Data Mining

Module 1

Introduction: Fundamentals of data mining

- Data mining is the process of discovering these patterns among the data and is therefore also known as Knowledge Discovery from Data (KDD).

The knowledge discovery process is shown in Figure 1.4 as an iterative sequence of the following steps:

1. Data cleaning (to remove noise and inconsistent data)
2. Data integration (where multiple data sources may be combined)
3. Data selection (where data relevant to the analysis task are retrieved from the database)
4. Data transformation (where data are transformed and consolidated into forms appropriate for mining by performing summary or aggregation operations)
5. Data mining (an essential process where intelligent methods are applied to extract data patterns)
6. Pattern evaluation (to identify the truly interesting patterns representing knowledge based on interestingness measures)
7. Knowledge presentation (where visualization and knowledge representation techniques are used to present mined knowledge to users)

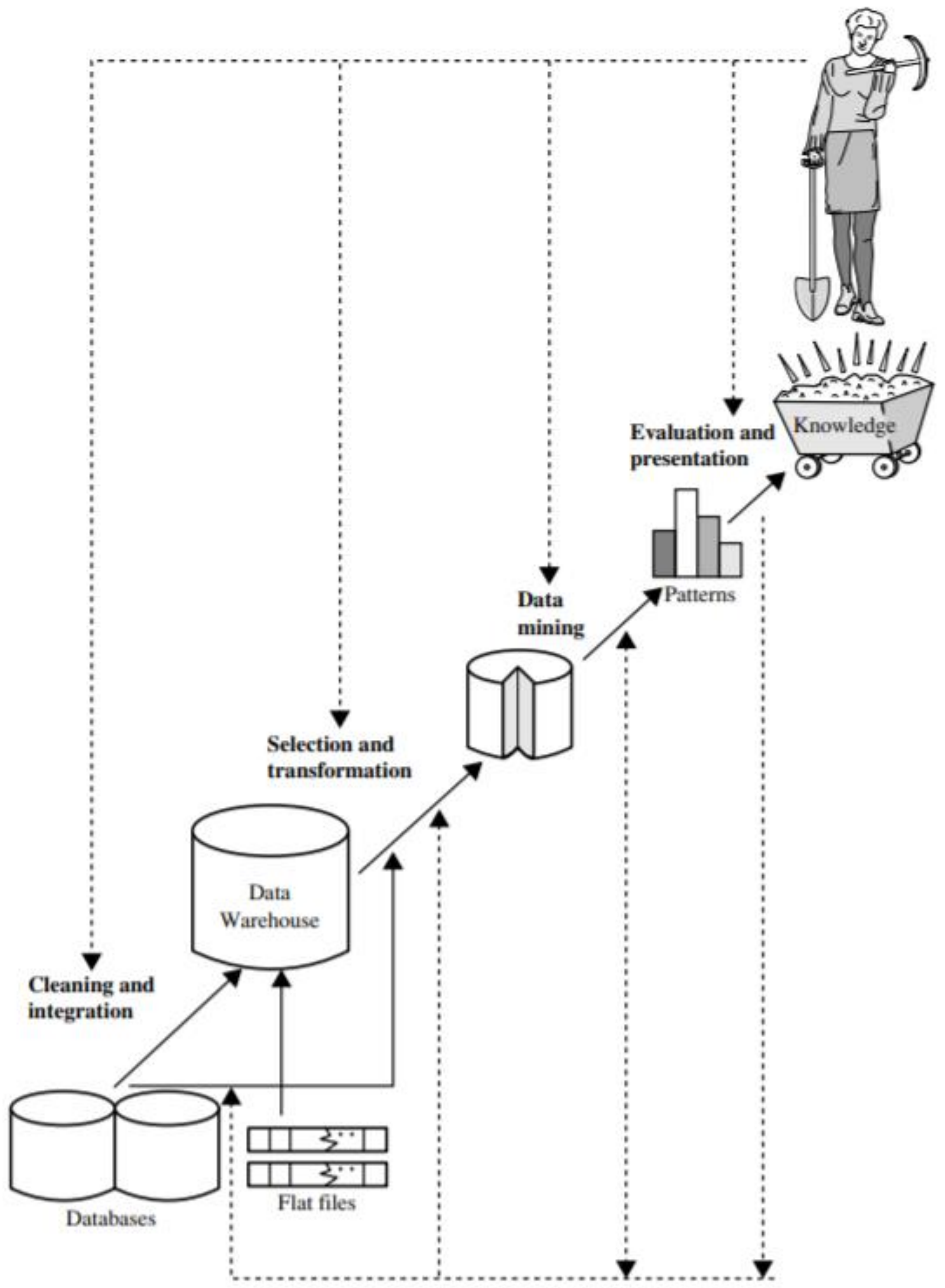


Figure 1.4 Data mining as a step in the process of knowledge discovery.

