UNIT – 1

 Introduction

**1.1 What Motivated Data Mining? Why Is It Important?**

Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge.

The information and knowledge gained can be used for applications ranging from market

analysis, fraud detection, and customer retention, to production control and science exploration.

Data mining can be viewed as a result of the natural evolution of information technology. The database system industry has witnessed an evolutionary path in the development of the following functionalities

1. Data collection

 2. Database creation

 3. Data management

 4. Advanced data analysis

For instance, the early development of data collection and database creation mechanisms served as a prerequisite for later development of effective mechanisms for data storage and retrieval, and query and transaction processing. With numerous database systems offering query and transaction processing as common practice, advanced data analysis has naturally become the next target.

Since the 1960s, database and information technology has been evolving systematically from primitive file processing systems to sophisticated and powerful database systems. The research and development in database systems since the 1970s has progressed from early hierarchical and network database systems to the development of relational database systems data modeling tools, and indexing and accessing methods. In addition, users gained convenient and flexible data access through query languages, user interfaces, optimized query processing, and transaction management. Efficient methods for on-line transaction processing (OLTP), where a query is viewed as a read-only transaction, have contributed substantially to the evolution and wide acceptance of relational technology as a major tool for efficient storage, retrieval, and management of large amounts of data. Heterogeneous database systems and Internet-based global information systems such as the World Wide Web (WWW) have also emerged and play a vital role in the information industry.

Data can now be stored in many different kinds of databases and information repositories. One data repository architecture that has emerged is the data warehouse a repository of multiple heterogeneous data sources organized under a unified schema at a single site in order to facilitate management decision making. Data warehouse technology includes data cleaning, data integration, and on-line analytical processing (OLAP), that is, analysis techniques with functionalities such as summarization, consolidation, and aggregation as well as the ability to view information from different angles.

Although OLAP tools support multidimensional analysis and decision making, additional data analysis tools are required for in-depth analysis, such as data classification, clustering, and the characterization of data changes over time.



 Figure: The evolution of database system technology.

The abundance of data, coupled with the need for powerful data analysis tools, has been described as a *data rich but information poor* situation. The fast-growing, tremendous amount of data, collected and stored in large and numerous data repositories, has far exceeded our human ability for comprehension without powerful tools As a result, data collected in large data repositories become “data tombs”—data archives that are seldom visited. Consequently, important decisions are often made based not on the information-rich data stored in data repositories, but rather on a decision maker’s intuition, simply because the decision maker does not have the tools to extract the valuable knowledge embedded in the vast amounts of data. In addition, consider expert system technologies, which typically rely on users or domain experts to *manually* input knowledge into knowledge bases. Unfortunately, this procedure is prone to biases and errors, and is extremely time-consuming and costly. Data mining tools perform data analysis and may uncover important data patterns, contributing greatly to business strategies, knowledge bases, and scientific and medical research. The widening gap between data and information calls for a systematic development of *data mining tools* that will turn data tombs into “golden nuggets” of knowledge.



Figure: We are data rich, but information poor.

**1.1.1 What Is Data Mining?**

Simply stated, data mining refers to *extracting or “mining” knowledge from large amounts of data*. The term is actually a misnomer. Remember that the mining of gold from rocksor sand is referred to as *gold* mining rather than rock or sand mining. Thus, data miningshould have been more appropriately named “knowledge mining from data,” which isunfortunately somewhat long. “Knowledge mining,” a shorter term may not reflect theemphasis on mining from large amounts of data. Nevertheless, mining is a vivid termcharacterizing the process that finds a small set of precious nuggets from a great deal ofraw material.

 

 Figure: Data mining—searching for knowledge (interesting patterns) in your data.

