data Objects and Attribute Types
- Basic Statistical Descriptions of Data
- Data Visualization
- Measuring Data Similarity and Dissimilarity
- Summary



Basic Statistical Descriptions of Data

- Motivation
 - To better understand the data: central tendency, variation and spread
- Data dispersion characteristics
 - Median, max, min, quantiles, outliers, variance, ...
- 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: (1) Mean

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

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

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

sample mean

$$\mu = \frac{1}{N} \sum_{j=1}^N x_j$$

population mean

- Weighted arithmetic mean:

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

- Trimmed mean:

- Chopping extreme values (e.g., Olympics gymnastics score computation)

Measuring the Central Tendency: (2) Median

□ Median:

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

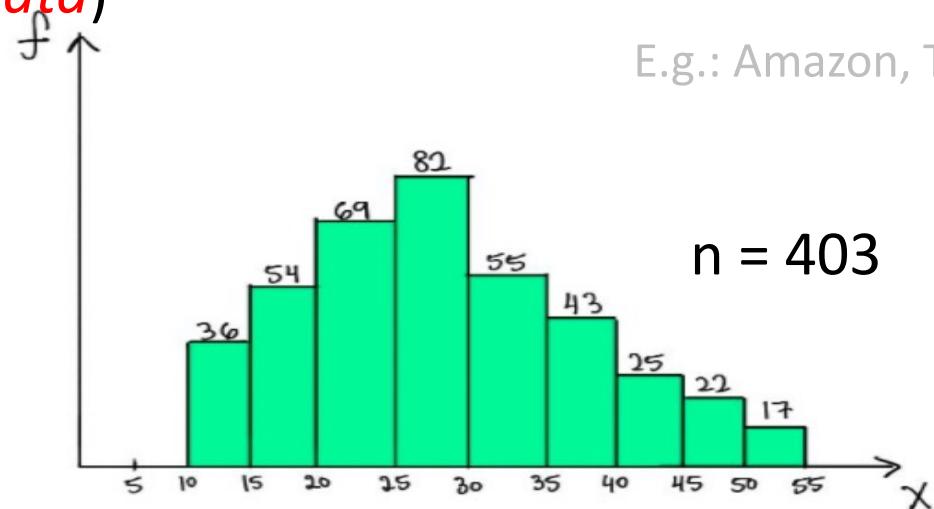
Bins: 1,2,...,M

Frequencies: f_1, f_2, \dots, f_M

Cumulative frequencies $F_m = \sum_{l=1}^m f_l$

Median falls in the m^{th} bin, $F_{m-1} \leq n/2 \leq F_m$

Interval $[L_m, L_{m+1}]$, e.g., [25,30]



Approximate median

Sum before the median interval

Interval width

$$\text{median} \approx L_m + \left(\frac{n/2 - F_{m-1}}{f_m} \right) \times (L_{m+1} - L_m)$$

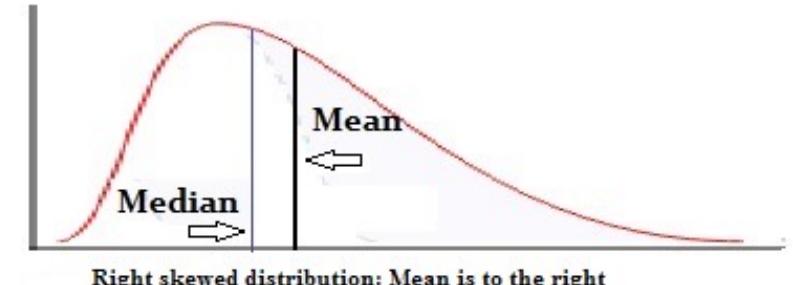
Low interval limit

Measuring the Central Tendency: (3) Mode

- Mode: Value that occurs most frequently in the data

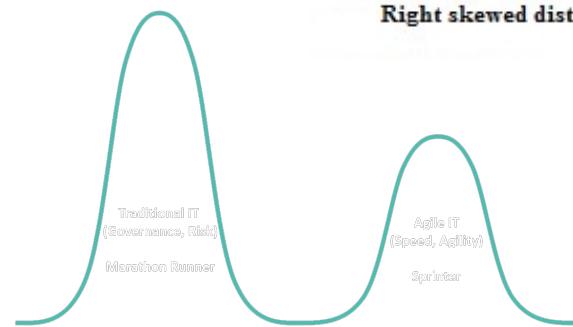
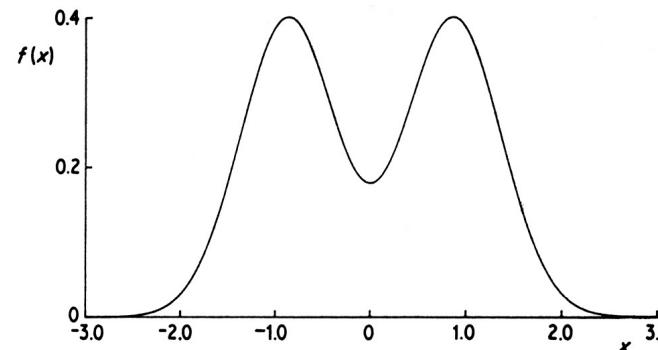
- Unimodal

- Empirical formula: $mean - mode = 3 \times (mean - median)$

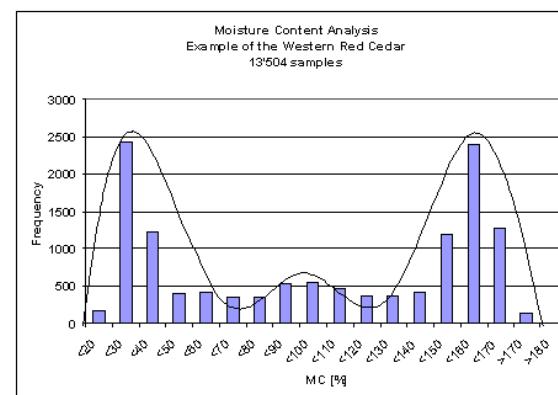


- Multi-modal

- Bimodal

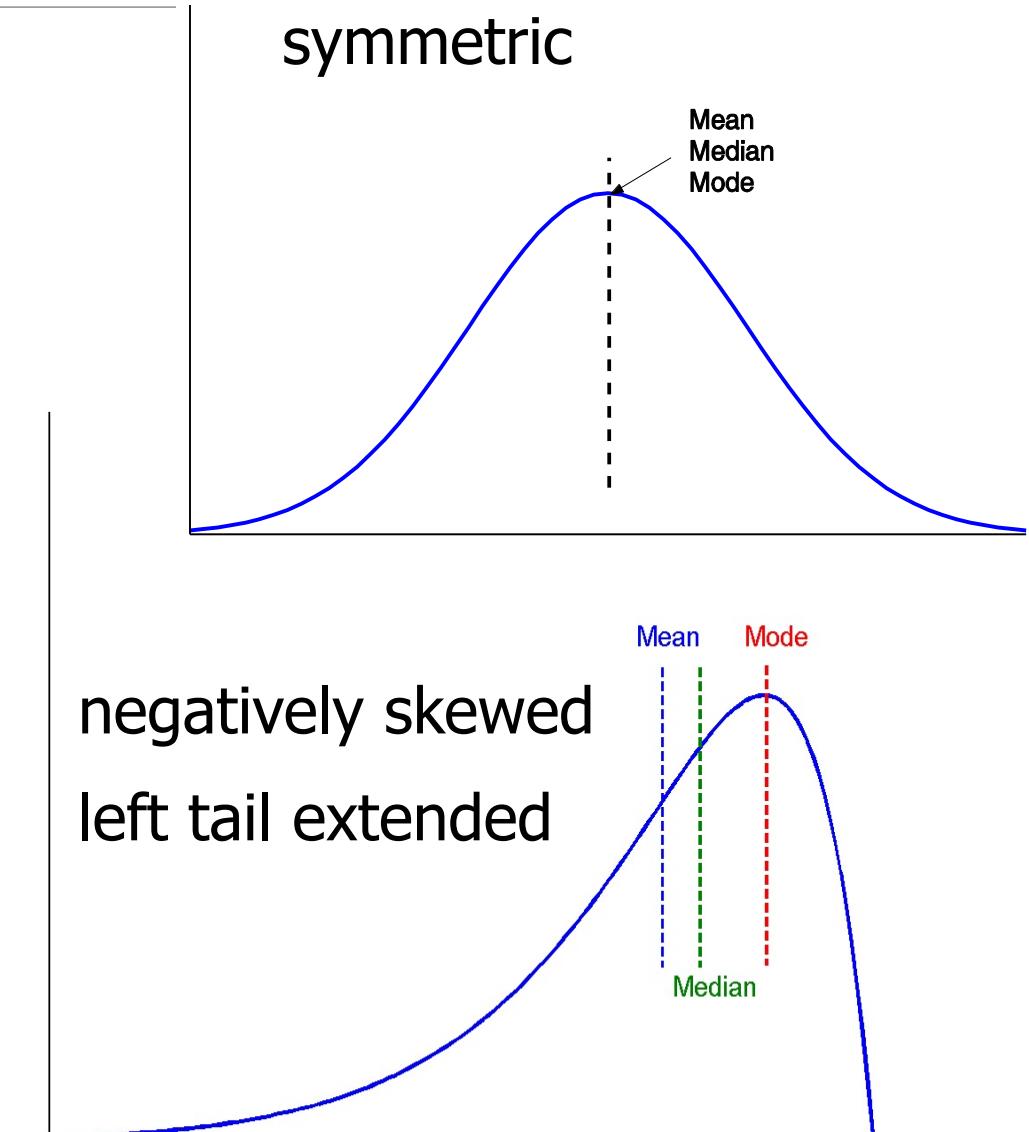
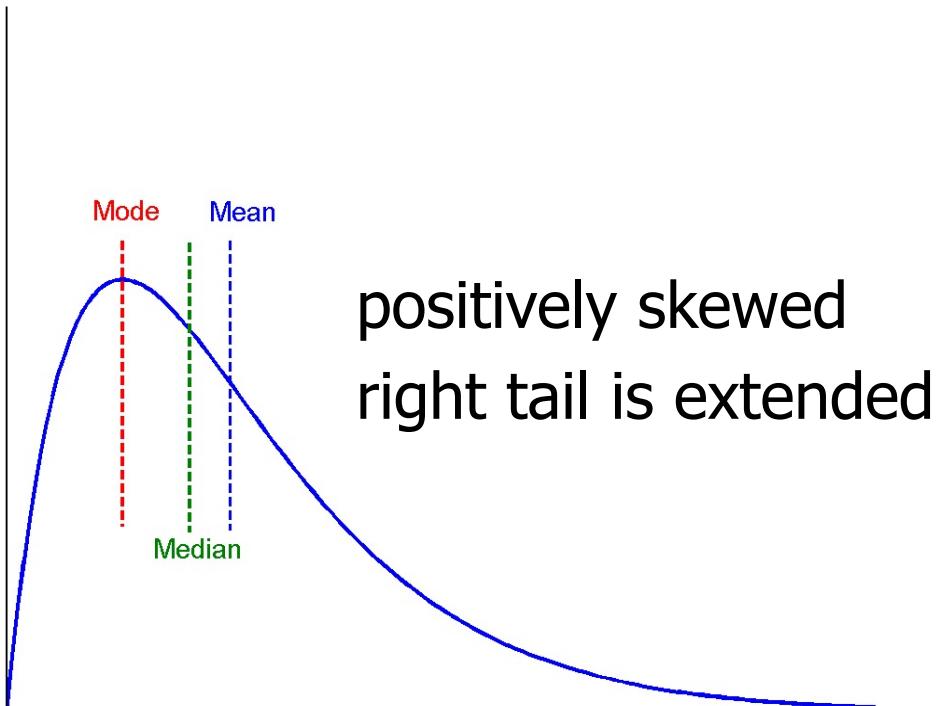


- Trimodal



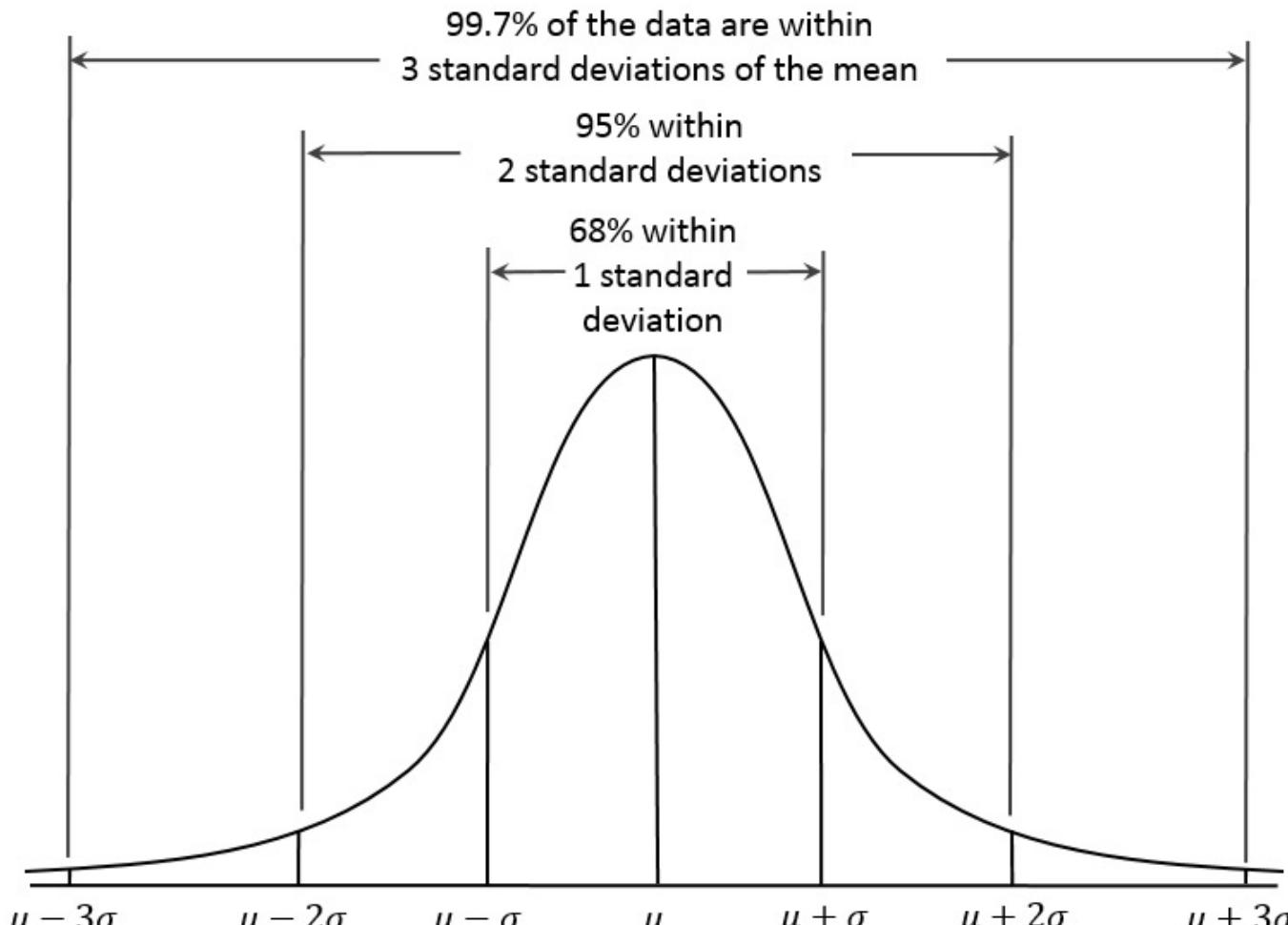
Symmetric vs. Skewed Data

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



Properties of Normal Distribution Curve

← ----- Represent data dispersion, spread ----- →



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

$f(x)$ = probability density function

σ = standard deviation

μ = mean



Represent central tendency

Measures Data Distribution: Variance and Standard Deviation

- Variance and standard deviation (*sample, population*)
 - **Variance:** (algebraic, scalable computation)
 - Q: Can you compute it incrementally and efficiently?

$$\hat{\sigma}_n^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 = \frac{1}{n-1} \left[\sum_{i=1}^n x_i^2 - \frac{1}{n} \left(\sum_{i=1}^n x_i \right)^2 \right]$$

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

Note: The subtle difference of formulae for sample vs. population

- n : the size of the sample
- N : the size of the population

- **Standard deviation** is the square root of variance

Covariance of Numeric Data

- Two numeric attributes A, B, and n observations: $\{(a_1, b_1), \dots, (a_n, b_n)\}$
- Expected values:

$$E(A) = \bar{A} = \frac{\sum_{i=1}^n a_i}{n} \quad E(B) = \bar{B} = \frac{\sum_{i=1}^n b_i}{n}$$

- Covariance

$$\begin{aligned} Cov(A, B) &= E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{n} \\ &= E(A \cdot B) - \bar{A}\bar{B} \end{aligned}$$

- Correlation coefficient

$$r_{A,B} = \frac{Cov(A, B)}{\sigma_A \sigma_B}$$

Correlation Coefficient of Numeric Data

- Correlation Coefficient

$$r_{A,B} = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{n\sigma_A\sigma_B} = \frac{\sum_{i=1}^n (a_i b_i) - n\bar{A}\bar{B}}{n\sigma_A\sigma_B}$$

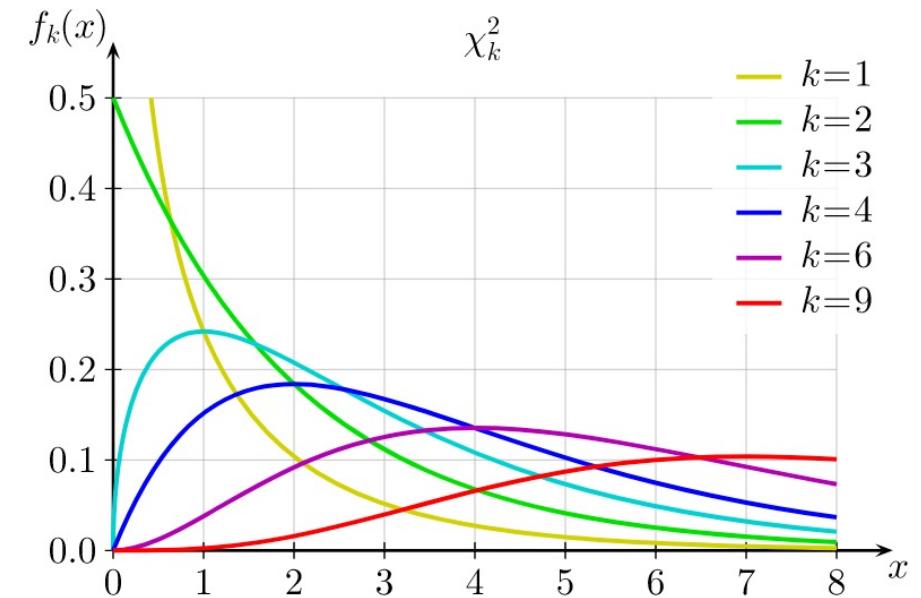
- $r_{A,B}$ lies in $[-1, +1]$
- Positive correlation: $r_{A,B} > 0$, typically increase/decrease together
- Negative correlation: $r_{A,B} < 0$, typically one increases when the other decreases
- Uncorrelated: $r_{A,B} = 0$, no correlation
- Correlation does not mean causation

Hypothesis Tests for Nominal Data

- Hypothesis testing, using a sample of n observations
- Consider a ‘null hypothesis’ and ‘alternative hypothesis’
- Consider a test statistic, and set a significance level
- Calculate the test statistic from observations
- Calculate the p-value, accept or reject the null hypothesis

- Nominal data: χ^2 test
- χ^2 -distribution with k degrees of freedom

$$Q \sim \chi_k^2 \quad \equiv \quad Q = \sum_{i=1}^k g_i^2, \quad g_i \sim N(0, 1)$$

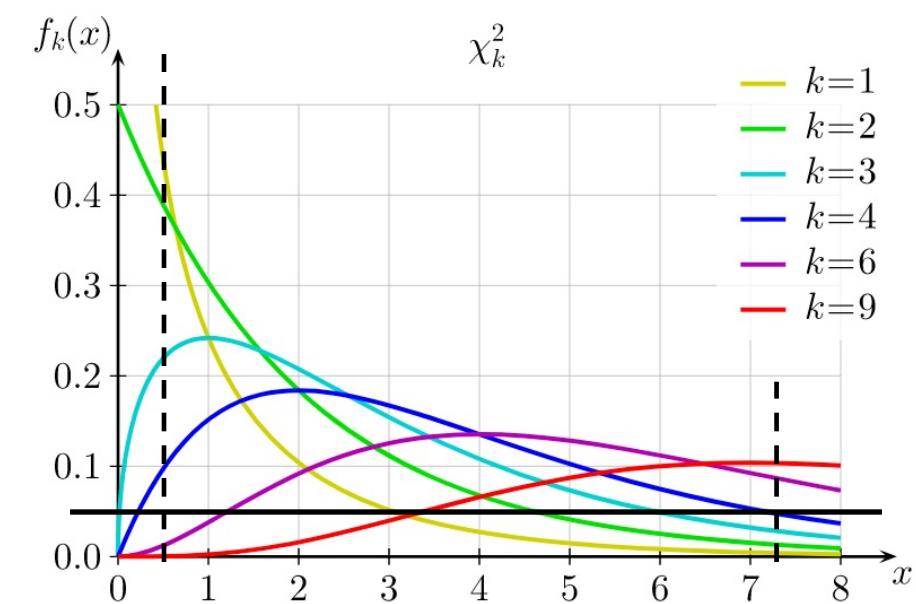


Example: Is My Coin Biased?

- Tossed a coin 100 times: 54 Heads, 46 Tails
 - Is my coin biased
- Hypothesis testing, using a sample of n observations
 - Null hypothesis: Coin is unbiased, *expected behavior* 50 Heads, 50 Tails
 - Test statistic: measures deviation from *expected behavior*

$$\frac{(54 - 50)^2}{50} + \frac{(46 - 50)^2}{50} = 0.64$$

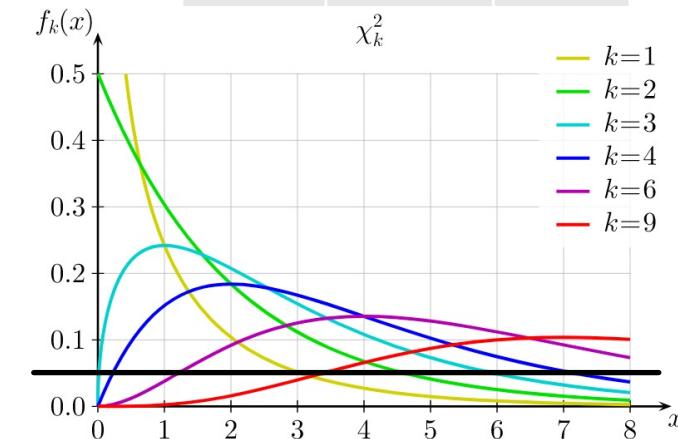
- Test statistic follows the chi-square distribution
 - Degrees of freedom $k = 1$
- Significance level: probability that the deviation is by random chance
 - $\alpha=0.05$ or $\alpha=0.01$ are typical



Chi-Squared Test for Nominal Data

- ❑ Hypothesis: Answers to multiple choice questions are uniformly distributed
 - ❑ Options: A, B, C, D; Null hypothesis says probability is $\frac{1}{4}$
 - ❑ Alternative hypothesis: Distribution is not uniform
- ❑ Set significance level $\alpha=0.05$
- ❑ Tail probability (distribution) threshold
- ❑ 3 degrees of freedom, χ^2 value to reject null = 7.815
- ❑ Sample n=100 questions for the analysis
- ❑ χ^2 statistic:
$$\sum_{i=1}^4 \frac{(o_i - e_i)^2}{e_i} = 6$$
- ❑ Probability is $\geq 0.1 > \alpha$, cannot reject null hypothesis

Correct Choice	Expected	Actual
A	25	20
B	25	20
C	25	25
D	25	35



Correlation Analysis for Nominal Data

- Two nominal attributes A and B
 - A has values $\{a_1, a_2, \dots, a_c\}$
 - B has values $\{b_1, b_2, \dots, b_r\}$
- Create a contingency table from data
 - Counts of joint events ($A = a_i, B = b_j$)
- Null hypothesis: A and B are independent
- The expected frequency in each entry: $e_{ij} = \frac{\text{count}(A = a_i) \times \text{count}(B = b_j)}{n}$
- The χ^2 statistic, $(r-1)(c-1)$ degrees of freedom

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

Table 2.2: Example 2.1's 2×2 Contingency Table Data

	male	female	Total
fiction	250 (90)	200 (360)	450
non-fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

Note: Are *gender* and *preferred_reading* correlated?

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

Example: Correlation Analysis for Nominal Data

- Compute expected frequency

$$e_{11} = \frac{\text{count}(male) \times \text{count}(fiction)}{n} = \frac{300 \times 450}{1500} = 90$$

- The χ^2 statistic

$$\begin{aligned}\chi^2 &= \frac{(250 - 90)^2}{90} + \frac{(50 - 210)^2}{210} + \frac{(200 - 360)^2}{360} + \frac{(1000 - 840)^2}{840} \\ &= 284.44 + 121.90 + 71.11 + 30.48 = 507.93.\end{aligned}$$

- With 1 degree of freedom, at 0.001 significance level
 - Value needed to reject null hypothesis: 10.828
 - More than that, hence null hypothesis is rejected

Table 2.2: Example 2.1's 2×2 Contingency Table Data

	male	female	Total
fiction	250 (90)	200 (360)	450
non-fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

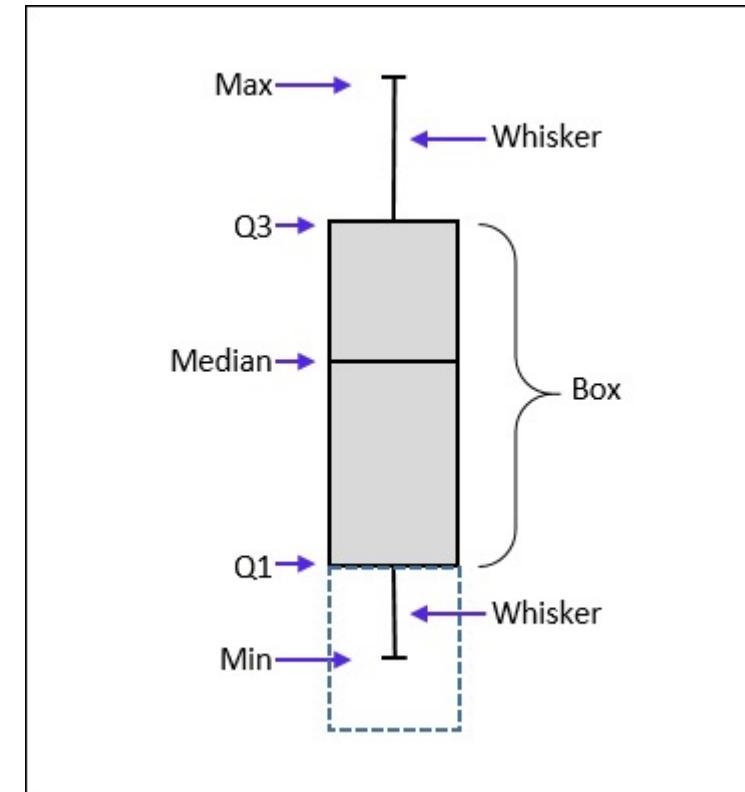
Note: Are *gender* and *preferred_reading* corre-

Graphic Displays of Basic Statistical Descriptions

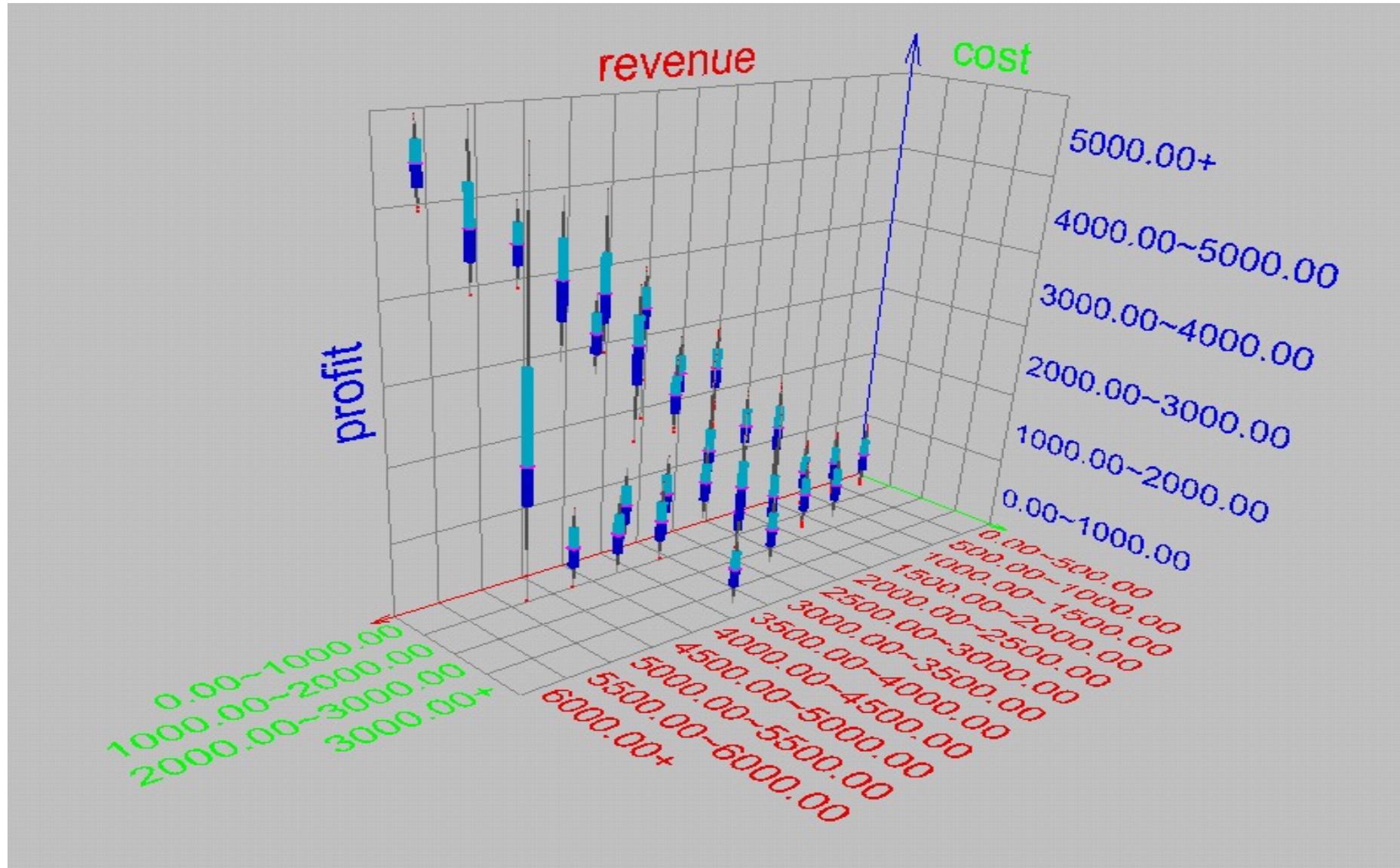
- **Boxplot:** graphic display of five-number summary
- **Histogram:** x-axis are values, y-axis repres. frequencies
- **Quantile plot:** each value x_i is paired with f_i indicating that approximately $100 f_i \%$ of data are $\leq x_i$
- **Quantile-quantile (q-q) plot:** graphs the quantiles of one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

Measuring the Dispersion of Data: Quartiles & 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 , median, Q_3 , max
- **Boxplot:** Data is represented with a box
 - Q_1 , Q_3 , IQR: The ends of the box are at the first and third quartiles, i.e., the height of the box is IQR
 - Median (Q_2) is marked by a line within the box
 - Whiskers: two lines outside the box extended to Minimum and Maximum
 - Outliers: points beyond a specified outlier threshold, plotted individually
 - **Outlier:** usually, a value higher/lower than $1.5 \times \text{IQR}$



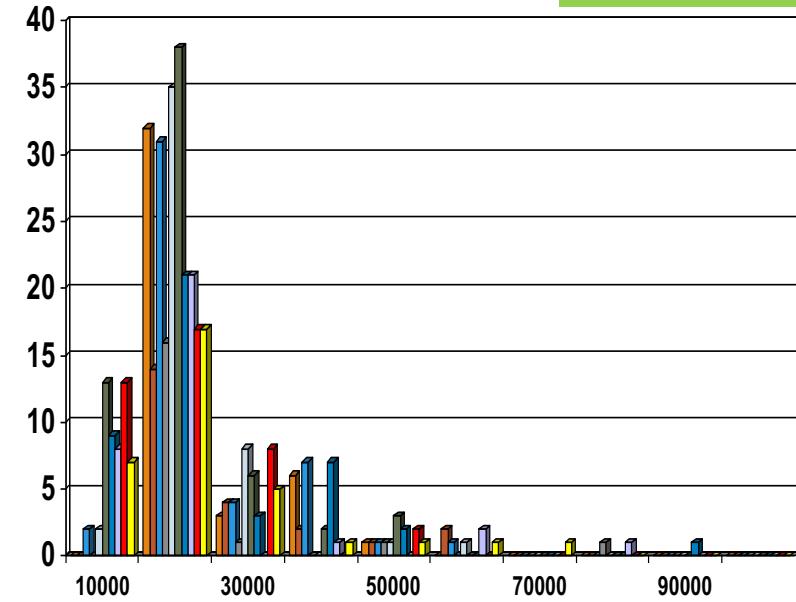
Visualization of Data Dispersion: 3-D Boxplots



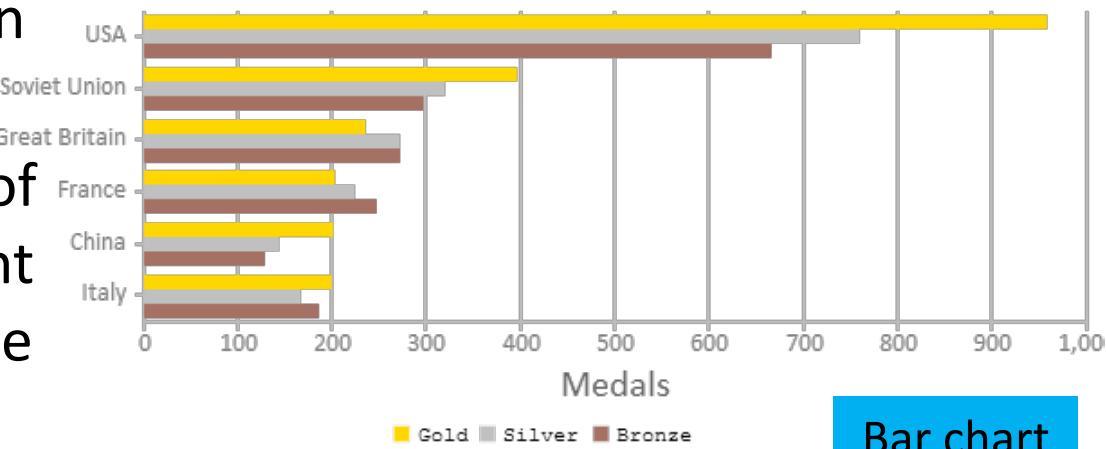
Histogram Analysis

- ❑ Histogram: Graph display of tabulated frequencies, shown as bars
- ❑ Differences between histograms and bar charts
 - ❑ Histograms are used to show distributions of variables while bar charts are used to compare variables
 - ❑ Histograms plot binned quantitative data while bar charts plot categorical data
 - ❑ Bars can be reordered in bar charts but not in histograms
 - ❑ Differs from a bar chart in that it is the area of the bar that denotes the value, not the height as in bar charts, a crucial distinction when the categories are not of uniform width

Histogram

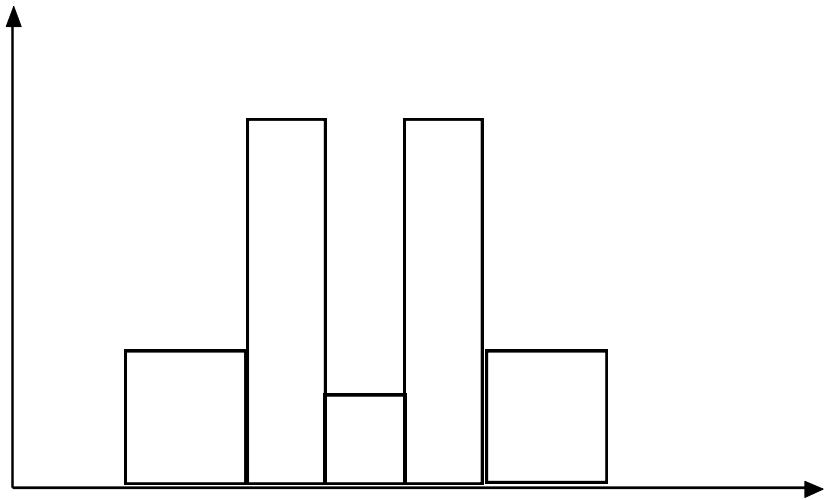


Olympic Medals of all Times (till 2012 Olympics)

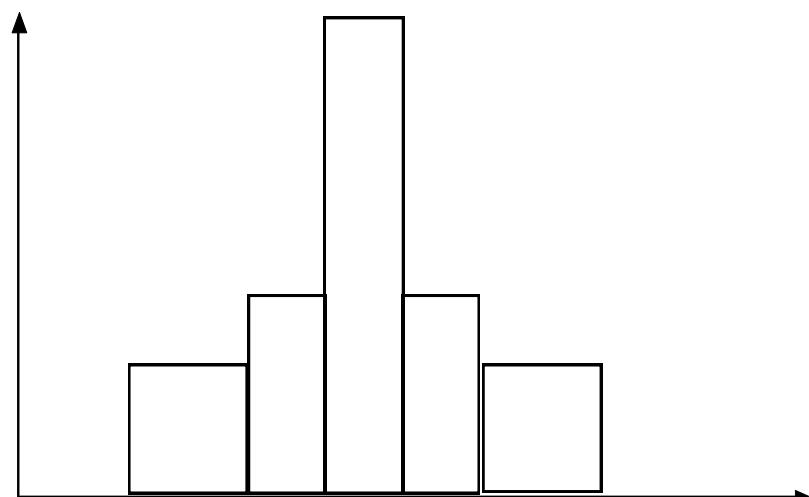


Bar chart

Histograms Often Tell More than Boxplots

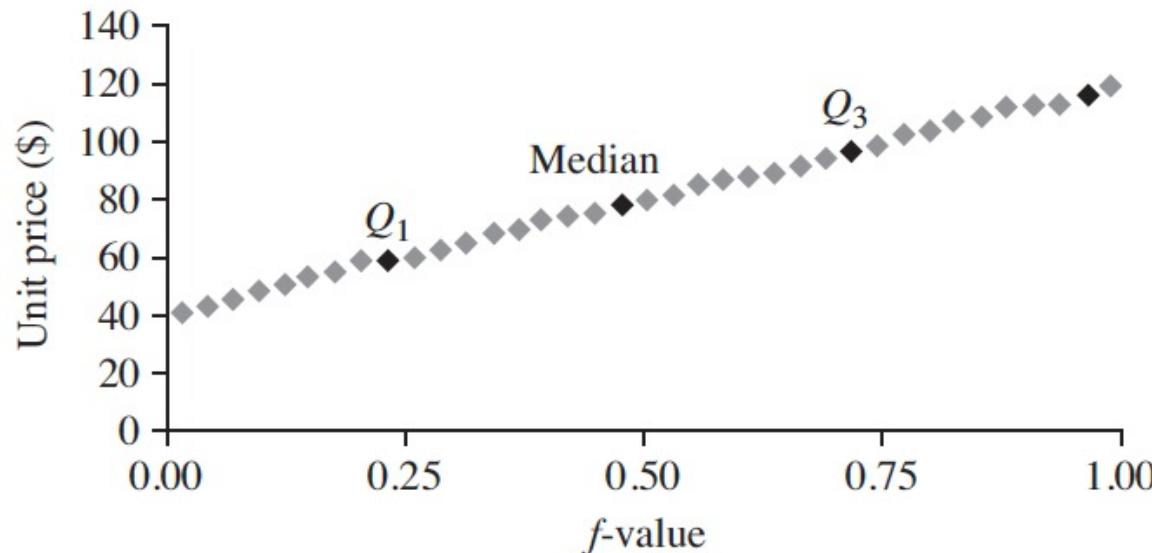


- The two histograms shown in the left may have the same boxplot representation
- The same values for: min, Q1, median, Q3, max
- But they have rather different data distributions



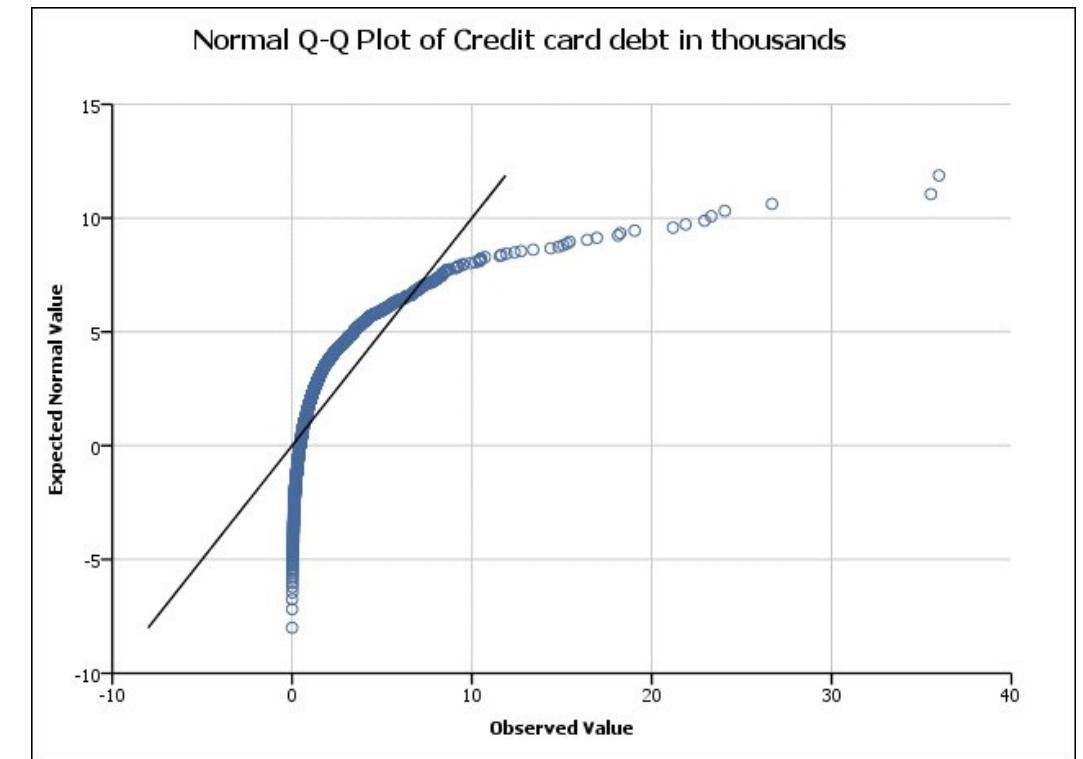
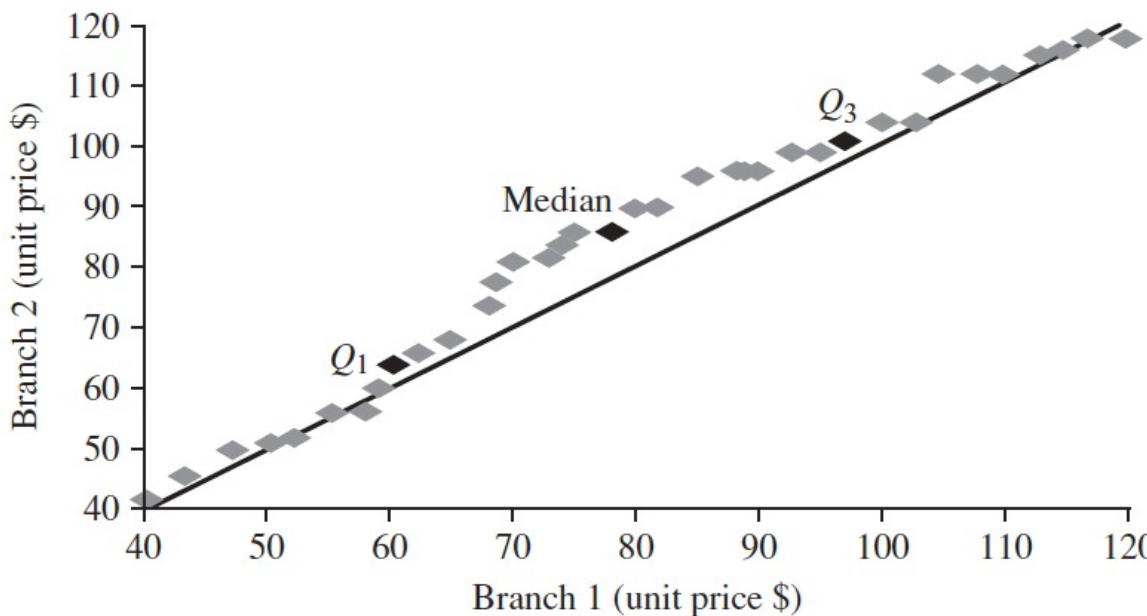
Quantile Plot

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

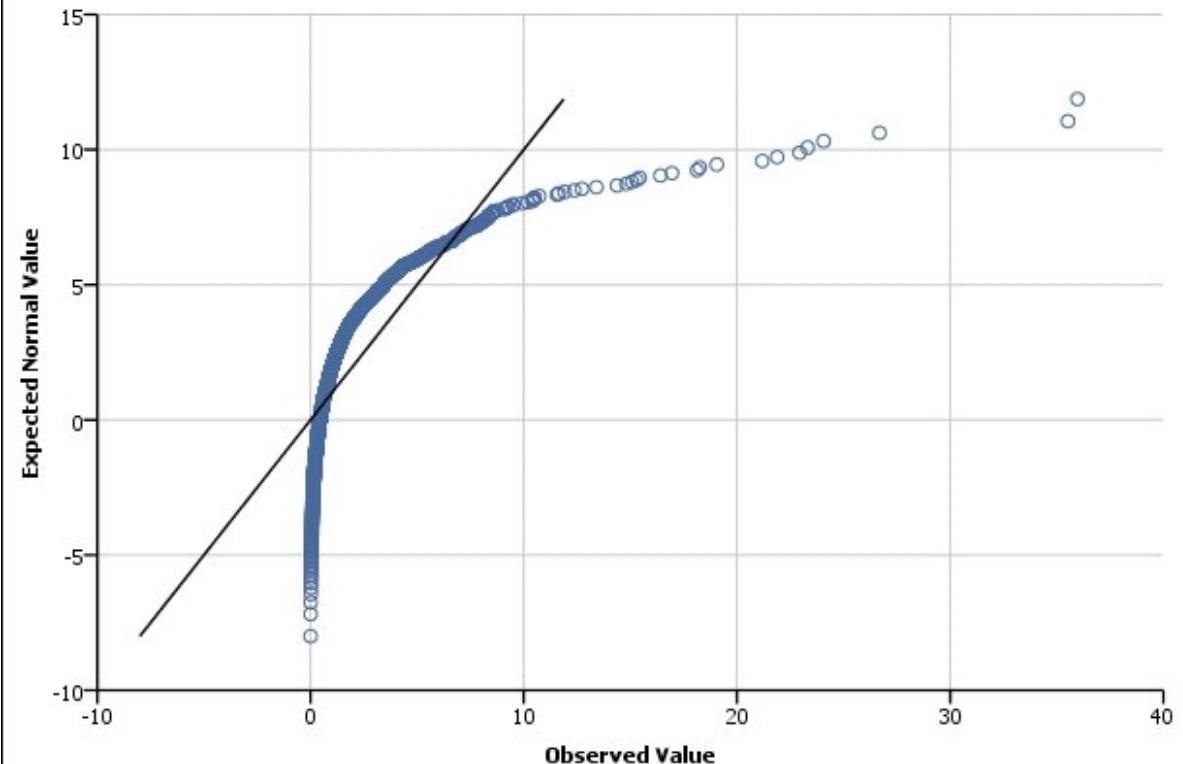


Quantile-Quantile (Q-Q) Plot

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

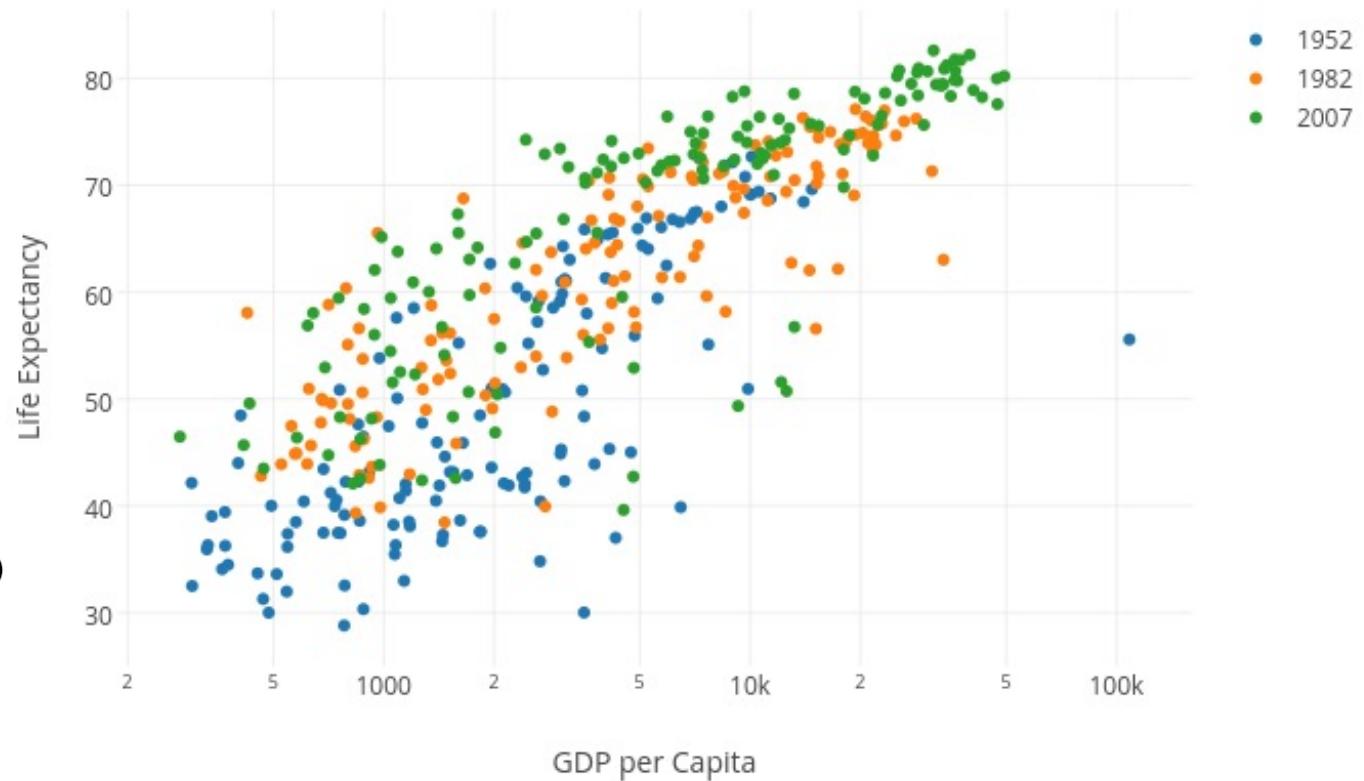
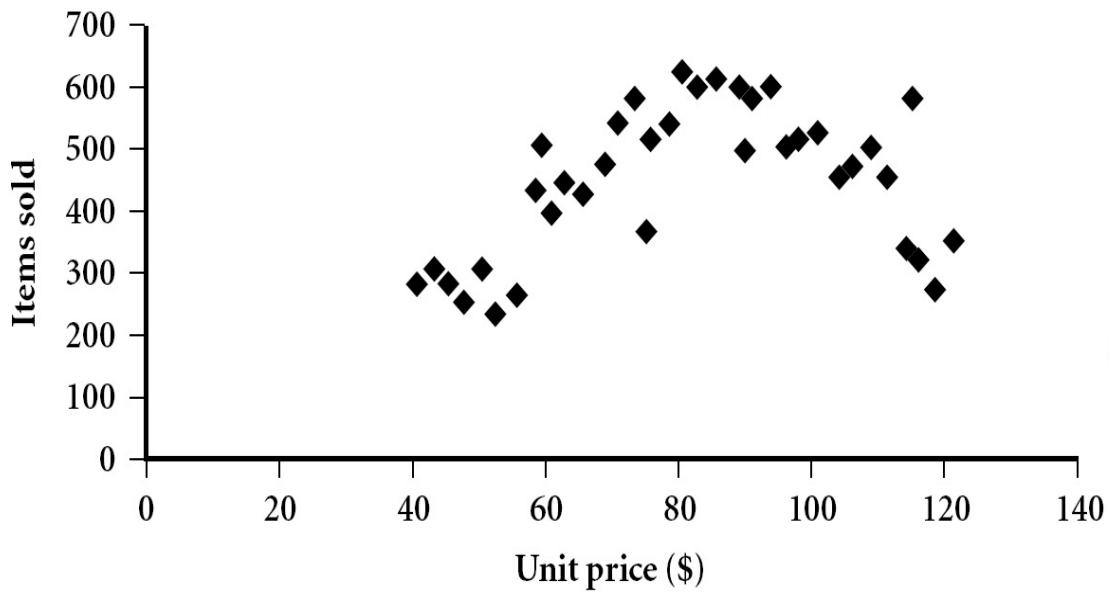


Normal Q-Q Plot of Credit card debt in thousands

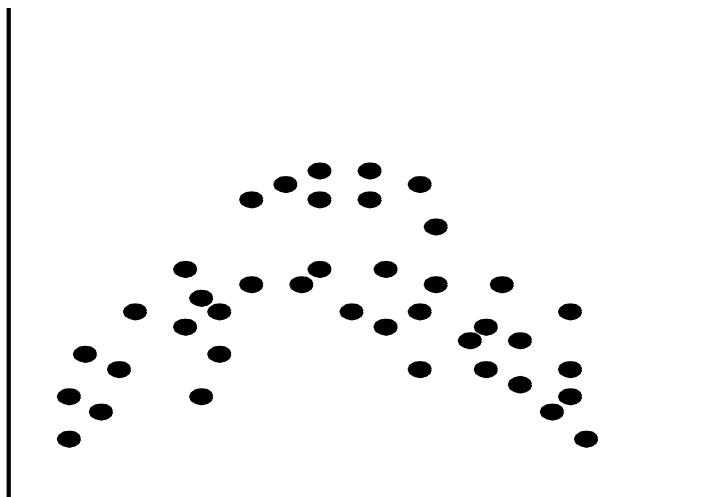
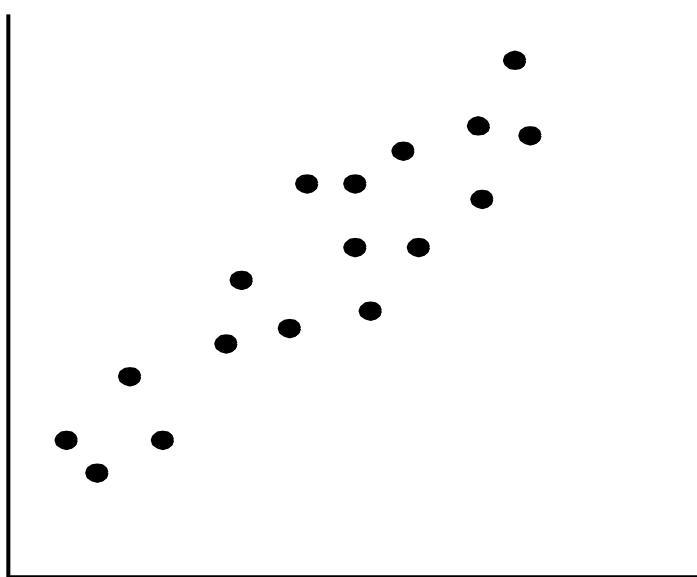


Scatter plot

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

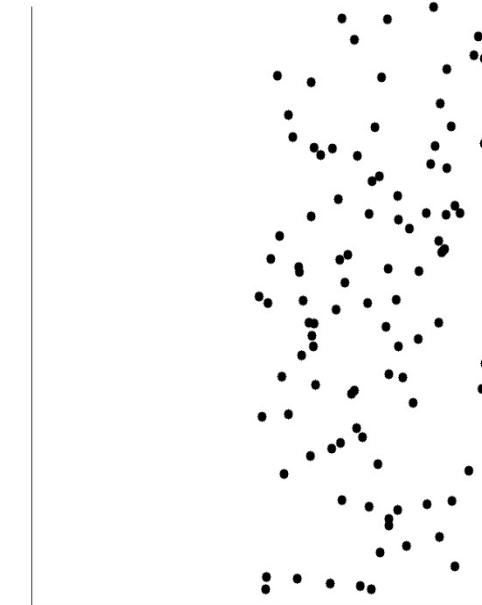
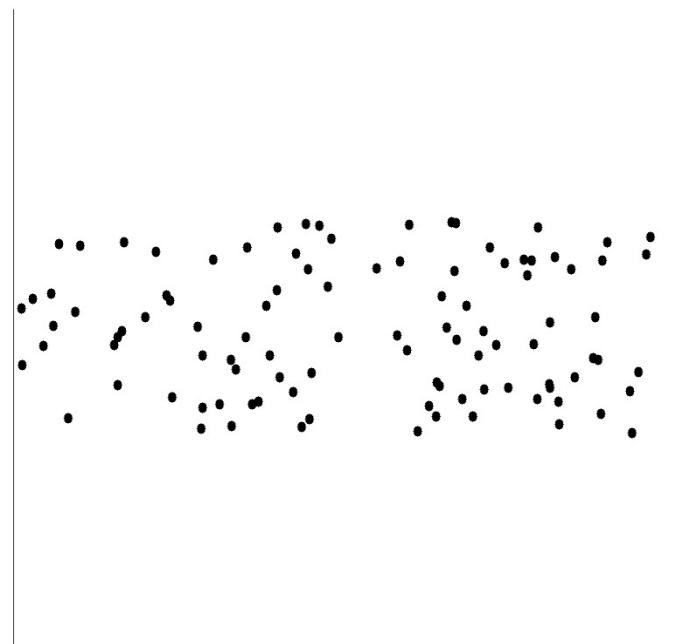
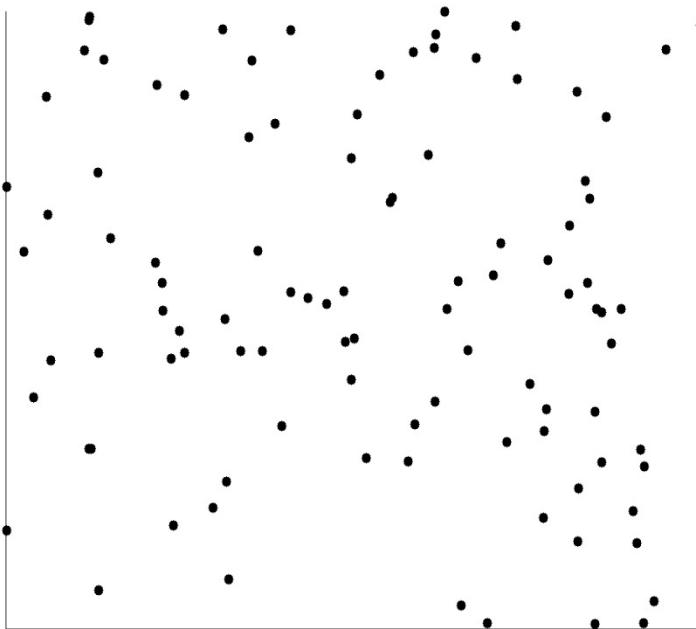


Positively and Negatively Correlated Data



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

Uncorrelated Data



Chapter 2. Getting to Know Your Data

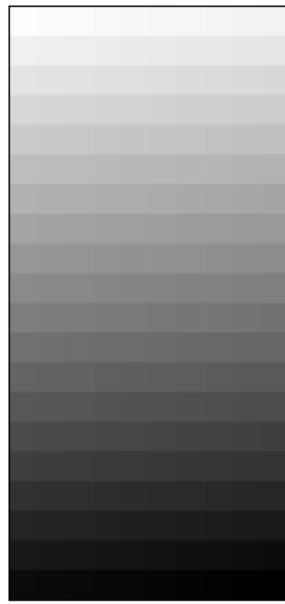
- Data Objects and Attribute Types
- Basic Statistical Descriptions of Data
- Data Visualization 
- Measuring Data Similarity and Correlation
- Summary

Data Visualization

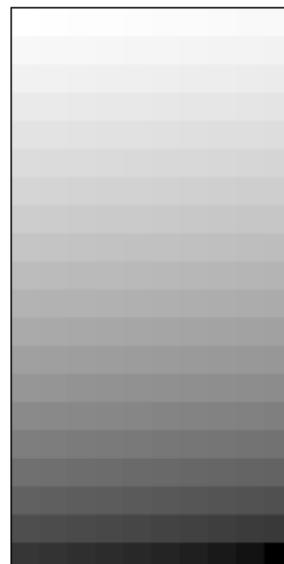
- Why data visualization?
 - Gain insight into an information space by mapping data onto graphical primitives
 - Provide qualitative overview of large data sets
 - Search for patterns, trends, structure, irregularities, relationships among data
 - Help find interesting regions and suitable parameters for further quantitative analysis
 - Provide a visual proof of computer representations derived
- Categorization of visualization methods:
 - Pixel-oriented visualization techniques
 - Geometric projection visualization techniques
 - Icon-based visualization techniques
 - Hierarchical visualization techniques
 - Visualizing complex data and relations

Pixel-Oriented Visualization Techniques

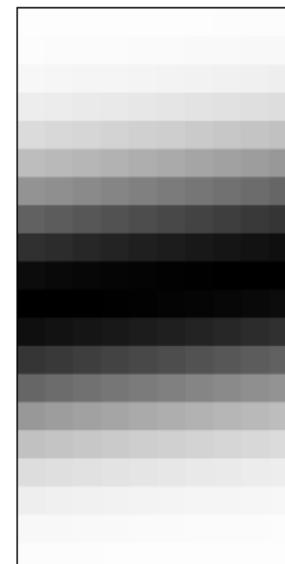
- ❑ For a data set of m dimensions, create m windows on the screen, one for each dimension
- ❑ The m dimension values of a record are mapped to m pixels at the corresponding positions in the windows
- ❑ The colors of the pixels reflect the corresponding values



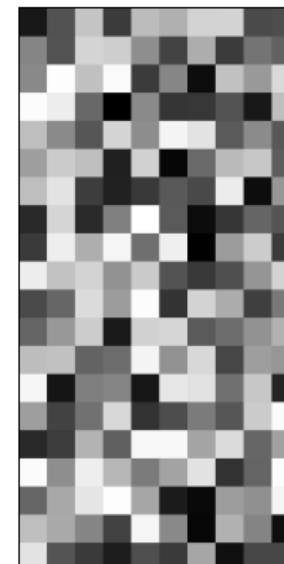
(a) Income



(b) Credit Limit



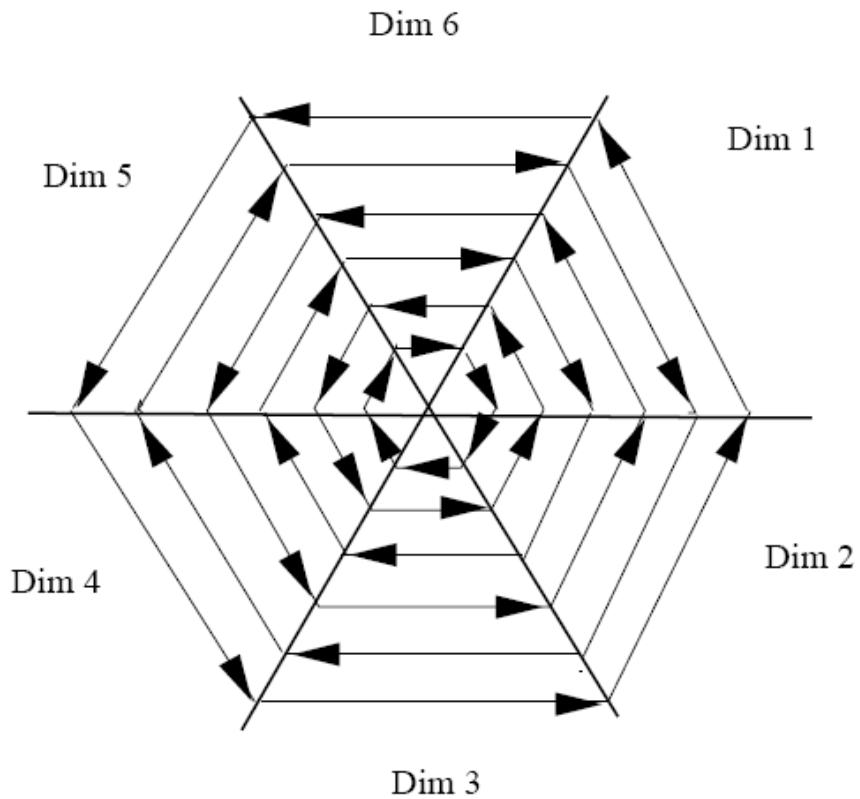
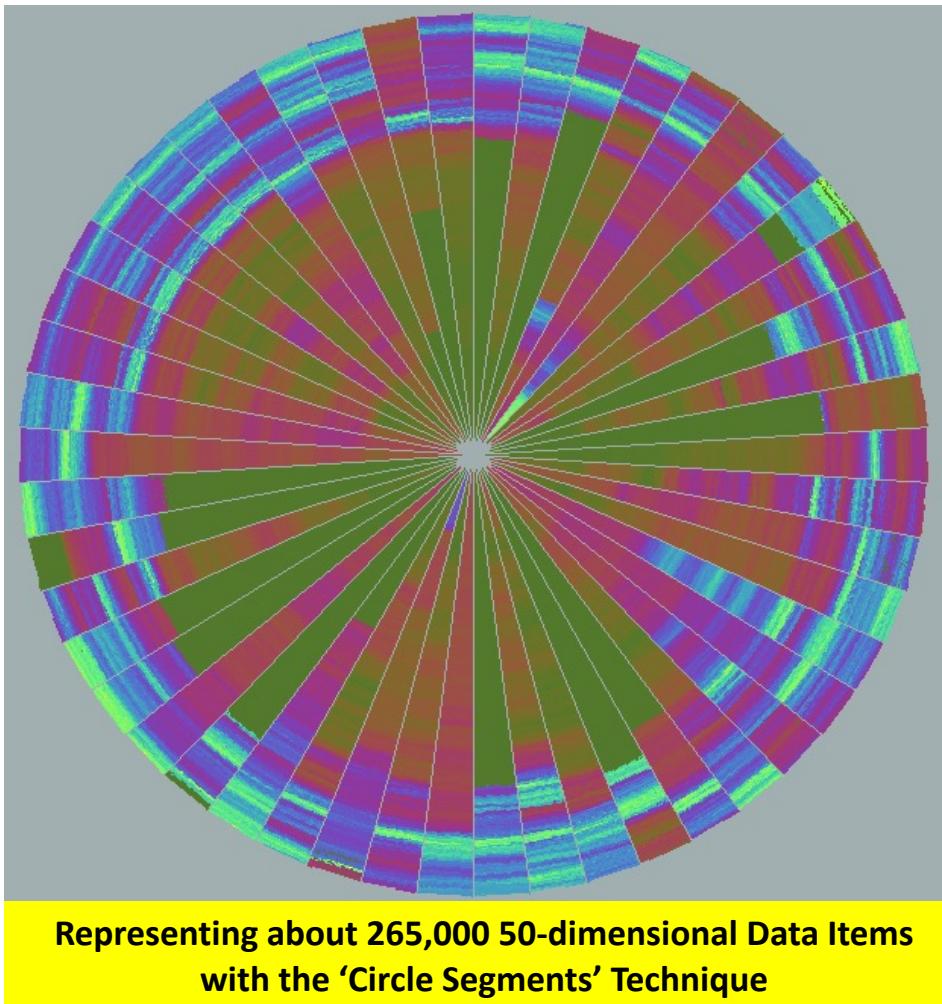
(c) transaction volume



(d) age

Laying Out Pixels in Circle Segments

- To save space and show the connections among multiple dimensions, space filling is often done in a circle segment



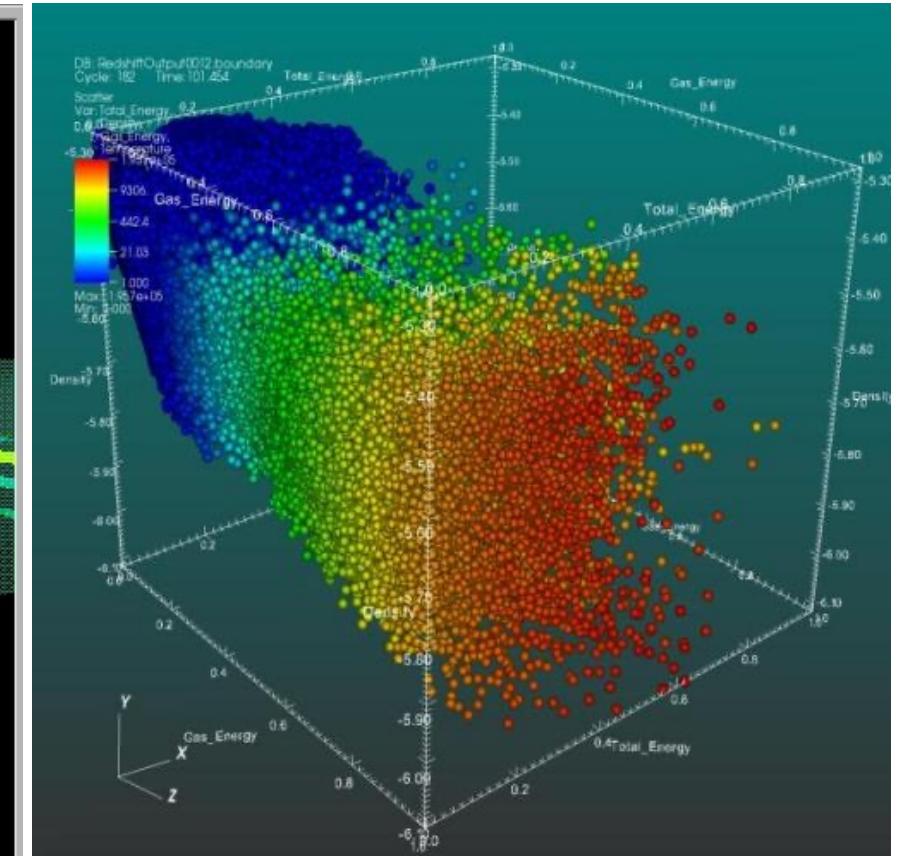
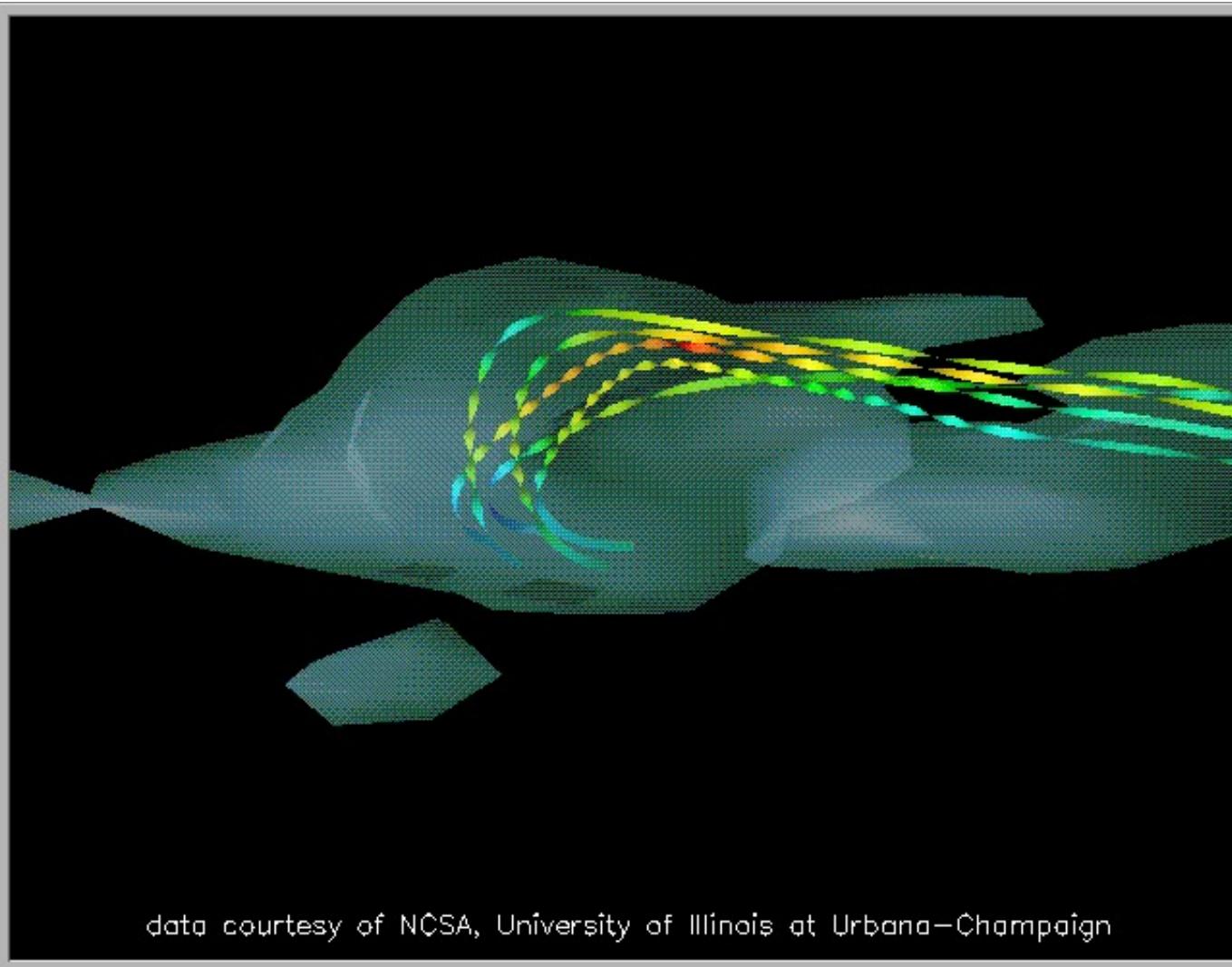
(b) Laying out pixels in circle segment

Geometric Projection Visualization Techniques

- Visualization of geometric transformations and projections of the data
- Methods
 - Direct visualization
 - Scatterplot and scatterplot matrices
 - Landscapes
 - Projection pursuit technique: Help users find meaningful projections of multidimensional data
 - Prosection views
 - Hyperslice
 - Parallel coordinates

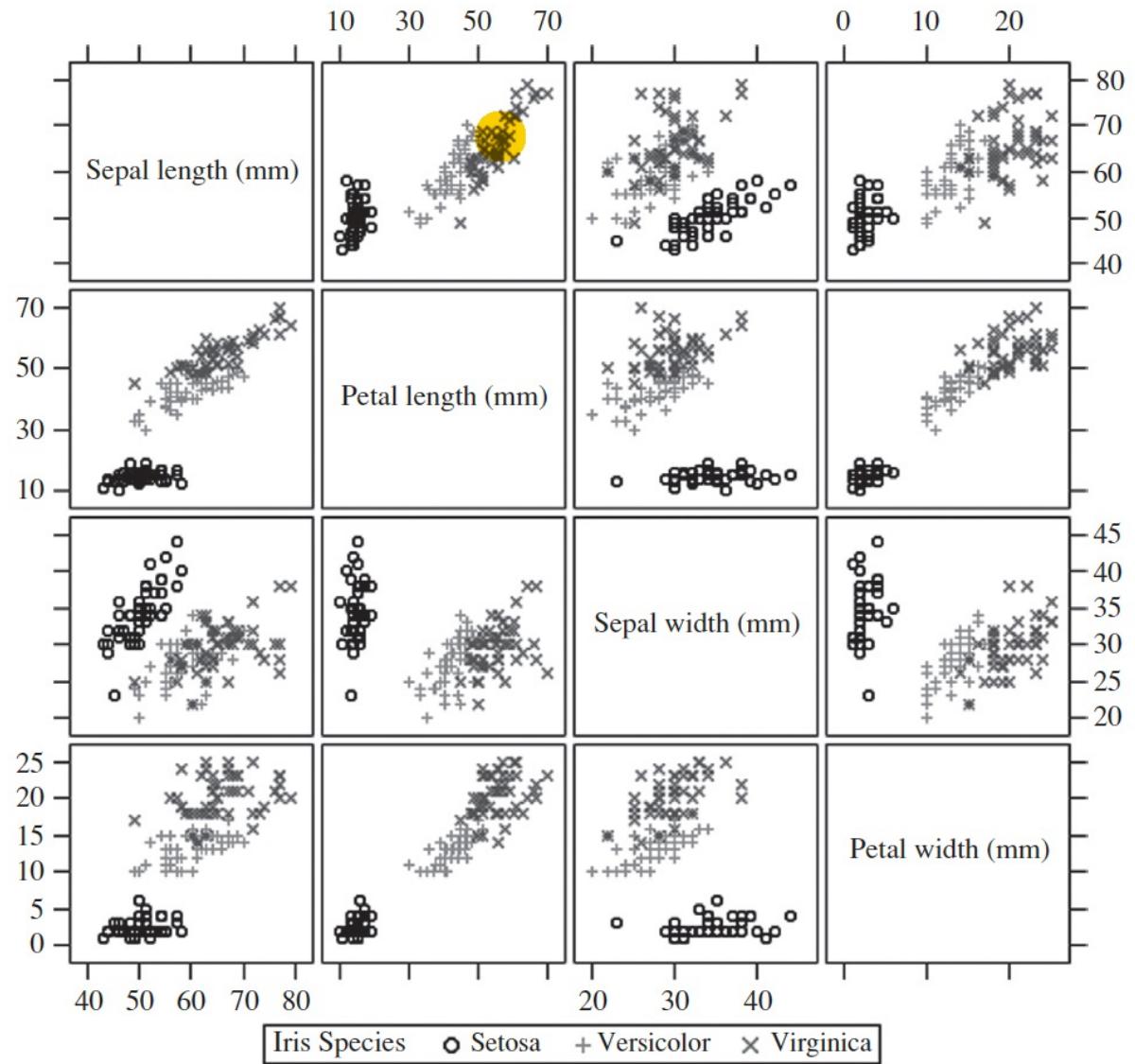
Direct Data Visualization

Ribbons with Twists Based on Vorticity



From Wiki: Scatter plot: A 3D scatter plot to visualize multivariate data

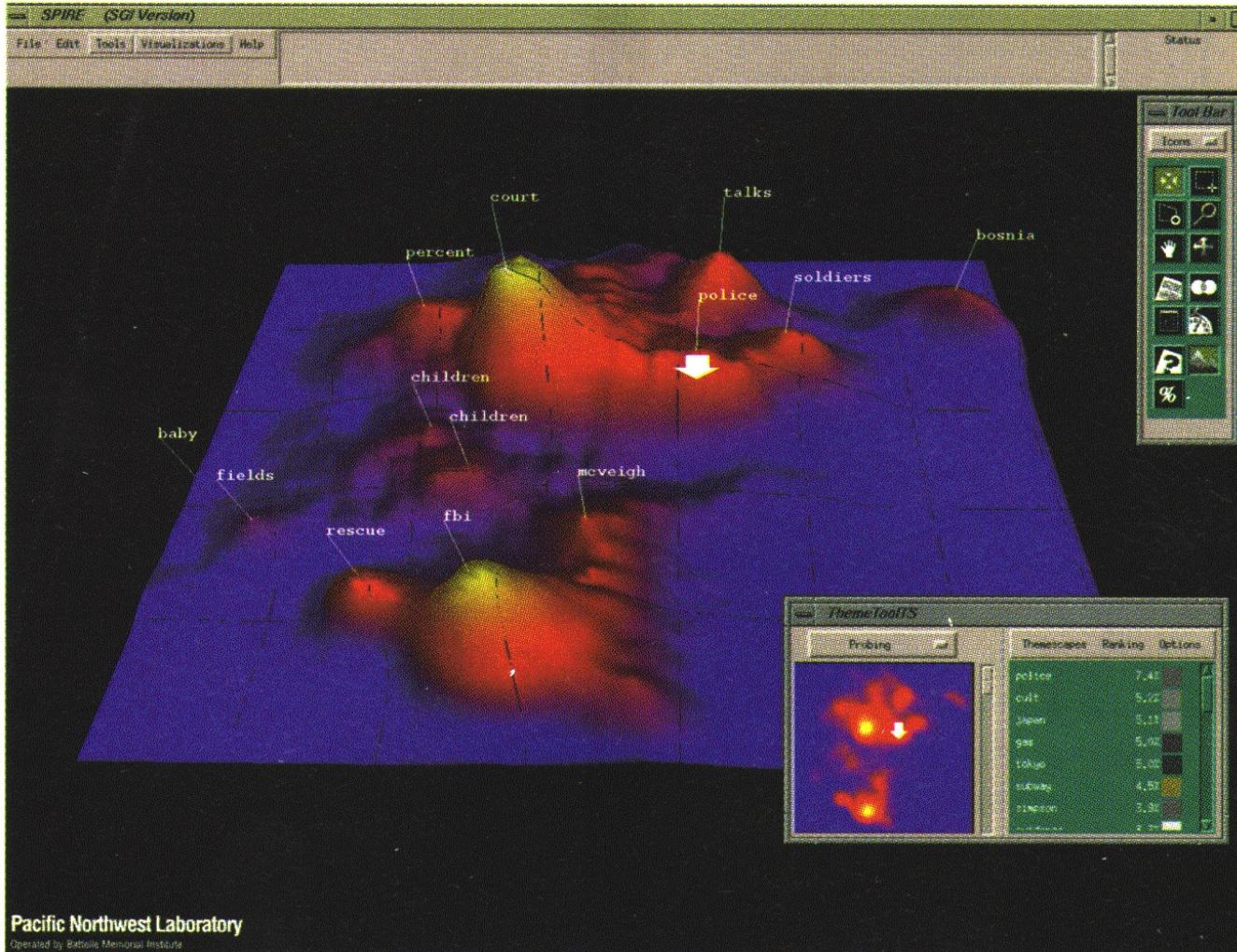
Scatterplot Matrices



- Matrix of scatterplots (x-y-diagrams) of the k-dim. data
- A total of $k(k-1)/2$ distinct scatterplots

Landscapes

Used by permission of B. Wright, Visible Decisions Inc.

