LESSON

6

DATABASE/OLAP SYSTEM

CONTENTS

6.0 AIMS AND OBJECTIVE

After studying this lesson, you should be able to:

- Explain multi-dimensional data model
- Describe OLAP operations in the multi-dimensional data model
- Describe physical implementation of an OLAP server
- State further developments in data cube technology

6.1 INTRODUCTION

OLAP is a broad term that also encompasses data warehousing. In this model data is stored in a format, which enables the efficient creation of data mining/reports.

The first major stepping stone in understanding Data Warehousing is to grasp the concepts and differences between the two overall database categories. The type most of us are used to dealing with is the On Line Transactional Processing (OLTP) category. The other major category is On Line Analytical Processing (OLAP).

6.2 OLAP DIMENSIONAL MODELING

The first major stepping stone in understanding Data Warehousing is to grasp the concepts and differences between the two overall database categories. The type most of us are used to dealing with is the On Line Transactional Processing (OLTP) category. The other major category is On Line Analytical Processing (OLAP).

OLTP is what we characterize as the ongoing day-to-day functional copy of the database. It is where data is added and updated but never overwritten or deleted. The main needs of the OLTP operational database being easily controlled insertion and updating of data with efficient access to data manipulation and viewing mechanisms. Typically only single record or small record-sets should be manipulated in a single operation in an OLTP designed database. The main thrust here is to avoid having the same data in different tables. This basic tenet of Relational Database modeling is known as "normalizing" data.

OLAP is a broad term that also encompasses data warehousing. In this model data is stored in a format, which enables the efficient creation of data mining/reports. OLAP design should accommodate reporting on very large record sets with little degradation in operational efficiency. The overall term used to describe taking data structures in an OLTP format and holding the same data in an OLAP format is "Dimensional Modeling" It is the primary building block of Data Warehousing.

Need to Build a Data Warehouse

You know that data warehouse queries are often complex. They involve the computation of large groups of data at summarized levels and may require the use of special data organization, access, and implementation methods based on multidimensional views. Processing OLAP queries in operational databases would substantially degrade the performance of operational tasks. Moreover, an operational database supports the concurrent processing of several transactions as well recovery mechanism such as locking and logging to ensure the consistency and robustness of transactions. An

OLAP query often needs read-only access of data records for summarization and aggregation. Concurrency control and recovery mechanisms, if applied for such OLAP operations, may jeopardize the execution of concurrent transactions and thus substantially reduce the throughput of an OLTP system.

Finally, the separation of operational databases from data warehouses is based on the different structures, contents, and uses of the data in these two systems.

6.3 MULTI-DIMENSIONAL DATA MODEL

Traditional access control models for transactional (relational) databases, based on tables, columns and rows, are not appropriate for Data Warehouses. Instead, security and audit rules defined for Data Warehouses must be specified based on the Multidimensional (MD) modeling used to design data warehouses.

Data Warehouses (DW), Multidimensional (MD) Databases, and Online Analytical Processing (OLAP) applications are used in conjunction to form a highly powerful mechanism for discovering crucial business information in strategic decision-making processes. Multidimensional modeling is the foundation of Data Warehouses, MD databases and OLAP applications.

In data warehouses, this needed is even much harder as the multidimensional modeling used to design the data warehouse is the same one that the final user will use to query the data warehouse by front-end tools such as OLAP tools.

Data warehouses and OLAP tools are based on a multi-dimensional data model. It is designed to solve complex queries in real time. The central attraction of the dimensional model of a business is its simplicity which is the fundamental key that allows users to understand databases, and allows software to navigate databases efficiently.

The multi-dimensional data model is composed of logical cubes, measures, dimensions, hierarchies, levels, and attributes. Figure 6.1 shows the relationships among the logical objects.

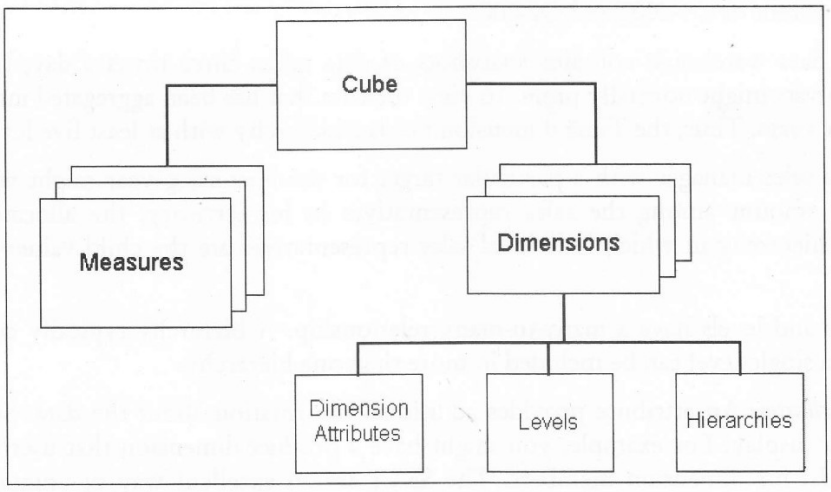


Figure 6.1: Diagram of the Logical Multi-dimensional Model

1. *Logical Cubes:* Logical cubes provide a means of organizing measures that have the same shape, that is, they have the exact same dimensions. Measures in the same cube have the same relationships to other logical objects and can easily be analyzed and displayed together.

2. *Logical Measures:* Measures populate the cells of a logical cube with the facts collected about business operations. Measures are organized by dimensions, which typically include a Time dimension.

   Measures are static and consistent while analysts are using them to inform their decisions. They are updated in a batch window at regular intervals: weekly, daily, or periodically throughout the day. Many applications refresh their data by adding periods to the time dimension of a measure, and may also roll off an equal number of the oldest time periods. Each update provides a fixed historical record of a particular business activity for that interval. Other applications do a full rebuild of their data rather than performing incremental updates.

3. *Logical Dimensions:* Dimensions contain a set of unique values that identify and categorize data. They form the edges of a logical cube, and thus of the measures within the cube. Because measures are typically multidimensional, a single value in a measure must be qualified by a member of each dimension to be meaningful. For example, the Sales measure has four dimensions: Time, Customer, Product, and Channel. A particular Sales value (43,613.50) only has meaning when it is qualified by a specific time period (Feb-01), a customer (Warren Systems), a product (Portable PCs), and a channel (Catalog).

4. *Logical Hierarchies and Levels:* A hierarchy is a way to organize data at different levels of aggregation. In viewing data, analysts use dimension hierarchies to recognize trends at one level, drill down to lower levels to identify reasons for these trends, and roll up to higher levels to see what affect these trends have on a larger sector of the business.

   Each level represents a position in the hierarchy. Each level above the base (or most detailed) level contains aggregate values for the levels below it. The members at different levels have a one-to-many parent-child relation. For example, Q1-2002 and Q2-2002 are the children of 2002, thus 2002 is the parent of Q1-2002 and Q2-2002.

   Suppose a data warehouse contains snapshots of data taken three times a day, that is, every 8 hours. Analysts might normally prefer to view the data that has been aggregated into days, weeks, quarters, or years. Thus, the Time dimension needs a hierarchy with at least five levels.

   Similarly, a sales manager with a particular target for the upcoming year might want to allocate that target amount among the sales representatives in his territory; the allocation requires a dimension hierarchy in which individual sales representatives are the child values of a particular territory.

   Hierarchies and levels have a many-to-many relationship. A hierarchy typically contains several levels, and a single level can be included in more than one hierarchy.

5. *Logical Attributes:* An attribute provides additional information about the data. Some attributes are used for display. For example, you might have a product dimension that uses Stock Keeping Units (SKUs) for dimension members. The SKUs are an excellent way of uniquely identifying thousands of products, but are meaningless to most people if they are used to label the data in a report or graph. You would define attributes for the descriptive labels.

Multi-dimensional Modeling

Instead of representing a multi-dimensional model at a logical level, we propose to adopt a conceptual perspective and use the ER-like graphical notations shown in Figure 6.2.

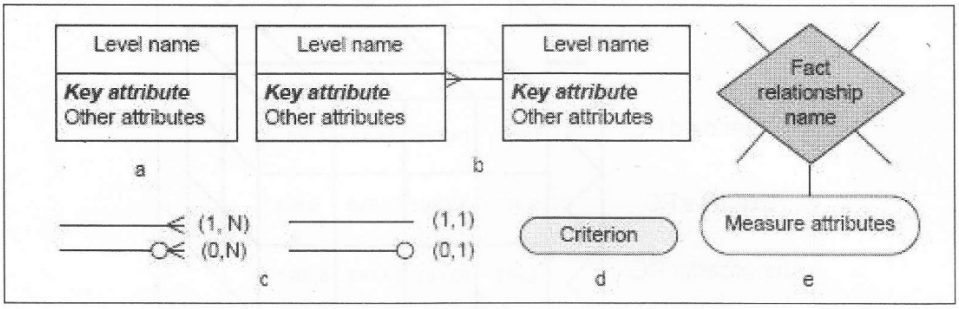


Figure 6.2: (a) Level, (b) Hierarchy, (c) Cardinalities, (d) Analysis Criterion, and (e) Fact Relationship

We define a *MultiDimER* schema as a finite set of dimensions and fact relationships. A *dimension* is an abstract concept for grouping data that shares a common semantic meaning within the domain being modeled. It represents either a level, or one or more hierarchies. Levels are represented as entity types [Figure 6.2(a)]. Every instance of a level is called *member*.

A *hierarchy* contains several related levels that are used for roll-up and drill-down operations. Since the binary relationship linking the levels of a hierarchy is only used for traversing from one level to the next one, we propose to represent hierarchies using the notation in [Figure 6.2(b)].

A relationship joining a child and a parent levels is characterized by the cardinalities and the analysis criterion. *Cardinalities* [Figure 6.2(c)] indicate the minimum and the maximum numbers of members in one level that can be related to a member in another level. The *analysis criterion* [Figure 6.2(d)] expresses different structures used for analysis, e.g., geographical location or organizational structure.

The MultiDimER model is mainly based on the existing ER model constructs with their usual semantics, i.e., entity types, attributes, relationship types.

6.3.1 Multi-dimensional Data Storage in Analytic Workspaces

In the logical multidimensional model, a cube represents all measures with the same shape, that is, the exact same dimensions. In a cube shape, each edge represents a dimension. The dimension members are aligned on the edges and divide the cube shape into cells in which data values are stored.

In an analytic workspace, the cube shape also represents the physical storage of multidimensional measures, in contrast with two-dimensional relational tables. An advantage of the cube shape is that it can be rotated: there is no one right way to manipulate or view the data. This is an important part of multidimensional data storage, calculation, and display, because different analysts need to view the data in different ways. For example, if you are the Sales Manager, then you need to look at the data differently from a product manager or a financial analyst.

Assume that a company collects data on sales. The company maintains records that quantify how many of each product was sold in a particular sales region during a specific time period. You can visualize the sales measure as the cube shown in Figure 6.3.

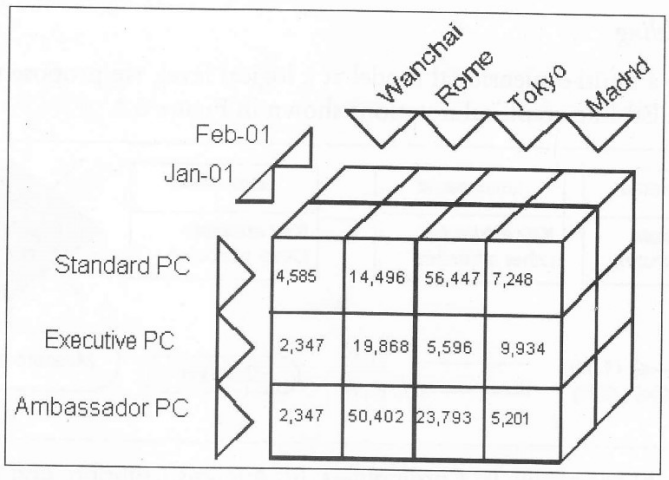


Figure 6.3: Comparison of Product Sales by City

Figure 6.3 compares the sales of various products in different cities for January 2001 (shown) and February 2001 (not shown). This view of the data might be used to identify products that are performing poorly in certain markets. Figure 6.4 shows sales of various products during a four-month period in Rome (shown) and Tokyo (not shown). This view of the data is the basis for trend analysis.

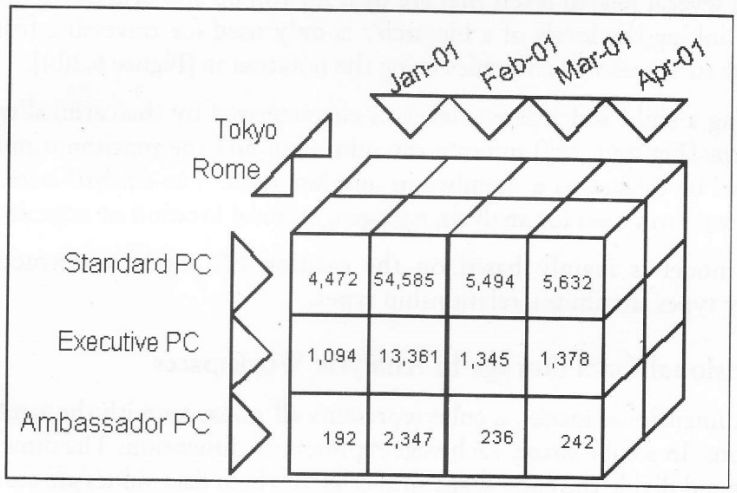


Figure 6.4: Comparison of Product Sales by Month

A cube shape is three dimensional. Of course, measures can have many more than three dimensions, but three dimensions are the maximum number that can be represented pictorially. Additional dimensions are pictured with additional cube shapes.

6.3.2 Relational Implementation of the Model

The relational implementation of the multi-dimensional data model is typically a star schema, as shown in Figure 6.5, a snowflake schema or a fact constellation schema.

Star Schema

A star schema is a convention for organizing the data into dimension tables, fact tables, and materialized views. Ultimately, all of the data is stored in columns, and metadata is required to identify the columns that function as multidimensional objects.

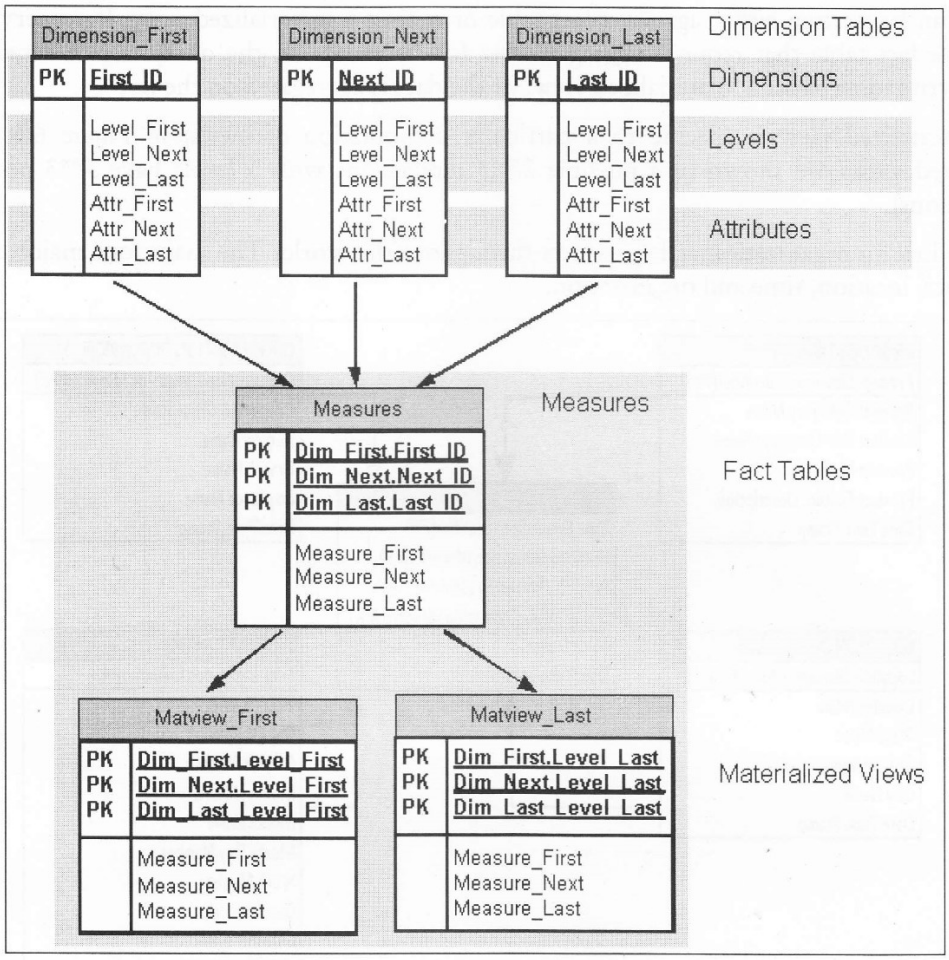


Figure 6.5: Diagram of a Star Schema

1. ***Dimension Tables:*** A star schema stores all of the information about a dimension in a single table. Each level of a hierarchy is represented by a column or column set in the dimension table. A dimension object can be used to define the hierarchical relationship between two columns (or column sets) that represent two levels of a hierarchy; without a dimension object, the hierarchical relationships are defined only in metadata. Attributes are stored in columns of the dimension tables.