Many people treat data mining as a synonym for another popularly used term, Knowledge

Discovery from Data, or KDD. Alternatively, others view data mining as simply anessential step in the process of knowledge discovery. Knowledge discovery as a processis depicted in Figure and consists of an iterative sequence of the following steps:

 

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

**1.** Data cleaning (to remove noise and inconsistent data)

**2.** Data integration (where multiple data sources may be combined)1

**3.** Data selection (where data relevant to the analysis task are retrieved from the database)

**4.** Data transformation (where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations, for instance)

**5.** Data mining (an essential process where intelligent methods are applied in order to extract data patterns)

**6.** Pattern evaluation (to identify the truly interesting patterns representing knowledge based on some interestingness measures)

**7.** Knowledge presentation (where visualization and knowledge representation techniques are used to present the mined knowledge to the user)

We agree that data mining is a step in the knowledge discovery process. However, in industry, in media, and in the database research milieu, the term data mining is becoming more popular than the longer term of knowledge discovery from data. We adopt a broad view of data mining functionality: data mining is the process of discovering interesting knowledge from large amounts of data stored in databases, data warehouses, or other information repositories.

**Database, data warehouse, World Wide Web, or other information repository:** This is one or a set of databases, data warehouses, spreadsheets, or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.

**Database or data warehouse server:** The database or data warehouse server is responsible for fetching the relevant data, based on the user’s data mining request.

Knowledge base: This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, used to organize attributes or attribute values into different levels of abstraction. Knowledge such as user beliefs, which can be used to assess a pattern’s interestingness based on its unexpectedness, may also be included. Other examples of domain knowledge are additional interestingness constraints or thresholds, and metadata (e.g., describing data from multiple heterogeneous sources).

**Data mining engine:** This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

**Pattern evaluation module:** This component typically employs interestingness measures and interacts with the data mining modules so as to *focus* the search toward interesting patterns. It may use interestingness thresholds to filter out discovered patterns. Alternatively, the pattern evaluation module may be integrated with the mining module, depending on the implementation of the data mining method used. For efficient data mining, it is highly recommended to push the evaluation of pattern interestingness as deep as possible into the mining process as to confine the search to only the interesting patterns.

**User interface:** This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse database and data warehouse schemas or data structures, evaluate mined patterns, and visualize the patterns in different forms.

 

 Figure: Architecture of a typical data mining system.

**1.2 Data Mining—On What Kind of Data?**

We examine a number of different data repositories on which mining can be performed. In principle, data mining should be applicable to any kind of data repository, as well as to transient data, such as data streams. Thus the scope of our examination of data repositories will include relational databases, data warehouses, and transactional databases, advanced database systems, flat files, data streams, and the World Wide Web. Advanced database systems include object-relational databases and specific application-oriented databases, such as spatial databases, time-series databases, text databases, and multimedia databases. The challenges and techniques of mining may differ for each of the repository systems.

A database system, also called a database management system (DBMS), consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data. The software programs involve mechanisms for the definition of database structures; for data storage; for concurrent, shared, or distributed data access; and for ensuring the consistency and security of the information stored, despite system crashes or attempts at unauthorized access.

**A relational database** is a collection of tables, each of which is assigned a unique name. Each table consists of a set of attributes (*columns* or *fields*) and usually stores a large set of tuples (*records* or *rows*). Each tuple in a relational table represents an object identified by a unique *key* and described by a set of attribute values. A semantic data model, such as an entity-relationship (ER) data model, is often constructed for relational databases. An ER data model represents the database as a set of entities and their relationships.

Example: A Relational database for ALL Electronics

 

 Figure: Fragments of relations from a relational database for *AllElectronics*.

Relational data can be accessed by database queries written in a relational query language, such as SQL, or with the assistance of graphical user interfaces.

Through the use of relational queries, you can ask things like “Show me a list of all items that were sold in the last quarter.” Relational languages also include aggregate functions such as sum, avg (average), count, max (maximum), and min (minimum). These allow you to ask things like “Show me the total sales of the last month, grouped by branch,” or “How many sales transactions occurred in the month of December?”

Relational databases are one of the most commonly available and rich information repositories, and thus they are a major data form in our study of data mining.

**Data Warehouses:**

Suppose that *AllElectronics* is a successful international company, with branches around the world. Each branch has its own set of databases. The president of *AllElectronics* has asked you to provide an analysis of the company’s sales per item type per branch for the third quarter. This is a difficult task, particularly since the relevant data are spread out over several databases, physically located at numerous sites.