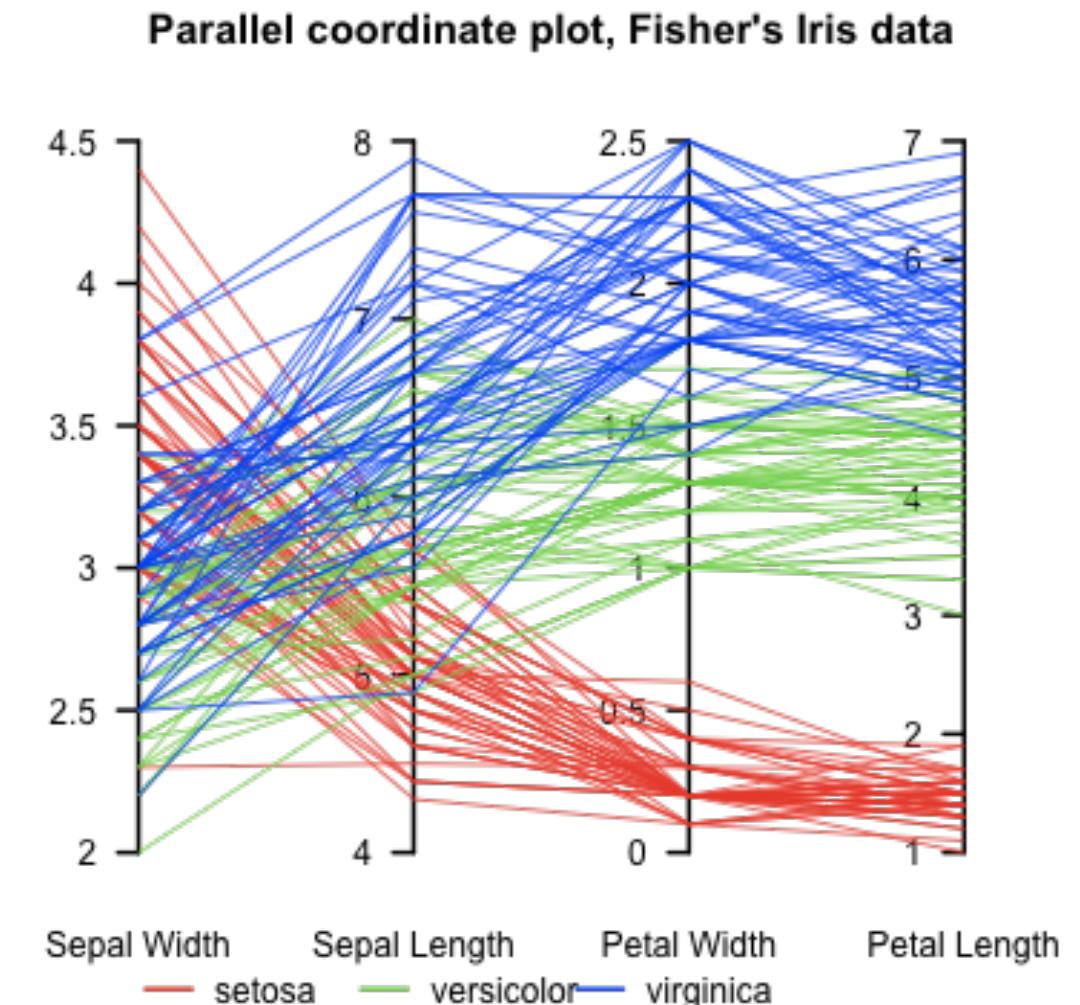


news articles visualized as a landscape

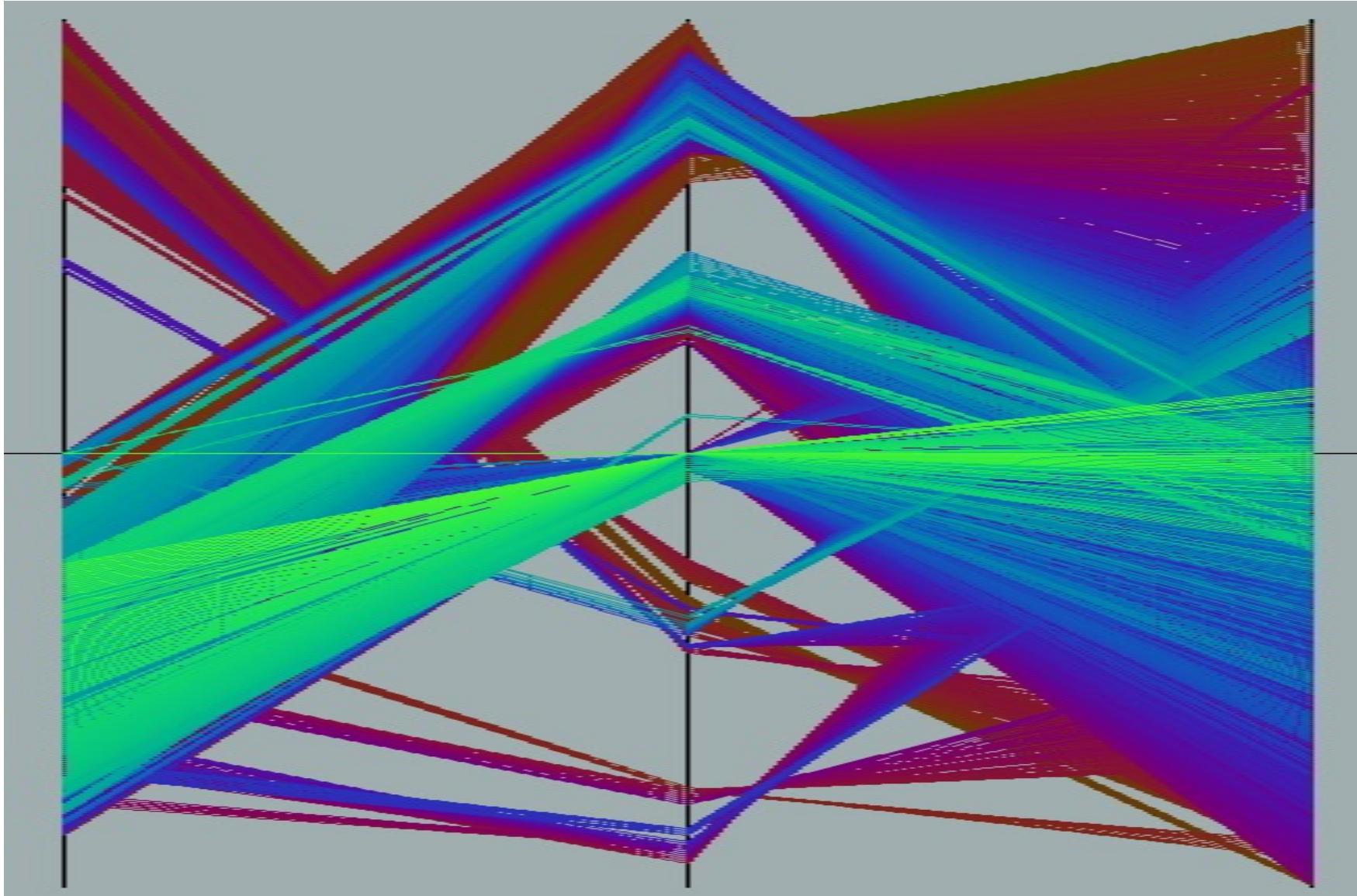
- Visualization of the data as perspective landscape
- The data needs to be transformed into a (possibly artificial) 2D spatial representation which preserves the characteristics of the data

Parallel Coordinates

- n equidistant axes which are parallel to one of the screen axes and correspond to the attributes
- The axes are scaled to the [minimum, maximum]: range of the corresponding attribute
- Every data item corresponds to a polygonal line which intersects each of the axes at the point which corresponds to the value for the attribute



Parallel Coordinates of a Data Set

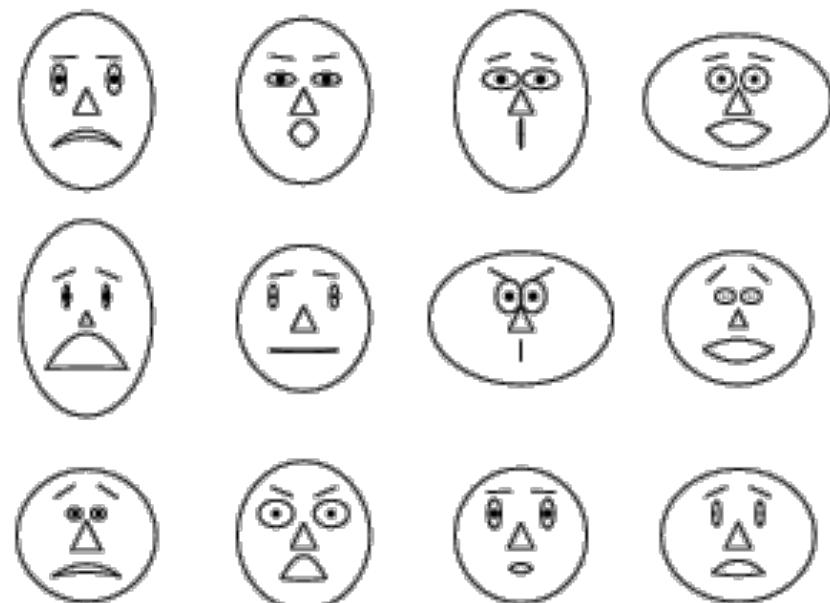


Icon-Based Visualization Techniques

- ❑ Visualization of the data values as features of icons
- ❑ Typical visualization methods
 - ❑ Chernoff Faces
 - ❑ Stick Figures
- ❑ General techniques
 - ❑ Shape coding: Use shape to represent certain information encoding
 - ❑ Color icons: Use color icons to encode more information
 - ❑ Tile bars: Use small icons to represent the relevant feature vectors in document retrieval

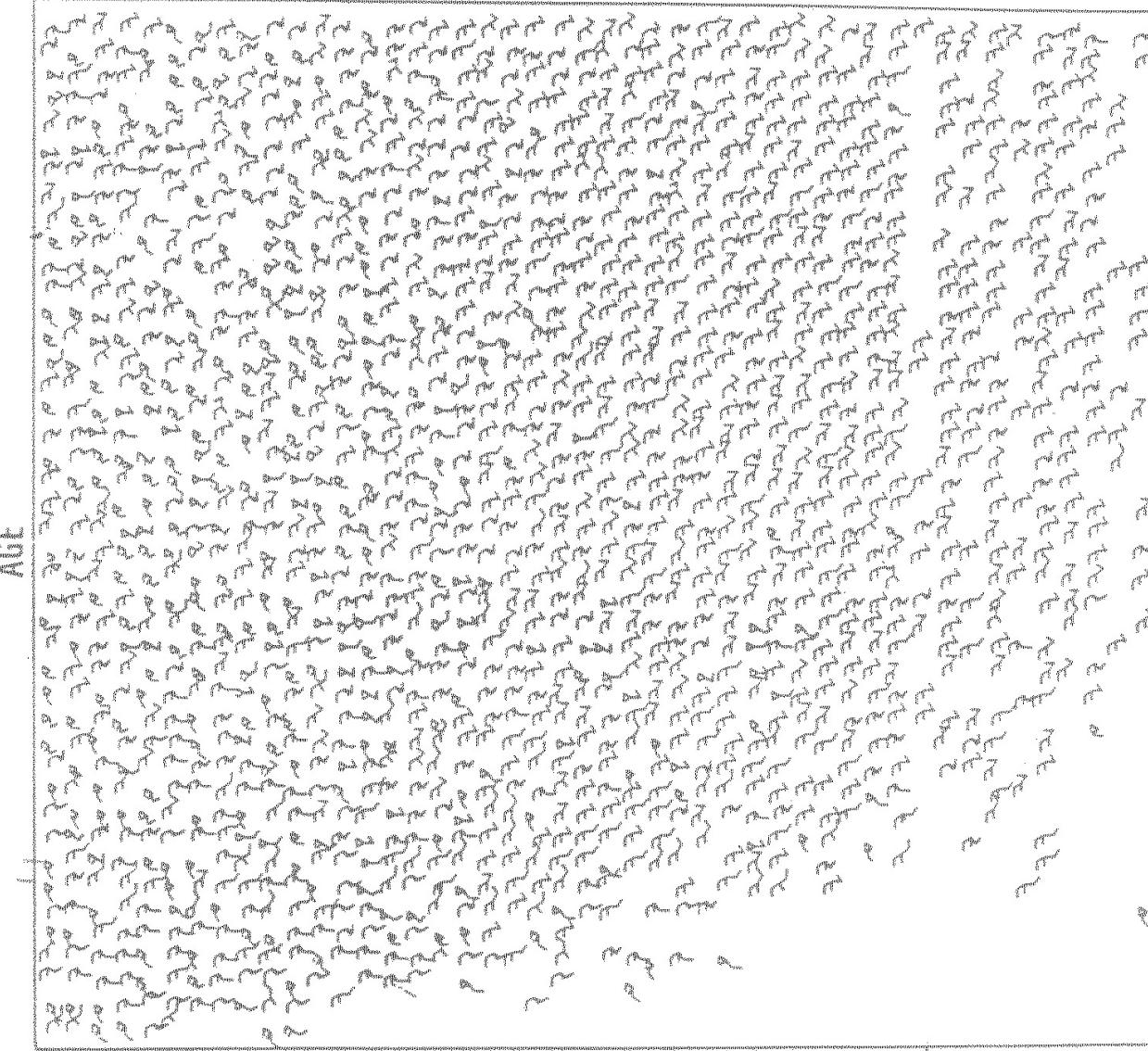
Chernoff Faces

- A way to display variables on a two-dimensional surface, e.g., let x be eyebrow slant, y be eye size, z be nose length, etc.
- The figure shows faces produced using 10 characteristics--head eccentricity, eye size, eye spacing, eye eccentricity, pupil size, eyebrow slant, nose size, mouth shape, mouth size, and mouth opening): Each assigned one of 10 possible values, generated using [*Mathematica*](#) (S. Dickson)
- REFERENCE: Gonick, L. and Smith, W. [*The Cartoon Guide to Statistics*](#). New York: Harper Perennial, p. 212, 1993
- Weisstein, Eric W. "Chernoff Face." From *MathWorld--A Wolfram Web Resource*.
mathworld.wolfram.com/ChernoffFace.html



Stick Figure

used by permission of G. Grinstein, University of Massachusetts at Lowell

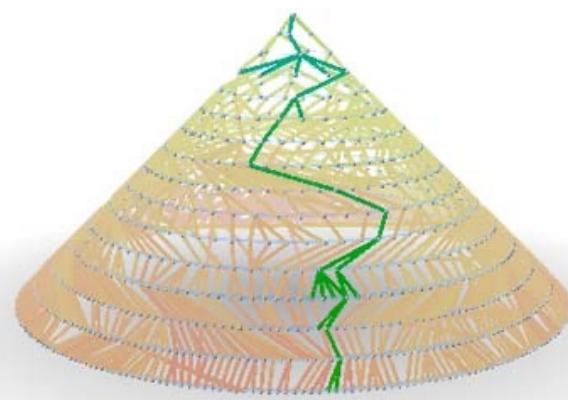
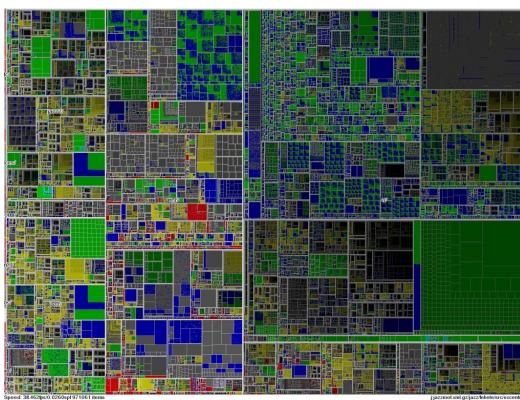
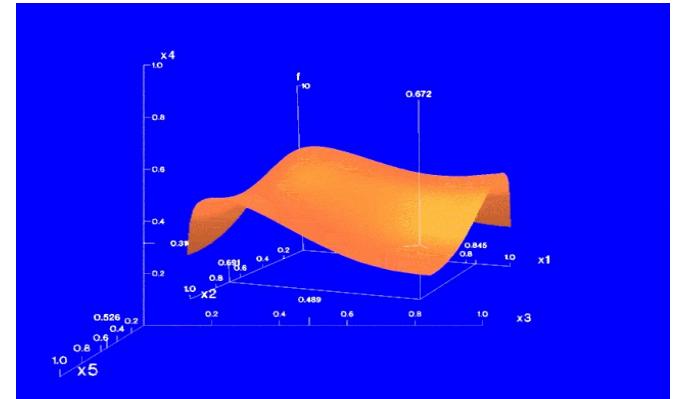
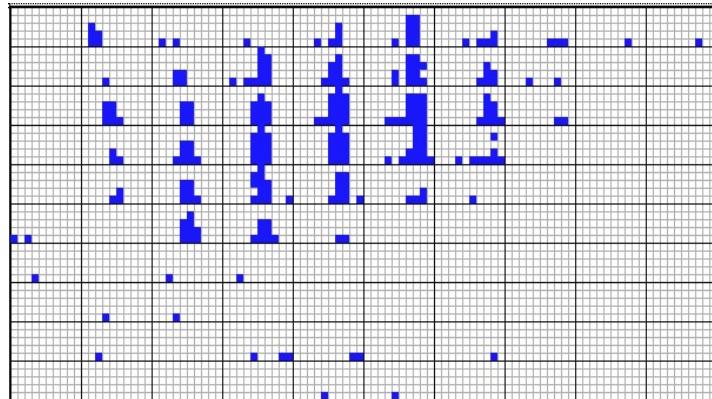


- A census data figure showing age, income, gender, education, etc.

- A 5-piece stick figure (1 body and 4 limbs w. different angle/length)

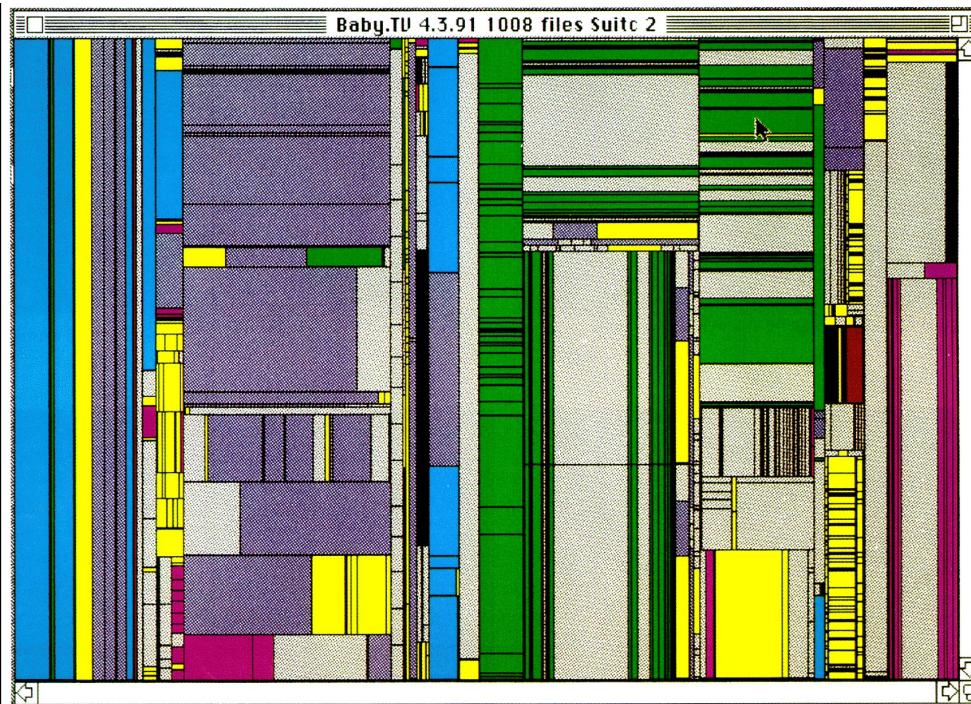
Hierarchical Visualization Techniques

- ❑ Visualization of the data using a hierarchical partitioning into subspaces
- ❑ Methods
 - ❑ Dimensional Stacking
 - ❑ Worlds-within-Worlds
 - ❑ Tree-Map
 - ❑ Cone Trees
 - ❑ InfoCube

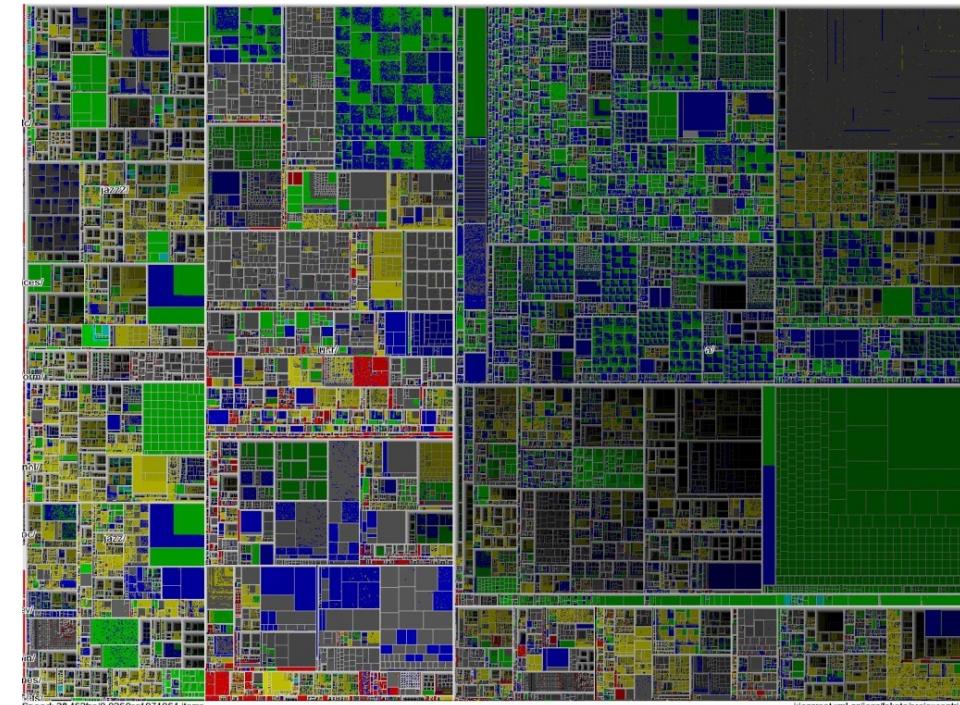


Tree-Map

- Screen-filling method which uses a hierarchical partitioning of the screen into regions depending on the attribute values
- The x- and y-dimension of the screen are partitioned alternately according to the attribute values (classes)

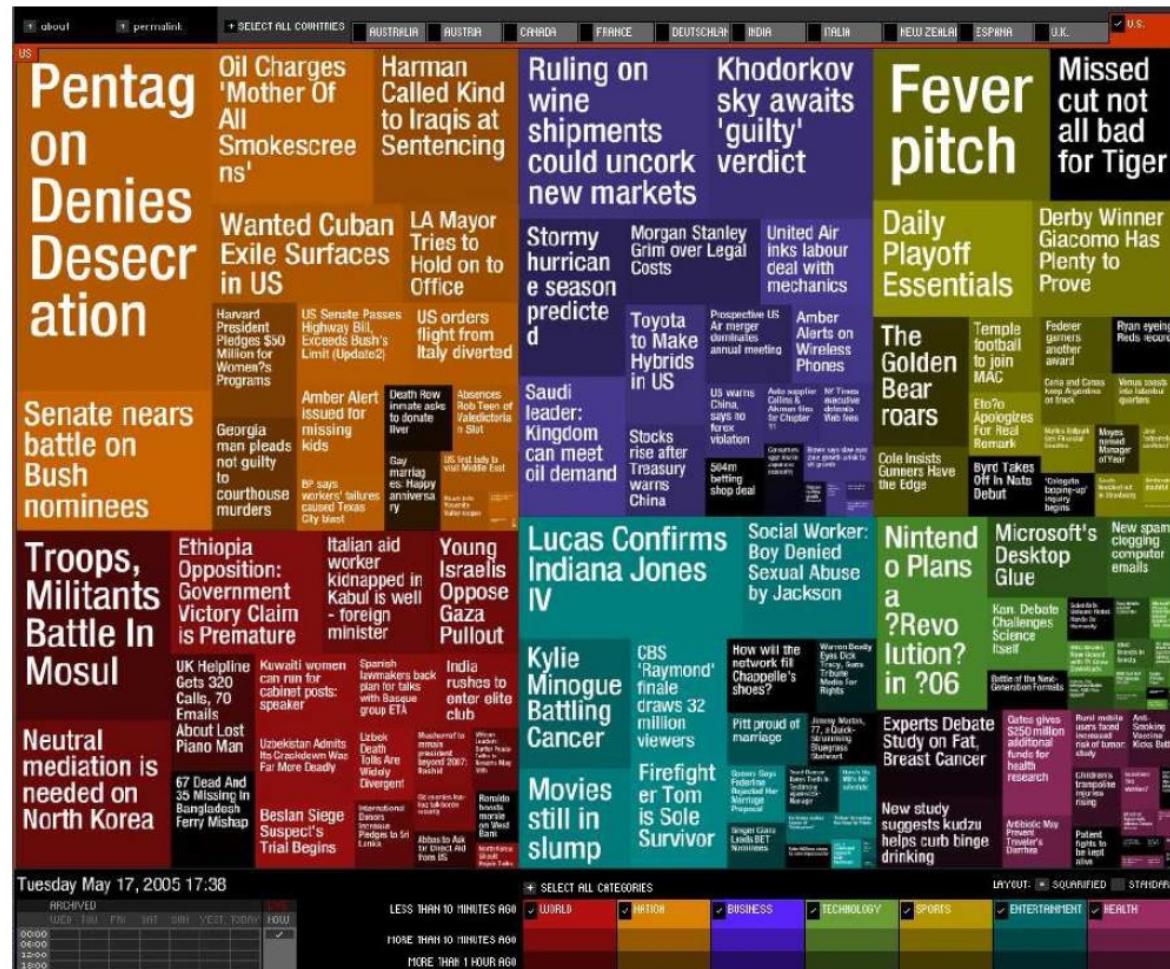


Schneiderman@UMD: Tree-Map of a File System



Schneiderman@UMD: Tree-Map to support
large data sets of a million items

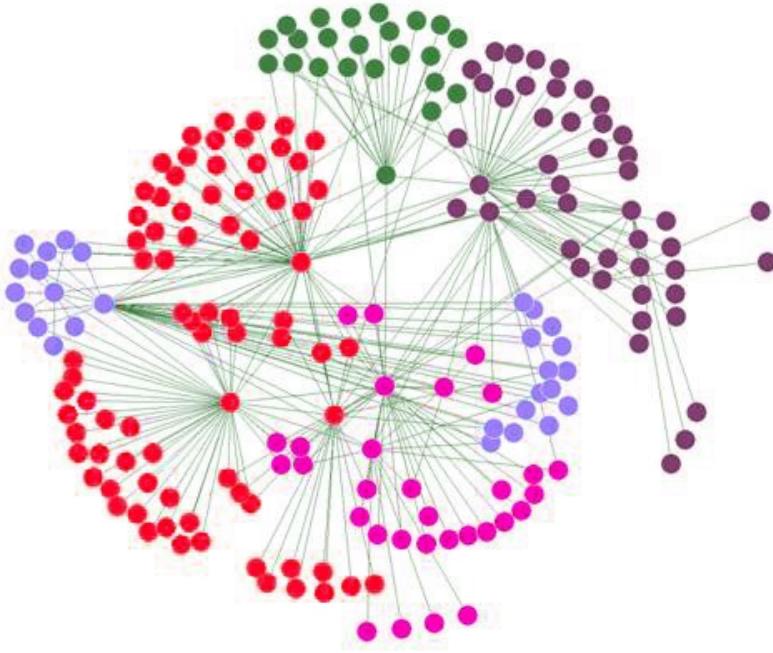
Newsmap: Tree-Maps for News



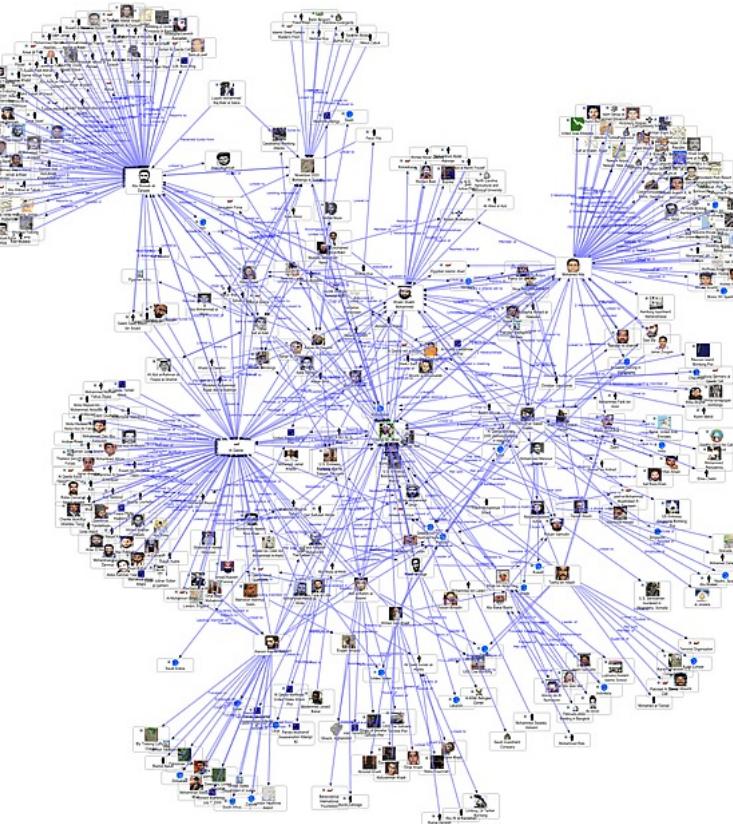
Google News Stories in 2005

Visualizing Complex Data and Relations: Social Networks

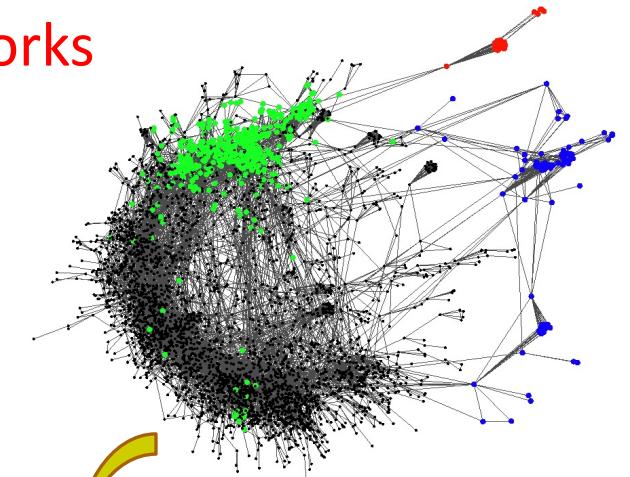
- Visualizing non-numerical data: social and information networks



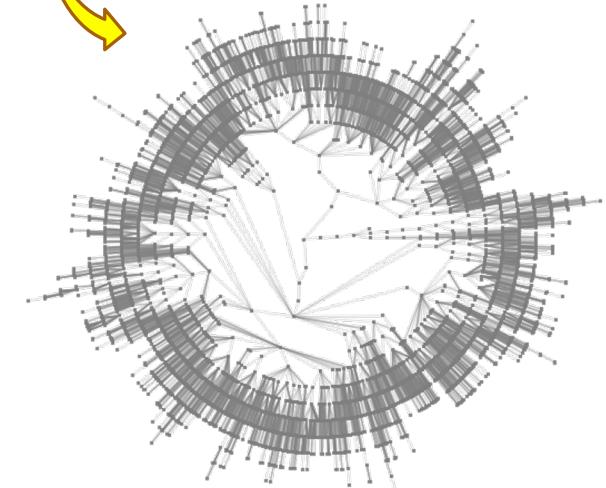
A typical network structure



A social network



organizing
information networks



Text Data: Tag Clouds



Chapter 2. Getting to Know Your Data

- Data Objects and Attribute Types
- Basic Statistical Descriptions of Data
- Data Visualization
- Measuring Data Similarity and Correlation
- Summary



Similarity, Dissimilarity, and Proximity

- **Similarity measure** or **similarity function**
 - A real-valued function that quantifies the similarity between two objects
 - Measure how two data objects are alike: The higher value, the more alike
 - Often falls in the range $[0,1]$: 0: no similarity; 1: completely similar
- **Dissimilarity** (or **distance**) **measure**
 - Numerical measure of how different two data objects are
 - In some sense, the inverse of similarity: The lower, the more alike
 - Minimum dissimilarity is often 0 (i.e., completely similar)
 - Range $[0, 1]$ or $[0, \infty)$, depending on the definition
- **Proximity** usually refers to either similarity or dissimilarity

Data Matrix and Dissimilarity Matrix

- Data matrix

- A data matrix of n data points with l dimensions

- Dissimilarity (distance) matrix

- n data points, but registers only the distance $d(i, j)$ (typically metric)

- Usually symmetric, thus a triangular matrix

- **Distance functions** are usually different for real, boolean, categorical, ordinal, ratio, and vector variables

- Weights can be associated with different variables based on applications and data semantics

$$D = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1l} \\ x_{21} & x_{22} & \dots & x_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nl} \end{pmatrix}$$

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

Proximity Measure for Categorical Attributes

- Categorical data, also called nominal attributes
 - Example: Color (red, yellow, blue, green), profession, etc.
- Method 1: Simple matching
 - m : # of matches, p : total # of variables

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

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

Proximity Measure for Binary Attributes

- A contingency table for binary data

		Object <i>j</i>		
		1	0	sum
Object <i>i</i>	1	q	r	$q + r$
	0	s	t	$s + t$
sum		$q + s$	$r + t$	p

- Distance measure for symmetric binary variables

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

- Distance measure for asymmetric binary variables:

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

- Jaccard coefficient (*similarity* measure for

asymmetric binary variables):

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

- Note: Jaccard coefficient is the same as

(a concept discussed in Pattern Discovery)

$$coherence(i, j) = \frac{sup(i, j)}{sup(i) + sup(j) - sup(i, j)} = \frac{q}{(q + r) + (q + s) - q}$$

Example: Dissimilarity between Asymmetric Binary Variables

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 (not counted in)
- The remaining attributes are asymmetric binary
- Let the values Y and P be 1, and the value N be 0
- Distance: $d(i, j) = \frac{r + s}{q + r + s}$

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

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

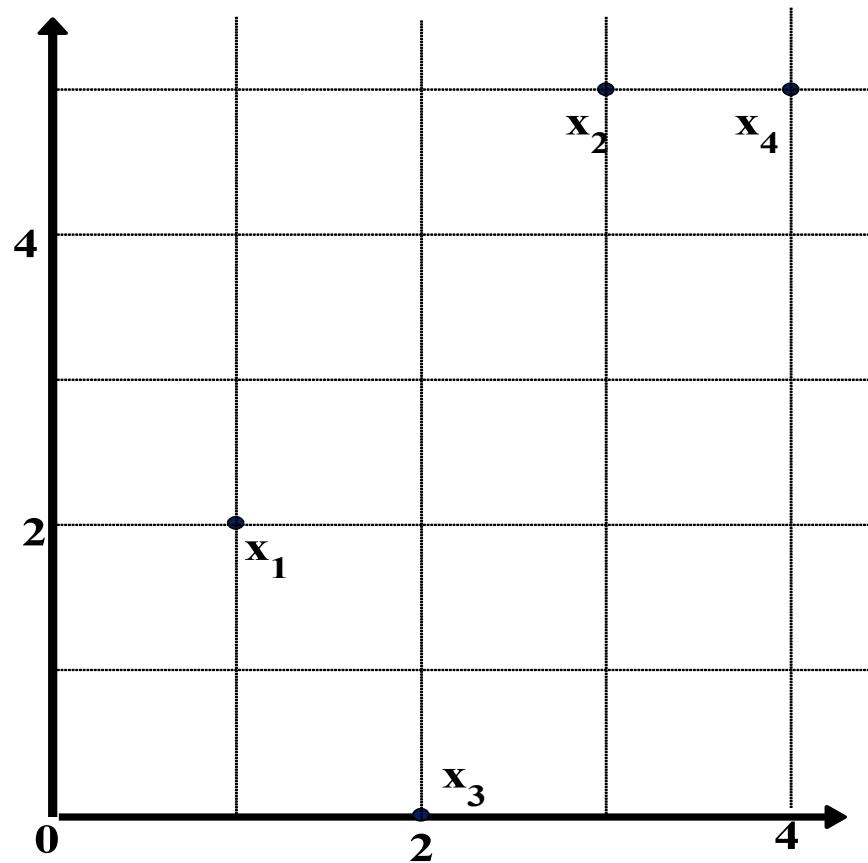
$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

		Mary		
		1	0	Σ_{row}
Jack		1	2	0
		0	1	3
Σ_{col}		3	3	6

		Jim		
		1	0	Σ_{row}
Jack		1	1	2
		0	1	3
Σ_{col}		2	4	6

		Mary		
		1	0	Σ_{row}
Jim		1	1	2
		0	2	4
Σ_{col}		3	3	6

Example: Data Matrix and Dissimilarity Matrix



Data Matrix

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

Dissimilarity Matrix (by Euclidean Distance)

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

Distance on Numeric Data: Minkowski Distance

- **Minkowski distance:** A popular distance measure

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

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

- Properties
 - $d(i, j) > 0$ if $i \neq j$, and $d(i, i) = 0$ (Positivity)
 - $d(i, j) = d(j, i)$ (Symmetry)
 - $d(i, j) \leq d(i, k) + d(k, j)$ (Triangle Inequality)
- A distance that satisfies these properties is a **metric**
- Note: There are nonmetric dissimilarities, e.g., set differences

Special Cases of Minkowski Distance

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

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

- $p = 2$: (L_2 norm) **Euclidean distance**

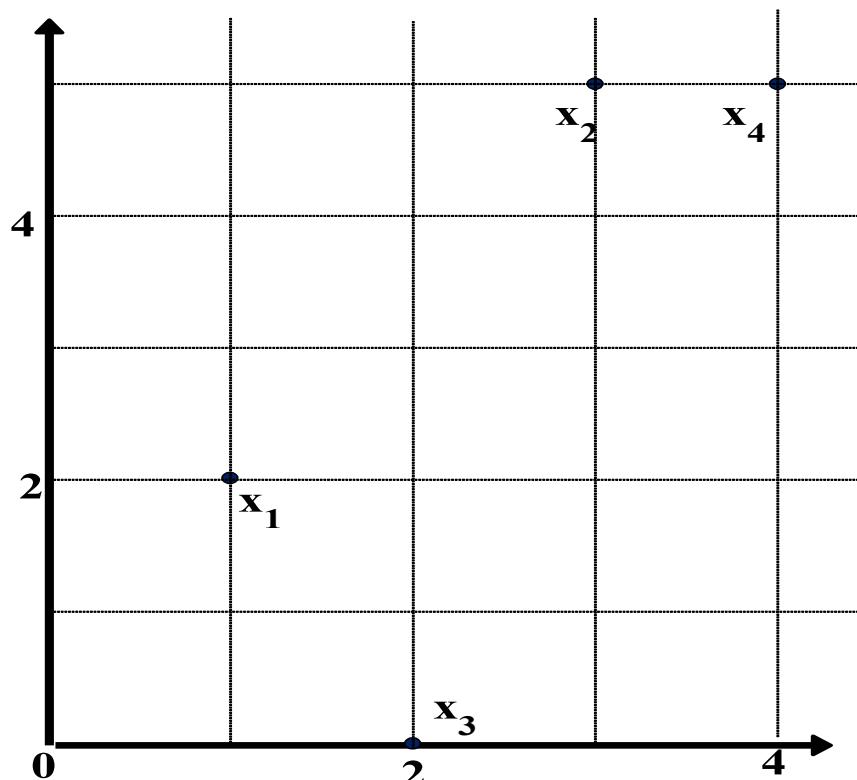
$$d(i, j) = \sqrt{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{il} - x_{jl}|^2}$$

- $p \rightarrow \infty$: (L_{\max} norm, L_∞ norm) **“supremum” distance**
 - The maximum difference between any component (attribute) of the vectors

$$d(i, j) = \lim_{p \rightarrow \infty} \sqrt[p]{|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \dots + |x_{il} - x_{jl}|^p} = \max_{f=1}^l |x_{if} - x_{jf}|$$

Example: Minkowski Distance at Special Cases

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



Manhattan (L_1)

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

Euclidean (L_2)

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

Supremum (L_∞)

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

Standardizing Numeric Data

- Z-score:

$$z = \frac{x - \mu}{\sigma}$$

- X: raw score to be standardized, μ : mean of the population, σ : standard deviation
- the distance between the raw score and the population mean in units of the standard deviation
- negative when the raw score is below the mean, “+” when above
- An alternative way: Calculate the mean absolute deviation

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

where

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

- standardized measure (z-score):
$$z_{if} = \frac{x_{if} - m_f}{s_f}$$
- Using mean absolute deviation is more robust than using standard deviation

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank (e.g., freshman, sophomore, junior, senior)
- Can be treated like interval-scaled
 - Replace *an ordinal variable value* by its 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
$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$
- Example: freshman: 0; sophomore: 1/3; junior: 2/3; senior 1
 - Then distance: $d(\text{freshman}, \text{senior}) = 1$, $d(\text{junior}, \text{senior}) = 1/3$
- Compute the dissimilarity using methods for interval-scaled variables

Attributes of Mixed Type

- A dataset may contain all attribute types
 - Nominal, symmetric binary, asymmetric binary, numeric, and ordinal
- One may use a weighted formula to combine their effects:

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

- If f is numeric: Use the normalized distance
- If f is binary or nominal: $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$; or $d_{ij}^{(f)} = 1$ otherwise
- If f is ordinal
 - Compute ranks z_{if} (where $z_{if} = \frac{r_{if} - 1}{M_f - 1}$)
 - Treat z_{if} as interval-scaled

Cosine Similarity of Two Vectors

- A **document** can be represented by a bag of terms or a long vector, with each attribute recording the *frequency* of a particular term (such as word, keyword, or phrase) in the document

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

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

$$\cos(d_1, d_2) = \frac{d_1 \bullet d_2}{\|d_1\| \times \|d_2\|}$$

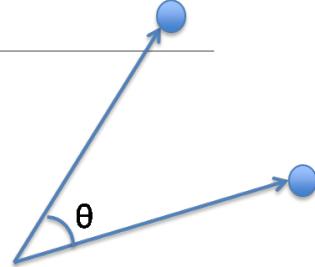
where \bullet indicates vector dot product, $\|d\|$: the length of vector d

Example: Calculating Cosine Similarity

- Calculating Cosine Similarity:

$$\cos(d_1, d_2) = \frac{d_1 \bullet d_2}{\|d_1\| \times \|d_2\|}$$

$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$



where • indicates vector dot product, ||d||: the length of vector d

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

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

- First, calculate vector dot product

$$d_1 \bullet d_2 = 5 \times 3 + 0 \times 0 + 3 \times 2 + 0 \times 0 + 2 \times 1 + 0 \times 1 + 0 \times 1 + 2 \times 1 + 0 \times 0 + 0 \times 1 = 25$$

- Then, calculate ||d₁|| and ||d₂||

$$\|d_1\| = \sqrt{5 \times 5 + 0 \times 0 + 3 \times 3 + 0 \times 0 + 2 \times 2 + 0 \times 0 + 0 \times 0 + 2 \times 2 + 0 \times 0 + 0 \times 0} = 6.481$$

$$\|d_2\| = \sqrt{3 \times 3 + 0 \times 0 + 2 \times 2 + 0 \times 0 + 1 \times 1 + 1 \times 1 + 0 \times 0 + 1 \times 1 + 0 \times 0 + 1 \times 1} = 4.12$$

- Calculate cosine similarity: $\cos(d_1, d_2) = 25 / (6.481 \times 4.12) = 0.94$

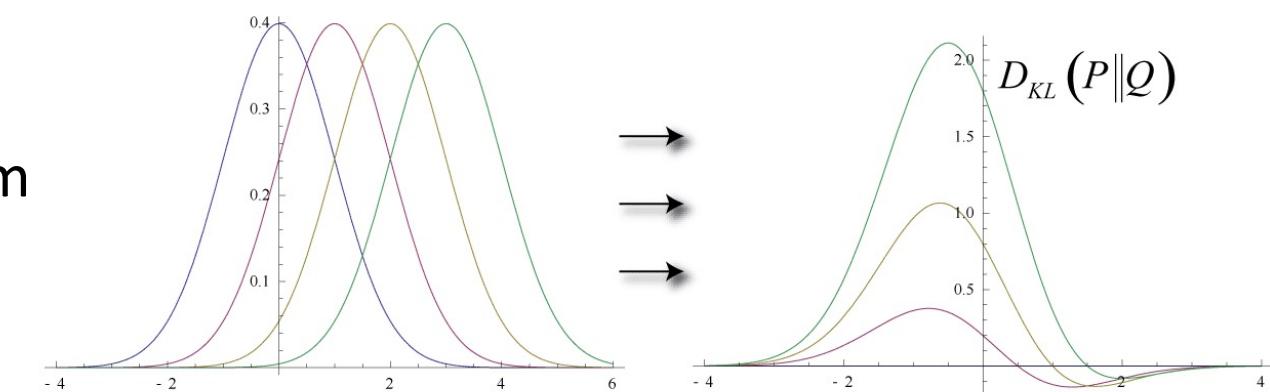
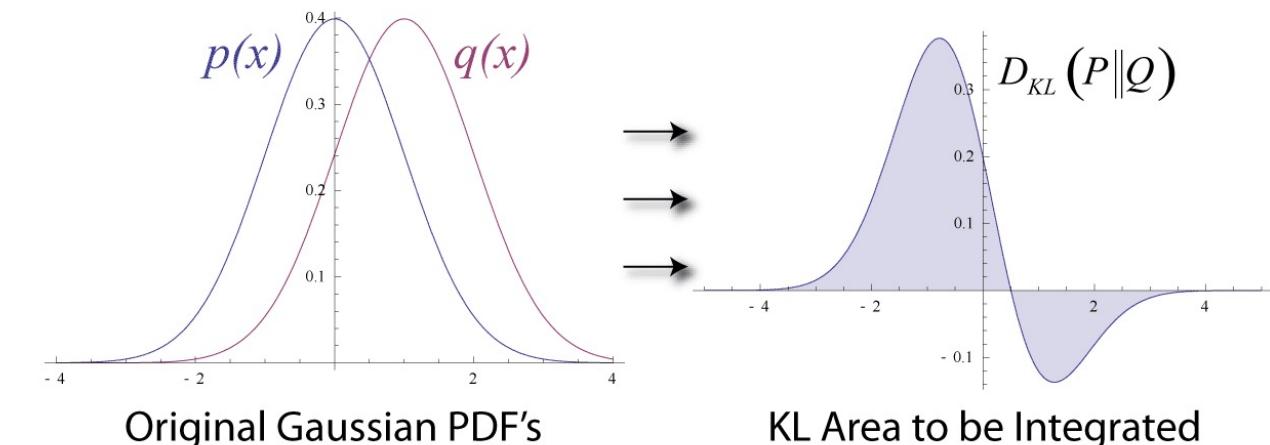
KL Divergence: Comparing Two Probability Distributions

- The Kullback-Leibler (KL) divergence:
Measure the difference between two probability distributions over the same variable x
- From information theory, closely related to *relative entropy*, *information divergence*, and *information for discrimination*
- $D_{KL}(p(x) || q(x))$: divergence of $q(x)$ from $p(x)$, measuring the information lost when $q(x)$ is used to approximate $p(x)$

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

Discrete form 

$$D_{KL}(p(x)||q(x)) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx$$



Ack.: Wikipedia entry: *The Kullback-Leibler (KL) divergence*

Continuous form 

More on KL Divergence

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

- The KL divergence measures the expected number of extra bits required to code samples from $p(x)$ ("true" distribution) when using a code based on $q(x)$, which represents a theory, model, description, or approximation of $p(x)$

$$\begin{aligned} D_{KL}(p(x)\|q(x)) &= E_{p(x)} \left[\log \frac{p(x)}{q(x)} \right] \\ &= E_{p(x)} \left[\log \frac{1}{q(x)} \right] - E_{p(x)} \left[\log \frac{1}{p(x)} \right] \end{aligned}$$

- The KL divergence is not a distance measure, not a metric: asymmetric, not satisfy triangular inequality ($D_{KL}(P\|Q)$ does not equal $D_{KL}(Q\|P)$)
- In applications, P typically represents the "true" distribution of data, observations, or a precisely calculated theoretical distribution, while Q typically represents a theory, model, description, or approximation of P .

More on KL Divergence

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

- The Kullback–Leibler divergence from Q to P , denoted $D_{\text{KL}}(P||Q)$, is a measure of the information gained when one revises one's beliefs from the prior probability distribution Q to the posterior probability distribution P . In other words, it is the amount of information lost when Q is used to approximate P .

- The KL divergence is sometimes also called the information gain achieved if P is used instead of Q . It is also called the relative entropy of P with respect to Q .

Computing the KL Divergence

- Base on the formula, $D_{KL}(P, Q) \geq 0$ and $D_{KL}(P || Q) = 0$ if and only if $P = Q$
- How about when $p = 0$ or $q = 0$?
 - $\lim_{p \rightarrow 0} p \log p = 0$
 - when $p \neq 0$ but $q = 0$, $D_{KL}(p || q)$ is defined as ∞ , i.e., if one event e is possible (i.e., $p(e) > 0$), and the other predicts it is absolutely impossible (i.e., $q(e) = 0$), then the two distributions are absolutely different
- However, in practice, P and Q are derived from frequency distributions, not counting the possibility of unseen events. Thus *smoothing* is needed
- Example: $P : (a : 3/5, b : 1/5, c : 1/5)$. $Q : (a : 5/9, b : 3/9, d : 1/9)$
 - need to introduce a small constant ϵ , e.g., $\epsilon = 10^{-3}$
 - The sample set observed in P , $SP = \{a, b, c\}$, $SQ = \{a, b, d\}$, $SU = \{a, b, c, d\}$
 - Smoothing, add missing symbols to each distribution, with probability ϵ
 - $P' : (a : 3/5 - \epsilon/3, b : 1/5 - \epsilon/3, c : 1/5 - \epsilon/3, d : \epsilon)$
 - $Q' : (a : 5/9 - \epsilon/3, b : 3/9 - \epsilon/3, c : \epsilon, d : 1/9 - \epsilon/3)$
 - $D_{KL}(P' || Q')$ can then be computed easily

$$D_{KL}(p(x) || q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

Chapter 2. Getting to Know Your Data

- Data Objects and Attribute Types
- Basic Statistical Descriptions of Data
- Data Visualization
- Measuring Data Similarity and Correlation
- Summary



Summary

- Data attribute types: nominal, binary, ordinal, interval-scaled, ratio-scaled
- Many types of data sets, e.g., numerical, text, graph, Web, image.
- Gain insight into the data by:
 - Basic statistical data description: central tendency, dispersion, graphical displays
 - Data visualization: map data onto graphical primitives
 - Measure data similarity and correlation
- Above steps are the beginning of data preprocessing
- Many methods have been developed but still an active area of research

References

- W. Cleveland, Visualizing Data, Hobart Press, 1993
- T. Dasu and T. Johnson. Exploratory Data Mining and Data Cleaning. John Wiley, 2003
- U. Fayyad, G. Grinstein, and A. Wierse. Information Visualization in Data Mining and Knowledge Discovery, Morgan Kaufmann, 2001
- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.
- H. V. Jagadish et al., Special Issue on Data Reduction Techniques. Bulletin of the Tech. Committee on Data Eng., 20(4), Dec. 1997
- D. A. Keim. Information visualization and visual data mining, IEEE trans. on Visualization and Computer Graphics, 8(1), 2002
- D. Pyle. Data Preparation for Data Mining. Morgan Kaufmann, 1999
- S. Santini and R. Jain," Similarity measures", IEEE Trans. on Pattern Analysis and Machine Intelligence, 21(9), 1999
- E. R. Tufte. The Visual Display of Quantitative Information, 2nd ed., Graphics Press, 2001
- C. Yu, et al., Visual data mining of multimedia data for social and behavioral studies, Information Visualization, 8(1), 2009





CS 412 Intro. to Data Mining

Chapter 3. Data Preparation

Arindam Banerjee, Computer Science, UIUC, Fall 2021



Chapter 3: Data Preparation

- Data Preprocessing: An Overview

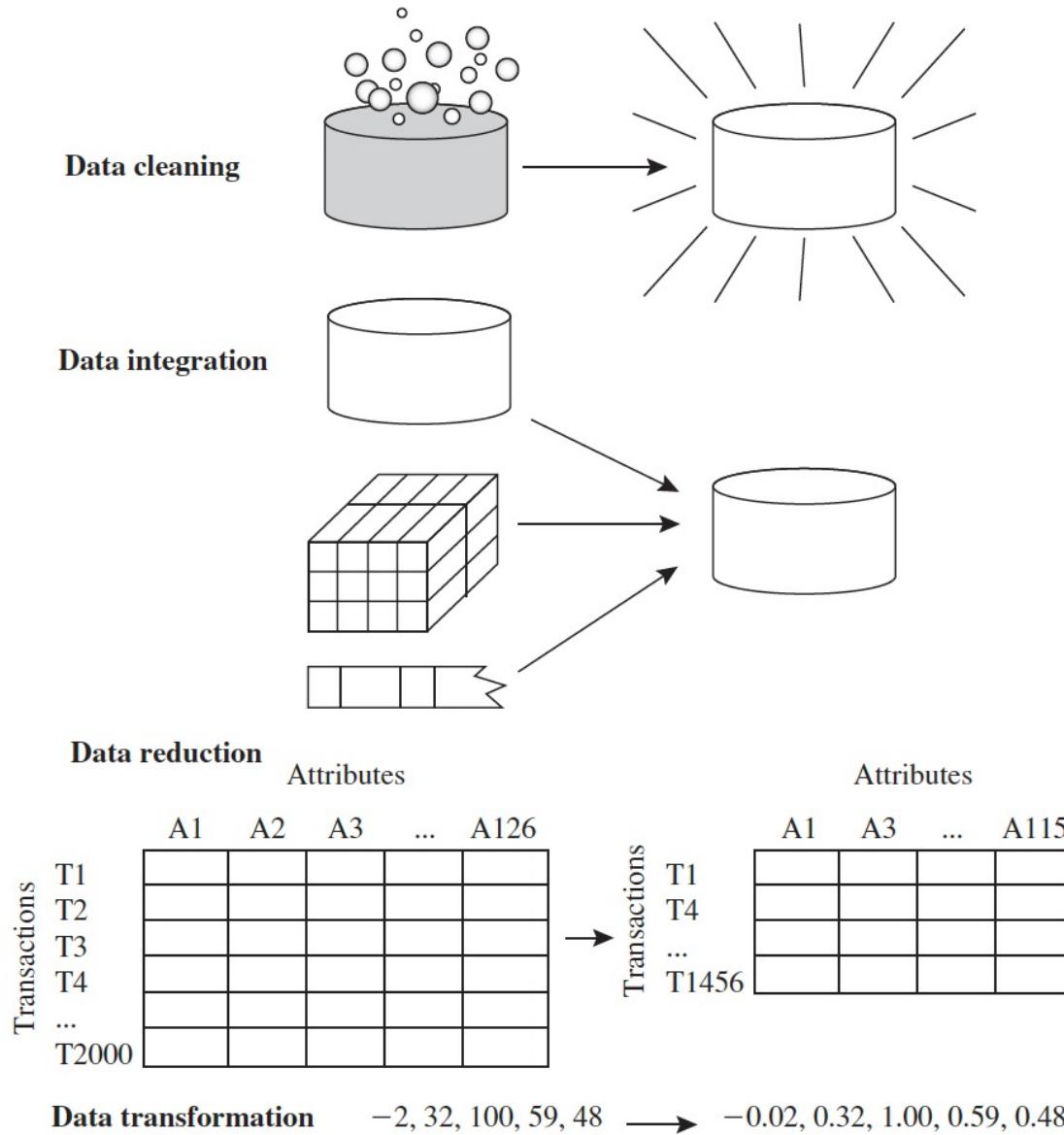


- Data Cleaning
- Data Integration
- Data Reduction and Transformation
- Dimensionality Reduction
- Summary

What is Data Preprocessing? – Major Tasks

- **Data cleaning**
 - Handle missing data, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- **Data integration**
 - Integration of multiple databases, data cubes, or files
- **Data reduction**
 - Dimensionality reduction
 - Numerosity reduction
 - Data compression
- **Data transformation and data discretization**
 - Normalization
 - Concept hierarchy generation

Forms of Data Preprocessing



Why Preprocess the Data? – Data Quality Issues

- Measures for data quality: A multidimensional view
 - Accuracy: correct or wrong, accurate or not
 - Completeness: not recorded, unavailable, ...
 - Consistency: some modified but some not, dangling, ...
 - Timeliness: timely update?
 - Believability: how trustworthy that the data are correct?
 - Interpretability: how easily can the data be understood?

Chapter 3: Data Preprocessing

- ❑ Data Preprocessing: An Overview

- ❑ Data Cleaning



- ❑ Data Integration

- ❑ Data Reduction and Transformation

- ❑ Dimensionality Reduction

- ❑ Summary

Data Cleaning

- ❑ Data in the Real World Is Dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, and transmission error
 - ❑ Incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - ❑ e.g., *Occupation* = “ ” (missing data)
 - ❑ Noisy: containing noise, errors, or outliers
 - ❑ e.g., *Salary* = “-10” (an error)
 - ❑ Inconsistent: containing discrepancies in codes or names, e.g.,
 - ❑ *Age* = “42”, *Birthday* = “03/07/2010”
 - ❑ Was rating “1, 2, 3”, now rating “A, B, C”
 - ❑ discrepancy between duplicate records
 - ❑ Intentional (e.g., *disguised missing data*)
 - ❑ Jan. 1 as everyone’s birthday?

Incomplete (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 were not entered due to misunderstanding
 - ❑ Certain data may not be considered important at the time of entry
 - ❑ Did 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 (when doing classification)—not effective when the % 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 be due to
 - Faulty data collection instruments
 - Data entry problems
 - Data transmission problems
 - Technology limitation
 - Inconsistency in naming convention
- **Other data problems**
 - 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
- Semi-supervised: Combined computer and human inspection
 - Detect suspicious values and check by human (e.g., deal with possible outliers)

Binning for Data Smoothing

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

Partition into (equal-frequency) bins:

Bin 1: 4, 8, 15

Bin 2: 21, 21, 24

Bin 3: 25, 28, 34

Smoothing by bin means:

Bin 1: 9, 9, 9

Bin 2: 22, 22, 22

Bin 3: 29, 29, 29

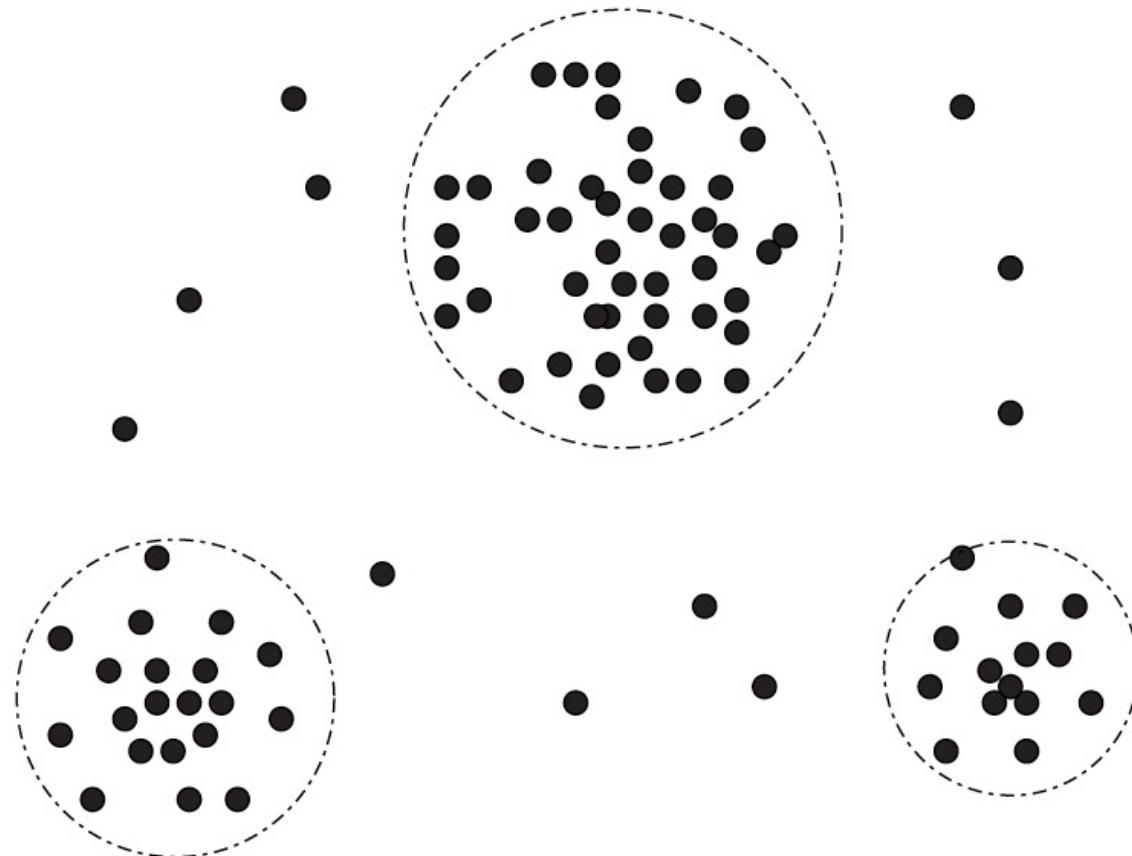
Smoothing by bin boundaries:

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 34

Clustering for Outlier Detection



Data Cleaning as a Process

- ❑ **Data discrepancy detection**
 - ❑ Use metadata (e.g., domain, range, dependency, distribution)
 - ❑ Check field overloading
 - ❑ Check uniqueness rule, consecutive rule, and null rule
 - ❑ Use commercial tools
 - ❑ Data scrubbing: use simple domain knowledge (e.g., postal code, spell-check) to detect errors and make corrections
 - ❑ Data auditing: by analyzing data to discover rules and relationship to detect violators (e.g., correlation and clustering to find outliers)
- ❑ **Data migration and integration**
 - ❑ Data migration tools: allow transformations to be specified
 - ❑ ETL (Extraction/Transformation>Loading) tools: allow users to specify transformations through a graphical user interface
- ❑ Integration of the two processes
 - ❑ Iterative and interactive (e.g., Potter's Wheels)

Chapter 3: Data Preprocessing

- ❑ Data Preprocessing: An Overview

- ❑ Data Cleaning

- ❑ Data Integration



- ❑ Data Reduction and Transformation

- ❑ Dimensionality Reduction

- ❑ Summary

Data Integration

- ❑ Data integration
 - ❑ Combining data from multiple sources into a coherent store
- ❑ Why data integration?
 - ❑ Help reduce/avoid noise
 - ❑ Get a more complete picture
 - ❑ Improve mining speed and quality
- ❑ Schema integration:
 - ❑ e.g., A.cust-id ≡ B.cust-#
 - ❑ Integrate metadata from different sources
- ❑ Entity identification:
 - ❑ Identify real world entities from multiple data sources, e.g., Bill Clinton = William Clinton

Handling Noise in Data Integration

- Detecting data value conflicts
 - For the same real world entity, attribute values from different sources are different
 - Possible reasons: no reason, different representations, different scales, e.g., metric vs. British units
- Resolving conflict information
 - Take the mean/median/mode/max/min
 - Take the most recent
 - Truth finding: consider the source quality
- Data cleaning + data integration

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
- What’s the problem?
 - $$Y = 2X \rightarrow Y = X_1 + X_2 \quad Y = 3X_1 - X_2 \quad Y = -1291X_1 + 1293X_2$$
- Redundant attributes may be detected by correlation analysis and covariance analysis

Example: stock market

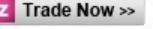
Yahoo! Finance		Day's Range: 93.80-95.71	Nasdaq
Green Mountain Coffee Roasters, (NasdaqGS: GMCR)	After Hours: 95.13 -0.01 (-0.02%) 4:07PM EDT	Last Trade: 95.14	Last Sale \$ 95.14
Trade Time: 4:00PM EDT	Change Net / % 1.69 ▲ 1.81%	Day's Range: 93.80 - 95.71	Change Net / % 1.69 ▲ 1.81%
Change: ↑ 1.69 (1.81%)	Best Bid / Ask \$ 95.03 / \$ 95.94	52wk Range: 25.38 - 95.71	Best Bid / Ask \$ 95.03 / \$ 95.94
Prev Close: 93.45	1y Target Est: \$ 95.00	Volume: 2,384,075	1y Target Est: \$ 95.00
Open: 94.01	Today's High / Low \$ 95.71 / \$ 93.80	Avg Vol (3m): 2,512,070	Today's High / Low \$ 95.71 / \$ 93.80
Bid: 95.03 x 100	Share Volume 2,384,175	Market Cap: 13.51B	Share Volume 2,384,175
Ask: 95.94 x 100	50 Day Avg. Daily Volume 2,751,062	P/E (ttm): 119.82	50 Day Avg. Daily Volume 2,751,062
1y Target Est:	Previous Close \$ 93.45	EPS (ttm): 0.79	Previous Close \$ 93.45
52wk Range: 25.38-95.71	52 Wk High / Low \$ 93.72 / \$ 25.38	N/A (N/A)	52 Wk High / Low \$ 93.72 / \$ 25.38
52 Wk: 25.38-93.72	Shares Outstanding 152,785,000		Shares Outstanding 152,785,000
	Market Value of Listed Security \$ 14,535,964,900		Market Value of Listed Security \$ 14,535,964,900
	P/E Ratio 120.43		P/E Ratio 120.43
	Forward P/E Ratio 63.57		Forward P/E Ratio 63.57
	Earnings Per Share \$ 0.79		Earnings Per Share \$ 0.79
	Annualized Dividend N/A		Annualized Dividend N/A
	Ex Dividend Date N/A		Ex Dividend Date N/A
	Dividend Payment Date N/A		Dividend Payment Date N/A
	Current Yield N/A		Current Yield N/A
	Beta 0.82		Beta 0.82
	NASDAQ Official Open Price: \$ 94.01		NASDAQ Official Open Price: \$ 94.01
	Date of NASDAQ Official Open Price: Jul. 7, 2011		Date of NASDAQ Official Open Price: Jul. 7, 2011
	NASDAQ Official Close Price: \$ 95.14		NASDAQ Official Close Price: \$ 95.14
	Date of NASDAQ Official Close Price: Jul. 7, 2011		Date of NASDAQ Official Close Price: Jul. 7, 2011

Example: stock market

SALVEPAR (SY)  Search InvestCenter ▶

Recent Quotes ▶ My Watchlist ▶ Top Indices ▶

SALVEPAR

 -0.8900 (-1.212%) at 72.55 EUR
70 in Volume

Add to: My Watchlist

Data as of 04:18 AM EDT Jul 7, 2011

Quote News Profile Research Community

SY 64.98 +0.00 (0.00%) Click Here to Receive Instant E-mail and RSS Alerts

Trade SY now with \$3.95 STOCK TRADES

SALVEPAR (SY)



©FinancialContent.com

Stock Details

Last Trade:	64.98
Change:	+0.00 (0.00%)
Prev Close:	64.98
Open:	14.73
Days Range:	64.98 – 64.98
52 Week Range:	33.54 - 66.00
Volume:	88168
P/E:	31.54
EPS:	2.06

SYBASE (SY)

SOURCE: NYSE

As of July 29, 2010 4:04 pm. Quotes are delayed by at least 15 minutes

+0.01

\$64.98  209,960 \$64.97

Last Trade +0.02% Volume Prev. Close

Change (%)

Last Trade: 64.98

Change: +0.00 (0.00%)

Prev Close: 64.98

Open: 14.73

Days Range: 64.98 – 64.98

52 Week Range: 33.54 - 66.00

Volume: 88168

P/E: 31.54

EPS: 2.06

Example: stock market

NASDAQ One-click options strategies on Trad
Trade free for 60 days + get up to \$600 cash.

QUICK FIND: ETFs | Tools | After Hours | Global Indices | Earn a Degree | Company List

Home ▾ Quotes & Research ▾ Extended Trading ▾ Market Activity ▾ News ▾

[add symbol](#) [edit symbol list](#) [symbol lookup](#)

Symbol List Views FlashQuotes InfoQuotes Stock Details Real-Time Quotes Summary Quotes After Hours Quotes Pre-market Quotes Historical Quotes Options Chain CHARTS Basic Charts Interactive Charts COMPANY NEWS Company Headlines Press Releases Sentiment STOCK ANALYSIS Analyst Research Guru Analysis

Home > Quotes > Stock Quote > TTI

TD Ameritrade Trade Free for 60 days + Get up to \$600 with Trade Architect from TD Ameritrade

TTI Save my stocks for next time Investor Tools Tracking T

⚠ Cookies disabled? Please note that beginning 5/13/2011, you must have cookies. Please contact jfeedback@nasdaq.com with any questions or concerns.

TTI: Stock Quote & Summary Data

\$ 13.11 \$ 0.51 ▲ 4.05%
Jul. 7, 2011 Market Closed
Update Quotes: On Updates every 7 Seconds.

for TTI Commentary for TTI Price Charts Company Financials

Last Sale	\$ 13.11
Change Net / %	+ 4.05%
1y Target Est:	\$ 16.00
Today's High / Low	\$ 13.11 / \$ 12.67
Share Volume	480,067
Previous Close	\$ 12.60
52 Wk High / Low	\$ 16 / \$ 8
Shares Outstanding	76,821,000
Market Value of Listed Security	\$ 1,007,123,310
P/E Ratio	NE
Forward P/E (1yr)	19.69
Earnings Per Share	\$ -0.68
Annualized Dividend	\$ 0.00

UPDOWN Beat the market. Earn real money. Zero risk.

HOME TRADING STOCKS COMMUNITY CO

Overview Market News Top Stock Picks GET QUOTE Sponsore

TETRA TECHNOLOGIES (TTI) 1

76.82B

Trade T

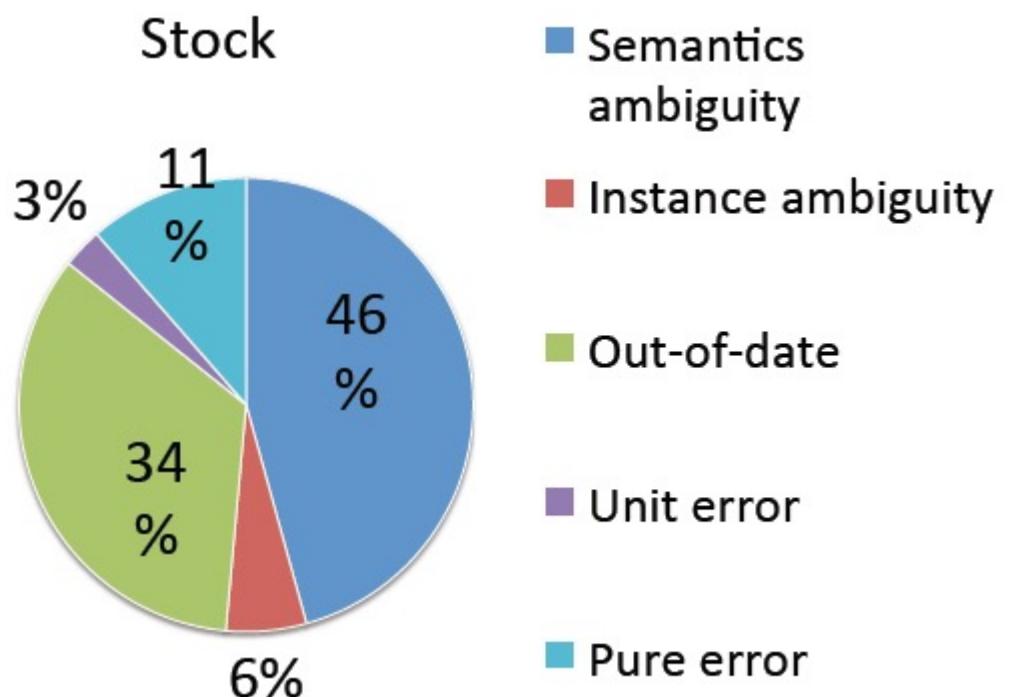
Overview Trade TTI Stock Picks Tweets

TTI \$ 13.11 \$ 0.51 (4.05%)

You need to update your Flash Player

Today	5d	1m	3m	1y	5y	10y
Last:	\$ 13.11			High:	\$ 13.15	
Prev Close:	\$ 12.60			Low:	\$ 12.67	
Open:	\$ 12.82			Mkt Cap:	\$ 968M	
Change:	\$ 0.51 (4.05%)			52Wk High:	\$ 16.00	
Vol:	472,608			52Wk Low:	\$ 8.00	
Avg Volume:	559,308			Shares:	76.82B	
EPS:	-			PE Ratio:	-	

Example: stock market



Source	Accuracy	Coverage
<i>Google Finance</i>	.94	.82
<i>Yahoo! Finance</i>	.93	.81
<i>NASDAQ</i>	.92	.84
<i>MSN Money</i>	.91	.89
<i>Bloomberg</i>	.83	.81

Xian Li, Xin Luna Dong, Kenneth Lyons, Weiyi Meng, and Divesh Srivastava. Truth finding on the Deep Web: Is the problem solved? In *VLDB*, 2013.

Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
- Data Cleaning
- Data Integration
- Data Reduction and Transformation
- Dimensionality Reduction
- Summary

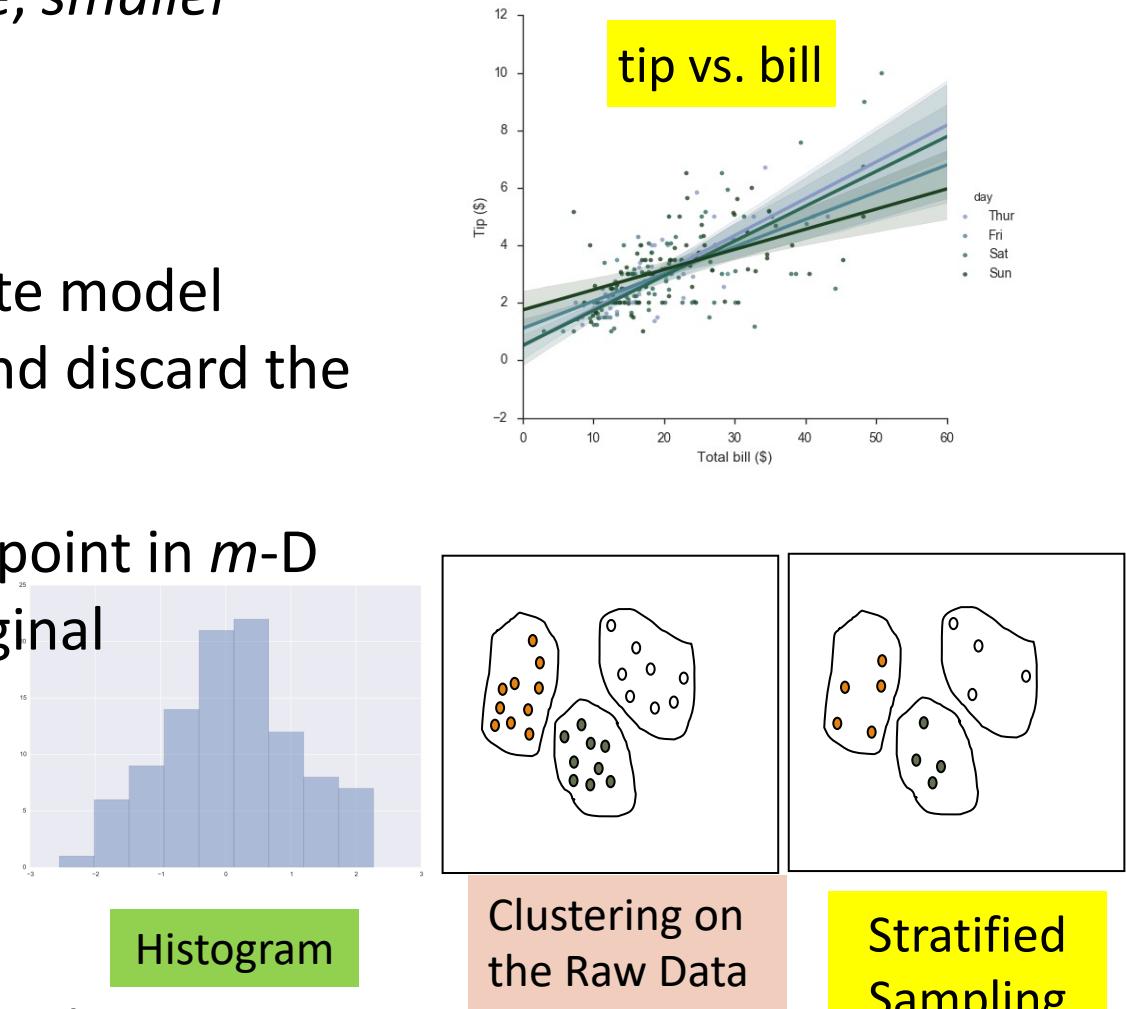


Data Reduction

- **Data reduction:**
 - Obtain a reduced representation of the data set
 - much smaller in volume but yet produces *almost* the same analytical results
 - Why data reduction?—A database/data warehouse may store terabytes of data
 - Complex analysis may take a very long time to run on the complete data set
- **Methods for data reduction** (also *data size reduction* or *numerosity reduction*)
 - Regression and Log-Linear Models
 - Histograms, clustering, sampling
 - Data cube aggregation
 - Data compression

Data Reduction: Parametric vs. Non-Parametric Methods

- Reduce data volume by choosing alternative, *smaller forms* of data representation
- **Parametric methods** (e.g., regression)
 - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
 - Ex.: 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, ...