   A snowflake schema normalizes the dimension members by storing each level in a separate table.

2. ***Fact Tables:*** Measures are stored in fact tables. Fact tables contain a composite primary key, which is composed of several foreign keys (one for each dimension table) and a column for each measure that uses these dimensions.

Materialized Views

Aggregate data is calculated on the basis of the hierarchical relationships defined in the dimension tables. These aggregates are stored in separate tables, called summary tables or materialized views. Oracle provides extensive support for materialized views, including automatic refresh and query rewrite.

Queries can be written either against a fact table or against a materialized view. If a query is written against the fact table that requires aggregate data for its result set, the query is either redirected by query rewrite to an existing materialized view, or the data is aggregated on the fly.

Each materialized view is specific to a particular combination of levels; in Figure 6.6, only two materialized views are shown of a possible 27 (3 dimensions with 3 levels have 3**3 possible level combinations).

Example: Let, an organization sells products throughout the world. The main four major dimensions are product, location, time and organization.

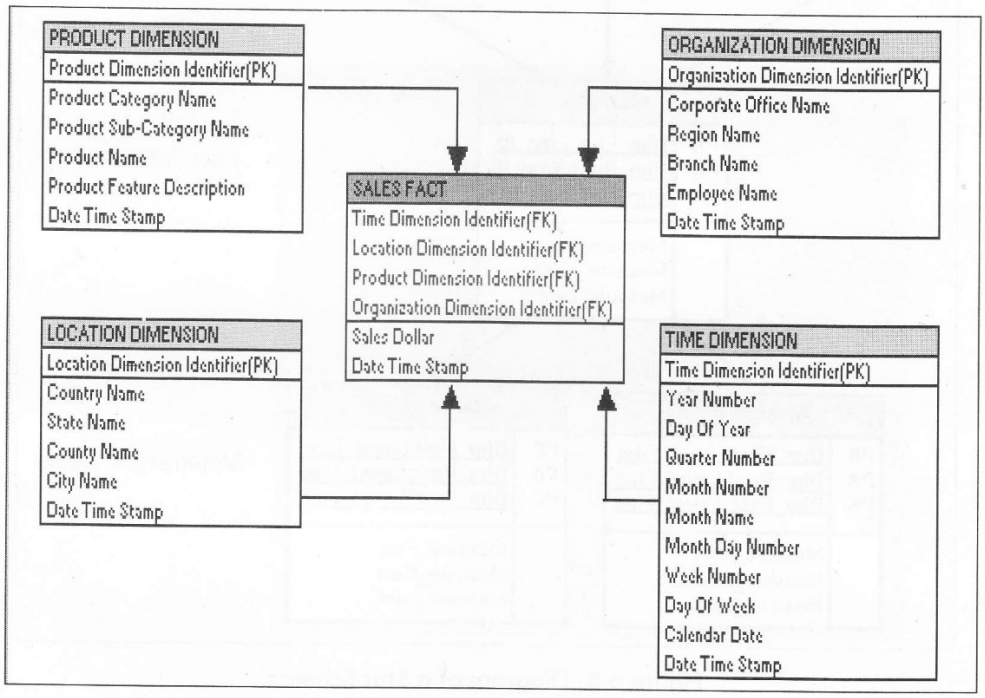


Figure 6.6: Example of Star Schema

In the example Figure 6.6, sales fact table is connected to dimensions location, product, time and organization. It shows that data can be sliced across all dimensions and again it is possible for the data to be aggregated across multiple dimensions. "Sales Dollar" in sales fact table can be calculated across all dimensions independently or in a combined manner, which is *explained below:*

- Sales Dollar value for a particular product
- Sales Dollar value for a product in a location
- Sales Dollar value for a product in a year within a location
- Sales Dollar value for a product in a year within a location sold or serviced by an employee

Snowflake Schema

The snowflake schema is a variant of the star schema model, where some dimension tables are *normalized*, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.

The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalized form. Such a table is easy to maintain and also saves storage space because a large dimension table can be extremely large when the dimensional structure is included as columns. Since much of this space is redundant data, creating a normalized structure will reduce the overall space requirement. However, the snowflake structure can reduce the effectiveness of browsing since more joins will be needed to execute a query. Consequently, the system performance may be adversely impacted. Performance benchmarking can be used to determine what is best for your design.

Example: In Snowflake schema, the example diagram shown in Figure 6.7 has 4 dimension tables, 4 lookup tables and 1 fact table. The reason is that hierarchies (category, branch, state, and month) are being broken out of the dimension tables (Product, Organisation, Location, and Time) respectively and shown separately. In OLAP, this Snowflake schema approach increases the number of joins and poor performance in retrieval of data.

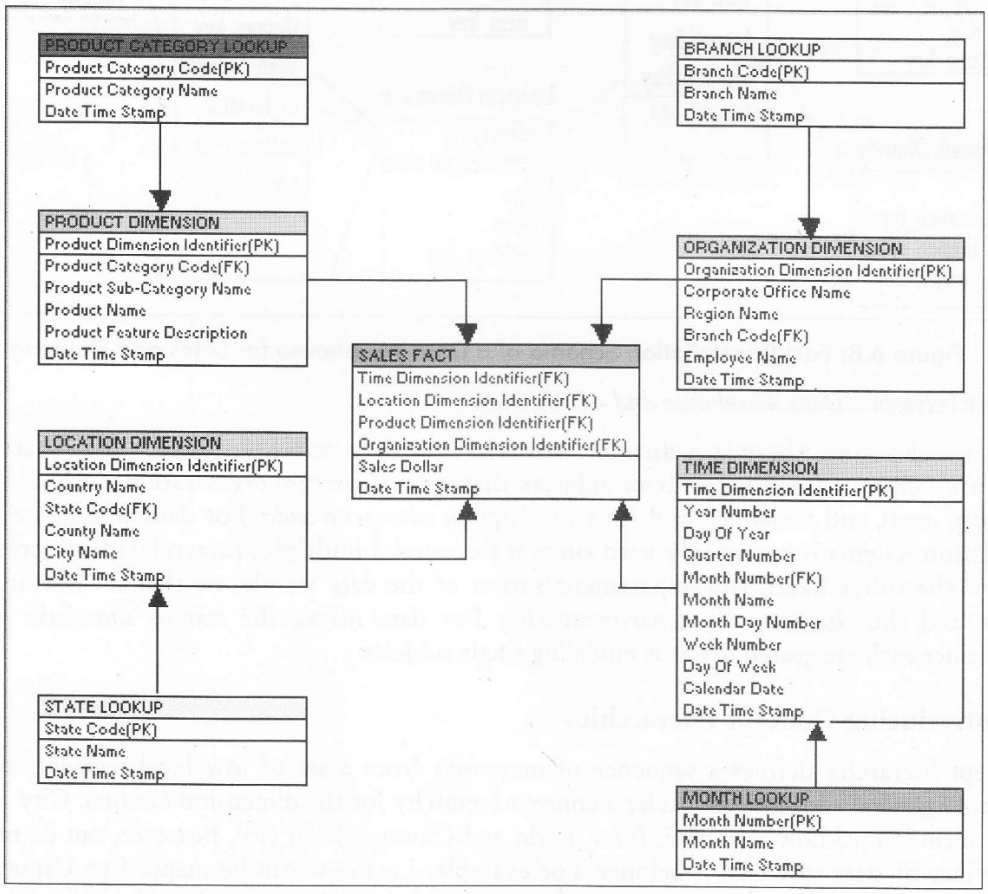


Figure 6.7: Example of Snowflake Schema

A compromise between the star schema and the snowflake schema is to adopt a mixed schema where only the very large dimension tables are normalized. Normalizing large dimension tables saves storage space, while keeping small dimension tables unnormalized may reduce the cost and performance degradation due to joins on multiple dimension tables. Doing both may lead to an overall performance gain. However, careful performance tuning could be required to determine which dimension tables should be normalized and split into multiple tables.

Fact Constellation

Sophisticated applications may require multiple fact tables to *share* dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation. For example, a fact constellation schema of a data warehouse for sales and shipping is shown in the following Figure 6.8.

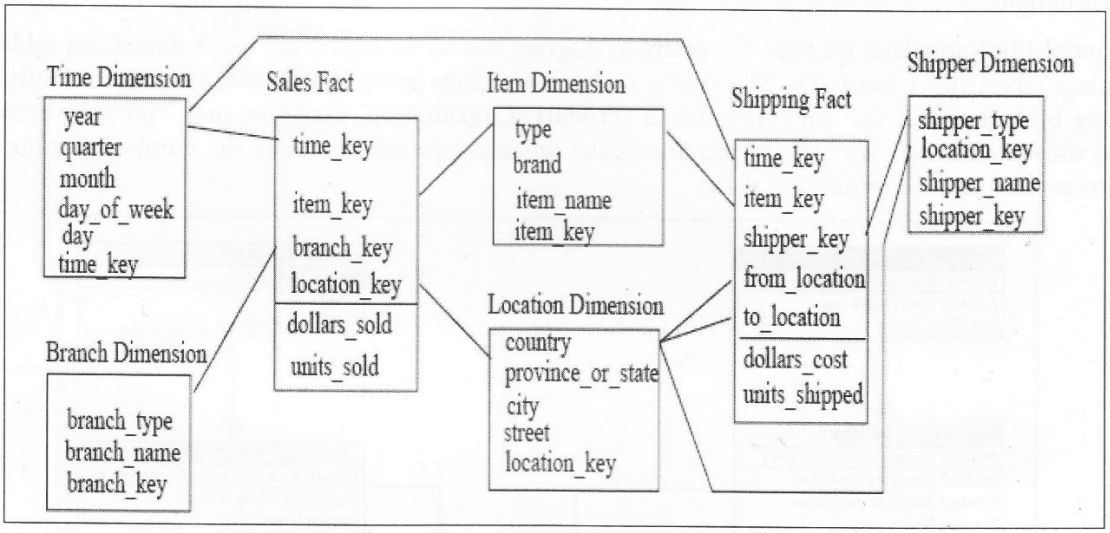


Figure 6.8: Fact Constellation Schema of a Data Warehouse for Sales and Shipping

Difference between a Data Warehouse and a Data Mart

In data warehousing, there is a distinction between a data warehouse and a data mart. A data warehouse collects information about subjects that span the *entire organization,* such as *customers, items, sales, assets,* and *personnel,* and thus its scope is *enterprise-wide.* For data warehouse, the fact constellation schema is commonly used since it can model multiple, interrelated subjects. A data mart, on the other hand, is a department subset of the data warehouse that focuses on selected subjects, and thus its scope is *department-wide.* For data marts, the *star* or *snowflake* schema is popular since each are geared towards modeling single subjects.

6.3.3 Introducing Concept Hierarchies

A concept hierarchy defines a sequence of mappings from a set of low level concepts to higher level, more general concepts. Consider a concept hierarchy for the dimension *location.* City values for *location* include Lucknow, Mumbai, New York, and Chicago. Each city, however, can be mapped to the province or state to which it belongs. For example, Lucknow can be mapped to Uttar Pradesh, and Chicago to Illinois. The provinces and states can in turn be mapped to the country to which

they belong, such as India or the USA. These mappings form a concept hierarchy for the dimension *location*, mapping a set of low level concepts (i.e., cities) to higher level, more general concepts (i.e., countries). The concept hierarchy described above is illustrated in Figure 6.9.

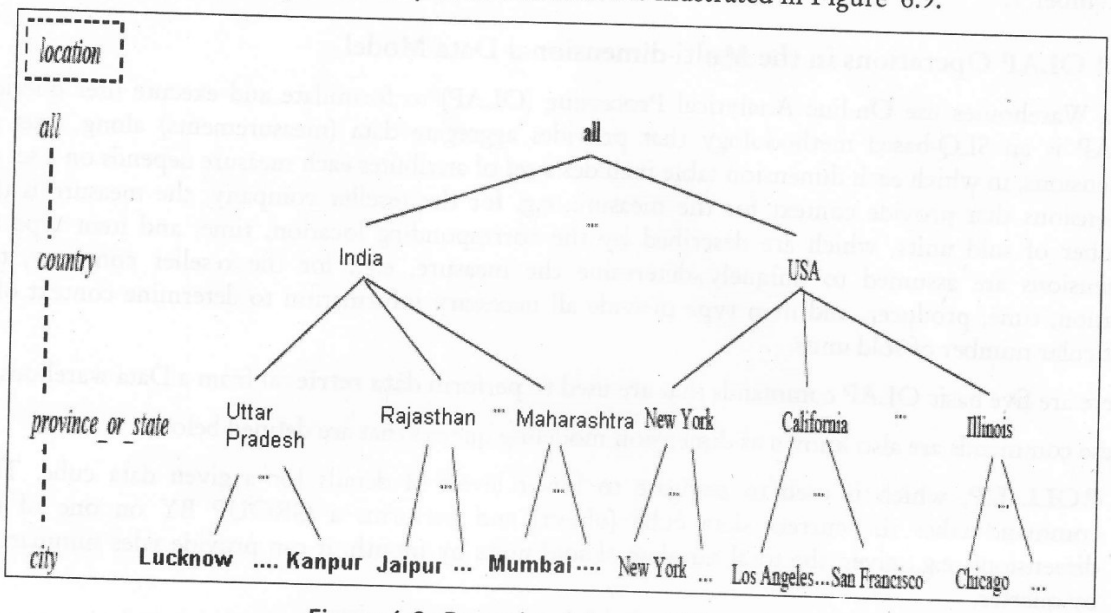


Figure 6.9: Example of Concept Hierarchy

Many concept hierarchies are implicit within the database schema. For example, suppose th at the dimension *location* is described by the attributes *number, street, city, province-or state, zip code,* and *country.* These attributes are related by a total order, forming a concept hierarchy such as *"street < city < province.or.state < country"*. This hierarchy is shown in Figure 6.10 (a).

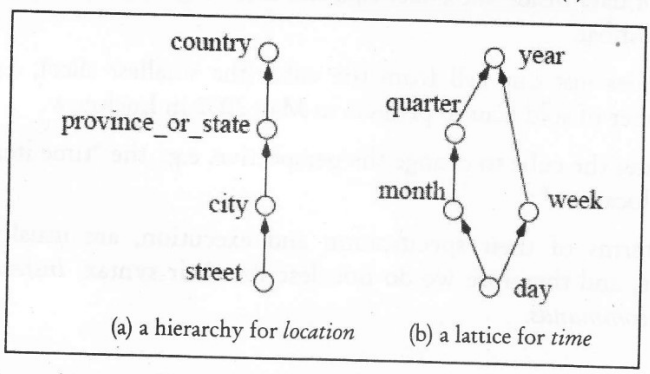


Figure 6.10: Hierarchical and Lattice Structures of Attributes in Warehouse Dimensions

Alternatively, the attributes of a dimension may be organized in a partial order, forming a lattice. An example of a partial order for the *time* dimension based on the attributes *day, week, month, quarter,* and *year* is *"day < {month <quarter; week} < year"*. This lattice structure is shown in Figure 6.10(b).

A concept hierarchy that is a total or partial order among attributes in a database schema is called a schema hierarchy. Concept hierarchies that are common to many applications may be predefined in the data mining system, such as the concept hierarchy for *time.* Data mining systems should

provide users with the flexibility to tailor predefined hierarchies according to their particular needs. For example, one may like to define a fiscal year starting on April 1, or an academic year starting on September 1.

6.3.4 OLAP Operations in the Multi-dimensional Data Model

Data Warehouses use On-line Analytical Processing (OLAP) to formulate and execute user queries. OLAP is an SLQ-based methodology that provides aggregate data (measurements) along a set of dimensions, in which each dimension table includes a set of attributes each measure depends on a set of dimensions that provide context for the measure, e.g. for the reseller company, the measure is the number of sold units, which are described by the corresponding location, time, and item type all dimensions are assumed to uniquely determine the measure, e.g., for the reseller company, the location, time, producer, and item type provide all necessary information to determine context of a particular number of sold units.

There are five basic OLAP commands that are used to perform **data retrieval** from a Data warehouse.

These commands are also known as dimension modeling queries that are defined below:

- **ROLL UP**, which is used to navigate to lower levels of details for a given data cube. This command takes the current data cube (object) and performs a GROUP BY on one of the dimensions, e.g., given the total number of sold units by month, it can provide sales summarised by quarter.
- **DRILL DOWN**, which is used to navigate to higher levels of detail. This command is the opposite of ROLL UP, e.g., given the total number of units sold for an entire continent, it can provide sales in the U.S.A.
- **SLICE**, which provides a cut through a given data cube. This command enables users to focus on some specific slice of data inside the cube, e.g., the user may want to look at the data concerning unit sales only in Mumbai.
- **DICE**, which provides just one cell from the cube (the smallest slice), e.g. it can provide data concerning the number of sold Canon printers in May 2002 in Lucknow.
- **PIVOT**, which rotates the cube to change the perspective, e.g., the "time item" perspective may be changed into "time location."

