Data Mining: Concepts and Techniques

— Slides for Textbook —
— Chapter 4 —

Jiawei Han and Micheline Kamber

Intelligent Database Systems Research Lab Simon Fraser University,

Ari Visa, , Institute of Signal Processing

Tampere University of Technology

Chapter 4: Data Mining Primitives, Languages, and System Architectures

- Data mining primitives: What defines a data mining task?
- A data mining query language
- Design graphical user interfaces based on a data mining query language
- Architecture of data mining systems
- Summary

Why Data Mining Primitives and Languages?

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

What Defines a Data Mining Task ?

- Task-relevant data
- Type of knowledge to be mined
- Background knowledge
- Pattern interestingness measurements
- Visualization of discovered patterns

What Defines a Data Mining Task ?

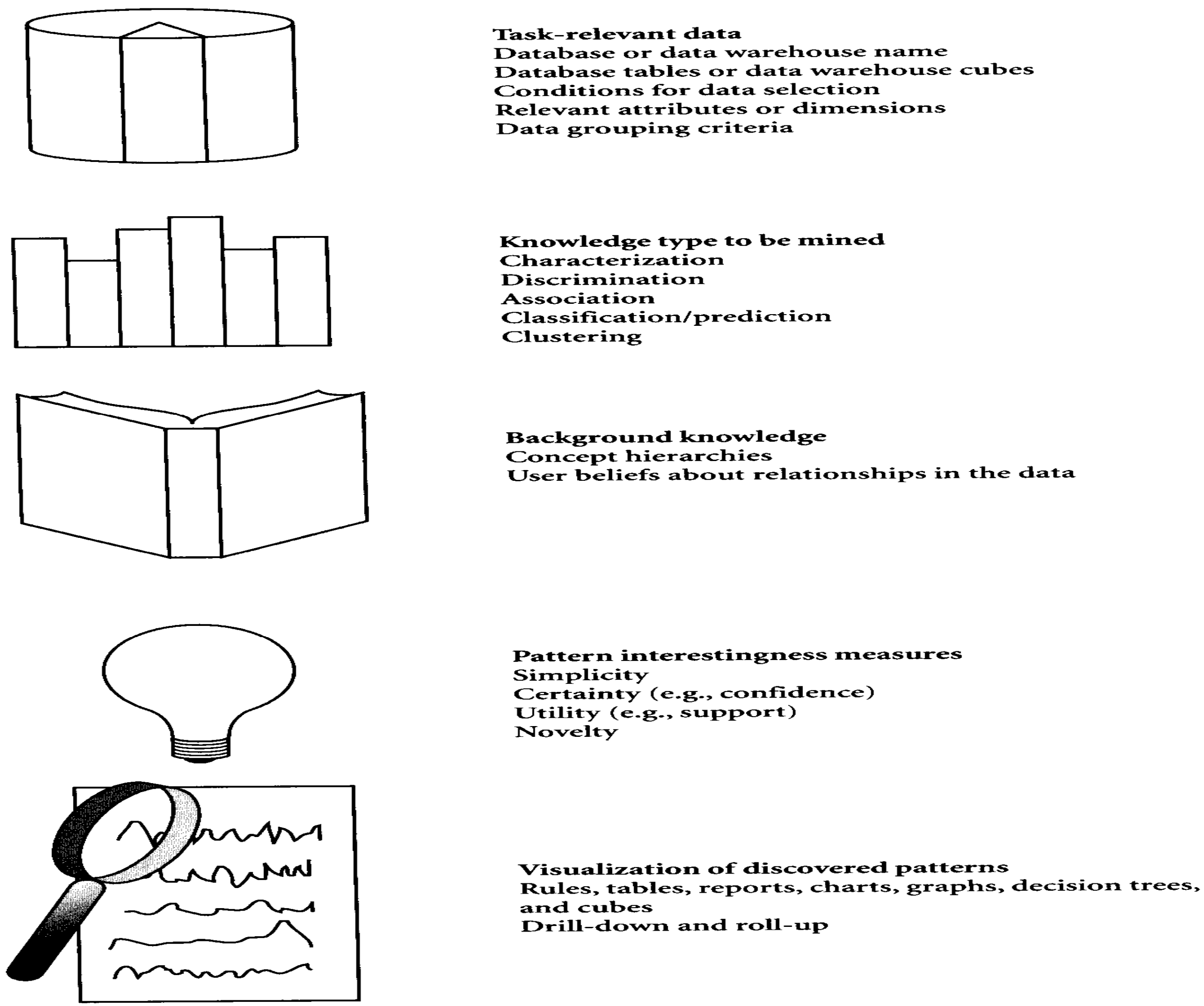


Figure 4.2 Primitives for specifying a data mining task.

Task-Relevant Data (Minable View)

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

Types of knowledge to be mined

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

Background Knowledge: Concept Hierarchies

- Schema hierarchy: a total/partial order among attributes
  – E.g., street < city < province_or_state < country
- Set-grouping hierarchy: organizes values for a given attribute/dimension into groups of constants/ranges
  – E.g., {20-39} = young, {40-59} = middle_aged
- Operation-derived hierarchy: specified by users/experts/systems
  – email address: login-name < department < university < country
- Rule-based hierarchy: defined by set of rules
  – low_profit_margin (X) <= price(X, P1) and cost (X, P2) and (P1 - P2) < $50