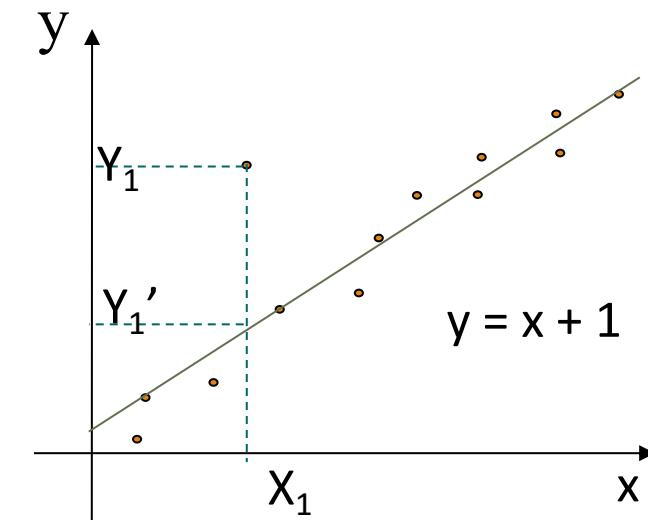


Parametric Data Reduction: Regression Analysis

- Regression analysis: A collective name for techniques for the modeling and analysis of numerical data consisting of values of a **dependent variable** (also called **response variable** or **measurement**) and of one or more **independent variables** (also known as **explanatory variables** or **predictors**), e.g.,

$$y = (wx + b) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

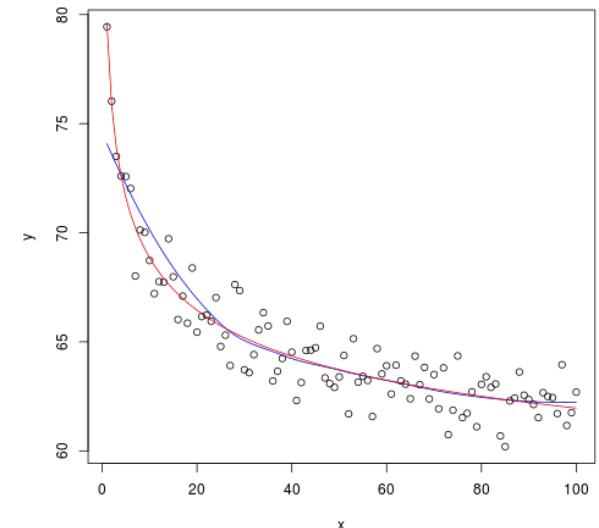
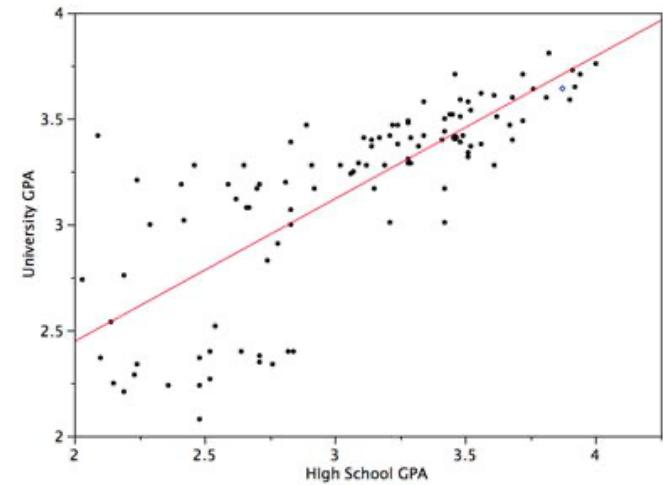
- The parameters are estimated so as to give a "best fit" of the data
- Most commonly the best fit is evaluated by using the **least squares method**, but other criteria have also been used



- Used for prediction (including forecasting of time-series data), inference, hypothesis testing, and modeling of causal relationships

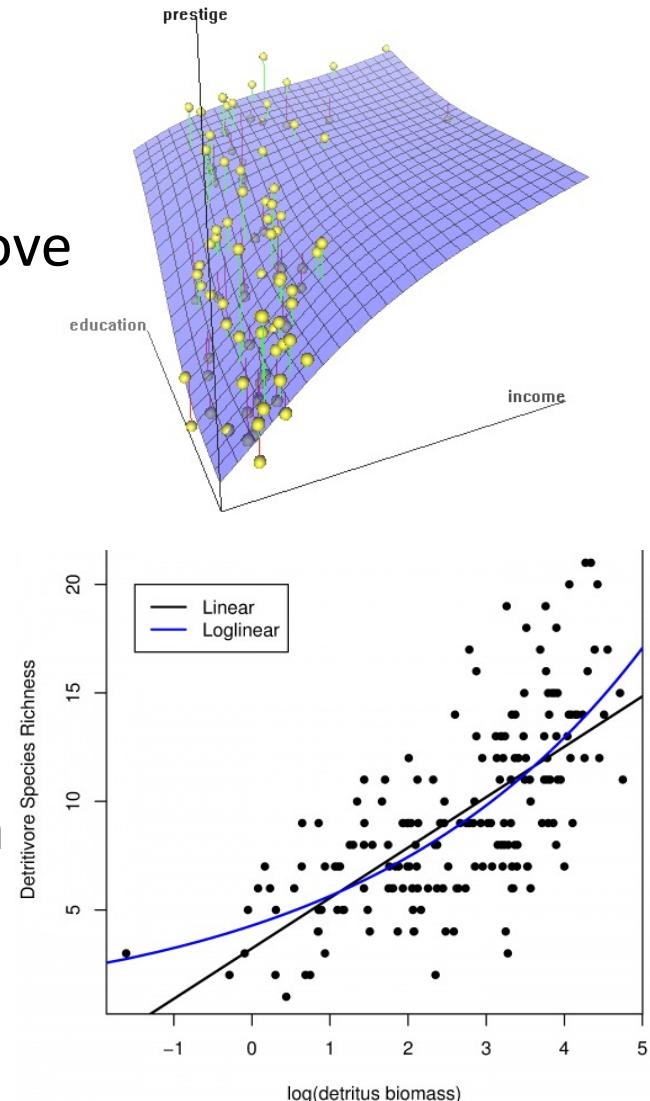
Linear and Multiple Regression

- Linear regression: $Y = w X + b$
 - Data modeled to fit a straight line
 - Often uses the least-square method to fit the line
 - 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 $Y_1, Y_2, \dots, X_1, X_2, \dots$
- Nonlinear regression:
 - Data are modeled by a function which is a nonlinear combination of the model parameters and depends on one or more independent variables
 - The data are fitted by a method of successive approximations



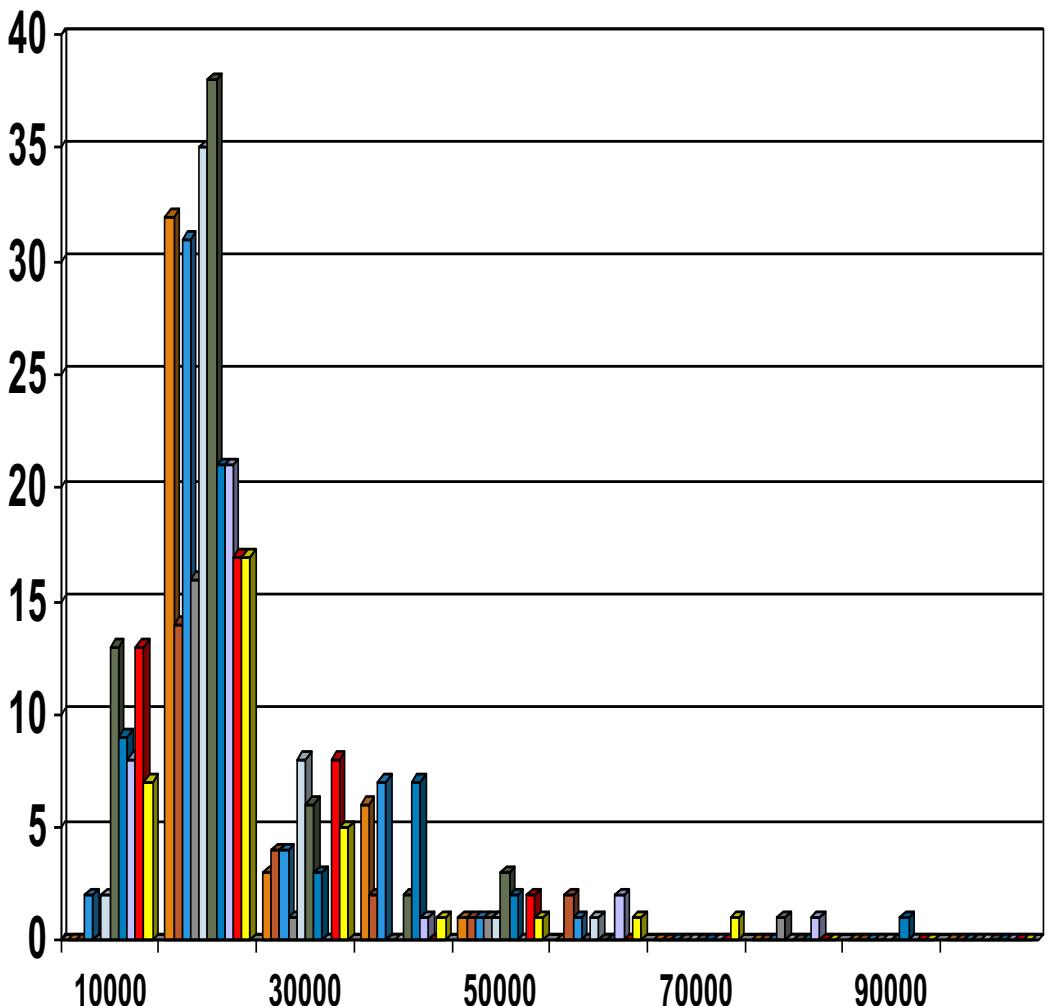
Multiple Regression and Log-Linear Models

- Multiple regression: $Y = b_0 + b_1 X_1 + b_2 X_2$
 - Allows a response variable Y to be modeled as a linear function of multidimensional feature vector
 - Many nonlinear functions can be transformed into the above
- Log-linear model:
 - A math model that takes the form of a function whose logarithm is a linear combination of the parameters of the model, which makes it possible to apply (possibly multivariate) linear regression
 - Estimate the probability of each point (tuple) in a multi-dimen. space for a set of discretized attributes, based on a smaller subset of dimensional combinations
 - Useful for dimensionality reduction and data smoothing



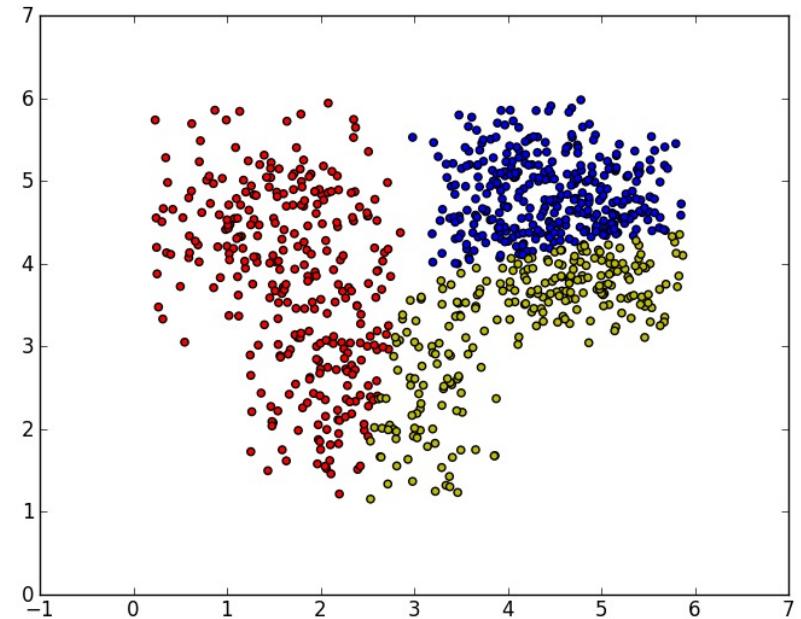
Histogram Analysis

- Divide data into buckets and store average (sum) for each bucket
- Partitioning rules:
 - Equal-width: equal bucket range
 - Equal-frequency (or equal-depth)



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 10

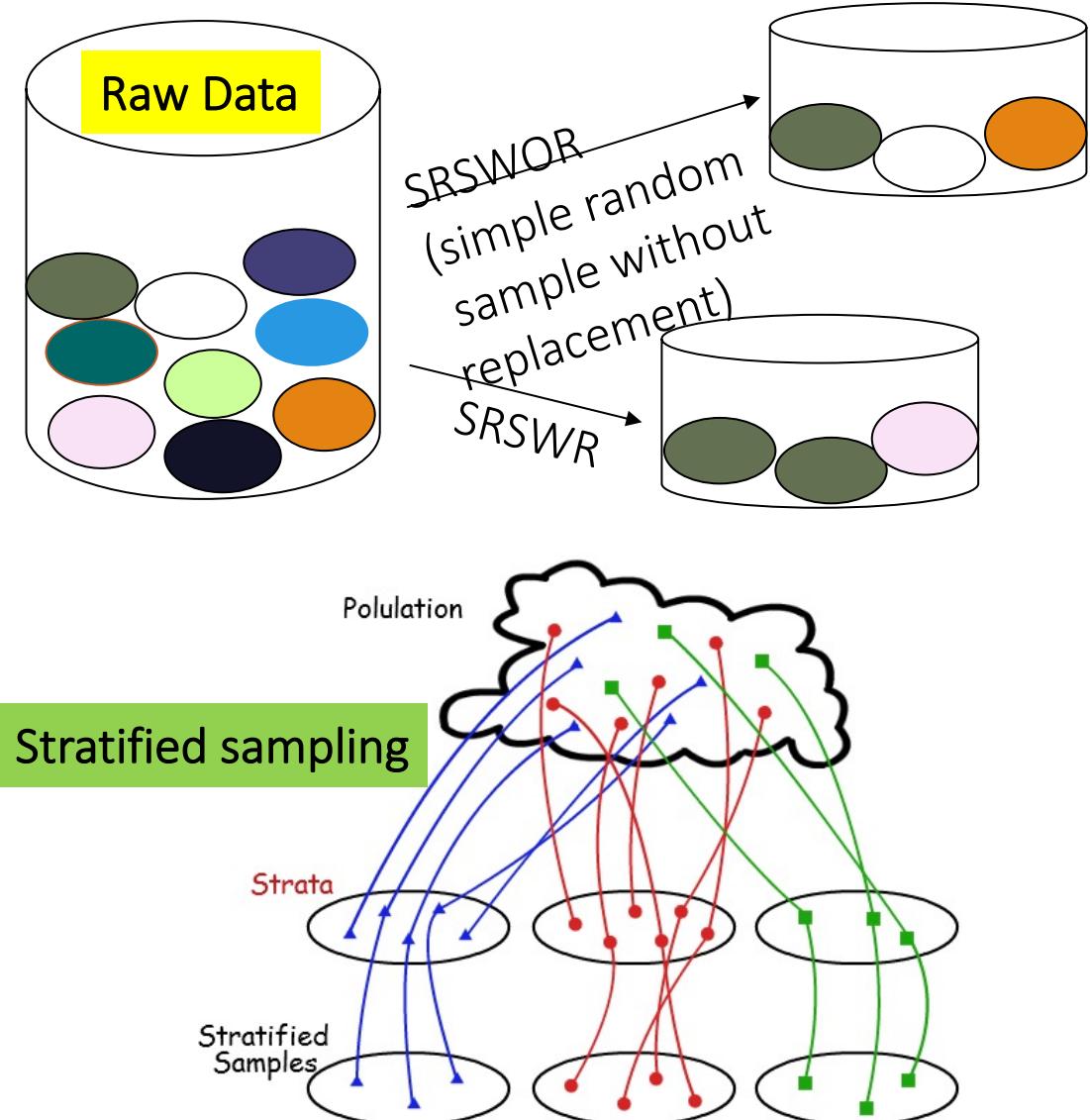


Sampling

- 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
- Key principle: Choose a **representative** subset of the data
 - Simple random sampling may have very poor performance in the presence of skew
 - Develop adaptive sampling methods, e.g., stratified sampling:
- Note: Sampling may not reduce database I/Os (page at a time)

Types of Sampling

- **Simple random sampling:** equal probability of selecting any particular item
- **Sampling without replacement**
 - Once an object is selected, it is removed from the population
- **Sampling with replacement**
 - A selected object is not removed from the population
- **Stratified sampling**
 - Partition (or cluster) the data set, and draw samples from each partition (proportionally, i.e., approximately the same percentage of the data)



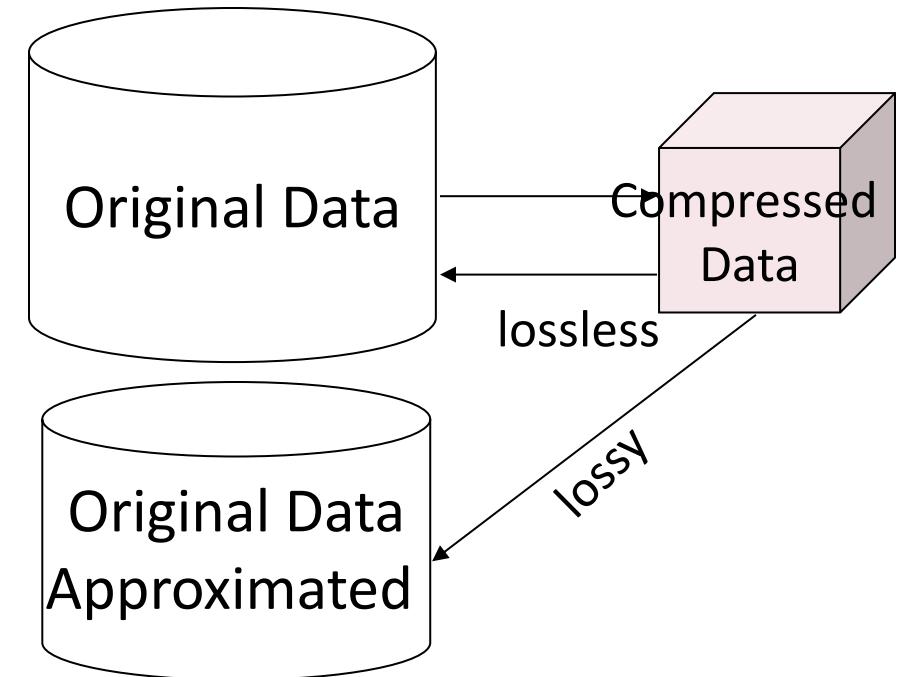
Data Cube Aggregation

- ❑ 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



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 reduction and dimensionality reduction may also be considered as forms of data compression



Lossy vs. lossless compression

Data Transformation

- ❑ A function that maps the entire set of values of a given attribute to a new set of replacement values s.t. each old value can be identified with one of the new values
- ❑ Methods
 - ❑ Smoothing: Remove noise from data
 - ❑ Attribute/feature construction
 - ❑ New attributes constructed from the given ones
 - ❑ Aggregation: Summarization, data cube construction
 - ❑ Normalization: Scaled to fall within a smaller, specified range
 - ❑ min-max normalization
 - ❑ z-score normalization
 - ❑ normalization by decimal scaling
 - ❑ Discretization: Concept hierarchy climbing

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):

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

Z-score: The distance between the raw score and the population mean in the unit of the standard deviation

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

- **Normalization by decimal scaling**

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

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
 - ❑ Numeric—real numbers, e.g., integer or real numbers
- ❑ Discretization: Divide the range of a continuous attribute into intervals
 - ❑ Interval labels can then be used to replace actual data values
 - ❑ Reduce data size by discretization
 - ❑ Supervised vs. unsupervised
 - ❑ Split (top-down) vs. merge (bottom-up)
 - ❑ Discretization can be performed recursively on an attribute
 - ❑ Prepare for further analysis, e.g., classification

Data Discretization Methods

- ❑ Binning
 - ❑ Top-down split, unsupervised
- ❑ Histogram analysis
 - ❑ Top-down split, unsupervised
- ❑ Clustering analysis
 - ❑ Unsupervised, top-down split or bottom-up merge
- ❑ Decision-tree analysis
 - ❑ Supervised, top-down split
- ❑ Correlation (e.g., χ^2) analysis
 - ❑ Unsupervised, bottom-up merge
- ❑ Note: All the methods can be applied recursively

Simple Discretization: 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

Example: 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 (**equal-depth**) bins:

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

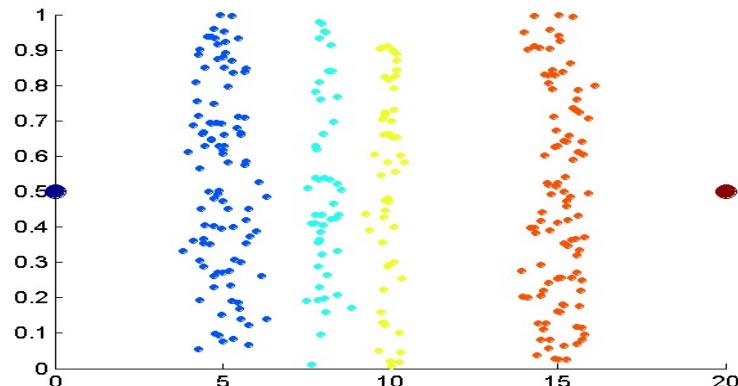
- * Smoothing by **bin means**:

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

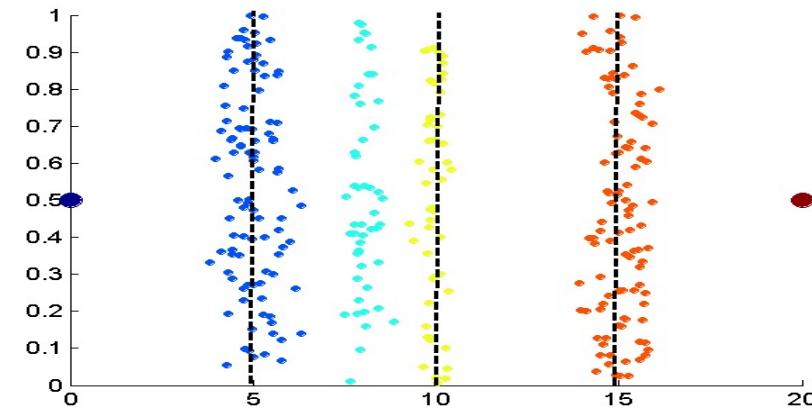
- * Smoothing by **bin boundaries**:

- Bin 1: 4, 4, 4, 15
- Bin 2: 21, 21, 25, 25
- Bin 3: 26, 26, 26, 34

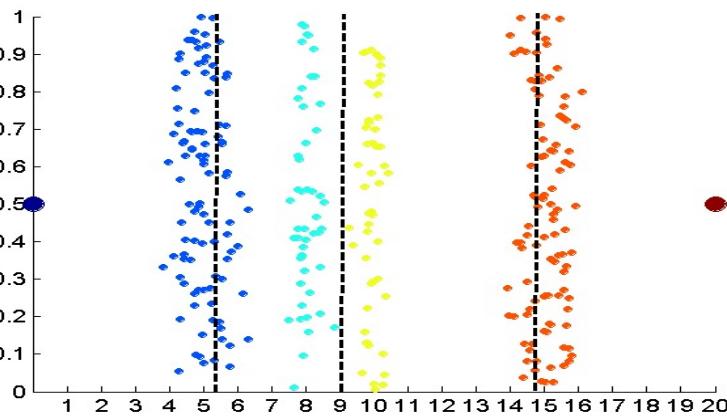
Discretization Without Supervision: Binning vs. Clustering



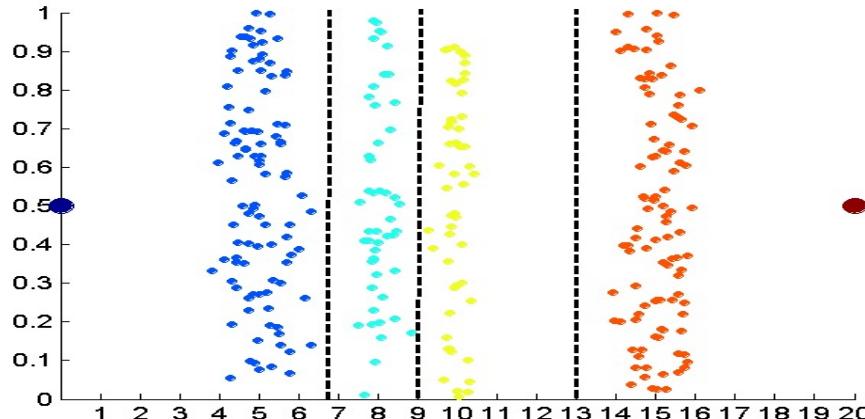
Data



Equal width (distance) binning



Equal depth (frequency) (binning)



K-means clustering leads to better results

Discretization by Classification & Correlation Analysis

- ❑ Classification (e.g., decision tree analysis)
 - ❑ Supervised: Given class labels, e.g., cancerous vs. benign
 - ❑ Using *entropy* to determine split point (discretization point)
 - ❑ Top-down, recursive split
 - ❑ Details to be covered in Chapter “Classification”
- ❑ Correlation analysis (e.g., Chi-merge: χ^2 -based discretization)
 - ❑ Supervised: use class information
 - ❑ Bottom-up merge: Find the best neighboring intervals (those having similar distributions of classes, i.e., low χ^2 values) to merge
 - ❑ Merge performed recursively, until a predefined stopping condition

Concept Hierarchy Generation

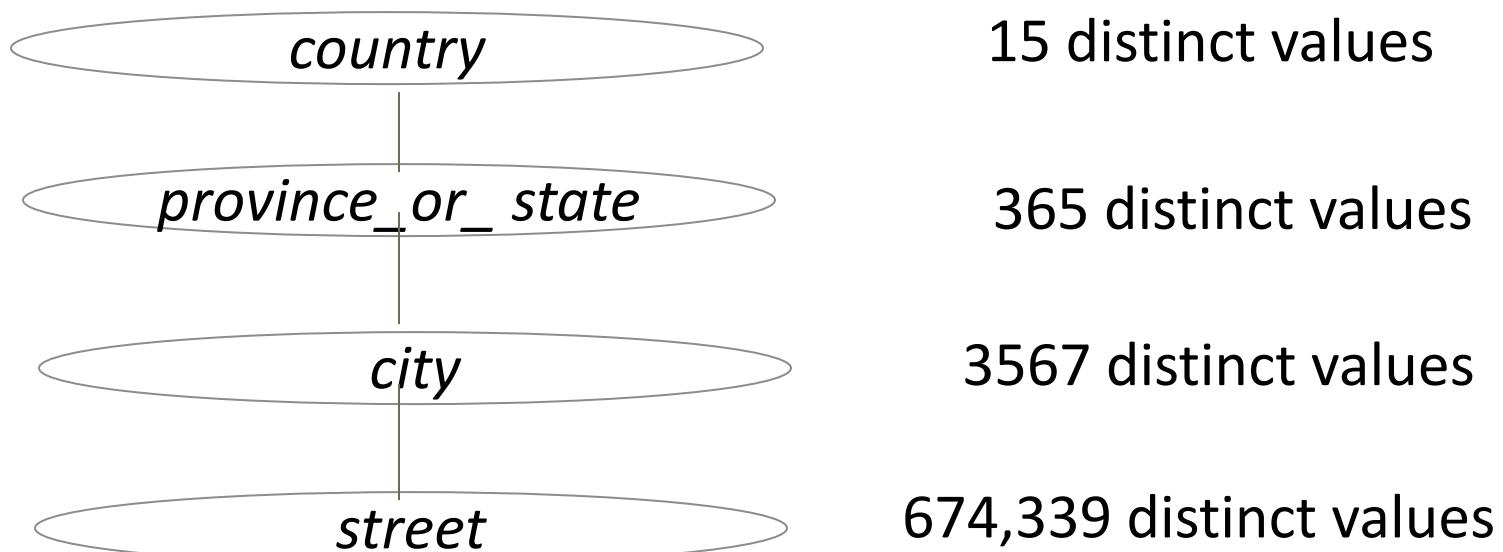
- **Concept hierarchy** organizes concepts (i.e., attribute values) hierarchically and is usually associated with each dimension in a data warehouse
- Concept hierarchies facilitate drilling and rolling in data warehouses to view data in multiple granularity
- Concept hierarchy formation: Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for *age*) by higher level concepts (such as *youth*, *adult*, or *senior*)
- Concept hierarchies can be explicitly specified by domain experts and/or data warehouse designers
- Concept hierarchy can be automatically formed for both numeric and nominal data—For numeric data, use discretization methods shown

Concept Hierarchy Generation for Nominal Data

- Specification of a partial/total ordering of attributes explicitly at the schema level by users or experts
 - $\text{street} < \text{city} < \text{state} < \text{country}$
- Specification of a hierarchy for a set of values by explicit data grouping
 - $\{\text{Urbana, Champaign, Chicago}\} < \text{Illinois}$
- Specification of only a partial set of attributes
 - E.g., only $\text{street} < \text{city}$, 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: $\{\text{street}, \text{city}, \text{state}, \text{country}\}$

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 3: Data Preprocessing

- Data Preprocessing: An Overview
- Data Cleaning
- Data Integration
- Data Reduction and Transformation
- Dimensionality Reduction
- Summary



Dimensionality Reduction

❑ Curse of dimensionality

- ❑ When dimensionality increases, data becomes increasingly sparse
- ❑ Density and distance between points, which is critical to clustering, outlier analysis, becomes less meaningful
- ❑ The possible combinations of subspaces will grow exponentially

❑ Dimensionality reduction

- ❑ Reducing the number of random variables under consideration, via obtaining a set of principal variables

❑ Advantages of dimensionality reduction

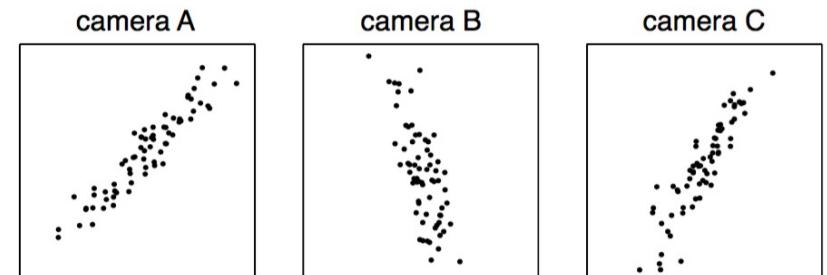
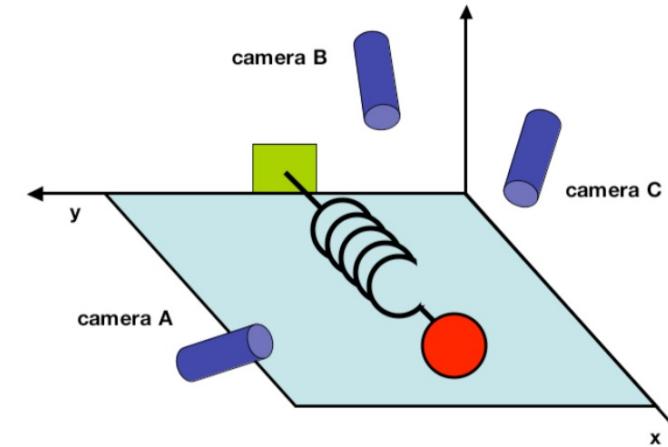
- ❑ Avoid the curse of dimensionality
- ❑ Help eliminate irrelevant features and reduce noise
- ❑ Reduce time and space required in data mining
- ❑ Allow easier visualization

Dimensionality Reduction Techniques

- Dimensionality reduction methodologies
 - **Feature selection:** Find a subset of the original variables (or features, attributes)
 - **Feature extraction:** Transform the data in the high-dimensional space to a space of fewer dimensions
- Some typical dimensionality methods
 - Principal Component Analysis
 - Supervised and nonlinear techniques
 - Feature subset selection
 - Feature creation

Principal Component Analysis (PCA)

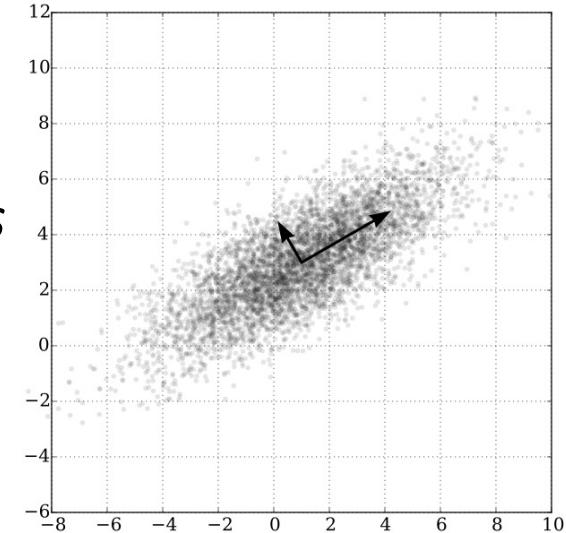
- PCA: A statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called ***principal components***
- The original data are projected onto a much smaller space, resulting in dimensionality reduction
- Method: Find the eigenvectors of the covariance matrix, and these eigenvectors define the new space



Ball travels in a straight line. Data from three cameras contain much redundancy

Principal Component Analysis (Method)

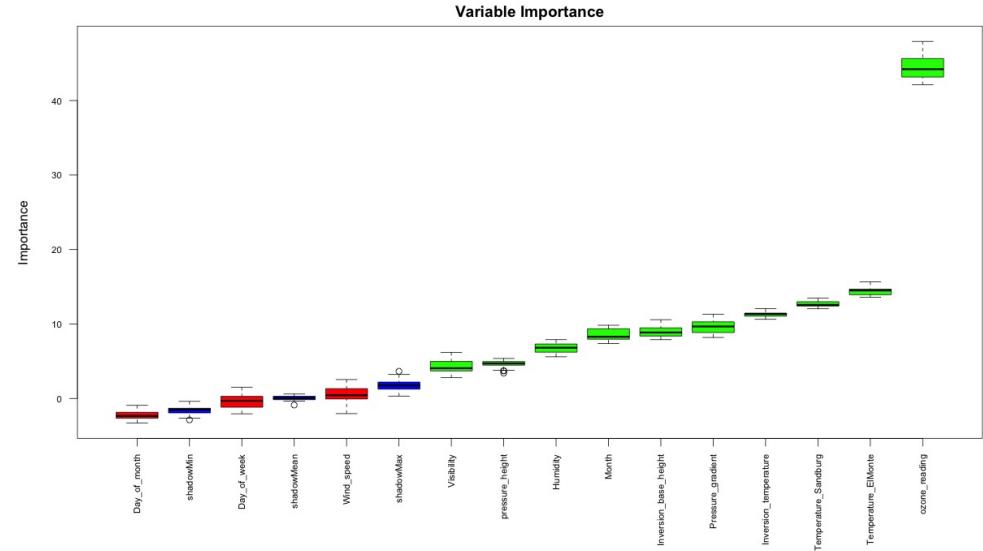
- Given N data vectors from n -dimensions, find $k \leq n$ orthogonal vectors (*principal components*) best used to represent data
 - Normalize input data: Each attribute falls within the same range
 - Compute k orthonormal (unit) vectors, i.e., *principal components*
 - Each input data (vector) is a 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, to reconstruct a good approximation of the original data)
- Works for numeric data only



Ack. Wikipedia: Principal Component Analysis

Attribute Subset Selection

- ❑ Another way to reduce dimensionality of data
- ❑ Redundant attributes
 - ❑ Duplicate much or all of the information contained in one or more other attributes
 - ❑ E.g., purchase price of a product and the amount of sales tax paid
- ❑ Irrelevant attributes
 - ❑ Contain no information that is useful for the data mining task at hand
 - ❑ Ex. A student's ID is often irrelevant to the task of predicting his/her GPA



Heuristic Search in Attribute Selection

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

Attribute Creation (Feature Generation)

- Create new attributes (features) that can capture the important information in a data set more effectively than the original ones
- Three general methodologies
 - Attribute extraction
 - Domain-specific
 - Mapping data to new space (see: data reduction)
 - E.g., Fourier transformation, wavelet transformation, manifold approaches (not covered)
 - Attribute construction
 - Combining features (see: discriminative frequent patterns in Chapter on “Advanced Classification”)
 - Data discretization

Summary

- **Data quality:** accuracy, completeness, consistency, timeliness, believability, interpretability
- **Data cleaning:** e.g. missing/noisy values, outliers
- **Data integration** from multiple sources:
 - Entity identification problem; Remove redundancies; Detect inconsistencies
- **Data reduction, data transformation and data discretization**
 - Numerosity reduction; Data compression
 - Normalization; Concept hierarchy generation
- **Dimensionality reduction**

References

- D. P. Ballou and G. K. Tayi. Enhancing data quality in data warehouse environments. Comm. of ACM, 42:73-78, 1999
- T. Dasu and T. Johnson. Exploratory Data Mining and Data Cleaning. John Wiley, 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), Dec. 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
- 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





CS 412 Intro. to Data Mining

Chapter 4. Data Warehousing and On-line Analytical Processing

Arindam Banerjee, Computer Science, UIUC, Fall 2021

Chapter 4: Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Summary



What is a Data Warehouse?

- Defined in many different ways, but not rigorously
 - Support decision
 - Maintained Separately
 - Information processing
- “A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management’s decision-making process.”—W. H. Inmon
- Data warehousing:
 - The process of constructing and using data warehouses

Data Warehouse—Subject-Oriented

- Help make decisions
 - A **simple** and **concise** view (modeling and analysis)
 - Not details (transaction processing)
 - Organizing around major subjects, such as **customer, product, sales**
 - Excluding data that are not useful in the decision support process

Data Warehouse—Integrated

- Integrating Multiple, heterogeneous sources
 - Ex. relational databases, flat files, on-line transaction records
- Consistency
 - Data cleaning and data integration techniques are applied
 - Ex. Hotel price: differences in currency, tax, breakfast covered, and parking
 - When data is moved to the warehouse, it is converted

Data Warehouse—Time Variant

Data Warehouse	Operational Database
Long time horizon (e.g., past 5-10 years)	current value data
Contains an element of time, explicitly or implicitly	data may or may not contain “time element”

Data Warehouse—Nonvolatile

- ❑ Independence – A physically separate store
- ❑ Static – No data management (updates, transaction processing, recovery, and concurrency control mechanisms)
- ❑ Requires only two operations in data accessing:
 - ❑ *initial loading of data* and *access of data*

Why a Separate Data Warehouse?

- Different functions and different data:
 - missing data: Decision support requires historical data which operational DBs do not typically maintain
 - data consolidation: DS requires consolidation (aggregation, summarization) of data from heterogeneous sources
 - data quality: different sources typically use inconsistent data representations, codes and formats which have to be reconciled
- Note: There are more and more systems which perform OLAP (online analytical processing) analysis directly on relational databases

OLTP vs. OLAP

- ❑ OLTP: Online transactional processing
 - ❑ DBMS operations
 - ❑ Query and transactional processing
- ❑ OLAP: Online analytical processing
 - ❑ Data warehouse operations
 - ❑ Drilling, slicing, dicing, etc.

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

OLTP vs. OLAP

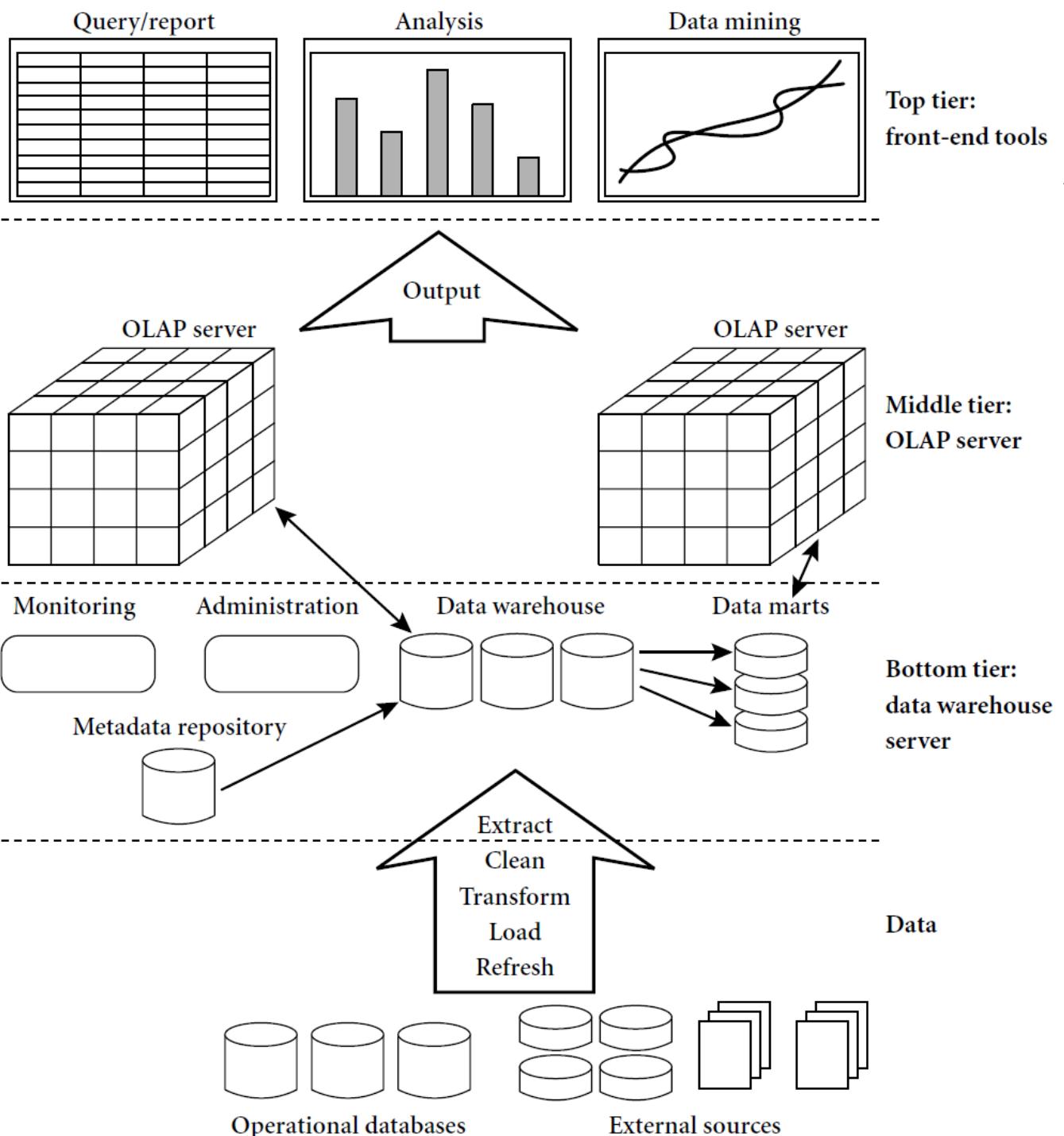
Table 4.1: Comparison of OLTP and OLAP Systems

<i>Feature</i>	<i>OLTP</i>	<i>OLAP</i>
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)
Function	day-to-day operations	long-term informational requirements decision support
DB design	ER-based, application-oriented	star/snowflake, subject-oriented
Data	current, guaranteed up-to-date	historic, accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
Number of records accessed	tens	millions
Number of users	thousands	hundreds
DB size	GB to high-order GB	\geq TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

Note: Table is partially based on Chaudhuri and Dayal [CD97].

Data Warehouse: A Multi-Tiered Architecture

- Top Tier: Front-End Tools
- Middle Tier: OLAP Server
- Bottom Tier: Data Warehouse Server
- Data



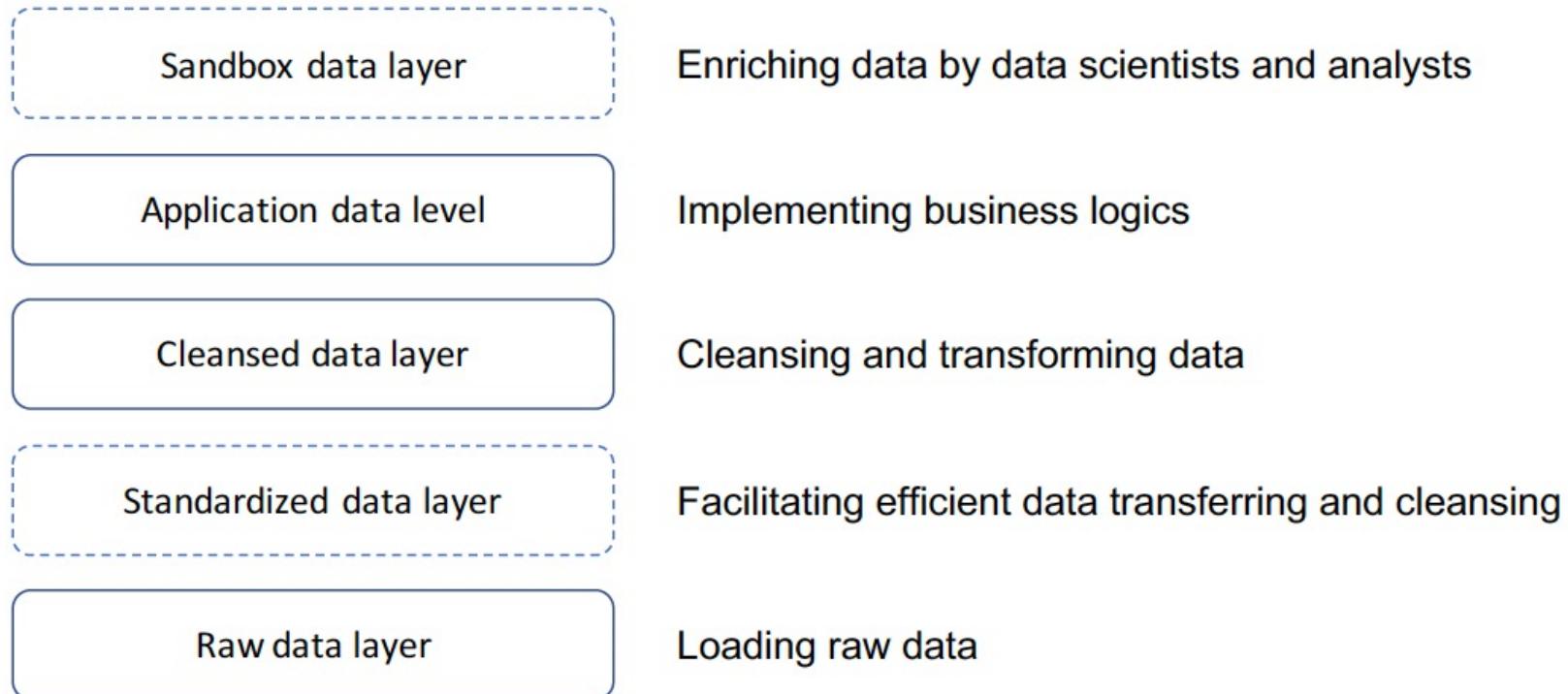
Extraction, Transformation, and Loading (ETL)

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

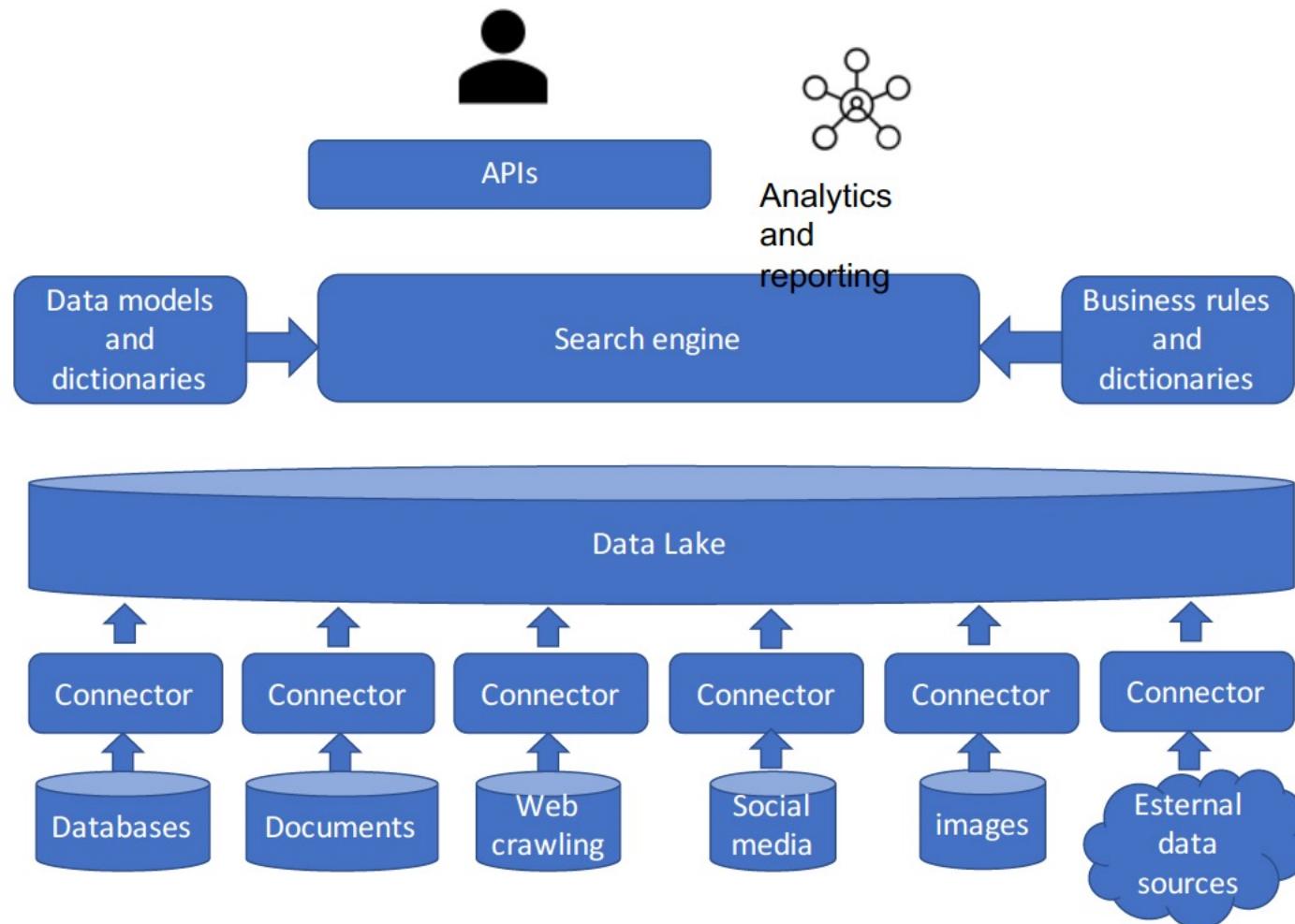
Three Data Warehouse Models

- ❑ **Enterprise warehouse** - Specially designed for the entire organization
- ❑ **Data Mart**
 - ❑ Specific, selected groups
 - ❑ Independent vs. dependent (directly from warehouse) data mart
- ❑ **Virtual warehouse**
 - ❑ A set of views over operational databases
 - ❑ Only some of the possible summary views may be materialized
- ❑ **Data Lakes**
 - ❑ Single repo of all enterprise data in natural (possibly different) format
 - ❑ Base for all data related tasks, for all users, not structured like a warehouse
 - ❑ Does not need the design and development time like a warehouse

Layers of Data Storage in Data Lakes



Conceptual Architecture of Data Lakes



Metadata Repository

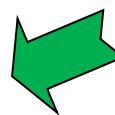
- ❑ **Meta data** is data about data. It stores:
 - ❑ Description of structure (schema, etc.)
 - ❑ Operational meta-data
 - ❑ data lineage (history of migrated data and transformation path), currency of data (active, archived, or purged), monitoring information (warehouse usage statistics, error reports, audit trails)
 - ❑ The algorithms used for summarization
 - ❑ The mapping from operational environment to the data warehouse
 - ❑ Data related to system performance
 - ❑ warehouse schema, view and derived data definitions
 - ❑ Business data
 - ❑ business terms and definitions, ownership of data, charging policies

Chapter 4: Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts

- Data Warehouse Modeling: Data Cube and OLAP

- Summary

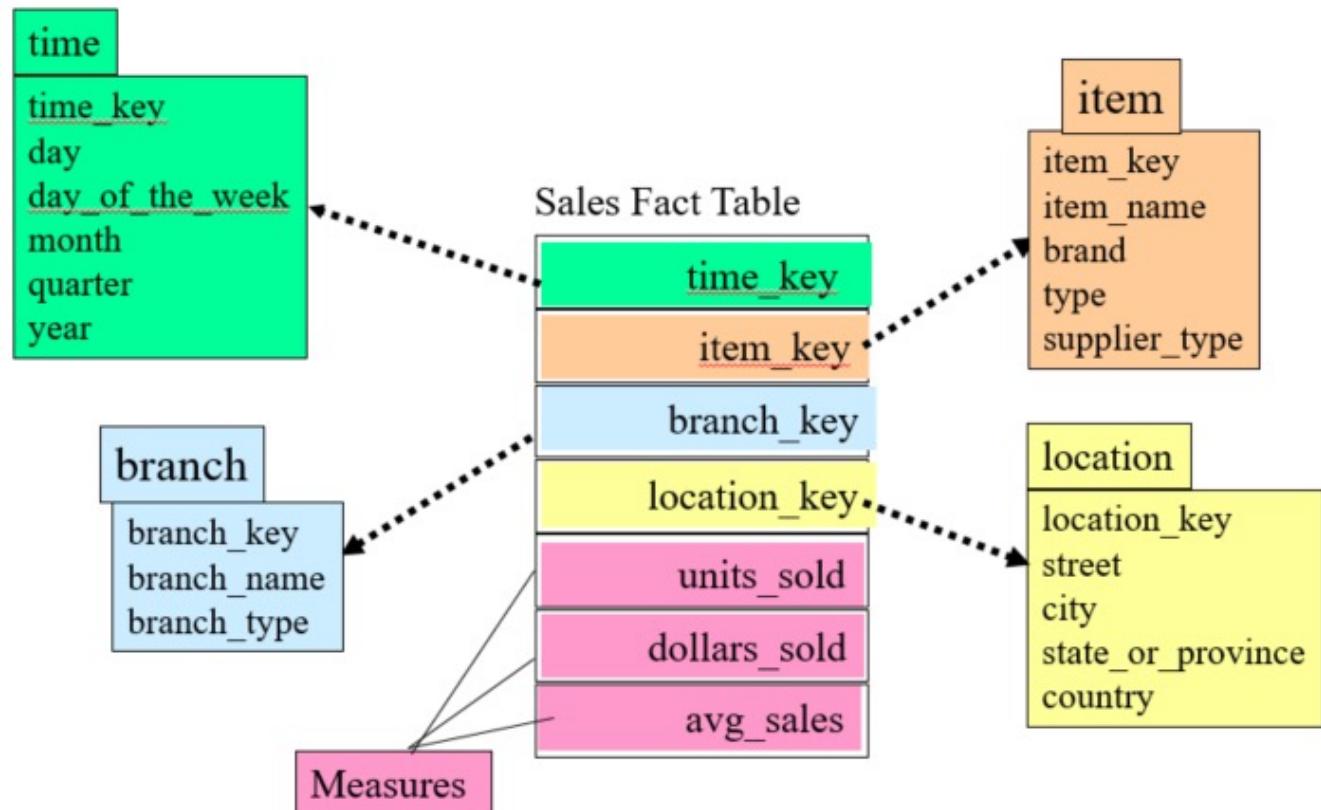


From Tables and Spreadsheets to Data Cubes

- A **data warehouse** is based on a multidimensional data model which views data in the form of a data cube
- Main function is to provide summarizations of the data
 - E.g., summarize the units or dollars sold at a particular store over a particular time period
- Can compute summarizations online (as they are requested)
 - Can be very slow
- Better to precalculate some summarizations

Design of Data Warehouses

- ❑ Dimension tables, such as item (item_name, brand, type), or time(day, week, month, quarter, year)
- ❑ Fact table contains measures (such as dollars_sold) and keys to each of the related dimension tables
- ❑ Different schema exist
 - ❑ Star
 - ❑ Snowflake
 - ❑ Fact constellation



Example: 2-D View of Sales Data

Table 4.2: 2-D View of Sales Data According to *time* and *item location* = “Vancouver”

<i>time</i> (<i>quarter</i>)	<i>item</i> (<i>type</i>)			
	<i>home</i>	<i>entertainment</i>	<i>computer</i>	<i>phone</i>
Q1		605	825	14
Q2		680	952	31
Q3		812	1023	30
Q4		927	1038	38

Note: The sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).

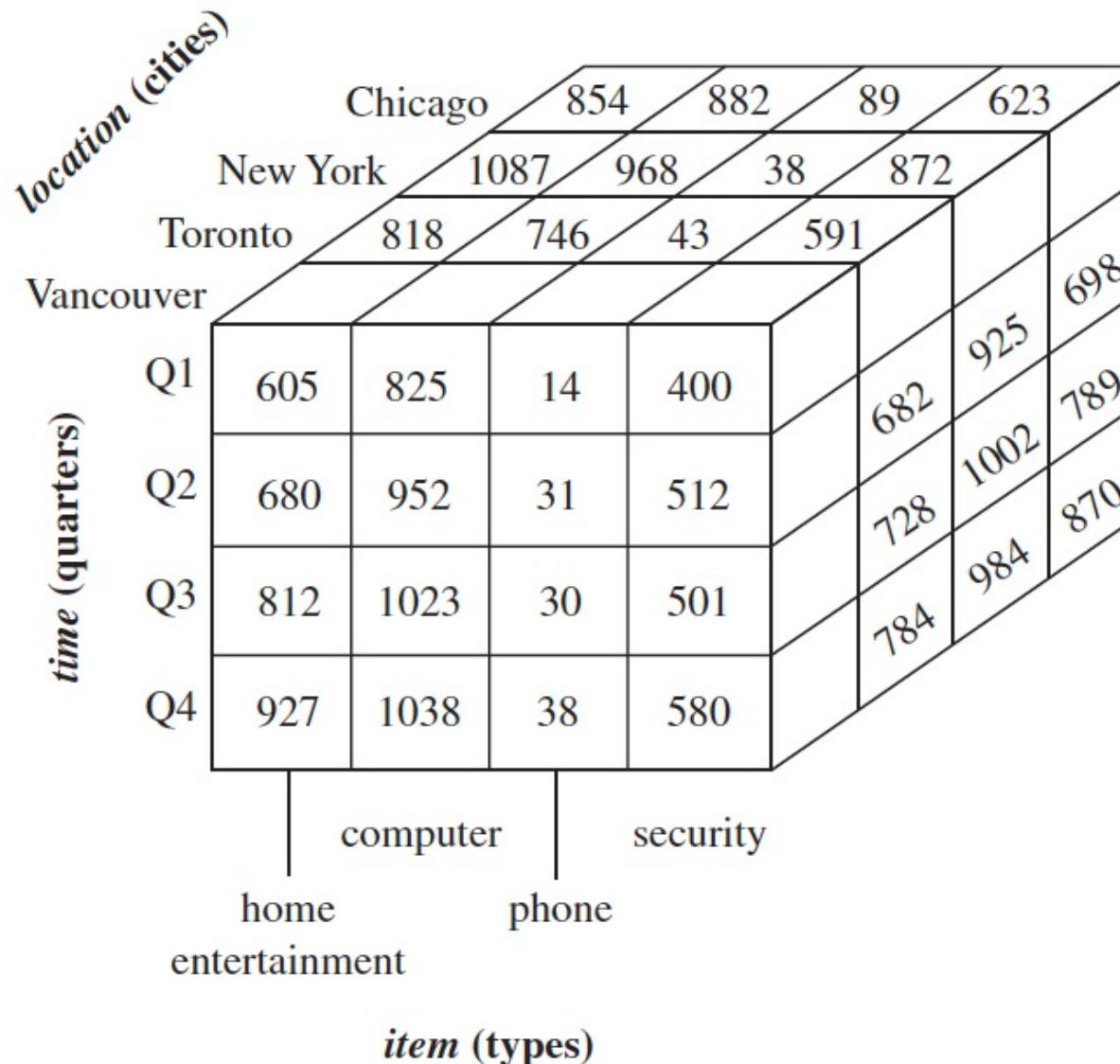
Example: 3-D View of Sales Data

Table 4.3: 3-D View of Sales Data According to *time*, *item*, and *location*
location = “Chicago” *location* = “New York” *location* = “Toronto” *location* = “Vancouver”

item																
home																
time	ent.	comp.	phone	sec.												
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580

Note: The measure displayed is *dollars_sold* (in thousands).

Example: 3-D View of Sales Data



Example: 4-D View of Sales Data

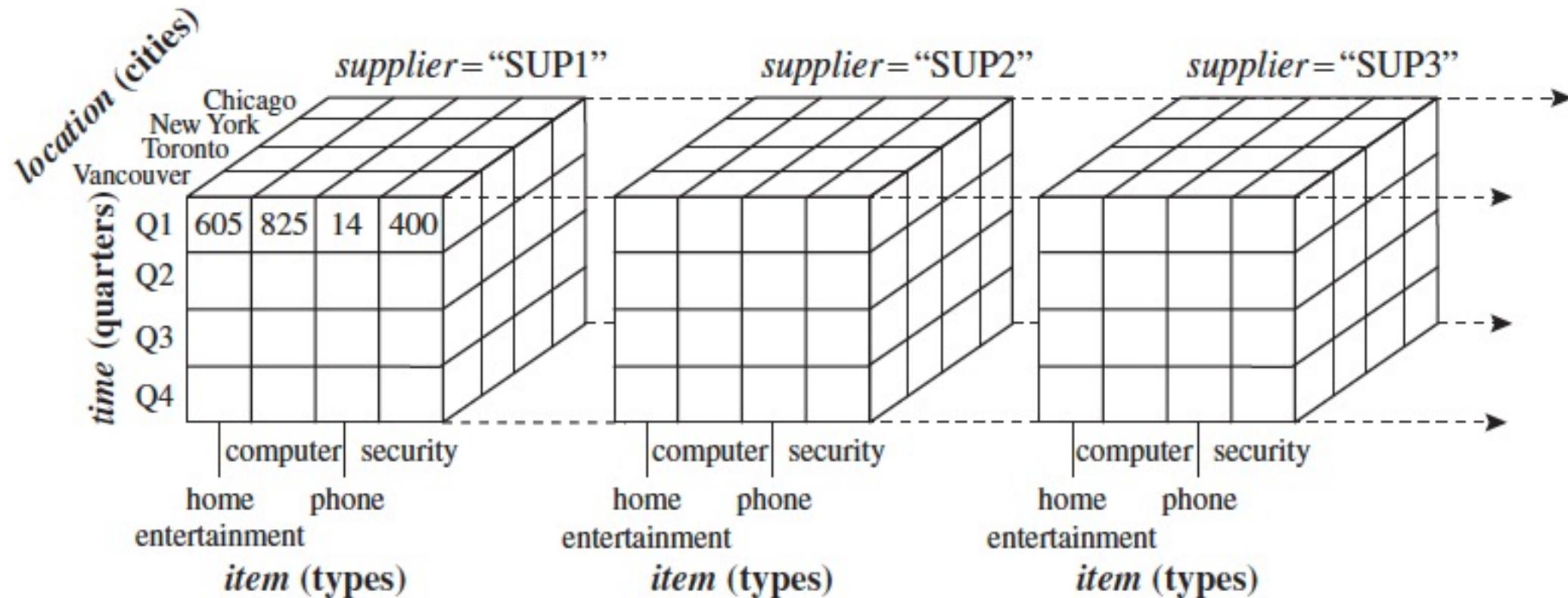
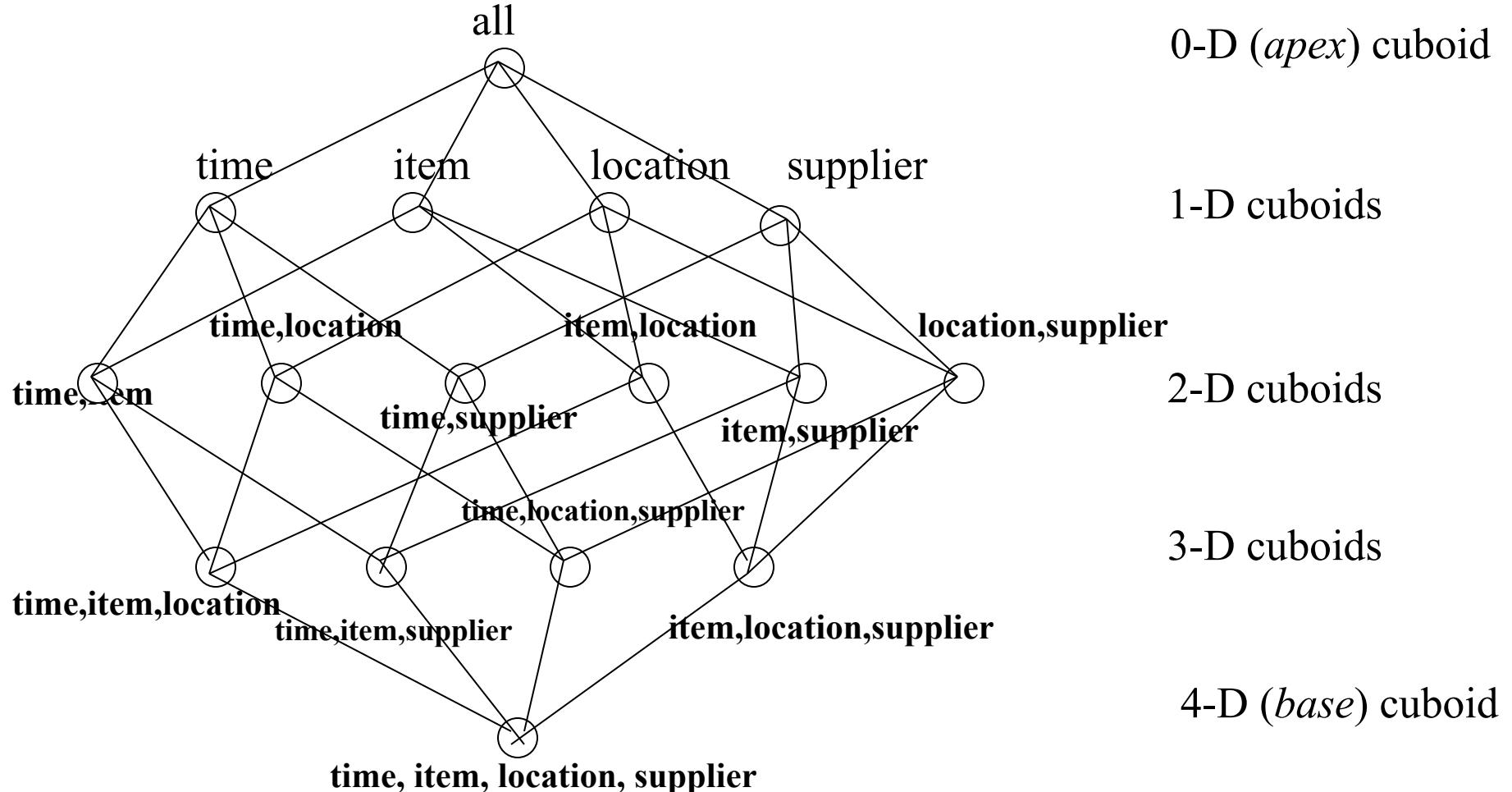


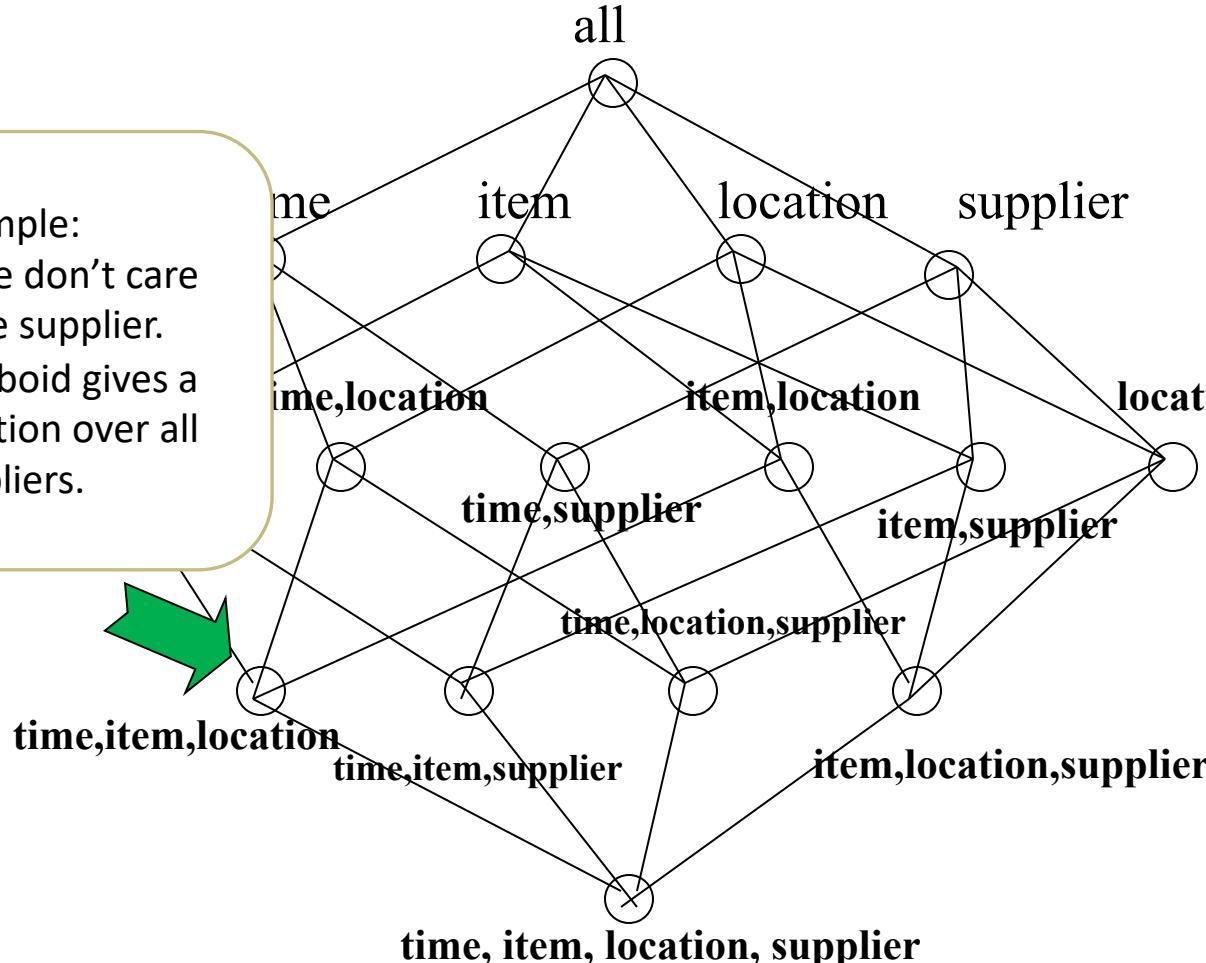
Figure 4.4: A 4-D data cube representation of sales data, according to *time*, *item*, *location*, and *supplier*. The measure displayed is *dollars_sold* (in thousands). For improved readability, only some of the cube values are shown.

Data Cube: A Lattice of Cuboids



Data Cube: A Lattice of Cuboids

Example:
Suppose we don't care about the supplier.
This 3-D cuboid gives a summarization over all suppliers.



0-D (*apex*) cuboid

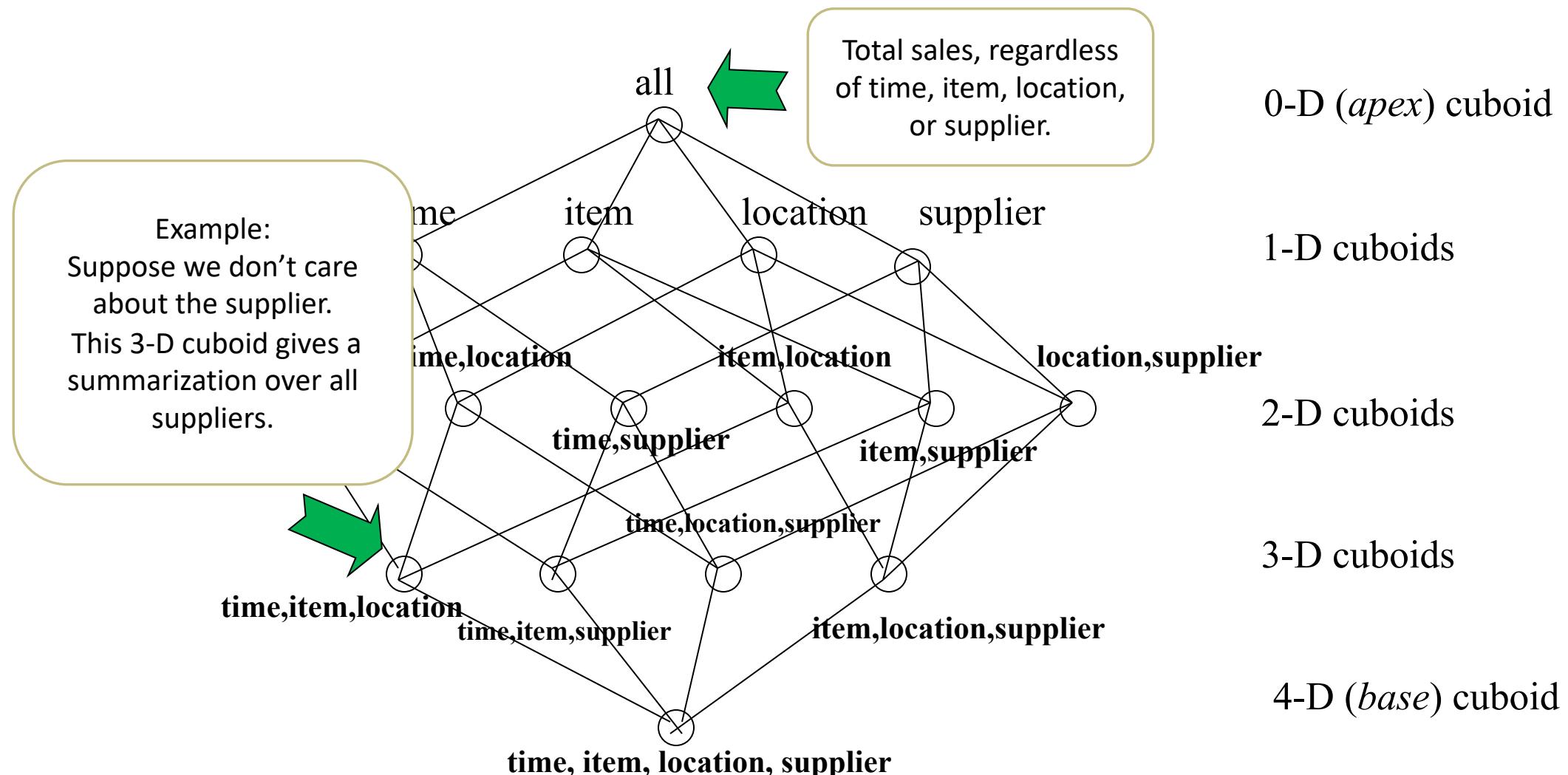
1-D cuboids

2-D cuboids

3-D cuboids

4-D (*base*) cuboid

Data Cube: A Lattice of Cuboids

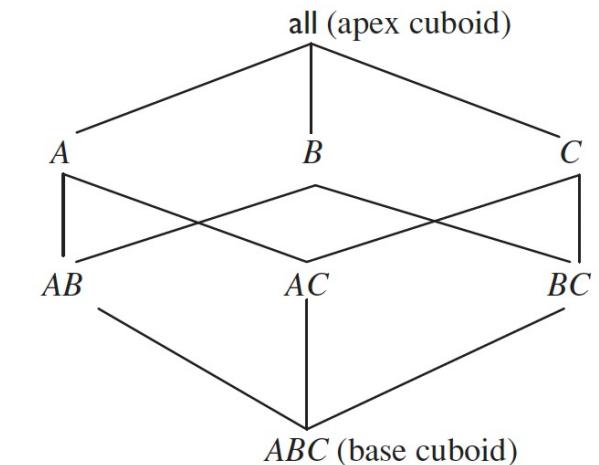


Calculating Number of Cuboids

- Consider dimensions as binary numbers
- Example: 4 dimensions
 - Each is either in the cuboid, or not in the cuboid
 - $(\ , \ , \ , \) \leftarrow$ choice of 0 or 1 for each element of vector
 - Sum up for each position: $2^3 + 2^2 + 2^1 + 2^0 + 1$ (0-d cuboid) = 2^4
- In general, 2^d cuboids (d = number of dimensions)

Calculating Number of Cuboids

- ❑ Tuple in a cuboid is a cell
 - ❑ Base cuboid is a base cell
 - ❑ Otherwise, aggregate cell
- ❑ Ancestor and descendant relationship
 - ❑ (Jan, *, *, 2800) is ancestor of (Jan, Chicago, *, 2800)
- ❑ Each dimension may have concept hierarchies
 - ❑ Li for dimension i



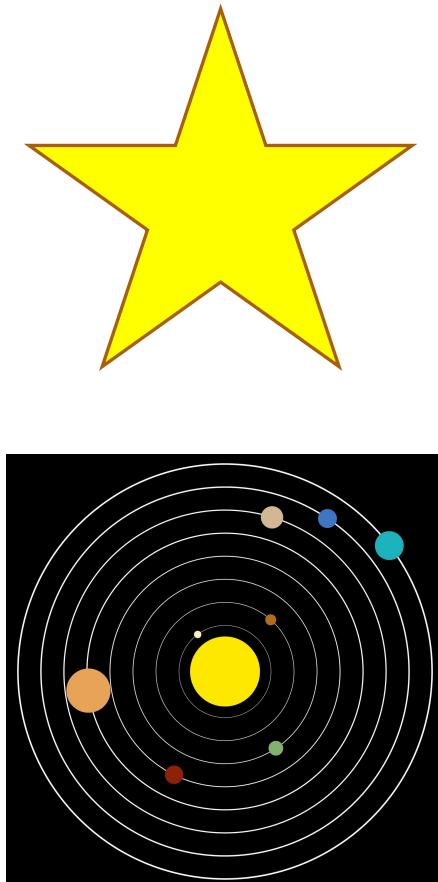
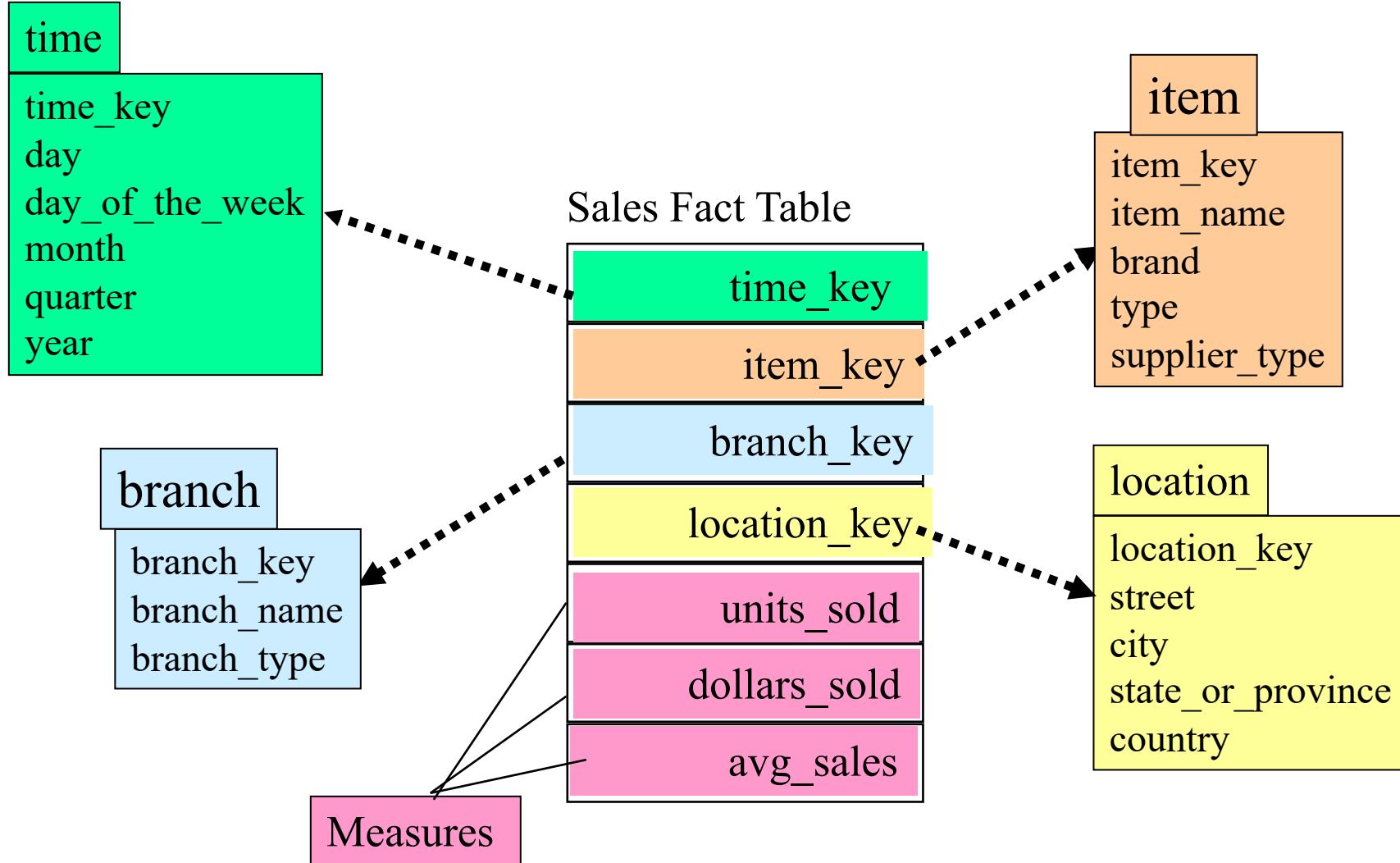
$$Total\ cuboids = \prod_{i=1}^n (L_i + 1)$$

Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
 - Star schema
 - Snowflake schema
 - Fact constellations

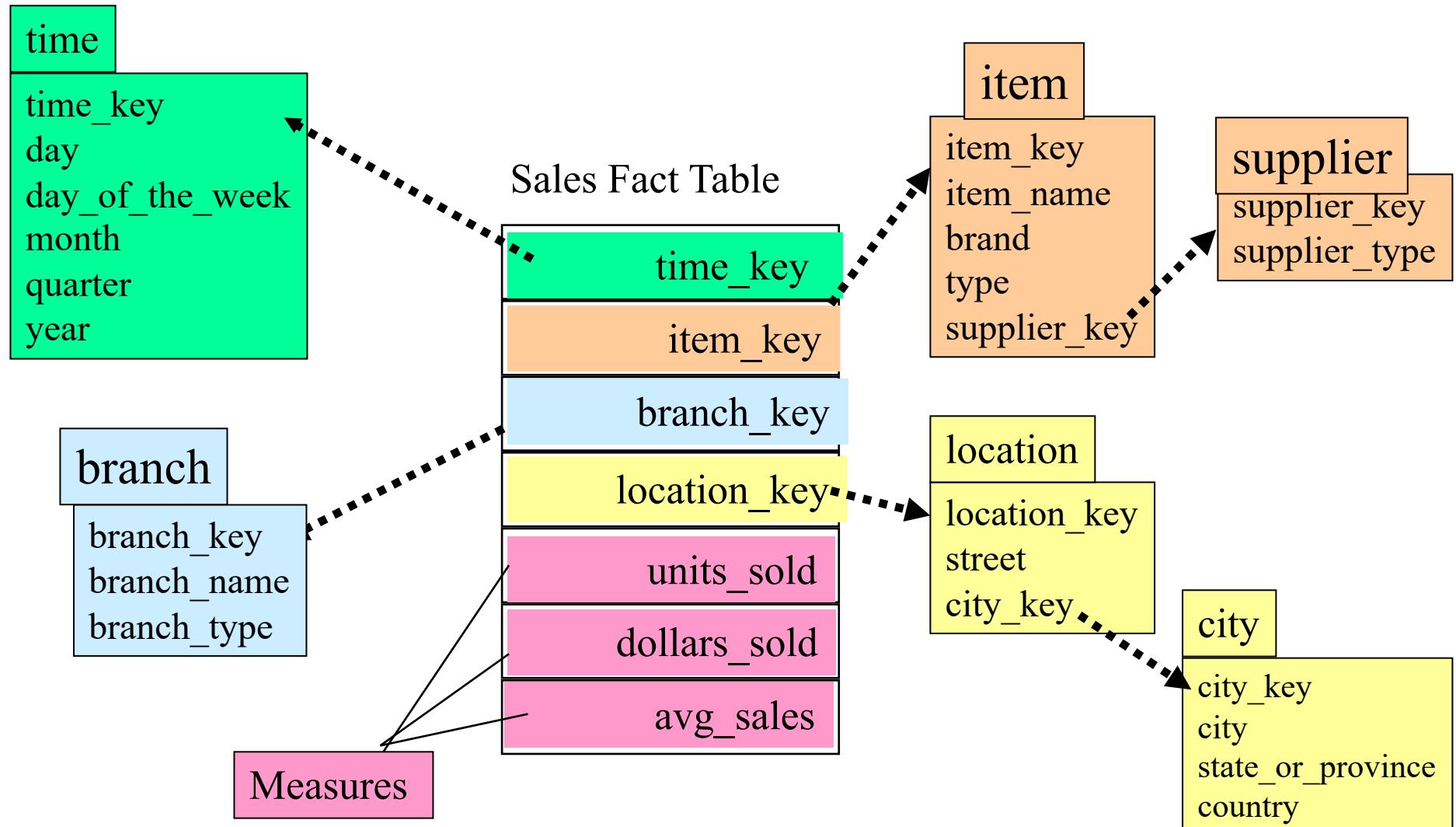
Star Schema: An Example

A fact table in the middle connected to a set of dimension tables



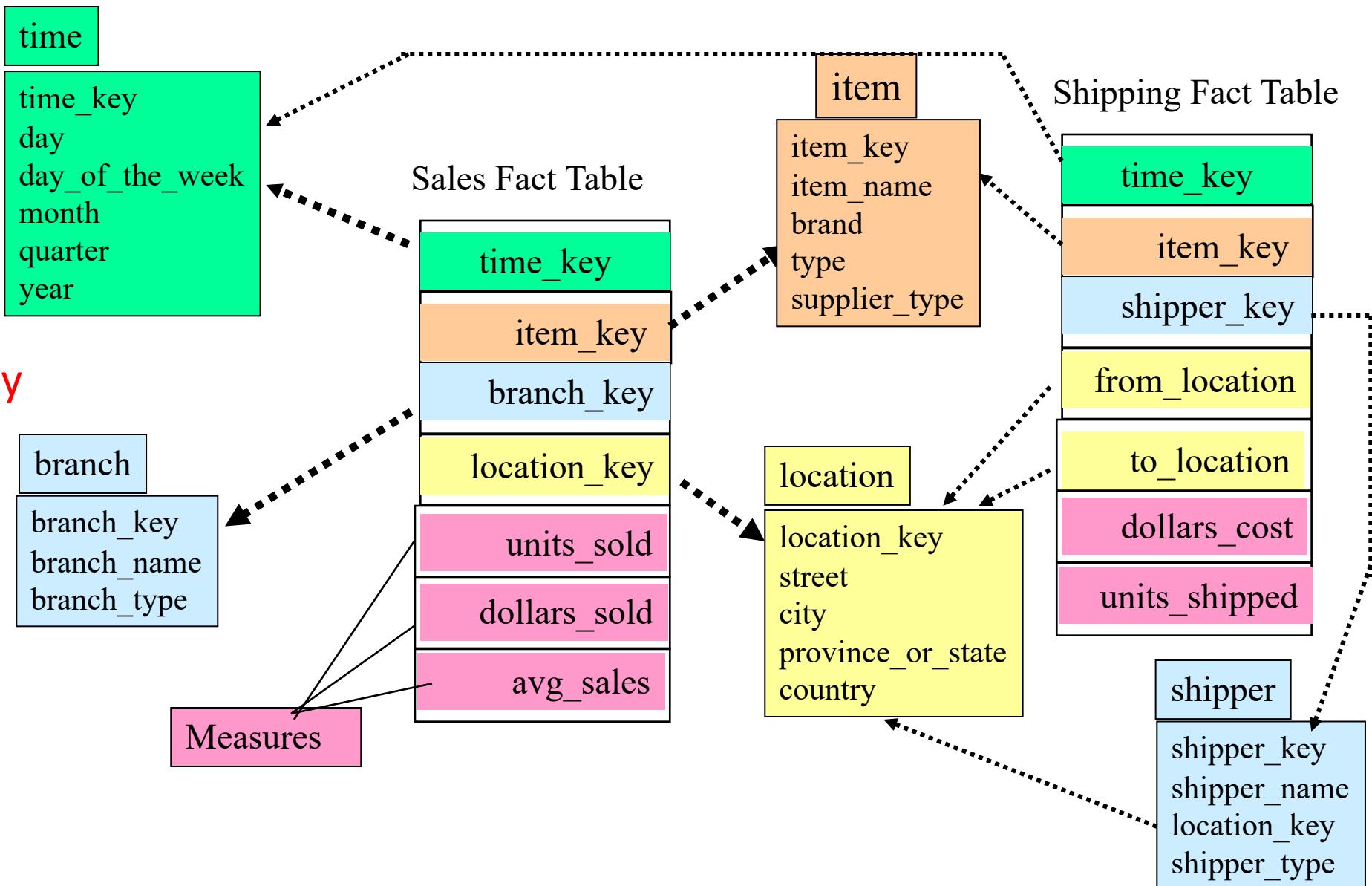
Snowflake Schema: An Example

A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake

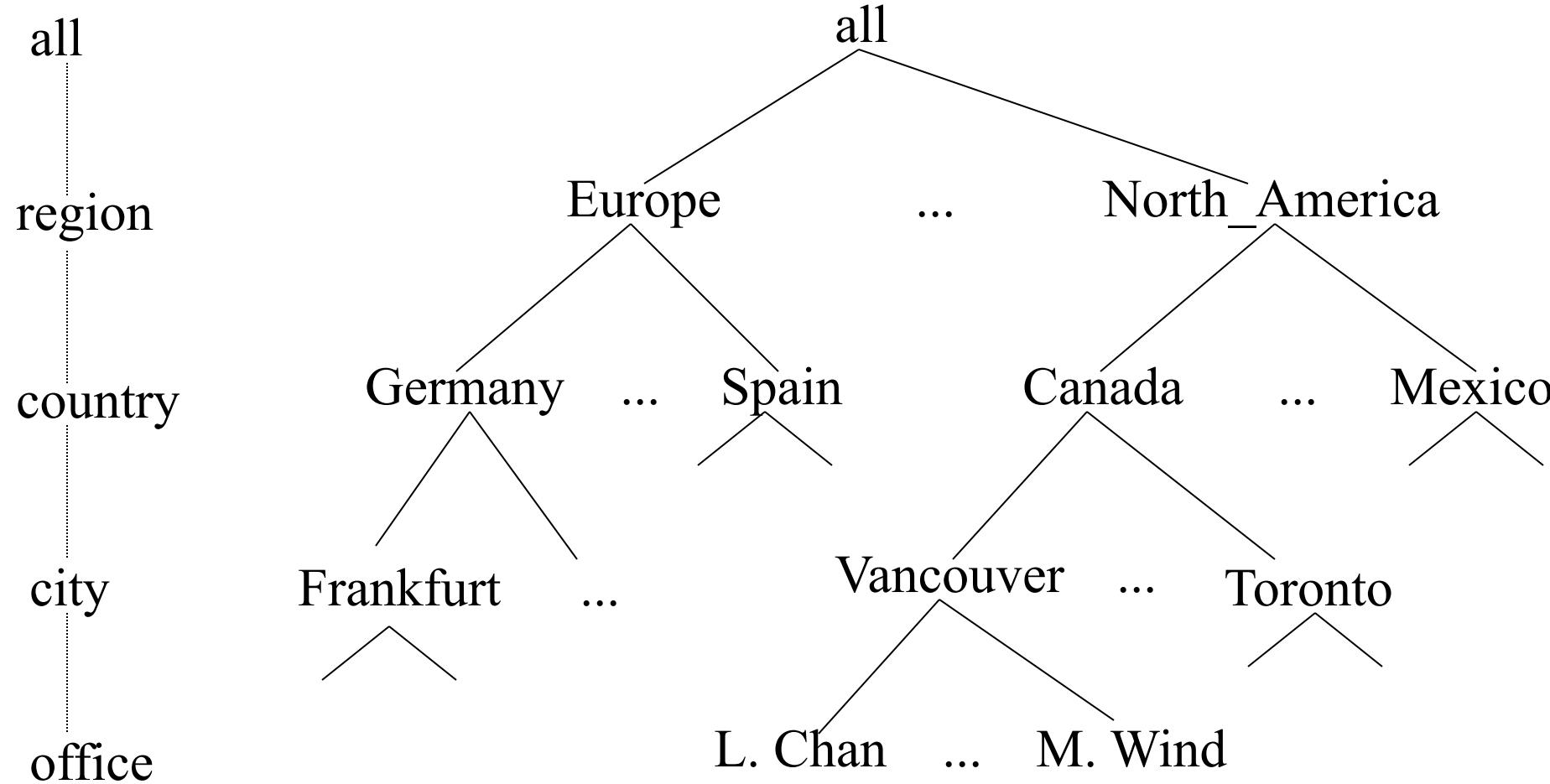


Fact Constellation: An Example

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



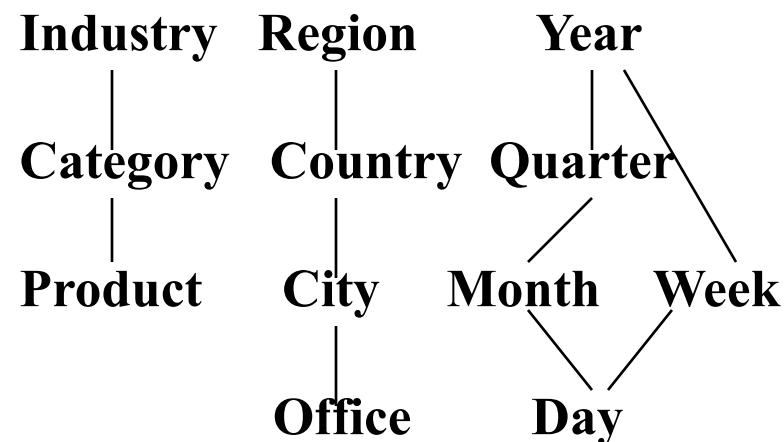
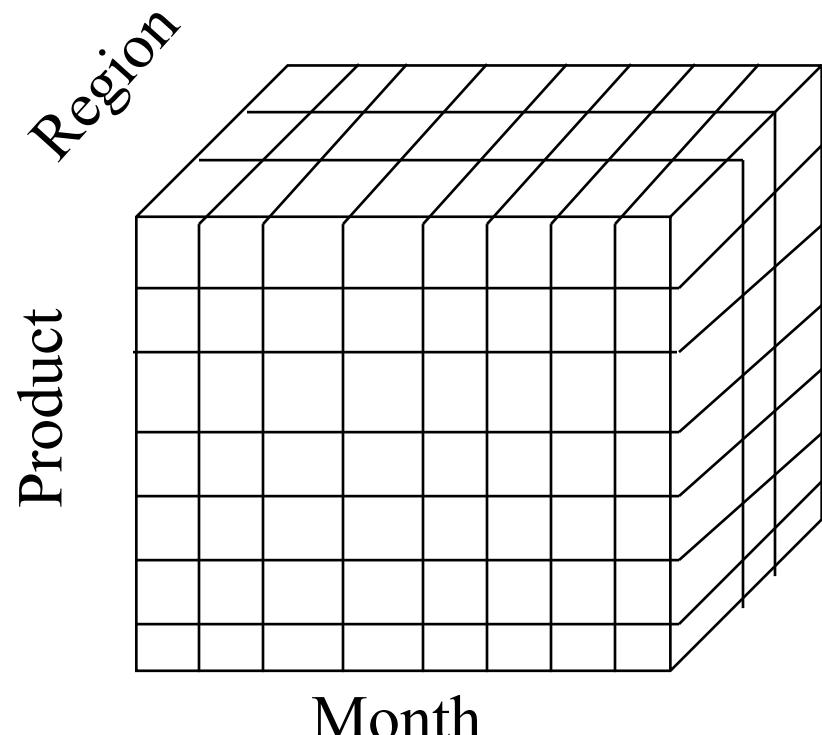
A Concept Hierarchy for a Dimension (location)