Data Mining Functionalities

- Data mining functionalities are used to specify the kind of patterns to be found in data mining tasks. In general, data mining tasks can be classified into two categories: descriptive and predictive.
- Descriptive mining tasks characterize the general properties of the data in the database.
- Predictive mining tasks perform inference on the current data in order to make predictions.

Classification of Data Mining systems

Data mining is an interdisciplinary field, the confluence of a set of disciplines, including database systems, statistics, machine learning, visualization, and information science.

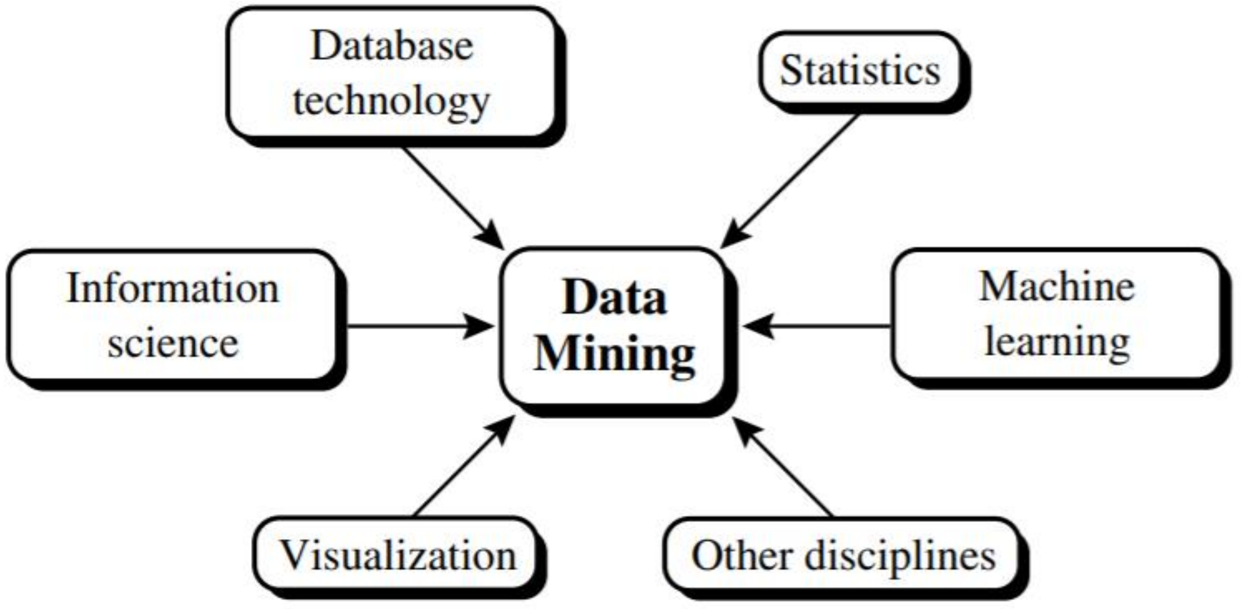


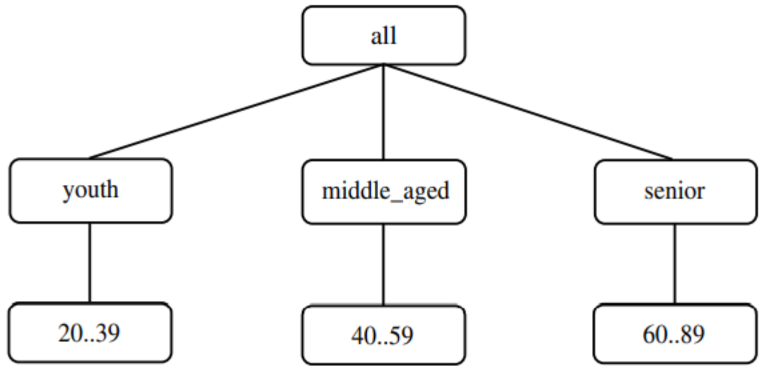
Figure 1. Data mining as a confluence of multiple disciplines.