Background Knowledge: Concept Hierarchies

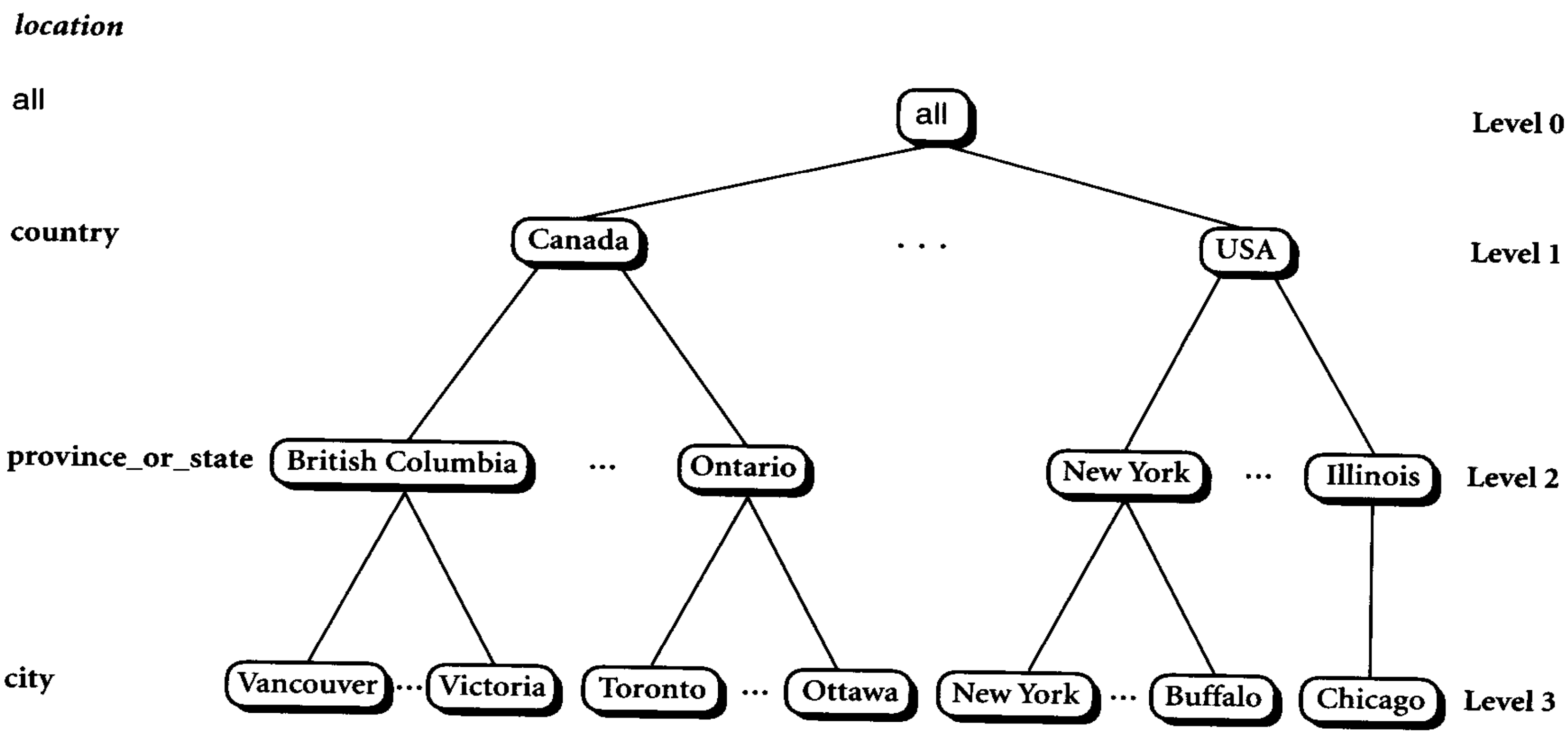


Figure 4.3 A concept hierarchy for the dimension *location*.

Measurements of Pattern Interestingness

- Simplicity

  e.g., (association) rule length, (decision) tree size
- Certainty

  e.g., confidence, P(A|B) = n(A and B)/ n (B), classification reliability or accuracy, certainty factor, rule strength, rule quality, discriminating weight, etc.
- Utility

  potential usefulness, e.g., support (association), noise threshold (description)
- Novelty

  not previously known, surprising (used to remove redundant rules, e.g., Canada vs. Vancouver rule implication support ratio

Visualization of Discovered Patterns

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