These commands, in terms of their specification and execution, are usually carried out using a point-and-click interface, and therefore we do not describe their syntax. *Instead, we give examples for each of the above OLAP commands.*

ROLL UP Command

The ROLL UP allows the user to summarise data into a more general level in hierarchy. For instance, if the user currently analyses the number of sold CPU units for each month in the first half of 2002, this command will allows him/her to aggregate this information into the first two quarters.

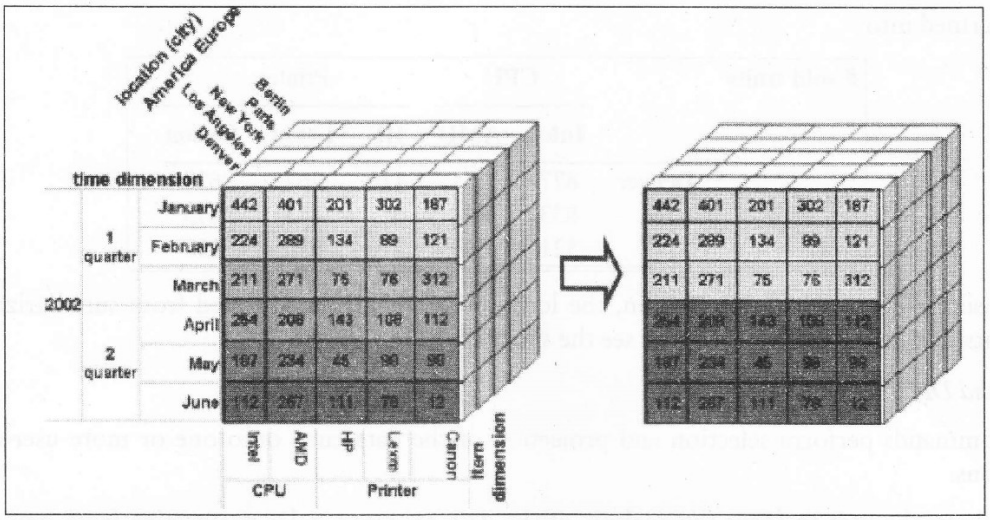


Figure 6.11: Example ROLL UP command

# sold units		2002					
		January	February	March	April	May	June
CPU	Intel	442	224	211	254	187	112
	AMD	401	289	271	208	234	267

is transformed into

# sold units		2002	
		Quarter 1	Quarter 2
CPU	Intel	877	553
	AMD	961	709

From the perspective of a three-dimensional cuboid, the time _y_ axis is transformed from months to quarters; see the shaded cells in Figure 6.11(a)

DRILL DOWN Command

The DRILL DOWN command provides a more detailed breakdown of information from lower in the hierarchy. For instance, if the user currently analyses the number of sold CPU and Printer units in Europe and U.S.A., it will allows him/her to find details of sales in specific cities in the U.S.A., i.e., the view.

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	USA	2231	2134	1801	1560	1129
	Europe	1981	2001	1432	1431	1876

is transformed into

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	Denver	877	961	410	467	620
	LA	833	574	621	443	213
	NY	521	599	770	650	296

Again, using a data cube representation, the location (z) axis is transformed from summarization by continents to sales for individual cities; see the shaded cells in Figure 6.11(b)

SLICE and DICE Command

These commands perform selection and projection of the data cube onto one or more user-specified dimensions.

SLICE allows the user to focus the analysis of the data on a particular perspective from one or more dimensions. For instance, if the user analyses the number of sold CPU and Printer units in all combined locations in the first two quarters of 2002, he/she can ask to see the units in the same time frame in a particular city, say in Los Angeles. The view

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
2002	1 quarter	2231	2001	2390	1780	1560
	2 quarter	2321	2341	2403	1851	1621

is transformed into the L.A. table

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
2002	1 quarter	666	601	766	187	730
	2 quarter	1053	759	323	693	501

The *DICE* command, in contrast to SLICE, requires the user to impose restrictions on all dimensions in a given data cube. An example SLICE command, which provides data about sales only in L.A., and DICE command, which provides data about sales of Canon printers in May 2002 in L.A.

PIVOT Command

PIVOT is used to rotate a given data cube to select a different view. Given that the user currently analyses the sales for particular products in the first quarter of 2002, he/she can shift the focus to see sales in the same quarter, but for different continents instead of for products, i.e., the view.

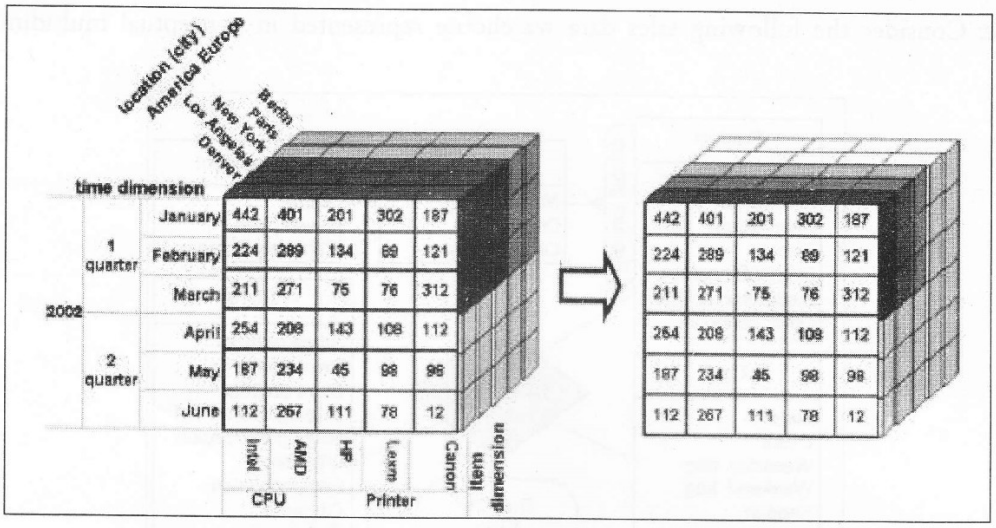


Figure 6.12: Example DRILL DOWN Command

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
1 quarter	January	442	401	201	302	187
	February	224	289	134	89	121
	March	211	271	75	76	312

is transformed into

# sold units		America		Europe		
		Denver	LA	NY	Paris	Berlin
1 quarter	January	556	321	432	432	341
	February	453	564	654	213	231
	March	123	234	345	112	232

6.3.5 Object Oriented Multi-dimensional Modeling

In the object oriented modeling the UML we use for Data Warehouse conceptual modeling. This approach has been specified by means of a UML profile which contains the necessary stereotypes for carrying out the multidimensional modeling at the conceptual level successfully. The main features of multidimensional modeling considered here are the relationships many-to-many between facts and one specific dimension, degenerated dimensions, multiple classification and alternative path hierarchies, and non-strict and complete hierarchies. In this approach, structural properties of multidimensional modeling are represented by means of a UML class diagram in which the information is clearly organized into facts (items of interest for a given business) and dimensions (context in which facts have to be analyzed).

Example: Consider the following sales data warehouse represented in conceptual multidimensional model.

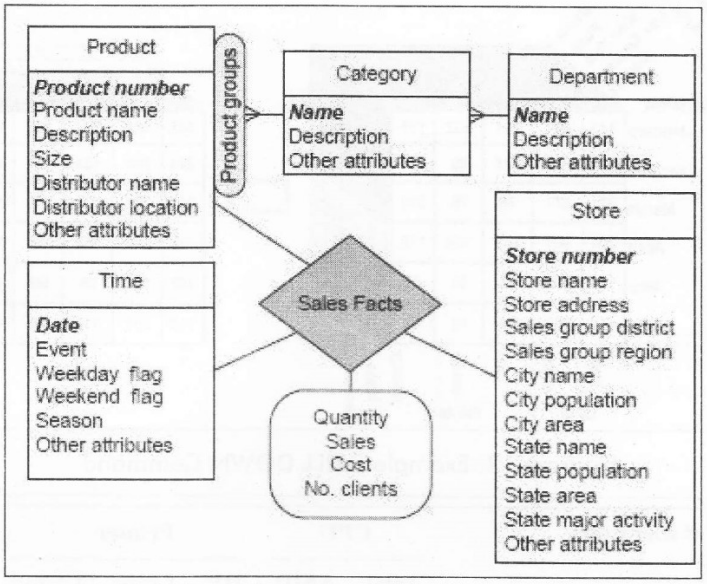


Figure 6.13

With this model we can easily trace the relationship among the entities and data can easily transformable.

6.3.6 Access Control and Audit Model

Access control is not a complete solution for securing a system as it must be coupled with auditing. Auditing requires the recording of all user requests and activities for their later analysis. Therefore, in our approach, we take both concepts into consideration for their integration in the conceptual multidimensional modeling design.

Access control models are typically composed of a set of authorization rules that regulate accesses to objects. Each authorization rule usually specifies the subject to which the rule applies, the object to which the authorization refers, the action to which the rule refers, and the sign describing whether the rule permits or denies the access. Once the data warehouse designer has developed the conceptual multidimensional model, he or she should discover all the security issues associated to this model, and specify them through our ACA model.

Access Control Model

In multilevel policies, an access class is assigned to each object and subject. The most common access class is defined as a security level and a set of categories. The security level is an element of a hierarchically ordered set, such as Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where, TS > S > C > U. The set of categories is a subset of an unordered set, whose elements reflect functional, or competence, areas. The access class is one element of a partially ordered set of classes, where an access class c1 dominates an access class c2 if the security level of c1 is greater than or equal to that of c2 and the categories of c1 include those of c2. We have considered a secrecy-based mandatory policy, so the two principles that must be satisfied to protect information confidentiality are: (i) no-

read-up (a subject is allowed a read-access to an object only if the access class of the subject dominates the access class of the object), and (ii) no-write-down (a subject is allowed a write-access to an object only if the access class of the subject is dominated by the access class of the object.

6.3.7 A Logical Model for Multi-dimensional Analysis

A logical model describes the available data from the perspective of the user. Therefore it is important to provide the user with a logical structuring of data that supports his or her intuitive interpretation of an MDD. From the user viewpoint, a logical structure of multidimensional data is necessary for organizing, managing and querying preprocessed data. The logical model can be considered as a variant of the star model because it contains fact and dimension tables. However, we call fact tables (multidimensional) data cubes.

Multidimensional database consists of a collection of multidimensional *data cubes,* a collection of *dimension tables* and a collection of *hierarchy tables.* In addition, the model contains mechanisms for specifying how data in different tables are semantically associated with each other. Multidimensional database contains historical data on the publications of persons and their grants.

Data Cubes:

authoring_table				
year	person	domain	refereed	non_refereed

grant_table			
year	person	domain	grants

citation_table		
year	article	citation_count

Figure 6.14

Data cubes contain the summary data on which multidimensional analysis is based. A complex multidimensional application (e.g., informatics) contains data related to various contexts, which, however, may share some dimensions. It is therefore clearer to produce one data cube per context than to use one data cube involving all the contexts. A data cube consists of only those dimension attributes shared by all its measure attributes. This means that the dimension attributes form the key of a data cube. If information from several data cubes is needed they may be joined on one or more common dimensions. Navigation among data cubes is invisible to the user. Figure 6.2.7 presents the schema of our sample multidimensional database data cubes.

Dimension Tables

Dimension tables represent dimension-specific properties. For each dimension attribute in the data cubes there may be at most one dimension table. Unlike the original star model, we allow data cube dimensions with no dimension tables. The schema level of a dimension table consists of attribute names, whereas the instance level consists of the values of these attributes.

Dimension Tables:

person_info			
person	position	degree	yob

article_info					
article	author	title	forum	year	cs_class

forum_info			
forum	type	publisher	refereed

Figure 6.15

Hierarchical Tables

Hierarchy tables are always associated with dimension attributes, while measure attributes do not have hierarchy tables at all. Hierarchy tables support the analysis of information in data cubes at different levels of detail. Unlike some other approaches, we do not represent hierarchy levels in dimension tables – they would contain different types of information requiring different types of processing. The model allows multiple hierarchies for each dimension, making this separation even more important. Figure 6.16 presents the schema of our sample multidimensional hierarchical tables.

Hierarchical Tables:

Dimension attribute	Hierarchy level	Immediate Sub-hierarchy level
domain	discipline	domains

Dimension attribute	Hierarchy level	Immediate Sub-hierarchy level
person	country	institution
person	institution	author

Dimension attribute	Hierarchy level	Immediate Sub-hierarchy level
year	all_times	years

Figure 6.16

6.4 OLAP SERVER ARCHITECTURES

We describe here the physical implementation of an OLAP server in a Data Warehouse. There are three different possible designs:

1. Relational OLAP (ROLAP)
2. Multidimensional OLAP (MOLAP)
3. Hybrid OLAP (HOLAP)

6.4.1 ROLAP

ROLAP stores the data based on the already familiar relational DBMS technology. In this case, data and the related aggregations are stored in RDBMS, and OLAP middleware is used to implement handling and exploration of data cubes. This architecture focuses on the optimisation of the RDBMS back end and provides additional tools and services such as data cube navigation logic. Due to the use of the RDBMS back end, the main advantage of ROLAP is scalability in handling large data volumes. Example ROLAP engines include the commercial IBM Informix Metacube (www.ibm.com) and the Micro-strategy DSS server (www.microstrategy.com), as well as the open-source product Mondrian (mondrian.sourceforge.net).

6.4.2 MOLAP

In contrast to ROLAP, which uses tuples as the data storage unit, the MOLAP uses a dedicated n-dimensional array storage engine and OLAP middleware to manage data. Therefore, OLAP queries are realised through a direct addressing to the related multidimensional views (data cubes). Additionally, this architecture focuses on pre-calculation of the transactional data into the aggregations, which results in fast query execution performance. More specifically, MOLAP precalculates and stores aggregated measures at every hierarchy level at load time, and stores and indexes these values for immediate retrieval. The full precalculation requires a substantial amount of overhead, both in processing time and in storage space. For sparse data, MOLAP uses sparse matrix compression algorithms to improve storage utilization, and thus in general is characterised by smaller on-disk size of data in comparison with data stored in RDBMS. Example MOLAP products are the commercial Hyperion Ebasse (www.hyperion.com) and the Applix TM1 (www.applix.com), as well as Palo (www.opensourceolap.org), which is an open-source product.

6.4.3 HOLAP

To achieve a tradeoff between ROLAP's scalability and MOLAP's query performance, many commercial OLAP servers are based on the HOLAP approach. In this case, the user decides which portion of the data to store in the MOLAP and which in the ROLAP. For instance, often the low-level data are stored using a relational database, while higher-level data, such as aggregations, are stored in a separate MOLAP. An example product that supports all three architectures is Microsoft's OLAP Services (www.microsoft.com/), which is part of the company's SQL Server.

6.5 FURTHER DEVELOPMENT OF DATA CUBE TECHNOLOGY

On-Line Analytical Processing (OLAP) characterizes the operations of summarizing, consolidating, viewing, applying formulae to, and synthesizing data along multiple dimensions. OLAP software helps analysts and managers gain insight into the performance of an enterprise through a wide variety of views of data organized to reflect the multi-dimensional nature of enterprise data. An increasingly popular data model for OLAP applications is the multi-dimensional database, also known as the data cube. We have discussed below Hypothesis-driven exploration and discovery-driven exploration of data cubes which are also known as **Fuzzy logic and Fuzzy sets** respectively.

6.5.1 Hypothesis-driven Exploration: Fuzzy Logic

A user or analyst can search for interesting patterns in the cube by specifying a number of OLAP operations, such as drill-down, roll-up, slice, and dice. While these tools are available to help the user

explore the data, the discovery process is not automated. It is the user who, following her own intuition or hypotheses, tries to recognise exceptions or anomalies in the data. This hypothesis-driven exploration has a number of disadvantages. The search space can be very large, making manual inspection of the data a daunting and overwhelming task. High-level aggregations may give no indication of anomalies at lower levels, making it easy to overlook interesting patterns. Even when looking at a subset of the cube, such as a slice, the user is typically faced with many data values to examine. The sheer volume of data values alone makes it easy for users to miss exceptions in the data if using hypothesis-driven exploration.

6.5.2 Discovery-driven Exploration of Data Cubes: Fuzzy Sets

A new discovery-driven method of data exploration overcomes the anomalies of hypothesis-driven exploration. In this method, analyst's search for anomalies is guided by precomputed indicators of exceptions at various levels of detail in the cube. This increases the chances of user noticing abnormal patterns in the data at any level of aggregation.

Intuitively, an exception is a data cube cell value that is significantly different from the value anticipated, based on a statistical model. The model considers variations and patterns in the measure value across *all of the dimensions* to which a cell belongs. For example, if the analysis of item-sales data reveals an increase in sales in December in comparison to all other months, this may seem like an exception in the time dimension. However, it is not an exception if the item dimension is considered, since there is a similar increase in sales for other items during December. The model considers exceptions hidden at all aggregated group-by's of a data cube. Visual cues such as background color are used to reflect the degree of exception of each cell, based on the precomputed exception indicators. The computation of exception indicators can be overlapped with cube construction, so that the overall construction of data cubes for discovery-driven exploration is efficient.