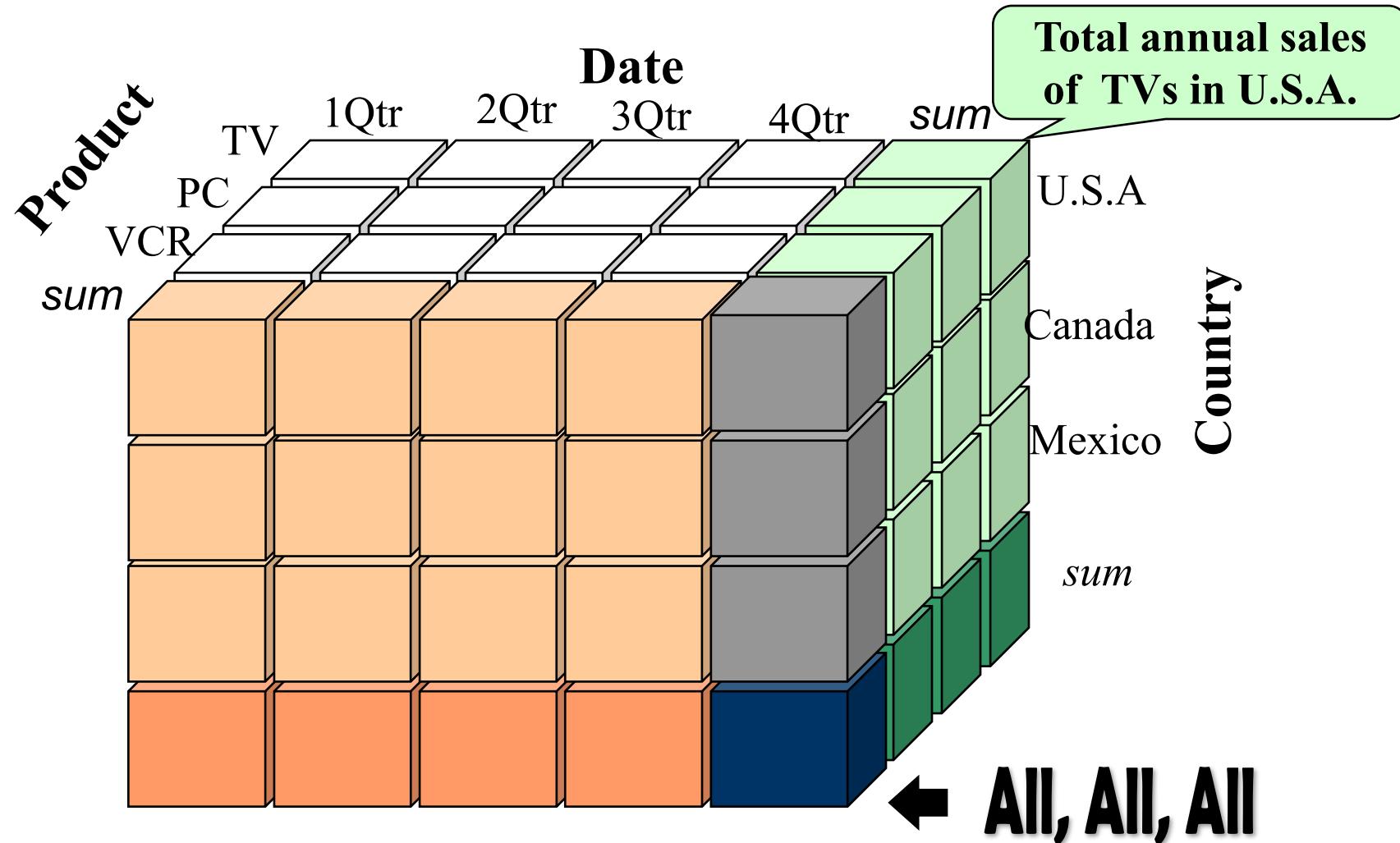
Multidimensional Data

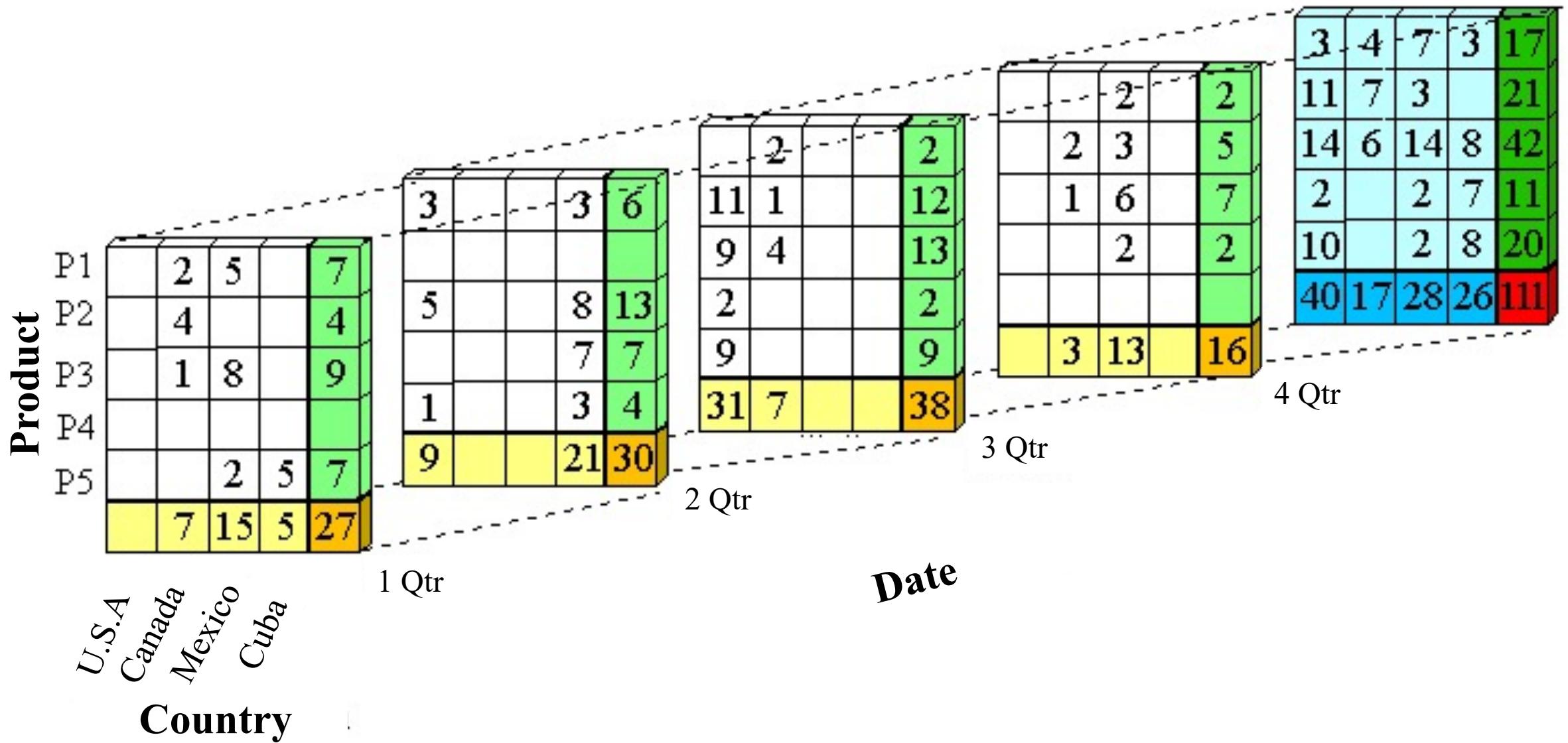
- Sales volume as a function of product, month, and region

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

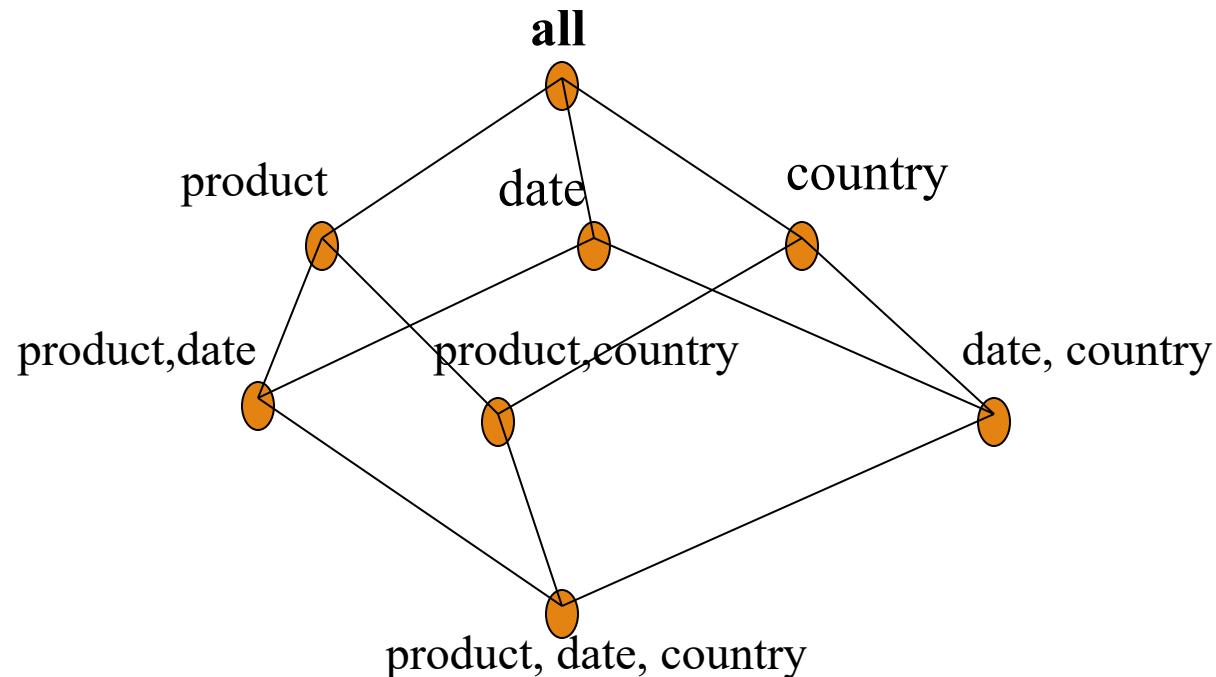


A Sample Data Cube





Cuboids Corresponding to the Cube



0-D (*apex*) cuboid

1-D cuboids

2-D cuboids

3-D (*base*) cuboid

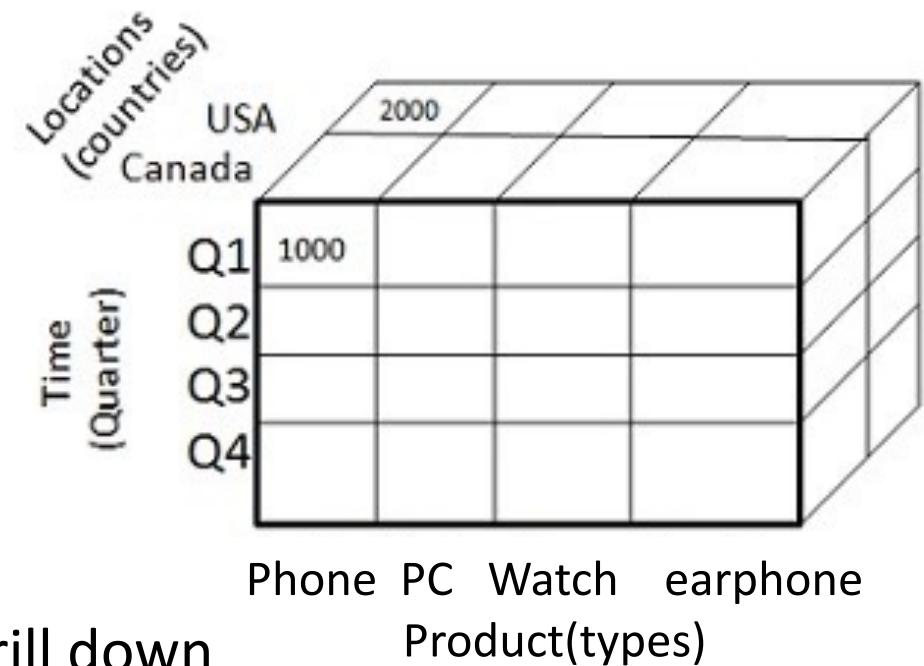
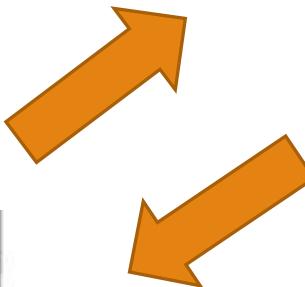
How can we play with the Cube?

Roll up & Drill down

Roll up (drill-up): summarize data
by climbing up hierarchy or by dimension reduction

		Product(types)				
		Phone	PC	Watch	earphone	
Time (Quarter)	Locations (cities)	Chicago				
		New York	1560			
		Toronto	395			
		Vancouver				
		Q1	605	825	14	
		Q2			400	
		Q3				
		Q4				

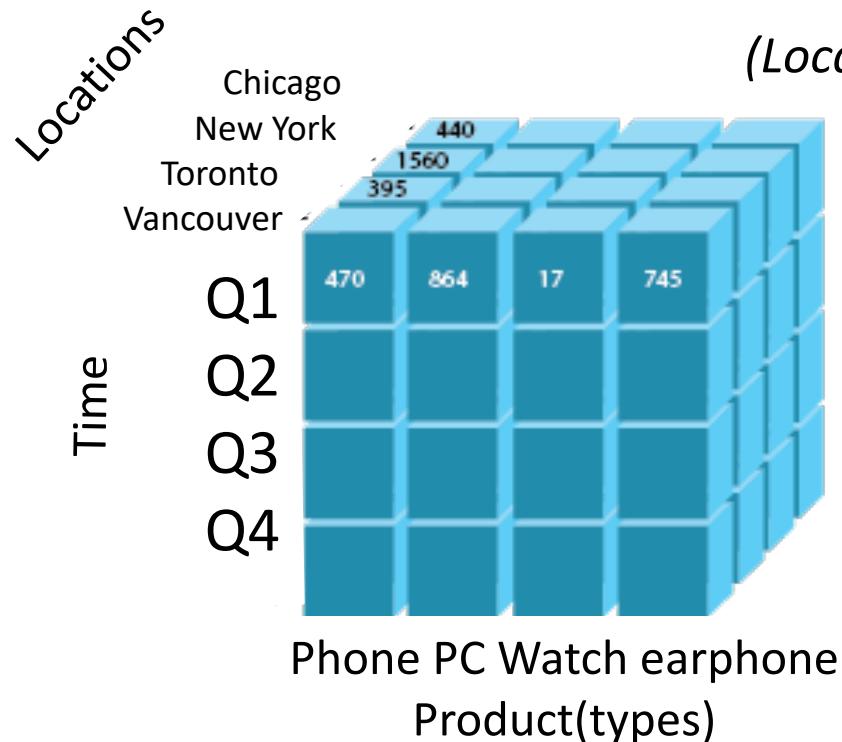
Roll up



Drill down

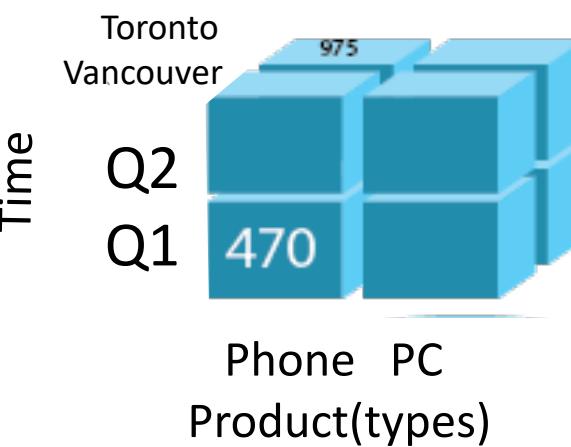
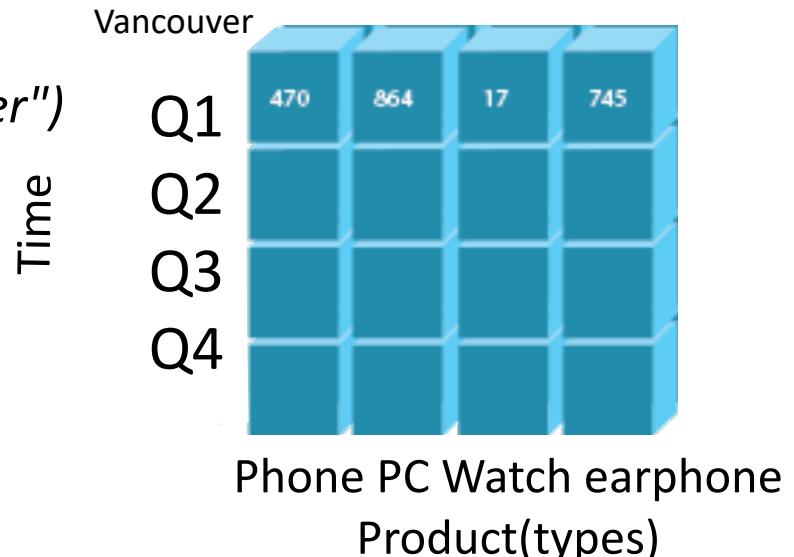
Drill down (roll down): reverse of roll-up
from higher level summary to lower level summary or detailed data, or introducing new dimensions

Dice and Slice



*Slice for
(Location = "Vancouver")*

*Dice for
(Location = "Toronto" or "Vancouver")
and (Time = "Q2" or "Q1") and
(Product = "Phone" or "PC")*



Chapter 4: Data Warehousing and On-line Analytical Processing

- Data Warehouse: Basic Concepts
- Data Warehouse Modeling: Data Cube and OLAP
- Summary 

Summary

- ❑ Data warehousing: A multi-dimensional model of a data warehouse
 - ❑ A data cube consists of *dimensions & measures*
 - ❑ Star schema, snowflake schema, fact constellations
 - ❑ OLAP operations: drilling, rolling, slicing, dicing and pivoting
- ❑ Data Warehouse Architecture, Design, and Usage
 - ❑ Multi-tiered architecture
 - ❑ Business analysis design framework
 - ❑ Information processing, analytical processing, data mining

References (I)

- 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
- D. Agrawal, A. E. Abbadi, A. Singh, and T. Yurek. Efficient view maintenance in data warehouses. SIGMOD'97
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997
- J. Gray, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.
- A. Gupta and I. S. Mumick. Materialized Views: Techniques, Implementations, and Applications. MIT Press, 1999
- J. Han. Towards on-line analytical mining in large databases. *SIGMOD Record*, 1998
- V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. SIGMOD'96

References (II)

- C. Imhoff, N. Galemmo, and J. G. Geiger. Mastering Data Warehouse Design: Relational and Dimensional Techniques. John Wiley, 2003
- W. H. Inmon. Building the Data Warehouse. John Wiley, 1996
- R. Kimball and M. Ross. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling. 2ed. John Wiley, 2002
- P. O'Neil and D. Quass. Improved query performance with variant indexes. SIGMOD'97
- S. Sarawagi and M. Stonebraker. Efficient organization of large multidimensional arrays. ICDE'94
- P. Valduriez. Join indices. ACM Trans. Database Systems, 12:218-246, 1987.
- J. Widom. Research problems in data warehousing. CIKM'95.
- K. Wu, E. Otoo, and A. Shoshani, Optimal Bitmap Indices with Efficient Compression, ACM Trans. on Database Systems (TODS), 31(1), 2006, pp. 1-38.



CS 412 Intro. to Data Mining

Chapter 5. Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

Arindam Banerjee, Computer Science, UIUC, Fall 2021

Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary



Pattern Discovery: Basic Concepts

- What Is Pattern Discovery? Why Is It Important?

- Basic Concepts: Frequent Patterns and Association Rules

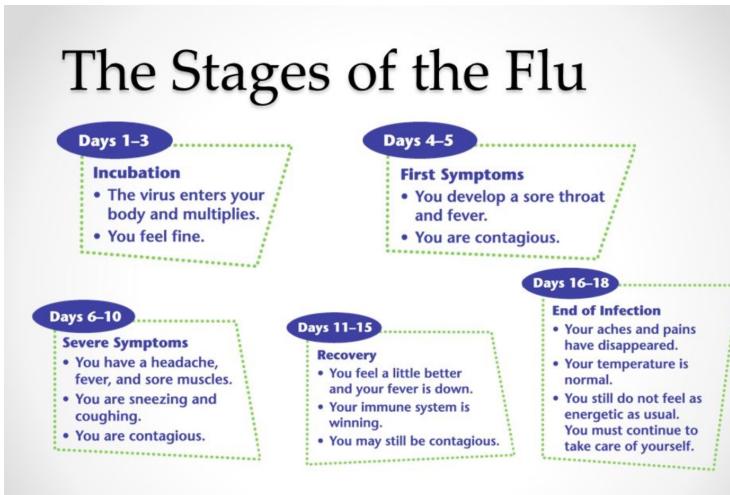
- Compressed Representation: Closed Patterns and Max-Patterns

What are Patterns?

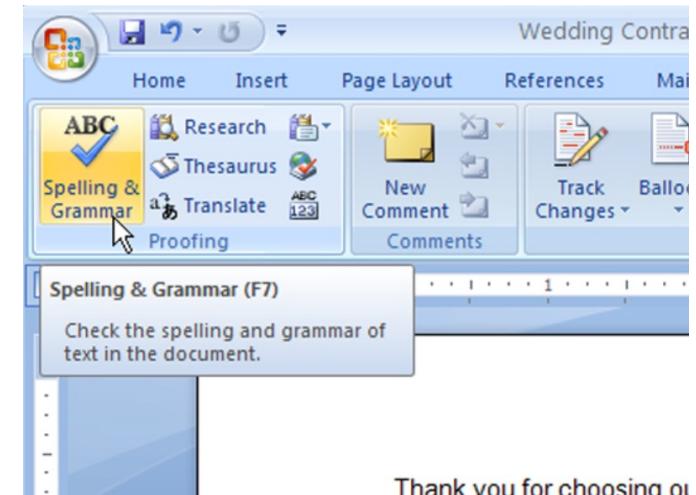
- What are patterns?
- Patterns: A set of items, subsequences, or substructures that occur frequently together (or strongly correlated) in a data set
- Patterns represent **intrinsic** and **important properties** of datasets



Frequent item set



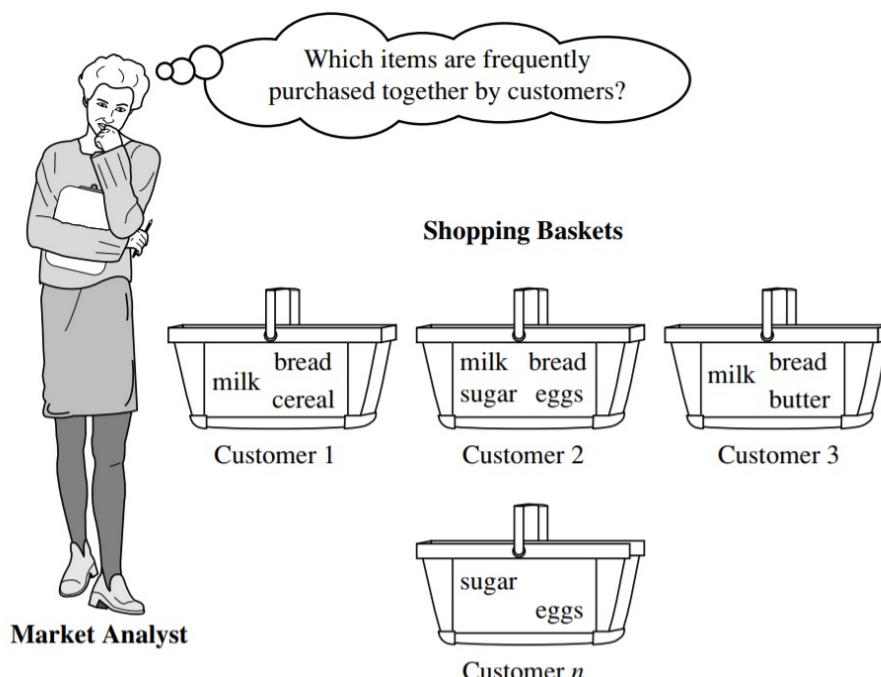
Frequent sequences



Frequent structures

What Is Pattern Discovery?

- **Pattern discovery:** Uncovering patterns from massive data sets
- It can answer questions such as:
 - What products were often purchased together?
 - What are the subsequent purchases after buying an iPad?



Pattern Discovery: Why Is It Important?

- Foundation for many essential data mining tasks
 - Association, correlation, and causality analysis
 - Mining **sequential**, structural (e.g., sub-graph) patterns
 - **Classification**: Discriminative pattern-based analysis
 - **Cluster** analysis: Pattern-based subspace clustering
- Broad applications
 - Market basket analysis, cross-marketing, catalog design, sale campaign analysis, Web log analysis, biological sequence analysis
 - Many types of data: spatiotemporal, multimedia, time-series, and stream data

Basic Concepts: Transactional Database

- ❑ Transactional Database (TDB)
- ❑ Each transaction is associated with an identifier, called a TID.
- ❑ May also have counts associated with each item sold

Tid	Items bought
1	Beer, Nuts, Diaper
2	Beer, Coffee, Diaper
3	Beer, Diaper, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Diaper, Eggs, Milk

Basic Concepts: k-Itemsets and Their Supports

- **Itemset:** A set of one or more items

$$I = \{ I_1, I_2, \dots, I_m \}$$

- **k-itemset:** An itemset containing k items:

$$X = \{x_1, \dots, x_k\}$$

- Ex. {Beer, Nuts, Diaper} is a 3-itemset

- **Absolute support (count)**

- $\text{sup}\{X\}$ = occurrences of an itemset X

- Ex. $\text{sup}\{\text{Beer}\} = 3$

- Ex. $\text{sup}\{\text{Diaper}\} = 4$

- Ex. $\text{sup}\{\text{Beer, Diaper}\} = 3$

- Ex. $\text{sup}\{\text{Beer, Eggs}\} = 1$

Tid	Items bought
1	Beer, Nuts, Diaper
2	Beer, Coffee, Diaper
3	Beer, Diaper, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Diaper, Eggs, Milk

- **Relative support**

- $s\{X\}$ = The fraction of transactions that contains X (i.e., the **probability** that a transaction contains X)

- Ex. $s\{\text{Beer}\} = 3/5 = 60\%$

- Ex. $s\{\text{Diaper}\} = 4/5 = 80\%$

- Ex. $s\{\text{Beer, Eggs}\} = 1/5 = 20\%$

Basic Concepts: Frequent Itemsets (Patterns)

- An itemset (or a pattern) X is *frequent* if the support of X is no less than a *minsup* threshold σ
- Let $\sigma = 50\%$ (σ : *minsup* threshold) for the given 5-transaction dataset
 - All the frequent 1-itemsets:
 - Beer: 3/5 (60%); Nuts: 3/5 (60%); Diaper: 4/5 (80%); Eggs: 3/5 (60%)
 - All the frequent 2-itemsets:
 - {Beer, Diaper}: 3/5 (60%)
 - All the frequent 3-itemsets?
 - None

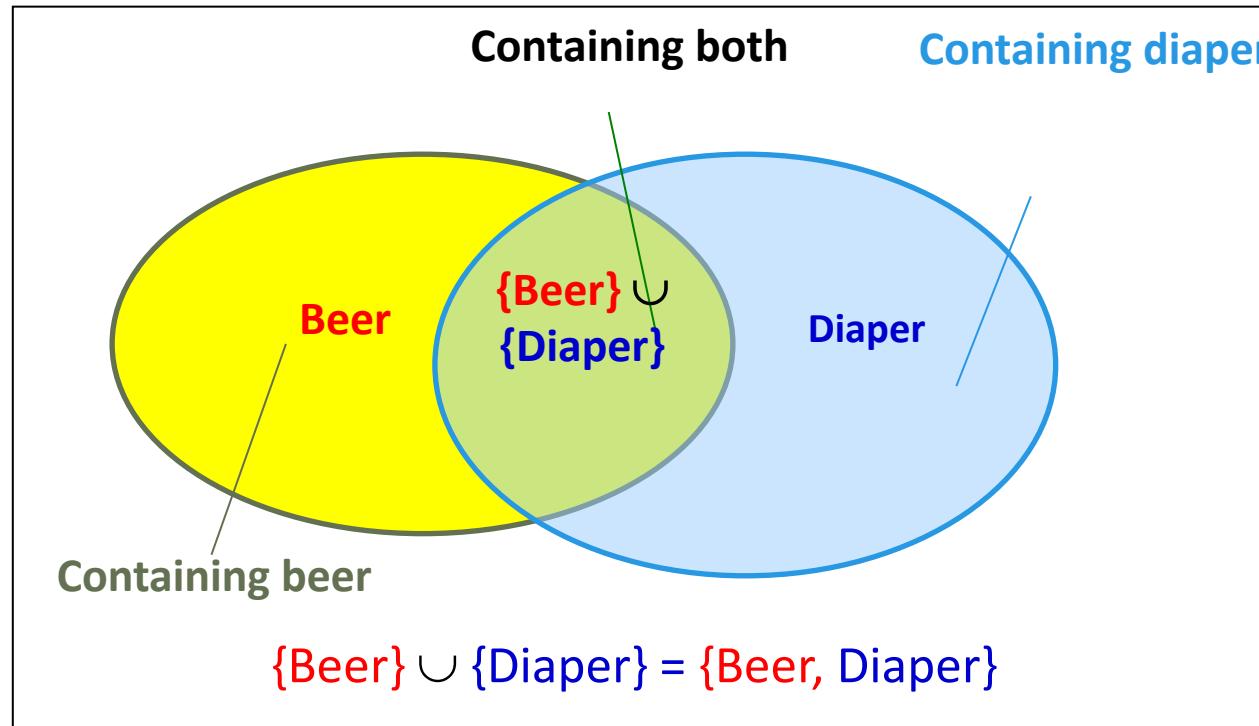


Tid	Items bought
1	Beer, Nuts, Diaper
2	Beer, Coffee, Diaper
3	Beer, Diaper, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Diaper, Eggs, Milk

- Why do these itemsets (shown on the left) form the complete set of frequent k -itemsets (patterns) for any k ?
- **Observation:** We may need an efficient method to mine a complete set of frequent patterns

From Frequent Itemsets to Association Rules

- Compared with itemsets, association rules can be more telling
 - Ex. *Diaper → Beer*
 - Buying diapers may likely lead to buying beers*



Note: $X \cup Y$: the union of two itemsets
■ The set contains both X and Y

Association Rules

- How do we compute the strength of an association rule $X \rightarrow Y$ (Both X and Y are itemsets)?
- We first compute the following two metrics, s and c .
 - **Support of $X \cup Y$**
 - Ex. $s\{\text{Diaper, Beer}\} = 3/5 = 0.6$ (i.e., 60%)
 - **Confidence of $X \rightarrow Y$**
 - The *conditional probability* that a transaction containing X also contains Y :
$$c = \text{sup}(X, Y) / \text{sup}(X)$$
 - Ex. $c = \text{sup}\{\text{Diaper, Beer}\}/\text{sup}\{\text{Diaper}\} = \frac{3}{4} = 0.75$
- In pattern analysis, we are often interested in those rules that dominate the database, and these two metrics ensure the popularity and correlation of X and Y .

Tid	Items bought
1	Beer, Nuts, Diaper
2	Beer, Coffee, Diaper
3	Beer, Diaper, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Diaper, Eggs, Milk

Mining Frequent Itemsets and Association Rules

- Association rule mining
 - Given two thresholds: $minsup$, $minconf$
 - Find **all** of the rules, $X \rightarrow Y$ (s, c) such that $s \geq minsup$ and $c \geq minconf$
- Let $minsup = 50\%$
 - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
 - Freq. 2-itemsets: {Beer, Diaper}: 3
- Let $minconf = 50\%$
 - $Beer \rightarrow Diaper$ (60%, 100%)
 - $Diaper \rightarrow Beer$ (60%, 75%)

(Q: Are these all the rules?)

Tid	Items bought
1	Beer, Nuts, Diaper
2	Beer, Coffee, Diaper
3	Beer, Diaper, Eggs
4	Nuts, Eggs, Milk
5	Nuts, Coffee, Diaper, Eggs, Milk



- Observations:
 - Mining association rules and mining frequent patterns are very close problems
 - Scalable methods are needed for mining large datasets

Challenge: There Are Too Many Frequent Patterns!

- A long pattern contains a combinatorial number of sub-patterns
- How many frequent itemsets does the following TDB₁ contain (minsup = 1)?

□ TDB₁: T₁: {a₁, ..., a₅₀}; T₂: {a₁, ..., a₁₀₀}

□ Let's have a try

1-itemsets: {a₁} : 2, {a₂} : 2, ..., {a₅₀} : 2, {a₅₁} : 1, ..., {a₁₀₀} : 1,

2-itemsets: {a₁, a₂} : 2, ..., {a₁, a₅₀} : 2, {a₁, a₅₁} : 1, ..., ..., {a₉₉, a₁₀₀} : 1,

..., ..., ..., ...

99-itemsets: {a₁, a₂, ..., a₉₉} : 1, ..., {a₂, a₃, ..., a₁₀₀} : 1

100-itemset: {a₁, a₂, ..., a₁₀₀} : 1

- The total number of frequent itemsets:

$$\binom{100}{1} + \binom{100}{2} + \binom{100}{3} + \dots + \binom{100}{100} = 2^{100} - 1$$

Too huge set for any one
to compute or store!



Expressing Patterns in Compressed Form

- How to reduce the redundancy of the list of all the frequent itemsets?
 - If $\{a_1, \dots, a_{99}\}$ and $\{a_1, \dots, a_{100}\}$ have the same support in the database, then we don't need to list both of them
- Solution 1: **Closed patterns**: A pattern (itemset) X is **closed** if X is *frequent*, and there exists *no super-pattern* Y $\supset X$, *with the same support* as X
 - Ex. TDB₁: T₁: $\{a_1, \dots, a_{50}\}$; T₂: $\{a_1, \dots, a_{100}\}$
 - Suppose *minsup* = 1. How many closed patterns does TDB₁ contain?
 - Two: P₁: “ $\{a_1, \dots, a_{50}\}$: 2”; P₂: “ $\{a_1, \dots, a_{100}\}$: 1”

Expressing Patterns in Compressed Form: Closed Patterns

- Closed pattern is a lossless compression of frequent patterns
- Reduces the # of patterns but does not lose the support information!
- Given $P_1: \{a_1, \dots, a_{50}\}: 2$; $P_2: \{a_1, \dots, a_{100}\}: 1$
- You will still be able to say: $\{a_2, \dots, a_{40}\}: 2$, $\{a_5, a_{51}\}: 1$

Expressing Patterns in Compressed Form: Maximal Patterns

- Solution 2: **Max-patterns**: A pattern X is a **maximal pattern** if X is frequent and there exists no frequent super-pattern $Y \supset X$
- Difference from close-patterns?
 - Do not care the real support of the sub-patterns of a max-pattern
 - Let Transaction DB TDB_1 : $T_1: \{a_1, \dots, a_{50}\}$; $T_2: \{a_1, \dots, a_{100}\}$
 - Suppose $minsup = 1$. How many max-patterns does TDB_1 contain?
 - One: P: “ $\{a_1, \dots, a_{100}\}$: 1”

Expressing Patterns in Compressed Form: Maximal Patterns

- Maximal pattern is a **lossy compression!**
 - We only know a subset of the max-pattern P , $\{a_1, \dots, a_{40}\}$, is frequent
 - But we do not know the real support of $\{a_1, \dots, a_{40}\}$, ..., any more!
- Thus in many applications, mining closed-patterns is more desirable than mining maximal patterns

Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary



Efficient Pattern Mining Methods

- The Downward Closure Property of Frequent Patterns
 - The Apriori Algorithm
 - Extensions or Improvements of Apriori
- Mining Frequent Patterns by Exploring Vertical Data Format
- FP-Growth: A Frequent Pattern-Growth Approach
- Mining Closed Patterns

The Downward Closure Property of Frequent Patterns

- Frequent itemset: $\{a_1, \dots, a_{50}\}$
 - Subsets are all frequent: $\{a_1\}, \{a_2\}, \dots, \{a_{50}\}, \{a_1, a_2\}, \dots, \{a_1, \dots, a_{49}\}, \dots$
- Downward closure (Apriori): Any subset of a frequent itemset must be frequent
 - If $\{\text{beer, diaper, nuts}\}$ is frequent, so is $\{\text{beer, diaper}\}$
 - If any subset of an itemset S is infrequent, then there is no chance for S to be frequent.



A sharp knife for pruning!

Apriori Pruning and Scalable Mining Methods

- Apriori pruning principle: If there is any itemset which is infrequent, its superset should not even be generated! (Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)
- Scalable mining Methods: Three major approaches
 - Level-wise, join-based approach: Apriori (Agrawal & Srikant@VLDB'94)
 - Vertical data format approach: Eclat (Zaki, Parthasarathy, Ogihsara, Li @KDD'97)
 - Frequent pattern projection and growth: FPgrowth (Han, Pei, Yin @SIGMOD'00)

Apriori: A Candidate Generation & Test Approach

- Outline of Apriori (level-wise, candidate generation and test)
 - Scan DB once to get frequent 1-itemset
 - Repeat
 - Generate length-(k+1) candidate itemsets from length-k frequent itemsets
 - Test the candidates against DB to find frequent (k+1)-itemsets
 - Set $k := k + 1$
 - Until no frequent or candidate set can be generated
 - Return all the frequent itemsets derived

The Apriori Algorithm (Pseudo-Code)

C_k : Candidate itemset of size k

F_k : Frequent itemset of size k

K := 1;

F_k := {frequent items}; // frequent 1-itemset

While ($F_k \neq \emptyset$) **do {** // when F_k is non-empty

C_{k+1} := candidates generated from F_k ; // candidate generation

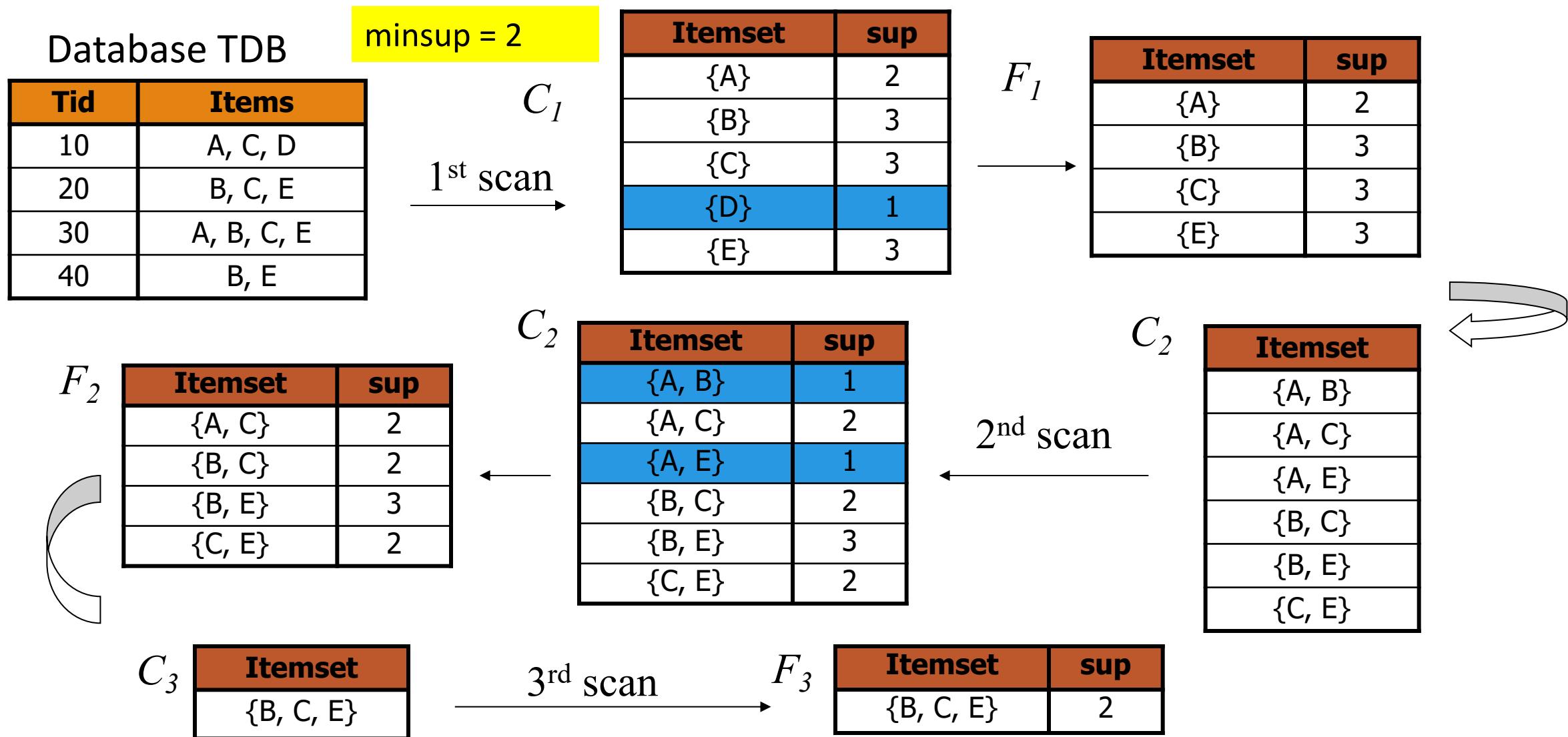
Derive F_{k+1} by counting candidates in C_{k+1} with respect to TDB at minsup;

k := k + 1

}

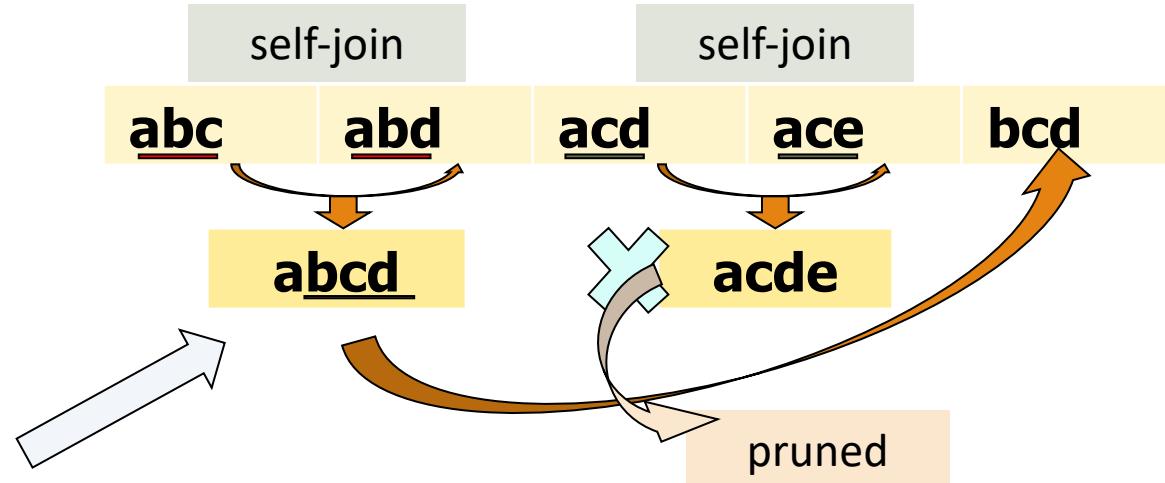
return $\cup_k F_k$ // return F_k generated at each level

The Apriori Algorithm—An Example



Apriori: Implementation Tricks

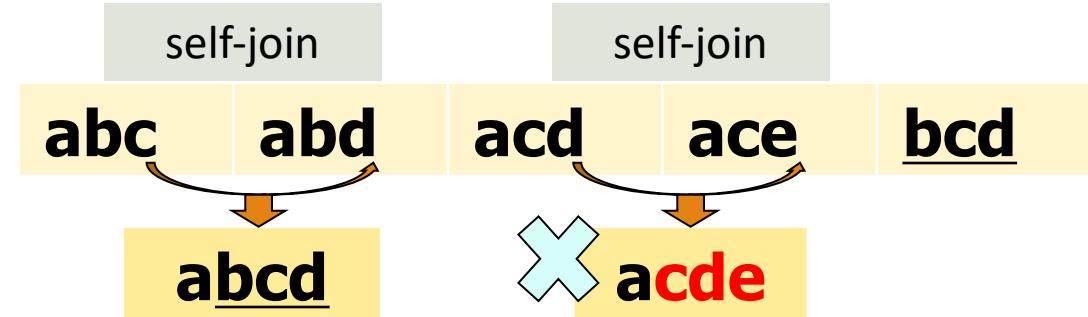
- ❑ How to generate candidates?
 - ❑ Step 1: self-joining F_k
 - ❑ Step 2: pruning
- ❑ Example of candidate-generation
 - ❑ $F_3 = \{abc, abd, acd, ace, bcd\}$
 - ❑ Self-joining: $F_3 * F_3$
 - ❑ $abcd$ from abc and abd
 - ❑ $acde$ from acd and ace
 - ❑ Pruning:
 - ❑ $acde$ is removed because ade is not in F_3
 - ❑ $C_4 = \{abcd\}$



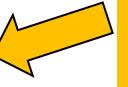
Candidate Generation (Pseudo-Code)

- Suppose the items in F_{k-1} are listed in an order
- // Step 1: Joining
 - for each p in F_{k-1}
 - for each q in F_{k-1}
 - if $p.item_1 = q.item_1, \dots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$ {
 $c = \text{join}(p, q)$ }
 - // Step 2: pruning
 - if `has_infrequent_subset(c, Fk-1)`
 `continue` // prune
 - else add c to C_k

}



Apriori: Improvements and Alternatives

- ❑ Reduce passes of transaction database scans
 - ❑ **Partitioning** (e.g., Savasere, et al., 1995)  To be discussed in subsequent slides
 - ❑ Dynamic itemset counting (Brin, et al., 1997)
- ❑ Shrink the number of candidates
 - ❑ **Hashing** (e.g., DHP: Park, et al., 1995)  To be discussed in subsequent slides
 - ❑ Pruning by support lower bounding (e.g., Bayardo 1998)
 - ❑ Sampling (e.g., Toivonen, 1996)
- ❑ Exploring special data structures
 - ❑ Tree projection (Agarwal, et al., 2001)
 - ❑ H-miner (Pei, et al., 2001)
 - ❑ Hypercube decomposition (e.g., LCM: Uno, et al., 2004)

Partitioning: Scan Database Only Twice

- Theorem: *Any itemset that is potentially frequent in TDB must be frequent in at least one of the partitions of TDB*

The diagram illustrates the decomposition of a database TDB into k partitions. It shows a large rectangle labeled TDB at the bottom right, which is partitioned into k smaller rectangles labeled $TDB_1, TDB_2, \dots, TDB_k$ from left to right. Between the first two partitions, there is a plus sign ($+$). Between the second and third partitions, there is a plus sign ($+$) followed by three small orange dots above the line, indicating continuation. Below the first partition, there is a plus sign ($+$) followed by the text "sup₁(X) > σ|TDB₁| or". Below the second partition, there is a plus sign ($+$) followed by the text "sup₂(X) > σ|TDB₂| or ...". Below the third partition, there is a plus sign ($+$) followed by the text "sup_k(X) > σ|TDB_k|". To the left of the first partition, there is a yellow diagonal box containing the text "Here is the proof!".

Partitioning: Scan Database Only Twice

- ❑ Method: Scan DB **twice** (A. Savasere, E. Omiecinski and S. Navathe, *VLDB'95*)
 - ❑ Scan 1: Partition database so that each partition can fit in main memory (why?)
 - ❑ Mine local frequent patterns in this partition
 - ❑ Scan 2: Consolidate global frequent patterns
 - ❑ Find global frequent itemset candidates (those frequent in at least one partition)
 - ❑ Find the true frequency of those candidates, by scanning TDB_i one more time

Direct Hashing and Pruning (DHP)

- ❑ Hashing: $v = \text{hash}(\text{itemset})$
- ❑ 1st scan: When counting the 1-itemset, hash 2-itemset to calculate the bucket count
- ❑ Example: At the 1st scan of TDB, count 1-itemset, and hash 2-itemsets in the transaction to its bucket
 - ❑ {ab, ad, ce}
 - ❑ {bd, be, de}
 - ❑ ...
- ❑ At the end of the first scan,
 - ❑ if $\text{minsup} = 80$, remove ab, ad, ce, since $\text{count}\{\text{ab, ad, ce}\} < 80$

V might be same for different itemset

Itemsets	Count
{ab, ad, ce}	35
{bd, be, de}	298
.....	...
{yz, qs, wt}	58

Hash Table

Check the minsup

Direct Hashing and Pruning (DHP)

Table 5.1: Transactional Data for an
AllElectronics Branch

<i>TID</i>	<i>List of item_IDs</i>
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

L_1

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

H_2

Hash for {Ix,Iy}

Create hash table H_2
using hash function

$$h(x, y) = ((\text{order of } x) \times 10 + (\text{order of } y)) \bmod 7$$



bucket address	0	1	2	3	4	5	6
bucket count	2	2	4	2	2	4	4
bucket contents	{I1, I4}	{I1, I5}	{I2, I3}	{I2, I4}	{I2, I5}	{I1, I2}	{I1, I3}
	{I3, I5}	{I1, I5}	{I2, I3}	{I2, I4}	{I2, I5}	{I1, I2}	{I1, I3}
			{I2, I3}			{I1, I2}	{I1, I3}
				{I2, I3}		{I1, I2}	{I1, I3}

Exploring Vertical Data Format: ECLAT

- ❑ ECLAT (Equivalence Class Transformation): A **depth-first search** algorithm using set intersection [Zaki et al. @KDD'97]

A transaction DB in Horizontal Data Format

Tid	Itemset
10	a, c, d, e
20	a, b, e
30	b, c, e

- ❑ Vertical format

- ❑ Properties of Tid-Lists

- ❑ $t(X) = t(Y)$: X and Y always happen together (e.g., $t(ac) = t(d)$)
 - ❑ $t(X) \subset t(Y)$: transaction having X always has Y (e.g., $t(ac) \subset t(ce)$)

- ❑ Frequent patterns: vertical set intersections

- ❑ Using **diffset** to accelerate mining

- ❑ Only keep track of differences of tids

- ❑ $t(e) = \{T_{10}, T_{20}, T_{30}\}$, $t(ce) = \{T_{10}, T_{30}\} \rightarrow \text{Diffset}(ce, e) = \{T_{20}\}$

The transaction DB in Vertical Data Format

Item	TidList
a	10, 20
b	20, 30
c	10, 30
d	10
e	10, 20, 30

Exploring Vertical Data Format: ECLAT

Table 5.3: The Vertical Data Format of the Transaction Data Set D of Table 5.1

itemset *TID_set*

I1	{T100, T400, T500, T700, T800, T900}
I2	{T100, T200, T300, T400, T600, T800, T900}
I3	{T300, T500, T600, T700, T800, T900}
I4	{T200, T400}
I5	{T100, T800}

Table 5.4: 2-Itemsets in Vertical Data

itemset *TID_set*

{I1, I2}	{T100, T400, T800, T900}
{I1, I3}	{T500, T700, T800, T900}
{I1, I4}	{T400}
{I1, I5}	{T100, T800}
{I2, I3}	{T300, T600, T800, T900}
{I2, I4}	{T200, T400}
{I2, I5}	{T100, T800}
{I3, I5}	{T800}

Table 5.5: 3-Itemsets in Vertical Data Format

<i>itemset</i>	<i>TID_set</i>
{I1, I2, I3}	{T800, T900}
{I1, I2, I5}	{T100, T800}

Why Mining Frequent Patterns by Pattern Growth?

- Apriori: A *breadth-first search* mining algorithm
 - First find the complete set of frequent k-itemsets
 - Then derive frequent (k+1)-itemset candidates
 - Scan DB again to find true frequent (k+1)-itemsets

Why Mining Frequent Patterns by Pattern Growth?

- ❑ Motivation for a different mining methodology
 - ❑ Can we develop a *depth-first search* mining algorithm?
 - ❑ For a frequent itemset ρ , can subsequent search be confined to only those transactions that containing ρ ?
- ❑ Such thinking leads to a frequent pattern growth approach:
 - ❑ **FPGrowth** (J. Han, J. Pei, Y. Yin, “Mining Frequent Patterns without Candidate Generation,” SIGMOD 2000)

Prerequisite: Find frequent 1-itemset

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

1. Scan DB once, find single item frequent pattern:

Let min_support = 3

f:4, a:3, c:4, b:3, m:3, p:3

2. Sort frequent items in frequency descending order, f-list

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

Example: Construct FP-tree from a Transaction DB

TID	Items in the Transaction	Ordered, frequent itemlist
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

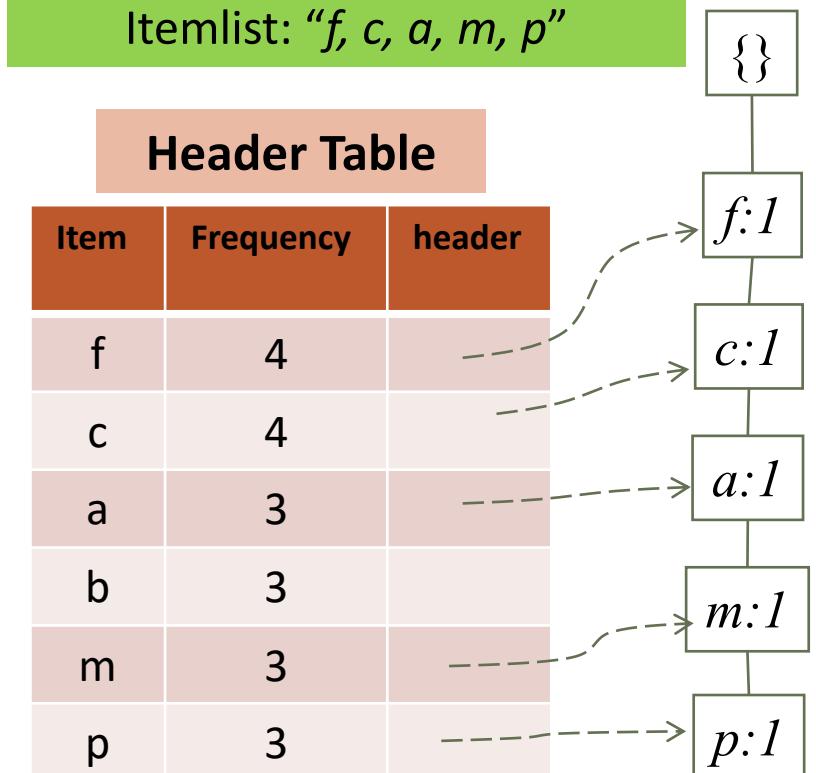
3. Scan DB again, find the ordered frequent itemlist for each transaction

Example: Construct FP-tree from a Transaction DB

TID	Items in the Transaction	Ordered, frequent itemlist
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

4. For each transaction, insert the ordered frequent itemlist into an FP-tree, with shared sub-branches merged, counts accumulated

After inserting the 1st frequent Itemlist: "f, c, a, m, p"

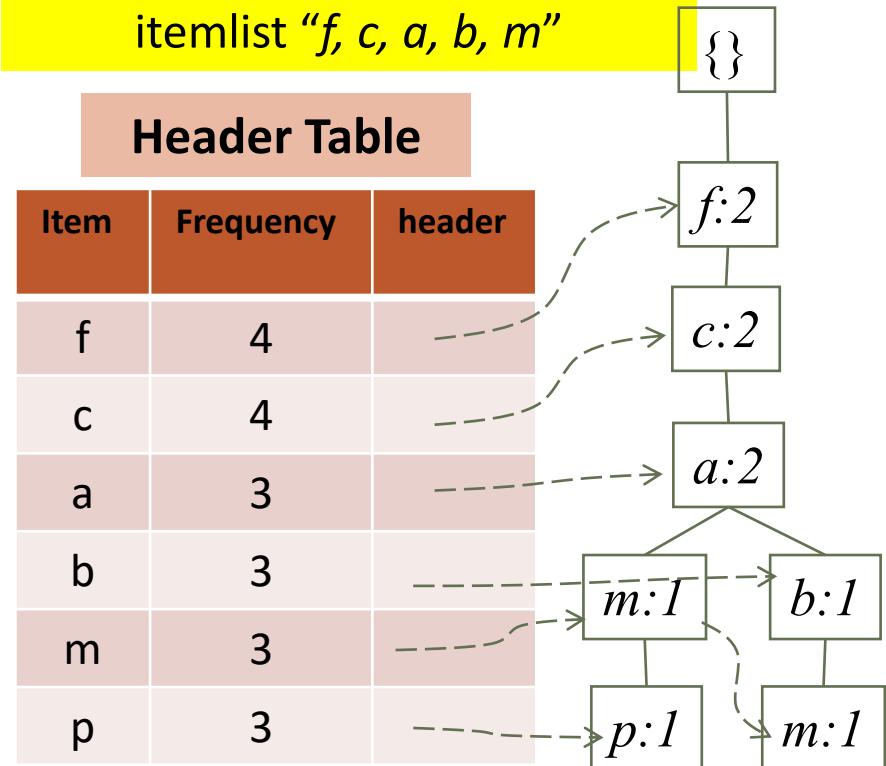


Example: Construct FP-tree from a Transaction DB

TID	Items in the Transaction	Ordered, frequent itemlist
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

4. For each transaction, insert the ordered frequent itemlist into an FP-tree, with shared sub-branches merged, counts accumulated

After inserting the 2nd frequent itemlist “f, c, a, b, m”

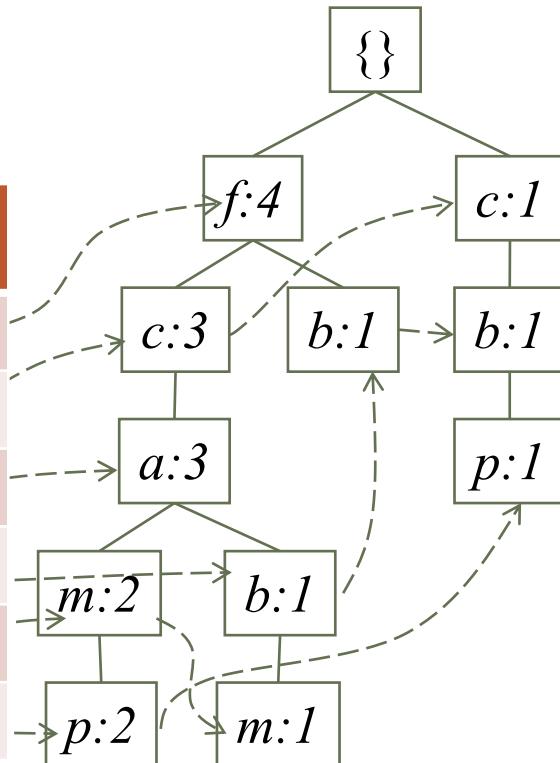


Example: Construct FP-tree from a Transaction DB

TID	Items in the Transaction	Ordered, frequent itemlist	After inserting all the frequent itemlists
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	

Header Table

Item	Frequency	header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



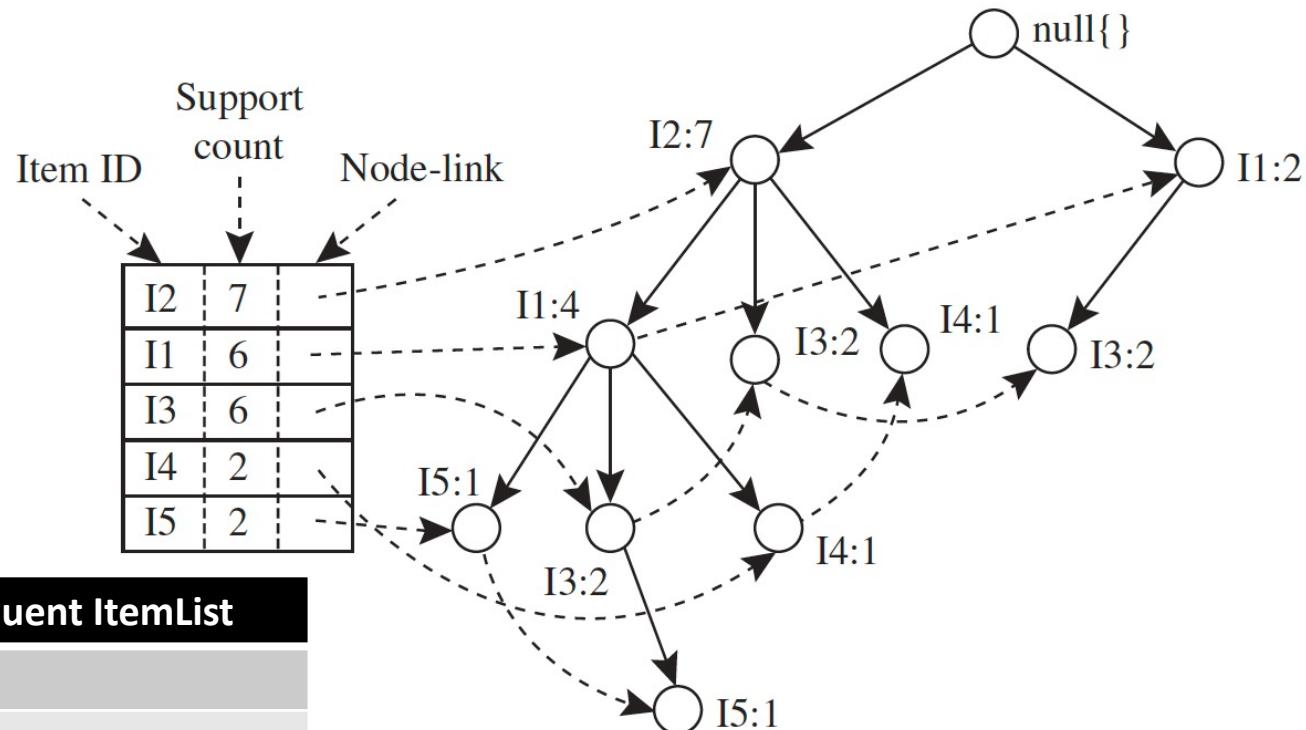
4. For each transaction, insert the ordered frequent itemlist into an FP-tree, with shared sub-branches merged, counts accumulated

Example: Construct FP-tree from a Transaction DB

Table 5.1: Transactional Data for an
AllElectronics Branch

<i>TID</i>	<i>List of item_IDs</i>
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

TID	Ordered, Frequent ItemList
T100	I2, I1, I5
T200	I2, I4
T300	I2, I3
T400	I2, I1, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I2, I1, I3, I5
T900	I2, I1, I3

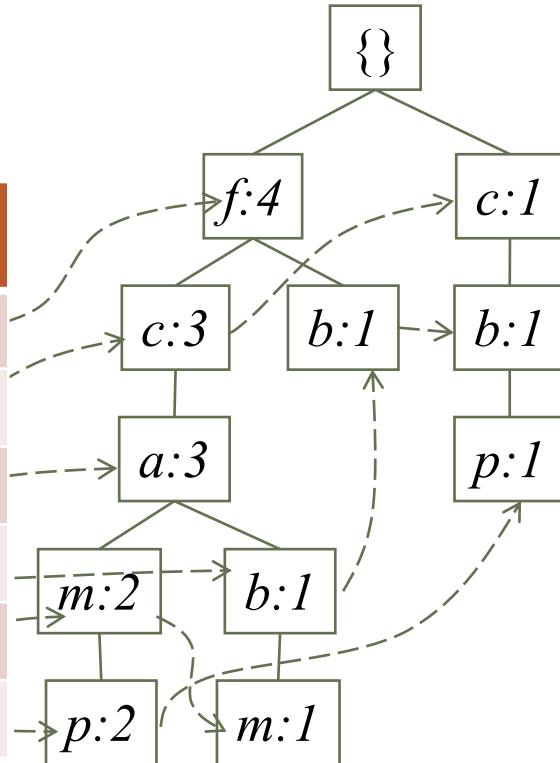


Example: Construct FP-tree from a Transaction DB

TID	Items in the Transaction	Ordered, frequent itemlist	After inserting all the frequent itemlists
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	

Header Table

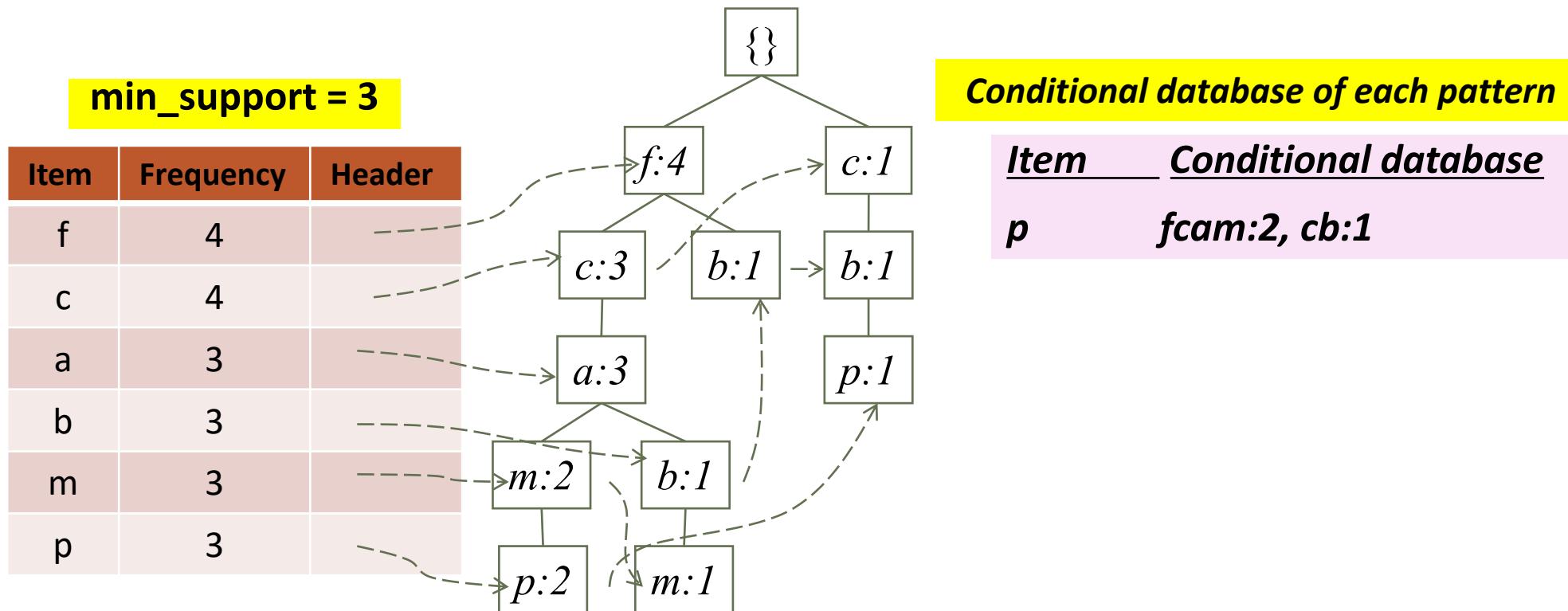
Item	Frequency	header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



4. For each transaction, insert the ordered frequent itemlist into an FP-tree, with shared sub-branches merged, counts accumulated

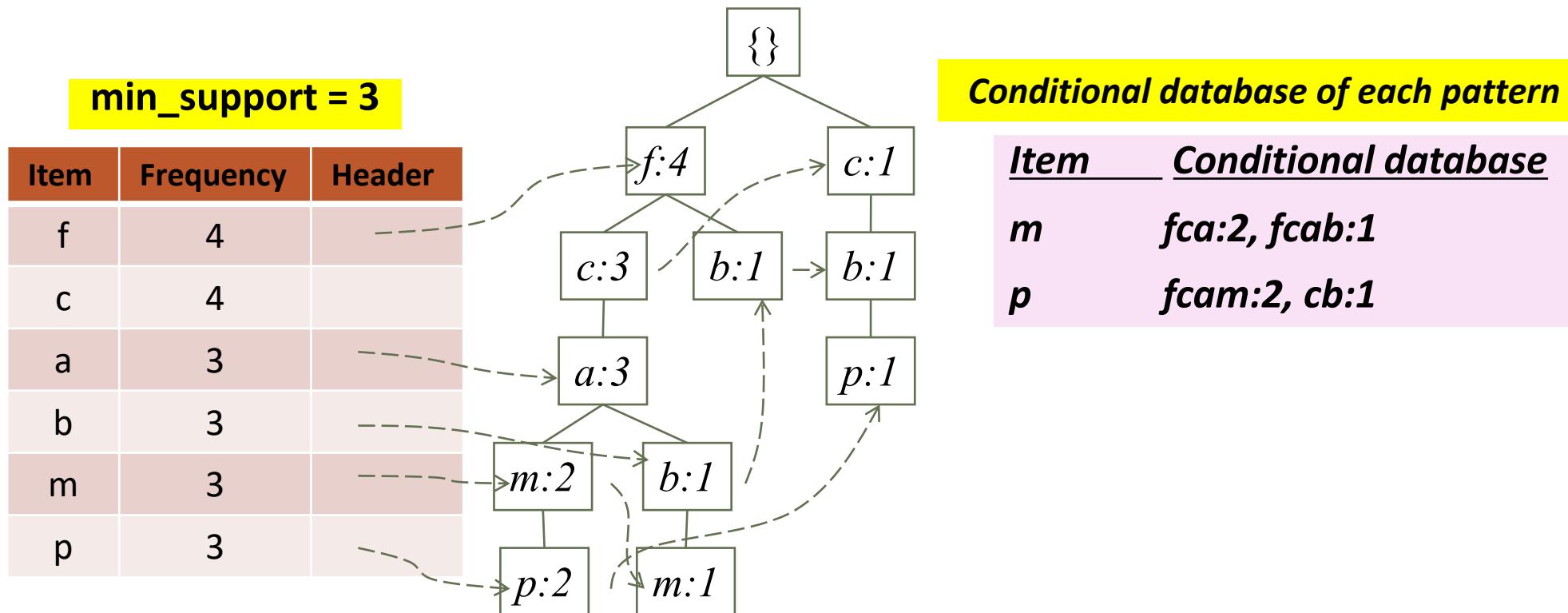
Mining FP-Tree: Divide and Conquer Based on Patterns and Data

- Pattern mining can be partitioned according to current patterns
 - We start to calculate the conditional database from bottom to top (from the least frequent item)
 - Conditional database: the database under the condition that p exists
 - p 's conditional database (Patterns containing p): $fcam:2, cb:1$



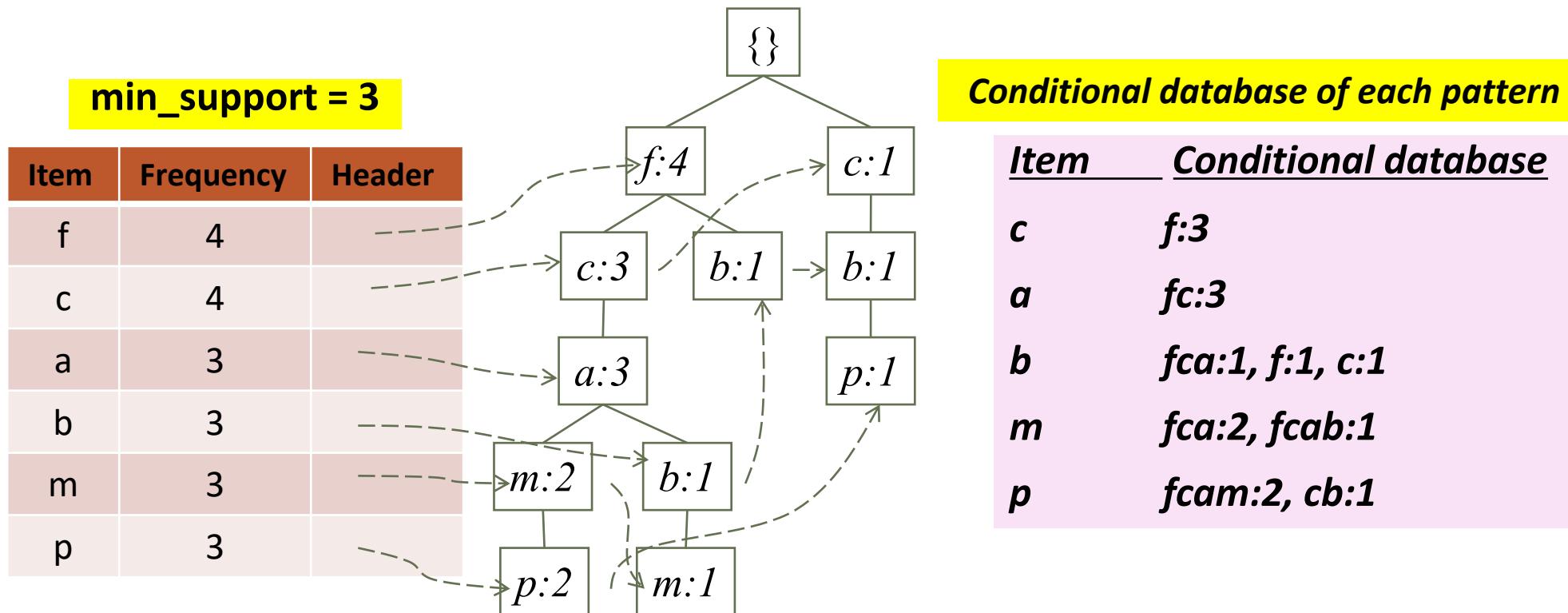
Mining FP-Tree: Divide and Conquer Based on Patterns and Data

- ❑ p 's conditional database (Patterns containing p): $fcam:2, cb:1$
- ❑ After calculating p 's conditional database, we calculate m 's conditional database



Mining FP-Tree: Divide and Conquer Based on Patterns and Data

- Repeat and calculate the conditional database of b , a , and c
- Since f is the most frequent item, we don't need to compute its conditional dataset



Mine Each Conditional Database Recursively

min_support = 3

p fcам:2, cb: 1

Constructing p's 1 itemlist,
sorted by frequency

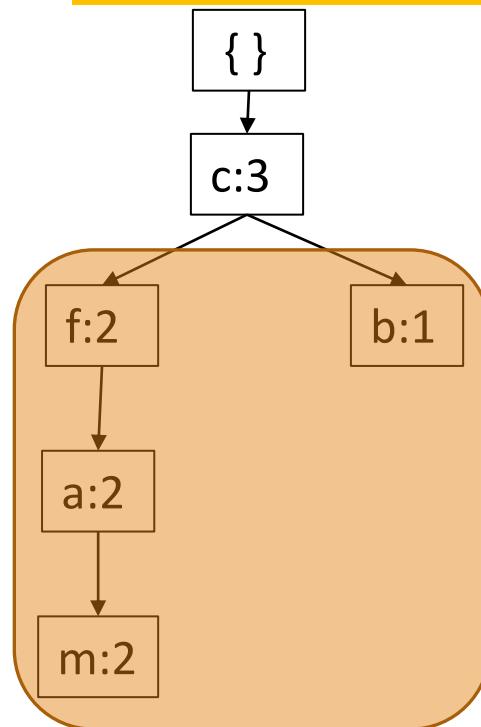
Item	Frequency
c	3
f	2
a	2
m	2
b	1

Original Ordered

f, c, a, m c, f, a, m
 c, b c, b

- For each conditional database
 - Mine single-item patterns
 - Construct its FP-tree & mine it

Conditional FP-tree for p



Frequent pattern cp: 3

Mine Each Conditional Database Recursively

min_support = 3

m fca:2, fcab: 1

Constructing m's 1 itemlist,
sorted by frequency

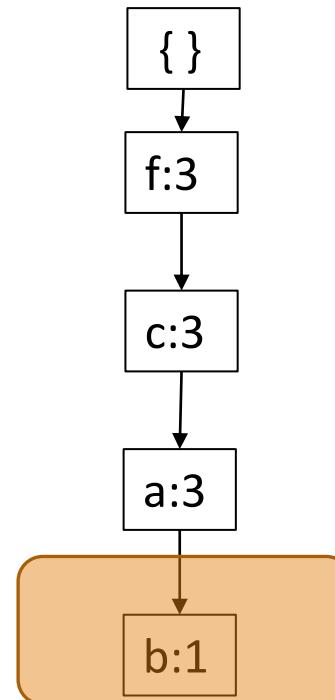
Item	Frequency
f	3
c	3
a	3
b	1

Original Ordered

f, c, a *f, c, a*
f, c, a, b *f, c, a, b*

- For each conditional database
 - Mine single-item patterns
 - Construct its FP-tree & mine it

Conditional FP-tree for **m**



Frequent pattern fcam: 3

Form conditional database:
a**m** **f****c**: 3

Mine Each Conditional Database Recursively

min_support = 3

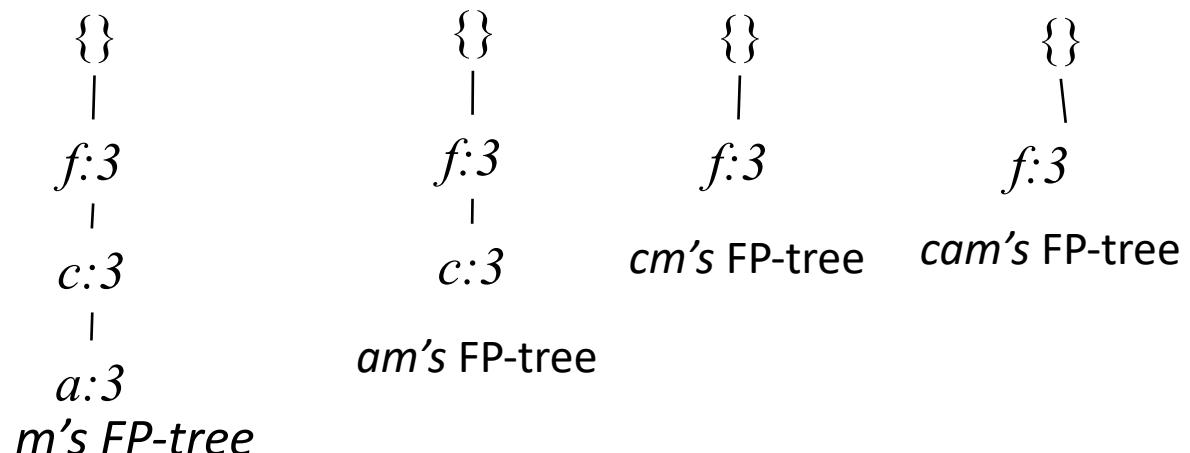
Conditional Data Bases

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

- For each conditional database

- Mine single-item patterns
- Construct its FP-tree & mine it

e.g., mining m's FP-tree

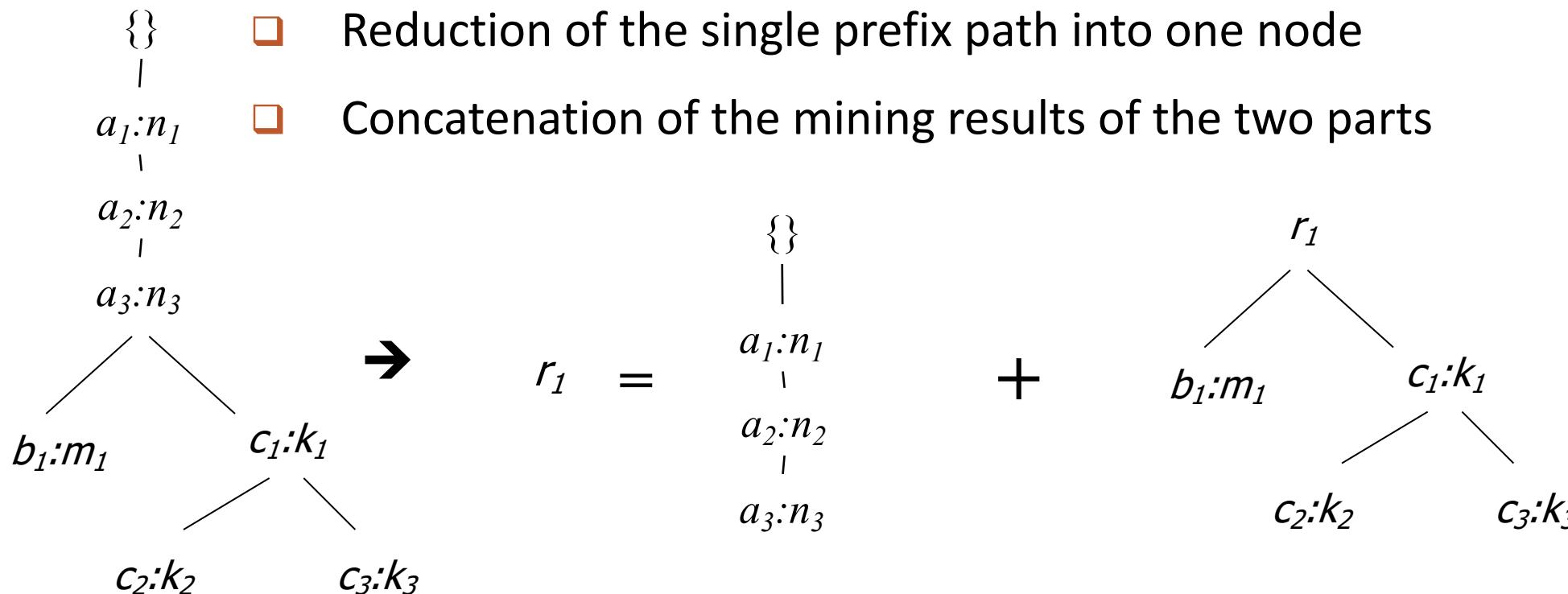


Actually, for single branch FP-tree, all the frequent patterns can be generated in one shot

m: 3
fm: 3, cm: 3, am: 3
fcm: 3, fam:3, cam: 3
fcam: 3

A Special Case: Single Prefix Path in FP-tree

- ❑ Suppose a (conditional) FP-tree T has a shared single prefix-path P
- ❑ Mining can be decomposed into two parts

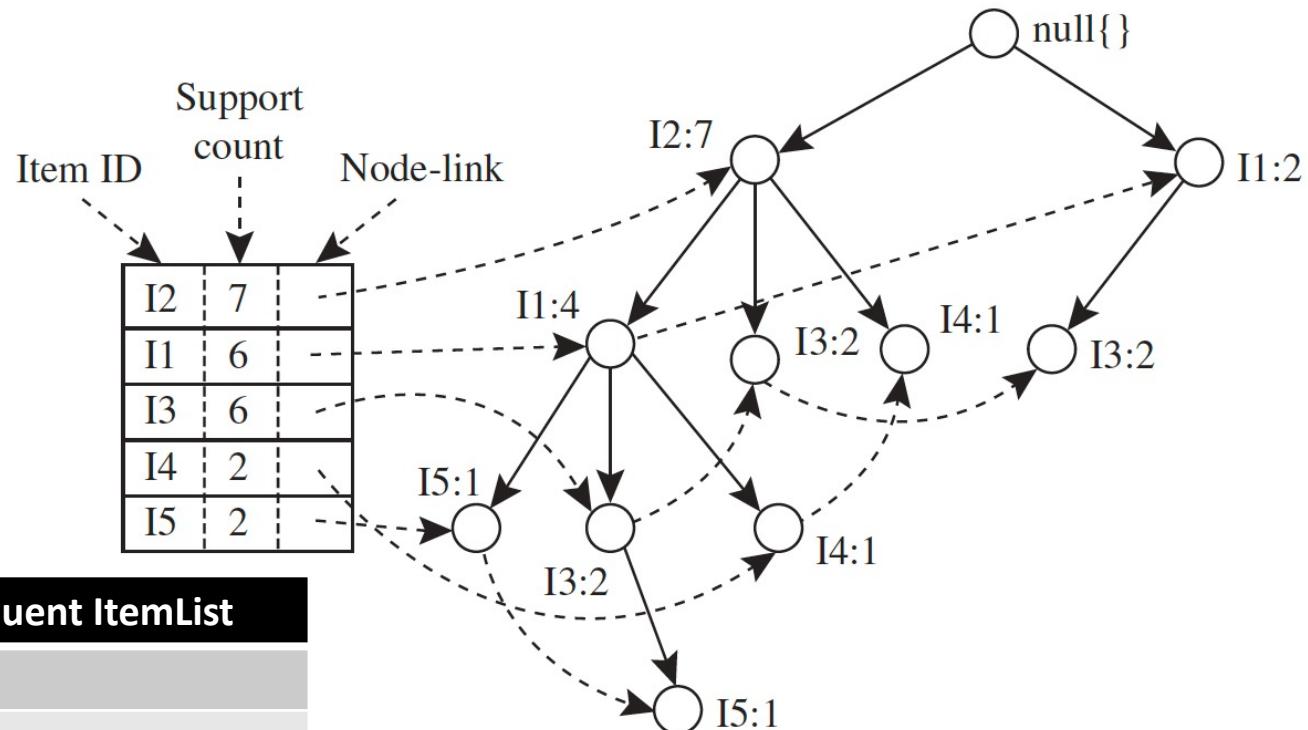


Example: Construct FP-tree from a Transaction DB

Table 5.1: Transactional Data for an
AllElectronics Branch

<i>TID</i>	<i>List of item_IDs</i>
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

TID	Ordered, Frequent ItemList
T100	I2, I1, I5
T200	I2, I4
T300	I2, I3
T400	I2, I1, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I2, I1, I3, I5
T900	I2, I1, I3



Example: Mining with Conditional FP-trees

<i>Item</i>	<i>Conditional Pattern Base</i>	<i>Conditional FP-tree</i>	<i>Frequent Patterns Generated</i>
I5	$\{\{I2, I1: 1\}, \{I2, I1, I3: 1\}\}$	$\langle I2: 2, I1: 2 \rangle$	$\{I2, I5: 2\}, \{I1, I5: 2\}, \{I2, I1, I5: 2\}$
I4	$\{\{I2, I1: 1\}, \{I2: 1\}\}$	$\langle I2: 2 \rangle$	$\{I2, I4: 2\}$
I3	$\{\{I2, I1: 2\}, \{I2: 2\}, \{I1: 2\}\}$	$\langle I2: 4, I1: 2 \rangle, \langle I1: 2 \rangle$	$\{I2, I3: 4\}, \{I1, I3: 4\}, \{I2, I1, I3: 2\}$
I1	$\{\{I2: 4\}\}$	$\langle I2: 4 \rangle$	$\{I2, I1: 4\}$

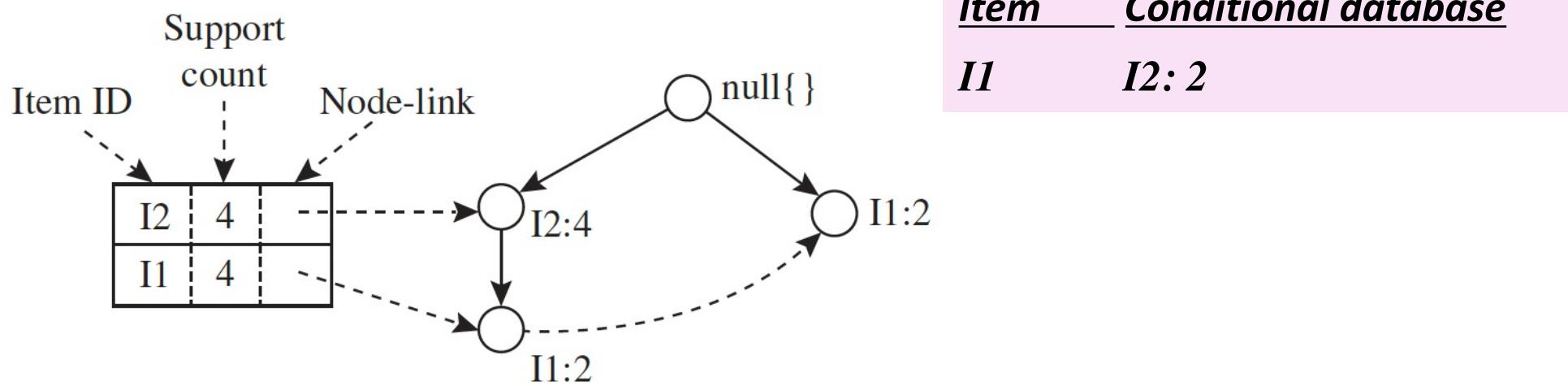


Figure 5.8: The conditional FP-tree associated with the conditional node I3.