Visualization of Discovered Patterns

Rules

age(X, "young") and income(X, "high") => class(X, "A")
age(X, "young") and income(X, "low") => class(X, "B")
age(X, "old") => class(X, "C")

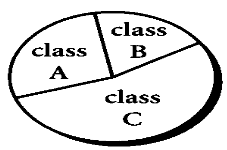
Table

age	income	class	count
young	high	A	1,402
young	low	B	1,038
old	high	C	786
old	low	C	1,374

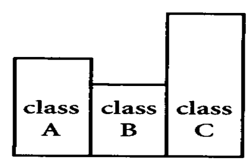
Crosstab

	income		class		
age	high	low	A	B	C
young	1,402	1,038	1,402	1,038	0
old	786	1,374	0	0	2,160
count	2,188	2,412	1,402	1,038	2,160

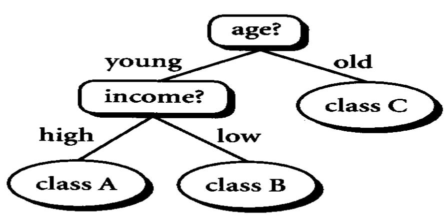
Pie chart



Bar chart



Decision tree



Data cube

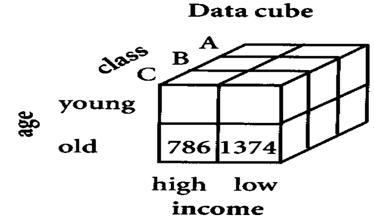


Figure 4.4 Various forms of presenting and visualizing the discovered patterns.