Three measures are used as exception indicators to help identify data anomalies. These measures indicate the degree of surprise that the quantity in a cell holds, with respect to its expected value. The measures are computed and associated with every cell, for all levels of aggregation. They are:

- ***SelfExp:*** This indicates the degree of surprise of the cell value, relative to other cells at the same level of aggregation.
- ***InExp:*** This indicates the degree of surprise somewhere beneath the cell, if we were to drill down from it.
- ***PathExp:*** This indicates the degree of surprise for each drill-down path from the cell.

The use of these measures for discovery-driven exploration of data cubes is illustrated in the following example.

Example: Consider a user looking at the monthly sales as a percentage difference from the previous month. Suppose the user starts by viewing the data aggregated over all products and markets for different months of the year as shown in Figure 6.17

Product	All
Region	(All)

Sum of Sales	Month											
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Total		2%	0%	2%	2%	4%	3%	0%	-8%	0%	-3%	4%

Figure 6.17: Change in Sales over Time

To find out what parts of the cube may be worthy of exploring further in terms of exceptions, the user invokes a "highlight exceptions" button that colors the background of each cell based on its SelfExp value. In addition, each cell is surrounded with a different colored box based on the InExp value. In both cases, the intensity of the color is varied with the degree of exception. In Figure 6.17, the months with a thick box around them have a high InExp value and thus need to be drilled down further for exceptions underneath them. Darker boxes (e.g., around "Aug", "Sep" and "Oct") indicate higher values of InExp than the lighter boxes (e.g., around "Feb" and "Nov").

There are two paths the user may drill down along from here: Product and Region. To evaluate which of these paths has more exceptions, the user selects a cell of interest and invokes a "path exception" module that colors each aggregated dimension based on the surprise value along that path. These are based on the PathExp values of the cell. In Figure 6.17 (top-left part) the path along dimension Product has more surprise than along Region indicated by darker color.

Drilling-down along Product yields 143 different sales values corresponding to different Product-Time combinations as shown in Figure 6.18.

Region	(All)											
Avg.Sales	Month											
Product	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Birch-B		10%	-7%	3%	-4%	15%	-12%	-3%	1%		-14%	-10%
Chery-S		1%	1%	4%	3%	5%	5%	-9%	-12%	1%	-5%	5%
Cola		-1%	2%	3%	4%	9%	4%	1%	-11%	-8%	-2%	7%
Cream-S		3%	1%	6%	3%	3%	8%	-3%	-12%	-2%	1%	10%
Diet-B		1%	1%	-1%	2%	1%	2%	0%	-6%	-1%	-4%	2%
Diet-C		3%	2%	5%	2%	4%	7%	-7%	-12%	-2%	-2%	8%
Diet-S		2%	-1%	0%	0%	4%	2%	4%	-9%	5%	-3%	0%
Grape-S		1%	1%	0%	4%	5%	1%	3%	-9%	-1%	-8%	4%
Jolt-C		-1%	-4%	2%	2%	0%	-4%	2%	6%	-2%	0%	0%
Kiwi-S		2%	1%	4%	1%	-1%	3%	-1%	-4%	4%	0%	1%
Old-B		4%	-1%	0%	1%	5%	2%	7%	-10%	3%	-3%	1%
Orang-S		1%	1%	3%	4%	2%	1%	-1%	-1%	-6%	-4%	9%
Saspria		-1%	2%	1%	3%	-3%	5%	-10%	-2%	-1%	1%	5%

Figure 6.18: Change in Sales Over Time for each Product

Instead of trying to find the exceptions by manual inspection, the user can click on the "highlight exception" button to quickly identify the exceptional values. In this figure there are a few cells with high SelfExp values and these appear as cells with a different background shade than the normal ones (darker shades indicate higher surprise). For instance, sales of "Birch-B(eer)" shows an exceptional difference of 42% in the month of "Oct". In addition, three other cells are also indicated to have large SelfExp values although the sales values themselves (6% for <Jolt-C, Sep>, -12% for <Birch-B, Jul> and -10% for <Birch-B, Dec>) are not exceptionally large when compared with all the other cells.

Figure 6.19 also shows some cells with large InExp values as indicated by the thick boxes around them. The highest InExp values are for Product "Diet-Soda" in the months of "Aug" and "Oct". The user may therefore choose to explore further details for "Diet-Soda" by drilling down along Region. Figure 2.16 shows the sales figures for "Diet-Soda" in different Regions. By highlighting exceptions in this plane, the user notices that in Region "E" (for Eastern), the sales of "Diet-Soda" has decreased by an exceptionally high value of 40% and 33% in the months of "Aug" and "Oct" respectively. Notice that

the sales of "Diet-Soda" in the Product-Time plane aggregated over different Regions gives little indication of these high exceptions in the Region-Product-Time space. This shows how the InExp value at higher level cells may be valuable in reaching at exceptions in lower level cells.

Product	Diet-S										
Avg.Sales	Month										
Region	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
C		0%	-2%	0%	1%	4%	1%	5%	-6%	2%	-2%	-2%
E		0%	2%	-8%	7%	0%	5%	[illegible]	10%	-33%	2%	8%
S		0%	-1%	3%	-2%	2%	-2%	19%	-1%	12%	-1%	0%
W		5%	1%	0%	-2%	6%	6%	2%	-17%	9%	-7%	2%

Figure 6.19: Change in sales of Product "Diet-Soda" over time in each Region

There are no other cells with high InExp in Figure 6.19 Therefore, the user may stop drilling down and go back to the Product-Time plane of Figure 6.18 to explore other cells with high InExp. Suppose, he chooses to drill-down along Product "Cola" in "Aug". Figure 6.20 shows the exceptions for "Cola" after drilling down along Region. The "Central" Region has a large InExp and may be drilled down further, revealing the SelfExp values in the Market-time plane for "Cola".

Market	(All)											
Product	Cola											
Avg.Sales	Month											
Region	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
C		3%	1%	4%	1%	4%	10%	-11%	-14%	-3%	5%	11%
E		-3%	3%	4%	4%	13%	2%	0%	-10%	-13%	-3%	8%
S		2%	-1%	1%	9%	6%	3%	21%	-15%	1%	-5%	4%
W		-2%	2%	2%	4%	12%	1%	1%	-9%	-11%	-4%	6%

Figure 6.20: Change in sales over Time for Product "Cola" in different Region

6.5.3 Defining Exceptions

The SelfExp, InExp, and PathExp measures are based on a statistical method for table analysis. They take into account the entire group-by (aggregations) in which a given cell value participates. A cell value is considered an exception based on how much it differs from its expected value, where its expected value is determined with a statistical model described below. The difference between a given cell value and its expected value is called a residual. Intuitively, the larger the residual, the more the given cell value is an exception. The comparison of residual values requires us to scale the values based on the expected standard deviation associated with the residuals. A cell value is therefore considered an exception if its scaled residual value exceeds a pre-specified threshold. The SelfExp, InExp, and PathExp measures are based on this scaled residual.

6.5.4 Complex Aggregation at Multiple Granularities: Multifeature Cubes

Data cubes facilitate the answering of data mining queries as they allow the computation of aggregate data at multiple levels of granularity. M*ultifeature cubes* compute complex queries involving multiple dependent aggregates at multiple granularities. These cubes are very useful in practice. Many complex data mining queries can be answered by multifeature cubes without any significant increase in

computational cost, in comparison to cube computation for simple queries with standard data cubes.

6.6 MAPPING A MULTIDIMENSIONAL MODEL TO A RELATIONAL MODEL

Relational approaches for implementing a multidimensional model are:

1. Data Warehouses are usually implemented in relational databases. They mostly use OLAP systems to manipulate data through roll-up and drill-down operations.
2. Relational databases use well-known strategies of data storage, are well standardized, tool independent, and readily available. In contrast, there is no consensus in the research community for defining a logical level for OLAP systems and additionally multidimensional OLAP (MOLAP) systems differ greatly in the structures used for data storage.
3. Much research has been done in relational databases for improving optimization techniques, indexing, join operations, and view materialization.
4. The relational model allows a logical-level representation of the different kinds of hierarchies. This would facilitate the subsequent development of MOLAP systems according to the specific data storage structures.
5. Even though MOLAP systems are frequently used for storage purposes, many commercial and prototype systems use ROLAP (relational OLAP) and HOLAP (Hybrid OLAP: a combination of ROLAP and MOLAP systems) for analysis.
6. Unlike relational databases, current OLAP tools impose limitations on the number of dimensions and the size of historical data that can be stored.
7. Commercial relational database systems include extensions to represent and manage multidimensional views of data.

Relational databases are an accepted resource layer in the Common Warehouse Model (CWM) from the Object Management Group (OMG).

Check Your Progress

Fill in the blanks:

1. Data warehouses and ..... tools are based on a multidimensional data model.
2. Data Warehouses (DW), ...Databases, and OLAP applications are used in conjunction to form a highly powerful mechanism for discovering crucial business information in strategic decision-making processes.

6.7 LET US SUM UP

A multidimensional data model is typically used for the design of corporate *data warehouses* and *departmental data marts.* Such a model can adopt either a *star schema, snowflake schema,* or *fact constellation schema.* The core of the *multidimensional model* is the data cube, which consists of a large

set of *facts* (or *measures)* and a number of *dimensions.* Dimensions are the entities or perspectives with respect to which an organization wants to keep records, and are hierarchical in nature.

Concept hierarchies organize the values of attributes or dimensions into gradual levels of abstraction. They are useful in mining at multiple levels of abstraction.

On-Line Analytical Processing (OLAP) can be performed in data warehouses/marts using the multidimensional data model. Typical OLAP operations include *roll-up, drill-(down, cross, through), slice-and-dice, pivot (rotate),* as well as statistical operations such as ranking, computing moving averages and growth rates, etc. OLAP operations can be implemented efficiently using the data cube structure.

Data warehouses often adopt a three-tier architecture. The bottom tier is a *warehouse database server,* which is typically a relational database system. The middle tier is an *OLAP server,* and the top tier is a *client,* containing query and reporting tools.

OLAP servers may use Relational OLAP (ROLAP), or Multidimensional OLAP (MOLAP), or Hybrid OLAP (HOLAP). A ROLAP server uses an extended relational DBMS that maps OLAP operations on multi-dimensional data to standard relational operations. A MOLAP server maps multi-dimensional data views directly to array structures. A HOLAP server combines ROLAP and MOLAP. For example, it may use ROLAP for historical data while maintaining frequently accessed data in a separate MOLAP store.

6.8 KEYWORDS

Data Warehouse: A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process.

OLTP: On Line Transactional Processing

OLAP: On Line Analytical Processing

Logical Cubes: Logical cubes provide a means of organizing measures that have the same shape, that is, they have the exact same dimensions.

Logical Measures: Measures populate the cells of a logical cube with the facts collected about business operations. Measures are organized by dimensions, which typically include a Time dimension.

Logical Dimensions: Dimensions contain a set of unique values that identify and categorize data. They form the edges of a logical cube, and thus of the measures within the cube.

Logical Hierarchies: A hierarchy is a way to organize data at different levels of aggregation.

Star Schema: A star schema is a convention for organizing the data into dimension tables, fact tables, and materialized views.

Snowflake Schema: The snowflake schema is a variant of the star schema model, where some dimension tables are *normalized,* thereby further splitting the data into additional tables.

Fact Constellation: Sophisticated applications may require multiple fact tables to *share* dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.

Concept Hierarchy: A concept hierarchy defines a sequence of mappings from a set of low level concepts to higher level, more general concepts.

ROLL UP: It is used to navigate to lower levels of details for a given data cube.

DRILL DOWN: It is used to navigate to higher levels of detail.

SLICE: It provides a cut through a given data cube.

DICE: It provides just one cell from the cube.

PIVOT: It rotates the cube to change the perspective, e.g., the "time item" perspective may be changed into "time location."

Relational OLAP (ROLAP) Model: It is an extended relational DBMS that maps operations on multidimensional data to standard relational operations.

Multidimensional OLAP (MOLAP) Model: It is a special purpose server that directly implements multidimensional data and operations.

6.8 QUESTIONS FOR DISCUSSION

1. What is the composition of multidimensional data model? Also draw a figure to show the relationships among the logical objects.
2. Explain five basic OLAP commands that are used to perform data retrieval from a Data warehouse.
3. Differentiate between the ROLAP, MOLAP and HOLAP.
4. Write a short note on "Further development of data cube technology."
5. What is Dimensional Modeling? Discuss dimensional modeling queries.
6. Discuss various approaches for implementing a multidimensional model.

Check Your Progress: Model Answers

1. OLAP
2. Multidimensional (MD)

6.9 SUGGESTED READINGS

Jiawei Han, Micheline Kamber, *Data Mining - Concepts and Techniques*, Morgan Kaufmann Publishers, First Edition, 2003.

Michael J. A. Berry, Gordon S Linoff, *Data Mining Techniques*, Wiley Publishing Inc, Second Edition, 2004.

Alex Berson, Stephen J. Smith, *Data warehousing, Data Mining & OLAP*, Tata McGraw Hill, Publications, 2004.

Sushmita Mitra, Tinku Acharya, *Data Mining - Multimedia, Soft Computing and Bioinformatics*, John Wiley & Sons, 2003.

Sam Anohory, Dennis Murray, *Data Warehousing in the Real World*, Addison Wesley, First Edition, 2000.

LESSON

7

DATA MINING TECHNIQUES

CONTENTS

Contd...

7.0 AIMS AND OBJECTIVE

After studying this lesson, you will be able to:

- Describe classical techniques of data mining
- Explain concepts of histogram, linear regression and clustering
- Describe the use of decision trees
- State concept of neural network
- Explain the concept of rule induction

7.1 INTRODUCTION

This overview provides a description of some of the most common data mining algorithms in use today. We have broken the discussion into two sections, each with a specific theme:

- *Classical Techniques:* Statistics, Neighborhoods and Clustering
- *Next Generation Techniques:* Trees, Networks and Rules

Each section will describe a number of data mining algorithms at a high level, focusing on the "big picture" so that the reader will be able to understand how each algorithm fits into the landscape of data mining techniques. Overall, six broad classes of data mining algorithms are covered. Although there are a number of other algorithms and many variations of the techniques described, one of the algorithms from this group of six is almost always used in real world deployments of data mining systems.

7.2 CLASSICAL TECHNIQUES

These two sections have been broken up based on when the data mining technique was developed and when it became technically mature enough to be used for business, especially for aiding in the optimization of customer relationship management systems. Thus this section contains descriptions of techniques that have classically been used for decades the next section represents techniques that have only been widely used since the early 1980s. The three classical techniques are discussed in the following sections.

7.3 STATISTICS

By strict definition "statistics" or statistical techniques are not data mining. They were being used long before the term data mining was coined to apply to business applications. However, statistical techniques are driven by the data and are used to discover patterns and build predictive models. And from the users perspective you will be faced with a conscious choice when solving a "data mining" problem as to whether you wish to attack it with statistical methods or other data mining techniques. For this reason it is important to have some idea of how statistical techniques work and how they can be applied.

Statistics is a branch of mathematics concerning the collection and the description of data. Usually statistics is considered to be one of those scary topics in college right up there with chemistry and physics. However, statistics is probably a much friendlier branch of mathematics because it really can be used every day. Statistics was in fact born from very humble beginnings of real world problems from business, biology, and gambling!

7.3.1 Data, Counting and Probability

One thing that is always true about statistics is that there is always data involved, and usually enough data so that the average person cannot keep track of all the data in their heads. This is certainly truer today than it was when the basic ideas of probability and statistics were being formulated and refined early this century. Today people have to deal with up to terabytes of data and have to make sense of it and glean the important patterns from it. Statistics can help greatly in this process by helping to answer several important questions about your data:

- What patterns are there in my database?
- What is the chance that an event will occur?
- Which patterns are significant?
- What is a high level summary of the data that gives me some idea of what is contained in my database?

Certainly statistics can do more than answer these questions but for most people today these are the questions that statistics can help answer.

Example: A pie chart reporting the number of US citizens of different eye colors, or the average number of annual doctor visits for people of different ages. Statistics at this level is used in the reporting of important information from which people may be able to make useful decisions. There are many different parts of statistics but the idea of collecting data and counting it is often at the base

of even these more sophisticated techniques. The first step then in understanding statistics is to understand how the data is collected into a higher-level form - one of the most notable ways of doing this is with the histogram.

7.3.2 Histograms

One of the best ways to summarize data is to provide a histogram of the data. In the simple example database shown in Table 7.1 we can create a histogram of eye color by counting the number of occurrences of different colors of eyes in our database. For this example database of 10 records this is fairly easy to do and the results are only slightly more interesting than the database itself. However, for a database of many more records this is a very useful way of getting a high level understanding of the database.