If *AllElectronics* had a data warehouse, this task would be easy. A **data warehouse** is a repository of information collected from multiple sources, stored under a unified schema, and that usually resides at a single site. Data warehouses are constructed via a process of data cleaning, data integration, data transformation, data loading, and periodic data refreshing.

The process is as shown in the figure

 

 Figure: Typical framework of a data warehouse for *AllElectronics*.

A data warehouse is usually modeled by a multidimensional database structure, where each dimension corresponds to an attribute or a set of attributes in the schema, and each cell stores the value of some aggregate measure, such as *count* or *sales amount*. The actual physical structure of a data warehouse may be a relational data store or a multidimensional data cube. A data cube provides a multidimensional view of data and allows the precomputation and fast accessing of summarized data.

Example: A data cube for *AllElectronics*. A data cube for summarized sales data of *AllElectronic.* The cube has three dimensions: *address* (with city values *Chicago, New York, Toronto, Vancouver*), *time* (with quarter values *Q1, Q2, Q3, Q4*), and *item*(with itemtype values *home entertainment, computer, phone, security*). The aggregatevalue stored in each cell of the cube is *sales amount* (in thousands). For example, the totalsales forthefirstquarter,*Q1*, for items relating to security systems in Vancouver is$400,000,as stored in cell *Vancouver, Q1, security* Additional cubes may be used to store aggregatesums over each dimension, corresponding to the aggregate values obtained using differentSQL group-bys (e.g., the total sales amount per city and quarter, or per city and item, or per quarter and item, or per each individual dimension).

By providing multidimensional data views and the precomputation of summarized data, data warehouse systems are well suited for on-line analytical processing, or OLAP. OLAP operations use background knowledge regarding the domain of the data being studied in order to allow the presentation of data at *different levels of* *abstraction*. Such operations accommodate different user viewpoints. Examples of OLAP operations include drill-down and roll-up, which allow the user to view the data at differing degrees of summarization. For instance, we can drill down on sales data summarized by *quarter* to see the data summarized by *month*. Similarly, we can roll up on sales data summarized by *city* to view the data summarized by *country*.

 

Figure: A multidimensional data cube, commonly used for data warehousing, (a) showing summarized data for *AllElectronics* and (b) showing summarized data resulting from drill-down and roll-up operations on the cube in (a). For improved readability, only some of the cube cell values are shown.

**Transactional Databases**

In general, a **transactional database** consists of a file where each record represents a transaction. A transaction typically includes a unique transaction identity number (*trans ID*) and a list of the items making up the transaction (such as items purchased in a store).

The transactional database may have additional tables associated with it, which contain other information regarding the sale, such as the date of the transaction, the customer ID number, the ID number of the salesperson and of the branch at which the sale occurred, and so on.

Example: A transactional database for *AllElectronics*. Transactions can be stored in a table, with one record per transaction. A fragment of a transactional database for *AllElectronics*

is shown in Figure shown below. From the relational database point of view, the *sales* table in

Figure shown below is a nested relation because the attribute *list of item IDs* contains a set of *items*. Because most relational database systems do not support nested relational structures, the transactional database is usually either stored in a flat file in a format similar to that of

the table in Figure shown below or unfolded into a standard relation in a format similar to that of the *items sold* table in Figure above.

 

Figure: Fragment of a transactional database for sales at *AllElectronics*

**Advanced Data and Information Systems and Advanced Applications**

Relational database systems have been widely used in business applications. With the progress of database technology, various kinds of advanced data and information systems have emerged and are undergoing development to address the requirements of new applications.

The new database applications include handling spatial data (such as maps), engineering design data (such as the design of buildings, system components, or integrated circuits), hypertext and multimedia data (including text, image, video, and audio data), time-related data (such as historical records or stock exchange data), stream data (such as video surveillance and sensor data, where data flow in and out like streams), and the World Wide Web (a huge, widely distributed information repository made available by the Internet). These applications require efficient data structures and scalable methods for handling complex object structures; variable-length records; semi structured or unstructured data; text, spatiotemporal, and multimedia data; and database schemas with complex structures and dynamic changes.