Data Mining Task Primitives

- A data mining task can be specified in the form of a data mining query, which is input to the data mining system. A data mining query is defined in terms of data mining task primitives. These primitives allow the user to interactively communicate with the data mining system during discovery in order to direct the mining process, or examine the findings from different angles or depths.

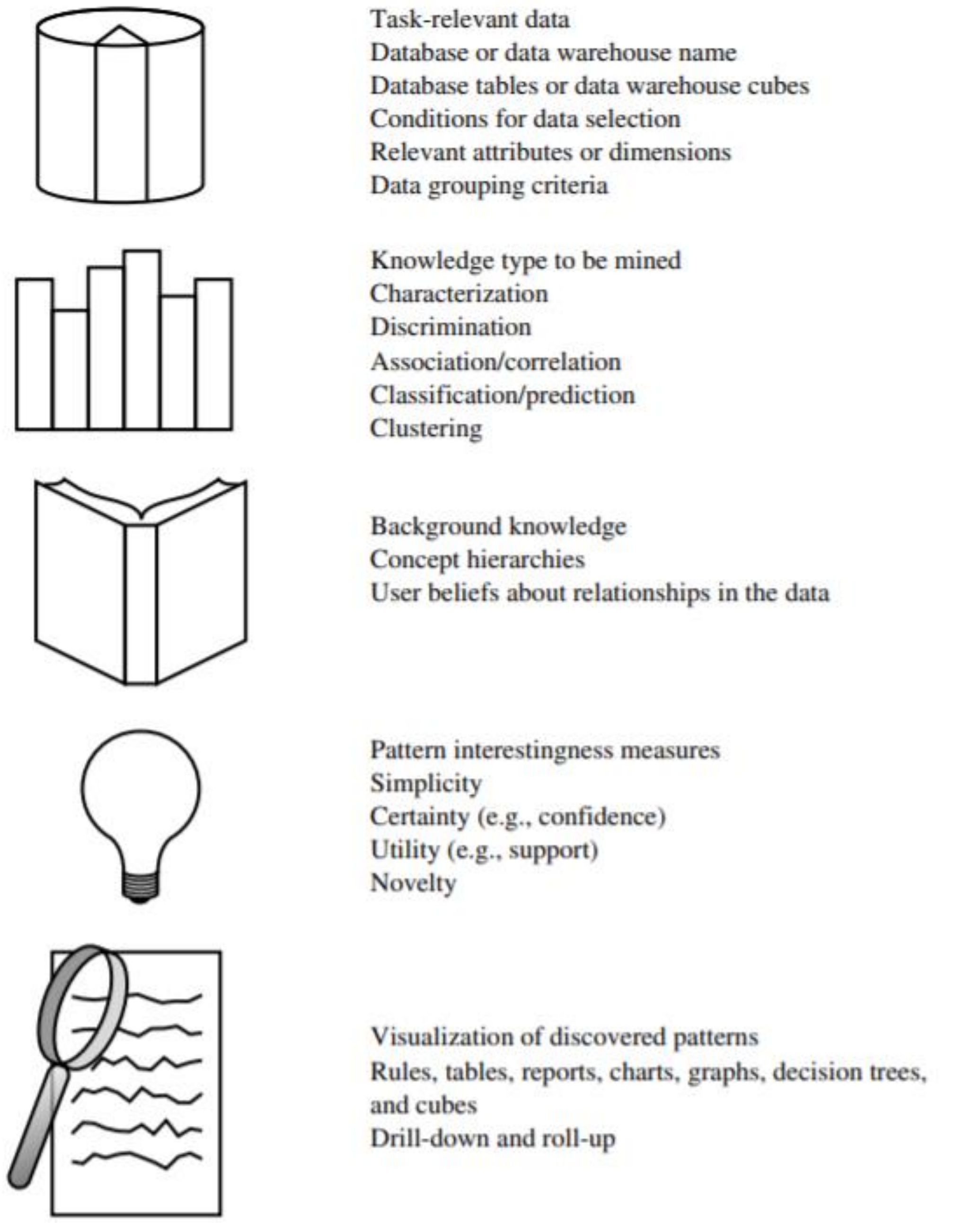
The data mining primitives specify the following, as illustrated in Figure 2.