Chapter 4: Data Mining Primitives, Languages, and System Architectures

- Data mining primitives: What defines a data mining task?
- A data mining query language
- Design graphical user interfaces based on a data mining query language
- Architecture of data mining systems
- Summary

A Data Mining Query Language (DMQL)

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

Syntax for DMQL

- Syntax for specification of
  - task-relevant data
  - the kind of knowledge to be mined
  - concept hierarchy specification
  - interestingness measure
  - pattern presentation and visualization
- Putting it all together — a DMQL query

Syntax for task-relevant data specification

- *use database* database_name, or *use data warehouse* data_warehouse_name
- *from relation*(s)/cube(s) [*where* condition]
- *in relevance to* att_or_dim_list
- *order by* order_list
- *group by* grouping_list
- *having* condition

Specification of task-relevant data

Example 4.11 This example shows how to use DMQL to specify the task-relevant data described in Example 4.1 for the mining of associations between items frequently purchased at *AllElectronics* by Canadian customers, with respect to customer *income* and *age*. In addition, the user specifies that she would like the data to be grouped by date. The data are retrieved from a relational database.

```
use database AllElectronics_db
in relevance to I.name, I.price, C.income, C.age
from customer C, item I, purchases P, items_sold S
where I.item_ID = S.item_ID and S.trans_ID = P.trans_ID and P.cust_ID = C.cust_ID
        and C.address = "Canada"
group by P.date
```

□

Syntax for specifying the kind of knowledge to be mined

- Characterization

  Mine_Knowledge_Specification ::=
  mine characteristics **[*as* pattern_name]**
  analyze **measure(s)**

- Discrimination

  Mine_Knowledge_Specification ::=
  mine comparison **[*as* pattern_name]**
  for **target_class** ***where*** **target_condition**
  {*versus* contrast_class_*i* *where* contrast_condition_*i*}
  analyze **measure(s)**

- Association

  Mine_Knowledge_Specification ::=
  mine associations [as pattern_name]

Syntax for specifying the kind of knowledge to be mined (cont.)

- Classification

  Mine_Knowledge_Specification ::=
  mine classification [*as* pattern_name]
  analyze classifying_attribute_or_dimension

- Prediction

  Mine_Knowledge_Specification ::=
  mine prediction [*as* pattern_name]
  analyze prediction_attribute_or_dimension
  {*set* {attribute_or_dimension_*i*= value_*i*}}

Syntax for concept hierarchy specification

- To specify what concept hierarchies to use

  use hierarchy <hierarchy> for <attribute_or_dimension>

- We use different syntax to define different type of hierarchies
  - schema hierarchies

    define hierarchy time_hierarchy on date as [date,month quarter,year]
  - set-grouping hierarchies

    define hierarchy age_hierarchy for age on customer as

    level1: {*young, middle_aged, senior*} < level0: all

    level2: {20, ..., 39} < level1: *young*

    level2: {40, ..., 59} < level1: *middle_aged*

    level2: {60, ..., 89} < level1: *senior*

Syntax for concept hierarchy specification (Cont.)

- operation-derived hierarchies

  define hierarchy age_hierarchy for age on customer as

  {age_category(1), ..., age_category(5)} := cluster(default, age, 5) < all(age)

- rule-based hierarchies

  define hierarchy profit_margin_hierarchy on item as

  level_1: low_profit_margin < level_0: all

  if (price - cost)< $50

  level_1: medium-profit_margin < level_0: all

  if ((price - cost) > $50) and ((price - cost) <= $250))

  level_1: high_profit_margin < level_0: all

  if (price - cost) > $250

Syntax for interestingness measure specification

- Interestingness measures and thresholds can be specified by the user with the statement:

  with **<interest_measure_name>** threshold = **threshold_value**

- **Example:**

  with support threshold **= 0.05**

  with confidence threshold **= 0.7**

Syntax for pattern presentation and visualization specification

- We have syntax which allows users to specify the display of discovered patterns in one or more forms

  display as <**result_form>**

- To facilitate interactive viewing at different concept level, the following syntax is defined:

Multilevel_Manipulation ::= *roll up on* attribute_or_dimension
| *drill down on* attribute_or_dimension
| *add* attribute_or_dimension
| *drop* attribute_or_dimension

