Why preprocess the data?

Data in the real world is:

- incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
- noisy: containing errors or outliers
- inconsistent: containing discrepancies in codes or names

No quality data, no quality mining results!

- Quality decisions must be based on quality data
- Data warehouse needs consistent integration of quality data

Multi-Dimensional Measure of Data Quality

A well-accepted multidimensional view:

- Accuracy
- Completeness
- Consistency
- Timeliness
- Believability
- Value added
- Interpretability
- Accessibility

Broad categories:

- intrinsic, contextual, representational, and accessibility.

Major Tasks in Data Preprocessing
Data cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies

Data integration

- Integration of multiple databases, data cubes, or files

Data transformation

- Normalization and aggregation

Data reduction

- Obtains reduced representation in volume but produces the same or similar analytical results

Data discretization

- Part of data reduction but with particular importance, especially for numerical data

Forms of data preprocessing
Data Cleaning

- Data cleaning tasks
  - Fill in missing values
  - Identify outliers and smooth out noisy data
  - Correct inconsistent data

Missing Data

- Data is not always available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data

- Missing data may be due to
  - equipment malfunction
  - inconsistent with other recorded data and thus deleted
  - data not entered due to misunderstanding
  - certain data may not be considered important at the time of entry
  - not register history or changes of the data

Missing data may need to be inferred

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing
- Fill in the missing value manually
- Use a global constant to fill in the missing value: ex. “unknown”
- Use the attribute mean to fill in the missing value
- Use the attribute mean for all samples belonging to the same class to fill in the missing value
Use the most probable value to fill in the missing value: inference-based such as Bayesian formula or decision tree

Noisy Data

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

How to Handle Noisy Data?

- Binning method:
  - first sort data and partition into (equal-frequency) bins
  - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries
- Clustering
  - detect and remove outliers
- Regression
  - smooth by fitting the data to a regression functions – linear regression

Simple Discretization Methods: Binning
Equal-width (distance) partitioning:

- It 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:

- It divides the range into \( N \) intervals, each containing approximately same number of samples
- Good data scaling

Managing categorical attributes can be tricky

**Binning Methods for Data Smoothing**

Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* Partition into (equi-depth) bins:
  - Bin 1: 4, 8, 9, 15
  - Bin 2: 21, 21, 24, 25
  - Bin 3: 26, 28, 29, 34

* Smoothing by bin means:
  - Bin 1: 9, 9, 9, 9
  - Bin 2: 23, 23, 23, 23
  - Bin 3: 29, 29, 29, 29

* Smoothing by bin boundaries:
  - Bin 1: 4, 4, 4, 15
  - Bin 2: 21, 21, 25, 25
Cluster Analysis

Regression

Data integration:

- Data integration:
  - combines data from multiple sources into a coherent store

- Schema integration
integrate metadata from different sources

Entity identification problem: identify real world entities from multiple data sources, e.g., A.cust-id = B.cust-#

Detecting and resolving data value conflicts

- for the same real world entity, attribute values from different sources are different
- possible reasons: different representations, different scales, e.g., metric vs. British units

Handling Redundant Data in Data Integration

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

Data Transformation

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

- min-max normalization

\[ v' = \frac{v - \text{min}_i}{\text{max}_i - \text{min}_i} (\text{new}_\text{max} - \text{new}_\text{min}) + \text{new}_\text{min} \]

- z-score normalization

\[ v' = \frac{v - \text{mean}_i}{\text{stand}_\text{dev}_i} \]

- normalization by decimal scaling

\[ v' = \frac{v}{10^j} \]

Data Reduction

- Warehouse may store terabytes of data: Complex data analysis/mining may take a very long time to run on the complete data set

- Data reduction

  - Obtains a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results

Data Reduction Strategies

- Data reduction strategies

  - Data cube aggregation
  
  - Attribute subset selection
  
  - Dimensionality reduction
  
  - Numerosity reduction
  
  - Discretization and concept hierarchy generation

Data Cube Aggregation
The lowest level of a data cube

- 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

Dimensionality Reduction

Feature selection (attribute subset selection):

- Select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
- reduce # of patterns in the patterns, easier to understand

Heuristic methods

- step-wise forward selection
- step-wise backward elimination
- combining forward selection and backward elimination
- decision-tree induction

Wavelet Transforms

Discrete wavelet transform (DWT): linear signal processing

Compressed approximation: store only a small fraction of the strongest of the wavelet coefficients

Similar to discrete Fourier transform (DFT), but better lossy compression, localized in space

Method:
- Length, L, must be an integer power of 2 (padding with 0s, when necessary)
- Each transform has 2 functions: smoothing, difference
- Applies to pairs of data, resulting in two set of data of length L/2
- Applies two functions recursively, until reaches the desired length

---

**Principal Component Analysis**

- Given \( N \) data vectors from \( k \)-dimensions, find \( c \leq k \) orthogonal vectors that can be best used to represent data
- The original data set is reduced to one consisting of \( N \) data vectors on \( c \) principal components (reduced dimensions)
- Each data vector is a linear combination of the \( c \) principal component vectors
- Works for numeric data only
- Used when the number of dimensions is large
Attribute subset selection

- Attribute subset selection reduces the data set size by removing irrelevant or redundant attributes.
- Goal is to find the minimum set of attributes.
- Uses basic heuristic methods of attribute selection.