FPGrowth: Mining Frequent Patterns by Pattern Growth

- Essence of frequent pattern growth (FPGrowth) methodology
- Find frequent single items and partition the database based on each such single item pattern
- Recursively grow frequent patterns by doing the above for each *partitioned database* (also called the pattern's *conditional database*)
- To facilitate efficient processing, an efficient data structure, FP-tree, can be constructed

FPGrowth: Mining Frequent Patterns by Pattern Growth

- Mining becomes
 - Recursively construct and mine (conditional) FP-trees
 - Until the resulting FP-tree is empty, or until it contains only one path—single path will generate all the combinations of its sub-paths, each of which is a frequent pattern

Mining Closed and Maximal Patterns

- Closed frequent itemsets
 - Find all frequent itemsets (FI), then remove itemsets which are (i) proper subsets of, and (ii) has same support as another FI
 - Can be time consuming
- More efficient approaches
 - If all transactions containing FI X also contains an itemset Y but not any proper superset of Y, then $X \cup Y$ is a frequent closed itemset
 - No need to search for any itemset containing X and Y
- Maximal frequent itemsets
 - Extensions of approaches for Closed FIs

Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary



Pattern Evaluation

- Limitation of the Support-Confidence Framework
- Interestingness Measures: Lift and χ^2
- Null-Invariant Measures
- Comparison of Interestingness Measures

How to Judge if a Rule/Pattern Is Interesting?

- Pattern-mining will generate a large set of patterns/rules
 - Not all the generated patterns/rules are interesting
- **Interestingness measures:** Objective vs. subjective
 - **Objective** interestingness measures
 - Support, confidence, correlation, ...
 - **Subjective** interestingness measures:
 - Different users may judge interestingness differently
 - Let a user specify
 - Query-based: Relevant to a user's particular request
 - Judge against one's knowledge-base
 - unexpected, freshness, timeliness

Limitation of the Support-Confidence Framework

- Are s and c interesting in association rules: “ $A \Rightarrow B$ ” [s, c]?
- Example: Suppose one school may have the following statistics on # of students who may play basketball and/or eat cereal:

	play-basketball	not play-basketball	sum (row)
eat-cereal	400	350	750
not eat-cereal	200	50	250
sum(col.)	600	400	1000

2-way contingency table

- Association rule mining may generate the following:
 - $play\text{-}basketball \Rightarrow eat\text{-}cereal$ [40%, 66.7%] (higher s & c)
- But this strong association rule is misleading: The overall % of students eating cereal is 75% > 66.7%, a more telling rule:
 - $\neg play\text{-}basketball \Rightarrow eat\text{-}cereal$ [35%, 87.5%] (high s & c)

Interestingness Measure: Lift

- Measure of dependent/correlated events: **lift**

$$lift(B, C) = \frac{c(B \rightarrow C)}{s(C)} = \frac{s(B \cup C)}{s(B) \times s(C)}$$

- Lift(B, C) may tell how B and C are correlated

- Lift(B, C) = 1: B and C are independent

- > 1: positively correlated

- < 1: negatively correlated

- For our example,

$$lift(B, C) = \frac{400/1000}{600/1000 \times 750/1000} = 0.89$$

$$lift(B, \neg C) = \frac{200/1000}{600/1000 \times 250/1000} = 1.33$$

- Thus, B and C are negatively correlated since $lift(B, C) < 1$;
- B and $\neg C$ are positively correlated since $lift(B, \neg C) > 1$

Lift is more telling than s & c

	B	$\neg B$	Σ_{row}
C	400	350	750
$\neg C$	200	50	250
$\Sigma_{\text{col.}}$	600	400	1000

Interestingness Measure: χ^2

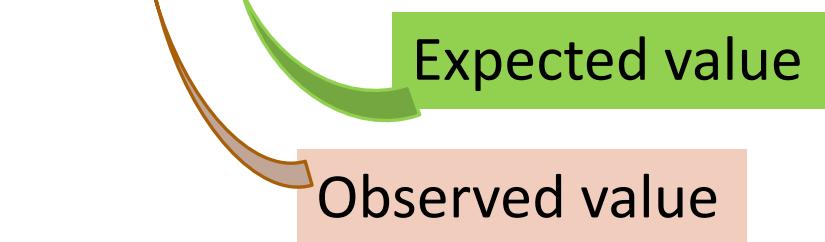
- Another measure to test correlated events: χ^2

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

- For the table on the right,

$$\chi^2 = \frac{(400 - 450)^2}{450} + \frac{(350 - 300)^2}{300} + \frac{(200 - 150)^2}{150} + \frac{(50 - 100)^2}{100} = 55.56$$

	B	$\neg B$	Σ_{row}
C	400 (450)	350 (300)	750
$\neg C$	200 (150)	50 (100)	250
Σ_{col}	600	400	1000



- Lookup χ^2 distribution table => B, C are correlated
- χ^2 -test shows B and C are negatively correlated since the expected value is 450 but the observed is only 400
- Thus, χ^2 is also more telling than the support-confidence framework

Lift and χ^2 : Are They Always Good Measures?

- ❑ Null transactions: Transactions that contain neither B nor C
- ❑ Let's examine the new dataset D
 - ❑ BC (100) is much rarer than B¬C (1000) and ¬BC (1000), but there are many ¬B¬C (100000)
 - ❑ Unlikely B & C will happen together!
 - ❑ But, Lift(B, C) = 8.44 >> 1 (Lift shows B and C are strongly positively correlated!)
 - ❑ $\chi^2 = 670$: Observed(BC) >> expected value (11.85)
 - ❑ *Too many null transactions may “spoil the soup”!*



	B	$\neg B$	Σ_{row}
C	100	1000	1100
$\neg C$	1000	100000	101000
$\Sigma_{\text{col.}}$	1100	101000	102100

null transactions

Contingency table with expected values added

	B	$\neg B$	Σ_{row}
C	100 (11.85)	1000	1100
$\neg C$	1000 (988.15)	100000	101000
$\Sigma_{\text{col.}}$	1100	101000	102100

Interestingness Measures & Null-Invariance

- *Null invariance* means: The number of null transactions does not matter.
Does not change the measure value.
- A few interestingness measures: Some are null invariant

Measure	Definition	Range	Null-Invariant?
$\chi^2(A, B)$	$\sum_{i,j} \frac{(e(a_i, b_j) - o(a_i, b_j))^2}{e(a_i, b_j)}$	$[0, \infty]$	No
$Lift(A, B)$	$\frac{s(A \cup B)}{s(A) \times s(B)}$	$[0, \infty]$	No
$Allconf(A, B)$	$\frac{s(A \cup B)}{\max\{s(A), s(B)\}}$	$[0, 1]$	Yes
$Jaccard(A, B)$	$\frac{s(A \cup B)}{s(A) + s(B) - s(A \cup B)}$	$[0, 1]$	Yes
$Cosine(A, B)$	$\frac{s(A \cup B)}{\sqrt{s(A) \times s(B)}}$	$[0, 1]$	Yes
$Kulczynski(A, B)$	$\frac{1}{2} \left(\frac{s(A \cup B)}{s(A)} + \frac{s(A \cup B)}{s(B)} \right)$	$[0, 1]$	Yes
$MaxConf(A, B)$	$\max\left\{\frac{s(A \cup B)}{s(A)}, \frac{s(A \cup B)}{s(B)}\right\}$	$[0, 1]$	Yes

Let

$$p = \frac{s(A \cup B)}{s(A)} = P(B|A)$$

$$q = \frac{s(A \cup B)}{s(B)} = P(A|B)$$

p, q are null invariant

Essentially min,
max, mean variants
of p, q

Null Invariance: An Important Property

- Why is null invariance crucial for the analysis of massive transaction data?
- Many transactions may contain neither milk nor coffee!

milk vs. coffee contingency table

	<i>milk</i>	$\neg\text{milk}$	Σ_{row}
<i>coffee</i>	<i>mc</i>	$\neg mc$	<i>c</i>
$\neg\text{coffee}$	$m\neg c$	$\neg m\neg c$	$\neg c$
Σ_{col}	<i>m</i>	$\neg m$	Σ

- Lift and χ^2 are not null-invariant: not good to evaluate data that contain too many or too few null transactions!
- Many measures are not null-invariant!

Data set	<i>mc</i>	$\neg mc$	$m\neg c$	$\neg m\neg c$	χ^2	<i>Lift</i>
<i>D</i> ₁	10,000	1,000	1,000	100,000	90557	9.26
<i>D</i> ₂	10,000	1,000	1,000	100	0	1
<i>D</i> ₃	100	1,000	1,000	100,000	670	8.44
<i>D</i> ₄	1,000	1,000	1,000	100,000	24740	25.75
<i>D</i> ₅	1,000	100	10,000	100,000	8173	9.18
<i>D</i> ₆	1,000	10	100,000	100,000	965	1.97

Comparison of Null-Invariant Measures

- ❑ Not all null-invariant measures are created equal
- ❑ Which one is better?
 - ❑ $D_4 - D_6$ differentiate the null-invariant measures
 - ❑ Kulc (Kulczynski 1927) holds firm and is in balance of both directional implications

All 5 are null-invariant

2-variable contingency table

	<i>milk</i>	$\neg milk$	Σ_{row}
<i>coffee</i>	<i>mc</i>	$\neg mc$	<i>c</i>
$\neg coffee$	<i>m</i> $\neg c$	$\neg m$ $\neg c$	$\neg c$
Σ_{col}	<i>m</i>	$\neg m$	Σ

Data set	<i>mc</i>	$\neg mc$	<i>m</i> $\neg c$	$\neg m$ $\neg c$	<i>AllConf</i>	Jaccard	Cosine	Kulc	MaxConf
D_1	10,000	1,000	1,000	100,000	0.91	0.83	0.91	0.91	0.91
D_2	10,000	1,000	1,000	100	0.91	0.83	0.91	0.91	0.91
D_3	100	1,000	1,000	100,000	0.09	0.05	0.09	0.09	0.09
D_4	1,000	1,000	1,000	100,000	0.5	0.33	0.5	0.5	0.5
D_5	1,000	100	10,000	100,000	0.09	0.09	0.29	0.5	0.91
D_6	1,000	10	100,000	100,000	0.01	0.01	0.10	0.5	0.99

Imbalance Ratio with Kulczynski Measure

- IR (Imbalance Ratio): measure the imbalance of two itemsets A and B in rule implications:
- $$IR(A, B) = \frac{|s(A) - s(B)|}{s(A) + s(B) - s(A \cup B)}$$
- Kulczynski and Imbalance Ratio (IR) together present a clear picture for all the three datasets D₄ through D₆
 - D₄ is neutral & balanced; D₅ is neutral but imbalanced
 - D₆ is neutral but very imbalanced

Data set	<i>mc</i>	$\neg mc$	<i>m</i> $\neg c$	$\neg m$ <i>c</i>	Jaccard	<i>Cosine</i>	<i>Kulc</i>	IR
<i>D</i> ₁	10,000	1,000	1,000	100,000	0.83	0.91	0.91	0
<i>D</i> ₂	10,000	1,000	1,000	100	0.83	0.91	0.91	0
<i>D</i> ₃	100	1,000	1,000	100,000	0.05	0.09	0.09	0
<i>D</i> ₄	1,000	1,000	1,000	100,000	0.33	0.5	0.5	0
<i>D</i> ₅	1,000	100	10,000	100,000	0.09	0.29	0.5	0.89
<i>D</i> ₆	1,000	10	100,000	100,000	0.01	0.10	0.5	0.99

Example: Analysis of DBLP Coauthor Relationships

- DBLP: Computer science research publication bibliographic database
 - > 3.8 million entries on authors, paper, venue, year, and other information

ID	Author A	Author B	$s(A \cup B)$	$s(A)$	$s(B)$	Jaccard	Cosine	Kulc
1	Hans-Peter Kriegel	Martin Ester	28	146	54	0.163 (2)	0.315 (7)	0.355 (9)
2	Michael Carey	Miron Livny	26	104	58	0.191 (1)	0.335 (4)	0.349 (10)
3	Hans-Peter Kriegel	Joerg Sander	24	146	36	0.152 (3)	0.331 (5)	0.416 (8)
4	Christos Faloutsos	Spiros Papadimitriou	20	162	26	0.119 (7)	0.308 (10)	0.446 (7)
5	Hans-Peter Kriegel	Martin Pfeifle	18	146	18	0.123 (6)	0.351 (2)	0.562 (2)
6	Hector Garcia-Molina	Wilbert Labio	16	144	18	0.110 (9)	0.314 (8)	0.500 (4)
7	Divyakant Agrawal	Wang Hsiung	16	120	16	0.133 (5)	0.365 (1)	0.567 (1)
8	Elke Rundensteiner	Murali Mani	16	104	20	0.148 (4)	0.351 (3)	0.477 (6)
9	Divyakant Agrawal	Oliver Po	12	120	12	0.100 (10)	0.316 (6)	0.550 (3)
10	Gerhard Weikum	Martin Theobald	12	106	14	0.111 (8)	0.312 (9)	0.485 (5)

Advisor-advisee relation: Kulc: high, Jaccard: low, cosine: middle

- Which pairs of authors are strongly related? Is A the advisor, or the advisee?
- Use Kulc to find Advisor-advisee, close collaborators

What Measures to Choose for Effective Pattern Evaluation?

- Null value cases are predominant in many large datasets
 - Neither milk nor coffee is in most of the baskets; neither Mike nor Jim is an author in most of the papers;
- *Null-invariance* is an important property
- Lift, χ^2 and cosine are good measures if null transactions are not predominant
 - Otherwise, *Kulczynski + Imbalance Ratio* should be used to judge the interestingness of a pattern

What Measures to Choose for Effective Pattern Evaluation?

- ❑ Exercise: Mining research collaborations from research bibliographic data
 - ❑ Find a group of frequent collaborators from research bibliographic data (e.g., DBLP)
 - ❑ Can you find the likely advisor-advisee relationship and during which years such a relationship happened?
 - ❑ Ref.: C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo, "Mining Advisor-Advisee Relationships from Research Publication Networks", KDD'10

Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary 

Summary

- ❑ Basic Concepts
 - ❑ What Is Pattern Discovery? Why Is It Important?
 - ❑ Basic Concepts: Frequent Patterns and Association Rules
 - ❑ Compressed Representation: Closed Patterns and Max-Patterns
- ❑ Efficient Pattern Mining Methods
 - ❑ The Downward Closure Property of Frequent Patterns
 - ❑ The Apriori Algorithm
 - ❑ Extensions or Improvements of Apriori
 - ❑ Mining Frequent Patterns by Exploring Vertical Data Format
 - ❑ FP-Growth: A Frequent Pattern-Growth Approach
 - ❑ Mining Closed Patterns
- ❑ Pattern Evaluation
 - ❑ Interestingness Measures in Pattern Mining
 - ❑ Interestingness Measures: Lift and χ^2
 - ❑ Null-Invariant Measures
 - ❑ Comparison of Interestingness Measures

Recommended Readings (Basic Concepts)

- R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases”, in Proc. of SIGMOD'93
- R. J. Bayardo, “Efficiently mining long patterns from databases”, in Proc. of SIGMOD'98
- N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, “Discovering frequent closed itemsets for association rules”, in Proc. of ICDT'99
- J. Han, H. Cheng, D. Xin, and X. Yan, “Frequent Pattern Mining: Current Status and Future Directions”, Data Mining and Knowledge Discovery, 15(1): 55-86, 2007

Recommended Readings (Efficient Pattern Mining Methods)

- R. Agrawal and R. Srikant, “Fast algorithms for mining association rules”, VLDB'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
- S. Sarawagi, S. Thomas, and R. Agrawal, “Integrating association rule mining with relational database systems: Alternatives and implications”, SIGMOD'98
- M. J. Zaki, S. Parthasarathy, M. Ogihsara, and W. Li, “Parallel algorithm for discovery of association rules”, Data Mining and Knowledge Discovery, 1997
- J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation”, SIGMOD'00
- M. J. Zaki and Hsiao, “CHARM: An Efficient Algorithm for Closed Itemset Mining”, SDM'02
- J. Wang, J. Han, and J. Pei, “CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets”, KDD'03
- C. C. Aggarwal, M.A., Bhuiyan, M. A. Hasan, “Frequent Pattern Mining Algorithms: A Survey”, in Aggarwal and Han (eds.): Frequent Pattern Mining, Springer, 2014

Recommended Readings (Pattern Evaluation)

- C. C. Aggarwal and P. S. Yu. A New Framework for Itemset Generation. PODS'98
- S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. SIGMOD'97
- M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM'94
- E. Omiecinski. Alternative Interest Measures for Mining Associations. TKDE'03
- P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. KDD'02
- T. Wu, Y. Chen and J. Han, Re-Examination of Interestingness Measures in Pattern Mining: A Unified Framework, Data Mining and Knowledge Discovery, 21(3):371-397, 2010



CS 412 Intro. to Data Mining

Chapter 6 : Advanced Pattern Mining

Arindam Banerjee, Computer Science, UIUC, Fall 2021



Chapter 6 : Advanced Frequent Pattern Mining

- Mining Diverse Patterns
- Constraint-Based Frequent Pattern Mining
- Sequential Pattern Mining
- Graph Pattern Mining
- Pattern Mining Application: Mining Software Copy-and-Paste Bugs
- Summary

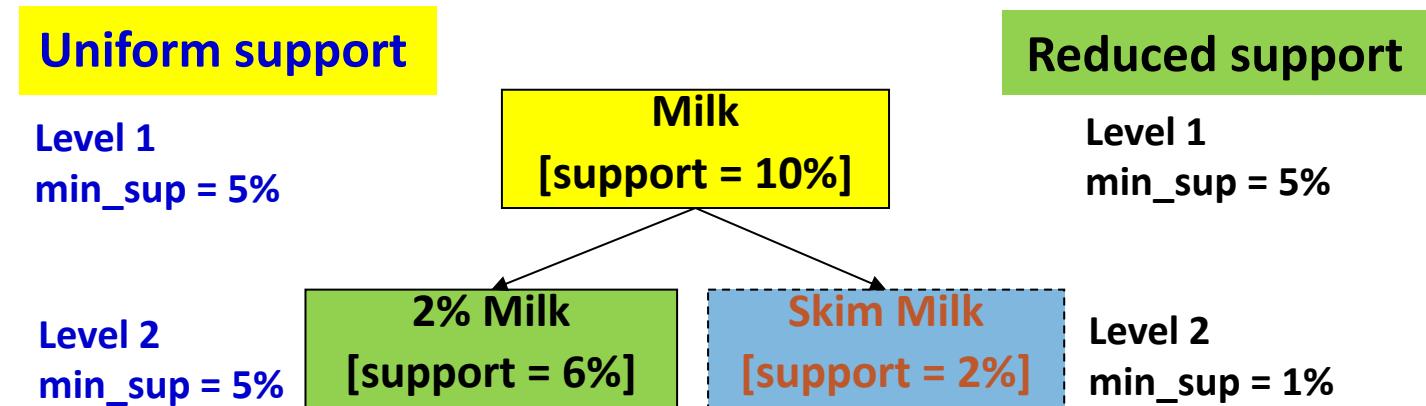


Mining Diverse Patterns

- Mining Multiple-Level Associations
- Mining Multi-Dimensional Associations
- Mining Quantitative Associations
- Mining Negative Correlations
- Mining Compressed and Redundancy-Aware Patterns

Mining Multiple-Level Frequent Patterns

- Min-support thresholds for hierarchy items
- **Uniform** min-support across multiple levels (reasonable?)
- **Level-reduced** min-support: Items at the lower level are expected to have lower support
- Efficient mining: *Shared* multi-level mining
- Use the lowest min-support to pass down the set of candidates



Redundancy Filtering at Mining Multi-Level Associations

- Redundancy filtering: redundant due to “ancestor” relationships
 - milk \Rightarrow wheat bread [support = 8%, confidence = 70%] (1)
 - 2% milk \Rightarrow wheat bread [support = 2%, confidence = 72%] (2)
 - Suppose the 2% milk sold is about $\frac{1}{4}$ of milk sold in gallons
 - (2) should be able to be “derived” from (1)

Redundancy Filtering at Mining Multi-Level Associations

- milk \Rightarrow wheat bread [support = 8%, confidence = 70%] (1)
- 2% milk \Rightarrow wheat bread [support = 2%, confidence = 72%] (2)
- A rule is *redundant* if its support is close to the “expected” value, according to its “ancestor” rule, and it has a similar confidence as its “ancestor”
- Rule (1) is an ancestor of rule (2), which one to prune?

Customized Min-Supports for Different Kinds of Items

- ❑ Same min-support threshold **for all** so far
- ❑ Diamonds, watches: valuable but **less frequent**
- ❑ One Method: Use **group-based** “individualized” min-support
 - ❑ E.g., {diamond, watch}: 0.05%; {bread, milk}: 5%; ...
 - ❑ How to mine such rules efficiently?
 - ❑ Existing scalable mining algorithms can be easily extended to cover such cases

Mining Multi-Dimensional Associations

- Single-dimensional rules (e.g., items are all in “product” dimension)
 - $\text{buys}(X, \text{"milk"}) \Rightarrow \text{buys}(X, \text{"bread"})$
- Multi-dimensional rules (i.e., items in ≥ 2 dimensions or predicates)
 - Inter-dimension association rules (*no repeated predicates*)
 - $\text{age}(X, \text{"18-25"}) \wedge \text{occupation}(X, \text{"student"}) \Rightarrow \text{buys}(X, \text{"coke"})$
 - Hybrid-dimension association rules (*repeated predicates*)
 - $\text{age}(X, \text{"18-25"}) \wedge \text{buys}(X, \text{"popcorn"}) \Rightarrow \text{buys}(X, \text{"coke"})$
- Attributes can be categorical or numerical
 - Categorical Attributes (e.g., *profession*, *product*: no ordering among values): Data cube for inter-dimension association
 - Quantitative Attributes: Numeric, implicit ordering among values—discretization, clustering, and gradient approaches

Mining Quantitative Associations

- ❑ Mining associations with numerical attributes
 - ❑ E.g.: Numerical attributes: **age** and **salary**
- ❑ Methods
 - ❑ **Static discretization** based on predefined concept hierarchies
 - ❑ Discretization on each dimension with hierarchy
 - ❑ age: {0-10, 10-20, ..., 90-100} → {young, mid-aged, old}
 - ❑ **Dynamic discretization** based on data distribution
 - ❑ **Clustering**: Distance-based association
 - ❑ First one-dimensional clustering, then association
 - ❑ **Deviation analysis**:
 - ❑ Gender = female ⇒ Wage: mean=\$7/hr (overall mean = \$9)

Mining Extraordinary Phenomena in Quantitative Association Mining

- ❑ Mining extraordinary (i.e., interesting) phenomena
 - ❑ E.g.: **Gender = female** \Rightarrow **Wage**: mean=\$7/hr (overall mean = \$9)
 - ❑ **LHS**: a subset of the population
 - ❑ **RHS**: an extraordinary behavior of this subset
- ❑ The rule is accepted only if a statistical test (e.g., Z-test) confirms the inference with high confidence

- ❑ Subrule: Highlights the extraordinary behavior of a subset of the population of the super rule
 - ❑ E.g.: **(Gender = female) ^ (South = yes)** \Rightarrow mean wage = \$6.3/hr
- ❑ Rule condition can be categorical or numerical (quantitative rules)
 - ❑ E.g.: **Education in [14-18] (yrs)** \Rightarrow mean wage = \$11.64/hr

Rare Patterns

- Rare patterns
 - Very low support but interesting (e.g., buying Rolex watches)
 - How to mine them? Setting individualized, group-based min-support thresholds for different groups of items

Negative Patterns

- ❑ Negative patterns
 - ❑ Negatively correlated: Unlikely to happen together
 - ❑ Ex.: Since it is unlikely that the same customer buys both a **Ford Expedition** (SUV) and a **Ford Fusion** (hybrid), buying a **Ford Expedition** and buying a **Ford Fusion** are likely negatively correlated patterns
 - ❑ How to define negative patterns?
- ❑ A support-based definition of negative correlated patterns
 - ❑ If itemsets A and B are **both frequent** but rarely occur together, i.e., $\text{sup}(A \cup B) \ll \text{sup}(A) \times \text{sup}(B)$

Does this remind you the definition of *lift*?

Defining Negative Correlated Patterns

Is this a good definition for large transaction datasets?

- Ex.: Suppose a store sold two types of vehicles A and B 100 times each, but only one transaction contained both A and B
- When there are in total 200 transactions (199 contain A or B), we have
 - $s(A \cup B) = 0.005, s(A) \times s(B) = 0.25, s(A \cup B) << s(A) \times s(B)$
- But when there are 10^5 transactions, we have
 - $s(A \cup B) = 1/10^5, s(A) \times s(B) = 1/10^3 \times 1/10^3, s(A \cup B) > s(A) \times s(B)$
- What is the problem?—Null transactions: The support-based definition is not null-invariant!

Defining Negative Correlation: Need Null-Invariance in Definition

- A Kulczynski measure-based definition
 - If itemsets A and B are frequent but $(s(A \cup B)/s(A) + s(A \cup B)/s(B))/2 < \epsilon$,
then A and B are negatively correlated
- For the same car buying problem:
 - Does not matter if there are in total 200 or 10^5 transactions
 - If $\epsilon = 0.01$, we have
$$(s(A \cup B)/s(A) + s(A \cup B)/s(B))/2 = (0.01 + 0.01)/2 < \epsilon$$

negative pattern threshold

Mining Compressed Patterns

Pat-ID	Item-Sets	Support
P1	{38,16,18,12}	205,227
P2	{38,16,18,12,17}	205,211
P3	{39,38,16,18,12,17}	101,758
P4	{39,16,18,12,17}	161,563
P5	{39,16,18,12}	161,576

- Closed patterns
 - P1, P2, P3, P4, P5
 - Emphasizes too much on support
- Max-patterns
 - P3: information loss
- Desired output (a good balance):
 - P2, P3, P4

- Why mining compressed patterns? Too many scattered patterns but not so meaningful

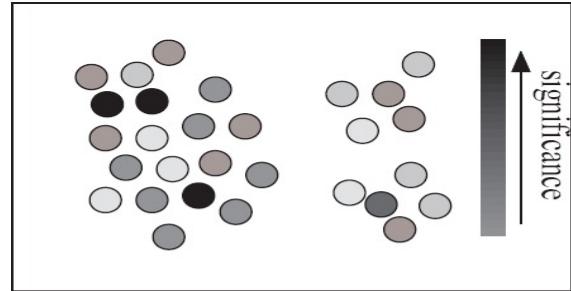
- Pattern distance measure

$$Dist(P_1, P_2) = 1 - \frac{|T(P_1) \cap T(P_2)|}{|T(P_1) \cup T(P_2)|}$$

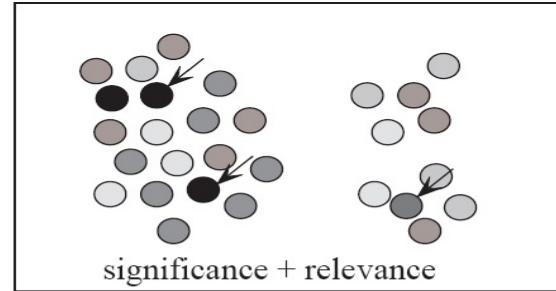
- δ -clustering: For each pattern P, find all patterns which can be expressed by P and whose distance to P is within δ (δ -cover)
- All patterns in the cluster can be represented by P

Redundancy-Aware Top-k Patterns

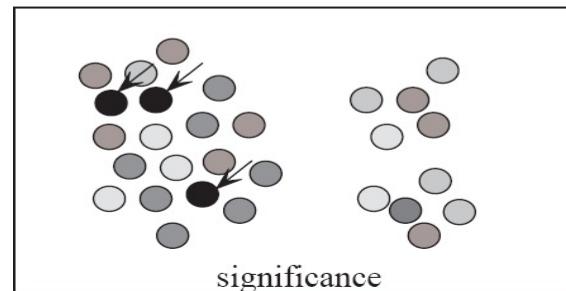
- Desired patterns: high significance & low redundancy



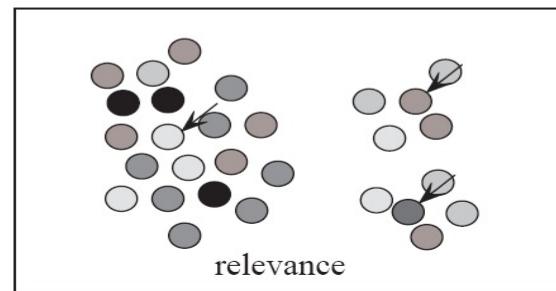
(a) a set of patterns



(b) redundancy-aware
top- k



(c) traditional top- k



(d) summarization

- Method: Use MMS (Maximal Marginal Significance) for measuring the combined significance of a pattern set
- Xin et al., Extracting Redundancy-Aware Top-K Patterns, KDD'06

Chapter 6 : Advanced Frequent Pattern Mining

- Mining Diverse Patterns
- Constraint-Based Frequent Pattern Mining
- Sequential Pattern Mining
- Graph Pattern Mining
- Pattern Mining Application: Mining Software Copy-and-Paste Bugs
- Summary



Constraint-Based Pattern Mining

- Why Constraint-Based Mining?
- Different Kinds of Constraints: Different Pruning Strategies
- Constrained Mining with Pattern Anti-Monotonicity
- Constrained Mining with Pattern Monotonicity
- Constrained Mining with Convertible Constraints
- Constrained Mining with Data Anti-Monotonicity
- Constrained Mining with Succinct Constraints
- Handling Multiple Constraints

Why Constraint-Based Mining?

- Pattern mining in practice: Often a user-guided, **interactive** process
 - User directs what to be mined using a **data mining query language** (or a graphical user interface), **specifying various kinds of constraints**
- What is constraint-based mining?
 - Mine together with user-provided constraints
- Why constraint-based mining?
 - User flexibility: User provides **constraints** on what to be mined
 - Optimization: System explores such constraints for mining efficiency
 - E.g., Push constraints deeply into the mining process

Various Kinds of User-Specified Constraints in Data Mining

- **Knowledge type constraint**—Specifying what kinds of knowledge to mine
 - E.g.: Classification, association, clustering, outlier finding, ...
- **Data constraint**—using SQL-like queries
 - E.g.: Find products sold together in **NY** stores **this year**
- **Dimension/level constraint**—similar to projection in relational database
 - E.g.: In relevance to **region**, **price**, **brand**, **customer category**
- **Interestingness constraint**—various kinds of thresholds
 - E.g.: Strong rules: $\text{min_sup} \geq 0.02$, $\text{min_conf} \geq 0.6$, $\text{min_correlation} \geq 0.7$
- **Rule (or pattern) constraint**  **The focus of this study**
 - E.g.: Small sales (price < \$10) triggers big sales (sum > \$200)

Pattern Space Pruning with Pattern Anti-Monotonicity

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-10
g	80	20
h	10	5

Note: item.price > 0
Profit can be negative

- A constraint c is **anti-monotone**
 - If an itemset S violates constraint c , so does any of its superset
 - That is, mining on itemset S can be terminated
- E.g. 1: $c_1: \text{sum}(S.\text{price}) \leq 160$ is **anti-monotone**
 - Sum grows as you add more items
 - Itemset abc violates c_1
- E.g. 2: $c_2: \text{range}(S.\text{profit}) \leq 15$ is **anti-monotone**
 - Itemset ab violates c_2 ($\text{range}(ab) = 40$)
 - So does every superset of ab

Pattern Space Pruning with Pattern Anti-Monotonicity

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

min_sup = 2

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-10
g	80	20
h	10	5

- E.g. 3. $c_3: \text{sum}(S.\text{Price}) \geq 160$ is **not anti-monotone**
 - ab violates the constraint, but super-sets of ab need not violate
 - abc, abd , etc., does not violate the constraint
 - Cannot be used for pruning super-sets

- E.g. 4. Is $c_4: \text{support}(S) \geq \sigma$ anti-monotone?
 - Yes! Apriori pruning is essentially pruning with an anti-monotonic constraint!

Pattern Monotonicity and Its Roles

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

min_sup = 2		
Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-10
g	80	20
h	10	5

- A constraint c is *monotone*: If an itemset S **satisfies** the constraint c , so does any of its superset
 - That is, we do not need to check c in subsequent mining
 - Not as beneficial as anti-monotone
- E.g. 1: $c_1: \text{sum}(S.\text{Price}) \geq 160$ is **monotone**
- E.g. 2: $c_2: \text{min}(S.\text{Price}) \leq 50$ is **monotone**
- E.g. 3: $c_3: \text{range}(S.\text{profit}) \geq 15$ is **monotone**
 - Itemset ab satisfies c_3
 - So does every superset of ab

Apriori for Pattern Anti-Monotone Constraint

Item	Price
1	1
2	2
3	3
4	4
5	5

Database D

TID	Items
10	1 3 4
20	2 3 5
30	1 2 3 5
40	2 5

Scan D

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

F_1

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

Can be
chopped
early

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}

Scan D

F_2

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



Scan D

itemset
{2 3 5}

F_3

itemset	sup.
{2 3 5}	2

Min_sup=2

Constraint:

$\text{Sum}\{\text{S.price}\} < 5$

Convertible Constraints: Ordering Data in Transactions

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

min_sup = 2

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-5
g	80	30
h	10	5

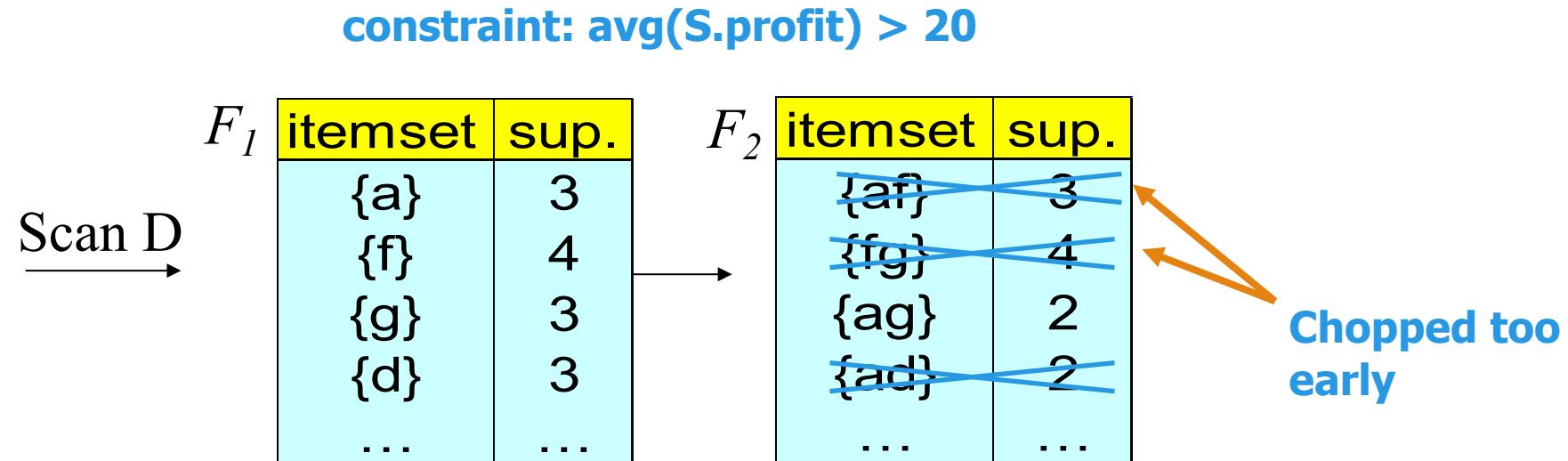
- Convert tough constraints into (anti-)monotone by proper ordering of items in transactions
- Examine $c_1: \text{avg}(S.\text{profit}) > 35$
 - Order items in (profit) value-descending order
 - $\langle a, g, h, b, f, d, c, e \rangle$
 - An itemset ag violates c_1 ($\text{avg}(ag) = 35$)
 - So does ag^* (i.e., ag -projected DB)
 - C_1 : anti-monotone if patterns grow in the right order!

Can item-reordering work for Apriori?

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

min_sup = 2

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-5
g	80	30
h	10	5



- $\text{avg}(fg) = 12.5 < 20$, $\text{avg}(af) = 17.5 < 20$, $\text{avg}(ag) = 35 > 20$
- But $\text{avg}(agf) = 21.7 > 20$
- Apriori will not generate “agf” as a candidate

Data Space Pruning with Data Anti-Monotonicity

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

min_sup = 2

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-10
g	80	20
h	10	5

- A constraint c is **data anti-monotone**: In the mining process, if a data entry t cannot contribute to a pattern p satisfying c , t cannot contribute to p 's superset either
- Data space pruning: Data entry t can be pruned
- E.g. 1: $c_1: \text{sum}(S.\text{Profit}) \geq v$ is **data anti-monotone**
 - Let constraint c_1 be: $\text{sum}(S.\text{Profit}) \geq 25$
 - $T_{30}: \{b, c, d, f, h\}$ can be removed since none of their combinations can make an S whose sum of the profit is ≥ 25

Data Space Pruning with Data Anti-Monotonicity

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

min_sup = 2

Item	Price	Profit
a	100	40
b	40	0
c	150	-20
d	35	-15
e	55	-30
f	45	-10
g	80	20
h	10	5

- ❑ A constraint c is **data anti-monotone**: In the mining process, if a data entry t cannot contribute to a pattern p satisfying c , t cannot contribute to p 's superset either
- ❑ Data space pruning: Data entry t can be pruned
- ❑ E.g. 2: $c_2: \min(S.\text{Price}) \leq v$ is **data anti-monotone**
 - ❑ Consider $v = 5$ but every item in a transaction, say T_{50} , has a price higher than 10
- ❑ E.g. 3: $c_3: \text{range}(S.\text{Profit}) > 25$ is **data anti-monotone**
 - ❑ Transaction $\{c,d,e,f\}$ has a range of 20

Data Space Pruning Should Be Explored Recursively

TID	Transaction	Item	Profit
10	a, b, c, d, f, h	a	40
20	b, c, d, f, g, h	b	0
30	b, c, d, f, g	c	-20
40	a, c, e, f, g	d	-15
min_sup = 2		e	-30
		f	-10
		g	20
		h	5

- ❑ Example. $c_3: \text{range}(S.\text{Profit}) > 25$
 - ❑ We check b's projected database
 - ❑ T_{10} satisfies c_3
 - ❑ But item “a” is infrequent ($\text{sup} = 1$)

b's-proj. DB



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

- ❑ After removing “a (40)” from T_{10}
 - ❑ T_{10} cannot satisfy c_3 any more
 - ❑ Since “b (0)” and “c (-20), d (-15), f (-10), h (5)”, we can remove “h”
 - ❑ By removing T_{10} , we can also prune “h”

b's-proj. DB

TID	Transaction
10	c, d, f, h
20	c, d, f, g, h
30	c, d, f, g

Data Space Pruning Should Be Explored Recursively

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

Recursive
Data
Pruning

single branch: cdfg: 2

Constraint:
 $\text{range}\{\text{S}.profit\} > 25$

Only a single branch “cdfg: 2”
to be mined in b’s projected DB

- ❑ Note: c_3 prunes T_{10} effectively only after “a” is pruned (by min-sup) in b’s projected DB

Succinctness: Pruning Both Data and Pattern Spaces

- Succinctness: If the constraint c can be enforced by directly manipulating the data
- E.g. 1: To find those patterns containing item i
 - Mine only i -projected DB (data space pruning)
- E.g. 2: To find those patterns without item i
 - Remove i from DB and then mine (pattern space pruning)
- E.g. 3: $c_3: \min(S.\text{Price}) \leq v$ is succinct
 - Start with only items whose price $\leq v$ and remove transactions with high-price items only (pattern + data space pruning)
- E.g. 4: $c_4: \sum(S.\text{Price}) \geq v$ is not succinct
 - It cannot be determined beforehand since sum of the price of itemset S keeps increasing

Constrained FP-Growth: Push a Succinct Constraint Deep

Item	Price
1	1
2	2
3	3
4	4
5	5

TID	Items
10	1 3 4
20	2 3 5
30	1 2 3 5
40	2 5

Remove infrequent length 1

TID	Items
10	1 3
20	2 3 5
30	1 2 3 5
40	2 5

Min_sup=2

Constraint:

$\min\{S.\text{price}\} \leq 2$

No Need to project on 3 or 5

1-Projected DB

TID	Items
10	3
30	2 3 5

2-Projected DB

TID	Items
20	3 5
30	1 3 5
40	5

Commonly Used Pattern Pruning Constraints

Table 6.3: Characterization of Commonly Used Pattern Pruning Constraints

<i>Constraint</i>	<i>Antimonotonic</i>	<i>Monotonic</i>	<i>Succinct</i>
$v \in S$	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	no
$\text{count}(S) \geq v$	no	yes	no
$\text{sum}(S) \leq v \ (\forall a \in S, a \geq 0)$	yes	no	no
$\text{sum}(S) \geq v \ (\forall a \in S, a \geq 0)$	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
$\text{all_confidence}(S) \geq \xi$	yes	no	no
$\text{all_confidence}(S) \leq \xi$	no	yes	no

Different Kinds of Constraints Lead to Different Pruning Strategies

- In summary, constraints can be categorized as **pattern space pruning** constraints vs. **data space pruning** constraints

Pattern space pruning constraints	Data space pruning constraints
<ul style="list-style-type: none">Anti-monotonic: If constraint c is violated, its further mining can be terminatedMonotonic: If c is satisfied, no need to check c againConvertible: c can be converted to monotonic or anti-monotonic if items can be properly ordered in processingSuccinct: If the constraint c can be enforced by directly manipulating the data	<ul style="list-style-type: none">Data succinct: Data space can be pruned at the initial pattern mining processData anti-monotonic: If a transaction t does not satisfy c, then t can be pruned to reduce data processing effort

How to Handle Multiple Constraints?

- It is beneficial to use multiple constraints in pattern mining
- But different constraints may require potentially conflicting item-ordering
 - If there exists conflict ordering between c_1 and c_2
 - Try to sort data and enforce *one constraint* first (which one?)
 - Then enforce the other constraint when mining the projected databases
- E.g. c_1 : $\text{avg}(S.\text{profit}) > 20$, and c_2 : $\text{avg}(S.\text{price}) < 50$
 - Assume c_1 has more pruning power
 - Sort in profit descending order and use c_1 first
 - For each project DB, sort transactions in price ascending order and use c_2 during mining

Chapter 6 : Advanced Frequent Pattern Mining

- Mining Diverse Patterns
- Constraint-Based Frequent Pattern Mining
- Sequential Pattern Mining 
- Graph Pattern Mining
- Pattern Mining Application: Mining Software Copy-and-Paste Bugs
- Summary

Sequential Pattern Mining

- Sequential Pattern and Sequential Pattern Mining
- GSP: Apriori-Based Sequential Pattern Mining
- SPADE: Sequential Pattern Mining in Vertical Data Format
- PrefixSpan: Sequential Pattern Mining by Pattern-Growth
- CloSpan: Mining Closed Sequential Patterns
- Constraint-Based Sequential-Pattern Mining

Sequential Pattern Mining

- What kind of patterns are sequential?
- Sequential – The order really matters. You can not swap two items in a sequence and have the same sequence.
- Examples: English language is sequential : Subject -> Verb -> Object, amino acid sequences, etc.

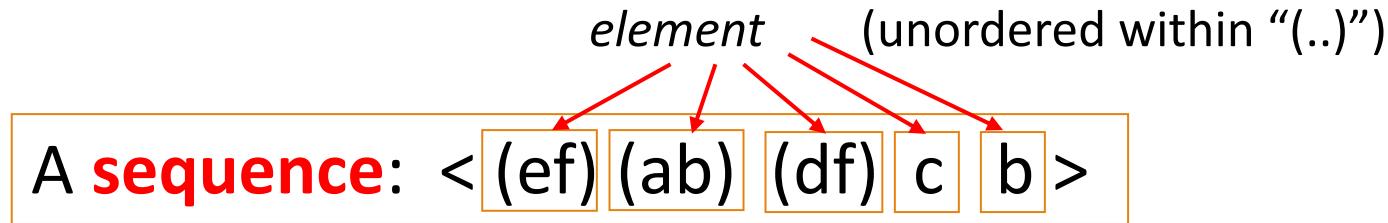
- Other points:
 - For Sequential Pattern Mining, the time at which the items occur is **not** considered.
 - Time Series Analysis does take into account the time at which an item occurred.

Sequential Pattern Examples

- Application of Sequential pattern Mining
 - **Customer shopping** → Purchase a laptop first, then a digital camera, and then a smartphone.
 - **Medical treatments** → Go to the doctor, get drugs, doctor monitors progress, doctor reacts accordingly -> more/less drugs
 - **Natural disasters** -> Before the disaster, during the disaster, after the disaster.
 - **Scientific Experiments** → Step 1, Step 2, Step 3.
 - **Stocks Markets** → Stocks go up and down together.
 - **Biological sequences, DNA /Protein**→ If you change the order of proteins, it is a different gene.

Sequential Pattern and Sequential Pattern Mining

- ❑ **Sequential pattern mining:** Given a set of sequences, find the **complete set of frequent subsequences** (i.e., satisfying the min_sup threshold)



- ❑ An element may contain a set of items (also called events)
 - * Items within an element are **unordered** and we list them alphabetically

A **sequence database**

SID	Sequence
10	<a(<u>abc</u>)(a <u>c</u>)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(<u>ab</u>)(df) <u>c</u> b>
40	<eg(af)cbc>

Sequential Pattern and Sequential Pattern Mining

- **Sequential pattern mining:** Given a set of sequences, find the **complete set of frequent subsequences** (i.e., satisfying the `min_sup` threshold)



- Given support threshold $min_sup = 2$, $\langle(ab)c\rangle$ is a sequential pattern

A **sequence database**

SID	Sequence
10	$\langle a(\underline{ab}c)(a\underline{c})d(cf) \rangle$
20	$\langle(ad)c(bc)(ae) \rangle$
30	$\langle(ef)(\underline{ab})(df)\underline{cb} \rangle$
40	$\langle eg(af)cbc \rangle$

Sequential Pattern Mining Algorithms

- Algorithm requirement: Efficient, scalable, finding complete set, incorporating various kinds of user-specific constraints
- The Apriori property still holds: If a subsequence s_1 is infrequent, none of s_1 's super-sequences can be frequent
- Representative algorithms
 - GSP (Generalized Sequential Patterns): Srikant & Agrawal @ EDBT'96)
 - Vertical format-based mining: SPADE (Zaki@Machine Learning'00)
 - Pattern-growth methods: PrefixSpan (Pei, et al. @TKDE'04)
- Mining closed sequential patterns: CloSpan (Yan, et al. @SDM'03)
- Constraint-based sequential pattern mining (to be covered in the constraint mining section)

GSP: Apriori-Based Sequential Pattern Mining

- Initial candidates: All 8-singleton sequences
 - <a>, , <c>, <d>, <e>, <f>, <g>, <h>
- Scan DB once, count support for each candidate

SID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

$min_sup = 2$



Cand.	sup
<a>	3
	5
<c>	4
<d>	3
<e>	3
<f>	2
<g>	1
<h>	1

GSP: Apriori-Based Sequential Pattern Mining

- Example: Generate length-2 candidate sequences

singleton * singleton – Total: $(6 * 6)$

min_sup = 2

Cand.	sup
<a>	3
	5
<c>	4
<d>	3
<e>	3
<f>	2
<g>	1
<h>	1



	<a>		<c>	<d>	<e>	<f>
<a>	<aa>	<ab>	<ac>	<ad>	<ae>	<af>
	<ba>	<bb>	<bc>	<bd>	<be>	<bf>
<c>	<ca>	<cb>	<cc>	<cd>	<ce>	<cf>
<d>	<da>	<db>	<dc>	<dd>	<de>	<df>
<e>	<ea>	<eb>	<ec>	<ed>	<ee>	<ef>
<f>	<fa>	<fb>	<fc>	<fd>	<fe>	<ff>

Sets (unordered) – Total: $(6 * 5) / 2$

	<a>		<c>	<d>	<e>	<f>
<a>		<(ab)>	<(ac)>	<(ad)>	<(ae)>	<(af)>
			<(bc)>	<(bd)>	<(be)>	<(bf)>
<c>				<(cd)>	<(ce)>	<(cf)>
<d>					<(de)>	<(df)>
<e>						<(ef)>
<f>						

Apriori Pruning

- w/o pruning
(includes g and h):

$$8 * 8 + 8 * 7 / 2 = 92$$

length-2 candidates

- w/ pruning:
 $6 * 6 + 6 * 5 / 2 = 51$
length-2 candidates

GSP Mining and Pruning

5th scan: 1 cand. 1 length-5 seq. pat.

<(bd)cba>

length

5

4th scan: 8 cand. 7 length-4 seq. pat.

<abba> <(bd)bc> ...

4

3rd scan: 46 cand. 20 length-3 seq. pat. 20
cand. not in DB at all

<abb> <aab> <aba> **<baa>** <bab> ...

3

2nd scan: **51** cand. 19 length-2 seq. pat.
10 cand. not in DB at all

<aa> <ab> ... <af> <ba> <bb> ... <ff> **<(ab)>** ... **<(ef)>**

2

1st scan: 8 cand. 6 length-1 seq. pat.

<a> <c> <d> <e> <f> **<g>** **<h>**

1

$$6*6 + 6*5/2 = 51$$

- Remove
 - Candidates not in DB
 - Candidates < min_sup

min_sup = 2

SID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

GSP Mining and Pruning

- Repeat, starting at $k=1$ until $k \leq \text{length}$
 - Scan DB to find “ $\text{length}-k$ ” frequent sequences
 - Generate “ $\text{length}-(k+1)$ ” candidate sequences from “ $\text{length}-k$ ” frequent sequences using **Apriori**
 - set $k = k+1$
- Until no frequent sequence or no candidate can be found

GSP (Generalized Sequential Patterns): Srikant & Agrawal @ EDBT'96)
-NOTE: Same team which developed Apriori

Sequential Pattern Mining in Vertical Data Format: The SPADE Algorithm

- A sequence database is mapped to: <SID, EID>
- Grow the subsequences (patterns) one item at a time by Apriori candidate generation

SID	Sequence
1	<a(<u>bc</u>)(ac)d(cf)>
2	<(ad)c(bc)(ae)>
3	<(ef)(ab)(df) <u>cb</u> >
4	<eg(af)cbc>
<i>min_sup = 2</i>	

Ref: SPADE (Sequential
Pattern Discovery
using Equivalent Class)
[M. Zaki 2001]

SID	EID	Items
1	1	a
1	2	abc
1	3	ac
1	4	d
1	5	cf
2	1	ad
2	2	c
2	3	bc
2	4	ae
3	1	ef
3	2	ab
3	3	df
3	4	c
3	5	b
4	1	e
4	2	g
4	3	af
4	4	c
4	5	b
4	6	c

a		b		...	
SID	EID	SID	EID	...	
1	1	1	2		
1	2	2	3		
1	3	3	2		
2	1	3	5		
2	4	4	5		
3	2				
4	3				

ab			...		
SID	EID (a)	EID(b)	SID	EID (b)	EID(a)
1	1	2	1	2	3
2	1	3	2	3	4
3	2	5			
4	3	5			

aba				...	
SID	EID (a)	EID(b)	EID(a)	...	
1	1	2	3		
2	1	3	4		

EID (b) < EID (a):
Corresponds to:
<a(bc)(ac)d(cf)>

ba ...
SID EID (b) EID(a)

1 2 3
2 3 4

3 5
4 5

aba ...
SID EID (a)

1 2
2 3

3 4

PrefixSpan: A Pattern-Growth Approach

SID	Sequence
10	<a(<u>abc</u>)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df) <u>cb</u> >
40	<eg(af)cbc>

min_sup = 2	
Prefix	<u>Suffix (Projection)</u>
<a>	<(abc)(ac)d(cf)>
<aa>	<(_bc)(ac)d(cf)>
<ab>	<(_c)(ac)d(cf)>

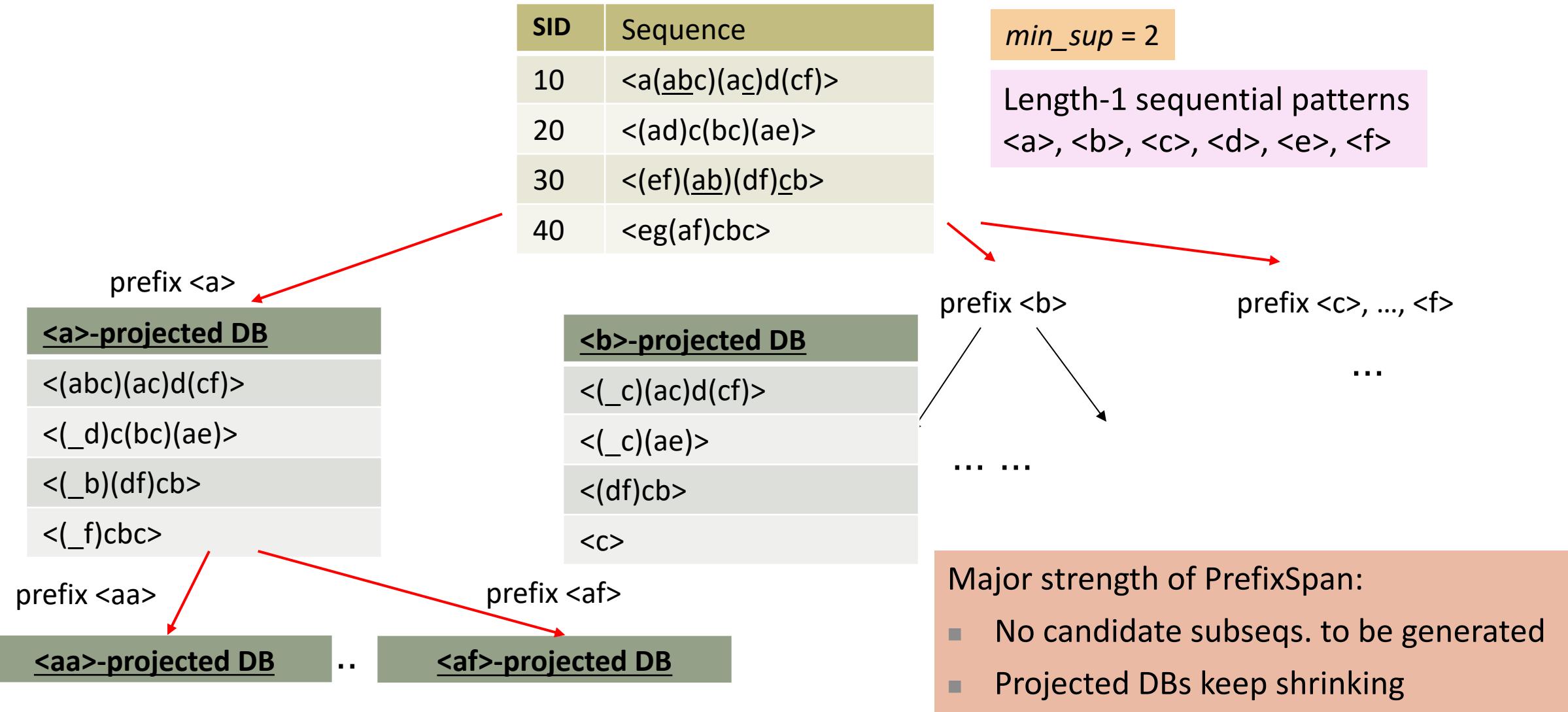
- ❑ PrefixSpan Mining: Prefix Projections
 - ❑ Step 1: Find length-1 sequential patterns
 - ❑ <a>, , <c>, <d>, <e>, <f>
 - ❑ Step 2: Divide search space and mine each projected DB
 - ❑ <a>-projected DB,
 - ❑ -projected DB,
 - ❑ ...
 - ❑ <f>-projected DB, ...

- ❑ Prefix and suffix
 - ❑ Given <a(abc)(ac)d(cf)>
 - ❑ Prefixes: <a>, <aa>, <a(ab)>, <a(abc)>, ...
- ❑ Suffix: Prefixes-based projection

"_" is placeholder for prefix

PrefixSpan (Prefix-projected Sequential pattern mining)
Pei, et al. @TKDE'04

PrefixSpan: Mining Prefix-Projected DBs



PrefixSpan: Key Steps

1. Let $\{\langle x_1 \rangle, \langle x_2 \rangle, \dots, \langle x_n \rangle\}$ be the complete set of length-1 sequential patterns in a sequence database, S . The complete set of sequential patterns in S can be partitioned into n disjoint subsets. The i^{th} subset ($1 \leq i \leq n$) is the set of sequential patterns with prefix $\langle x_i \rangle$.
2. Let α be a length- l sequential pattern and $\{\beta_1, \beta_2, \dots, \beta_m\}$ be the set of all length- $(l + 1)$ sequential patterns with prefix α . The complete set of sequential patterns with prefix α , except for α itself, can be partitioned into m disjoint subsets. The j^{th} subset ($1 \leq j \leq m$) is the set of sequential patterns prefixed with β_j .

PrefixSpan: Example

Sequence_ID	Sequence
1	$\langle a(abc)(ac)d(cf) \rangle$
2	$\langle (ad)c(bc)(ae) \rangle$
3	$\langle (ef)(ab)(df)cb \rangle$
4	$\langle eg(af)cbc \rangle$

1. Find length-1 patterns

$\langle a \rangle$: 4, $\langle b \rangle$: 4, $\langle c \rangle$: 4, $\langle d \rangle$: 3,
 $\langle e \rangle$: 3, $\langle f \rangle$: 3

2. Partition the search space,
based on prefix

3. Find subsets of patterns:

$\langle a \rangle$ -projected db, frequent
 $\langle a \rangle$: 2, $\langle b \rangle$: 4, $\langle _b \rangle$: 2, $\langle c \rangle$: 4,
 $\langle d \rangle$: 2, $\langle f \rangle$: 2

4. All patterns found recursively

prefix	projected database
$\langle a \rangle$	$\langle (abc)(ac)d(cf) \rangle,$ $\langle (_d)c(bc)(ae) \rangle,$ $\langle (_b)(df)cb \rangle,$ $\langle (_f)cbc \rangle$
$\langle b \rangle$	$\langle (_c)(ac)d(cf) \rangle,$ $\langle (_c)(ae) \rangle, \langle (df)cb \rangle,$ $\langle c \rangle$
$\langle c \rangle$	$\langle (ac)d(cf) \rangle,$ $\langle (bc)(ae) \rangle, \langle b \rangle,$ $\langle bc \rangle$
$\langle d \rangle$	$\langle (cf) \rangle, \langle c(bc)(ae) \rangle,$ $\langle (_f)cb \rangle$
$\langle e \rangle$	$\langle (_f)(ab)(df)cb \rangle,$ $\langle (af)cbc \rangle$
$\langle f \rangle$	$\langle (ab)(df)cb \rangle, \langle cbc \rangle$

PrefixSpan: Example (Contd.)

Sequence_ID	Sequence
1	$\langle a(abc)(ac)d(cf) \rangle$
2	$\langle(ad)c(bc)(ae) \rangle$
3	$\langle(ef)(ab)(df)cb \rangle$
4	$\langle eg(af)cbc \rangle$

1. Find length-1 patterns

$\langle a \rangle: 4, \langle b \rangle: 4, \langle c \rangle: 4, \langle d \rangle: 3,$
 $\langle e \rangle: 3, \langle f \rangle: 3$

2. Partition the search space,
based on prefix

3. Find subsets of patterns:

$\langle a \rangle$ -projected db, frequent
 $\langle a \rangle: 2, \langle b \rangle: 4, \langle c \rangle: 4,$
 $\langle d \rangle: 2, \langle f \rangle: 2$

4. All patterns found recursively

Projected databases:

$\langle aa \rangle : \langle(_bc)(ac)d(cf) \rangle, \langle(_e) \rangle$

not frequent, stop

$\langle ab \rangle : \langle(_c)(ac)d(cf) \rangle, \langle(_c)(ae) \rangle, \langle c \rangle$

frequent: $\langle(_c) \rangle, \langle a \rangle, \langle c \rangle$

$\langle a(bc) \rangle : \langle(ac)d(cf) \rangle, \langle(ae) \rangle$

frequent: $\langle a \rangle$

sequential patterns: $\langle a(bc) \rangle, \langle aba \rangle, \langle abc \rangle,$

$\langle a(bc)a \rangle$

$\langle(ab) \rangle : \langle(_c)(ac)d(cf) \rangle, \langle(df)cb \rangle$

frequent: $\langle c \rangle, \langle d \rangle, \langle f \rangle,$

$\langle(ab)d \rangle : \langle(cf) \rangle, \langle(_f)cb \rangle$

sequential patterns: $\langle(ab)c \rangle, \langle(ab)d \rangle, \langle(ab)f \rangle, \langle(ab)dc \rangle$

Similarly, for $\langle ac \rangle, \langle ad \rangle, \langle af \rangle$

PrefixSpan: Example (Contd.)

Sequence_ID	Sequence
1	$\langle a(abc)(ac)d(cf) \rangle$
2	$\langle (ad)c(bc)(ae) \rangle$
3	$\langle (ef)(ab)(df)cb \rangle$
4	$\langle eg(af)cbe \rangle$

1. Find length-1 patterns

$\langle a \rangle$: 4, $\langle b \rangle$: 4, $\langle c \rangle$: 4, $\langle d \rangle$: 3,
 $\langle e \rangle$: 3, $\langle f \rangle$: 3

2. Partition the search space,
 based on prefix

3. Find subsets of patterns:

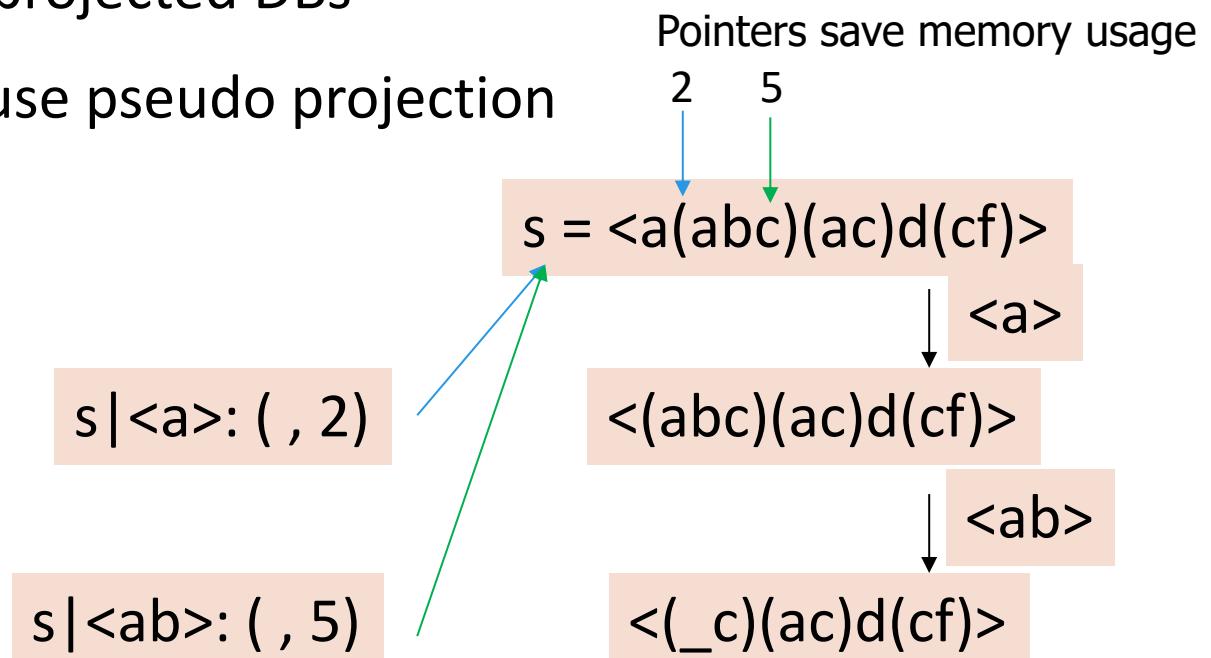
$\langle a \rangle$ -projected db, frequent
 $\langle a \rangle$: 2, $\langle b \rangle$: 4, $\langle _b \rangle$: 2, $\langle c \rangle$: 4,
 $\langle d \rangle$: 2, $\langle f \rangle$: 2

4. All patterns found recursively

prefix	projected database	sequential patterns
$\langle a \rangle$	$\langle (abc)(ac)d(cf) \rangle,$ $\langle (-d)c(bc)(ae) \rangle,$ $\langle (-b)(df)cb \rangle,$ $\langle (-f)cbe \rangle$	$\langle a \rangle, \langle aa \rangle, \langle ab \rangle, \langle a(bc) \rangle, \langle a(bc)a \rangle,$ $\langle aba \rangle, \langle abc \rangle, \langle (ab) \rangle, \langle (ab)c \rangle, \langle (ab)d \rangle,$ $\langle (ab)f \rangle, \langle (ab)dc \rangle, \langle ac \rangle, \langle aca \rangle, \langle acb \rangle,$ $\langle acc \rangle, \langle ad \rangle, \langle adc \rangle, \langle af \rangle$
$\langle b \rangle$	$\langle (-c)(ac)d(cf) \rangle,$ $\langle (-c)(ae) \rangle, \langle (df)cb \rangle,$ $\langle c \rangle$	$\langle b \rangle, \langle ba \rangle, \langle bc \rangle, \langle (bc) \rangle, \langle (bc)a \rangle, \langle bd \rangle,$ $\langle bdc \rangle, \langle bf \rangle$
$\langle c \rangle$	$\langle (ac)d(cf) \rangle,$ $\langle (bc)(ae) \rangle, \langle b \rangle,$ $\langle bc \rangle$	$\langle c \rangle, \langle ca \rangle, \langle cb \rangle, \langle cc \rangle$
$\langle d \rangle$	$\langle (cf) \rangle, \langle c(bc)(ae) \rangle,$ $\langle (-f)cb \rangle$	$\langle d \rangle, \langle db \rangle, \langle dc \rangle, \langle dcb \rangle$
$\langle e \rangle$	$\langle (-f)(ab)(df)cb \rangle,$ $\langle (af)cbe \rangle$	$\langle e \rangle, \langle ea \rangle, \langle eab \rangle, \langle eac \rangle, \langle eacb \rangle, \langle eb \rangle,$ $\langle ebc \rangle, \langle ec \rangle, \langle ecb \rangle, \langle ef \rangle, \langle effb \rangle, \langle efc \rangle,$ $\langle f \rangle, \langle fb \rangle, \langle fbc \rangle, \langle fc \rangle, \langle fcb \rangle$
$\langle f \rangle$	$\langle (ab)(df)cb \rangle, \langle cbe \rangle$	$\langle f \rangle, \langle fb \rangle, \langle fbc \rangle, \langle fc \rangle, \langle fcb \rangle$

Implementation Consideration: Pseudo-Projection vs. Physical Projection

- Major cost of PrefixSpan: Constructing projected DBs
 - Suffixes largely repeating in recursive projected DBs
- When DB can be held in main memory, use pseudo projection
 - No physically copying suffixes
 - Pointer to the sequence
 - Offset of the suffix
- But if it does not fit in memory
 - Physical projection
- Suggested approach:
 - Integration of physical and pseudo-projection
 - Swapping to pseudo-projection when the data fits in memory

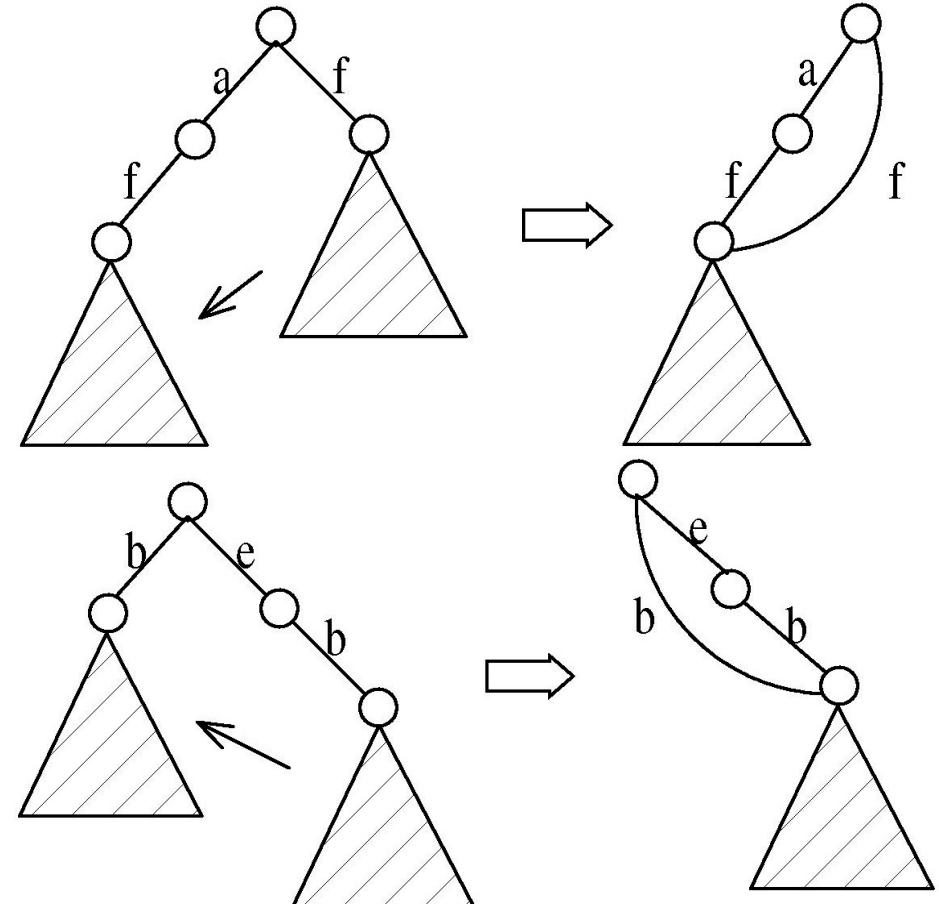


CloSpan: Mining Closed Sequential Patterns

- A **closed sequential pattern** s : There exists no superpattern s' such that $s' \supset s$, and s' and s have the same support
- Which ones are closed? $\langle abc \rangle: 20, \langle abcd \rangle: 20, \langle abcde \rangle: 15$

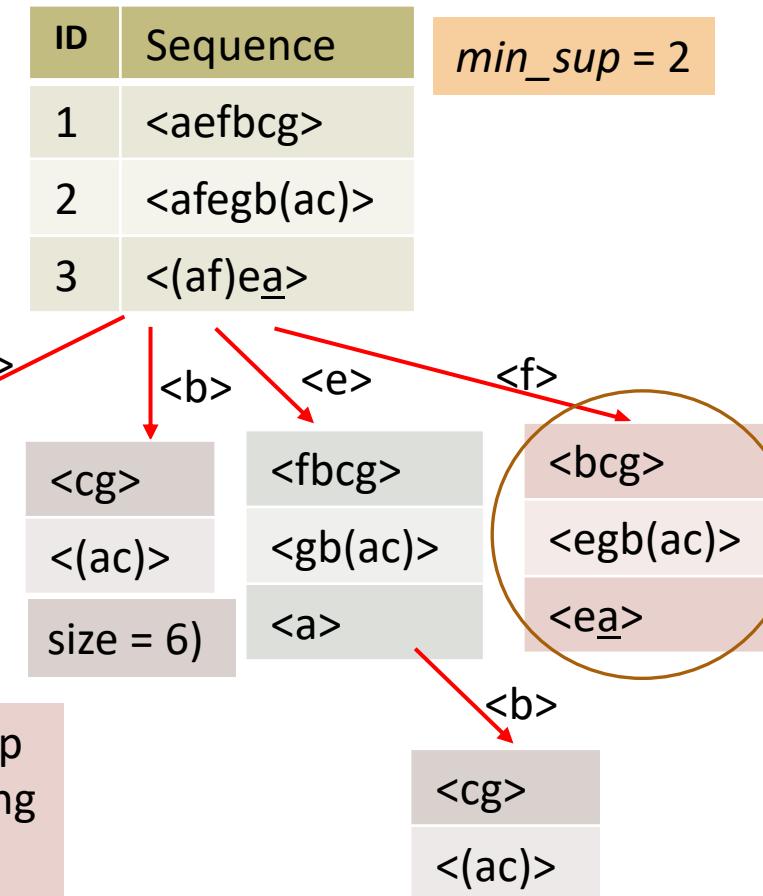
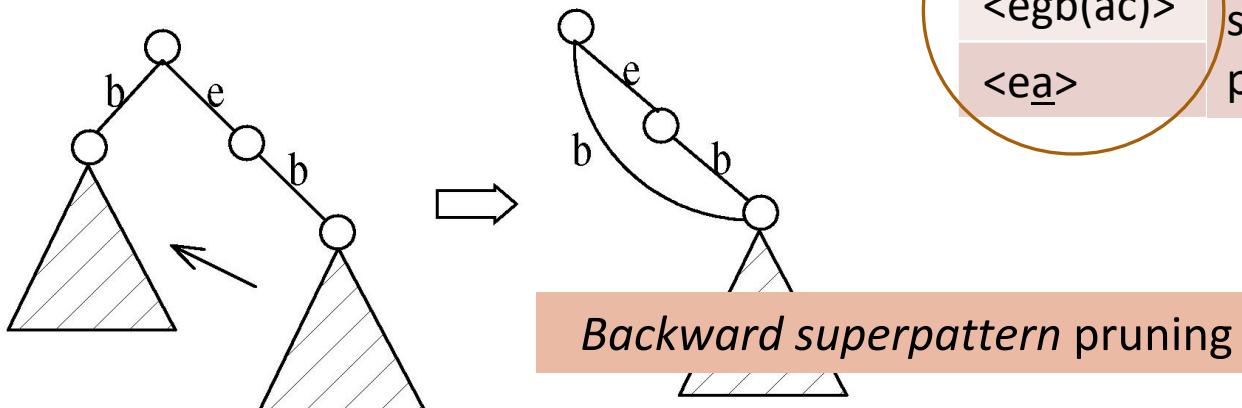
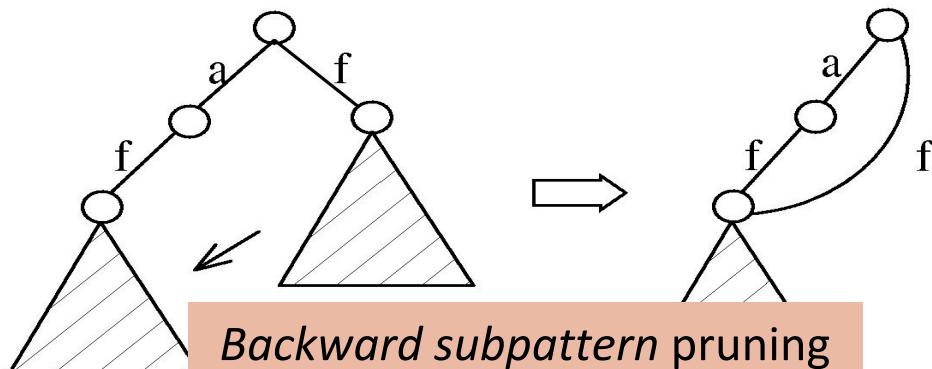
CloSpan: Mining Closed Sequential Patterns

- Why directly mine closed sequential patterns?
 - Reduce # of (redundant) patterns
 - Attain the same expressive power
- Property P₁: If $s \supset s_1$, s is closed iff two project DBs have the same size
- Explore *Backward Subpattern* and *Backward Superpattern* pruning to prune redundant search space
- Greatly enhances efficiency (Yan, et al., SDM'03)



CloSpan: When Two Projected DBs Have the Same Size

- If $s \supset s_1$, s is closed iff two project DBs have the same size
- When two projected sequence DBs have the same size?
- Here is one example:



Constraint-Based Sequential-Pattern Mining

- Share many similarities with constraint-based itemset mining
- **Anti-monotonic:** If S violates c , the super-sequences of S also violate c
 - $\text{sum}(S.\text{price}) < 150; \text{min}(S.\text{value}) > 10$
- **Monotonic:** If S satisfies c , the super-sequences of S also do so
 - $\text{element_count}(S) > 5; S \supseteq \{\text{PC}, \text{digital_camera}\}$
- **Data anti-monotonic:** If a sequence s_1 with respect to S violates c_3 , s_1 can be removed
 - $c_3: \text{sum}(S.\text{price}) \geq v$
- **Succinct:** Enforce constraint c by explicitly manipulating data
 - $S \supseteq \{\text{i-phone}, \text{MacAir}\}$
- **Convertible:** Projection based on the sorted value not sequence order
 - $\text{value_avg}(S) < 25; \text{profit_sum}(S) > 160$
 - $\text{max}(S)/\text{avg}(S) < 2; \text{median}(S) - \text{min}(S) > 5$

Timing-Based Constraints in Seq.-Pattern Mining

- **Order constraint:** Some items must happen before the other
 - $\{\text{algebra, geometry}\} \rightarrow \{\text{calculus}\}$ (where “ \rightarrow ” indicates ordering)
 - Anti-monotonic: Constraint-violating sub-patterns pruned
- **Min-gap/max-gap constraint:** Confines two elements in a pattern
 - E.g., mingap = 1, maxgap = 4
 - Succinct: Enforced directly during pattern growth
- **Max-span constraint:** Maximum allowed time difference between the 1st and the last elements in the pattern
 - E.g., maxspan (S) = 60 (days)
 - Succinct: Enforced directly when the 1st element is determined
- **Window size constraint:** Events in an element do not have to occur at the same time: Enforce max allowed time difference
 - E.g., window-size = 2: Various ways to merge events into elements

Episodes and Episode Pattern Mining

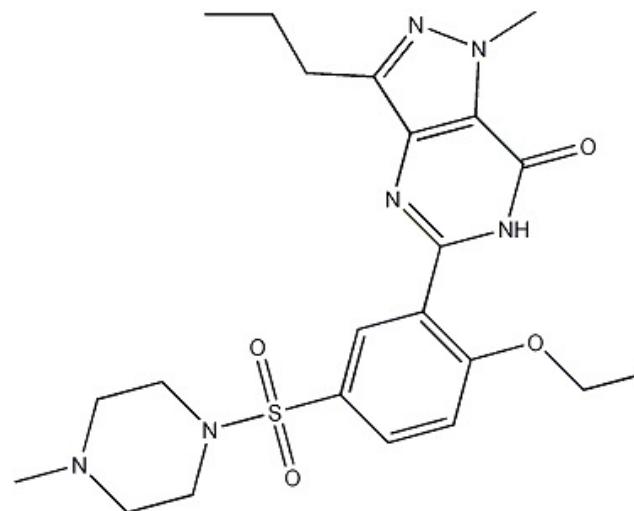
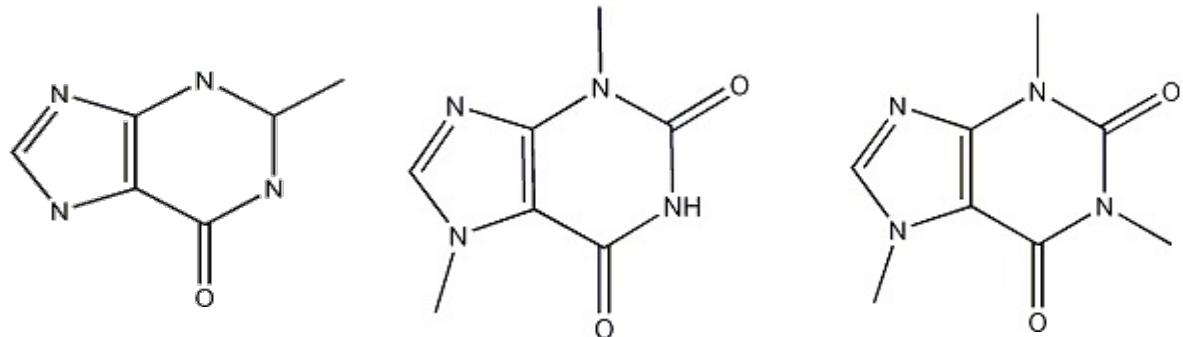
- ❑ Episodes and regular expressions: Alternative to seq. patterns
 - ❑ Serial episodes: AB  a total order relationship: first A then B
 - ❑ Parallel episodes: A|B  a partial order relationship: A and B can be in any order
 - ❑ Regular expressions: (A|B)C*(DE)  (DE) means D, E happen in the same time window
- ❑ E.g. Given a large shopping sequence database, one may like to find
 - ❑ Suppose the pattern order follows the template (A|B)C*(D E), and
 - ❑ Sum of the prices of A, B, C*, D, and E is greater than \$100, where C* means C appears *-times
 - ❑ How to efficiently mine such episode patterns?