Table 7.1: An Example Database of Customers with Different Predictor Types

ID	Name	Prediction	Age	Balance	Income	Eyes	Gender
1	Amy	No	62	$0	Medium	Brown	F
2	Al	No	53	$1,800	Medium	Green	M
3	Betty	No	47	$16,543	High	Brown	F
4	Bob	Yes	32	$45	Medium	Green	M
5	Carla	Yes	21	$2,300	High	Blue	F
6	Carl	No	27	$5,400	High	Brown	M
7	Donna	Yes	50	$165	Low	Blue	F
8	Don	Yes	46	$0	High	Blue	M
9	Edna	Yes	27	$500	Low	Blue	F
10	Ed	No	68	$1,200	Low	Blue	M

A simple predictor (eye color) which will have only a few different values no matter if there are 100 customer records in the database or 100 million. There are, however, other predictors that have many more distinct values and can create a much more complex histogram.

When there are many values for a given predictor the histogram begins to look smoother and smoother (compare the difference between the two histograms above). Sometimes the shape of the distribution of data can be calculated by an equation rather than just represented by the histogram. This is what is called a data distribution. Like a histogram a data distribution can be described by a variety of statistics. In classical statistics the belief is that there is some "true" underlying shape to the data distribution that would be formed if all possible data was collected. The shape of the data distribution can be calculated for some simple examples. The statistician's job then is to take the limited data that may have been collected and from that make their best guess at what the "true" or at least most likely underlying data distribution might be.

7.3.3 Linear Regression

In statistics prediction is usually synonymous with regression of some form. There are a variety of different types of regression in statistics but the basic idea is that a model is created that maps values from predictors in such a way that the lowest error occurs in making a prediction. The simplest form

of regression is simple linear regression that just contains one predictor and a prediction. The relationship between the two can be mapped on a two dimensional space and the records plotted for the prediction values along the Y axis and the predictor values along the X axis. The simple linear regression model then could be viewed as the line that minimized the error rate between the actual prediction value and the point on the line (the prediction from the model). The simplest form of regression seeks to build a predictive model that is a line that maps between each predictor value to a prediction value. Of the many possible lines that could be drawn through the data the one that minimizes the distance between the line and the data points is the one that is chosen for the predictive model.

On average if you guess the value on the line it should represent an acceptable compromise amongst all the data at that point giving conflicting answers. Likewise if there is no data available for a particular input value the line will provide the best guess at a reasonable answer based on similar data. The line will take a given value for a predictor and map it into a given value for a prediction. The actual equation would look something like: Prediction = a + b * Predictor. Which is just the equation for a line Y = a + bX.

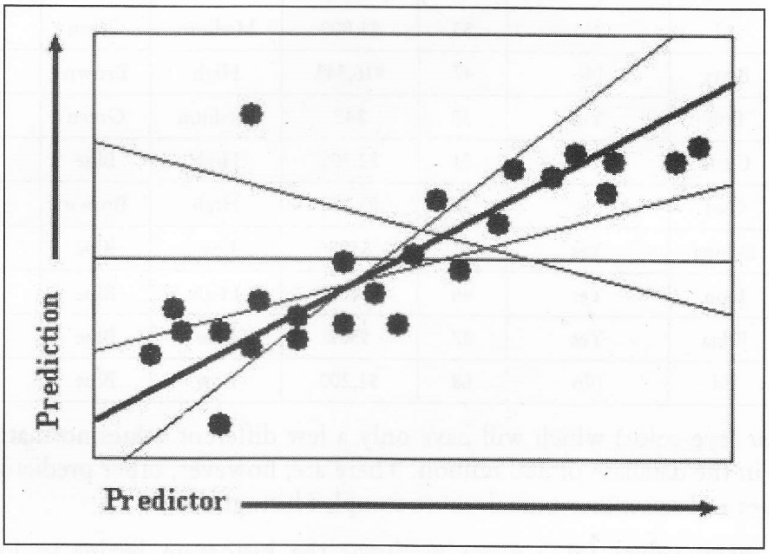


Figure 7.1

What if the pattern in my data doesn't look like a straight line?

Regression can become more complicated than the simple linear regression we've introduced so far. It can get more complicated in a variety of different ways in order to better model particular database problems. There are, however, three main modifications that can be made:

1. More predictors than just one can be used.
2. Transformations can be applied to the predictors.
3. Predictors can be multiplied together and used as terms in the equation.
4. Modifications can be made to accommodate response predictions that just have yes/no or 0/1 values.

Adding more predictors to the linear equation can produce more complicated lines that take more information into account and hence make a better prediction. This is called multiple linear regression and might have an equation like the following if 5 predictors were used (X1, X2, X3, X4, X5):

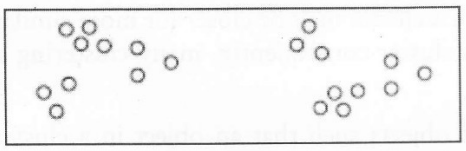
$Y = a + b1(X1) + b2(X2) + b3(X3) + b4(X4) + b5(X5)$

This equation still describes a line but it is now a line in a6 dimensional space rather than the two dimensional space.

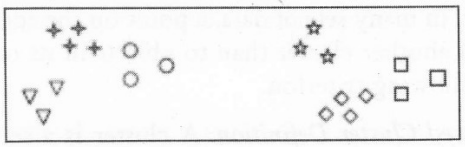
By transforming the predictors by squaring, cubing or taking their square root it is possible to use the same general regression methodology and now create much more complex models that are no longer simple shaped like lines. This is called non-linear regression. A model of just one predictor might look like this: $Y = a + b1(X1) + b2 (X12)$. In many real world cases analysts will perform a wide variety of transformations on their data just to try them out. If they do not contribute to a useful model their coefficients in the equation will tend toward zero and then they can be removed. The other transformation of predictor values that is often performed is multiplying them together. For instance a new predictor created by dividing hourly wage by the minimum wage might be a much more effective predictor than hourly wage by itself.

7.4 CLUSTERING

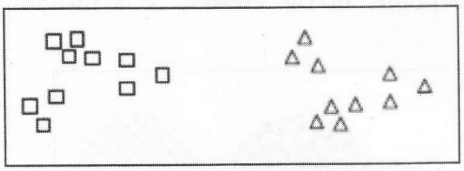
Cluster analysis groups objects (observations, events) based on the information found in the data describing the objects or their relationships. The goal is that the objects in a group will be similar (or related) to one other and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group, and the greater the difference between groups, the "better" or more distinct the clustering.



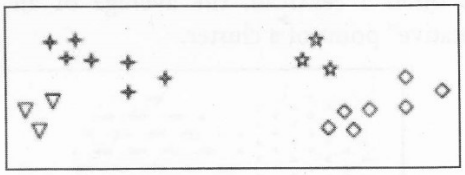
Initial Point



Six Cluster



Two Cluster



Four Cluster

To better understand the difficulty of deciding what constitutes a cluster which show twenty points and three different ways that they can be divided into clusters. If we allow clusters to be nested, then the most reasonable interpretation of the structure of these points is that there are two clusters, each of which has three subclusters. However, the apparent division of the two larger clusters into three subclusters may simply be an artifact of the human visual system. Finally, it may not be unreasonable to say that the points form four clusters. Thus, we stress once again that the definition of what constitutes a cluster is imprecise, and the best definition depends on the type of data and the desired results.

7.4.1 Overview of Cluster Analysis

- Cluster analysis is an exploratory data analysis tool for solving classification problems.
- Cluster analysis is a tool of discovery.
- Cluster analysis leads the researcher to classification, definition of structure, new knowledge, and finally discovery.

Definitions

Well-separated Cluster Definition: A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster. Sometimes a threshold is used to specify that all the points in a cluster must be sufficiently close (or similar) to one another.

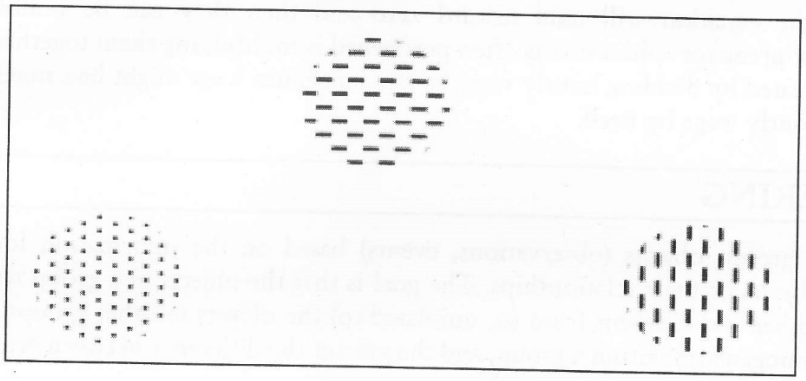


Figure 7.2

However, in many sets of data, a point on the edge of a cluster may be closer (or more similar) to some objects in another cluster than to objects in its own cluster consequently, many clustering algorithms use the following criterion.

Center-based Cluster Definition: A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster. The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster.

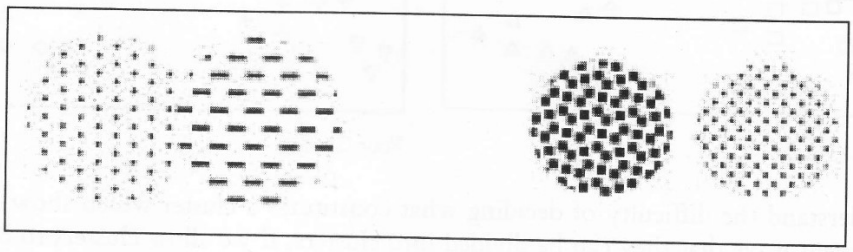


Figure 7.3

Contiguous Cluster Definition (Nearest Neighbor or Transitive Clustering): A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.

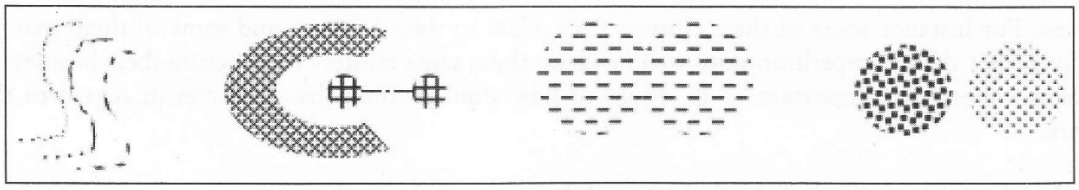


Figure 7.4

Density-based Definition: A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density. This definition is more often used when the clusters are irregular or intertwined, and when noise and outliers are present. Note that the contiguous definition would find only one cluster in Figure 7.5. Also note that the three curves don't form clusters since they fade into the noise, as does the bridge between the two small circular clusters.

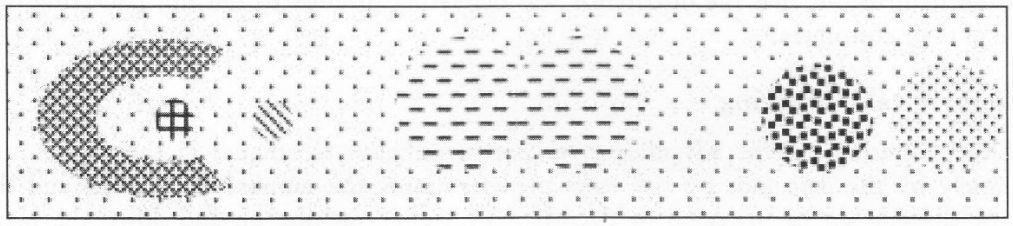


Figure 7.5

Similarity-based Cluster Definition: A cluster is a set of objects that are "similar", and objects in other clusters are not "similar." A variation on this is to define a cluster as a set of points that together create a region with a uniform local property, e.g., density or shape.

7.4.2 Clustering for Clarity

Clustering is the method by which like records are grouped together. Usually this is done to give the end user a high level view of what is going on in the database. Clustering is sometimes used to mean segmentation - which most marketing people will tell you is useful for coming up with a birds eye view of the business. Two of these clustering systems are the PRIZM™ system from Claritas corporation and MicroVision™ from Equifax corporation. These companies have grouped the population by demographic information into segments that they believe are useful for direct marketing and sales. To build these groupings they use information such as income, age, occupation, housing and race collect in the US Census. Then they assign memorable "nicknames" to the clusters. Some examples are shown in Table 7.2.

Table 7.2

Name	Income	Age	Education	Vendor
Blue Blood Estates	Wealthy	35-54	College	Claritas Prizm™
Shotguns and Pickups	Middle	35-64	High School	Claritas Prizm™

This clustering information is then used by the end user to tag the customers in their database. Once this is done the business user can get a quick high level view of what is happening within the cluster. Once the business user has worked with these codes for some time they also begin to build intuitions about how these different customers clusters will react to the marketing offers particular to their

business. For instance some of these clusters may relate to their business and some of them may not. But given that their competition may well be using these same clusters to structure their business and marketing offers it is important to be aware of how you customer base behaves in regard to these clusters.

7.4.3 Finding the ones that don't fit in - Clustering for Outliers

Sometimes clustering is performed not so much to keep records together as to make it easier to see when one record sticks out from the rest. *For instance:*

Most wine distributors selling inexpensive wine in Missouri and that ship a certain volume of product produce a certain level of profit. There is a cluster of stores that can be formed with these characteristics. One store stands out, however, as producing significantly lower profit. On closer examination it turns out that the distributor was delivering product to but not collecting payment from one of their customers.

7.4.4 Hierarchical Clustering

Hierarchical clustering has the advantage over non-hierarchical techniques in that the clusters are defined solely by the data (not by the users predetermining the number of clusters) and that the number of clusters can be increased or decreased by simple moving up and down the hierarchy. Figure 7.6 shows the hierarchical clustering.

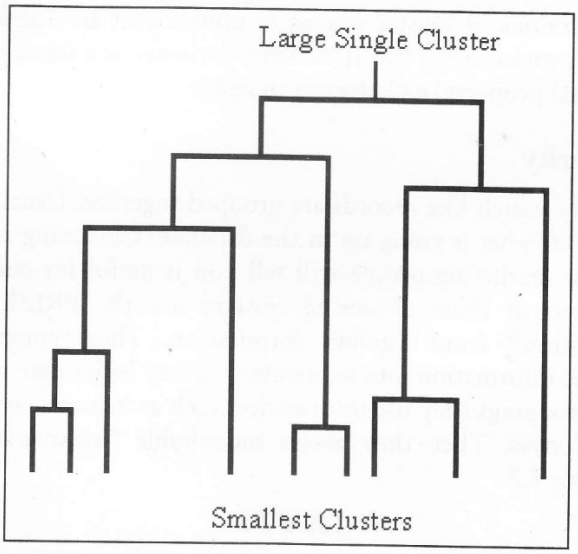


Figure 7.6

Hierarchical techniques produce a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom. Each intermediate level can be viewed as combining (splitting) two clusters from the next lower (next higher) level. (While most hierarchical algorithms involve joining two clusters or splitting a cluster into two sub-clusters, some hierarchical algorithms join more than two clusters in one step or split a cluster into more than two sub-clusters.)

The following figures indicate different ways of graphically viewing the hierarchical clustering process. Figures 7.7 and 7.8 illustrate the more "traditional" view of hierarchical clustering as a process of

merging two clusters or splitting one cluster into two. Figure 7.7 gives a "nested set" representation of the process, while Figure 7.8 shows a "tree' representation, or dendogram. Figure 7.9 and 7.10 show a different, hierarchical clustering, one in which points p1 and p2 are grouped together in the same step that also groups points p3 and p4 together.

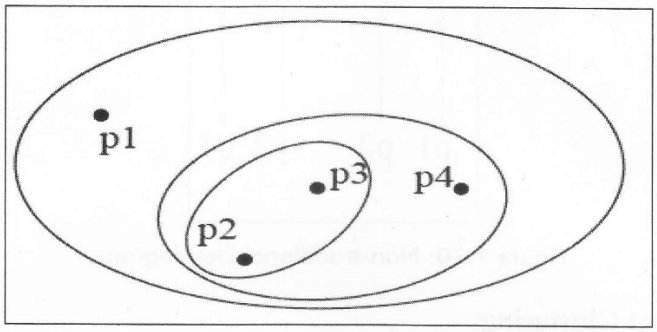


Figure 7.7: Traditional Nested Set

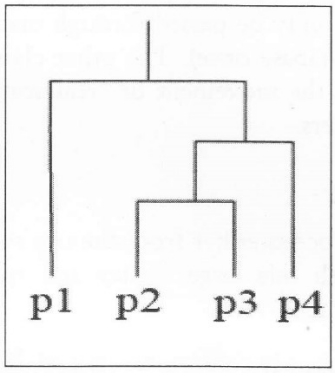


Figure 7.8: Traditional Dendogram

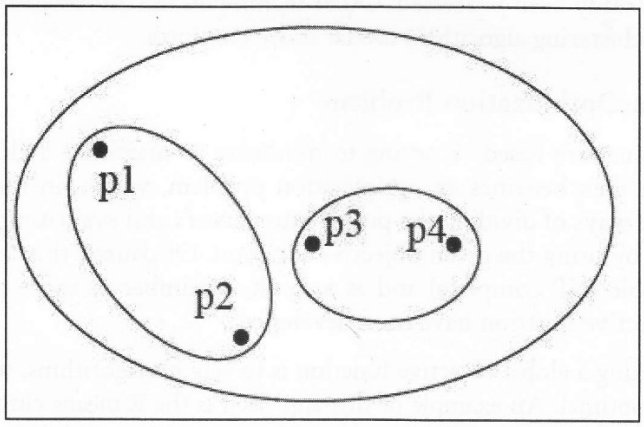


Figure 7.9: Non-traditional Nested Set

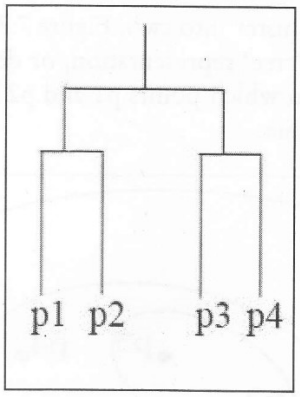


Figure 7.10: Non-traditional Dendogram

7.4.5 Non-hierarchical Clustering

There are two main non-hierarchical clustering techniques. Both of them are very fast to compute on the database but have some drawbacks. The first are the single pass methods. They derive their name from the fact that the database must only be passed through once in order to create the clusters (i.e. each record is only read from the database once). The other class of techniques is called reallocation methods. They get their name from the movement or "reallocation" of records from one cluster to another in order to create better clusters.