**Object-Relational Databases**

Conceptually, the object-relational data model inherits the essential concepts of object-oriented databases, where, in general terms, each entity is considered as an object. Following the *AllElectronics* example, objects can be individual employees, customers, or items. Data and code relating to an object are *encapsulated* into a single unit. Each object has associated with it the following:

* A set of variables that describe the objects. These correspond to attributes in the entity-relationship and relational models.
* A set of messages that the object can use to communicate with other objects, or with the rest of the database system.
* A set of methods, where each method holds the code to implement a message. Upon receiving a message, the method returns a value in response. For instance, the method for the message *get photo* (*employee*) will retrieve and return a photo of the given employee object.

Objects that share a common set of properties can be grouped into an object class.

**Temporal Databases, Sequence Databases, and Time-Series Databases**

* A temporal database typically stores relational data that include time-related attributes. These attributes may involve several timestamps, each having different semantics.
* A sequence database stores sequences of ordered events, with or without a concrete notion of time. Examples include customer shopping sequences, Web click streams, and biological sequences.
* A time-series database stores sequences of values or events obtained over repeated measurements of time (e.g., hourly, daily, weekly). Examples include data collected from the stock exchange, inventory control, and the observation of natural phenomena (like temperature and wind).

**Spatial Databases and Spatiotemporal Databases**

Spatial databases contain spatial-related information. Examples include geographic (map) databases, very large-scale integration (VLSI) or computed-aided design databases, and medical and satellite image databases. Spatial data may be represented in raster format, consisting of *n*-dimensional bit maps or pixel maps. For example, a 2-D satellite image may be represented as raster data, where each pixel registers the rainfall in a given area. Maps can be represented in vector format, where roads, bridges, buildings, and lakes are represented as unions or overlays of basic geometric constructs, such as points, lines, polygons, and the partitions and networks formed by these components.

* 1. **Data Mining Functionalities—What Kinds of Patterns Can Be Mined?**

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.

**1.3.1 Concept/Class Description: Characterization and Discrimination**

Data can be associated with classes or concepts. For example, in the *AllElectronics* store, classes of items for sale include *computers* and *printers*, and concepts of customers include *bigSpenders* and *budgetSpenders*. It can be useful to describe individual classes and concepts in summarized, concise, and yet precise terms. Such descriptions of a class or a concept are called class/concept descriptions.

These descriptions can be derived via

(1**) *data characterization***, by summarizing the data of the class under study (often called the target class) in general terms, or

 (2) ***data discrimination***, by comparison of the target class with one or a set of comparative classes (often called the contrasting classes), or

(3) both **data characterization and discrimination**.

**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

There are several methods for effective data summarization and characterization. Simple data summaries based on statistical measures and plots are described. The data cube–based OLAP roll-up operation can be used to perform user-controlled data summarization along a specified dimension.

The output of data characterization can be presented in various forms. Examples include pie charts, bar charts, curves, multidimensional data cubes, and multidimensional tables, including crosstabs. The resulting descriptions can also be presented as generalized relations or in rule form(called characteristic rules).

Example:

**Data characterization:** A data mining system should be able to produce a description summarizing the characteristics of customers who spend more than $1,000 a year at *AllElectronics*. The result could be a general profile of the customers, such as they are 40–50 years old, employed, and have excellent credit ratings. The system should allow users to drill down on any dimension, such as on *occupation* in order to view these customers according to their type of employment.

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.

*“How are discrimination descriptions output?”* The forms of output presentation are similar to those for characteristic descriptions, although discrimination descriptions should include comparative measures that help distinguish between the target and contrasting classes. Discrimination descriptions expressed in rule form are referred to as discriminant rules.

Example:

Data discrimination: A data mining system should be able to compare two groups of *AllElectronics* customers, such as those who shop for computer products regularly (more than two times a month) versus those who rarely shop for such products (i.e., less than three times a year). The resulting description provides a general comparative profile of the customers, such as 80% of the customers who frequently purchase computer products are between 20 and 40 years old and have a university education, whereas 60% of the customers who infrequently buy such products are either seniors or youths, and have no university degree. Drilling down on a dimension, such as *occupation*, or adding new dimensions, such as *income level*, may help in finding even more discriminative features between the two classes.