- The set of *task-relevant data* to be mined: This specifies the portions of the database or the set of data in which the user is interested. This includes the database attributes or data warehouse dimensions of interest (referred to as the relevant attributes or dimensions).
- The *kind of knowledge* to be mined: This specifies the data mining functions to be performed, such as characterization, discrimination, association or correlation analysis, classification, prediction, clustering, outlier analysis, or evolution analysis.
- The *background knowledge* to be used in the discovery process: This knowledge about the domain to be mined is useful for guiding the knowledge discovery process and for evaluating the patterns found. Concept hierarchies are a popular form of background knowledge, which allow data to be mined at multiple levels of abstraction. An example of a concept hierarchy for the attribute (or dimension) age is shown in Figure. User beliefs regarding relationships in the data are another form of background knowledge.



A concept hierarchy for the attribute (or dimension) *age*. The root node represents the most general abstraction level, denoted as all.

- The *interestingness measures and thresholds* for pattern evaluation: They may be used to guide the mining process or, after discovery, to evaluate the discovered patterns. Different kinds of knowledge may have different interestingness measures. For example, interestingness measures for association rules include support and confidence. Rules whose support and confidence values are below user-specified thresholds are considered uninteresting.
- The expected *representation for visualizing* the discovered patterns: This refers to the form in which discovered patterns are to be displayed, which may include rules, tables, charts, graphs, decision trees, and cubes.



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

Knowledge type to be mined
Characterization
Discrimination
Association/correlation
Classification/prediction
Clustering

Background knowledge
Concept hierarchies
User beliefs about relationships in the data

Pattern interestingness measures
Simplicity
Certainty (e.g., confidence)
Utility (e.g., support)
Novelty

Visualization of discovered patterns
Rules, tables, reports, charts, graphs, decision trees, and cubes
Drill-down and roll-up

Figure 2. Primitives for specifying a data mining task.

- Data characterization is a summarization of the general characteristics or features of a target class of data. The data corresponding to the user-specified class are typically collected by a database query.

  For example, to study the characteristics of software products whose sales increased by 10% in the last year, the data related to such products can be collected by executing an SQL query.

- Data discrimination is a comparison of the general features of target class data objects with the general features of objects from one or a set of contrasting classes. The target and contrasting classes can be specified by the user, and the corresponding data objects retrieved through database queries.

  For example, the user may like to compare the general features of software products whose sales increased by 10% in the last year with those whose sales decreased by at least 30% during the same period. The methods used for data discrimination are similar to those used for data characterization.

Integration of a Data Mining System with a Database or a Data Warehouse System

- when a DM system works in an environment that requires it to communicate with other information system components, such as DB and DW systems, possible integration schemes include no coupling, loose coupling, semitight coupling, and tight coupling. We examine each of these schemes, as follows:
- No coupling: No coupling means that a DM system will not utilize any function of a DB or DW system. It may fetch data from a particular source (such as a file system), process data using some data mining algorithms, and then store the mining results in another file.

- Loose coupling: Loose coupling means that a DM system will use some facilities of a DB or DW system, fetching data from a data repository managed by these systems, performing data mining, and then storing the mining results either in a file or in a designated place in a database or data warehouse.
- Semitight coupling: Semitight coupling means that besides linking a DM system to a DB/DW system, efficient implementations of a few essential data mining primitives (identified by the analysis of frequently encountered data mining functions) can be provided in the DB/DW system.
- Tight coupling: Tight coupling means that a DM system is smoothly integrated into the DB/DW system. The data mining subsystem is treated as one functional component of an information system.

- With this analysis, it is easy to see that a data mining system should be coupled with a DB/DW system. Loose coupling, though not efficient, is better than no coupling because it uses both data and system facilities of a DB/DW system. Tight coupling is highly desirable, but its implementation is nontrivial and more research is needed in this area. Semitight coupling is a compromise between loose and tight coupling. It is important to identify commonly used data mining primitives and provide efficient implementations of such primitives in DB or DW systems

Major issues in Data Mining

The major issues in data mining regarding mining methodology, user interaction, performance, and diverse data types. These issues are introduced below:

Mining methodology and user interaction issues:

- Mining different kinds of knowledge in databases
- Interactive mining of knowledge at multiple levels of abstraction
- Incorporation of background knowledge
- Data mining query languages and ad hoc data mining
- Presentation and visualization of data mining results
- Handling noisy or incomplete data
- Pattern evaluation—the interestingness problem

Performance issues: These include efficiency, scalability, and parallelization of data mining algorithms.

