Data Mining Primitives & DMQL

Dr.M. Venkatesan
Associate Professor
School of Computing Science and Engineering
VIT University

Outline

- 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

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

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
- Outlier analysis
- Other data mining tasks

Background Knowledge: Concept Hierarchies

- Schema hierarchy
  - E.g., street < city < province_or_state < country
- Set-grouping hierarchy
  - E.g., {20-39} = young, {40-59} = middle_aged
- Operation-derived hierarchy
  - email address: Robert.Gyorodi@ul.ie
    login-name < department < university < country
- Rule-based hierarchy
  - low_profit_margin (X) <= price(X, P_1) and cost (X, P_2) and (P_1 - P_2) < $50

Measurements of Pattern Interestingness

- Simplicity

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

- Certainty

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

- Utility

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

- Novelty

  not previously known, surprising (used to remove redundant rules, e.g., 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 perspectives to data
- Different kinds of knowledge require different representation: association, classification, clustering, etc.

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: Specification of Task-Relevant Data

- *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: 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)**

  E.g. mine comparison as purchaseGroups
  for bigSpenders where avg(I.price) >= $100
  versus budgetSpenders where avg(I.price) < $100
  analyze count

Syntax: Kind of knowledge to Be Mined

- Association

  Mine_Knowledge_Specification **::=**
  mine associations **[*as* pattern_name]**
  [*matching* <metapattern>]

  E.g. mine associations as buyingHabits
  matching P(X:custom, W) ^ Q(X, Y)=>buys(X, Z)

- Classification

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

- Other Patterns

  clustering, outlier analysis, prediction …

Syntax: 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: Concept Hierarchy Specification

- 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

Specification of Interestingness Measures

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

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

- **Example:**

  with support threshold **= 0.05**

  with confidence threshold **= 0.7**

Specification of Pattern Presentation

- Specify the display of discovered patterns

  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: 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

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

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.

Today's lecture …

- 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