**1.3.2 Mining Frequent Patterns, Associations, and Correlations**

Frequent patterns, as the name suggests, are patterns that occur frequently in data. There are many kinds of frequent patterns, including item sets, sub sequences, and substructures. A *frequent item set* typically refers to a set of items that frequently appear together in a transactional data set, such as milk and bread. A frequently occurring subsequence, such as the pattern that customers tend to purchase first a PC, followed by a digital camera, and then a memory card, is a (*frequent*) *sequential pattern*. A substructure can refer to different structural forms, such as graphs, trees, or lattices, which may be combined with item sets or sub sequences. If a substructure occurs frequently, it is called a (*frequent*) *structured pattern*. Mining frequent patterns leads to the discovery of interesting associations and correlations within data.

Example:

**Association analysis**: Suppose, as a marketing manager of *AllElectronics*, youwould like to determine which items are frequently purchased together within the same transactions. An example of such a rule, mined from the *AllElectronics* transactional database, is

*buys*(*X*; “*computer*”))*buys*(*X*; “*software*”) [*support* = 1%; *confidence* = 50%]

where *X* is a variable representing a customer. A confidence, or certainty, of 50% means that if a customer buys a computer, there is a 50% chance that she will buy software as well. A 1% support means that 1% of all of the transactions under analysis showed that computer and software were purchased together. This association rule involves a single attribute or predicate (i.e., *buys*) that repeats.Association rules that contain a single predicate are referred to as single-dimensional association rules. Dropping the predicate notation, the above rule can be written simply as “*computer*)*software* [1%, 50%]”.

Suppose, instead, that we are given the *AllElectronics* relational database relating to purchases. A data mining system may find association rules like

*age*(*X*, “20:::29”)^*income*(*X*, “20K:::29K”))*buys*(*X*, “*CD player*”) [*support* = 2%, *confidence* = 60%]

The rule indicates that of the *AllElectronics* customers under study, 2% are 20 to29 years of age with an income of 20,000 to 29,000 and have purchased a CD player at *AllElectronics*. There is a 60% probability that a customer in this age and income group will purchase a CD player. Note that this is an association between more than one attribute, or predicate (i.e., *age, income*, and *buys*). Adopting the terminology used in multidimensional databases, where each attribute is referred to as a dimension, the above rule can be referred to as a **multidimensional association rule.**

Typically, association rules are discarded as uninteresting if they do not satisfy both a minimum support threshold and a minimum confidence threshold. Additional analysis can be performed to uncover interesting statistical correlations between associated attribute-value pairs.

**1.3.3** **Classification and Prediction**

**Classification** is the process of finding a model (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known).

A **decision tree** is a flow-chart-like tree structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distributions. Decision trees can easily be converted to classification rules.

A **neural network**, when used for classification, is typically a collection of neuron-like processing units with weighted connections between the units. There are many other methods for constructing classification models, such as naïve Bayesian classification, support vector machines, and *k*-nearest neighbour classification.

Whereas classification predicts categorical (discrete, unordered) labels, **prediction** models continuous-valued functions. That is, it is used to predict missing or unavailable *numerical data values* rather than class labels. Although the term *prediction* may refer to both numeric prediction and class label prediction, in this book we use it to refer primarily to numeric prediction.

**Regression analysis** is a statistical methodology that is most often used for numeric prediction, although other methods exist as well. Prediction also encompasses the identification of distribution *trends* based on the available data.

Classification and prediction may need to be preceded by **relevance analysis**, which attempts to identify attributes that do not contribute to the classification or prediction process. These attributes can then be excluded.

 

Figure: A classification model can be represented in various forms, such as (a) IF-THEN rules, (b) a decision tree, or a (c) neural network.

**1.3.4** **Cluster Analysis**

*“What is cluster analysis?”*Unlike classification and prediction, which analyze class-labelled data objects, clustering analyzes data objects without consulting a known class label. In general, the class labels are not present in the training data simply because they are not known to begin with. Clustering can be used to generate such labels. The objects are clustered or grouped based on the principle of *maximizing the intraclass similarity and* *minimizing the interclass similarity.*

Example*:* Cluster analysis can be performed on *AllElectronics* customer data in order to identify homogeneous subpopulations of customers. These clusters may represent individual target groups for marketing. Figure shows a 2-D plot of customers with respect to customer locations in a city. Three clusters of data points are evident.