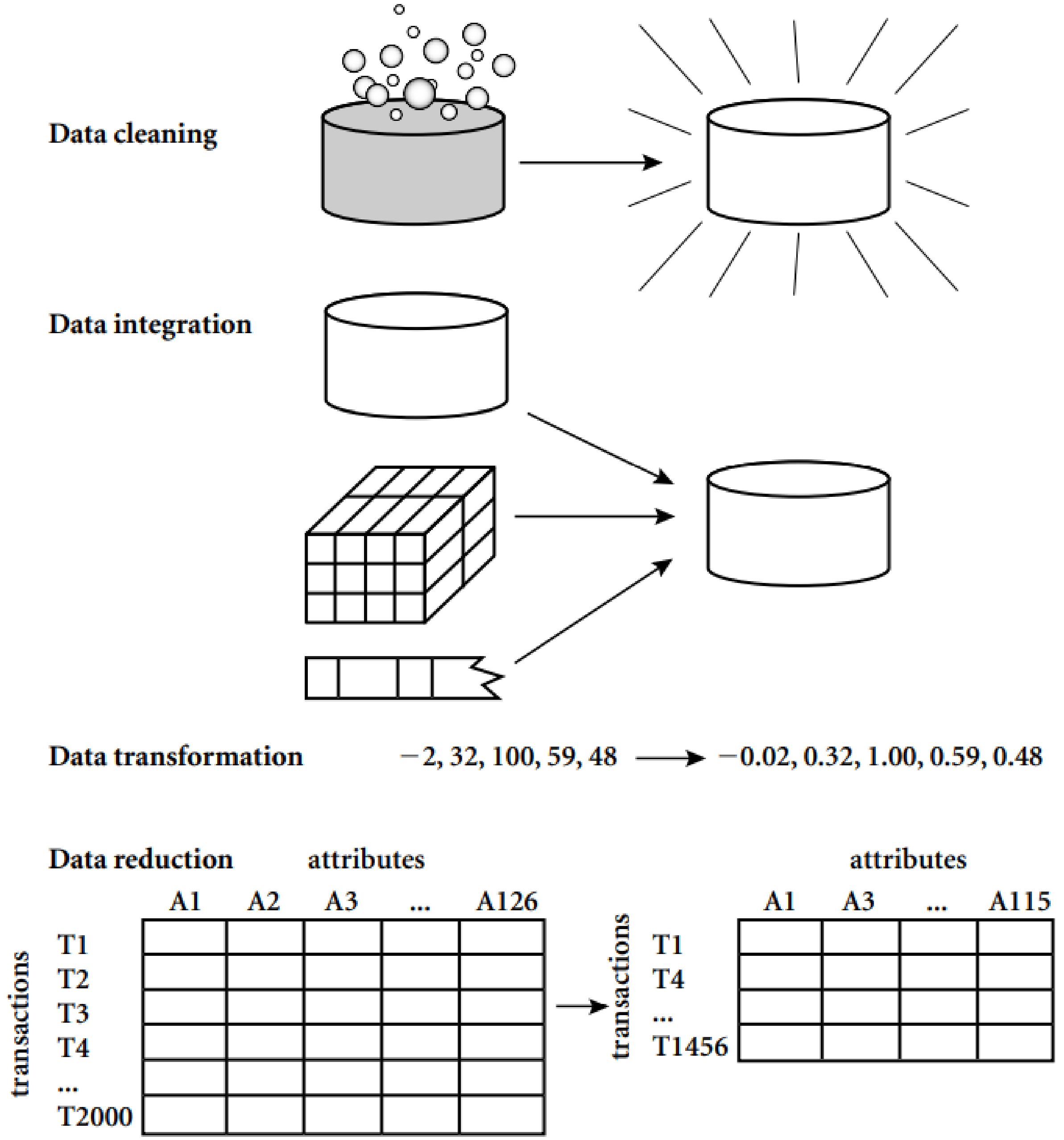
- Efficiency and scalability of data mining algorithms: The running time of a data mining algorithm must be predictable and acceptable in large databases.
- Parallel, distributed, and incremental mining algorithms: Due to huge size of data, algorithms divide the data into partitions, which are processed in parallel. The results from the partitions are then merged. Moreover, the high cost of some data mining processes promotes the need for incremental data mining algorithms that incorporate database updates without having to mine the entire data again "from scratch."