7.4.6 Divisive vs. Agglomerative

Hierarchical clustering techniques proceed either from the top to the bottom or from the bottom to the top, i.e., a technique starts with one large cluster and splits it, or starts with clusters each containing a point, and then merges them.

7.4.7 Incremental or Non-incremental

Some clustering techniques work with an item at a time and decide how to cluster it given the current set of points that have already been processed. Other techniques use information about all the points at once. Non-incremental clustering algorithms are far more common.

7.4.8 Clustering as an Optimization Problem

Many clustering techniques are based on trying to minimize or maximize a global objective function. The clustering problem then becomes an optimization problem, which, in theory, can be solved by enumerating all possible ways of dividing the points into clusters and evaluating the "goodness" of each potential set of clusters by using the given objective function. Of course, this "exhaustive" approach is computationally infeasible (NP complete) and as a result, a number of more practical techniques for optimizing a global objective function have been developed.

One approach to optimizing a global objective function is to rely on algorithms, which find solutions that are often good, but not optimal. An example of this approach is the K-means clustering algorithm, which tries to minimize the sum of the squared distances (error) between objects and their cluster centers.

Another approach is to fit the data to a model. An example of such techniques is mixture models, which assume that the data is a "mixture" of a number of underlying statistical distributions. These

clustering algorithms seek to find a solution to a clustering problem by finding the maximum likelihood estimate for the statistical parameters that describe the clusters.

Still another approach is to forget about global objective functions. In particular, hierarchical clustering procedures proceed by making local decisions at each step of the clustering process. These 'local' or 'per-step' decisions are also based on an objective function, but the overall or global result is not easily interpreted in terms of a global objective function.

7.5 SPECIFIC CLUSTERING TECHNIQUES

7.5.1 Center-based Partitional Clustering

As described earlier, partitional clustering techniques create a one-level partitioning of the data points. There are a number of such techniques, but we shall only describe two approaches in this section: K-means and K-medoid.

Both these techniques are based on the idea that a center point can represent a cluster. For K-means we use the notion of a centroid, which is the mean or median point of a group of points. Note that a centroid almost never corresponds to an actual data point. For K-medoid we use the notion of a medoid, which is the most representative (central) point of a group of points. By its definition a medoid is required to be an actual data point.

K-Means Clustering

Basic Algorithm

The K-means clustering technique is very simple and we immediately begin with a description of the basic algorithm. We elaborate in the following sections.

Basic K-means Algorithm for finding K Clusters

1. Select *K* points as the initial centroids.
2. Assign all points to the closest centroid.
3. Recompute the centroid of each cluster.
4. Repeat steps 2 and 3 until the centroids don't change.

In the absence of numerical problems, this procedure always converges to a solution, although the solution is typically a local minimum. The following diagram gives an example of this. Figure 7.11 shows the case when the cluster centers coincide with the circle centers. This is a global minimum. Figure 7.12 shows a local minima.

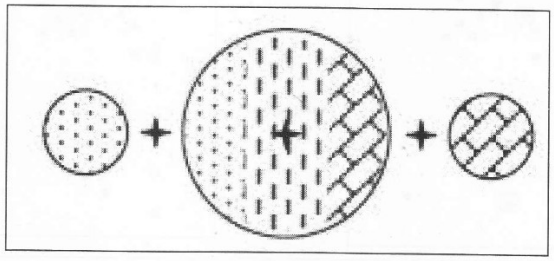


Figure 7.11: A Globally Minimal Clustering Solution

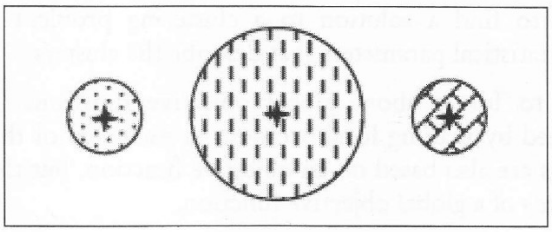


Figure 7.12: A Locally Minimal Clustering Solution

Time and Space Complexity

Since only the vectors are stored, the space requirements are basically *O (mn)*, where *m* is the number of points and *n* is the number of attributes. The time requirements are *O(I*K*m*n)*, where *I* is the number of iterations required for convergence. *I* is typically small and can be easily bounded as most changes occur in the first few iterations. Thus, K-means is linear in *m*, the number of points, and is efficient, as well as simple, as long as the number of clusters is significantly less than *m*.

Choosing Initial Centroids

Choosing the proper initial centroids is the key step of the basic K-means procedure. It is easy and efficient to choose initial centroids randomly, but the results are often poor. It is possible to perform multiple runs, each with a different set of randomly chosen initial centroids – one study advocates 30 – but this may still not work depending on the data set and the number of clusters sought. We start with a very simple example of three clusters and 16 points. Figure 7.13 indicates the "natural" clustering that results when the initial centroids are "well" distributed. Figure 7.14 indicates a "less natural" clustering that happens when the initial centroids are poorly chosen.

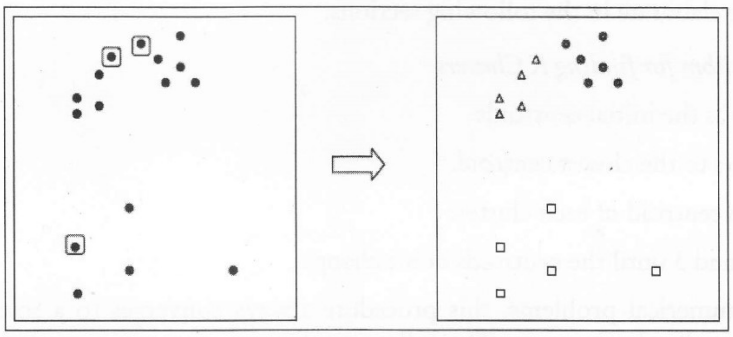


Figure 7.13: Good Starting Centroids and Natural Clustering

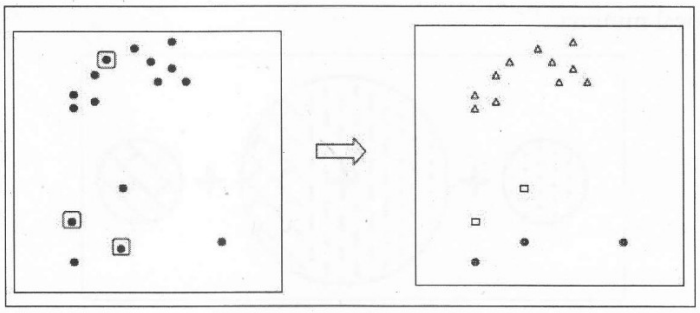


Figure 7.14: Bad Starting Centroids and a "less natural" Clustering

We have also constructed the artificial data set, shown in Figure 7.15 as another illustration of what can go wrong. The figure consists of 10 pairs of circular clusters, where each cluster of a pair of clusters is close to each other, but relatively far from the other clusters. The probability that an initial centroid will come from any given cluster is 0.10, but the probability that each cluster will have exactly one initial centroid is 10!/1010 = 0.00036. (Technical note: This assumes sampling with replacement, i.e., those two initial centroids could be the same point.)

There isn't any problem as long as two initial centroids fall anywhere in a pair of clusters, since the centroids will redistribute themselves, one to each cluster, and so achieve a globally minimal error. However, it is very probable that one pair of clusters will have only one initial centroid. In that case, because the pairs of clusters are far apart, the K-means algorithm will not redistribute the centroids between pairs of clusters, and thus, only a local minima will be achieved. When starting with an uneven distribution of initial centroids as shown in Figure 7.16 we get a non-optimal clustering, as is shown in Figure 7.17, where different fill patterns indicate different clusters. One of the clusters is split into two clusters, while two clusters are joined in a single cluster.

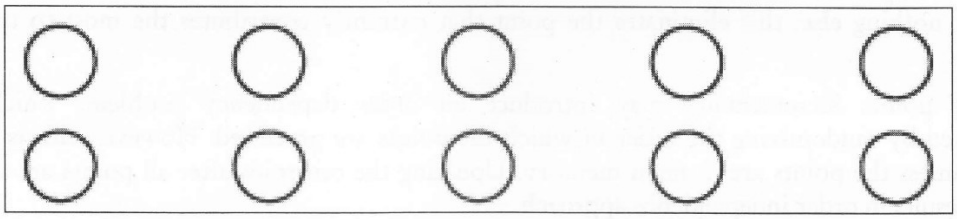


Figure 7.15: Data Distributed in 10 Circular Regions

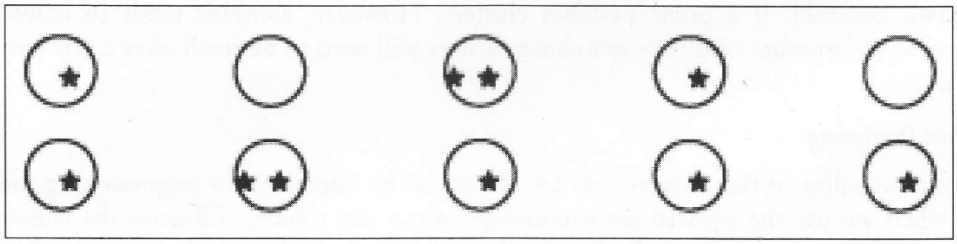


Figure 7.16: Initial Centroids

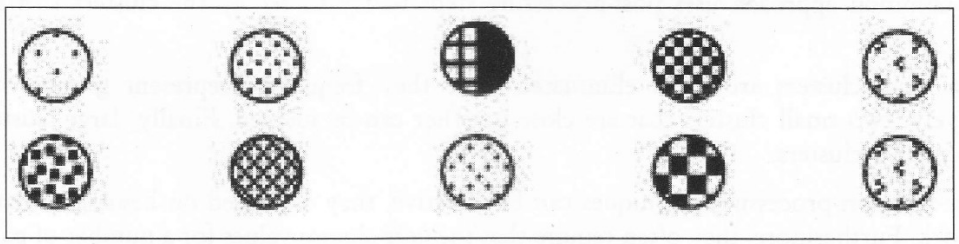


Figure 7.17: K-means Clustering Result

Because random sampling may not cover all clusters, other techniques are often used for finding the initial centroids. For example, initial centroids are often chosen from dense regions, and so that they are well separated, i.e., so that no two centroids are chosen from the same cluster.

Updating Centroids Incrementally

Instead of updating the centroid of a cluster after all points have been assigned to clusters, the centroids can be updated as each point is assigned to a cluster. In addition, the relative weight of the point being added may be adjusted. The goal of these modifications is to achieve better accuracy and faster convergence. However, it may be difficult to make a good choice for the relative weight. These update issues are similar to those of updating weights for artificial neural nets.

Incremental update also has another advantage - empty clusters are not produced. (All clusters start with a single point and if a cluster ever gets down to one point, then that point will always be reassigned to that cluster.) Empty clusters are often observed when centroid updates are performed only after all points have been assigned to clusters.

This imposes the need for a technique to choose a new centroid for an empty cluster, for otherwise the squared error will certainly be larger than it would need to be.

A common approach is to choose as the new centroid the point that is farthest away from any current center. If nothing else, this eliminates the point that currently contributes the most to the squared error.

Updating points incrementally may introduce an order dependency problem, which can be ameliorated by randomizing the order in which the points are processed. However, this is not really feasible unless the points are in main memory. Updating the centroids after all points are assigned to clusters results in order independence approach.

Finally, note that when centers are updated incrementally, each step of the process may require updating two centroids if a point switches clusters. However, K-means tends to converge rather quickly and so the number of points switching clusters will tend to be small after a few passes over all the points.

Pre and Post Processing

Sometimes the quality of the clusters that are found can be improved by preprocessing the data. For example, when we use the squared error criteria, outliers can unduly influence the clusters that are found. Thus, it is common to try to find such values and eliminate them with a preprocessing step.

Another common approach uses post-processing steps to try to fix up the clusters that have been found.

Example: Small clusters are often eliminated since they frequently represent groups of outliers. Alternatively, two small clusters that are close together can be merged. Finally, large clusters can be split into smaller clusters.

While pre-and post-processing techniques can be effective, they are based on heuristics that may not always work. Furthermore, they often require that the user choose values for a number of parameters.

Limitations and Problems

K-means attempts to minimize the squared or absolute error of points with respect to their cluster centroids. While this is sometimes a reasonable criterion and leads to a simple algorithm, K-means, it has a number of limitations and problems. In particular, Figures 7.18 and 7.19 show the problems that result when clusters have widely different sizes or have convex shapes.

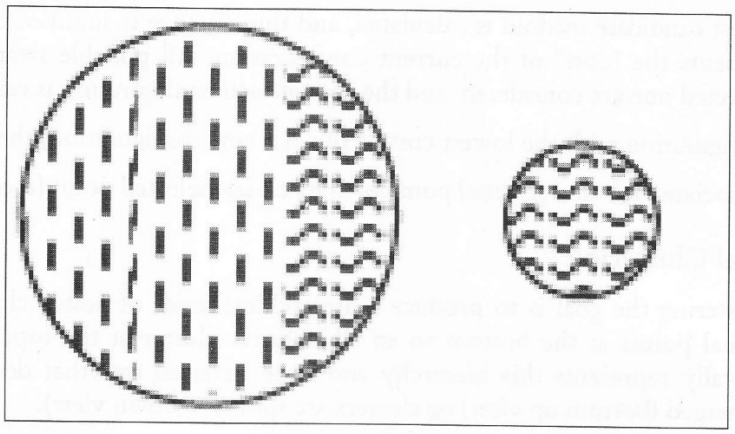


Figure 7.18: Different Size

The difficulty in these two situations is that the K-means objective function is a mismatch for the kind of clusters we are trying to find. The K-means objective function is minimized by globular clusters of equal size or by clusters that are well separated. The K-means algorithm is also normally restricted to data in Euclidean spaces because in many cases the required means and medians do not make sense. (This is not true in all cases, e.g., documents.) A related technique, K-medoid clustering, does not have this restriction and is discussed in the next section.

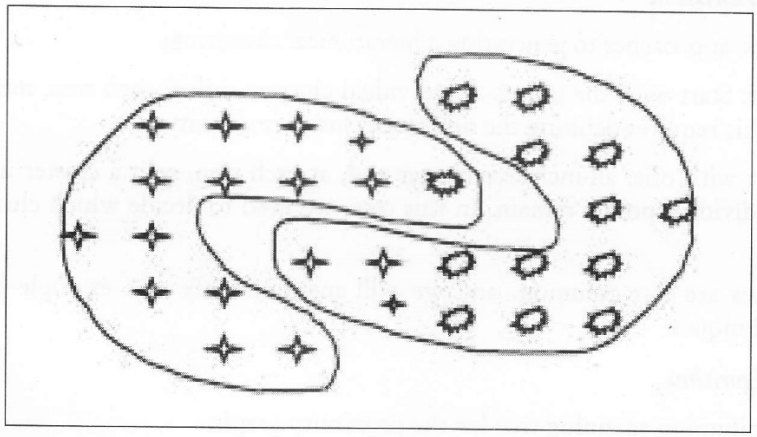


Figure 7.19: Convex Shapes

K-Medoid Clustering

The objective of K-medoid clustering is to find a non-overlapping set of clusters such that each cluster has a most representative point, i.e., a point that is most centrally located with respect to some measure, e.g., distance. These representative points are called medoids.

Basic K-medoid Algorithm for finding K Clusters

1. ***Select K initial points:*** These points are the candidate medoids and are intended to be the most central points of their clusters.
2. Consider the effect of replacing one of the selected objects (medioids) with one of the non-selected objects. Conceptually, this is done in the following way. The distance of each non-selected point

from the closest candidate medoid is calculated, and this distance is summed over all points. This distance represents the "cost" of the current configuration. All possible swaps of a non-selected point for a selected one are considered, and the cost of each configuration is calculated.

3. Select the configuration with the lowest cost. If this is a new configuration, then repeat step 2.
4. Otherwise, associate each non-selected point with its closest selected point (medoid) and stop.