 

Figure: A 2-D plot of customer data with respect to customer locations in a city, showing three data clusters. Each cluster “centre” is marked with a “+”.

**1.3.5 Outlier Analysis**

A database may contain data objects that do not comply with the general behaviour or model of the data. These data objects are outliers. Most data mining methods discard outliers as noise or exceptions. However, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring ones. The analysis of outlier data is referred to as outlier mining.

**Example:** Outlier analysis. Outlier analysis may uncover fraudulent usage of credit cards by detecting purchases of extremely large amounts for a given account number in comparison to regular charges incurred by the same account. Outlier values may also be detected with respect to the location and type of purchase, or the purchase frequency.

**1.3.6 Evolution Analysis**

Data evolution analysis describes and models regularities or trends for objects whose behaviour changes over time. Although this may include characterization, discrimination, association and correlation analysis, classification, prediction, or clustering of *time related* data, distinct features of such an analysis include time-series data analysis, sequence or periodicity pattern matching, and similarity-based data analysis.

**Example:** Evolution analysis. Suppose that you have the major stock market (time-series) data of the last several years available from the New York Stock Exchange and you would like to invest in shares of high-tech industrial companies. A data mining study of stock exchange data may identify stock evolution regularities for overall stocks and for the stocks of particular companies. Such regularities may help predict future trends in stock market prices, contributing to your decision making regarding stock investments.

**1.4. Are All of the Patterns Interesting?**

A data mining system has the potential to generate thousands or even millions of patterns, or rules. *“So,”* you may ask, *“are all of the patterns interesting?”* Typically not—only a small fraction of the patterns potentially generated would actually be of interest to any given user.

This raises some serious questions for data mining. You may wonder, *“What makes a pattern interesting? Can a data mining system generate all of the interesting patterns? Can a data mining system generate only interesting patterns?”*

To answer the first question, a pattern is interesting if it is

(1) *Easily understood* by humans,

 (2) *Valid* on new or test data with some degree of *certainty*,

 (3) Potentially *useful*,

(4) Novel

Several objective measures of pattern interestingness exist. These are based on the structure of discovered patterns and the statistics underlying them. An objective measure for association rules of the form *X* )*Y* is rule support, representing the percentage of transactions from a transaction database that the given rule satisfies. This is taken to be the probability *P*(*X* [*Y*),where *X* [*Y* indicates that a transaction contains both *X* and *Y*, that is, the union of itemsets *X* and *Y*. Another objective measure for association rules is confidence, which assesses the degree of certainty of the detected association. This is taken to be the conditional probability *P*(*Y*j*X*), that is, the probability that a transaction containing *X* also contains *Y*. More formally, support and confidence are defined as

 

For example, patterns describing the characteristics of customers who shop frequently at *AllElectronics* should interest the marketing manager, but may be of little interest to analysts studying the same database for patterns on employee performance. Furthermore, many patterns that are interesting by objective standards may represent common knowledge and, therefore, are actually uninteresting. **Subjective interestingness measures** are based on user beliefs in the data. These measures find patterns interesting if they are **unexpected** (contradicting a user’s belief) or offer strategic information on which the user can act. In the latter case, such patterns are referred to as **actionable**. Patterns that are **expected** can be interesting if they confirm a hypothesis that the user

wished to validate, or resemble a user’s hunch.

**The second question—“*Can a data mining system generate all of the interesting patterns?*”**—

refers to the **completeness** of a data mining algorithm. It is often unrealisticand inefficient for data mining systems to generate all of the possible patterns.Instead, user-provided constraints and interestingness measures should be used to focusthe search.

**Finally, the third question—*“Can a data mining system generate only interesting patterns?”*—**

is an optimization problem in data mining. It is highly desirable for data mining systems to generate only interesting patterns. This would be much more efficient for users and data mining systems, because neither would have to search through the patterns generated in order to identify the truly interesting ones. Progress has been made in this direction; however, such optimization remains a challenging issue in data mining.