Issues relating to the diversity of database types:

- Handling of relational and complex types of data: other databases may contain complex data objects, hypertext and multimedia data, spatial data, temporal data, or transaction data. It is unrealistic to expect one system to mine all kinds of data, given the diversity of data types and different goals of data mining.
- Mining information from heterogeneous databases and global information systems: The discovery of knowledge from different sources of structured, semistructured, or unstructured data with diverse data semantics poses great challenges to data mining.

Data Preprocessing: Need for Preprocessing the Data

- The data you wish to analyze by data mining techniques are incomplete (lacking attribute values or certain attributes of interest, or containing only aggregate data), noisy (containing errors, or outlier values that deviate from the expected), and inconsistent (e.g., containing discrepancies in the department codes used to categorize items).



Forms of data preprocessing.

Data Cleaning

Real-world data tend to be incomplete, noisy, and inconsistent. Data cleaning (or data cleansing) routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data. In this section, you will study basic methods for data cleaning. Section 2.3.1 looks at ways of handling missing values. Section 2.3.2 explains data smoothing techniques. Section 2.3.3 discusses approaches to data cleaning as a process.

2.3.1 **Missing Values**

Imagine that you need to analyze AllElectronics sales and customer data. You note that many tuples have no recorded value for several attributes, such as customer income. How can you go about filling in the missing values for this attribute? Let's look at the following methods:

1. Ignore the tuple: This is usually done when the class label is missing (assuming the mining task involves classification). This method is not very effective, unless the tuple contains several attributes with missing values. It is especially poor when the percentage of missing values per attribute varies considerably

2. Fill in the missing value manually: In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.

3. Use a global constant to fill in the missing value: Replace all missing attribute values by the same constant, such as a label like “Unknown” or $-\infty$. If missing values are replaced by, say, “Unknown,” then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of “Unknown.” Hence, although this method is simple, it is not foolproof.

4. Use the attribute mean to fill in the missing value: For example, suppose that the average income of AllElectronics customers is $56,000. Use this value to replace the missing value for income.

5. Use the attribute mean for all samples belonging to the same class as the given tuple: For example, if classifying customers according to credit risk, replace the missing value with the average income value for customers in the same credit risk category as that of the given tuple.

6. Use the most probable value to fill in the missing value: This may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction. For example, using the other customer attributes in your data set, you may construct a decision tree to predict the missing values for income.

Noisy Data

“What is noise?” Noise is a random error or variance in a measured variable. Given a numerical attribute such as, say, price, how can we “smooth” out the data to remove the noise? Let’s look at the following data smoothing techniques:

1. Binning: Binning methods smooth a sorted data value by consulting its “neighborhood,” that is, the values around it. The sorted values are distributed into a number of “buckets,” or bins. Because binning methods consult the neighborhood of values, they perform local smoothing. Figure 2.11 illustrates some binning techniques. In this example, the data for price are first sorted and then partitioned into equal-frequency bins of size 3 (i.e., each bin contains three values). In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is 9. Therefore, each original value in this bin is replaced by the value 9. Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value. In general, the larger the width, the greater the effect of the 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