Chapter 6 : Advanced Frequent Pattern Mining

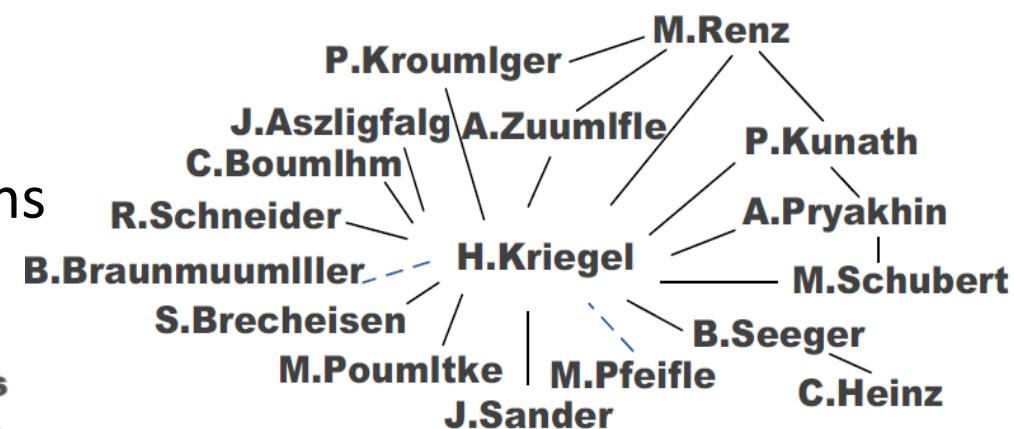
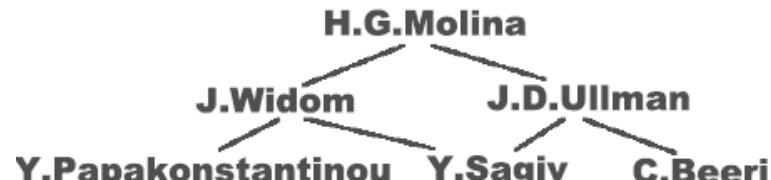
- Mining Diverse Patterns
- Constraint-Based Frequent Pattern Mining
- Sequential Pattern Mining
- Graph Pattern Mining 
- Pattern Mining Application: Mining Software Copy-and-Paste Bugs
- Summary

What Is Graph Pattern Mining?

- Chem-informatics:
 - Mining frequent chemical compound structures

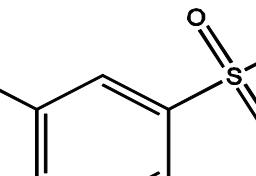


- Social networks, web communities, tweets, ...
 - Finding frequent research collaboration subgraphs

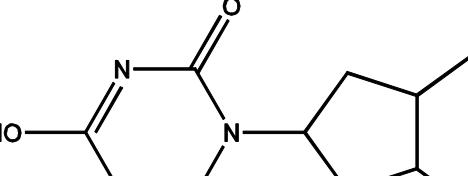


Frequent (Sub)Graph Patterns

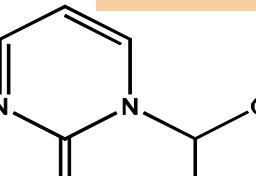
- Given a labeled graph dataset $D = \{G_1, G_2, \dots, G_n\}$, the supporting graph set of a subgraph g is $D_g = \{G_i \mid g \subseteq G_i, G_i \in D\}$
 - $\text{support}(g) = |D_g| / |D|$
 - A (sub)graph g is **frequent** if $\text{support}(g) \geq \text{min_sup}$
 - Ex.: Chemical structures
 - Alternative:
 - Mining frequent subgraph patterns from a single large graph or network



(A)



(B)

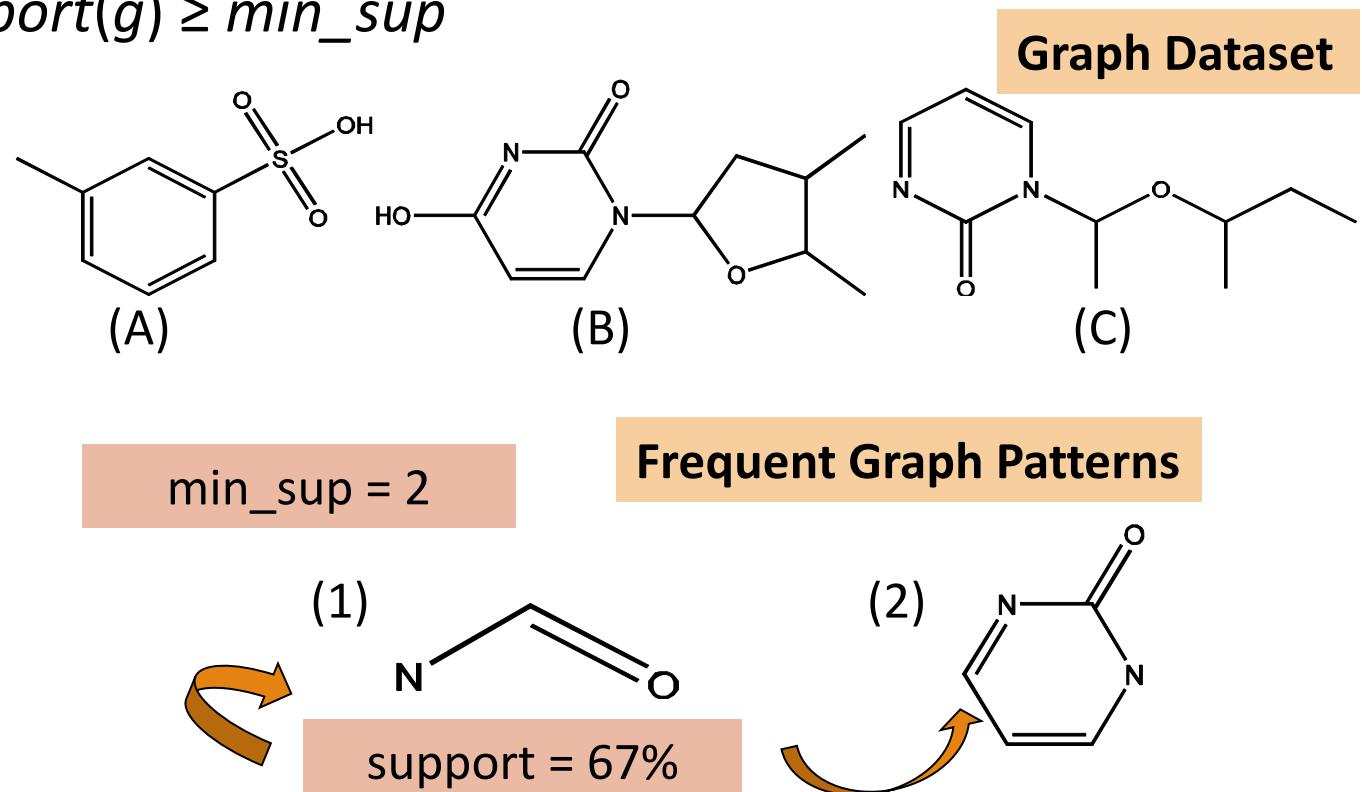


(C)

Graph D

min_sup = 2

Frequent Graph Patterns



Applications of Graph Pattern Mining

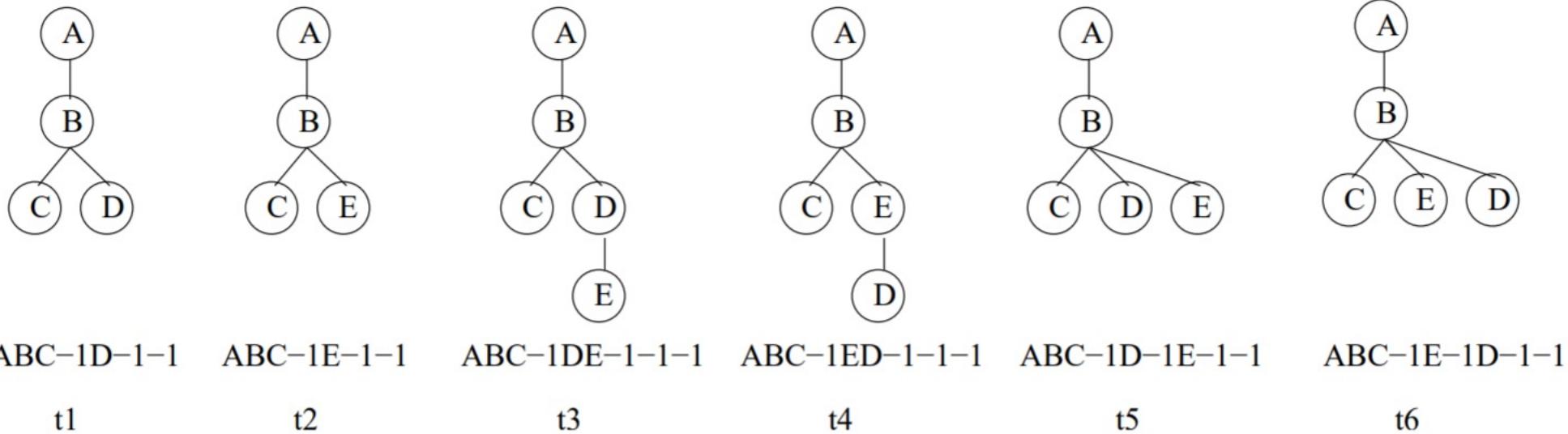
- Bioinformatics
 - Gene networks, protein interactions, metabolic pathways
- Chem-informatics: Mining chemical compound structures
- Social networks, web communities, tweets, ...
- Cell phone networks, computer networks, ...
- Web graphs, XML structures, Semantic Web, information networks
- Software engineering: Program execution flow analysis
- Building blocks for graph classification, clustering, compression, comparison, and correlation analysis
- Graph indexing and graph similarity search

Graph Pattern Mining Algorithms: Different Methodologies

- Generation of candidate subgraphs
 - Apriori vs. pattern growth (e.g., FSG vs. gSpan)
- Search order
 - Breadth vs. depth
- Elimination of duplicate subgraphs
 - Passive vs. active (e.g., gSpan [Yan & Han, 2002])
- Support calculation
 - Store embeddings (e.g., GASTON [Nijssen & Kok, 2004], FFSM [Huan, Wang, & Prins, 2003], MoFa [Borgelt & Berthold, ICDM'02])
- Order of pattern discovery
 - Path → tree → graph (e.g., GASTON [Nijssen & Kok, 2004])

Key Ideas: Frequent Trees

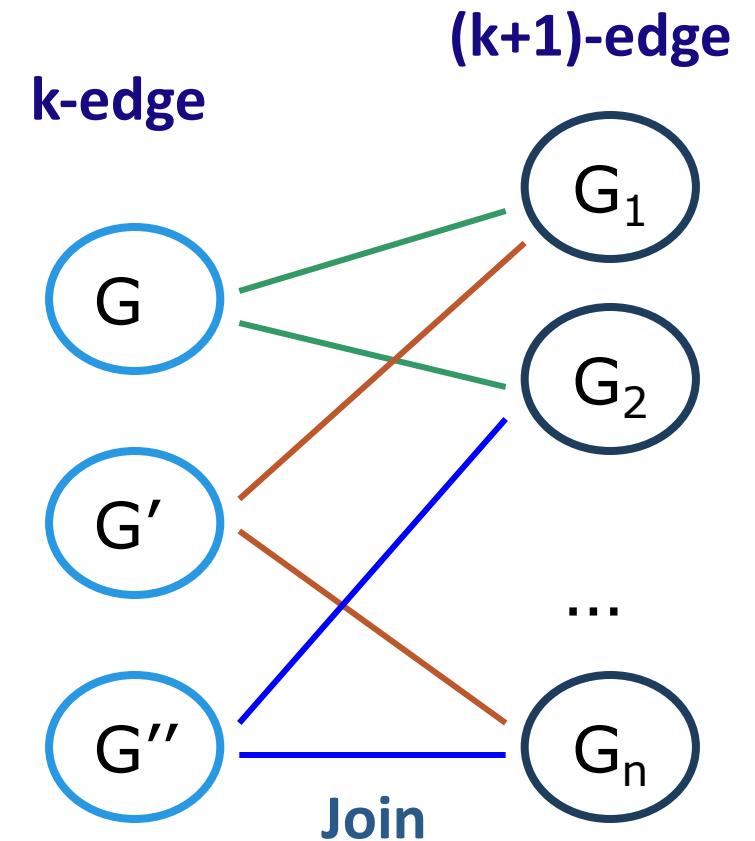
- String encoding (TreeMiner [Zaki, 2002])



- Frequent sequence mining

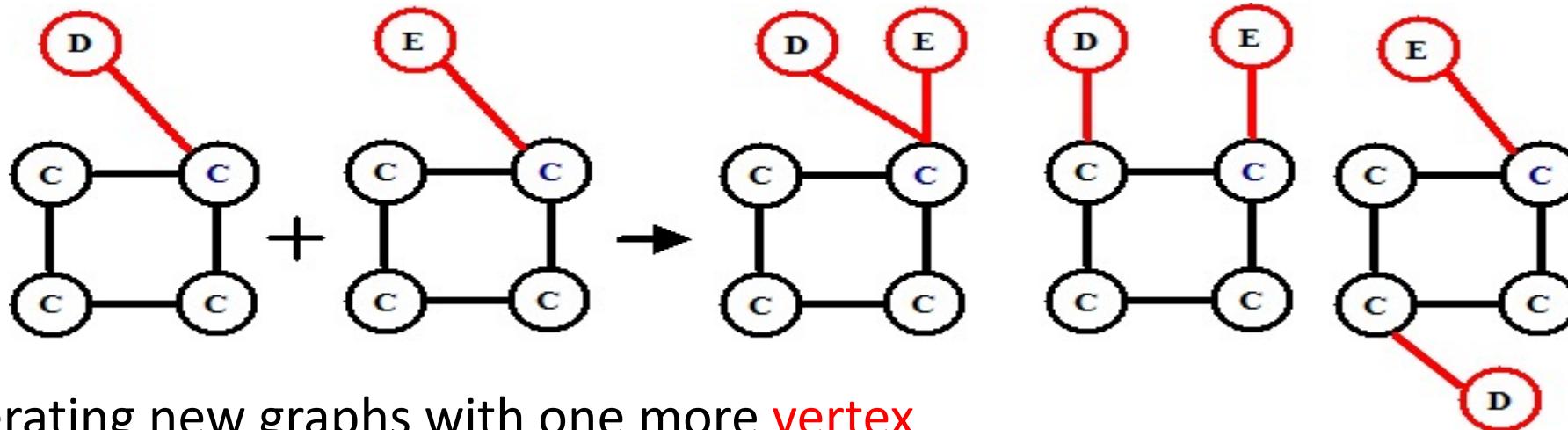
Apriori-Based Approach

- The Apriori property (anti-monotonicity): A size- k subgraph is **frequent** if and only if all of its **subgraphs are frequent**
- A candidate size- $(k+1)$ edge/vertex subgraph is generated if its corresponding two k -edge/vertex subgraphs are frequent
- Iterative mining process:
 - Candidate-generation → candidate pruning → support counting → candidate elimination



Candidate Generation: Vertex Growing vs. Edge Growing

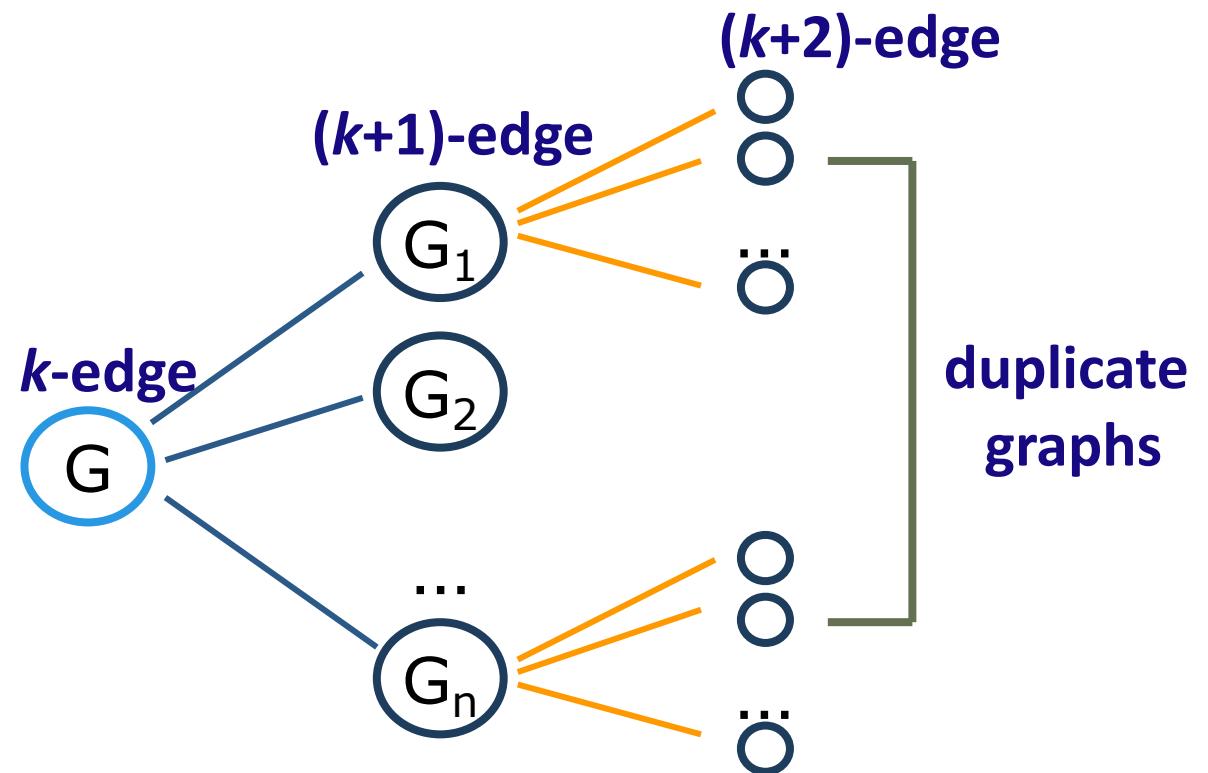
- Methodology: Breadth-search, Apriori joining two size- k graphs
 - Many possibilities at generating size- $(k+1)$ candidate graphs



- Generating new graphs with one more **vertex**
 - AGM (Inokuchi, Washio, & Motoda, PKDD'00)
- Generating new graphs with one more **edge**
 - FSG (Kuramochi & Karypis, ICDM'01)
- Performance shows *via edge growing* is more efficient

Pattern-Growth Approach

- ❑ Depth-first growth of subgraphs from k -edge to $(k+1)$ -edge, then $(k+2)$ -edge subgraphs
- ❑ Major challenge
 - ❑ Generating many duplicate subgraphs
- ❑ Major idea to solve the problem
 - ❑ Define an order to generate subgraphs
 - ❑ DFS spanning tree: Flatten a graph into a sequence using depth-first search
 - ❑ gSpan (Yan & Han, ICDM'02)



Pattern Growth on Graphs

- Pattern Growth can use BFS (like Apriori) or DFS
- Pattern growth by adding a new edge
 - May or may not introduce a new vertex
- General strategy: PatternGrowthGraph
 - May not be able to avoid duplicate graphs, can be inefficient

Algorithm: PatternGrowthGraph(g, D, minsup, S)

Input: A frequent graph g , a graph data set D , and the support threshold minsup .

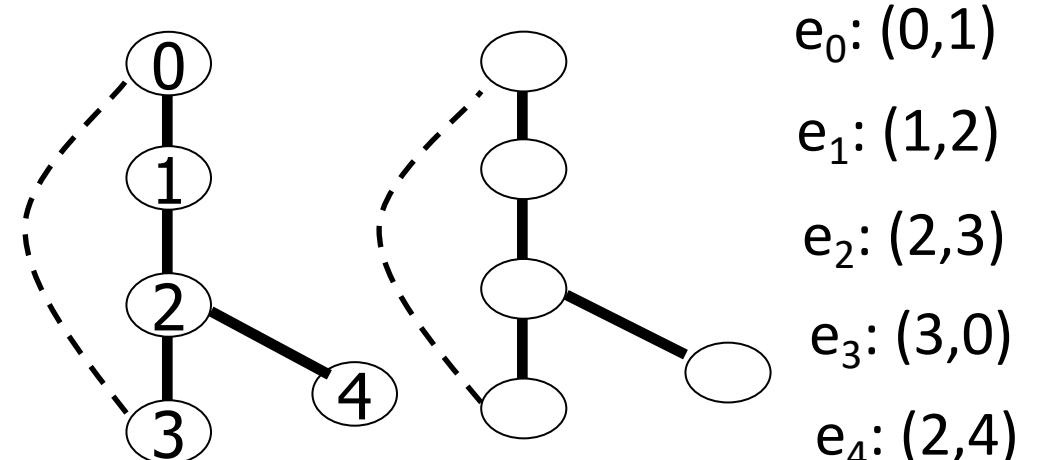
Output: The frequent graph set S .

```
1: if  $g \in S$  then return;  
2: else insert  $g$  to  $S$ ;  
3: scan  $D$  once, find all the edges  $e$  such that  $g$  can be extended to  $g \diamond_x e$  ;  
4: for each frequent  $g \diamond_x e$  do  
5:   Call PatternGrowthGraph( $g \diamond_x e, D, \text{minsup}, S$ );  
6: return;
```

Figure 6.15: PatternGrowthGraph.

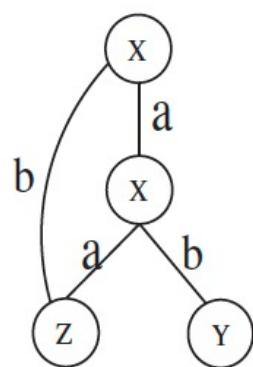
gSPAN: Graph Pattern Growth in Order

- **Right-most path extension** in subgraph pattern growth
- Right-most path: The path from root to the right-most leaf (choose the vertex with the **smallest** index at each step)
- Reduce generation of duplicate subgraphs
- **Completeness:** The enumeration of graphs using right-most path extension is complete
- DFS code: Flatten a graph into a sequence using depth-first search

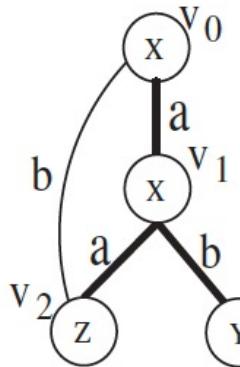


gSpan: Graph Traversal, DFS subscripting

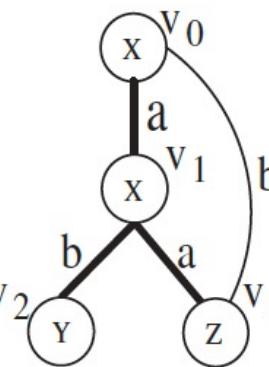
- Traverses the graph by DFS (Depth First Search), starting from some initial vertex
 - Create a full DFS tree
- Different DFS trees for the same graph
- Visit order of vertices in DFS tree noted as v_0 (root), v_1, v_2, \dots, v_n (right most vertex)
- DFS tree T is the DFS subscripting of G



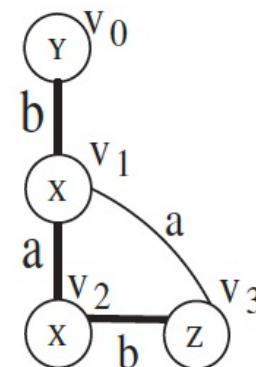
(a)



(b)



(c)



(d)

Right most path:
Straight path from v_0 to v_n

(b), (c) v_0, v_1, v_3
(d) v_0, v_1, v_2, v_3

Figure 6.16: DFS subscripting.

Graph Extension: Backward and Forward

- Restrict edge extension, on DFS tree T
- Backward extension: Right-most vertex & vertices in right most edge
- Forward extension: New vertex & vertices in the right most path

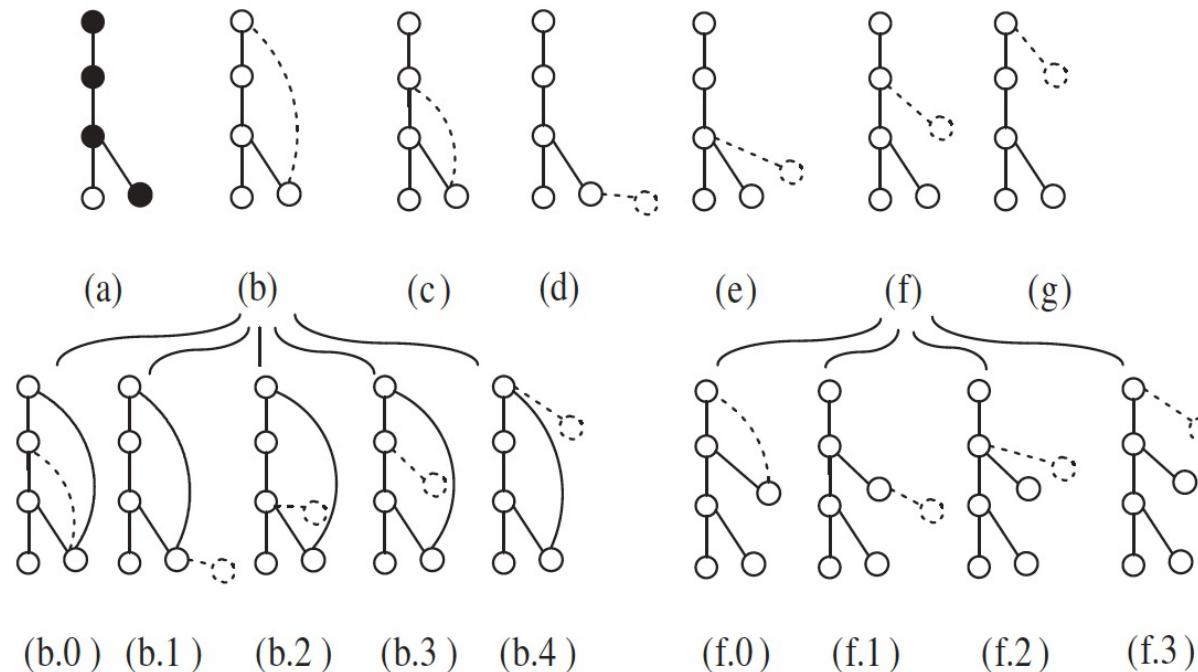


Figure 6.17: Right-Most Extension

Handling Multiple DFS Trees: DFS Codes

- Convert each DFS tree subscripted graph to an edge sequence: DFS code
- Two kinds of orders:
 - Edge order: Forward edges in DFS, add backward edges before forward edges from vertex
 - Sequence order: Order among edge sequences, i.e., graphs
 - Forward edge order for 6.16(b): $(0,1), (1,2), (1,3)$
 - Edge order for 6.16(b): $(0,1), (1,2), (2,0), (1,3)$
 - Represent edge as a tuple $(i,j, l_i, l_{(i,j)}, l_j)$, order based on
 - Edge order (i,j) , labels in vertices l_i, l_j and edges $l_{(i,j)}$

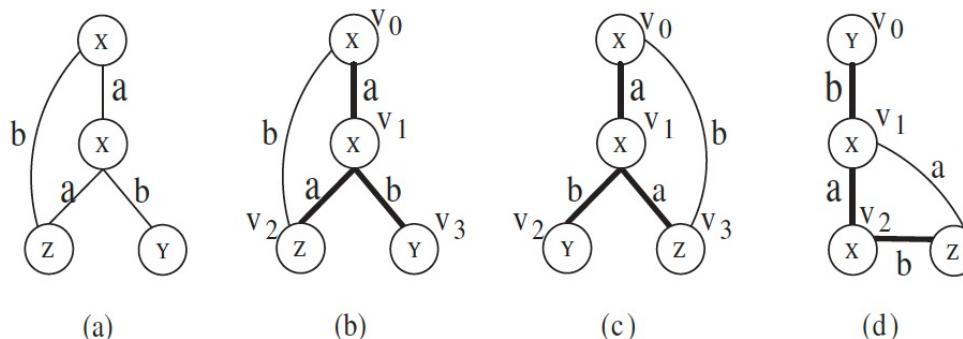


Figure 6.16: DFS subscripting.

Table 6.6: DFS code for Figure 6.16(b), 6.16(c), and 6.16(d).

edge	γ_0	γ_1	γ_2
e_0	$(0, 1, X, a, X)$	$(0, 1, X, a, X)$	$(0, 1, Y, b, X)$
e_1	$(1, 2, X, a, Z)$	$(1, 2, X, b, Y)$	$(1, 2, X, a, X)$
e_2	$(2, 0, Z, b, X)$	$(1, 3, X, a, Z)$	$(2, 3, X, b, Z)$
e_3	$(1, 3, X, b, Y)$	$(3, 0, Z, b, X)$	$(3, 1, Z, a, X)$

DFS Codes: Lexicographic Order

- One graph may have several DFS codes
- Edge as a 5-tuple: $(i, j, l_i, l_{(i,j)}, l_j)$, order based on
 - Edge order (i, j) , labels in vertices l_i, l_j and edges $l_{(i,j)}$
- Ordering
 - Edge order (i, j)
 - Vertex label l_i
 - Edge label $l_{(i,j)}$,
 - Vertex label l_j
- Use the minimum DFS code (G) for graph G
- Graphs G, G' are isomorphic if $(G) = (G')$

DFS Codes: Lexicographic Order

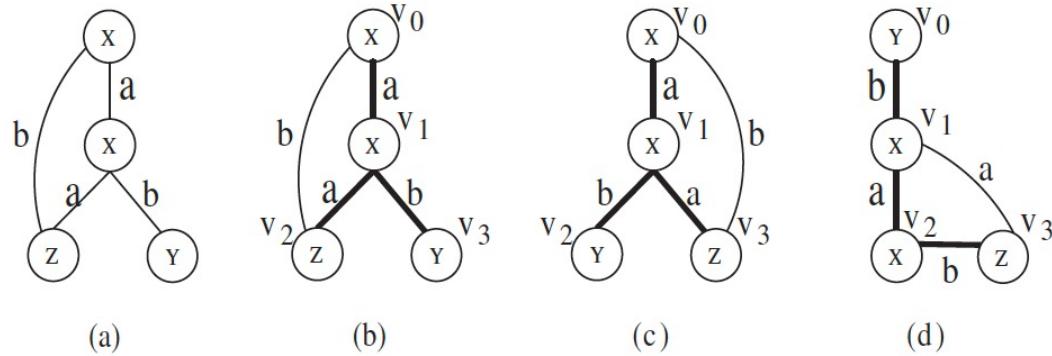


Figure 6.16: DFS subscripting.

- We have: $(0,1,X,a,X) < (0,1,Y,b,Y)$ so that $\gamma_0, \gamma_1 < \gamma_2$
 - We have: $(1,2,X,a,Z) < (1,2,X,b,Y)$ so that $\gamma_0 < \gamma_1$
 - Final ordering: $\gamma_0 < \gamma_1 < \gamma_2$
-
- Use the minimum DFS code (G) for graph G
 - Graphs G, G' are isomorphic if $(G) = (G')$
 - Right most extensions only on the minimum DFS codes, guarantees completeness

edge	γ_0	γ_1	γ_2
e_0	$(0, 1, X, a, X)$	$(0, 1, X, a, X)$	$(0, 1, Y, b, X)$
e_1	$(1, 2, X, a, Z)$	$(1, 2, X, b, Y)$	$(1, 2, X, a, X)$
e_2	$(2, 0, Z, b, X)$	$(1, 3, X, a, Z)$	$(2, 3, X, b, Z)$
e_3	$(1, 3, X, b, Y)$	$(3, 0, Z, b, X)$	$(3, 1, Z, a, X)$

Table 6.6: DFS code for Figure 6.16(b), 6.16(c), and 6.16(d).

Lexicographic Search Tree

- Arranging DFS codes in a search tree, with right most extensions
- Each node: DFS code encoding a graph
- Each edge: Right-most extension from $(k-1)$ - to k -length DFS code
- Left sibling < right sibling, in lexicographic order
- Work with minimum DFS codes
 - Can drop non-minimum DFS codes
 - Key difference between PatternGrowth and gSpan

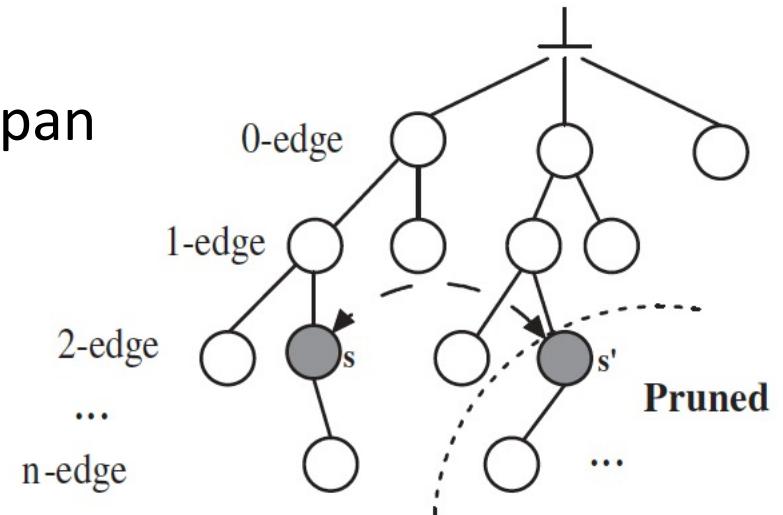


Figure 6.18: Lexicographic search tree.

gSpan Algorithm

Algorithm: $\text{gSpan}(s, D, \text{minsup}, S)$

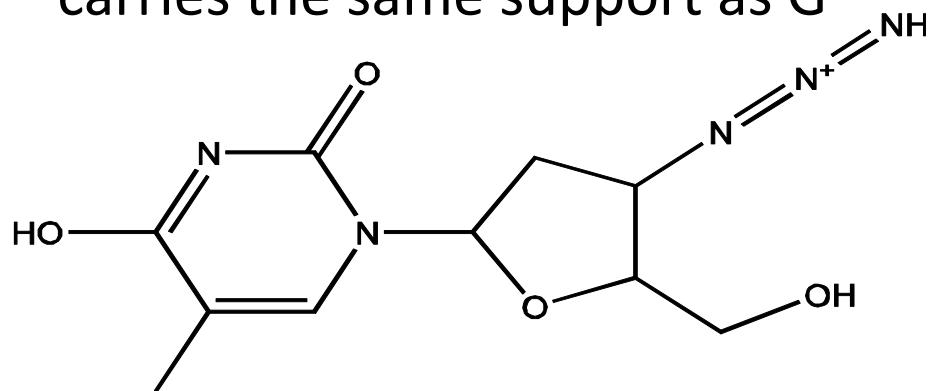
Input: A DFS code s , a graph data set D , and .

Output: The frequent graph set S .

- 1: **if** $s \neq \text{dfs}(s)$, **then**
- 2: **return**;
- 3: insert s into S ;
- 4: set C to \emptyset ;
- 5: scan D once, find all the edges e such that s can be *right-most* extended to $s \diamond_r e$;
 insert $s \diamond_r e$ into C and count its frequency;
- 6: sort C in DFS lexicographic order;
- 7: **for each** frequent $s \diamond_r e$ in C **do**
- 8: Call $\text{gSpan}(s \diamond_r e, D, \text{minsup}, S)$;
- 9: **return**;

Why Mine Closed Graph Patterns?

- 2^n subgraphs -> *closed frequent subgraphs*
- A frequent graph G is *closed* if there exists no supergraph of G that carries the same support as G

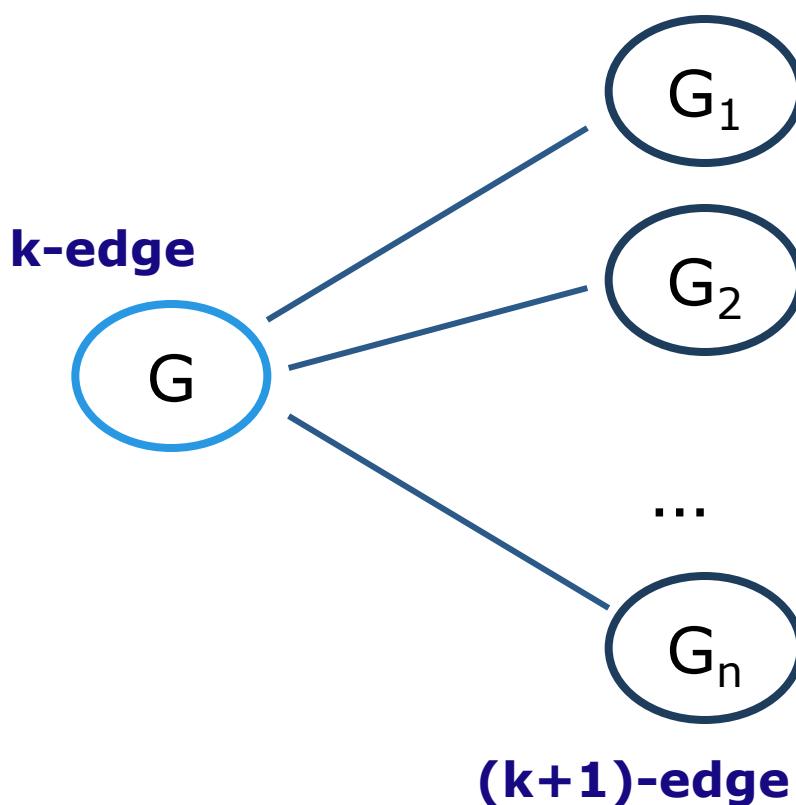


If this subgraph is *closed* in the graph dataset, it implies that none of its frequent super-graphs carries the same support

- *Lossless compression*: Does not contain non-closed graphs, but still ensures that the mining result is complete
- Algorithm CloseGraph: Mines closed graph patterns directly

CloseGraph: Directly Mining Closed Graph Patterns

- CloseGraph: Mining closed graph patterns by extending gSpan (Yan & Han, KDD'03)

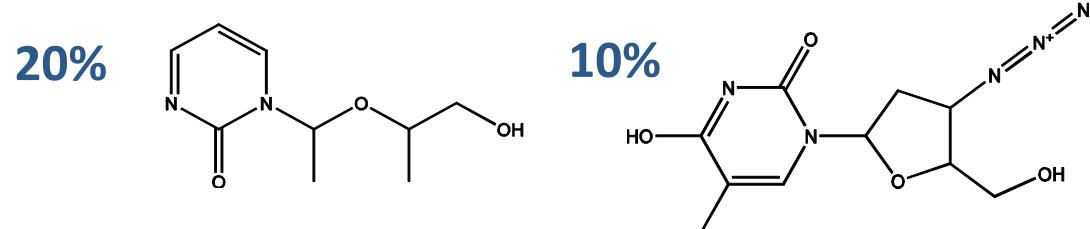


At what condition can we stop searching their children, i.e., early termination?

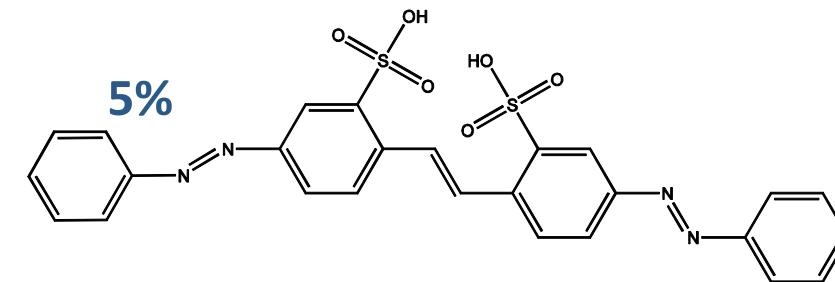
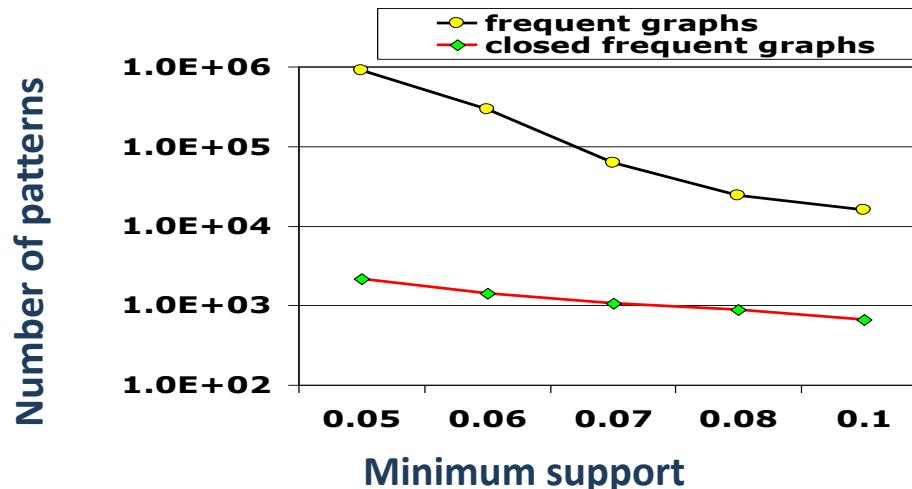
- Suppose G and G_1 are frequent, and G is a subgraph of G_1
- If **in any part of the graph in the dataset where G occurs, G_1 also occurs**, then we need not grow G (except some special, subtle cases), since none of G 's children will be closed except those of G_1

Experiment and Performance Comparison

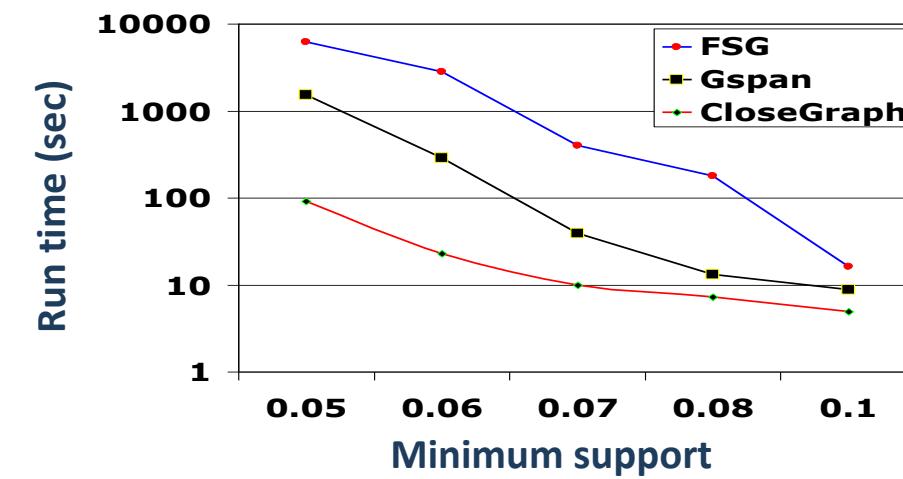
- The AIDS antiviral screen compound dataset from NCI/NIH
- The dataset contains 43,905 chemical compounds
- Discovered patterns: The smaller minimum support, the bigger and more interesting subgraph patterns discovered



of Patterns: Frequent vs. Closed



Runtime: Frequent vs. Closed



Chapter 6 : Advanced Frequent Pattern Mining

- ❑ Mining Diverse Patterns
- ❑ Constraint-Based Frequent Pattern Mining
- ❑ Sequential Pattern Mining
- ❑ Graph Pattern Mining
- ❑ Pattern Mining Application: Mining Software Copy-and-Paste Bugs 
- ❑ Summary

Pattern Mining Application: Software Bug Detection

- **Mining rules from source code**
 - Bugs as deviant behavior (e.g., by statistical analysis)
 - Mining programming rules (e.g., by frequent itemset mining)
 - Mining function precedence protocols (e.g., by frequent subsequence mining)
 - Revealing neglected conditions (e.g., by frequent itemset/subgraph mining)
- **Mining rules from revision histories**
 - By frequent itemset mining
- **Mining copy-paste patterns from source code**
 - Find copy-paste bugs (e.g., CP-Miner [Li et al., OSDI'04]) (to be discussed here)
 - Reference: Z. Li, S. Lu, S. Myagmar, Y. Zhou, “[CP-Miner](#): A Tool for Finding Copy-paste and Related Bugs in Operating System Code”, OSDI’04

Application Example: Mining Copy-and-Paste Bugs

- ❑ Copy-pasting is common
 - ❑ 12% in Linux file system
 - ❑ 19% in X Window system
- ❑ Copy-pasted code is error-prone
- ❑ Mine “*forget-to-change*” bugs by sequential pattern mining
 - ❑ Build a sequence database from source code
 - ❑ Mining sequential patterns
 - ❑ Finding mismatched identifier names & bugs

```
void __init prom_meminit(void)
{
    .....
    for (i=0; i<n; i++) {
        total[i].adr = list[i].addr;
        total[i].bytes = list[i].size;
        total[i].more = &total[i+1];
    }
    .....
    for (i=0; i<n; i++) {
        taken[i].adr = list[i].addr;
        taken[i].bytes = list[i].size,
        taken[i].more = &total[i+1];
    }
}
```

Code copy-and-pasted but **forget to change “id”!**

Chapter 6 : Advanced Frequent Pattern Mining

- Mining Diverse Patterns
- Constraint-Based Frequent Pattern Mining
- Sequential Pattern Mining
- Graph Pattern Mining
- Pattern Mining Application: Mining Software Copy-and-Paste Bugs
- Summary 

Summary: Advanced Frequent Pattern Mining

- ❑ Mining Diverse Patterns
 - ❑ Mining Multiple-Level Associations
 - ❑ Mining Multi-Dimensional Associations
 - ❑ Mining Quantitative Associations
 - ❑ Mining Negative Correlations
 - ❑ Mining Compressed and Redundancy-Aware Patterns
- ❑ Sequential Pattern Mining
 - ❑ Sequential Pattern and Sequential Pattern Mining
 - ❑ GSP: Apriori-Based Sequential Pattern Mining
 - ❑ SPADE: Sequential Pattern Mining in Vertical Data Format
 - ❑ PrefixSpan: Sequential Pattern Mining by Pattern-Growth
 - ❑ CloSpan: Mining Closed Sequential Patterns
- ❑ Constraint-Based Frequent Pattern Mining
 - ❑ Why Constraint-Based Mining?
 - ❑ Constrained Mining with Pattern Anti-Monotonicity
 - ❑ Constrained Mining with Pattern Monotonicity
 - ❑ Constrained Mining with Data Anti-Monotonicity
 - ❑ Constrained Mining with Succinct Constraints
 - ❑ Constrained Mining with Convertible Constraints
 - ❑ Handling Multiple Constraints
 - ❑ Constraint-Based Sequential-Pattern Mining
- ❑ Graph Pattern Mining
 - ❑ Graph Pattern and Graph Pattern Mining
 - ❑ Apriori-Based Graph Pattern Mining Methods
 - ❑ gSpan: A Pattern-Growth-Based Method
 - ❑ CloseGraph: Mining Closed Graph Patterns
- ❑ Pattern Mining Application: Mining Software Copy-and-Paste Bugs

References: Mining Diverse Patterns

- R. Srikant and R. Agrawal, "Mining generalized association rules", VLDB'95
- Y. Aumann and Y. Lindell, "A Statistical Theory for Quantitative Association Rules", KDD'99
- K. Wang, Y. He, J. Han, "Pushing Support Constraints Into Association Rules Mining", IEEE Trans. Knowledge and Data Eng. 15(3): 642-658, 2003
- D. Xin, J. Han, X. Yan and H. Cheng, "On Compressing Frequent Patterns", Knowledge and Data Engineering, 60(1): 5-29, 2007
- D. Xin, H. Cheng, X. Yan, and J. Han, "Extracting Redundancy-Aware Top-K Patterns", KDD'06
- J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent Pattern Mining: Current Status and Future Directions", Data Mining and Knowledge Discovery, 15(1): 55-86, 2007
- F. Zhu, X. Yan, J. Han, P. S. Yu, and H. Cheng, "Mining Colossal Frequent Patterns by Core Pattern Fusion", ICDE'07

References: Constraint-Based Frequent 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
- G. Grahne, 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
- F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi, “ExAnte: Anticipated Data Reduction in Constrained Pattern Mining”, PKDD'03
- F. Zhu, X. Yan, J. Han, and P. S. Yu, “gPrune: A Constraint Pushing Framework for Graph Pattern Mining”, PAKDD'07

References: Sequential Pattern Mining

- R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements", EDBT'96
- M. Zaki, "SPADE: An Efficient Algorithm for Mining Frequent Sequences", Machine Learning, 2001
- J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach", IEEE TKDE, 16(10), 2004
- X. Yan, J. Han, and R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Datasets", SDM'03
- J. Pei, J. Han, and W. Wang, "Constraint-based sequential pattern mining: the pattern-growth methods", J. Int. Inf. Sys., 28(2), 2007
- M. N. Garofalakis, R. Rastogi, K. Shim: Mining Sequential Patterns with Regular Expression Constraints. IEEE Trans. Knowl. Data Eng. 14(3), 2002
- H. Mannila, H. Toivonen, and A. I. Verkamo, "Discovery of frequent episodes in event sequences", Data Mining and Knowledge Discovery, 1997

References: Graph Pattern Mining

- C. Borgelt and M. R. Berthold, Mining molecular fragments: Finding relevant substructures of molecules, ICDM'02
- J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraph in the presence of isomorphism, ICDM'03
- A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data, PKDD'00
- M. Kuramochi and G. Karypis. Frequent subgraph discovery, ICDM'01
- S. Nijssen and J. Kok. A Quickstart in Frequent Structure Mining can Make a Difference. KDD'04
- N. Vanetik, E. Gudes, and S. E. Shimony. Computing frequent graph patterns from semistructured data, ICDM'02
- X. Yan and J. Han, gSpan: Graph-Based Substructure Pattern Mining, ICDM'02
- X. Yan and J. Han, CloseGraph: Mining Closed Frequent Graph Patterns, KDD'03
- X. Yan, P. S. Yu, J. Han, Graph Indexing: A Frequent Structure-based Approach, SIGMOD'04
- X. Yan, P. S. Yu, and J. Han, Substructure Similarity Search in Graph Databases, SIGMOD'05





CS 412 Intro. to Data Mining

Chapter 7. Classification: Basic Concepts

Arindam Banerjee, Computer Science, UIUC, Fall 2021



Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Linear Classifier
- Model Evaluation and Selection
- Lazy Learning
- Ensemble Methods
- Imbalanced Data Sets
- Summary

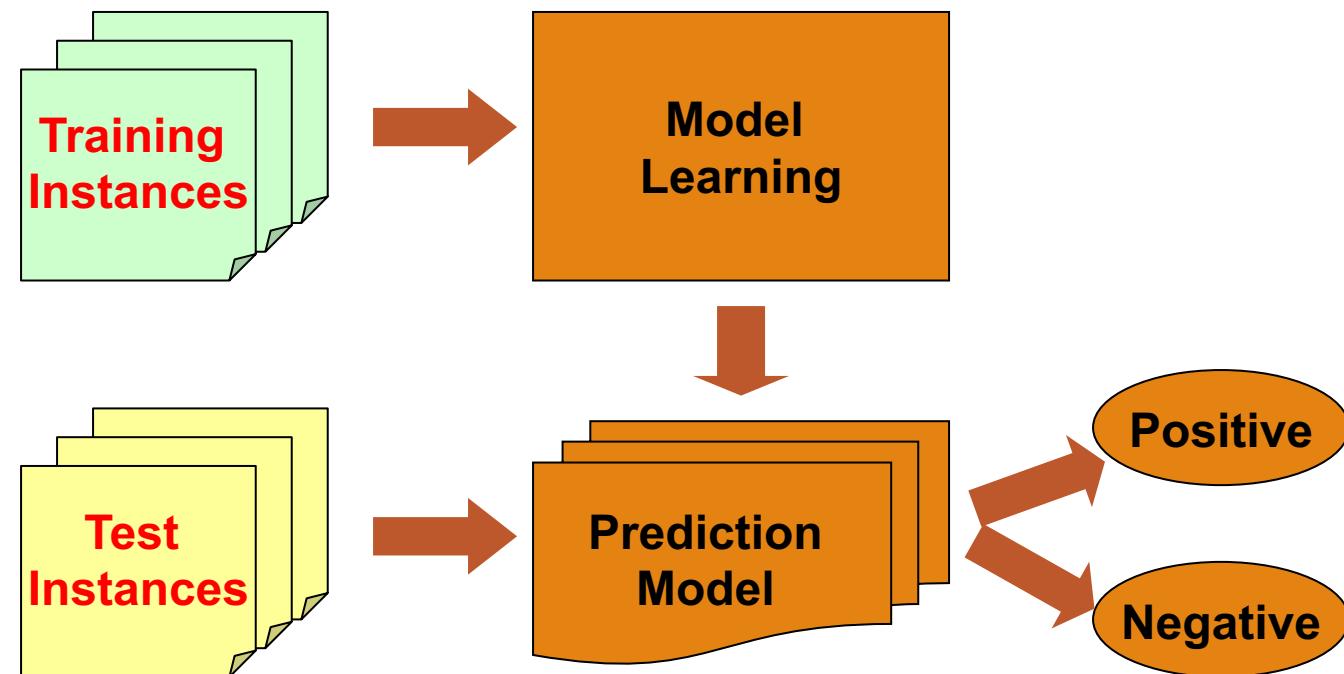


Supervised vs. Unsupervised Learning (1)

- Supervised learning (classification)
 - Supervision: The training data such as observations or measurements are accompanied by **labels** indicating the classes which they belong to
 - New data is classified based on the models built from the training set

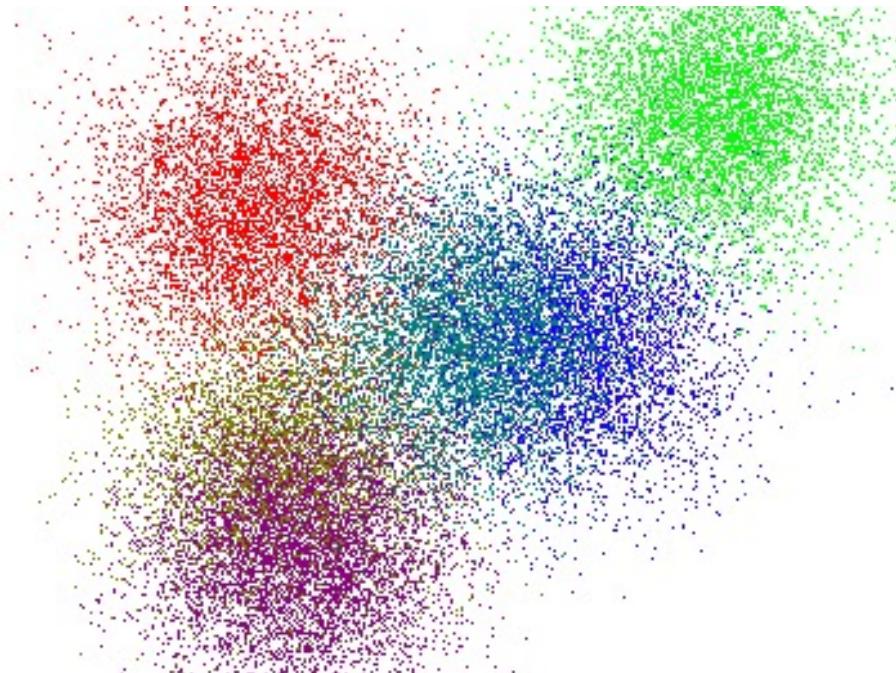
Training Data with class label:

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No



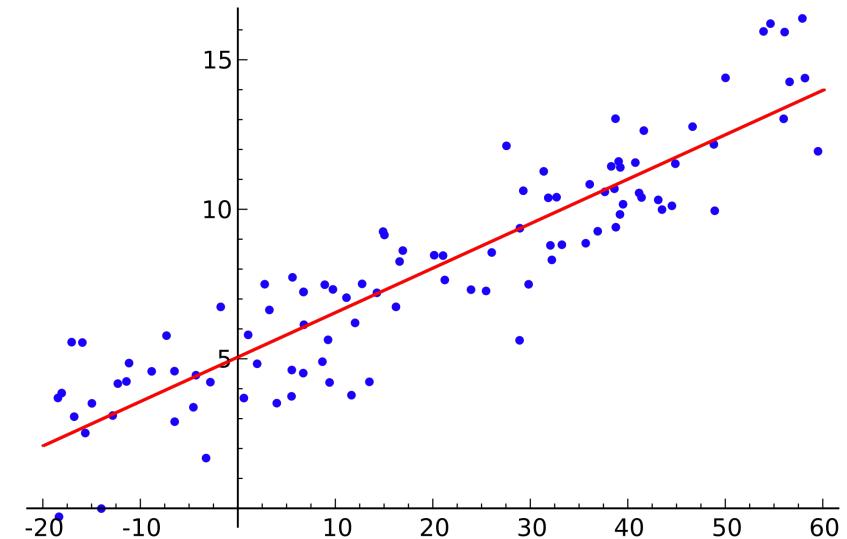
Supervised vs. Unsupervised Learning (2)

- Unsupervised learning (clustering)
 - The class labels of training data are **unknown**
 - Given a set of observations or measurements, establish the possible existence of classes or clusters in the data



Prediction Problems: Classification vs. Numeric Prediction

- ❑ Classification
 - ❑ Predict **categorical class labels** (discrete or nominal)
 - ❑ Construct a model based on the training set and the **class labels** (the values in a classifying attribute) and use it in classifying new data
- ❑ Numeric prediction
 - ❑ Model **continuous-valued functions** (i.e., predict unknown or missing values)



Prediction Problems: Classification vs. Numeric Prediction

- ❑ Typical applications of classification
 - ❑ Credit/loan approval
 - ❑ Medical diagnosis: if a tumor is cancerous or benign
 - ❑ Fraud detection: if a transaction is fraudulent
 - ❑ Web page categorization: which category it is

Classification—Model Construction, Validation and Testing

□ Model Construction and Training

- Model: Represented as decision trees, rules, mathematical formulas, or other forms
- Assumption: Each sample belongs to a predefined class /**class label**
- Training Set: The set of samples used for model construction

Classification—Model Construction, Validation and Testing

- **Model Validation and Testing:**
 - **Test:** Estimate accuracy of the model
 - The known label of test sample VS. the classified result from the model
 - **Accuracy:** % of test set samples that are correctly classified by the model
 - Test set is **independent** of training set
 - **Validation:** If *the test set* is used to select or refine models, it is called **validation (or development) (test) set**
 - **Model Deployment:** If the accuracy is acceptable, use the model to classify new data

Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Linear Classifier
- Model Evaluation and Selection
- Imbalanced Data Sets
- Additional Concepts on Classification
- Summary

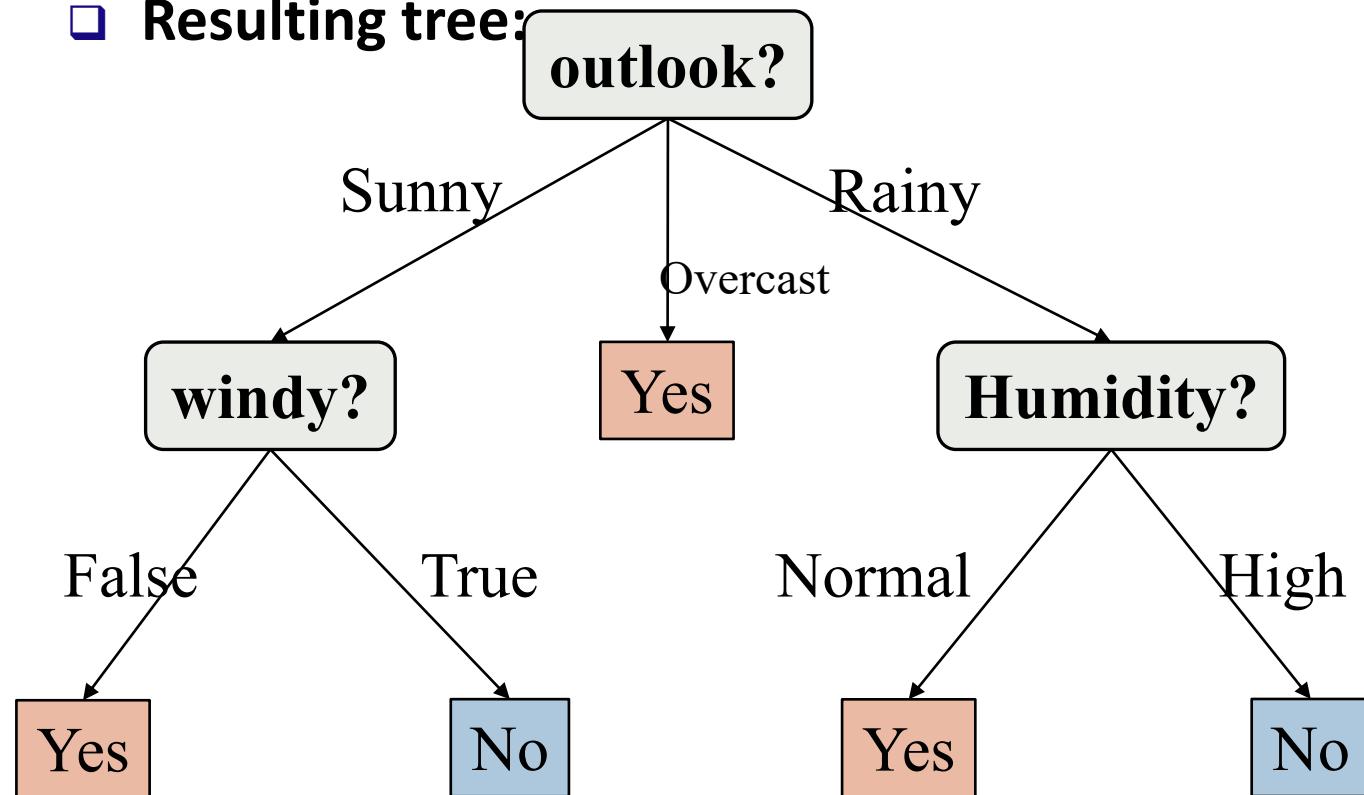


Decision Tree Induction: An Example

□ Decision tree construction:

- A top-down, recursive, divide-and-conquer process

□ Resulting tree:



Training data set: Play Golf?

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Decision Tree Induction: Algorithm

- Basic algorithm
 - Tree is constructed in a **top-down, recursive, divide-and-conquer manner**
 - At start, all the training examples are at the root
 - Examples are partitioned recursively based on selected attributes
 - On each node, attributes are selected based on the training examples on that node, and a heuristic or statistical measure (e.g., **information gain, Gini index**)

Decision Tree Induction: Algorithm

- Conditions for stopping partitioning
 - All samples for a given node belong to the same class
 - There are no remaining attributes for further partitioning
 - There are no samples left
- Prediction
 - **Majority voting** is employed for classifying the leaf

How to Handle Continuous-Valued Attributes?

- **Method 1:** Discretize continuous values and treat them as categorical values
 - E.g., age: < 20, 20..30, 30..40, 40..50, > 50
- **Method 2:** Determine the *best split point* for continuous-valued attribute A
 - Sort:, e.g. 15, 18, 21, 22, 24, 25, 29, 31, ...
 - *Possible split point:* $(a_i + a_{i+1})/2$
 - e.g., $(15+18)/2 = 16.5, 19.5, 21.5, 23, 24.5, 27, 30, \dots$
 - The point with the *maximum information gain* for A is selected as the **split-point** for A
- Split: Based on split point P
 - The set of tuples in D satisfying $A \leq P$ vs. those with $A > P$

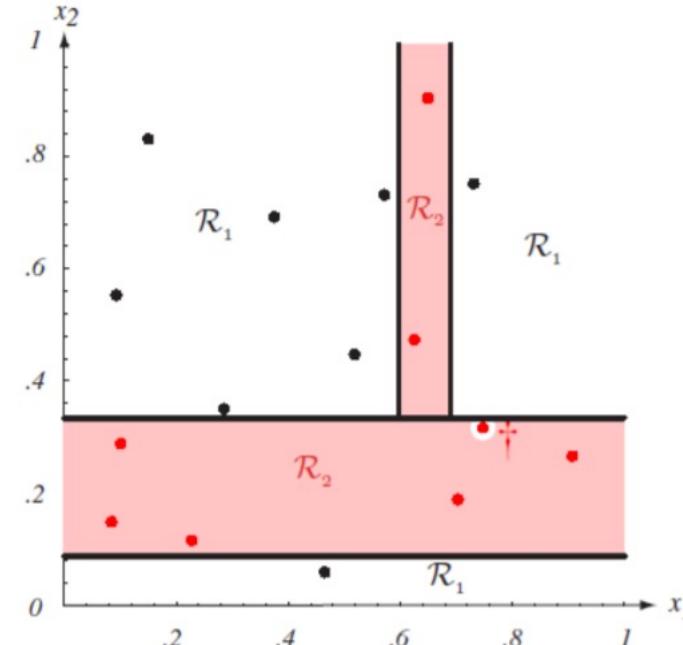
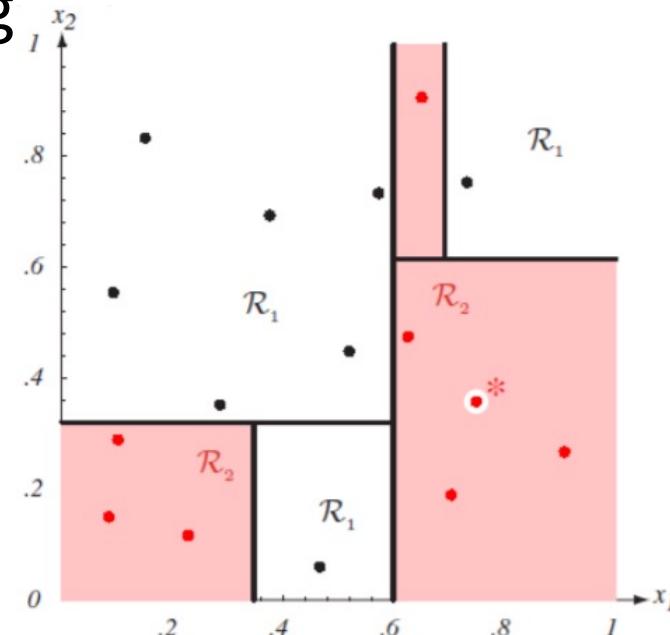
Pros and Cons

□ Pros

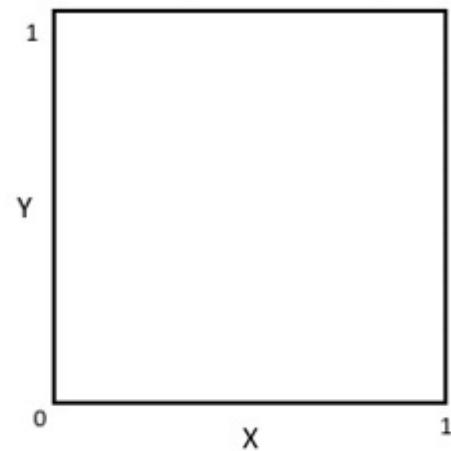
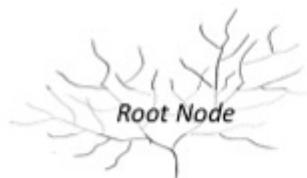
- Easy to explain (even for non-expert)
- Easy to implement (many software)
- Efficient
- Can tolerate missing data
- White box
- No need to normalize data
- Non-parametric: No assumption on data distribution, no assumption on attribute independency
- Can work on various attribute types

Pros and Cons

- ❑ Cons
 - ❑ Unstable. Sensitive to noise
 - ❑ Accuracy may be not good enough (depending on your data)
 - ❑ The optimal splitting is NP. Greedy algorithms are used
 - ❑ Overfitting



Overview



For more tutorials: annalyzin.wordpress.com

Example

Examples described by attribute values (Boolean, discrete, continuous, etc.)

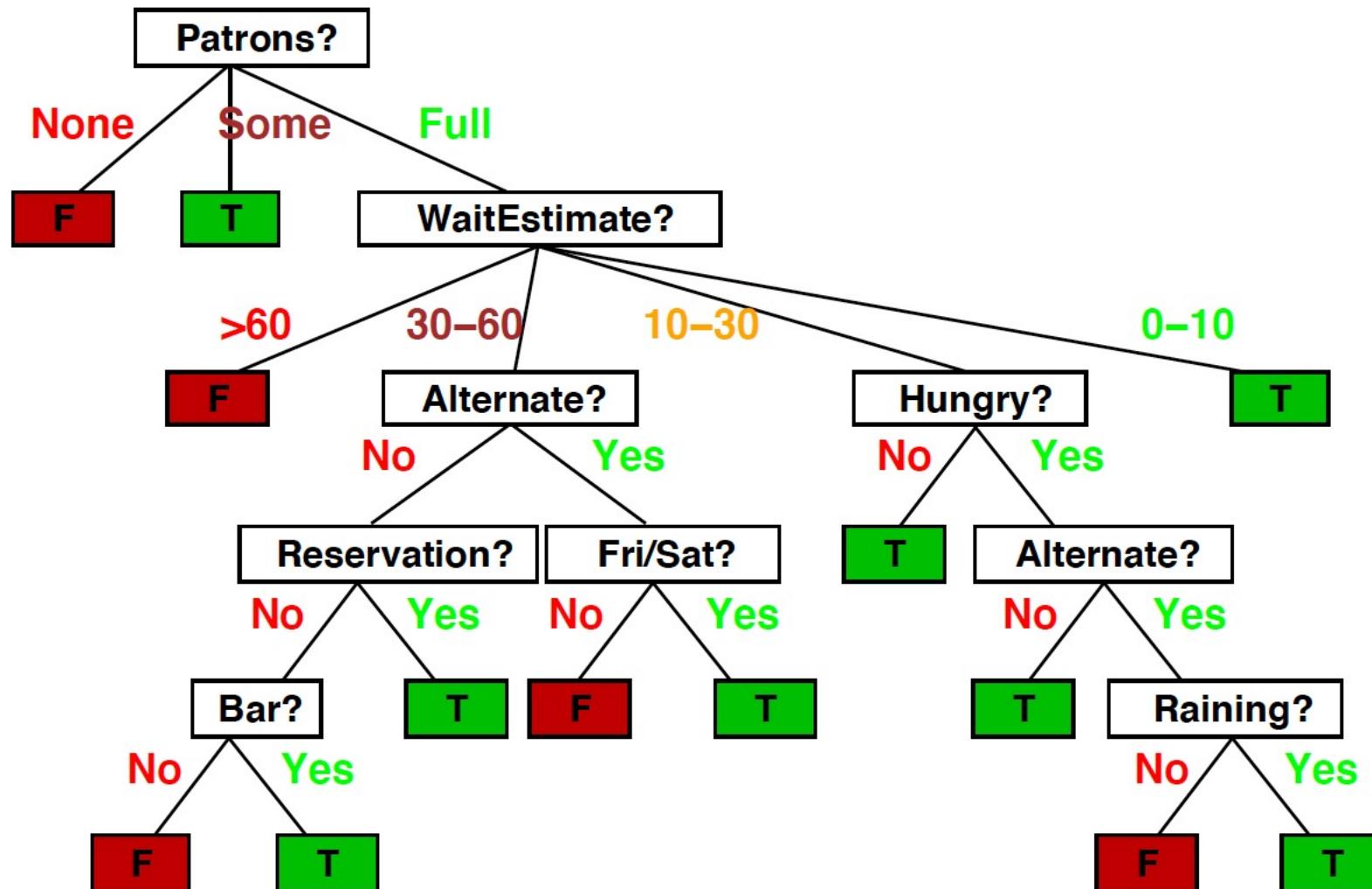
Example	Attributes										Target WillWait
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
X ₁	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X ₂	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X ₃	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X ₄	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X ₅	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X ₆	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X ₇	F	T	F	F	None	\$	T	F	Burger	0–10	F
X ₈	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X ₉	F	T	T	F	Full	\$	T	F	Burger	>60	F
X ₁₀	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X ₁₁	F	F	F	F	None	\$	F	F	Thai	0–10	F
X ₁₂	T	T	T	T	Full	\$	F	F	Burger	30–60	T

Classification of examples is positive (T) or negative (F)

Example

- Problem: Decide whether to wait for a table at a restaurant, based on the following attributes:
 - Alternate: is there an alternative restaurant nearby?
 - Bar: is there a comfortable bar area to wait in?
 - Fri/Sat: is today Friday or Saturday?
 - Hungry: are we hungry?
 - Patrons: number of people in the restaurant (None, Some, Full)
 - Price: price range (\$, \$\$, \$\$\$)
 - Raining: is it raining outside?
 - Reservation: have we made a reservation?
 - Type: kind of restaurant (French, Italian, Thai, Burger)
 - WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, > 60)

Example: Decision Tree

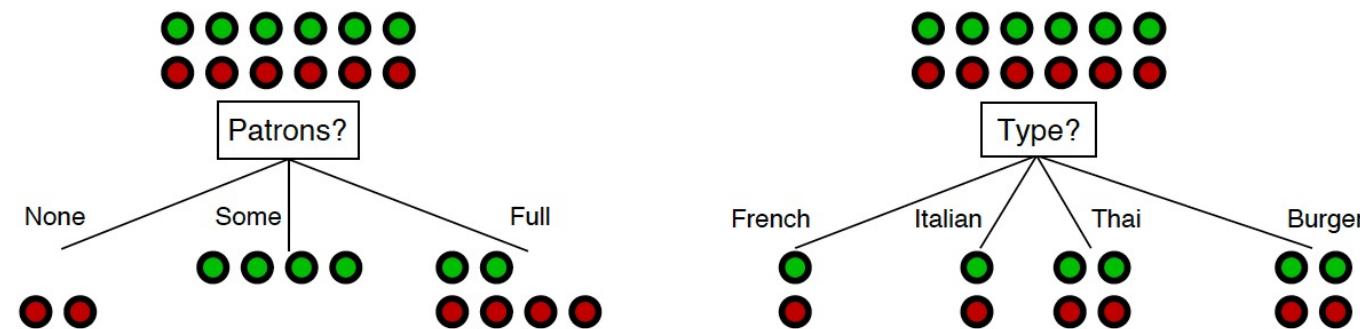


Example: Decision Tree Construction

Aim: Find a small tree consistent with the training examples

Recursively choose “most significant” attribute as root of (sub)tree

Good attribute splits the examples (ideally) into “pure” subsets



Patrons? is a better choice

—gives *information* about the classification

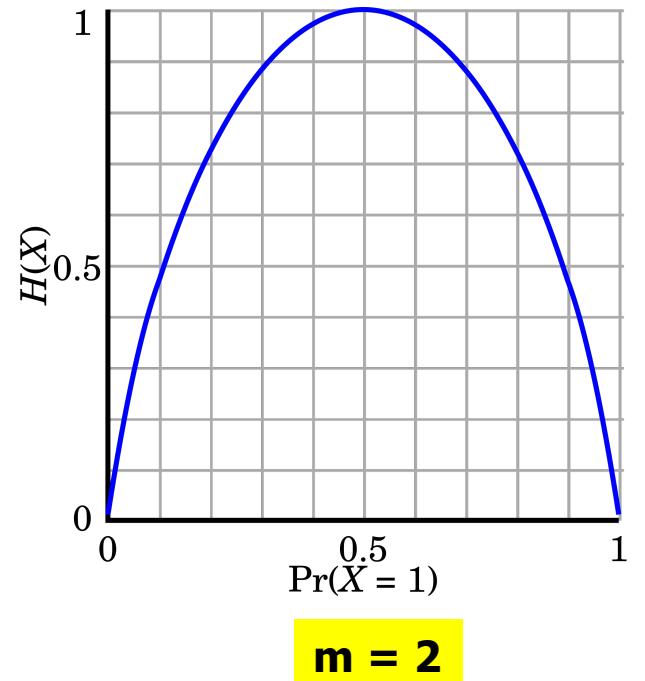
Splitting Measures: Information Gain

- Entropy (Information Theory)
 - A measure of uncertainty associated with a random number
 - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, y_2, \dots, y_m\}$

$$H(Y) = - \sum_{i=1}^m p_i \log(p_i) \text{ where } p_i = P(Y = y_i)$$

- Interpretation
 - Higher entropy \rightarrow higher uncertainty
 - Lower entropy \rightarrow lower uncertainty
- Conditional entropy

$$H(Y|X) = \sum_x p(x) H(Y|X = x)$$



Information Gain: An Attribute Selection Measure

- Select the attribute with the highest information gain (used in typical decision tree induction algorithm: ID3/C4.5)

- 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:

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

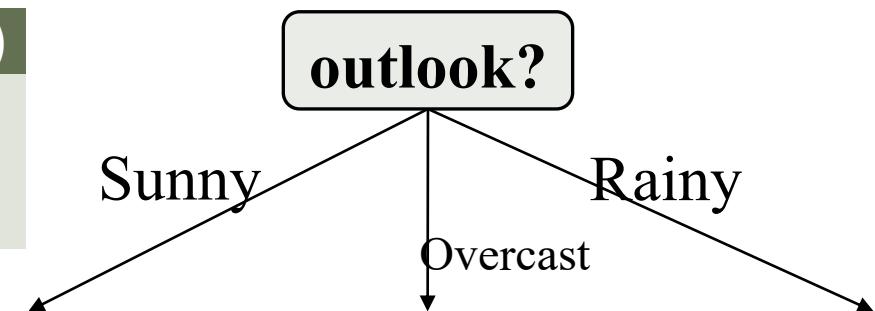
- Information gained by branching on attribute A

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

Example: Attribute Selection with Information Gain

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

outlook	yes	no	I(yes, no)
rainy	2	3	0.971
overcast	4	0	0
sunny	3	2	0.971



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

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

$\frac{5}{14} I(2,3)$ means “outlook=rainy” has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

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

Example: Attribute Selection with Information Gain

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Temp	Yes	No	I(Yes, No)
Hot	2	2	?
Mild	4	2	?
Cool	3	1	?