**1.5. 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. Moreover, depending on the data mining approach used, techniques from other disciplines may be applied, such as neural networks, fuzzy and/or rough set theory, knowledge representation, inductive logic programming, or high-performance computing.

Depending on the kinds of data to be mined or on the given data mining application, the data mining system may also integrate techniques from spatial data analysis, information retrieval, pattern recognition, image analysis, signal processing, computer graphics, Web technology, economics, business, bioinformatics, or psychology.

Because of the diversity of disciplines contributing to data mining, data mining research is expected to generate a large variety of data mining systems. Therefore, it is necessary to provide a clear classification of data mining systems, which may help potential users distinguish between such systems and identify those that best match their needs. Data mining systems can be categorized according to various criteria, as follows:

 

 Figure: Data mining as a confluence of multiple disciplines.

**Classification according to the *kinds of databases* mined:** A data mining system can be classified according to the kinds of databases mined. Database systems can be classified according to different criteria (such as data models, or the types of data or applications involved), each of which may require its own data mining technique. Data mining systems can therefore be classified accordingly.

**Classification according to the *kinds of knowledge* mined:** Data mining systems can be categorized according to the kinds of knowledge they mine, that is, based on data mining functionalities, such as characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis. A comprehensive data mining system usually provides multiple and/or integrated data mining functionalities.

**Classification according to the *kinds of techniques* utilized:** Data mining systems can be categorized according to the underlying data mining techniques employed. These techniques can be described according to the degree of user interaction involved (e.g., autonomous systems, interactive exploratory systems, query-driven systems) or the methods of data analysis employed (e.g., database-oriented or data warehouse– oriented techniques, machine learning, statistics, visualization, pattern recognition, neural networks, and so on).

**Classification according to the *applications adapted*:** Data mining systems can also be categorized according to the applications they adapt. For example, data mining systems may be tailored specifically for finance, telecommunications, DNA, stock markets, e-mail, and so on. Different applications often require the integration of application-specific methods. Therefore, a generic, all-purpose data mining system may not fit domain-specific mining tasks.

**1.6***.* **Data Mining Task Primitives:**

Each user will have a data mining task in mind, that is, some form of data analysis that he or she would like to have performed. 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.

* **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 below figure.
* **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*.
* **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.

A **data mining query language** can be designed to incorporate these primitives, allowing users to flexibly interact with data mining systems. Having a data mining query language provides a foundation on which user-friendly graphical interfaces can be built.

 

 Figure: Primitives for specifying a data mining task.

Designing a comprehensive data mining language is challenging because data mining covers a wide spectrum of tasks, from data characterization to evolution analysis. Each task has different requirements. The design of an effective data mining query language requires a deep understanding of the power, limitation, and underlying mechanisms of the various kinds of data mining tasks.

 

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

There are several proposals on data mining languages and standards. In this book, we use a data mining query language known as DMQL (Data Mining Query Language), which was designed as a teaching tool, based on the above primitives.

Example:

**Mining classification rules.** Suppose, as a marketing manager of *AllElectronics*, you would like to classify customers based on their buying patterns. You are especially interested in those customers whose salary is no less than $40,000, and who have bought more than $1,000 worth of items, each of which is priced at no less than $100. In particular, you are interested in the customer’s age, income, the types of items purchased, the purchase location, and where the items were made. You would like to view the resulting classification in the form of rules. This data mining query is expressed in DMQL as follows, where each line of the query has been enumerated to aid in our discussion.