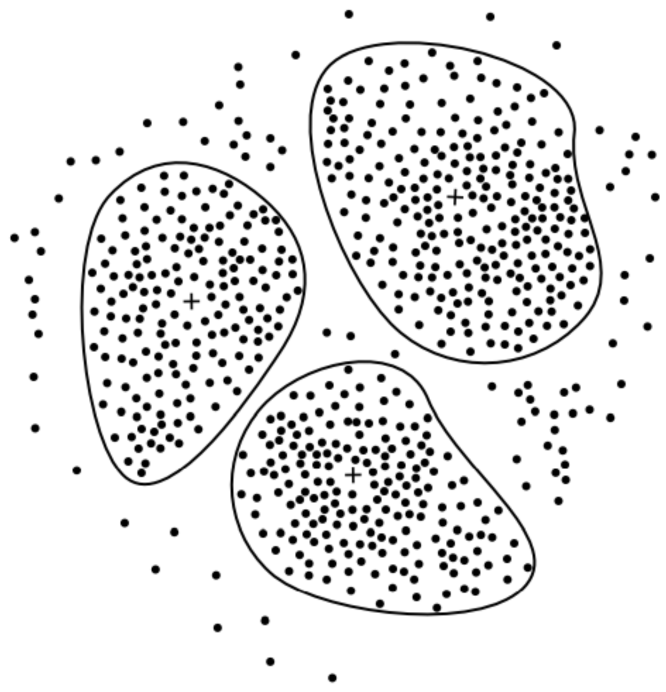
Smoothing by bin boundaries:

Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

Binning methods for data smoothing.

2. Regression: Data can be smoothed by fitting the data to a function, such as with regression. Linear regression involves finding the "best" line to fit two attributes (or variables), so that one attribute can be used to predict the other. Multiple linear regression is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

3. Clustering: Outliers may be detected by clustering, where similar values are organized into groups, or "clusters." Intuitively, values that fall outside of the set of clusters may be considered Outliers.



A 2-D plot of customer data with respect to customer locations in a city, showing three data clusters. Each cluster centroid is marked with a "+", representing the average point in space for that cluster. Outliers may be detected as values that fall outside of the sets of clusters.

Data Integration

- It is likely that your data analysis task will involve data integration, which combines data from multiple sources into a coherent data store, as in data warehousing. These sources may include multiple databases, data cubes, or flat files.
- There are a number of issues to consider during data integration. Schema integration and object matching can be tricky. How can equivalent real-world entities from multiple data sources be matched up? This is referred to as the entity identification problem. For example, how can the data analyst or the computer be sure that customer id in one database and cust_number in another refer to the same attribute? Examples of metadata for each attribute include the name, meaning, data type, and range of values permitted for the attribute, and null rules for handling blank, zero, or null values (Section 2.3). Such metadata can be used to help avoid errors in schema integration. The metadata may also be used to help transform the data (e.g., where data codes for pay type in one database may be "H" and "S", and 1 and 2 in another). Hence, this step also relates to data cleaning, as described earlier.
- Redundancy is another important issue. An attribute (such as annual revenue, for instance) may be redundant if it can be "derived" from another attribute or set of attributes. Inconsistencies in attribute or dimension naming can also cause redundancies in the resulting data set.

Data Transformation

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

- Smoothing, which works to remove noise from the data. Such techniques include binning, regression, and clustering.
- Aggregation, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for analysis of the data at multiple granularities.
- Generalization of the data, where low-level or “primitive” (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, categorical attributes, like street, can be generalized to higher-level concepts, like city or country. Similarly, values for numerical attributes, like age, may be mapped to higher-level concepts, like youth, middle-aged, and senior.

- Normalization, where the attribute data are scaled so as to fall within a small specified range, such as −1.0 to 1.0, or 0.0 to 1.0.
- Attribute construction (or feature construction), where new attributes are constructed and added from the given set of attributes to help the mining process.

Data Reduction

Imagine that you have selected data from the AllElectronics data warehouse for analysis. The data set will likely be huge! Complex data analysis and mining on huge amounts of data can take a long time, making such analysis impractical or infeasible.

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results. Strategies for data reduction include the following:

1. Data cube aggregation, where aggregation operations are applied to the data in the construction of a data cube.

2. Attribute subset selection, where irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed.

3. Dimensionality reduction, where encoding mechanisms are used to reduce the data set size.

4. Numerosity reduction, where the data are replaced or estimated by alternative, smaller data representations such as parametric models (which need store only the model parameters instead of the actual data) or nonparametric methods such as clustering, sampling, and the use of histograms.

5. Discretization and concept hierarchy generation, where raw data values for attributes are replaced by ranges or higher conceptual levels. Data discretization is a form of numerosity reduction that is very useful for the automatic generation of concept hierarchies. Discretization and concept hierarchy generation are powerful tools for data mining, in that they allow the mining of data at multiple levels of abstraction.

A concept hierarchy for price, for example, may map real price values into inexpensive, moderately priced, and expensive, thereby reducing the number of data values to be handled by the mining process