7.5.2 Hierarchical Clustering

In hierarchical clustering the goal is to produce a hierarchical series of nested clusters, ranging from clusters of individual points at the bottom to an all-inclusive cluster at the top. A diagram called a dendogram graphically represents this hierarchy and is an inverted tree that describes the order in which points are merged (bottom-up view) or clusters are split (top-down view).

One of the attractions of hierarchical techniques is that they correspond to taxonomies that are very common in the biological sciences.

Example: Kingdom, phylum, genus, species, ...(Some cluster analysis work occurs under the name of "mathematical taxonomy.") Another attractive feature is that hierarchical techniques do not assume any particular number of clusters. Instead any desired number of clusters can be obtained by "cutting" the dendogram at the proper level.

Agglomeration and Division

There are two basic approaches to generating a hierarchical clustering:

1. *Agglomerative:* Start with the points as individual clusters and, at each step, merge the closest pair of clusters. This requires defining the notion of cluster proximity.
2. *Divisive:* Start with one, all-inclusive cluster and, at each step, split a cluster until only singleton clusters of individual points remain. In this case, we need to decide which cluster to split at each step.

Divisive techniques are less common, and we will mention only one example before focusing on agglomerative techniques.

Simple Divisive Algorithm

1. Compute a minimum spanning tree for the proximity graph.
2. Create a new cluster by breaking the link corresponding to the largest distance (smallest similarity).
3. Repeat step 2 until only singleton clusters remain.

This approach is the divisive version of the "single link" agglomerative technique that we will see shortly.

Basic Agglomerative Hierarchical Clustering Algorithm

Many hierarchical agglomerative techniques can be expressed by the following algorithm, which is known as the Lance-Williams algorithm.

Basic Agglomerative Hierarchical Clustering Algorithm

1. Compute the proximity graph, if necessary. (Sometimes the proximity graph is all that is available.)
2. Merge the closest (most similar) two clusters.
3. Update the proximity matrix to reflect the proximity between the new cluster and the original clusters.
4. Repeat steps 3 and 4 until only a single cluster remains.

The key step of the previous algorithm is the calculation of the proximity between two clusters, and this is where the various agglomerative hierarchical techniques differ. Any of the cluster proximities that we discuss in this section can be viewed as a choice of different parameters (in the Lance-Williams formula) for the proximity between clusters

Q and R, where R is formed by merging clusters A and B.

$p(R, Q) = \langle A\, p(A, Q) + \langle B\, p(B, Q) + ® p(A, Q) + © | p(A, Q) - p(B, Q) |$

In words, this formula says that after you merge clusters A and B to form cluster R, then the distance of the new cluster, R, to an existing cluster, Q, is a linear function of the distances of Q from the original clusters A and B.

Time and Space Complexity

Hierarchical clustering techniques typically use a proximity matrix. This requires the computation and storage of *m*2 proximities, a factor that limits the size of data sets that can be processed. (It is possible to compute the proximities on the fly and save space, but this increases computation time.) Once the proximity matrix is available, the time required for hierarchical clustering is *O(m2)*.

7.6 NEAREST NEIGHBOUR TECHNIQUE

The Nearest Neighbour Technique is similar to Clustering

The nearest neighbour algorithm is basically a refinement of clustering in the sense that they both use distance in some feature space to create either structure in the data or predictions. The nearest neighbor algorithm is a refinement since part of the algorithm usually is a way of automatically determining the weighting of the importance of the predictors and how the distance will be measured within the feature space. Clustering is one special case of this where the importance of each predictor is considered to be equivalent.

Difference between Clustering and Nearest Neighbour Prediction

There are two main types of clustering techniques, those that create a hierarchy of clusters and those that do not. The hierarchical clustering techniques create a hierarchy of clusters from small to big. The main reason for this is that, as was already stated, clustering is an unsupervised learning technique, and as such, there is no absolutely correct answer. For this reason and depending on the particular application of the clustering, fewer or greater numbers of clusters may be desired. With a hierarchy of clusters defined it is possible to choose the number of clusters that are desired.

Nearest Neighbour	Clustering
Used for prediction as well as consolidation.	Used mostly for consolidating data into a high-level view and general grouping of records into like behaviors.
Space is defined by the problem to be solved (supervised learning).	Space is defined as default n-dimensional space, or is defined by the user, or is a predefined space driven by past experience (unsupervised learning).
Generally only uses distance metrics to determine nearness.	Can use other metrics besides distance to determine nearness of two records-for example linking two points together.

7.7 THE NEXT GENERATION TECHNIQUES

The data mining techniques in this section represent the most often used techniques that have been developed over the last two decades of research. They also represent the vast majority of the techniques that are being spoken about when data mining is mentioned in the popular press. These techniques can be used for either discovering new information within large databases or for building predictive models. Though the older decision tree techniques such as CHAID are currently highly used the new techniques such as CART are gaining wider acceptance.

7.8 DECISION TREES

A decision tree is a predictive model that, as its name implies, can be viewed as a tree. Specifically each branch of the tree is a classification question and the leaves of the tree are partitions of the dataset with their classification. For instance if we were going to classify customers who churn (don't renew their phone contracts) in the Cellular Telephone Industry a decision tree might look something like that found in Figure 7.20.

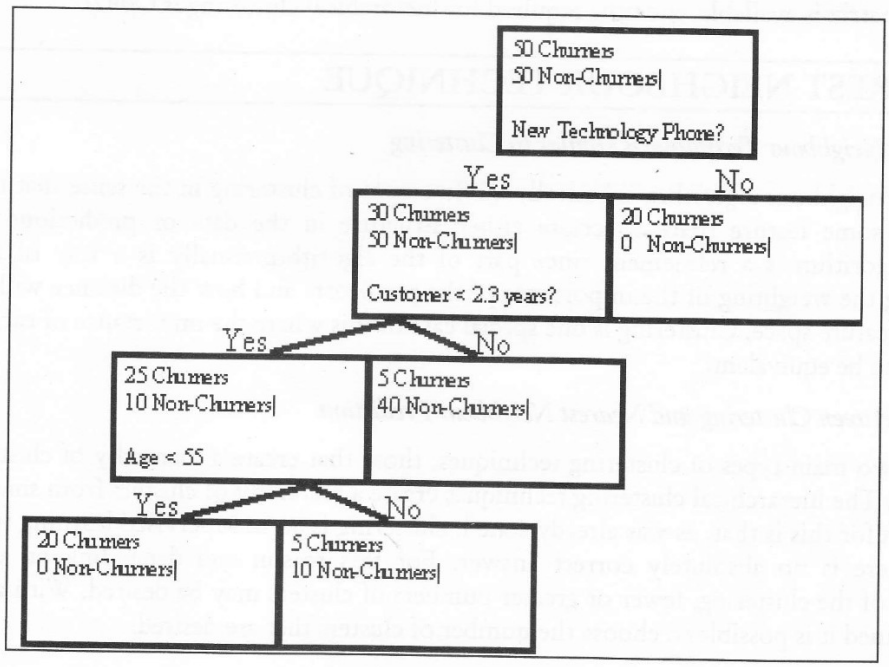


Figure 7.20

You may notice some interesting things about the tree:

- It divides up the data on each branch point without losing any of the data (the number of total records in a given parent node is equal to the sum of the records contained in its two children).
- The number of churners and non-churners is conserved as you move up or down the tree
- It is pretty easy to understand how the model is being built (in contrast to the models from neural networks or from standard statistics).
- It would also be pretty easy to use this model if you actually had to target those customers that are likely to churn with a targeted marketing offer.

Decision trees are linear models that use axis parallel "splits" to recursively partition the labeled points, so that each region is as "pure" as possible in terms of the labels. A decision tree consists of internal nodes that represent decisions or splits, and leaf nodes that are labeled with a class.

For example,

Id	Age	Car	Risk
1	25	Sports	L
2	20	Vintage	H
3	25	Sports	L
4	45	SUV	H
5	20	Sports	H
6	25	SUV	H

In this example, Age is numeric and Car is categorical. Risk gives the class label for each point: high (H) or low (L).

Depending on the type of attribute, the decisions at the internal nodes are of different types. For numeric data, we consider decisions of the kind $A < v$, where A is the attribute and v is some value in the domain of A. For categorical data, we use a decision of the kind $A \in V$, where V is some subset of the values of the attribute. For example, consider the example dataset shown above. For Age, we might have a decision of the kind $Age < 25$, whereas for Car, we may have a decision of the kind $Car \in \{sports, SUV\}$.

Let us consider the following example of a recognition problem. During a doctor's examination of some patients the following characteristics are determined:

Example: X_1 - temperature, X_2 - coughing, X_3 - a reddening throat, $Y = \{W_1, W_2, W_3, W_4, W_5\} = \{$a cold, quinsy, the influenza, a pneumonia, is healthy$\}$ - a set from the possible diagnoses, demanding more profound inspection.

It is required to find a model, where Y depends on X. The example illustrates such a model, which can be seen as a decision tree.

The ordinary tree consists of one root, branches, nodes (places where branches are divided) and leaves. In the same way the decision tree consists of nodes which stand for circles, the branches stand for segments connecting the nodes. A decision tree is usually drawn from left to right or beginning from the root downwards, so it is easier to draw it. The first node is a root. The end of the chain "root - branch – node - ... - node" is called "leaf". From each internal node (i.e. not a leaf) may grow out two

or more branches. Each node corresponds with a certain characteristic and the branches correspond with a range of values. These ranges of values must give a partition of the set of values of the given characteristic.

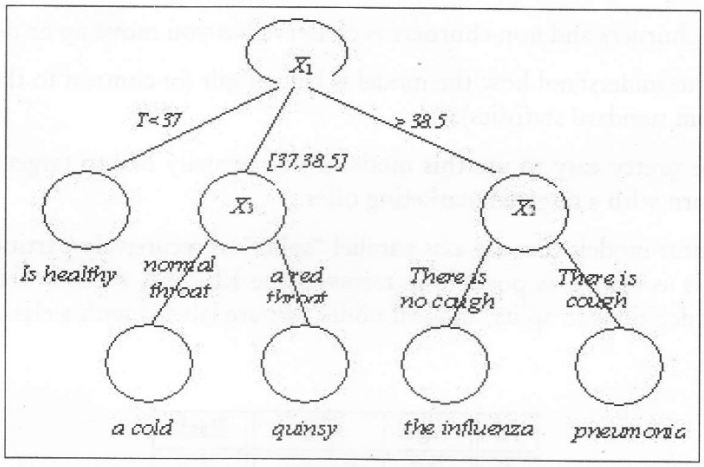


Figure 7.21

When precisely two branches grow out from an internal node (the tree of such type is called a dichotomic tree), each of these branches can give a true of false statement concerning the given characteristic as is shown on Figure 7.22

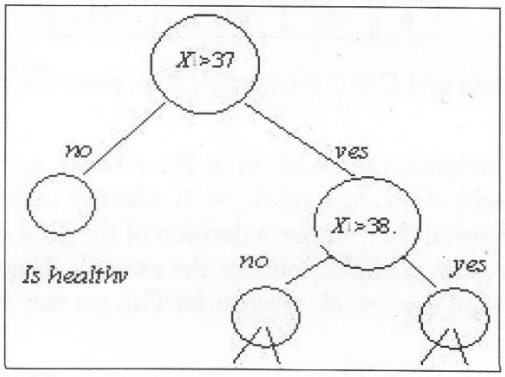


Figure 7.22

The value Y is ascribed for each terminal node of a tree (named "leaf"). In case of pattern recognition problem the given value is a certain class, and in a regression analysis case the given value represents a real number.

For any observation of _, using a decision tree, we can find the predicted value Y. For this purpose we start with a root of a tree, we consider the characteristic, corresponding to a root and we define, to which branch the observed value of the given characteristic corresponds. Then we consider the node in which the given branch comes. We repeat the same operations for this node etc., until we reach a leaf. The value YS ascribed to S-th leaf will be the forecast for x. Thus, the decision tree gives the model T of dependence Y from X: Y = T(X).

7.8.1 Decision Tree Algorithm

The decision tree algorithm, given in Algorithm, is a recursive method that takes a given subset of the data D and evaluates all possible splits (or decisions), considering both numeric and categorical attributes. The best decision/split is chosen to partition the data into two subsets, on which the method is called recursively. The method stops when certain stopping conditions are met.

Decision Tree (D)

```
1.      if (stop condition met) then
2.          choose majority class in D as the leaf label
3.      for each (attribute A in D) do
4.          if (A is numeric) then
5.              Try all possible splits: A ≤ v
6.          if (A is categorical) then
7.              Try all possible splits: A ∈ V
8.      Choose the best split based on highest information gain
9.      Partition D into DL and DR
10.     Decision Tree (DL)
11.     Decision Tree (DR)
```

Stopping Criteria

A number of stopping conditions can be used to stop the recursive process:

1. If all the points in D have the same label, then stop, since in this case the sample D is already "pure" in terms of labels.
2. Stop if most of the points are already of the same class. This is a generalization of the first approach, with some error threshold (for better generalizability). For example, we may stop if at least a given fraction of the points in D share the same label.
3. If $|D| \_ s$, where s is some minimum leaf size threshold, then stop. This condition prevents over-fitting the model to the training set, since we avoid trying to model very small subsets.

Selecting Splits

We now need objective criteria for judging how good a split is. Intuitively, we want to select a split that gives the best separation or discrimination between the different labels.

One measure of the purity of a sample is measured using the concept of Entropy. Entropy, in general, measures the amount of disorder or uncertainty. In the classification setting, higher entropy (i.e., more disorder) corresponds to a sample that has a mixed collection of labels. Lower entropy corresponds to a case where we have mostly pure partitions. In information theory, the entropy of a sample D is defined as follows:

$$H(D) = -\sum_{i=1}^{k} P(c_i|D)\ \log P(c_i|D)$$

where $P(c_i|D)$ is the probability of a data point in D being labeled with class c_i, and k is the number of classes. $P(c_i|D)$ can be estimated directly from the data as follows:

$$P(c_i|D) = \frac{\left|\{x_j \in D | x_j \text{ has label } y_j = c_i\}\right|}{|D|}$$

How to Build Decision Tree

The procedure of the formation of a decision tree by statistical data is also called construction of a tree. In this paragraph we will get acquainted to some ways of construction of trees and also ways of definition of decision tree quality.

For each specific target of the statistical analysis, there is a large number (frequently even indefinitely) of different variants of decision trees. There is a question: which tree is the best and how to find it? To answer on the first question, we will consider various ways of definition of the parameters describing the quality of a tree. Theoretically, we can consider the expected error of forecasting as the basic parameter. However, this value can be defined only if we know the probabilistic distributive law of the examined variables. In practice however, this law, as a rule, is unknown. Therefore we can estimate quality only approximately, using the set of observations given to us.

An Estimation of Quality on a Control Sample

Control (test) sample is called sample, which is not used for building a tree, but used for an estimation of quality of the constructed tree. Two parameters are calculated: for the recognition problem the relative number of mistakes and for the regression analysis problem the variance on control sample. Since this sample does not participate in the tree construction, these parameters reflect the «true» unknown error more objectively. The bigger the control sample size, the higher the degree of approximation.

For a recognition problem, under the condition of independency of the observations, the frequency of mistakes belongs to binomial distribution. Therefore, knowing the number of errors on the control sample, it is possible to find a confidence interval to which, with the given probability, the unknown value of misclassification error belongs. In work are given diagrams in which it is possible to define a confidence interval for the given control sample size and number of errors in the control.

7.8.2 Methods of Construction of Decision Tree

Existing methods (there are dozens of methods) can be divided into two basic groups. The first group is methods of building of a strict-optimum tree (by the given criteria of quality of a tree) and the second group is methods of construction of an approximately optimum tree.

The problem of searching the optimum variant of a tree can be related to a discrete programming problem or a choice from finite (but very large) numbers of variants. It follows from the fact that for finite training sample the number of variants of branching (see below) for each characteristic is finite.

Three basic kinds of methods in discrete programming are considered: exhausting search, a method of dynamic programming and a branch and bounds method. However, these methods, in the application to decision trees, as a rule, are very laborious, especially for the large number of observations and

characteristics. Therefore, we will consider approximate methods: method of consecutive branching, a method of pruning and a recursive method.

Operations of Branching (division)

This operation is the basic operation for tree constructions. We will consider a node of a tree and some characteristic X_j. Let the range of definition of this characteristic be divided on L_j subsets (ways of a choice of such subsets we will consider below). In case of the quantitative characteristic, these subsets represent a set subintervals of splitting, in case of the qualitative characteristic a subsets of values and in case of the ordered characteristic the subsets including the neighbouring values.