(1) **use database** AllElectronics db

(2) **use hierarchy** location\_hierarchy for T.branch, age\_hierarchy for C.age

(3)**mine classification** as promising\_customers

(4) **in relevance** to C.age, C.income, I.type, I.place made, T.branch

(5) from customer C, item I, transaction T

(6) **where** I.item ID = T.item ID and C.cust ID = T.cust ID and

 C.income ≥40,000 and I.price ≥100

(7) **group by** T.cust ID

(8) **having** sum(I.price) ≥1,000

(9) **display** as rule

**1.7.** **Integration of a Data Mining System with a Database or Data Warehouse System:**

A critical question in the design of a data mining (DM) system is how to integrate or *couple* the DM system with a database (DB) system and/or a data warehouse (DW) system. If a DM system works as a stand-alone system or is embedded in an application program, there are no DB or DW systems with which it has to communicate. This simple scheme is called *no coupling*, where the main focus of the DM design rests on developing effective and efficient algorithms for mining the available data sets. However, 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, semi tight 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. Such a system, though simple, suffers from several drawbacks. First, a DB system provides a great deal of flexibility and efficiency at storing, organizing, accessing, and processing data. Without using a DB/DW system, a DM system may spend a substantial amount of time finding, collecting, cleaning, and transforming data. In DB and/or DW systems, data tend to be well organized, indexed, cleaned, integrated, or consolidated, so that finding the task-relevant, high-quality data becomes an easy task. Second, there are many tested, scalable algorithms and data structures implemented in DB and DW systems. It is feasible to realize efficient, scalable implementations using such systems.
* **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. Loose coupling is better than no coupling because it can fetch any portion of data stored in databases or data warehouses by using query processing, indexing, and other system facilities. It incurs some advantages of the flexibility, efficiency, and other features provided by such systems.
* **Semi tight coupling**: *Semi tight 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. These primitives can include sorting, indexing, aggregation, histogram analysis, multi way join, and precomputation of some essential statistical measures, such as sum, count, max, min, standard deviation, and so on.
* **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. Data mining queries and functions are optimized based on mining query analysis, data structures, indexing schemes, and query processing methods of a DB or DW system. With further technology advances, DM, DB, and DW systems will evolve and integrate together as one information system with multiple functionalities. This will provide a uniform information processing environment.

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. Semi tight 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.

**1.8. Major Issues in Data Mining:**

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**: These reflect the kinds of knowledge mined the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad hoc mining, and knowledge visualization.

* ***Mining different kinds of knowledge in databases****:* Because different users can be interested in different kinds of knowledge, data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis (which includes trend and similarity analysis).
* ***Interactive mining of knowledge at multiple levels of abstraction****:* Because it is difficult to know exactly what can be discovered within a database, the data mining process should be *interactive*. For databases containing a huge amount of data, appropriate sampling techniques can first be applied to facilitate interactive data exploration. Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results.
* ***Incorporation of background knowledge****:* Background knowledge, or information regarding the domain under study, may be used to guide the discovery process and allow discovered patterns to be expressed in concise terms and at different levels of abstraction.
* ***Data mining query languages and ad hoc data mining****:* Relational query languages (such as SQL) allow users to pose ad hoc queries for data retrieval. In a similar vein, high-level data mining query languages need to be developed to allow users to describe ad hoc data mining tasks by facilitating the specification of the relevant sets of data for analysis, the domain knowledge, the kinds of knowledge to be mined, and the conditions and constraints to be enforced on the discovered patterns
* ***Presentation and visualization of data mining results****:* Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that the knowledge can be easily understood and directly usable by humans.
* ***Handling noisy or incomplete data****:* The data stored in a database may reflect noise, exceptional cases, or incomplete data objects. When mining data regularities, these objects may confuse the process, causing the knowledge model constructed to over fit the data. As a result, the accuracy of the discovered patterns can be poor.
* ***Pattern evaluation—the interestingness problem:***A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, either because they represent common knowledge or lack novelty.

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

* ***Efficiency and scalability of data mining algorithms****:* To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable. In other words, the running time of a data mining algorithm must be predictable and acceptable in large databases.
* ***Parallel, distributed, and incremental mining algorithms****:* The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of parallel and distributed data mining algorithms. Such algorithms divide the data into partitions, which are processed in parallel.

**Issues relating to the diversity of database types:**

* ***Handling of relational and complex types of data****:* Because relational databases and data warehouses are widely used, the development of efficient and effective data mining systems for such data is important. However, 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****:* Local- and wide-area computer networks (such as the Internet) connect manysources of data, forming huge, distributed, and heterogeneous databases. The discoveryof knowledge from different sources of structured, semi structured, orunstructured data with diverse data semantics poses great challenges to datamining.

The above issues are considered major requirements and challenges for the further evolution of data mining technology. Some of the challenges have been addressed in recent data mining research and development, *to a certain extent*, and are now considered *requirements*, while others are still at the research stage