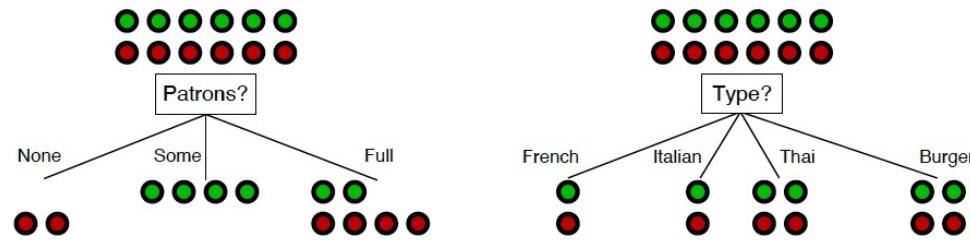
Windy	Yes	No	I(Yes, No)
True	?	?	?
False	?	?	?

Humidity	Yes	No	I(Yes, No)
Normal	6	1	?
High	3	4	?

Similarly, we can get

$$\begin{aligned}Gain(Temp) &= 0.029, \\Gain(humidity) &= 0.151, \\Gain(Windy) &= 0.048\end{aligned}$$

Example: Attribute Selection with Information Gain



- Patrons:
 - $H(X) = -6/12 \log_2 6/12 - 6/12 \log_2 6/12 = 1$
 - $H(X|None) = -0/2 \log_2 0/2 - 2/2 \log_2 2/2 = 0$
 - $H(X|Some) = -4/4 \log_2 4/4 - 0/4 \log_2 0/4 = 0$
 - $H(X|Full) = -2/6 \log_2 2/6 - 4/6 \log_2 4/6 = 0.9183$
 - $H(X) - H(X|Y) = 1 - 6/12(0.9183) = 0.5409$
- Type:
 - $H(X) = -6/12 \log_2 6/12 - 6/12 \log_2 6/12 = 1$
 - $H(X|French)=H(X|Italian)=-1/2 \log_2 1/2-1/2 \log_2 1/2=1$
 - $H(X|Thai) = H(X|Burger) = -2/4 \log_2 2/4-2/4 \log_2 2/4 = 1$
 - $H(X)-H(X|Y)=1-2/12(1)-2/12(1)-4/12(1)-4/12(1)=0$
- Choose the attribute that maximizes information gain:
Patrons?

Gain Ratio: A Refined Measure for Attribute Selection

- Information gain measure is biased towards attributes with a large number of values (e.g. ID)
- Gain ratio: Overcomes the problem (as a normalization to information gain)

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

- $\text{GainRatio}(A) = \text{Gain}(A)/\text{SplitInfo}(A)$
- The attribute with the maximum gain ratio is selected as the splitting attribute
- Gain ratio is used in a popular algorithm C4.5 (a successor of ID3) by R. Quinlan
- Example
 - $\text{SplitInfo}_{\text{temp}}(D) = -\frac{4}{14} \log_2 \frac{4}{14} - \frac{6}{14} \log_2 \frac{6}{14} - \frac{4}{14} \log_2 \frac{4}{14} = 1.557$
 - $\text{GainRatio}(\text{temp}) = 0.029/1.557 = 0.019$

Another Measure: Gini Index

- Gini index (or Gini impurity): Used in CART, and also in IBM IntelligentMiner
- CART is a binary tree
- If a data set D contains examples from n classes, gini index, $gini(D)$ is defined as
 - $$gini(D) = 1 - \sum_{j=1}^n p_j^2$$
 - p_j is the relative frequency of class j in D
- What is the range of Gini index?
 - The minimum= 0, meaning pure
 - The maximum=? What is the case that Gini index reach the maximum?

Another Measure: Gini Index

- 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

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

- 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 (*For binary tree, need to enumerate all the possible splitting points for each attribute*)

Computation of Gini Index

- Example: D has 9 tuples in play_golf= “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 temp partitions D into 10 in D_1 : {cool, mild} and 4 in D_2

$$\begin{aligned} gini_{temp \in \{cool, mild\}}(D) &= \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2) \\ &= \frac{10}{14} \left(1 - \left(\frac{7}{10} \right)^2 - \left(\frac{3}{10} \right)^2 \right) + \frac{4}{14} \left(1 - \left(\frac{2}{4} \right)^2 - \left(\frac{2}{4} \right)^2 \right) \\ &= 0.443 \end{aligned}$$

Gini_{cool,hot} is 0.458; Gini_{mild,hot} is 0.450

Thus, split on the {cool,mild} (and {hot}) since it has the lowest Gini index

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Comparing Three 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

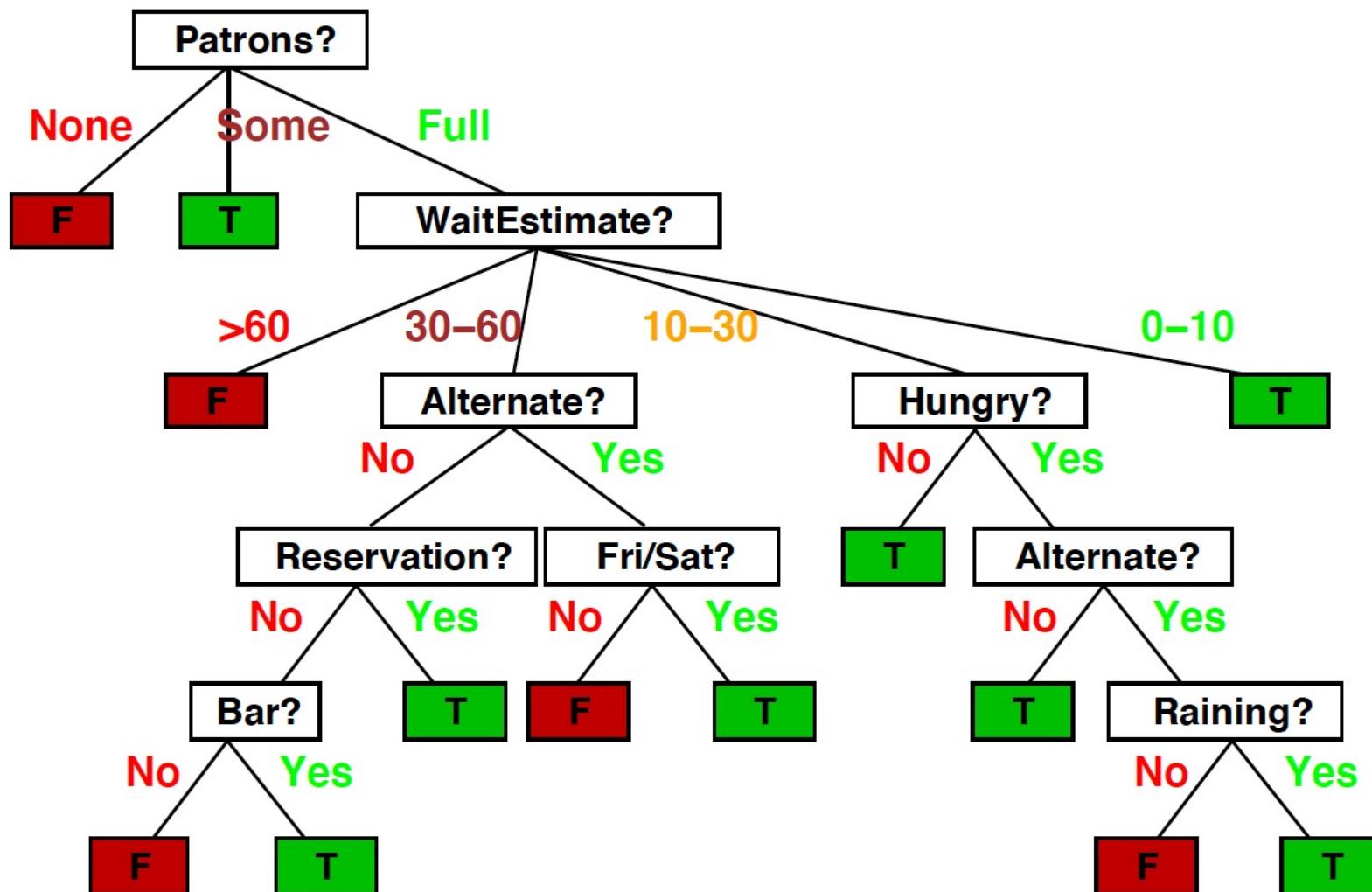
Comparing Three Attribute Selection Measures

- In reality
- Theoretical comparison between the gini index and information gain criteria
- It only matters in 2% of the cases.
- Entropy might be a little slower to compute (because of the logarithm).

Other Attribute Selection Measures

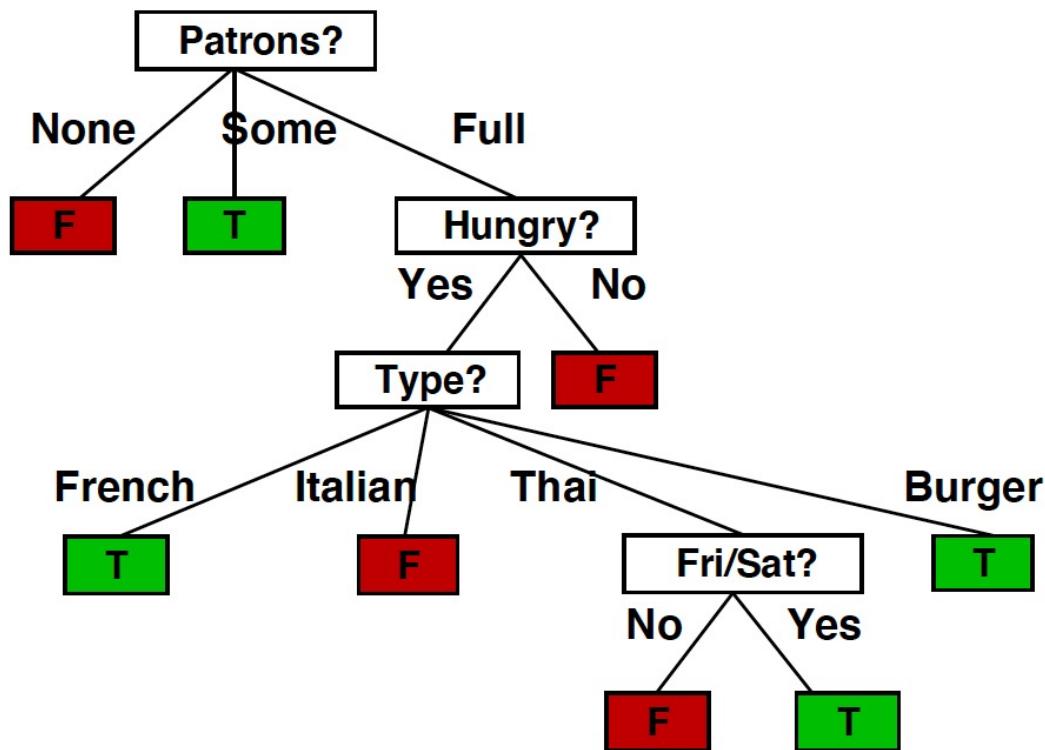
- Minimal Description Length (MDL) principle
 - Philosophy: The simplest solution is preferred
 - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- CHAID: a popular decision tree algorithm, measure based on χ^2 test for independence
- Multivariate splits (partition based on multiple variable combinations)
 - CART: finds multivariate splits based on a linear combination of attributes
- There are many other measures proposed in research and applications
 - E.g., G-statistics, C-SEP
- Which attribute selection measure is the best?
 - Most give good results, none is significantly superior than others

Example: Decision Tree



Example: Smaller Decision Tree

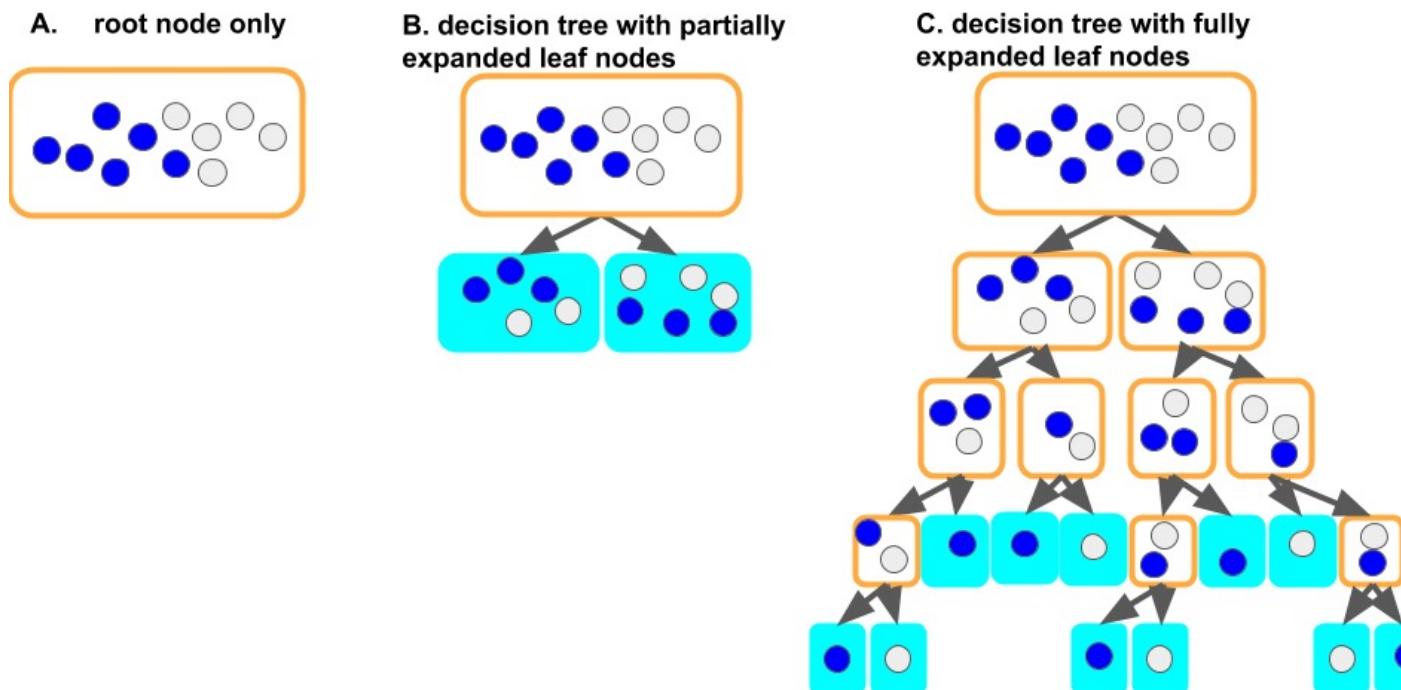
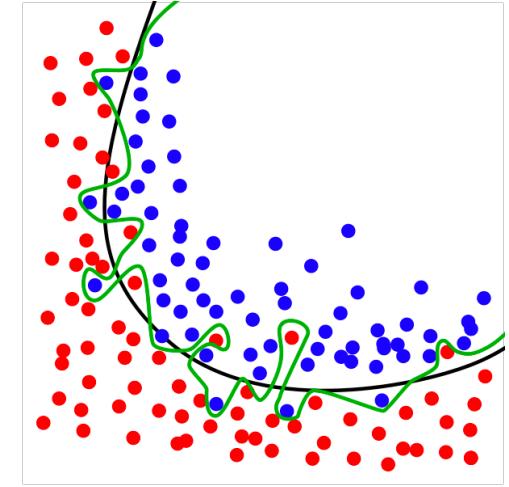
- Decision tree learned from the 12 examples:



- Substantially simpler than “true” tree
 - More complex hypothesis is not justified by a small dataset

Overfitting and Tree Pruning

- ❑ Overfitting:
 - ❑ Too many branches, some may reflect anomalies due to noise or outliers
 - ❑ Poor accuracy for **unseen** samples



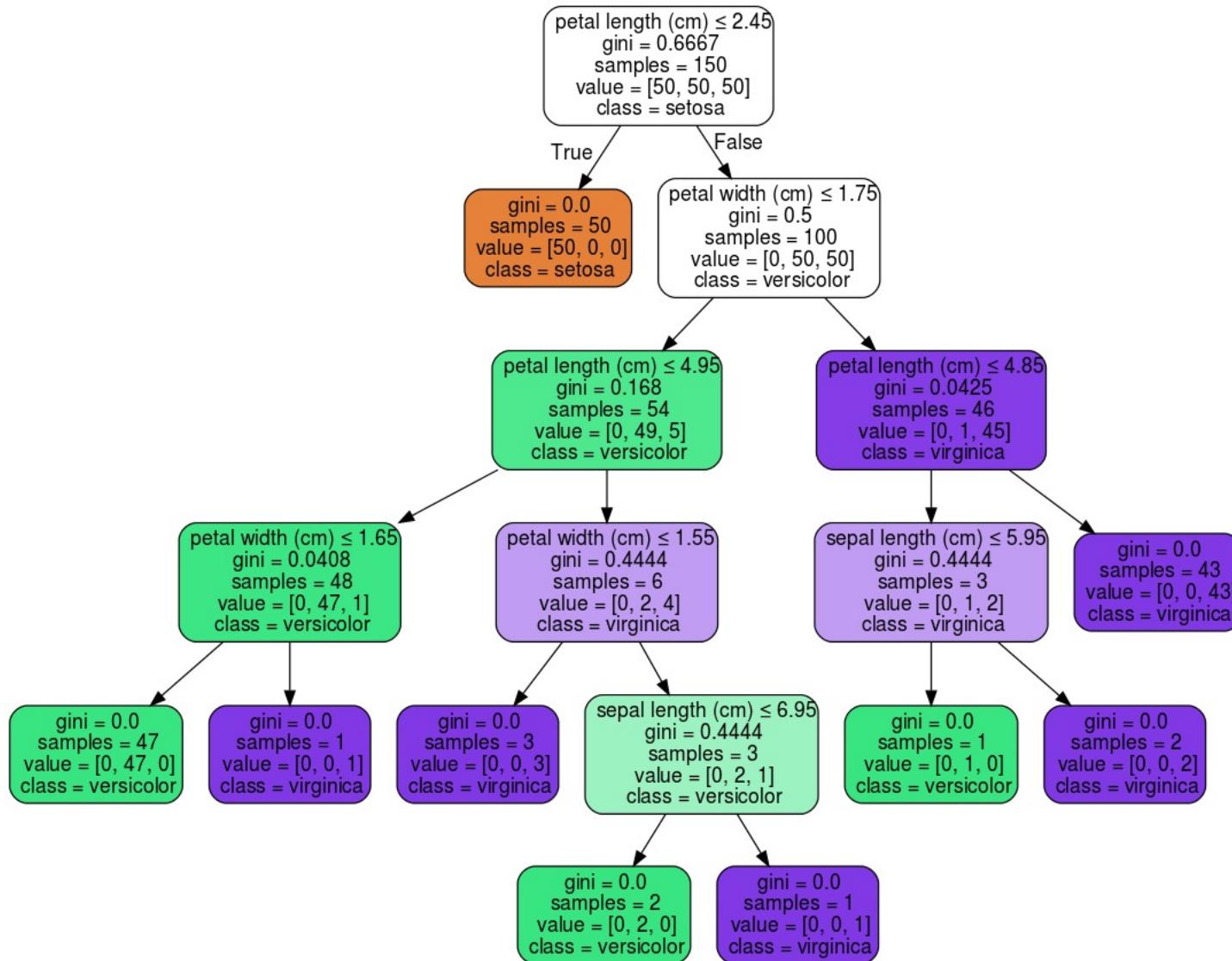
Overfitting and Tree Pruning

- Two approaches to avoid overfitting
 - Errors: use a cross-validation to compute
 - Pre-pruning (Early stop): Error does not decrease *significantly* -> stop splitting
 - Efficient but prone to under-fit (stop too early)
- Post-pruning: After the full tree is constructed, prune back to the point where the cross-validation error is minimum
 - Extra computations but mathematically rigorous
- Can be used alone, in combination, or not at all
 - For different purposes (accuracy, efficiency, interpretability)

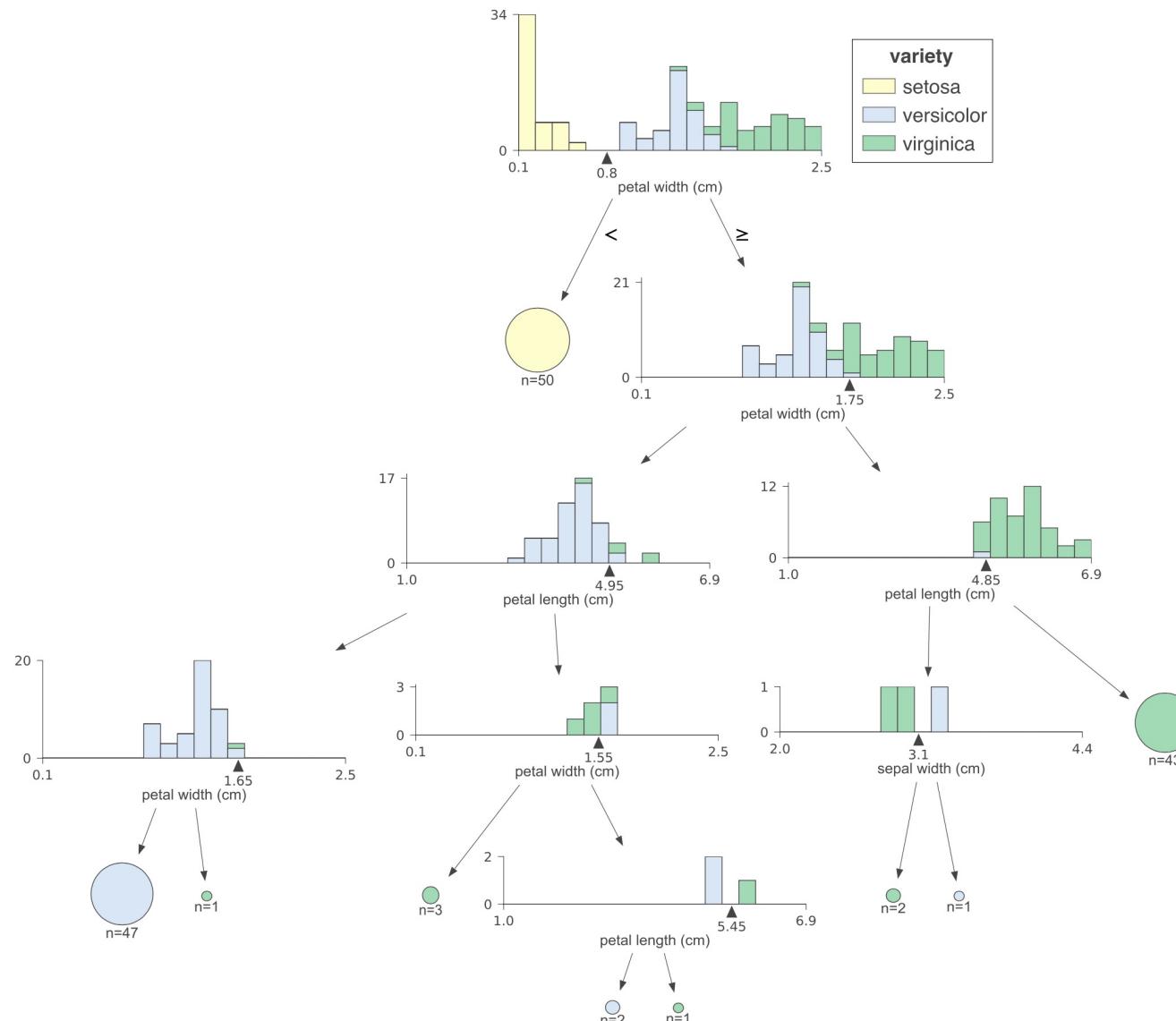
Classification in Large Databases

- Scalability: Classifying data sets with **millions** of examples and hundreds of attributes with **reasonable speed**
- Why is decision tree induction popular?
 - Relatively fast learning speed
 - Convertible to simple and easy to understand classification rules
 - Easy to be adapted to database system implementations (e.g., using SQL)
 - Comparable classification accuracy with other methods
 - Easy to ensemble, i.e., random forests, xgboost

Visualization of a Decision Tree (in scikit-learn)



Visualization of a Decision Tree



Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ Summary



Bayes' Theorem: Basics

- Total probability Theorem:

$$p(B) = \sum_i p(B|A_i)p(A_i)$$

- Bayes' Theorem:

$$p(H|X) = \frac{p(X|H)p(H)}{p(X)} \propto p(X|H) p(H)$$

posteriori probability

What we should choose

likelihood

What we just see

prior probability

What we knew previously

- X : a data sample ("evidence")

Prediction can be done based on Bayes' Theorem:

- H : X belongs to class C

Classification is to derive the maximum posteriori

Bayes' Theorem Example 1: Cancer Tests

	Cancer (1%)	No Cancer (99%)
Test Pos	80%	9.6%
Test Neg	20%	90.4%

- Only 1% people have cancer
 - How accurate is the test?
 - 80%? 99%? 1%?
-
- $P(X,H) = P(X|H)P(H)$
 - Chance of true positive is thus
 $1\% * 80\% = 0.008$

	Cancer (1%)	No Cancer (99%)
Test Pos	True Pos $1\% \times 80\% = .008$	False Pos $99\% \times 9.6\% = .09504$
Test Neg	False Neg $1\% \times 20\% = .002$	True Neg $99\% \times 90.4\% = .89496$

- According to Bayes' Theorem, $P(H|X) = P(X|H)P(H)/P(X)$, the chance of having a cancer given positive test results is
 $\text{True pos} / (\text{True pos} + \text{False pos}) = 0.008 / (0.008+0.09504) = 7.76\%$
- The Theorem lets us correct for the skewness introduced by false positives

Bayes' Theorem Example 2: Picnic Day

- The morning is cloudy ☹
 - What is the chance of rain? $P(\text{Rain} \mid \text{Cloud}) = ?$
 - 50% of all rainy days start off cloudy. $P(\text{Cloud} \mid \text{Rain}) = 50\%$
 - Cloudy mornings are common (40% of days start cloudy) $P(\text{Cloud}) = 40\%$
 - This is usually a dry month (only 3 of 30 days tend to be rainy) $P(\text{Rain}) = 10\%$
- $P(\text{Rain} \mid \text{Cloud}) = P(\text{Rain}) P(\text{Cloud} \mid \text{Rain}) / P(\text{Cloud}) = 10\% * 50\% / 40\% = 12.5\%$
- Again, the chance of rain is probably not as high as expected ☺
 - Bayes' Theorem allows us to tell back and forth between posterior and likelihood (e.g., $P(\text{Rain} \mid \text{Cloud})$ and $P(\text{Cloud} \mid \text{Rain})$), tests and reality, which is the most important trick in Bayesian Inference

Bayes' Theorem: Multiple Steps

- Usually more than one hypothesis: H_1, H_2, \dots, H_m

$$p(H_i|X) = \frac{p(X|H_i)p(H_i)}{p(X)}$$

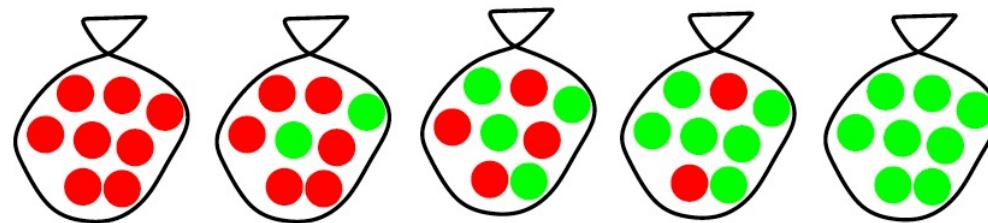
- Observations (evidence) over multiple steps: X_1, X_2, \dots, X_T

$$p(H_i|X_1) = \frac{p(X_1|H_i)p(H_i)}{p(X_1)}$$

Example: Bayes' Theorem, Multiple Steps

Suppose there are five kinds of bags of candies:

- 10% are h_1 : 100% cherry candies
- 20% are h_2 : 75% cherry candies + 25% lime candies
- 40% are h_3 : 50% cherry candies + 50% lime candies
- 20% are h_4 : 25% cherry candies + 75% lime candies
- 10% are h_5 : 100% lime candies

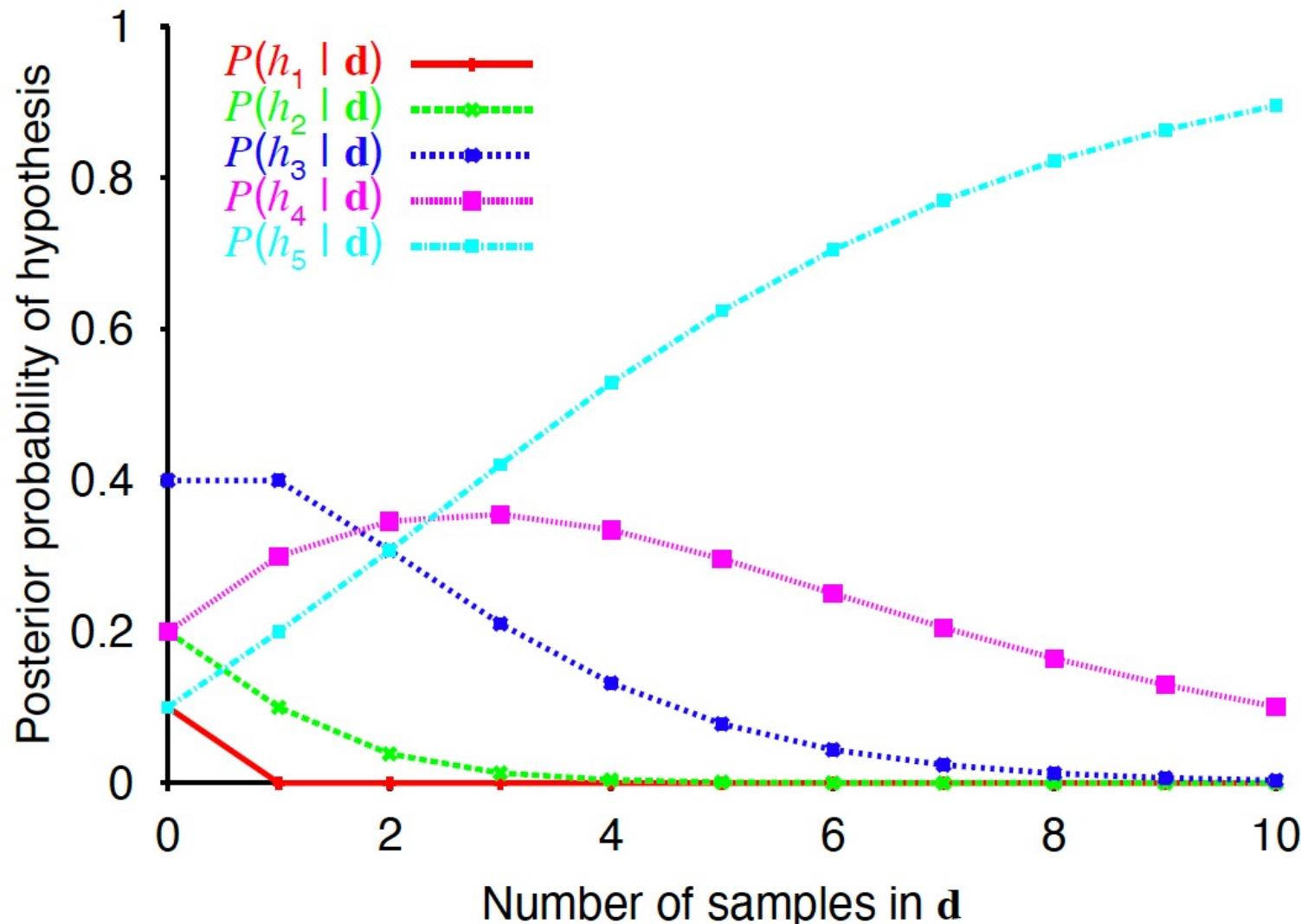


Then we observe candies drawn from some bag:

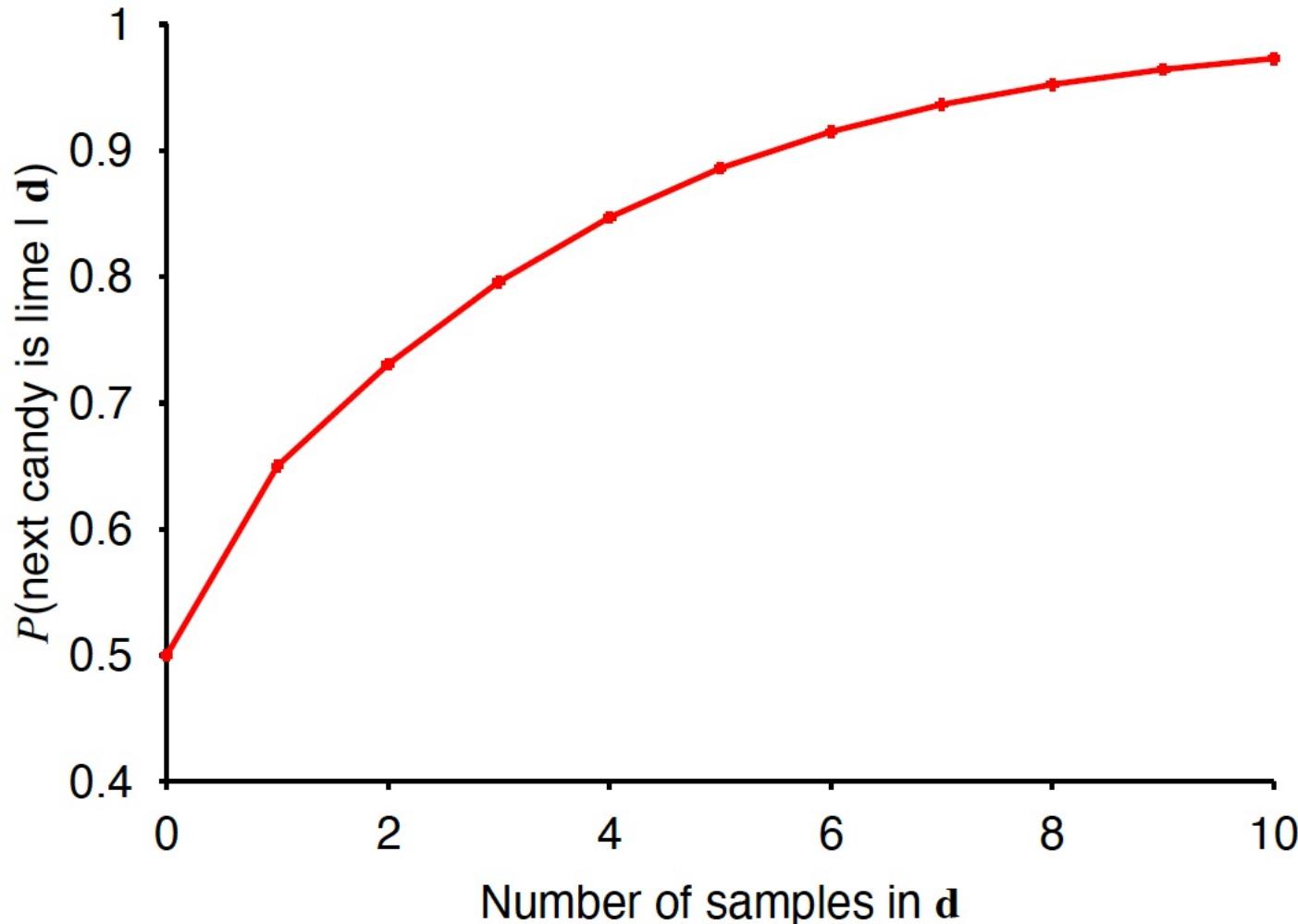


What kind of bag is it? What flavor will the next candy be?

Example: Bayes' Theorem, Multiple Steps



Example: Bayes' Theorem, Multiple Steps



Naïve Bayes Classifier: Making a Naïve Assumption

- Based on the Bayes' Theorem, we can derive a Bayes Classifier to compute the posterior probability of classifying an object X to a class C
- $P(C|X) \propto P(X|C)P(C) = P(x_1|C)P(x_2|x_1,C)\dots P(x_n|x_1,\dots,C)P(C)$
- A naïve assumption to simplify the complex dependencies: *features are conditionally independent!*
 - $P(C|X) \propto P(X|C)P(C) \approx P(x_1|C)P(x_2|C)\dots P(x_n|C)P(C)$
- Super efficient: each feature only conditions on the class (boils down to sample counting)
- Achieves surprisingly comparable performance

Naïve Bayes Classifier: Categorical vs. Continuous Valued Features

- If feature x_k is categorical, $p(x_k = v_k | C_i)$ is the # of tuples in C_i with $x_k = v_k$, divided by $|C_{i,D}|$ (# of tuples of C_i in D)

$$p(X|C_i) = \prod_k p(x_k|C_i) = p(x_1|C_i) \cdot p(x_2|C_i) \cdots p(x_n|C_i)$$

- If feature x_k is continuous-valued, $p(x_k = v_k | C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

$$p(x_k = v_k | C_i) = N(x_k | \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{(x-v_{C_i})^2}{2\sigma^2}}$$

Naïve Bayes Classifier Example 1: Training Dataset

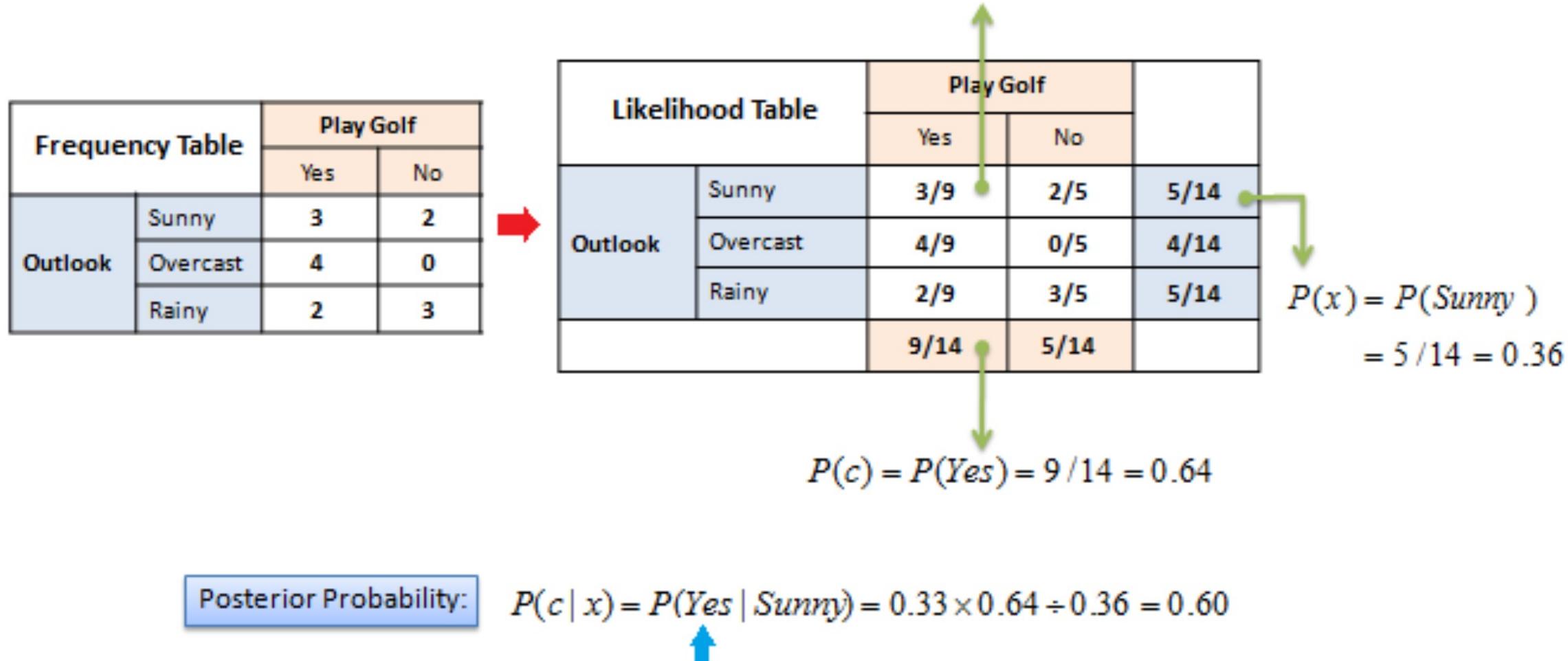
Class:

play golf= 'yes'

play golf = 'no'

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Naïve Bayes Classifier Example $P(Yes | Sunny)$



Posterior Probability:

$$P(c | x) = P(Yes | Sunny) = 0.33 \times 0.64 \div 0.36 = 0.60$$



Naïve Bayes Classifier Example: P(No | Sunny)

Frequency Table		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3



		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
		9	5	14

$$P(x | c) = P(\text{Sunny} | \text{No}) = 2 / 5 = 0.4$$

Play Golf

Yes No

Outlook
Sunny
Overcast
Rainy

3 2

5

4 0

4

2 3

5

9 5

14

$$\begin{aligned}P(x) &= P(\text{Sunny}) \\&= 5 / 14 = 0.36\end{aligned}$$

$$P(c) = P(\text{No}) = 5 / 14 = 0.36$$

Posterior Probability:

$$P(c | x) = P(\text{No} | \text{Sunny}) = 0.40 \times 0.36 \div 0.36 = 0.40$$



Naïve Bayes Classifier Example: Likelihood Tables

Frequency Table

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

Likelihood Table

		Play Golf	
		Yes	No
Outlook	Sunny	3/9	2/5
	Overcast	4/9	0/5
	Rainy	2/9	3/5

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1

		Play Golf	
		Yes	No
Humidity	High	3/9	4/5
	Normal	6/9	1/5

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1

		Play Golf	
		Yes	No
Temp.	Hot	2/9	2/5
	Mild	4/9	2/5
	Cool	3/9	1/5

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3

		Play Golf	
		Yes	No
Windy	False	6/9	2/5
	True	3/9	3/5

Naïve Bayes Classifier Example: Likelihood Tables

Outlook	Temp	Humidity	Windy	Play
Rainy	Cool	High	True	?

$$P(Yes | X) = P(Rainy | Yes) \times P(Cool | Yes) \times P(High | Yes) \times P(True | Yes) \times P(Yes)$$

$$P(Yes | X) = 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.00529 \quad 0.2 = \frac{0.00529}{0.02057 + 0.00529}$$

$$P(No | X) = P(Rainy | No) \times P(Cool | No) \times P(High | No) \times P(True | No) \times P(No)$$

$$P(No | X) = 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.02057 \quad 0.8 = \frac{0.02057}{0.02057 + 0.00529}$$

Naïve Bayes Classifier: Example 2

Class:

C1:`buys_computer = 'yes'`

C2:`buys_computer = 'no'`

Data to be classified:

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 Bayes Classifier: Example 2

Compute $P(X|C_i)$ for each class:

$$P(\text{age} = “\leq 30” | \text{buys_computer} = “\text{yes}\text{”}) = 2/9 = 0.222$$

$$P(\text{age} = “\leq 30” | \text{buys_computer} = “\text{no}\text{”}) = 3/5 = 0.6$$

$$P(\text{income} = “\text{medium}” | \text{buys_computer} = “\text{yes}\text{”}) = 4/9 = 0.444$$

$$P(\text{income} = “\text{medium}” | \text{buys_computer} = “\text{no}\text{”}) = 2/5 = 0.4$$

$$P(\text{student} = “\text{yes}” | \text{buys_computer} = “\text{yes}\text{”}) = 6/9 = 0.667$$

$$P(\text{student} = “\text{yes}” | \text{buys_computer} = “\text{no}\text{”}) = 1/5 = 0.2$$

$$P(\text{credit_rating} = “\text{fair}” | \text{buys_computer} = “\text{yes}\text{”}) = 6/9 = 0.667$$

$$P(\text{credit_rating} = “\text{fair}” | \text{buys_computer} = “\text{no}\text{”}) = 2/5 = 0.4$$

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 Bayes Classifier: Example 2

$X = (\text{age} \leq 30, \text{income} = \text{medium}, \text{student} = \text{yes}, \text{credit_rating} = \text{fair})$

$P(X|C_i) :$

$P(X|\text{buys_computer} = \text{"yes"}) =$

$P(\text{age} = \text{"}<=30\text{"} | \text{buys_computer} = \text{"yes"})$

$P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"yes"})$

$P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"yes"})$

$P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"yes"})$

$$= 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$$

$P(X|\text{buys_computer} = \text{"no"}) =$

$P(\text{age} = \text{"}<= 30\text{"} | \text{buys_computer} = \text{"no"})$

$P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"no"})$

$P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"no"})$

$P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"no"})$

$$= 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$$

$P(X C_i)$	$C1 = \text{yes}$	$C2 = \text{no}$
age ≤ 30	0.222	0.6
Inc. = med.	0.444	0.4
Stu. = yes	0.667	0.2
Credit = fair	0.667	0.4

Conditional probability

$P(X|C_i)*P(C_i) :$

$P(X|\text{buys_computer} = \text{"yes"}) * P(\text{buys_computer} = \text{"yes"}) = 0.028$

$P(X|\text{buys_computer} = \text{"no"}) * P(\text{buys_computer} = \text{"no"}) = 0.007$

Therefore, X is classified to class ("buys_computer = yes")

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional probability be **non-zero**
 - Otherwise, the predicted probability will be zero

$$p(X|C_i) = \prod_k p(x_k|C_i) = p(x_1|C_i) \cdot p(x_2|C_i) \cdots p(x_n|C_i)$$

- Example. 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*

$$\text{Prob}(\text{income} = \text{low}) = 1/(1000 + 3)$$

$$\text{Prob}(\text{income} = \text{medium}) = (990 + 1)/(1000 + 3)$$

$$\text{Prob}(\text{income} = \text{high}) = (10 + 1)/(1000 + 3)$$

$$\begin{aligned}\hat{P}(w_i | c) &= \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)} \\ &= \frac{\text{count}(w_i, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V|}\end{aligned}$$

- The “corrected” probability estimates are close to their “uncorrected” counterparts

Naïve Bayes Classifier: Strength vs. Weakness

- Strength
 - Performance: A *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

Naïve Bayes Classifier: Strength vs. Weakness

- Weakness
 - Assumption: attributes conditional independence, therefore loss of accuracy
 - E.g., Patient's Profile: (age, family history),
 - Patient's Symptoms: (fever, cough),
 - Patient's Disease: (lung cancer, diabetes).
 - Dependencies among these cannot be modeled by Naïve Bayes Classifier
 - How to deal with these dependencies?
Use Bayesian Belief Networks (to be covered in the next chapter)

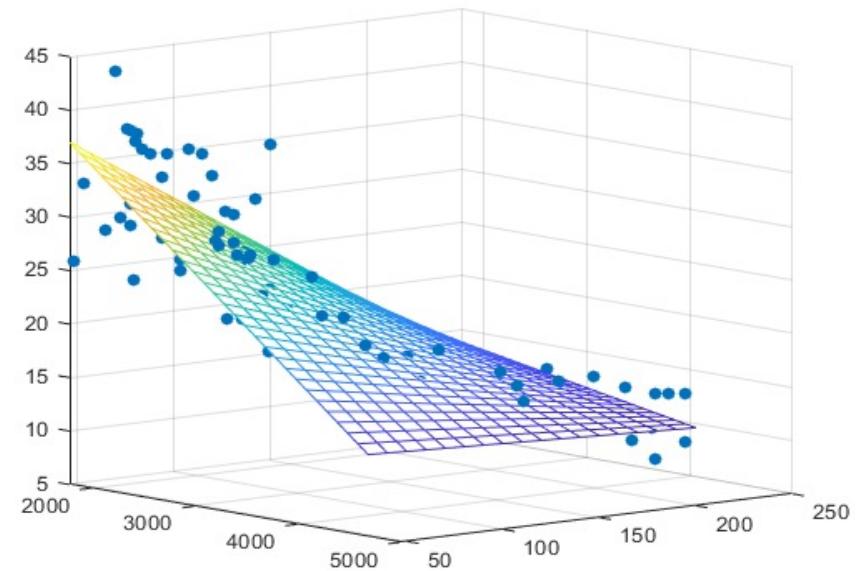
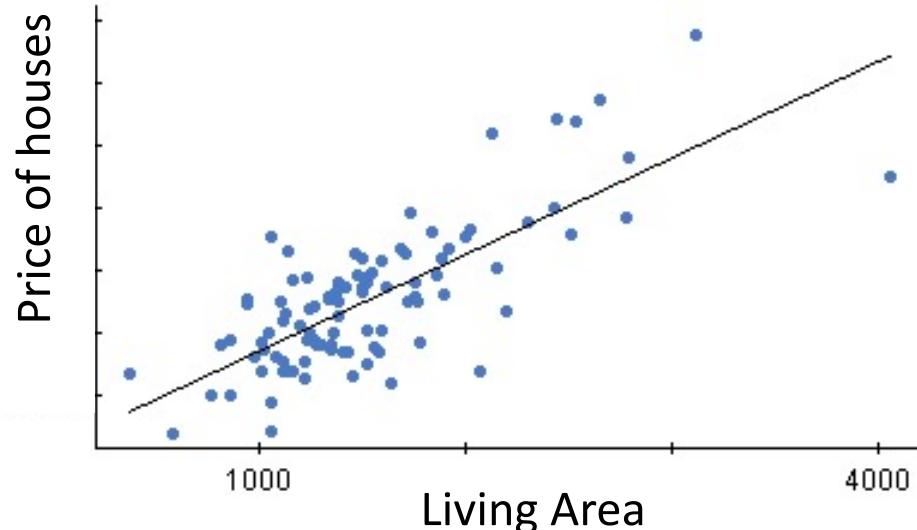
Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ Summary



Linear Regression Problem: Example

- Mapping from independent attributes to **continuous value**: $x \Rightarrow y$
- {living area} \Rightarrow Price of the house
- {college; major; GPA} \Rightarrow Future Income



Linear Regression Problem: Model

- Linear regression
 - Data: n independent objects
 - Observed Value: $y_i, i = 1, 2, 3, \dots, n$
 - p-dimensional attributes: $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})^T, i = 1, 2, 3 \dots, n$
- Model:
 - Weight vector: $w = (w_1, w_2, \dots, w_p)$
 - $y_i = w^T x_i + b$
 - The weight vector w and bias b is the model parameter learnt by data

Linear Regression Model: Solution

- Least Square Method
- Cost / Loss Function: $L(w, b) = \sum_{i=1}^m (y_i - wx_i - b)^2$
- Optimization Goal: $\operatorname{argmin}_{(w,b)} L(w, b) = \sum_{i=1}^m (y_i - wx_i - b)^2$
- Closed-form solution:
 - $w = \frac{\sum_{i=1}^m x_i(y_i - \bar{y})}{\sum_{i=1}^m x_i^2 - 1/m(\sum_{i=1}^m x_i)^2}$ $b = \frac{1}{m} \sum_{i=1}^m (y_i - wx_i)$

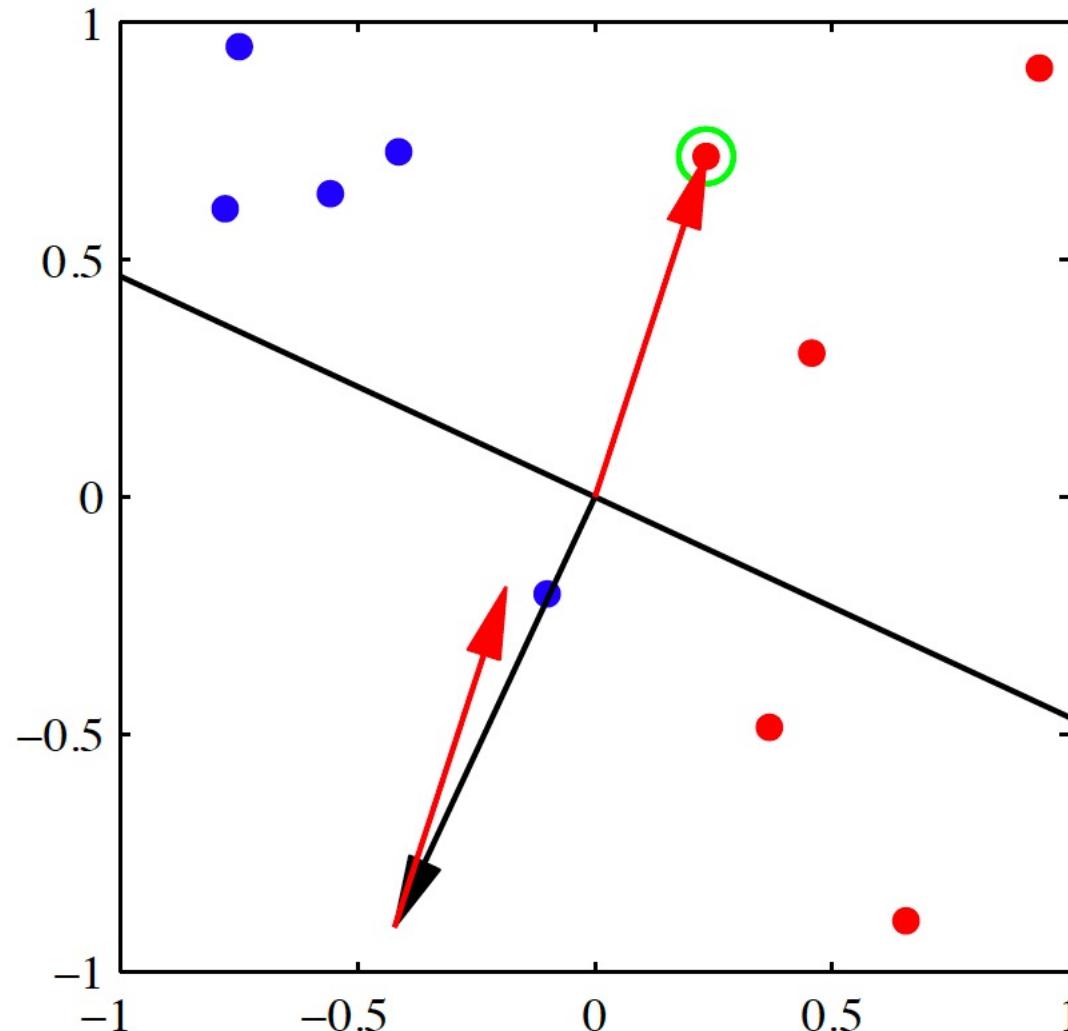
Linear Classification: Perceptron

- Linear classifiers with weight vector \mathbf{w}
- Prediction on \mathbf{x}_i is incorrect if $y_i \mathbf{w}^T \mathbf{x}_i < 0$.
- Let $\mathcal{M}(\mathbf{w})$ be the set of points on which prediction is incorrect
- The objective function to be minimized

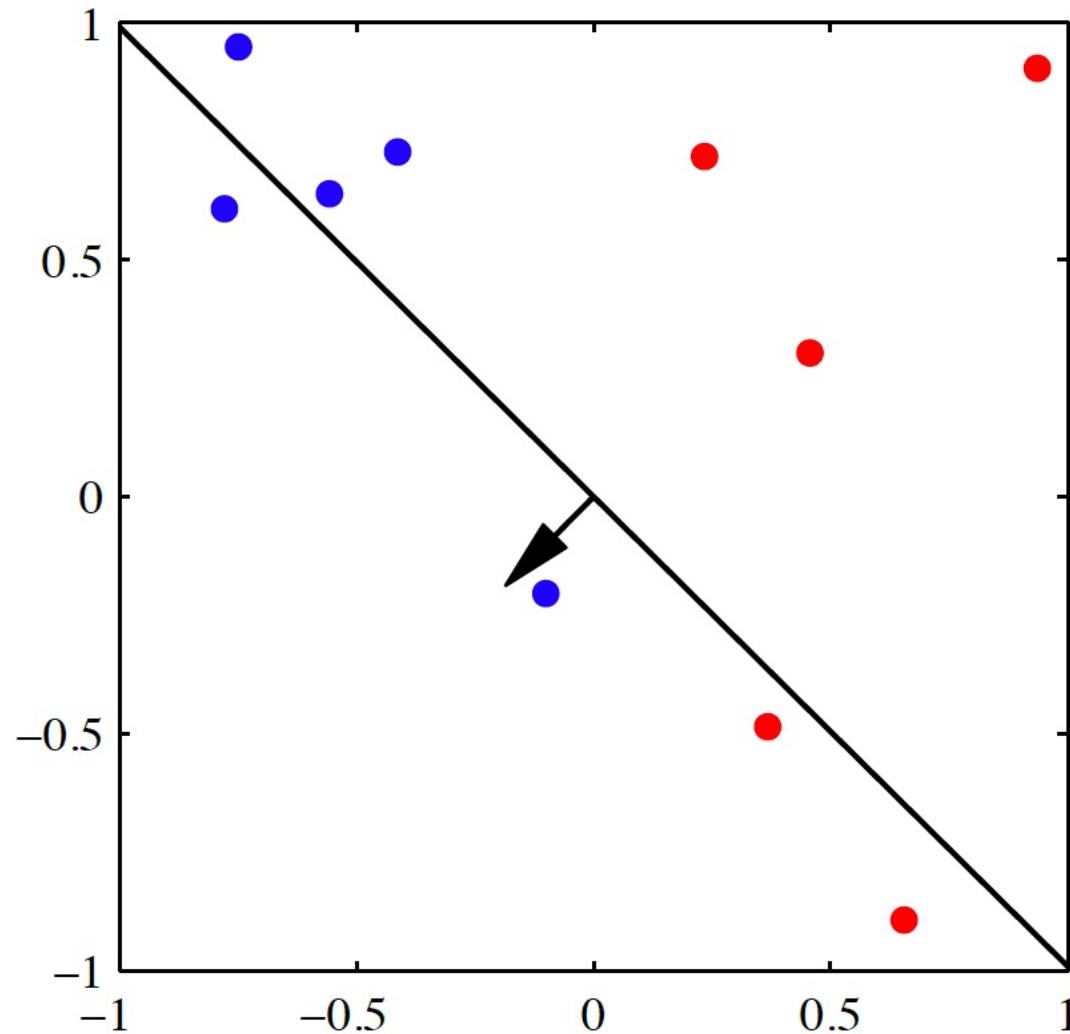
$$E(\mathbf{w}) = - \sum_{i \in \mathcal{M}(\mathbf{w})} y_i \mathbf{w}^T \mathbf{x}_i$$

- Algorithm goes through all the points sequentially
 - If prediction is correct, do not change anything
 - If prediction is wrong, and $y_i = +1$ then $\mathbf{w}^{(new)} = \mathbf{w}^{(old)} + \mathbf{x}_i$
 - If prediction is wrong, and $y_i = -1$ then $\mathbf{w}^{(new)} = \mathbf{w}^{(old)} - \mathbf{x}_i$
 - For any point $i \in \mathcal{M}(\mathbf{w})$, gradient $\nabla E_i(\mathbf{w}) = -y_i \mathbf{x}_i$
 - The gradient based update
- $$\mathbf{w}^{(new)} = \mathbf{w}^{(old)} - \eta \nabla E_i(\mathbf{w}) = \mathbf{w}^{(old)} + \eta y_i \mathbf{x}_i$$
- The learning rate parameter η can be set to 1
 - No update corresponding to the correctly predicted points

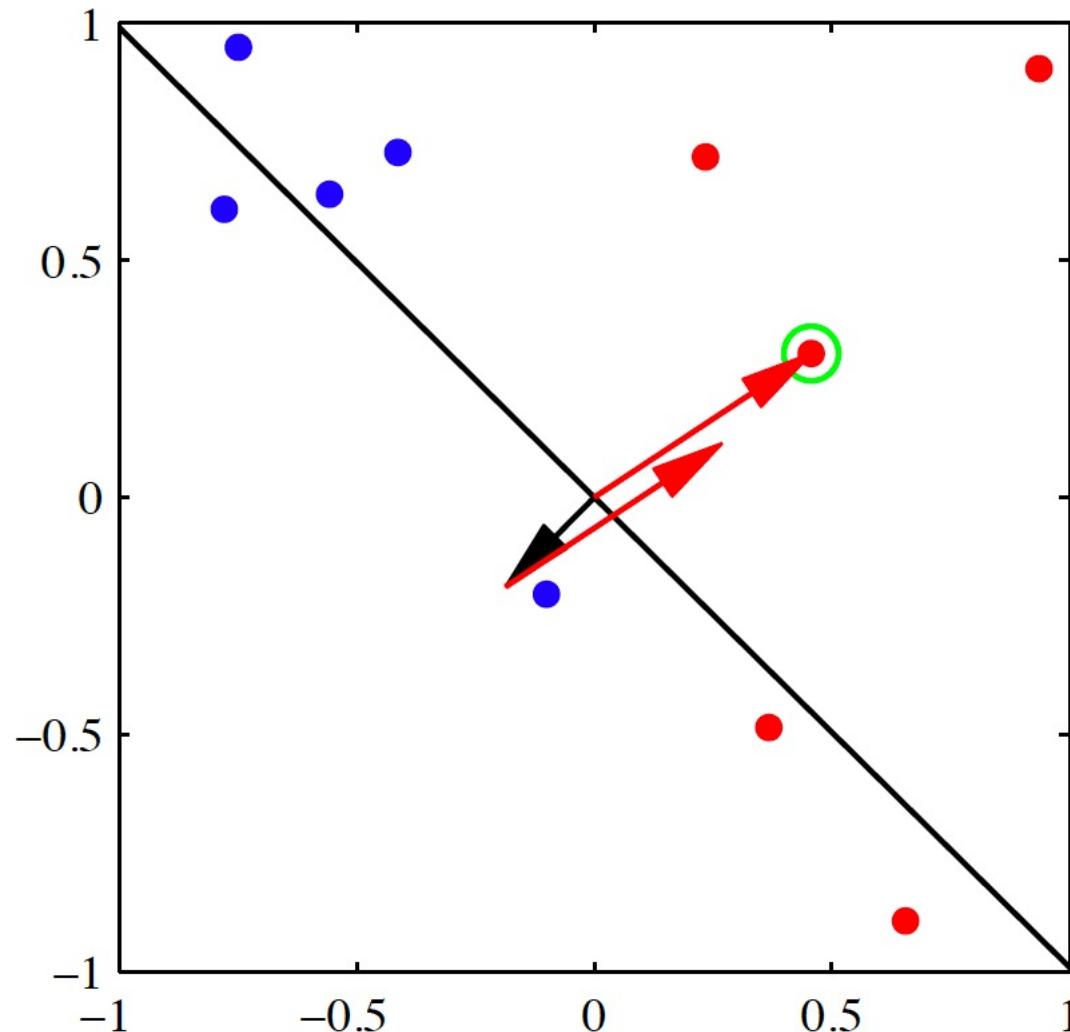
Linear Classification: Example



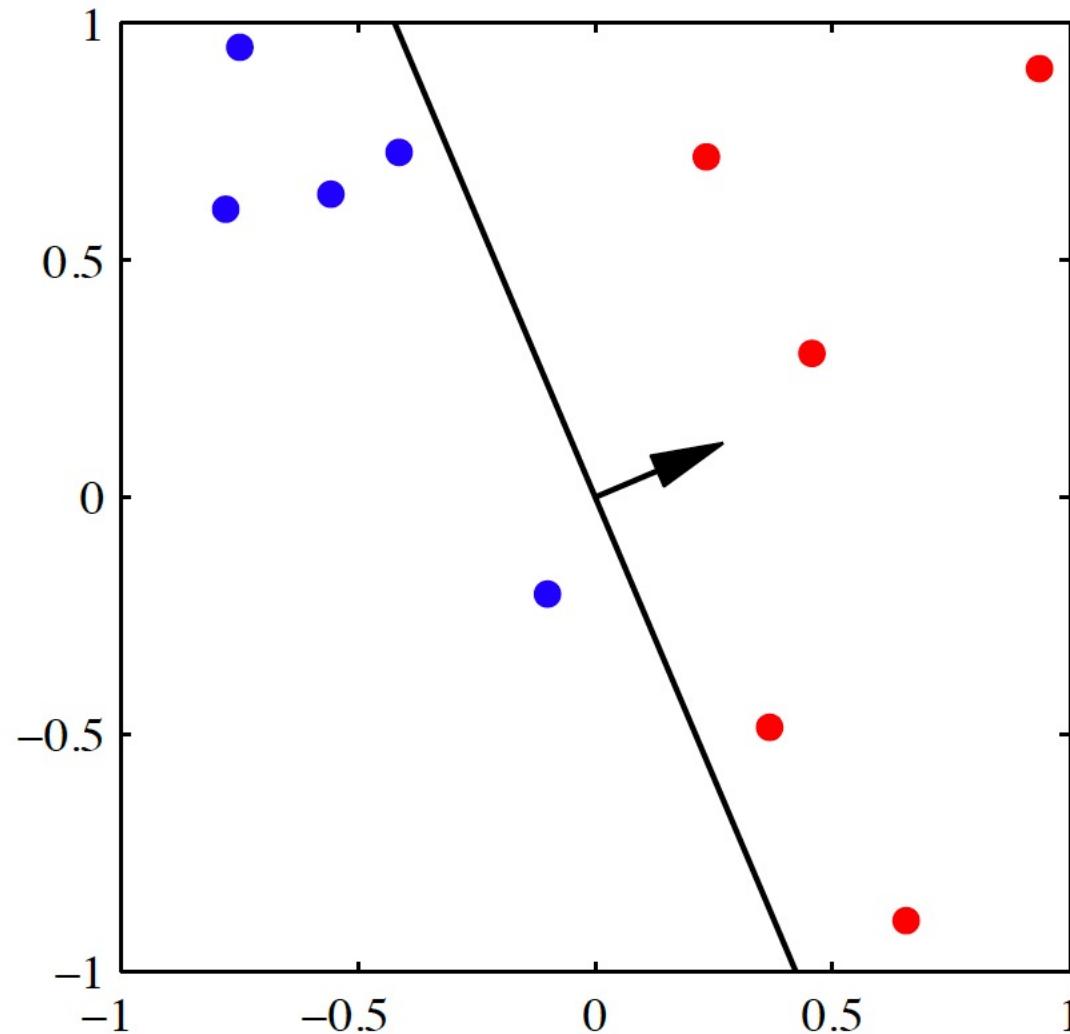
Linear Classification: Example



Linear Classification: Example

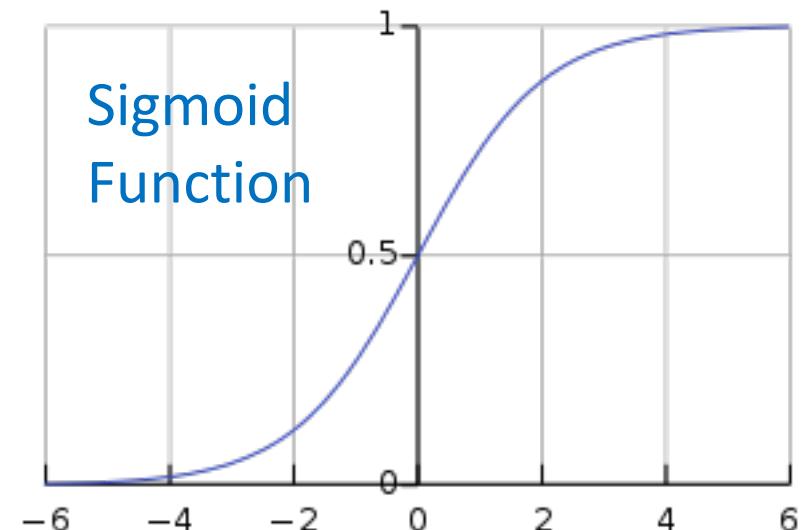


Linear Classification: Example



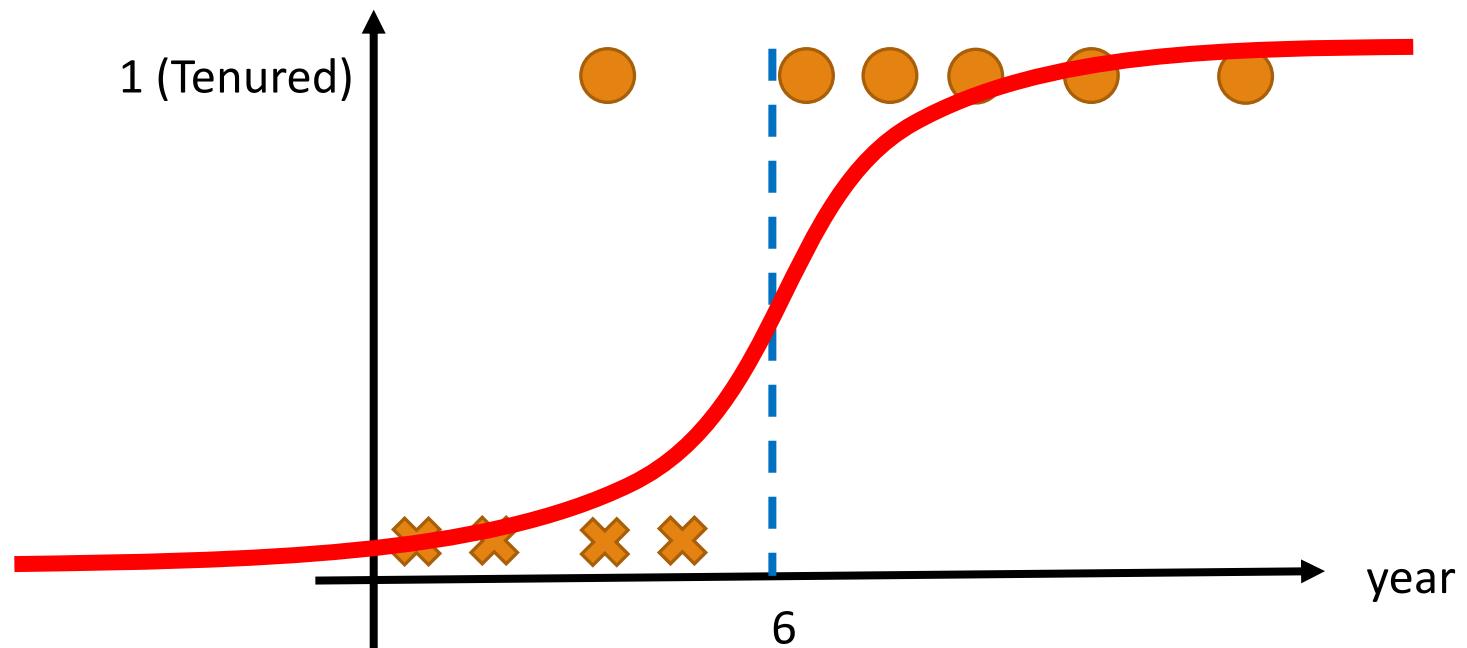
Logistic Regression: General Ideas

- How to solve “classification” problems by regression?
- Key idea of Logistic Regression
 - We need to transform the real value Y into a probability value $\in [0,1]$
- Sigmoid function (differentiable function) :
 - $\sigma(z) = \frac{1}{1+e^{-z}} = \frac{e^z}{e^z+1}$
 - Projects $(-\infty, +\infty)$ to $[0, 1]$
 - Not only LR uses this function, but also neural network, deep learning
- The projected value change sharply around zero point
- Notice that $\ln \frac{y}{1-y} = w^T x + b$



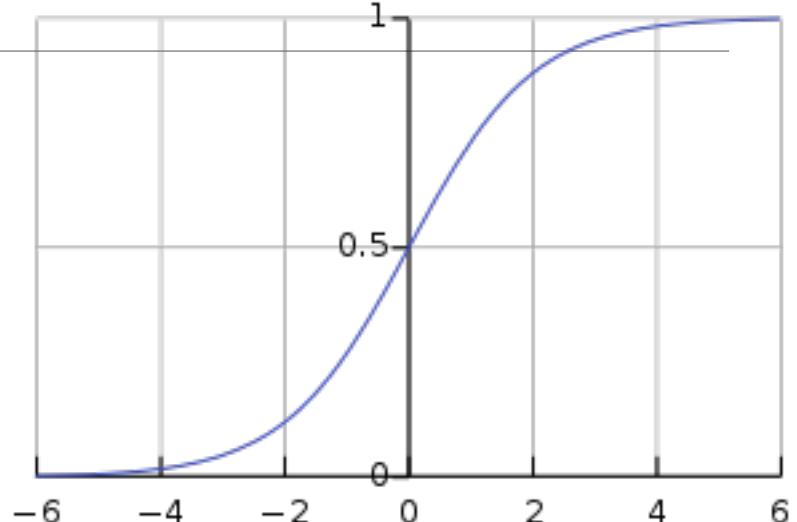
Logistic Regression: An Example

- ❑ Suppose we only consider the year as feature
- ❑ Data points are converted by sigmoid function (“activation” function)



Logistic Regression: Model

- ❑ The prediction function to learn
- ❑ Probability that $Y=1$:
 - ❑ $p(Y = 1 | X = x; \mathbf{w}) = \text{Sigmoid}(w_0 + \sum_{i=1}^n w_i \cdot x_i)$
 - ❑ $\mathbf{w} = (w_0, w_1, w_2, \dots, w_n)$ are the parameters
- ❑ A single data object with attributes x_i and class label y_i
 - ❑ Suppose the probability of $p(\hat{y}_i = 1 | x_i, \mathbf{w}) = p_i$, then $p(\hat{y}_i = 0 | x_i, \mathbf{w}) = 1 - p_i$
 - ❑ $p(\hat{y}_i = y_i) = p_i^{y_i} (1 - p_i)^{1-y_i}$
- ❑ Maximum Likelihood Estimation
 - ❑ $L = \prod_i p_i^{y_i} (1 - p_i)^{1-y_i} = \prod_i \left(\frac{\exp(w^T x_i)}{1 + \exp(w^T x_i)} \right)^{y_i} \left(\frac{1}{1 + \exp(w^T x_i)} \right)^{1-y_i}$



Logistic Regression: Optimization

- Maximum Likelihood Estimation

- $L = \prod_i p_i^{y_i} (1 - p_i)^{1-y_i} = \prod_i \left(\frac{\exp(w^T x_i)}{1 + \exp(w^T x_i)} \right)^{y_i} \left(\frac{1}{1 + \exp(w^T x_i)} \right)^{1-y_i}$

- Log likelihood:

$$\begin{aligned} l(w) &= \sum_{i=1}^N y_i \log p(Y = 1 | X = x_i; w) + (1 - y_i) \log(1 - p(Y = 1 | X = x_i; w)) \\ &= \sum_{i=1}^N y_i x_i^T w - \log(1 + \exp(w^T x_i)) \end{aligned}$$

- There's no closed form solution
- Gradient Descent

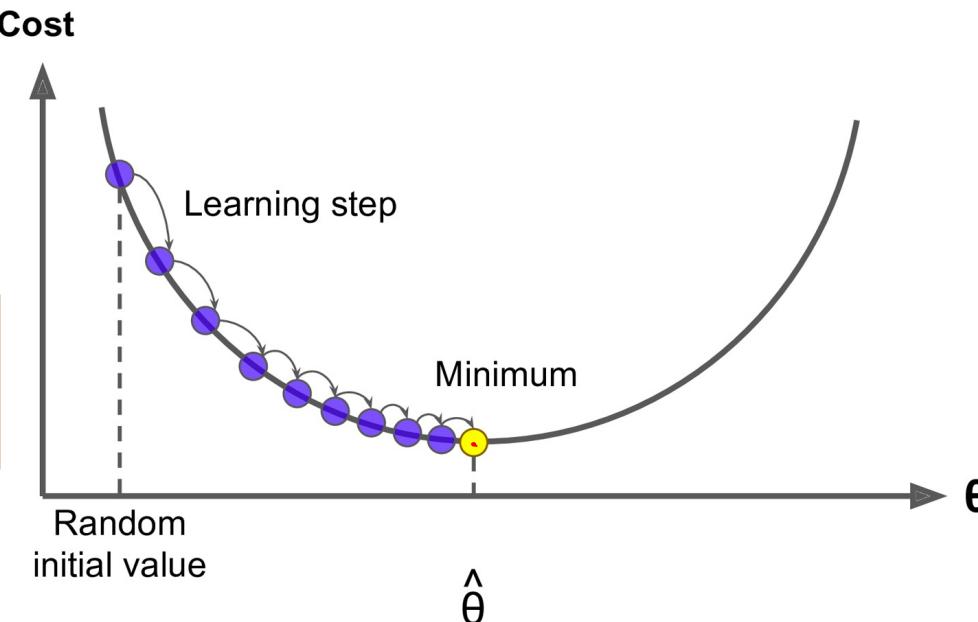
Gradient Descent

- Gradient Descent is an iterative optimization algorithm for finding the minimum of a function (e.g., the negative log likelihood)
- For a function $F(x)$ at a point a , $F(x)$ decreases fastest if we go in the direction of the negative gradient of a

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma \nabla F(\mathbf{a}_n)$$

Step size

When the gradient is zero, we arrive at the local minimum



Convergence Problem

x	y
-5	0
-4	0
-3	0
-2	0
-1	0
1	1
2	1
3	1
4	1
5	1

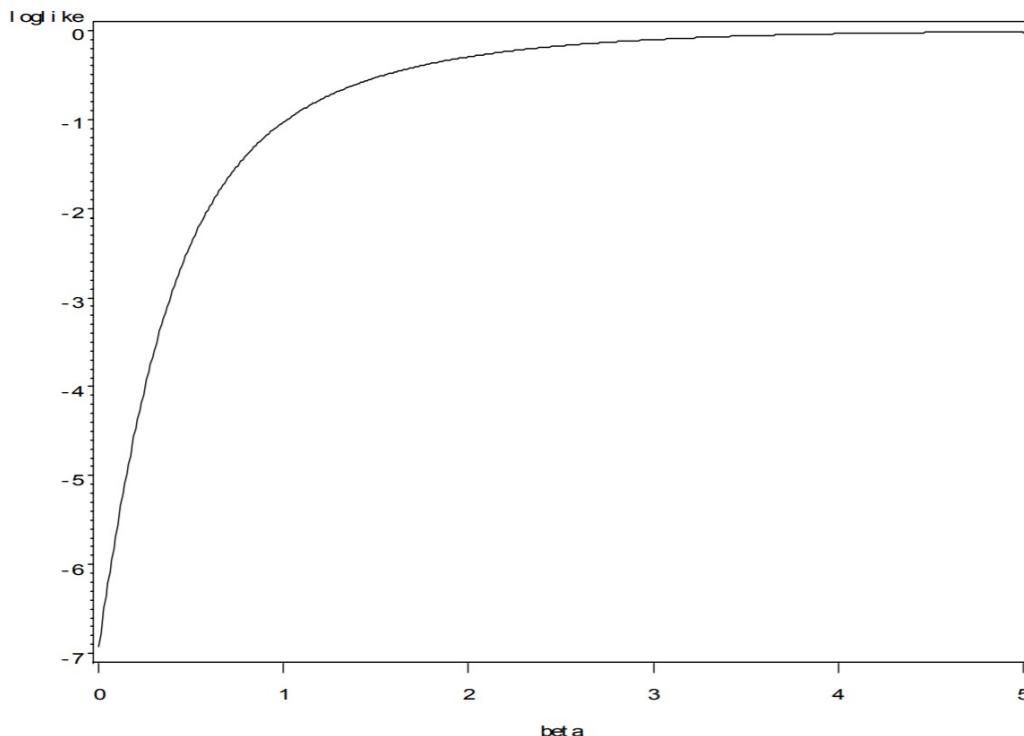


Figure 1. Log-likelihood as a function of the slope under complete separation

x	y
1	0
1	5
0	10

x	y
1	0
0	15
10	

x	y
-5	0
-4	0
-3	0
-2	0
-1	0
0	0
0	1
1	1
2	1
3	1
4	1
5	1

Linear Regression Vs. Logistic Regression

- Linear Regression
 - Y : Continuous Value $\in [-\infty, +\infty]$
 - $Y = W^T X + b$
 - Often used in value prediction problems

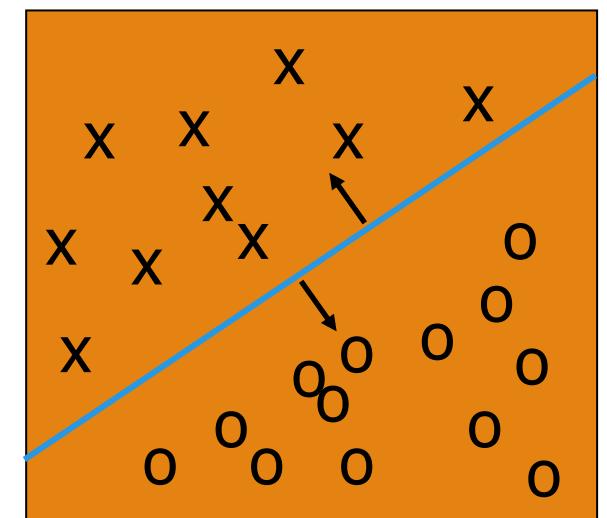
- Logistic Regression
 - Y : A discrete value from m classes
 - $P(Y = C_i) \in [0,1]$ and $\sum_{i=1}^m P(Y = C_i) = 1$
 - Often used in classification problems

Comments on Logistic Regression

- Pros
 - Can handle multiple types of features
 - Fast and easy
 - Generally speaking, more robust and better performance than tree
 - Interpretable: both weights and predicted value
 - Predicted value: probability
 - Weights: effect of the feature. Unit change of log odds
- Cons
 - Linear model: if the decision boundary is not linear, then LR is not good

Linear Classifier: General Ideas

- Binary Classification
- $f(x)$ is a linear function based on the example's attribute values
 - The prediction is based on the value of $f(x)$
 - Data above the blue line belongs to class 'x' (i.e., $f(x) > 0$)
 - Data below blue line belongs to class 'o' (i.e., $f(x) < 0$)
- Classical Linear Classifiers
 - Logistic Regression
 - Linear Discriminant Analysis (LDA)
 - Perceptron
 - SVM

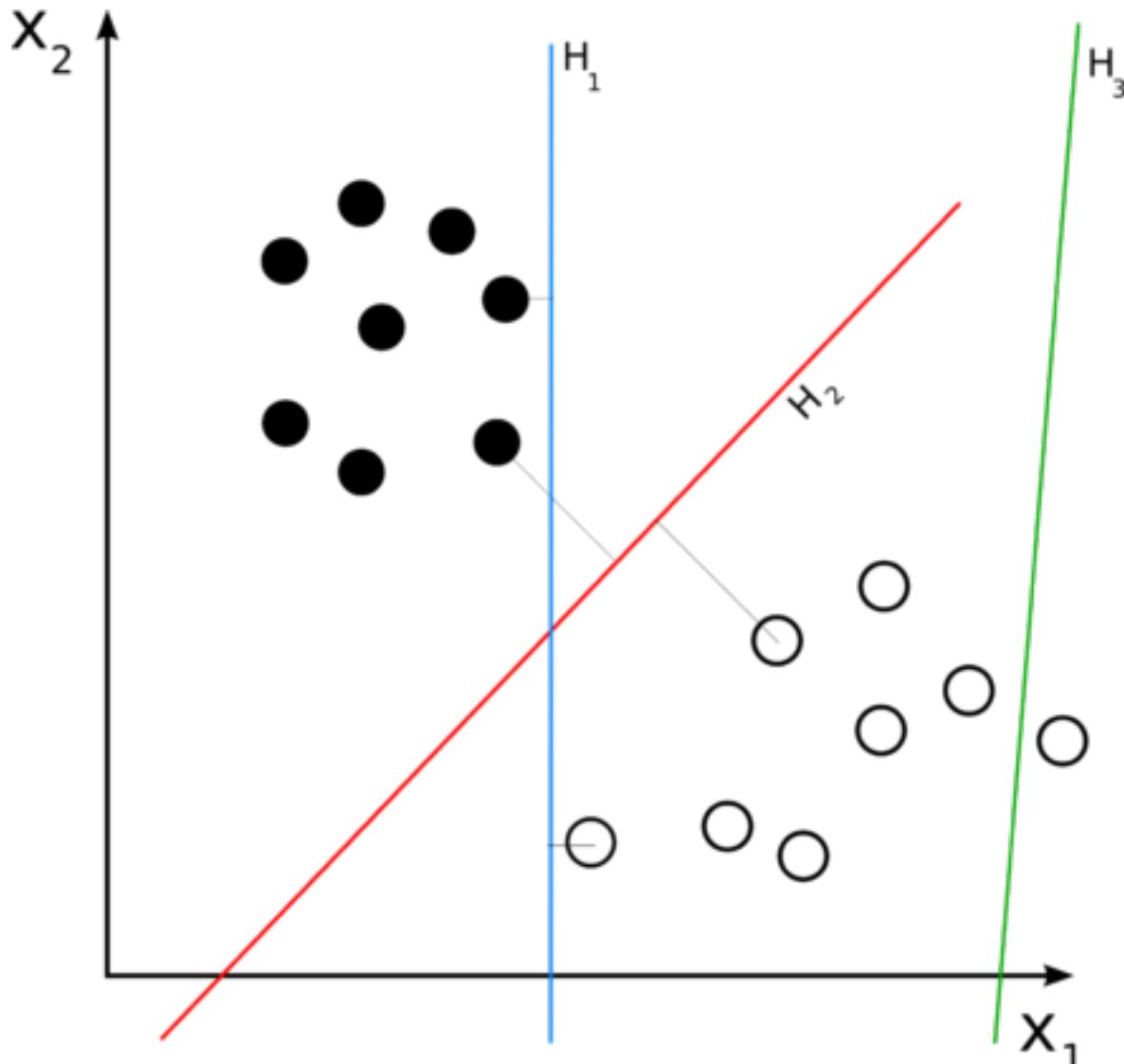


Linear Classifier: An Example

- A toy rule to determine whether a faculty member has tenure
 - Year ≥ 6 or Title = “Professor” \Leftrightarrow Tenure
- How to express the rule as a linear classifier?
- Features
 - $x_1 (x_1 \geq 0)$ is an integer denoting the year
 - x_2 is a Boolean denoting whether the title is “Professor”
- A feasible linear classifier: $f(x) = (x_1 - 5) + 6 \cdot x_2$
 - When x_2 is True, because $x_1 \geq 0$, $f(x)$ is always greater than 0
 - When x_2 is False, because $f(x) > 0 \Leftrightarrow x_1 \geq 6$
- There are many more feasible classifiers
 - $f(x) = (x_1 - 5.5) + 6 \cdot x_2$
 - $f(x) = 2 \cdot (x_1 - 5) + 11 \cdot x_2$
 -

Key Question: Which Line Is Better?