Heuristic Selection Methods

- There are $2^d$ possible sub-features of $d$ features.
- Several heuristic selection methods:
  - Stepwise forward selection
  - Stepwise backward elimination
  - Combination of forward selection and backward elimination
  - Decision tree induction

Example of Decision Tree Induction

Initial attribute set:

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

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

- **Parametric methods**
  - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
  - 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

Regression and Log-Linear Models

- **Linear regression**: Data are modeled to fit a straight line
  - Often uses the least-square method to fit the line

- **Multiple regression**: allows a response variable \( Y \) to be modeled as a linear function of multidimensional feature vector

- **Log-linear model**: approximates discrete multidimensional probability distributions

Regress Analysis and Log-Linear Models

- **Linear regression**: \( Y = \alpha + \beta X \)
  - Two parameters, \( \alpha \) and \( \beta \) specify the line and are to be estimated by using the data at hand.
  - using the least squares criterion to the known values of \( Y_1, Y_2, ..., X_1, X_2, ... \)

- **Multiple regression**: \( Y = b_0 + b_1 X_1 + b_2 X_2 \)
  - Many nonlinear functions can be transformed into the above.

- **Log-linear models**: 
The multi-way table of joint probabilities is approximated by a product of lower-order tables.

Probability: $p(a, b, c, d) = \alpha a \beta b \gamma c \delta d$

### Histograms

- A popular data reduction technique
- Divide data into buckets and store average (sum) for each bucket
- Can be constructed optimally in one dimension using dynamic programming

Related to quantization problems

### Clustering

- Partition data set into clusters, and one can store cluster representation 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.

### Sampling

- Allows a large data set to be represented by a much smaller of the data.
- Let a large data set $D$, contains $N$ tuples.
Methods to reduce data set D:

- Simple random sample without replacement (SRSWOR)
- Simple random sample with replacement (SRSWR)
- Cluster sample
- Straight sample

Discretization

Three types of attributes:

- Nominal — values from an unordered set
- Ordinal — values from an ordered set
- Continuous — real numbers

Discretization: divide the range of a continuous attribute into intervals

- Some classification algorithms only accept categorical attributes.
- Reduce data size by discretization
- Prepare for further analysis

Discretization by intuitive partitioning

3-4-5 rule can be used to segment numeric data into
relatively uniform, "natural" intervals.

* If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equal-width intervals

* If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 intervals

* If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 intervals

Concept hierarchy generation for categorical data

- Specification of a partial ordering of attributes explicitly at the schema level by users or experts
- Specification of a portion of a hierarchy by explicit data grouping
- Specification of a set of attributes, but not of their partial ordering
- Specification of only a partial set of attributes

Specification of a set of attributes

Concept hierarchy can be automatically generated based on the number of distinct values per attribute in the given attribute set. The attribute with the most distinct values is placed at the lowest level of the hierarchy.
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**
  - street < city < province_or_state < country
- **Set-grouping hierarchy**
  - \{20-39\} = young, \{40-59\} = middle_aged
- **Operation-derived hierarchy**
  - email address: login-name < department < university < country
Rule-based hierarchy

- low_profit_margin (X) <= price(X, P1) and cost (X, P2) and (P1 - P2) < $50

Measurements of Pattern Interestingness

- Simplicity
  
  association rule length, decision tree size

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

- Utility
  
  potential usefulness, support (association), noise threshold (description)

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

Visualization of Discovered Patterns

- Different backgrounds/usages may require different forms of representation
  
  - rules, tables, cross tabs, pie/bar chart

- 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

A data mining query language
Motivation

- A DMQL can provide the ability to support ad-hoc and interactive data mining
- By providing a standardized language like SQL
  - 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

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

— 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

Syntax for specifying the kind of knowledge to be mined

- **Characterization**
  
  \[
  \text{Mine\_Knowledge\_Specification} ::= \\
  \text{mine characteristics [as pattern\_name]} \\
  \text{analyze measure(s)}
  \]

- **Discrimination**
  
  \[
  \text{Mine\_Knowledge\_Specification} ::= \\
  \text{mine comparison [as pattern\_name]} \\
  \text{for target\_class where target\_condition} \\
  \{\text{versus contrast\_class\_i where contrast\_condition\_i}\} \\
  \text{analyze measure(s)}
  \]

- **Association**
  
  \[
  \text{Mine\_Knowledge\_Specification} ::= \\
  \text{mine associations [as pattern\_name]}
  \]
  
  \[
  \quad \text{Classification}
  \]

  \[
  \text{Mine\_Knowledge\_Specification} ::= \\
  \text{mine classification [as pattern\_name]} \\
  \text{analyze classifying\_attribute\_or\_dimension}
  \]
  
  \[
  \quad \text{Prediction}
  \]

  \[
  \text{Mine\_Knowledge\_Specification} ::= \\
  \text{mine prediction [as pattern\_name]} \\
  \text{analyze prediction\_attribute\_or\_dimension} \\
  \{\text{set [attribute\_or\_dimension\_i= value\_i]}\}
  \]

Syntax for concept hierarchy specification

- **To specify what concept hierarchies to use**
use hierarchy `<hierarchy>` for `<attribute_or_dimension>`

- 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`
  - operation-derived hierarchies
    define hierarchy `age_hierarchy` for `age` on `customer` as
    ```
    \{age\text{-}category(1), \ldots, age\text{-}category(5)} := \text{cluster}(\text{default, age, 5}) < \text{all(age)}
    ```
  - rule-based hierarchies
    define hierarchy `profit_margin_hierarchy` on `item` as
    - level1: `low_profit_margin < level0: all`
      - if `(price - cost) < $50`
    - level1: `medium_profit_margin < level0: all`
      - if `((price - cost) > $50) \land ((price - cost) \leq $250))`
    - level1: `high_profit_margin < level0: 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

- 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

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
Design 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

Architecture of data mining systems

Coupling data mining system with DB/DW system

- No coupling—flat file processing,
- 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—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