Putting it all together: the full specification of a DMQL query

use database AllElectronics_db
use hierarchy location_hierarchy for B.address
mine characteristics as customerPurchasing
analyze count%
in relevance to C.age, I.type, I.place_made
from customer C, item I, purchases P, items_sold S, works_at W, branch
where I.item_ID = S.item_ID and S.trans_ID = P.trans_ID
    and P.cust_ID = C.cust_ID and P.method_paid = ``AmEx''
    and P.empl_ID = W.empl_ID and W.branch_ID = B.branch_ID and B.address = ``Canada'' and I.price >= 100
with noise threshold = 0.05
display as table

Other Data Mining Languages & Standardization Efforts

- Association rule language specifications
  - MSQL (Imielinski & Virmani'99)
  - MineRule (Meo Psaila and Ceri'96)
  - Query flocks based on Datalog syntax (Tsur et al'98)
- OLEDB for DM (Microsoft'2000)
  - Based on OLE, OLE DB, OLE DB for OLAP
  - Integrating DBMS, data warehouse and data mining
- CRISP-DM (CRoss-Industry Standard Process for Data Mining)
  - Providing a platform and process structure for effective data mining
  - Emphasizing on deploying data mining technology to solve business problems

Chapter 4: Data Mining Primitives, Languages, and System Architectures

- Data mining primitives: What defines a data mining task?
- A data mining query language
- Design graphical user interfaces based on a data mining query language
- Architecture of data mining systems
- Summary

Designing Graphical User Interfaces based on a data mining query language

- What tasks should be considered in the design GUIs based on a data mining query language?
  - Data collection and data mining query composition
  - Presentation of discovered patterns
  - Hierarchy specification and manipulation
  - Manipulation of data mining primitives
  - Interactive multilevel mining
  - Other miscellaneous information

Chapter 4: Data Mining Primitives, Languages, and System Architectures

- Data mining primitives: What defines a data mining task?
- A data mining query language
- Design graphical user interfaces based on a data mining query language
- Architecture of data mining systems
- Summary

Data Mining System Architectures

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

Chapter 4: Data Mining Primitives, Languages, and System Architectures

- Data mining primitives: What defines a data mining task?
- A data mining query language
- Design graphical user interfaces based on a data mining query language
- Architecture of data mining systems
- Summary

Summary

- Five primitives for specification of a data mining task
  - task-relevant data
  - kind of knowledge to be mined
  - background knowledge
  - interestingness measures
  - knowledge presentation and visualization techniques to be used for displaying the discovered patterns
- Data mining query languages
  - DMQL, MS/OLEDB for DM, etc.
- Data mining system architecture
  - No coupling, loose coupling, semi-tight coupling, tight coupling

References

- E. Baralis and G. Psaila. Designing templates for mining association rules. Journal of Intelligent Information Systems, 9:7-32, 1997.
- Microsoft Corp., OLEDB for Data Mining, version 1.0, http://www.microsoft.com/data/oledb/dm, Aug. 2000.
- J. Han, Y. Fu, W. Wang, K. Koperski, and O. R. Zaiane, “DMQL: A Data Mining Query Language for Relational Databases”, DMKD'96, Montreal, Canada, June 1996.
- T. Imielinski and A. Virmani. MSQL: A query language for database mining. Data Mining and Knowledge Discovery, 3:373-408, 1999.
- M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM’94, Gaithersburg, Maryland, Nov. 1994.
- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96, pages 122-133, Bombay, India, Sept. 1996.
- A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. IEEE Trans. on Knowledge and Data Engineering, 8:970-974, Dec. 1996.
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98, Seattle, Washington, June 1998.
- D. Tsur, J. D. Ullman, S. Abitboul, C. Clifton, R. Motwani, and S. Nestorov. Query flocks: A generalization of association-rule mining. SIGMOD'98, Seattle, Washington, June 1998.