- ❑ There might be many feasible linear functions
 - ❑ Both H_1 and H_2 will work
- ❑ Which one is better?
 - ❑ H_2 looks “better” in the sense that it is also furthest from both groups
- ❑ We will introduce more in the SVM section



Generative vs. Discriminative Classifiers

- X: observed variables (features)
- Y: target variables (class labels)
- A generative classifier models $p(Y, X)$
 - It models how the data was "generated"? "what is the likelihood this or that class generated this instance?" and pick the one with higher probability
- Naïve Bayes
- Bayesian Networks
- A discriminative classifier models $p(Y|X)$
 - It uses the data to create a decision boundary
- Logistic Regression
- Support Vector Machines

Further Comments on Discriminative Classifiers

- Strength
 - Prediction accuracy is generally high
 - As compared to generative models
 - Robust, works when training examples contain errors
 - Fast evaluation of the learned target function
 - Comparing to [\(covered in future\)](#) Bayesian networks (which 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

Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ Summary



Model Evaluation and Selection

- Evaluation metrics
 - How can we measure accuracy?
 - Other metrics to consider?
- Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier's accuracy
 - Holdout method
 - Cross-validation
 - Bootstrap
- Comparing classifiers:
 - ROC Curves

Classifier Evaluation Metrics: Confusion Matrix

- Confusion Matrix:

Actual class\Predicted class	C_1	$\neg C_1$
C_1	True Positives (TP)	False Negatives (FN)
$\neg C_1$	False Positives (FP)	True Negatives (TN)

- In a confusion matrix w. m classes, $CM_{i,j}$ indicates # of tuples in class i that were labeled by the classifier as class j
 - May have extra rows/columns to provide totals
- Example of Confusion Matrix:

Actual class\Predicted class	play_golf = yes	play_golf = no	Total
play_golf = yes	6954	46	7000
play_golf = no	412	2588	3000
Total	7366	2634	10000

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	$\neg C$	
C	TP	FN	P
$\neg C$	FP	TN	N
	P'	N'	All

Real-world truth

Predictions

- **Classifier accuracy**, or recognition rate
 - Percentage of test set tuples that are correctly classified
$$\text{Accuracy} = (\text{TP} + \text{TN})/\text{All}$$
- **Error rate**: $1 - \text{accuracy}$, or
$$\text{Error rate} = (\text{FP} + \text{FN})/\text{All}$$

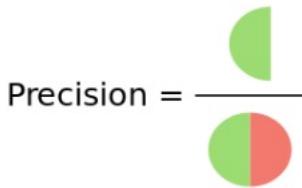
- **Class imbalance problem**
 - One class may be *rare*
 - E.g., fraud, or HIV-positive
 - Significant *majority of the negative class* and minority of the positive class
- Measures handle the class imbalance problem
 - **Sensitivity** (recall): True positive recognition rate
 - $\text{Sensitivity} = \text{TP}/\text{P}$
 - **Specificity**: True negative recognition rate
 - $\text{Specificity} = \text{TN}/\text{N}$

Classifier Evaluation Metrics: Precision and Recall, and F-measures

A\P	C	$\neg C$	
C	TP	FN	P
$\neg C$	FP	TN	N
	P'	N'	All

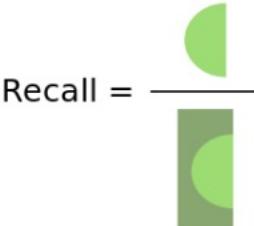
- **Precision:** Exactness: what % of tuples that the classifier labeled as positive are actually positive?

$$P = \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

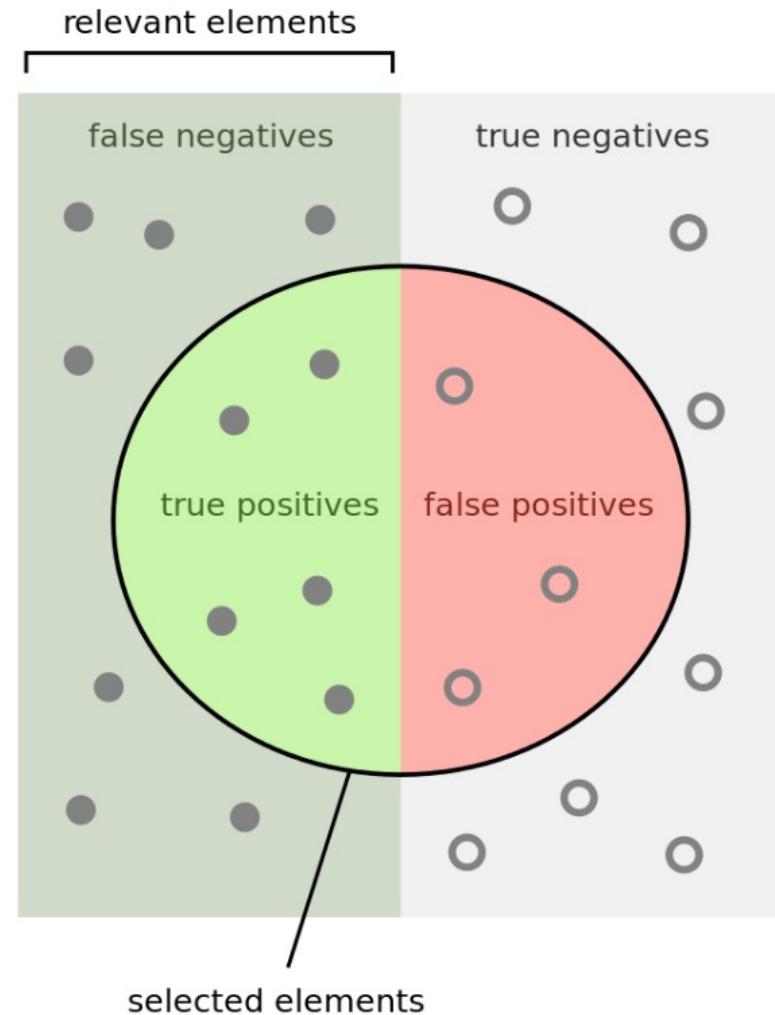


- **Recall:** Completeness: what % of positive tuples did the classifier label as positive?

$$R = \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$



- **Range:** [0, 1]



Classifier Evaluation Metrics: Precision and Recall, and F-measures

- The “inverse” relationship between precision & recall
- ***We want one number to say if a classifier is good or not***
- **F measure (or F-score):** harmonic mean of precision and recall
 - In general, it is the weighted measure of precision & recall

$$F_{\beta} = \frac{1}{\alpha \cdot \frac{1}{P} + (1 - \alpha) \cdot \frac{1}{R}} = \frac{(\beta^2 + 1)P * R}{\beta^2 P + R}$$

Assigning β times as much weight to recall as to precision)

- ***F1-measure (balanced F-measure)***

- That is, when $\beta = 1$,

$$F_1 = \frac{2P * R}{P + R}$$

Classifier Evaluation Metrics: Example

- ❑ Use the same confusion matrix, calculate the measure just introduced

Actual Class\Predicted class	cancer = yes	cancer = no	Total
cancer = yes	90	210	300
cancer = no	140	9560	9700
Total	230	9770	10000

- ❑ Sensitivity =

- ❑ Specificity =

- ❑ Accuracy =

- ❑ Error rate =

- ❑ Precision =

- ❑ Recall =

- ❑ F1 =

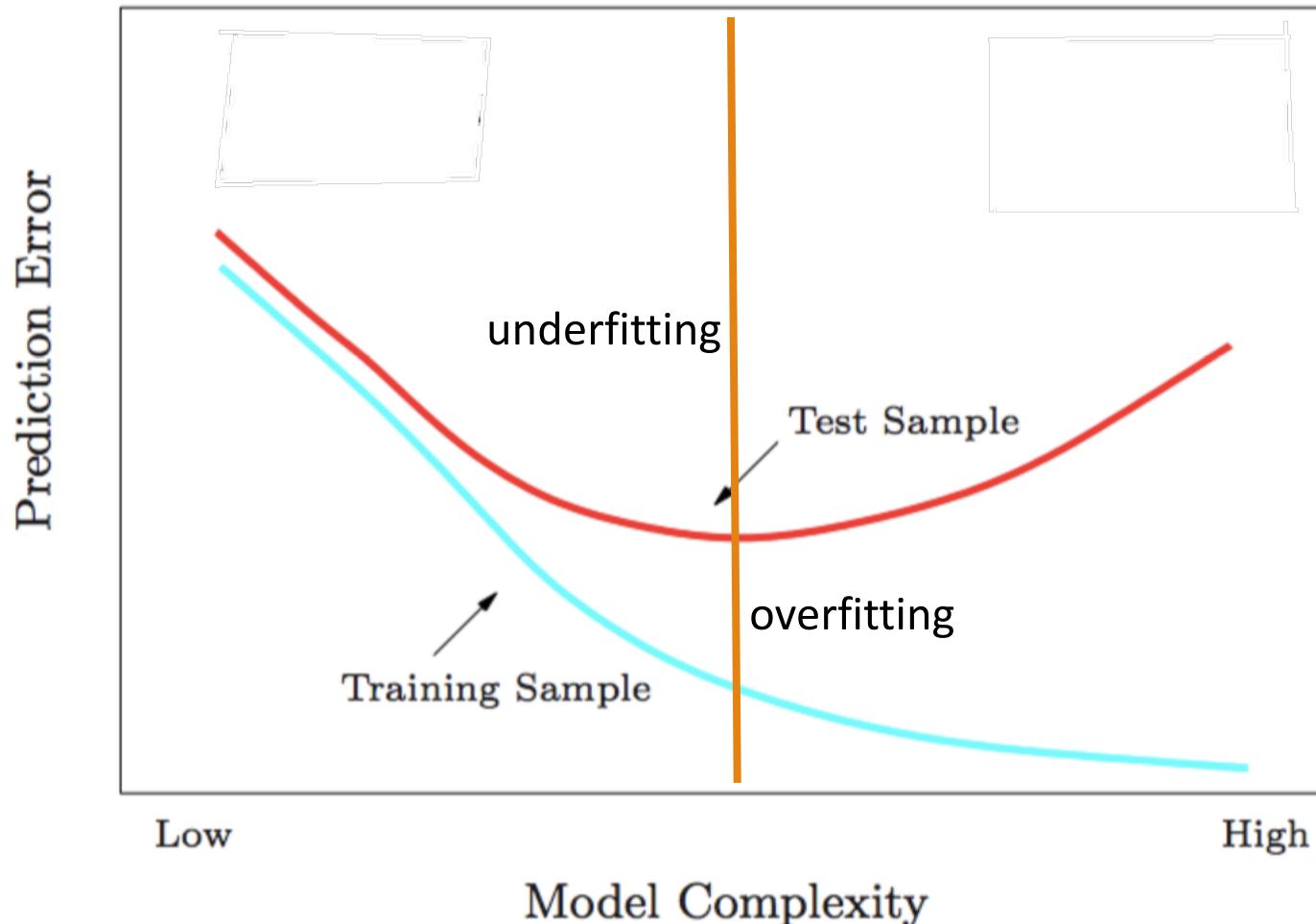
Classifier Evaluation Metrics: Example

- Use the same confusion matrix, calculate the measure just introduced

Actual Class\Predicted class	cancer = yes	cancer = no	Total
cancer = yes	90	210	300
cancer = no	140	9560	9700
Total	230	9770	10000

- Sensitivity = $TP/P = 90/300 = 30\%$
- Specificity = $TN/N = 9560/9700 = 98.56\%$
- Accuracy = $(TP + TN)/All = (90+9560)/10000 = 96.50\%$
- Error rate = $(FP + FN)/All = (140 + 210)/10000 = 3.50\%$
- Precision = $TP/(TP + FP) = 90/(90 + 140) = 90/230 = 39.13\%$
- Recall = $TP/ (TP + FN) = 90/(90 + 210) = 90/300 = 30.00\%$
- $F1 = 2 P \times R / (P + R) = 2 \times 39.13\% \times 30.00\% / (39.13\% + 30\%) = 33.96\%$

Training Error VS Testing Error



Classifier Evaluation: Holdout

- **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
 - Repeated random sub-sampling validation: a variation of holdout
 - Repeat holdout k times, accuracy = avg. of the accuracies obtained

Classifier Evaluation: Cross-Validation

- **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 distribution, in each fold is approximately the same as that in the initial data

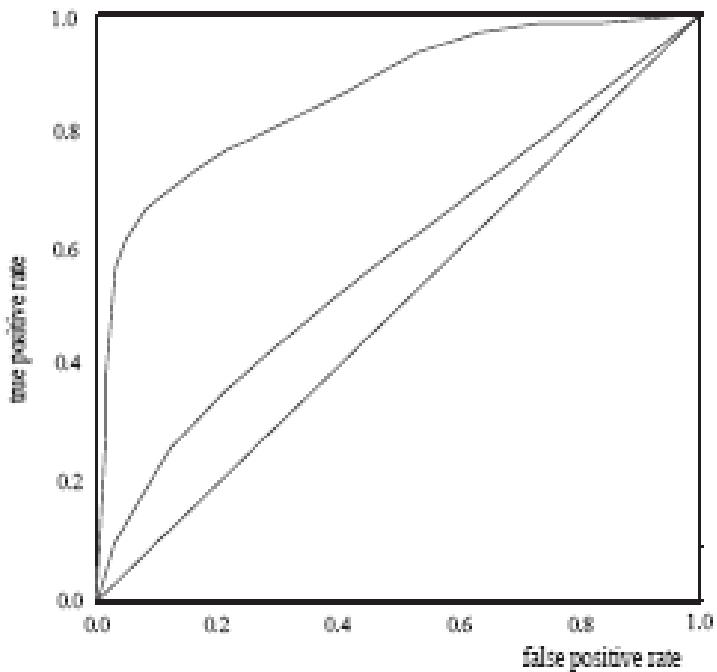
Classifier Evaluation: Bootstrap

- **Bootstrap**
 - Works well with small data sets
 - Samples the given training tuples uniformly *with replacement*
 - 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**
 - A data set with d tuples 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 end up in the bootstrap, and the remaining 36.8% 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:

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

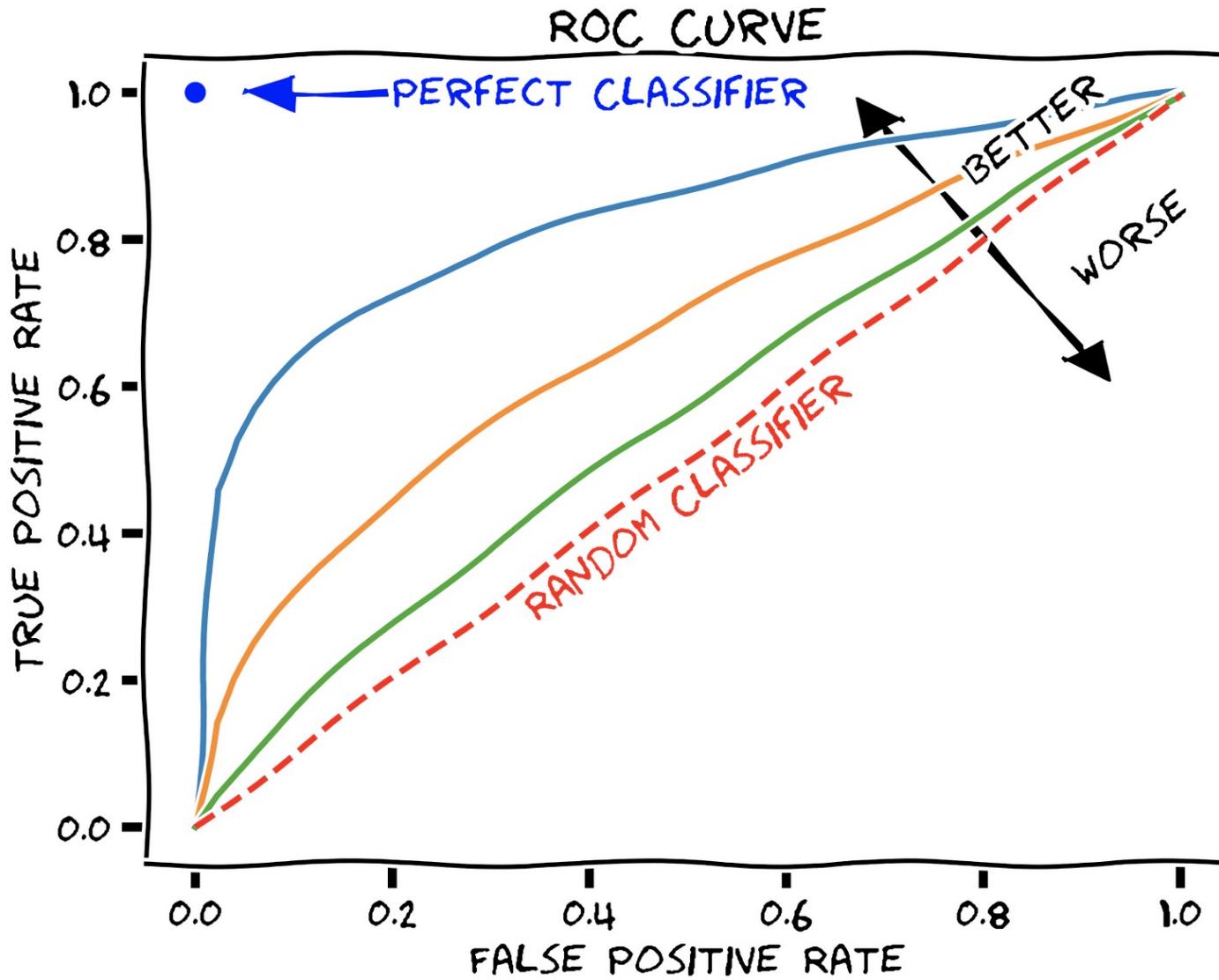
Model Selection: ROC Curves

- ❑ 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 (**AUC**: Area Under 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 (TP/P)
- ❑ Horizontal axis rep. the false positive rate (FP/N)
- ❑ The plot also shows a diagonal line
- ❑ A model with perfect accuracy will have an area of 1.0

ROC and AUC



Issues Affecting Model Selection

- **Accuracy**
 - classifier accuracy: predicting class label
- **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

Model Selection, Statistical Significance

- k-Fold Cross-Validation: Models M1, M2
 - Each classifier gets k error rates $\text{err}(M1)_i, \text{err}(M2)_i, i=1,\dots,k$
 - Comparing average error rate is tricky
 - Need to consider the variance, more generally the distribution of error rates
- Hypothesis testing: Student's t-test
 - Individual error rates follow a t-distribution (under some assumptions)
 - Null hypothesis: the distributions are the same
 - t-statistic for pairwise comparison

$$t = \frac{\overline{\text{err}}(M_1) - \overline{\text{err}}(M_2)}{\sqrt{\text{var}(M_1 - M_2)/k}},$$

where

$$\text{var}(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^k [\text{err}(M_1)_i - \text{err}(M_2)_i - (\overline{\text{err}}(M_1) - \overline{\text{err}}(M_2))]^2$$

Model Selection, Statistical Significance

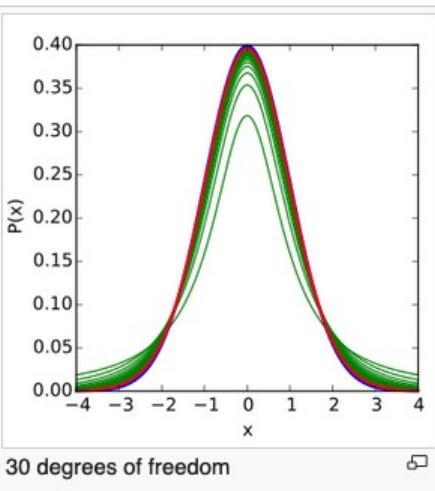
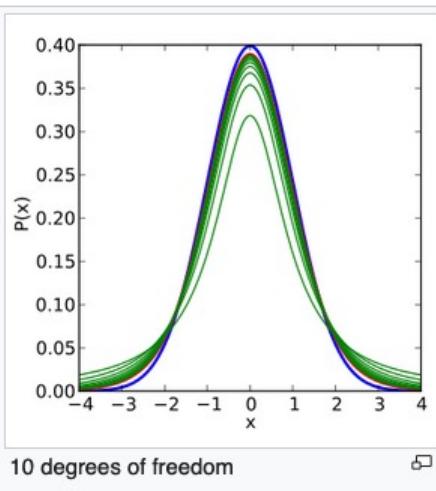
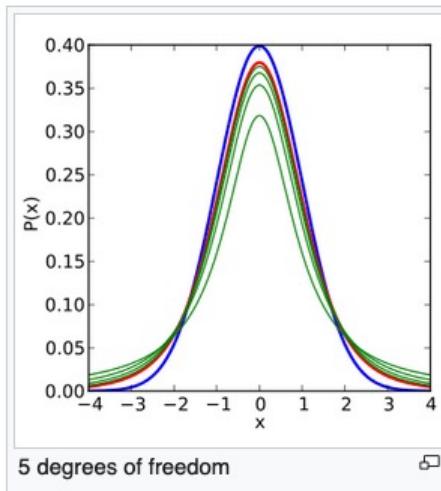
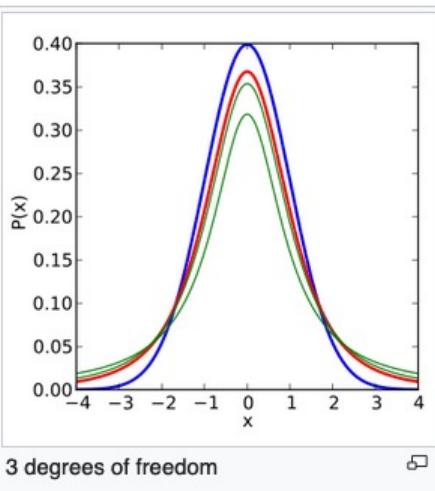
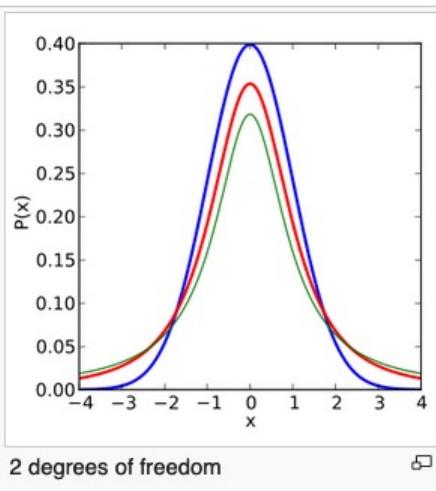
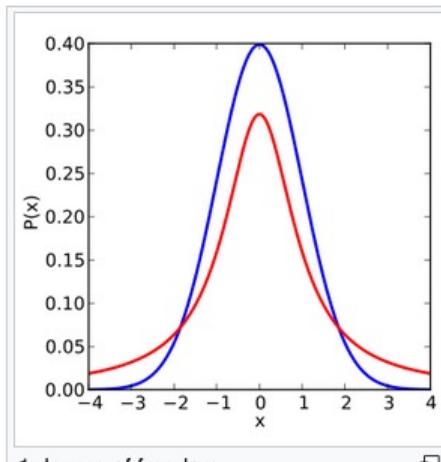
- Compute t-statistic from k-Fold error rates of M1, M2
 - Degrees of freedom is (k-1)
- Consider a significance level, say $\alpha=5\%$ (or 1%)
 - Use the fact that t-distribution is symmetric, upper/lower tail at level $\alpha/2$
 - Look up the statistic value t_0 at this level
 - If $t > t_0$ or $t < -t_0$, then the difference is significant, reject null hypothesis
- If there are test sets, two sample t-test based on variance of difference

$$\text{var}(M_1 - M_2) = \frac{\text{var}(M_1)}{k_1} + \frac{\text{var}(M_2)}{k_2},$$

Student's t-distribution

Density of the t -distribution (red) for 1, 2, 3, 5, 10, and 30 degrees of freedom compared to the standard normal distribution (blue).

Previous plots shown in green.



Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ 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 tuples, 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 an implicit global approximation to the target function
 - Eager: must commit to a single hypothesis that covers the entire instance space

Lazy Learning

- ❑ Nearest neighbor (NN) based
- ❑ Prediction is “lazy”
 - ❑ Training: keep points around, “lazy”
 - ❑ Prediction: find nearest neighbors, make prediction
- ❑ Most work happens during prediction
 - ❑ Training: proper indexing to help NN search
 - ❑ Prediction: find similar points, cases, etc.
- ❑ Properties
 - ❑ Can be computationally expensive
 - ❑ Supports incremental learning

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

k-Nearest Neighbor

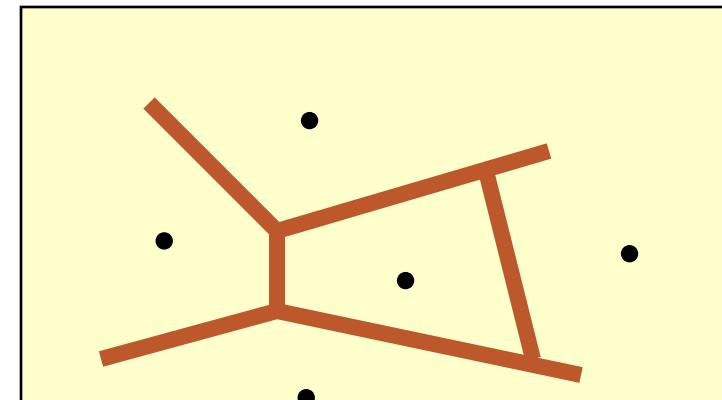
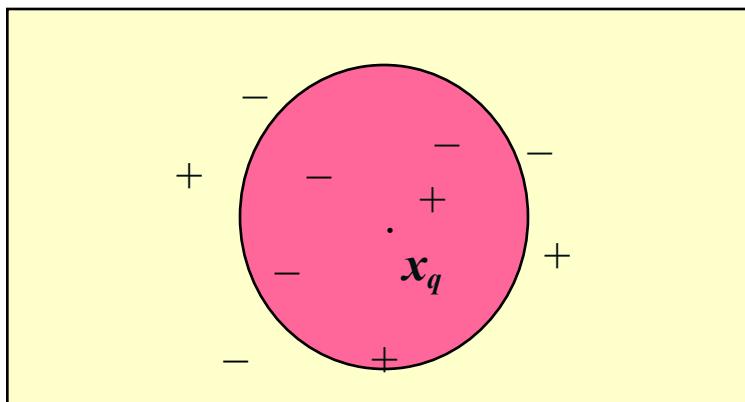
- Nearest neighbor (NN) based
 - Points based on n attributes
 - Have a notion of distance between points
 - Given test point, find k nearest neighbors
 - Prediction: (weighted) majority of k nearest neighbors
- Distance measure
 - Metric, such as Euclidean distance
 - Normalize attributes
 - Generalization to nominal attributes, e.g., same (0) vs different (1)

k-Nearest Neighbor

- Nearest neighbor (NN) for numeric prediction
 - Data points (x_i, y_i) , $i=1, \dots, n$
 - Given test point x_{test} , find k nearest neighbors, from x_i
 - Prediction y_{test} : (weighted) sum of responses (y_i) of k nearest neighbors
 - Weighting based on distance to x_{test}
- Missing values: Worst possible distance, imputation
- Choice of k
 - Should increase with n , choose based on validation set
 - For $k=1$, gets twice “Bayes error rate” as $n \rightarrow \infty$

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, $\text{dist}(X_1, X_2)$
- 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



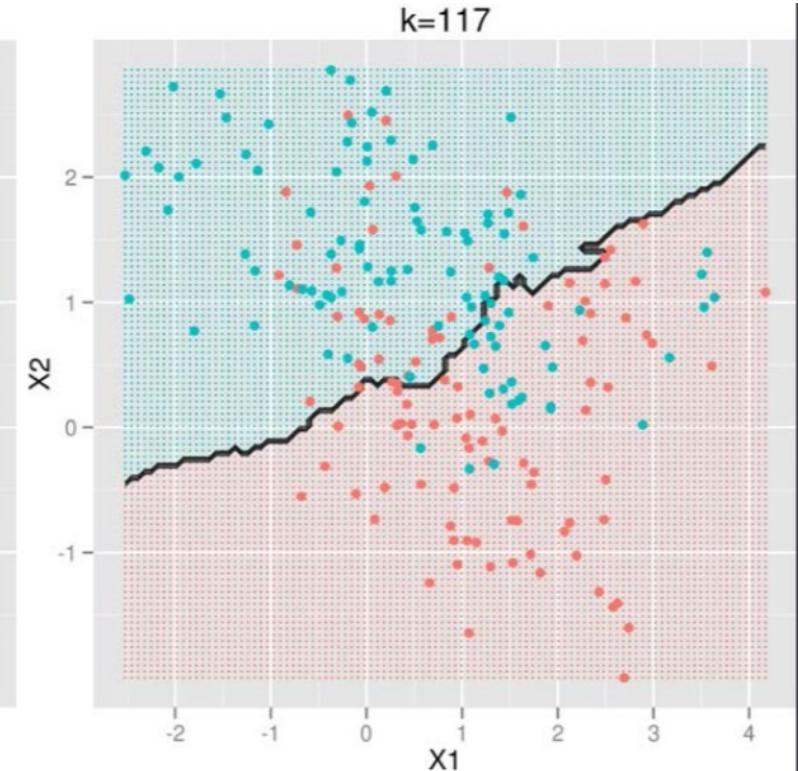
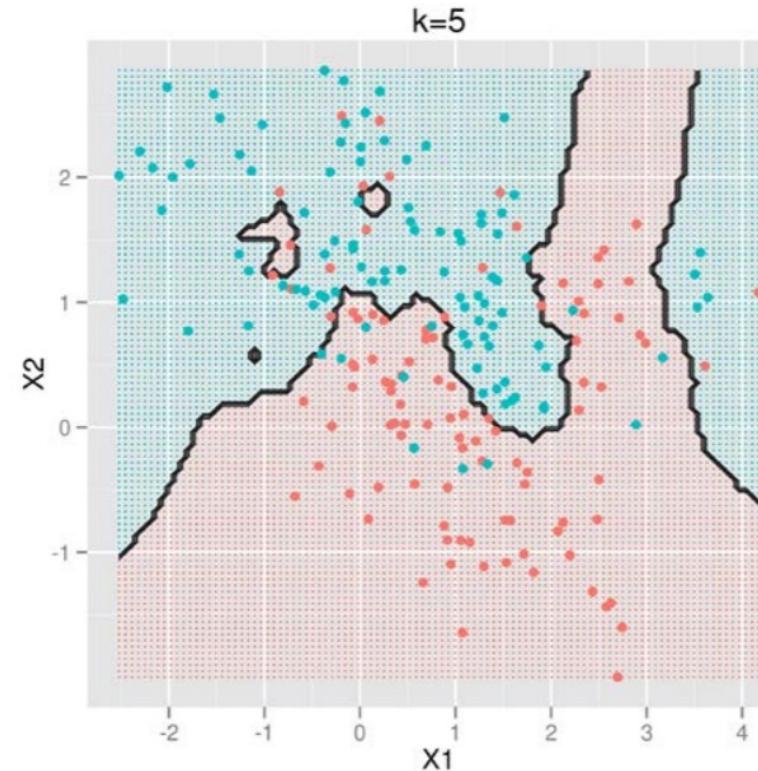
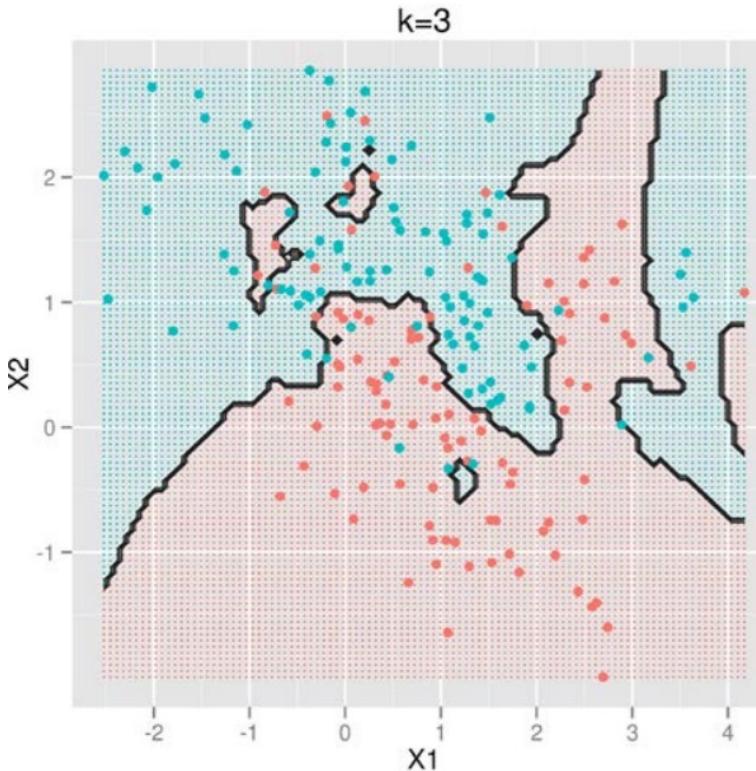
Discussion on the k -NN Algorithm

- 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

$$w = \frac{1}{d(x_q, x_i)^2}$$

Selection of k for kNN

- The number of neighbors k
 - Small k: overfitting (high var., low bias)
 - Big k: bringing too many irrelevant points (high bias, low var.)



k-Nearest Neighbor

- Importance of distance in Nearest neighbor (NN)
 - Can suffer based on bad distances, learn (parametric) distances
 - Can suffer due to noisy attributes; weighting, pruning of noisy attributes
- Can be extremely slow
 - $O(|D|)$ comparisons are needed to classify
 - Using search trees, can be reduced to $O(\log |D|)$
 - Parallel implementations
- Speeding up computations
 - Partial distance calculations
 - Editing, compressing to reduce number of points
 - Locality sensitive hashing, approximate NN, with high probability

Case-Based Reasoning

- Store “cases” with symbolic representation
- Prediction based on NN type idea
 - Find if an identical case exists
 - If not, find cases which have similar components
 - E.g., cases are graphs, find similarity on subgraphs
 - Combine solution from such related cases to solve the new case
- Challenges
 - Good similarity metric among cases
 - Salient features for indexing training cases
- Trade-offs: Accuracy vs. Efficiency
 - Prune, compress, index, etc., for efficiency

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

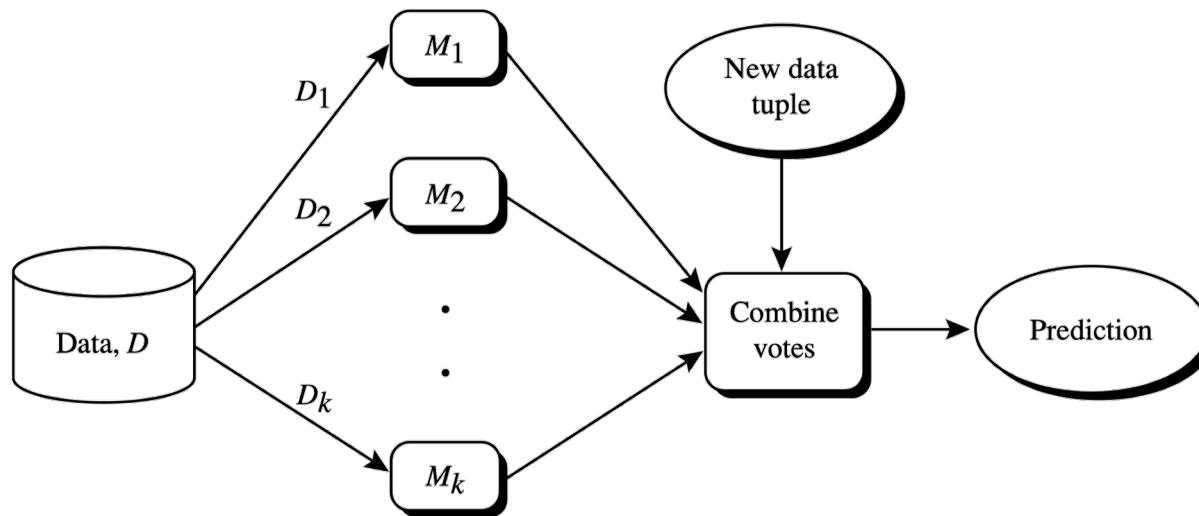
Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ 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^*



Ensemble Methods: Increasing the Accuracy

- What are the requirements to generate an improved model?
 - Example: majority voting

	x_1	x_2	x_3	
Base model performance	M_1	✓	✓	X
	M_2	X	✓	✓
	M_3	✓	X	✓
Ensemble performance	Voting Ensemble	✓	✓	✓

Case 1:
Ensemble has positive effect

	x_1	x_2	x_3	
	M_1	✓	✓	X
	M_2	✓	✓	X
	M_3	✓	✓	X
Voting Ensemble	Voting Ensemble	✓	✓	X

Case 2:
Ensemble has no effect

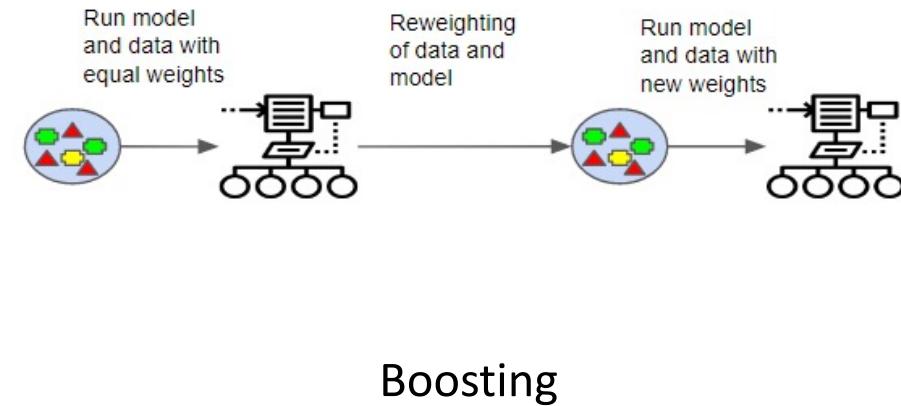
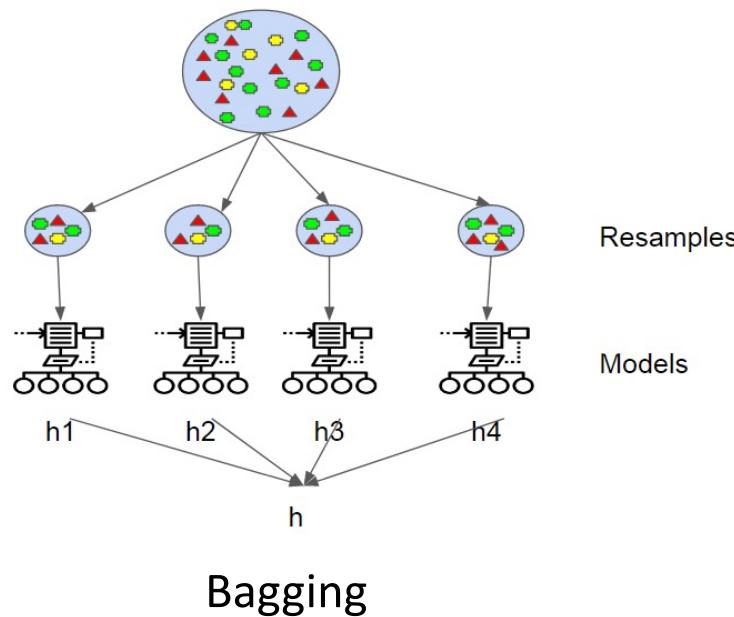
	x_1	x_2	x_3	
	M_1	✓	X	X
	M_2	X	✓	X
	M_3	X	X	✓
Voting Ensemble	Voting Ensemble	X	X	X

Case 3:
Ensemble has negative effect

- Base models should be
 - Accurate
 - Diverse

Ensemble Methods: Increasing the Accuracy

- Popular ensemble methods
 - Bagging: Trains each model using a subset of the training set, and models learned in parallel
 - Boosting: Trains each new model instance to emphasize the training instances that previous models mis-classified, and models learned in order



Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
 - For $i = 1$ to k
 - create bootstrap sample, D_i , by sampling D with replacement;
 - use D_i and the learning scheme to derive a model, M_i ;
- Classification: classify an unknown sample X
 - let each of the k models classify X and return the majority vote
- Prediction:
 - To predict continuous variables, use average prediction instead of vote

Bagging: Bootstrap Aggregation

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D , a set of d training tuples;
- k , the number of models in the ensemble;
- a classification learning scheme (e.g., decision tree algorithm, naïve Bayes, etc.).

Output: The ensemble—a composite model, M^* .

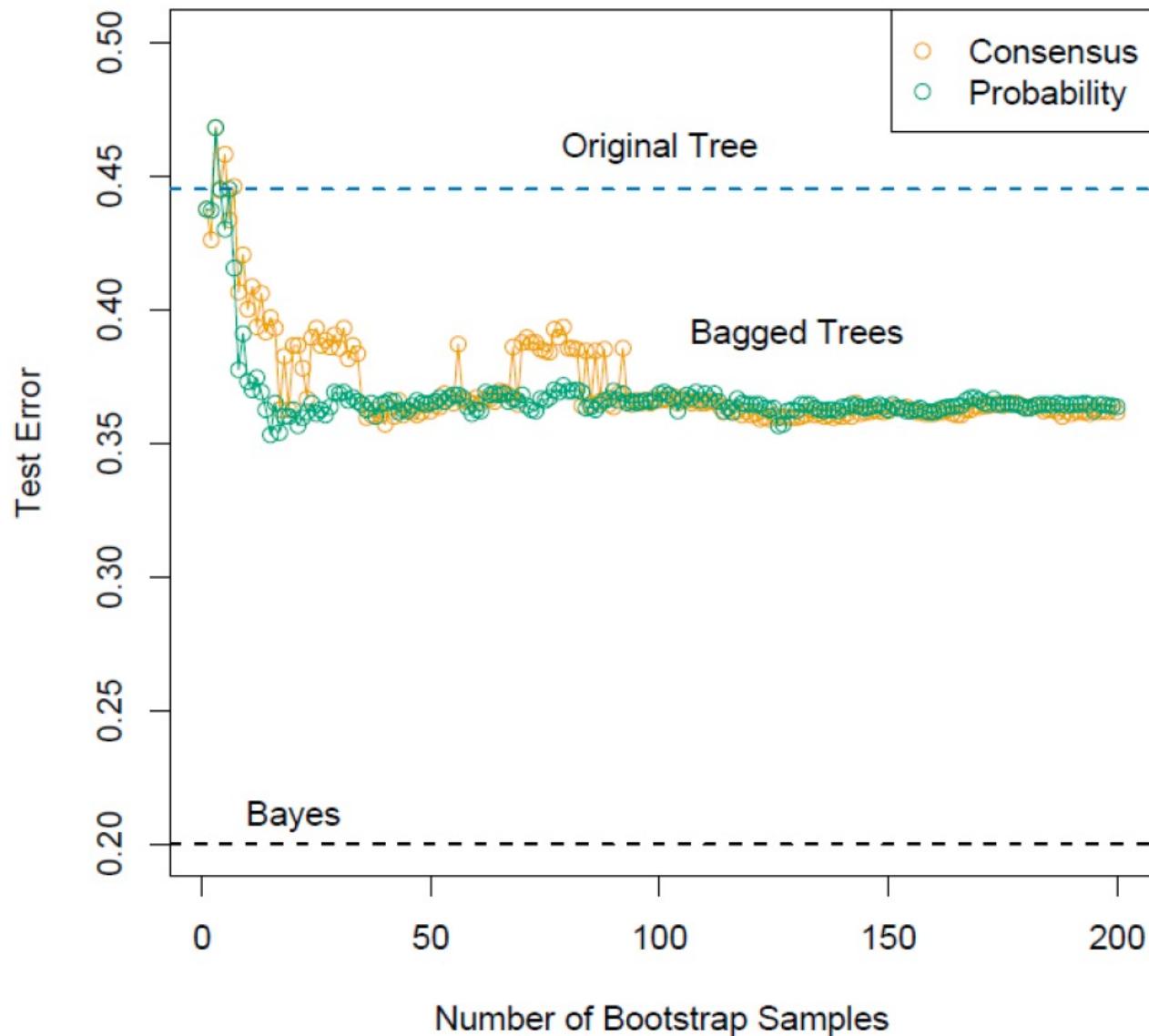
Method:

- (1) **for** $i = 1$ to k **do** // create k models:
- (2) create bootstrap sample, D_i , by sampling D with replacement;
- (3) use D_i and the learning scheme to derive a model, M_i .
- (4) **endfor**

To use the ensemble to classify a tuple, X :

- (1) let each of the k models classify X and return the majority vote;

Bagging: Bootstrap Aggregation



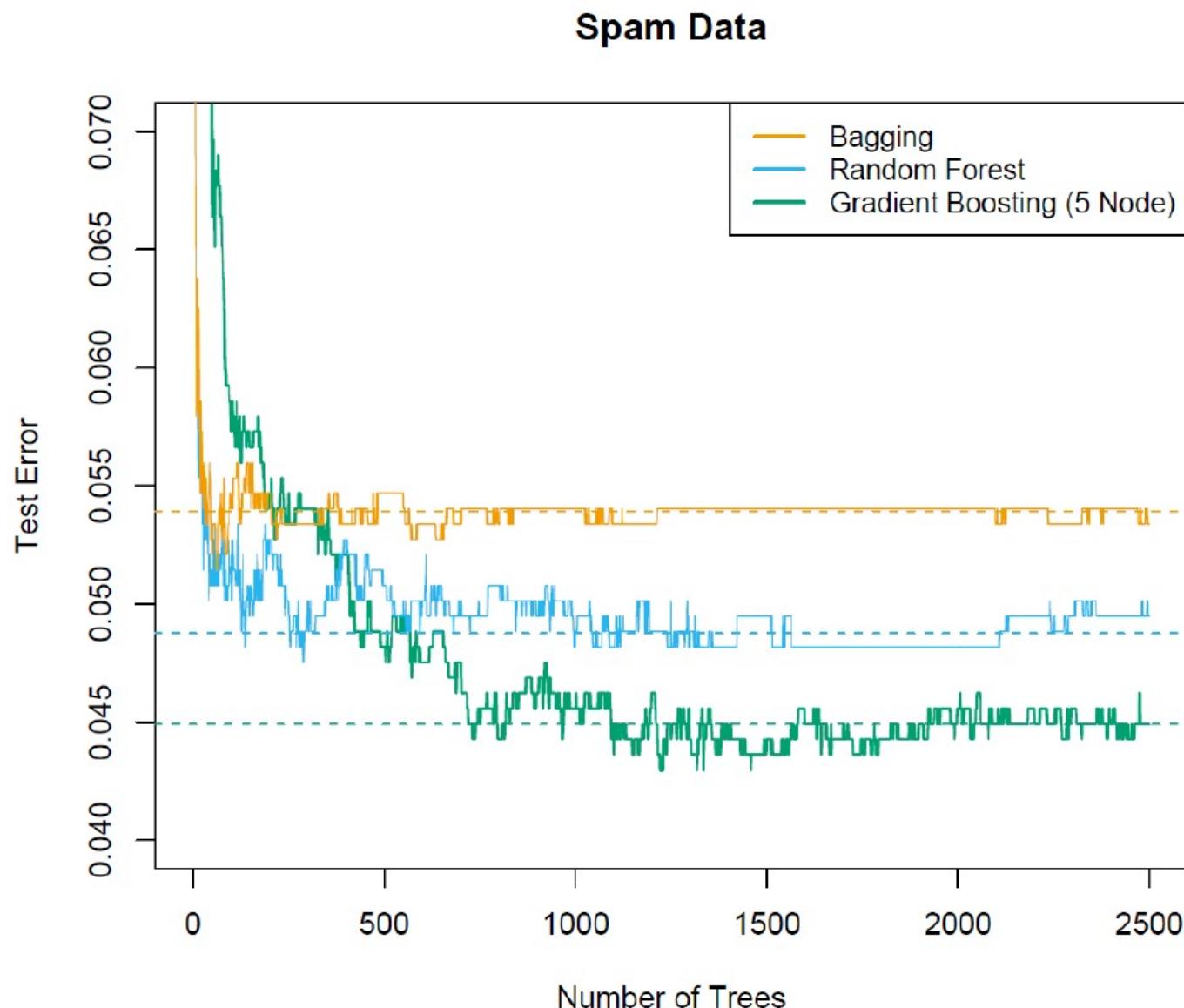
Random Forest: Basic Concepts

- ❑ Random Forest (first proposed by L. Breiman in 2001)
 - ❑ Bagging with **decision trees** as base models
 - ❑ *Data bagging*
 - ❑ Use a **subset of training data** by sampling with replacement for each tree
 - ❑ *Feature bagging* ← Advantage of decision trees – more diversity
 - ❑ At each node use a **random selection of attributes** as candidates and split by the best attribute among them
- ❑ During classification, each tree votes and the most popular class is returned

Random Forest

- ❑ Two Methods to construct Random Forest:
 - ❑ Forest-RI (*random input selection*): Randomly select, at each node, F attributes as candidates for the split at the node. The CART methodology is used to grow the trees to maximum size
 - ❑ Forest-RC (*random linear combinations*): Creates new attributes (or features) that are a linear combination of the existing attributes (reduces the correlation between individual classifiers)
- ❑ Comparable in accuracy to Adaboost, but more robust to errors and outliers
- ❑ Insensitive to the number of attributes selected for consideration at each split, and faster than typical bagging or boosting

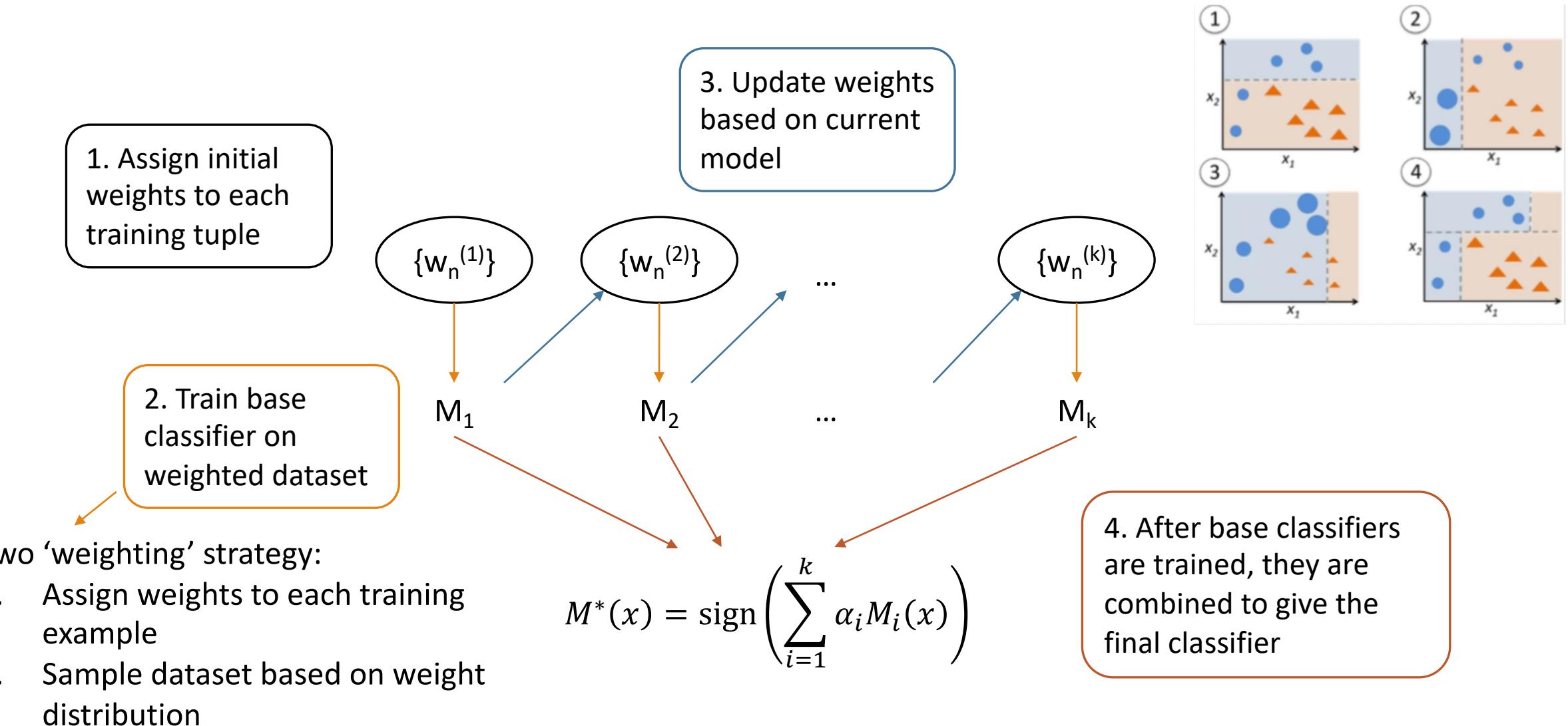
Ensemble Methods: Comparison



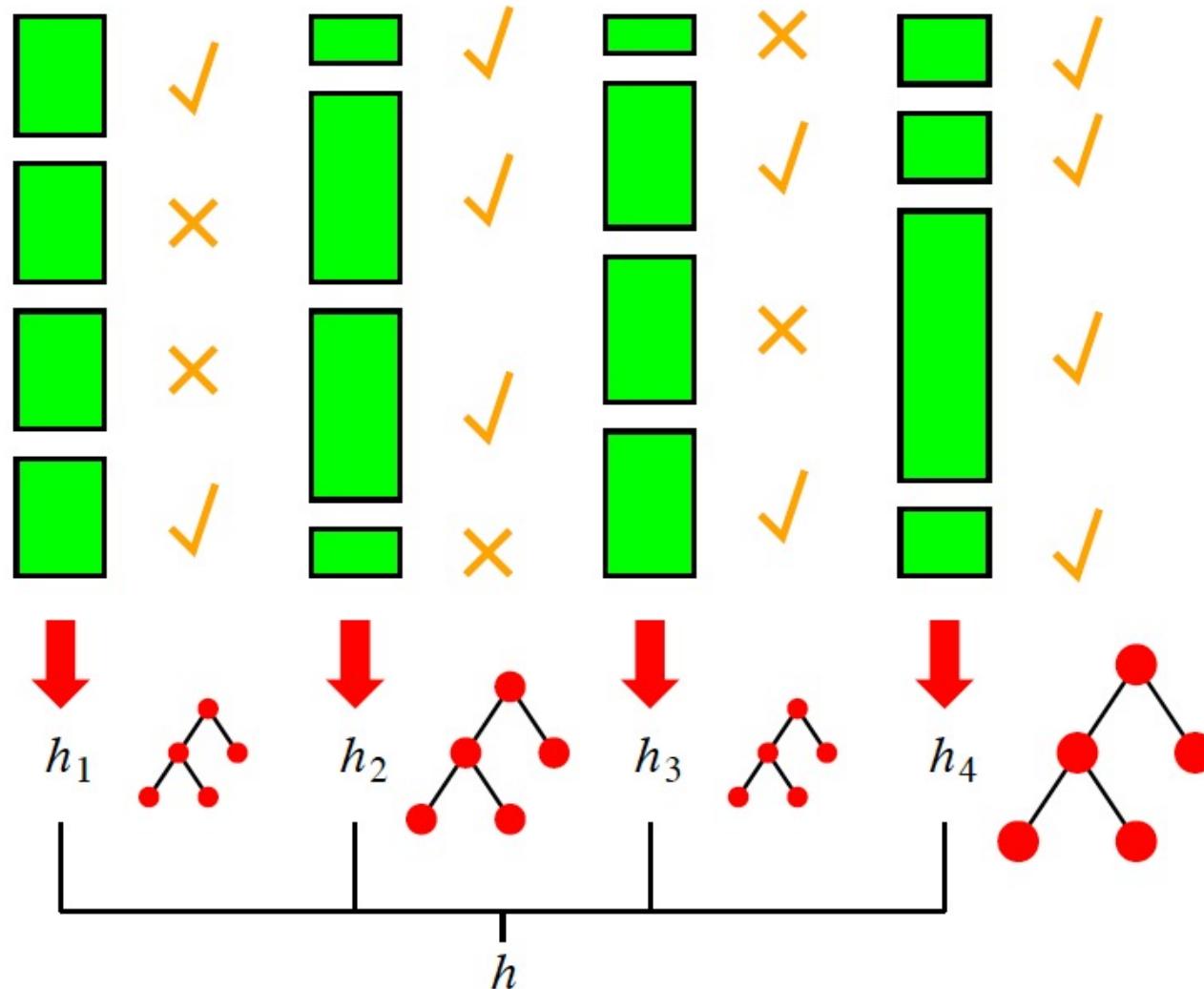
Boosting

- ❑ Analogy: Consult several doctors, based on a combination of weighted diagnoses—weight assigned based on the previous diagnosis accuracy
- ❑ How boosting works?
 - ❑ A series of k classifiers are iteratively learned
 - ❑ After a classifier M_i is learned, set 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
- ❑ Boosting algorithm can be extended for numeric prediction

Adaboost (Freund and Schapire, 1997)



Adaboost



Adaboost (Freund and Schapire, 1997)

- **Input:** Training set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$
- Initialize all weights $\{w_n^{(1)}\}$ to $1/N$
- For round $i = 1$ to k ,

- Fit a classifier M_i based on weighted error function

$$J_m = \sum_{n=1}^N w_n^{(i)} I(M_i(x_n) \neq y_n)$$

- Evaluate error rate $\epsilon_i = J_m / \sum w_n^{(i)}$ (stop iteration if $\epsilon_i < \text{threshold}$)

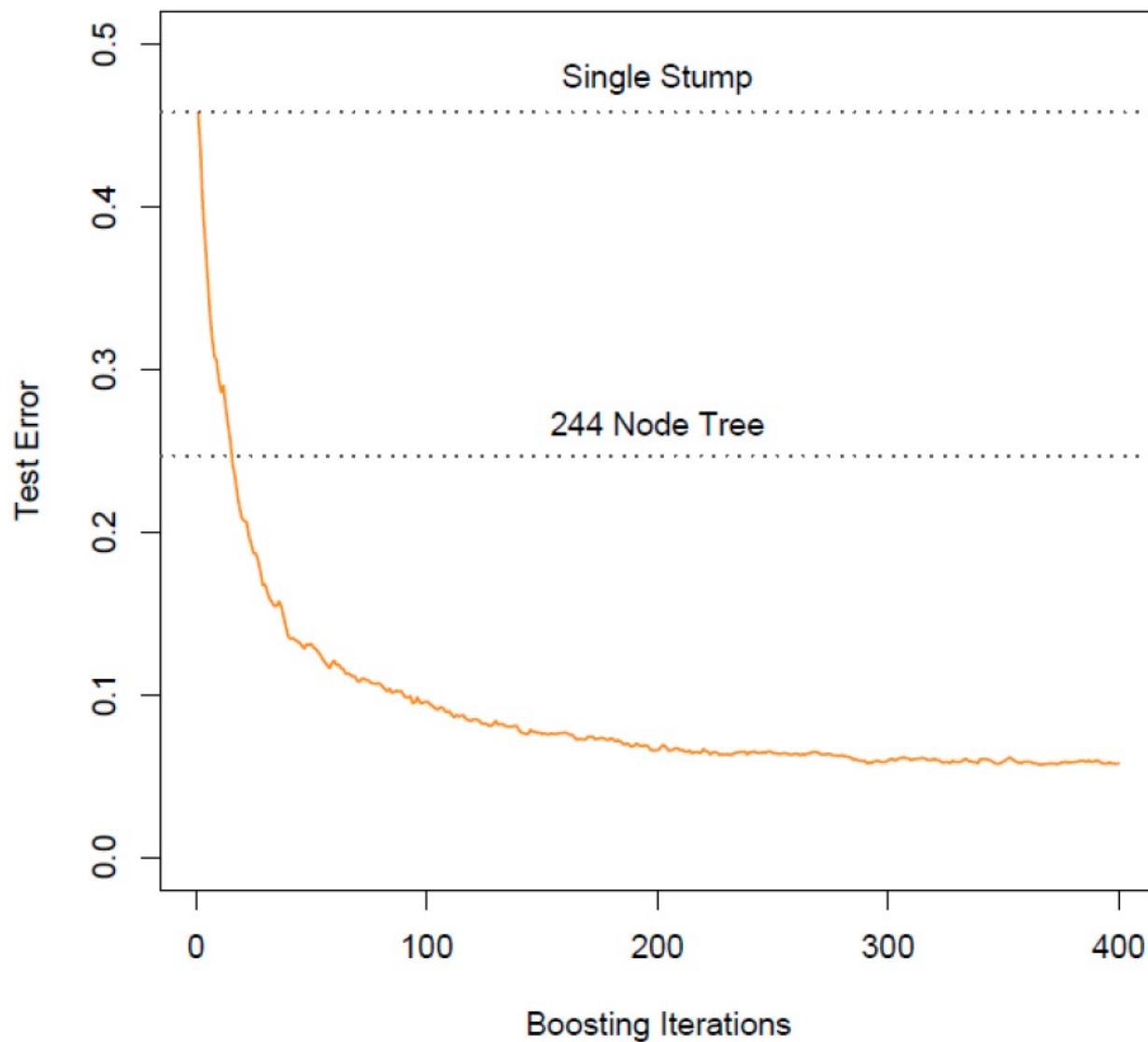
and the base model M_i 's vote $\alpha_i = \frac{1}{2} \ln \left(\frac{1-\epsilon_i}{\epsilon_i} \right)$

- Update weights

$$w_n^{(i+1)} = w_n^{(i)} \exp\{\alpha_i \cdot I(M_i(x_n) \neq y_n)\}$$

- The final model is given by voting based on $\{\alpha_n\}$

Adaboost, with Decision Stumps



Gradient Boosting

- Operates on:
 - A differentiable loss function
 - A weak learner to make predictions (usually trees)
 - An additive model to add weak learners to minimize the loss function
- Each time adds an additional weak learner

$$\begin{aligned}\hat{y}_i^{(0)} &= 0 \\ \hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\ \hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\ &\dots \\ \hat{y}_i^{(t)} &= \sum_{k=1}^t f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)\end{aligned}$$

Previous model New weak learner



- Scalable implementation: XGBoost

Gradient Boosting

Algorithm: Gradient Tree Boosting for Regression.

Input:

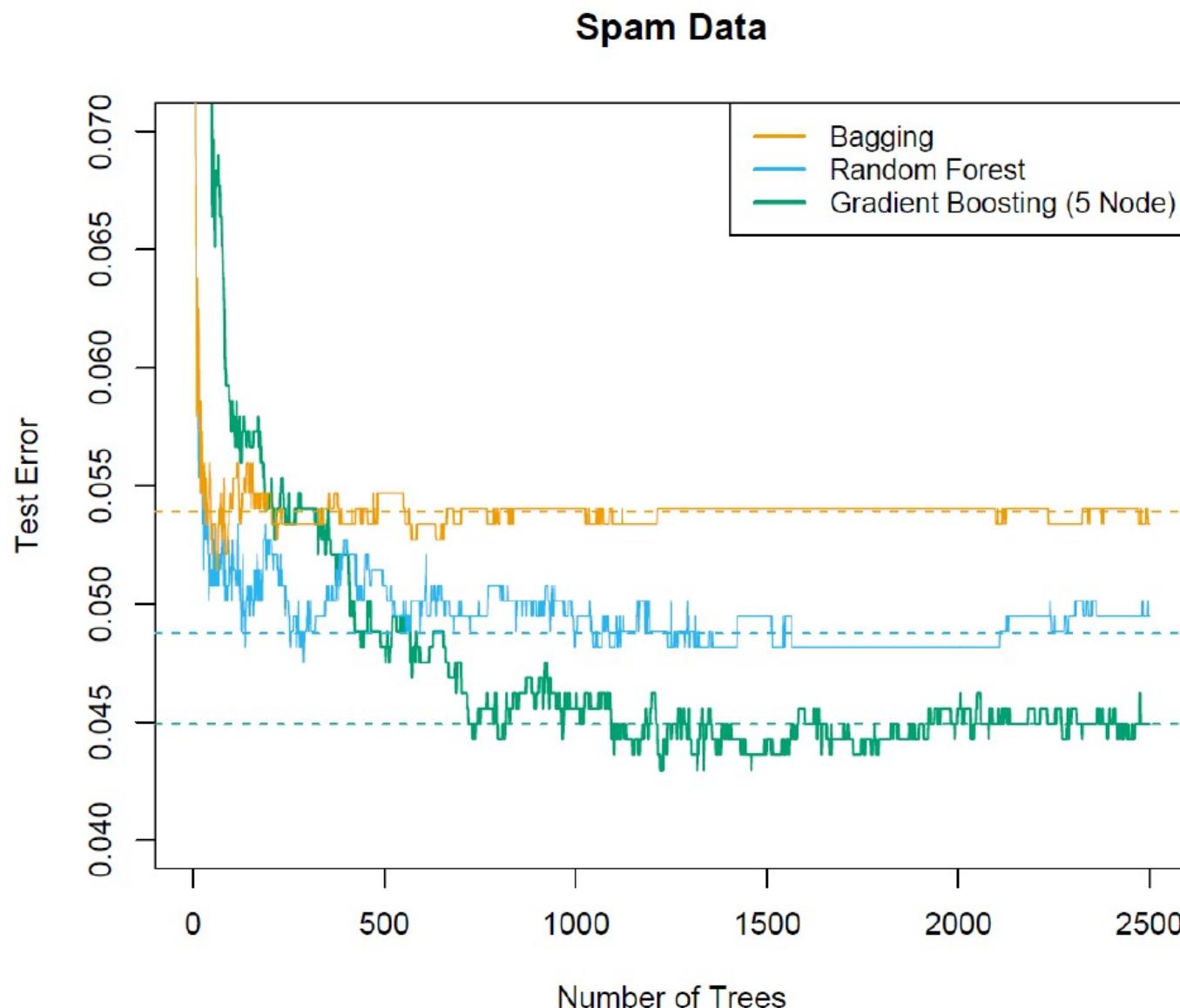
- D , a set of n training tuples $\{(x_1, y_1), \dots, (x_n, y_n)\}$, where x_i is the attribute vector of the i^{th} training tuple and y_i is its true target output value;
- k , the number of rounds (one base regression model is generated per round);
- a differential loss function $\text{Loss} = \sum_{i=1}^n L(y_i, F(x_i))$.

Output: A composite regression model $F(x)$.

Method:

- (1) initialize the regression model $F(x) = \frac{\sum_{i=1}^n y_i}{n}$;
- (2) **for** $t = 1$ to k **do** // construct a new weak learner $M_t(x)$ for each round:
 - (3) **for** $i = 1$ to n //each training tuple:
 - (4) calculate $\hat{y}_i = F(x_i)$; //predicted value by the current model $F(x)$
 - (5) calculate the negative gradient $r_i = -\frac{\partial L(y_i, \hat{y}_i)}{\partial \hat{y}_i}$;
 - (6) **endfor**
 - (7) fit a regression tree model $M_t(x)$ for the training set $\{(x_1, r_1), \dots, (x_n, r_n)\}$;
 - (8) update the composite regression model $F(x) \leftarrow F(x) + M_t(x)$.
 - (9) **endfor**

Ensemble Methods: Comparison



Ensemble Methods Recap

- ❑ Random forest and XGBoost are the most commonly used algorithms for tabular data
- ❑ Pros
 - ❑ Good performance for tabular data, requires no data scaling
 - ❑ Can scale to large datasets
 - ❑ Can handle missing data to some extent
- ❑ Cons
 - ❑ Can overfit to training data if not tuned properly
 - ❑ Lack of interpretability (compared to decision trees)

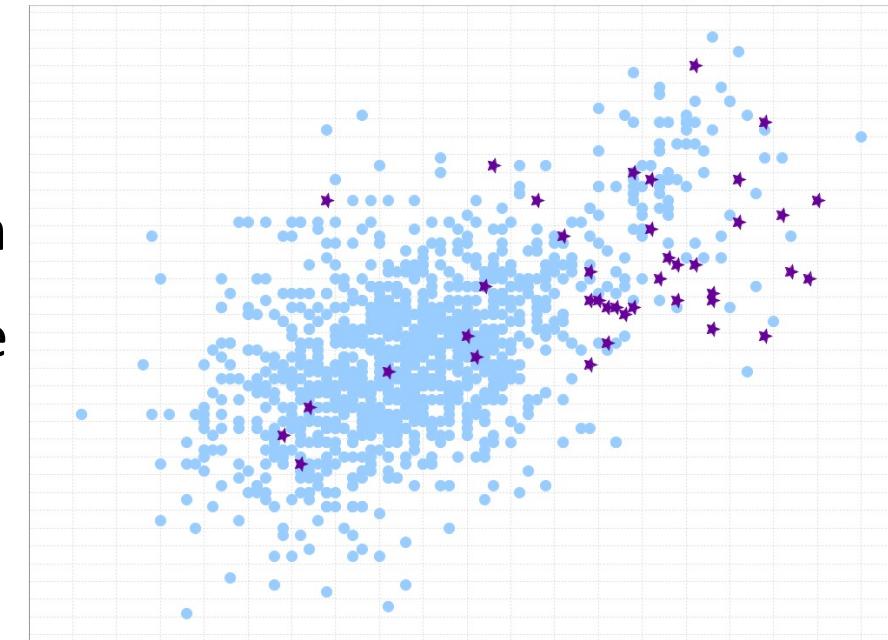
Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ Summary



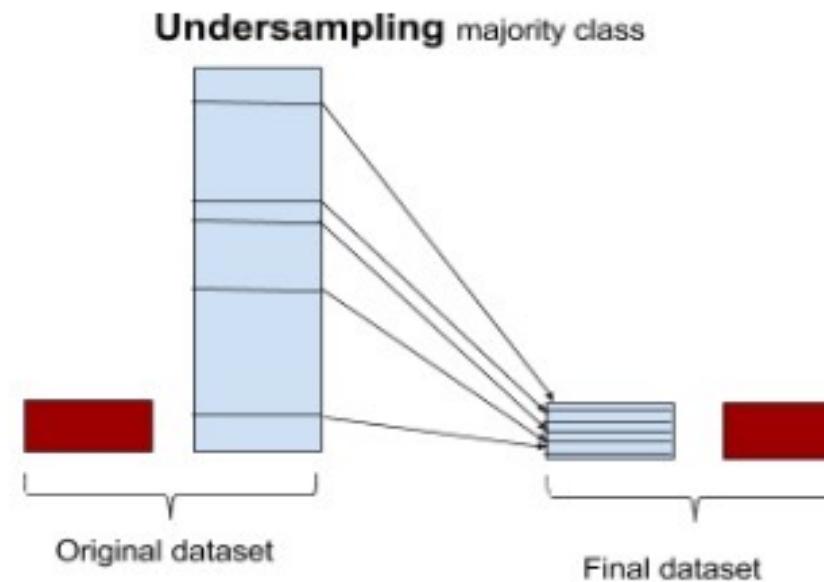
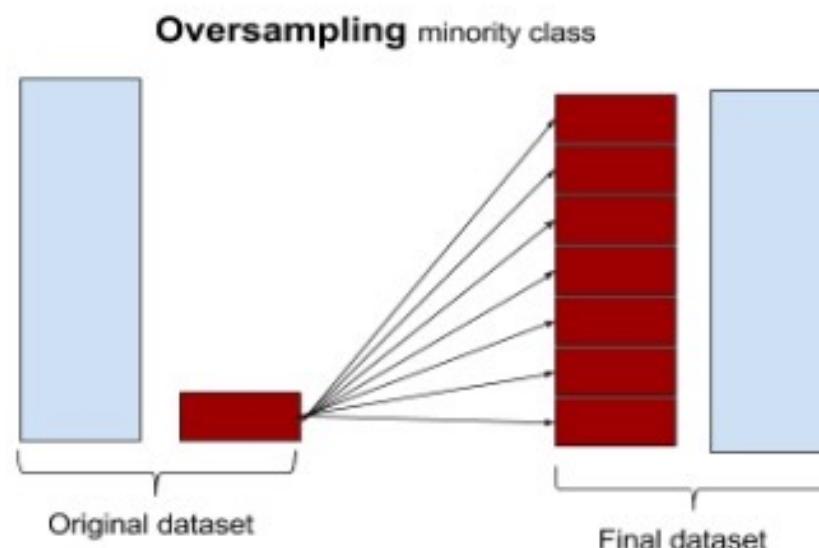
Classification of Class-Imbalanced Data Sets

- ❑ Traditional methods assume a balanced distribution of classes and equal error costs. But in real world situations, we may face imbalanced data sets, which has rare positive examples but numerous negative ones.
- ❑ Medical diagnosis: Medical screening for a condition is usually performed on a large population of people without the condition, to detect a small minority with it (e.g., HIV prevalence in the USA is ~0.4%)
- ❑ Fraud detection: About 2% of credit card accounts are defrauded per year. (Most fraud detection domains are heavily imbalanced.)
- ❑ Product defect, accident (oil-spill), disk drive failures, etc.



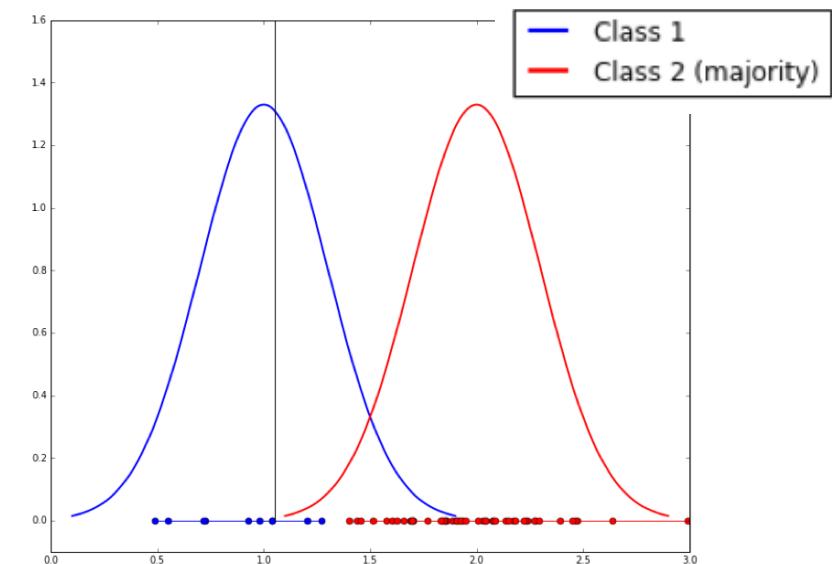
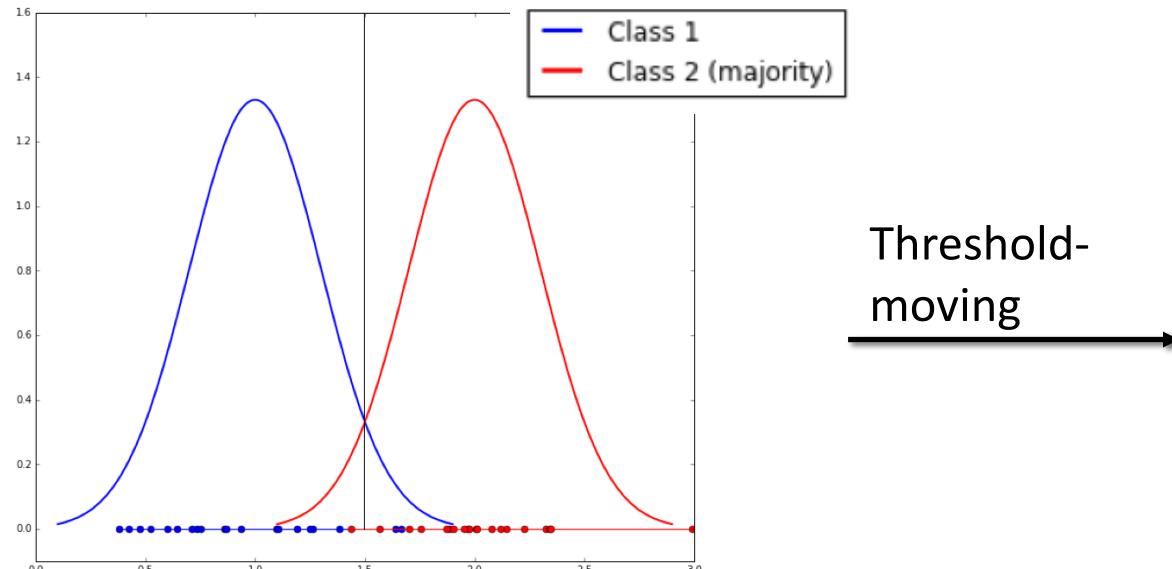
Classification of Class-Imbalanced Data Sets

- Typical methods on imbalanced data (Balance the training set)
 - **Oversampling:** Oversample the minority class.
 - **Under-sampling:** Randomly eliminate tuples from majority class
 - **Synthesizing:** Synthesize new minority classes



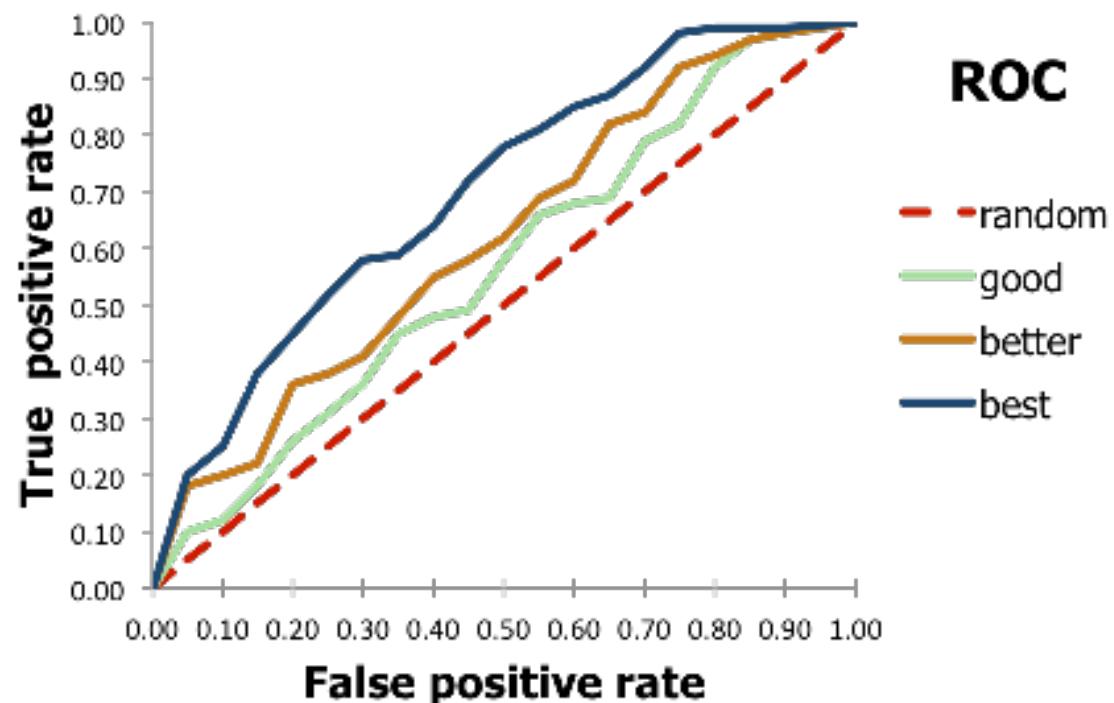
Classification of Class-Imbalanced Data Sets

- Typical methods on imbalanced data (At the algorithm level)
 - **Threshold-moving:** Move the decision threshold, t , so that the rare class tuples are easier to classify, and hence, less chance of costly false negative errors
 - **Class weight adjusting:** Since false negative costs more than false positive, we can give larger weight to false negative
- **Ensemble techniques:** Ensemble multiple classifiers introduced in the following chapter



Evaluate imbalanced data classifier

- ❑ Can we use Accuracy to evaluate imbalanced data classifier?
- ❑ Accuracy simply counts the number of errors. If a data set has 2% positive labels and 98% negative labels, a classifier that map all inputs to negative class will get an accuracy of 98%!
- ❑ ROC Curve



Chapter 8. Classification: Basic Concepts

- ❑ Classification: Basic Concepts
- ❑ Decision Tree Induction
- ❑ Bayes Classification Methods
- ❑ Linear Classifier
- ❑ Model Evaluation and Selection
- ❑ Lazy Learning
- ❑ Ensemble Methods
- ❑ Imbalanced Data Sets
- ❑ Summary 

Summary

- Classification: Model construction from a set of training data
- Effective and scalable methods
 - Decision tree induction, Bayes classification methods, linear classifier, ...
 - No single method has been found to be superior over all others for all data sets
- Evaluation metrics: Accuracy, sensitivity, specificity, precision, recall, F measure
- Model evaluation: Holdout, cross-validation, bootstrapping, ROC curves (AUC)
- Improve Classification Accuracy: Bagging, boosting
- Additional concepts on classification: Multiclass classification, semi-supervised classification, active learning, transfer learning, weak supervision

References (1)

- C. Apte and S. Weiss. **Data mining with decision trees and decision rules.** Future Generation Computer Systems, 13, 1997
- P. K. Chan and S. J. Stolfo. **Learning arbiter and combiner trees from partitioned data for scaling machine learning.** KDD'95
- A. J. Dobson. **An Introduction to Generalized Linear Models.** Chapman & Hall, 1990.
- R. O. Duda, P. E. Hart, and D. G. Stork. **Pattern Classification**, 2ed. John Wiley, 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.

References (2)

- ❑ 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. **C4.5: Programs for Machine Learning.** Morgan Kaufmann, 1993.
- ❑ J. R. Quinlan. **Bagging, boosting, and c4.5.** AAAI'96.

References (3)

- 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. Indurkhy. **Predictive Data Mining.** Morgan Kaufmann, 1997
- I. H. Witten and E. Frank. **Data Mining: Practical Machine Learning Tools and Techniques,** 2ed. Morgan Kaufmann, 2005