Let us associate with each of these subsets a branch of the tree leaving the given (parent) node and going into a new node which is called descendant. Thus, the node "has branched" ("has divided") on L_j new nodes (Figure 7.23)

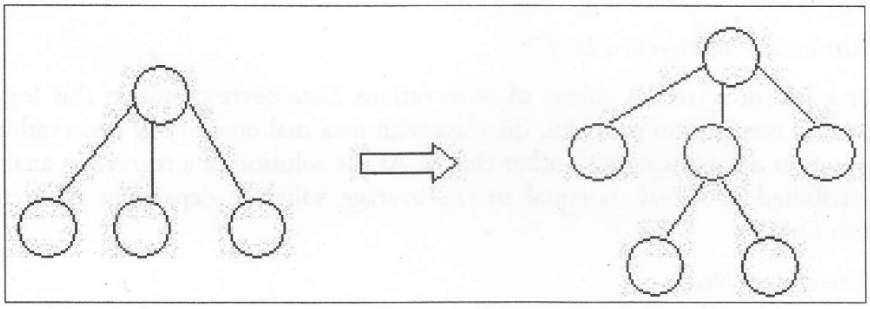


Figure 7.23

Notice that for binary trees L_j is always equal to two. If L_j is always equal to three such trees are called *ternary*. If L_j is always equal to four we receive quadratic-trees.

How to obtain the splitting of a range of definition? We take a set of the observations corresponding to the given node and consider values of characteristic X_j for these observations.

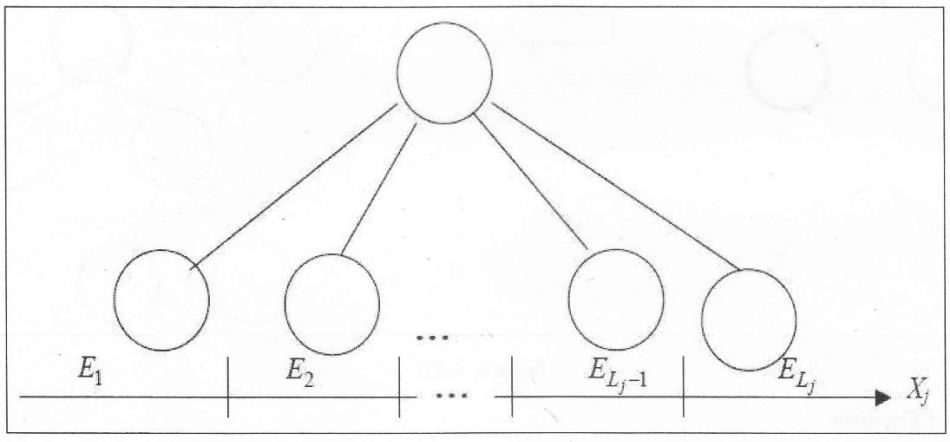


Figure 7.24

Consider a quantitative characteristic. In this case, boundaries are in the middle of intervals between the neighbor values and splitting carried out on these boundaries (Figure 7.24).

Operation of the Definition of Degree of Promise for Branching Node (Rule of a Stopping)

Let us consider a dangling node of a tree, i.e. the node which is not branched, but it is not clear, whether this node will be a leaf or whether we need further branching. We will consider the subset of observations appropriate to the given node. We will divide the nodes into two cases. First, if these observations are homogeneous, i.e. basically belong to the same class (a pattern recognition problem, RP), or if the variance of Y for them is small enough (a regression analysis problem, RA). The variant when the values of characteristic are equal for all observations corresponds also to this case. Second, if the number of observations is not enough.

The node, unpromising for the further branching, is called leaf.

For definition of degree of promise, it is possible to set the following parameters: an allowable error for node (PR problem), an allowable variance (RA problem) and a threshold on the quantity of observations.

Operation "to Attribute a Solution to a Leaf"

Let us consider a leaf of a tree. A subset of observations *Data* corresponds to this leaf. During the solution of a pattern recognition problem, the class with maximal quantity of observations from *Data* is assigned to a leaf, in comparison with other classes. At the solution of a regression analysis problem, the solution attributed to a leaf, is equal to the average value of dependent characteristic Y for observation from *Data*.

Operation of "Growth" of Node

This operation represents a sequence of operations of branching for each of the new nodes of a tree. As a result of this operation, the given node is replaced with some sub tree (i.e. a part of a full tree which also looks like a tree (Figure 7.25)

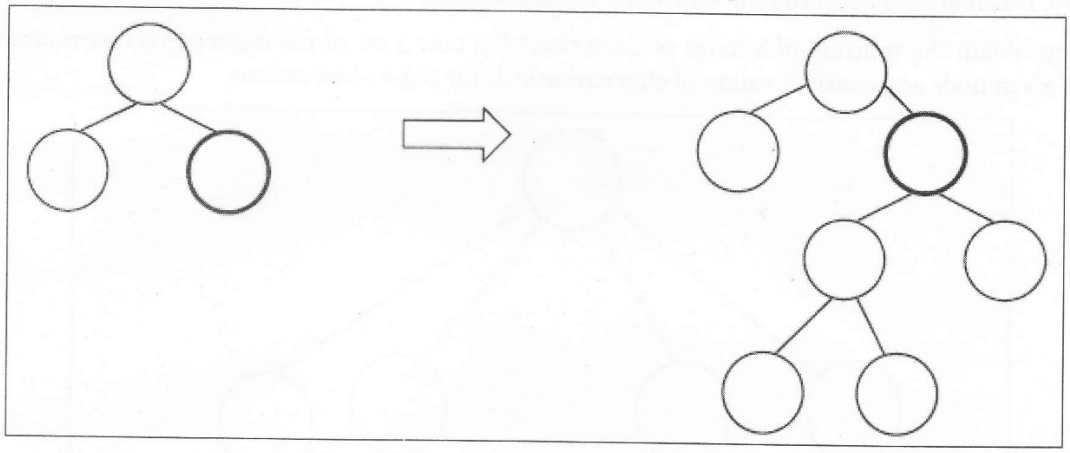


Figure 7.25

Operation of Pruning

This operation is the opposite of operations of growth, i.e. for the given node appropriate sub tree for which this node is a root completely cuts (Figure 7.26) the node is then called a leaf.

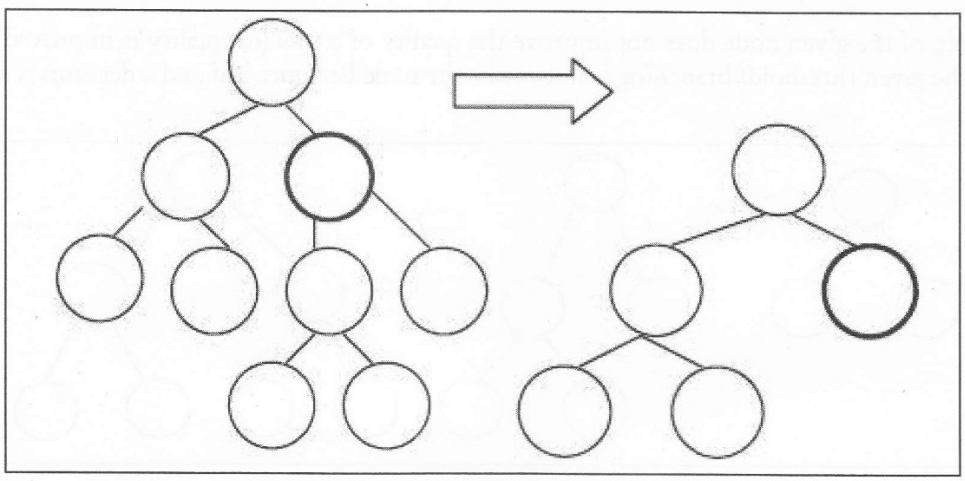


Figure 7.26

Operation of "Aggregation" of Nodes or ("Join")

Let the node be divided on L new nodes. We will take any pair of these nodes and we will unite them in one node and connect it with the parental node. Thus subsets of the values to the appropriate aggregation nodes are united.

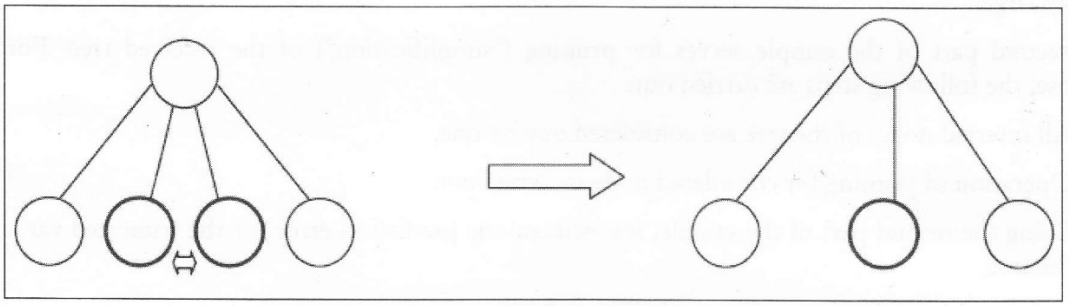


Figure 7.27

The aggregation of nodes, which correspond to the neighbor subintervals or subsets of values of quantitative and ordered characteristics is allowed.

Method of Sequential Branching

The given method represents a procedure of step-by-step branching at which on each step the best variant of division gets out. Let the possible allowable complexity of a tree (for example, number of leaves *Mmax*) be given. The method consists of the following steps (Figure 7.28)

To divide the root into the given number of new nodes, using all variants of division by each characteristic X_i from $X_1,..., X_n$ by turns. The best variant of division by the given criterion of quality is saved.

To check up the degree of promise of branching for each of new child nodes. If a node becomes a leaf, we give to it the appropriate decision.

Each node (not leaf) is divided into new nodes similarly to item 1.

If branching of the given node does not improve the quality of a tree (or quality is improved, but it is less than the given threshold) branching is not made; the node becomes leaf and a decision is attributed to it.

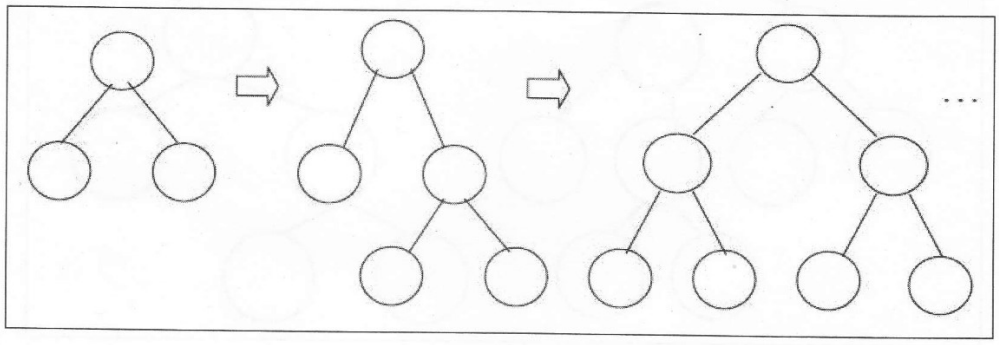


Figure 7.28

Method of Pruning

In a pruning method, the training sample is divided in two parts. The first part is used for tree construction by a method of consecutive branching. Parameters of a stop are set in such way to provide the maximal possible accuracy of the received decision. The number of leaves of the tree can be very large.

The second part of the sample serves for pruning ("simplification") of the received tree. For this purpose, the following steps are carried out.

- All internal nodes of the tree are considered one by one.
- Operation of pruning for considered node is carried out.
- Using the second part of the sample, we estimate the prediction error for the truncated variant of the tree.

The variant with minimal error is a result (Figure 7.29)

The described method gives more objective estimation of quality, however, if the initial tree is far from optimum, then the truncated variant will not be ideal also.

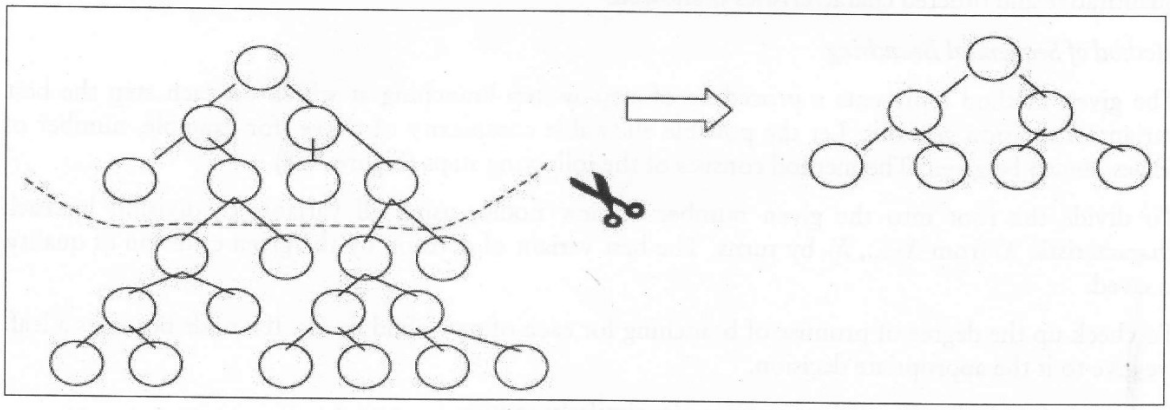


Figure 7.29

From Decision Tree to Decision Forest

A decision forest is a set of several decision trees. These trees can be formed by various methods (or by one method, but with various parameters of work), by different sub-samples of observations over one and the same phenomenon, by use of different characteristics. Such many sided consideration of a problem, as a rule, gives the improvement of quality of forecasting and a better understanding of laws of the researched phenomenon.

Let us consider a set of trees and an observation x. Each tree gives a forecast for _. How to find the general (collective) decision for the predicted variable Y?

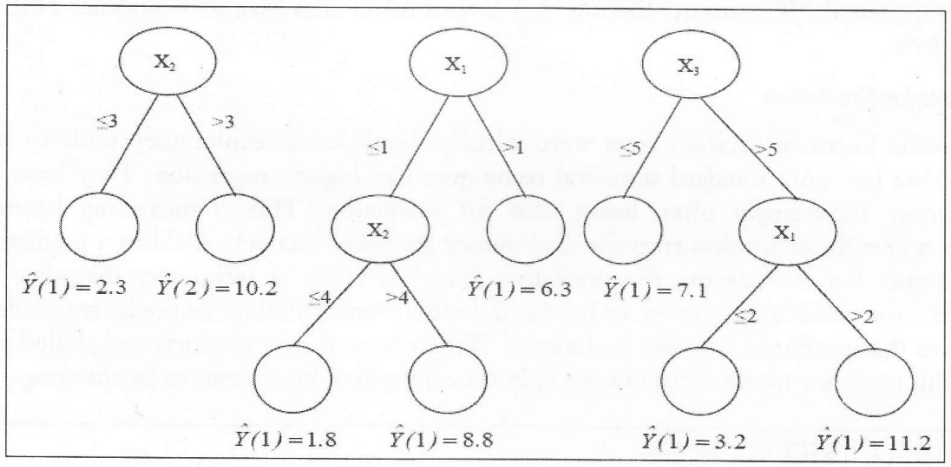


Figure 7.30

The simplest way to obtain the collective forecast, which gives the given decision forest, is voting method (PR problem) or method of averaging (RA problem).

Using a voting method, a class attributed to observation x is a class which the majority of trees prefer. In the regression analysis problem, the predicted value is a mean of forecasts of all trees. We will consider, for example, a set of regression trees, which are shown at Figure 7.30. Consider observation x = (3, 4, 8). The collective decision will be equal to $Y(x)$ = (10.2 + 6.3 + 11.2)/3 = 9.233.

Next to simple averaging or voting with equal contributions of each vote, it is possible to use a procedure in which the number of mistakes accomplished by each tree on training sample is taken into account. The fewer mistakes, the greater weight the appropriate voice has.

7.8.3 Applying Decision Trees to Business

Because of their tree structure and ability to easily generate rules decision trees are the favored technique for building understandable models. Because of this clarity they also allow for more complex profit and ROI models to be added easily in on top of the predictive model. For instance once a customer population is found with high predicted likelihood to attrite a variety of cost models can be used to see if an expensive marketing intervention should be used because the customers are highly valuable or a less expensive intervention should be used because the revenue from this sub-population of customers is marginal.

Because of their high level of automation and the ease of translating decision tree models into SQL for deployment in relational databases the technology has also proven to be easy to integrate with existing

IT processes, requiring little preprocessing and cleansing of the data, or extraction of a special purpose file specifically for data mining.

Using Decision Trees for Exploration

The decision tree technology can be used for exploration of the dataset and business problem. This is often done by looking at the predictors and values that are chosen for each split of the tree. Often times these predictors provide usable insights or propose questions that need to be answered. For instance if you ran across the following in your database for cellular phone churn you might seriously wonder about the way your telesales operators were making their calls and maybe change the way that they are compensated: "IF customer lifetime < 1.1 years AND sales channel = telesales THEN chance of churn is 65%.

Decision Tree for Prediction

Although some forms of decision trees were initially developed as exploratory tools to refine and preprocess data for more standard statistical techniques like logistic regression. They have also been used and more increasingly often being used for prediction. This is interesting because many statisticians will still use decision trees for exploratory analysis effectively building a predictive model as a by product but then ignore the predictive model in favor of techniques that they are most comfortable with. Sometimes veteran analysts will do this even excluding the predictive model when it is superior to that produced by other techniques. With a host of new products and skilled users now appearing this tendency to use decision trees only for exploration now seems to be changing.

7.9 NEURAL NETWORKS

When data mining algorithms are talked about these days most of the time people are talking about either decision trees or neural networks. Of the two neural networks have probably been of greater interest through the formative stages of data mining technology. As we will see neural networks do have disadvantages that can be limiting in their ease of use and ease of deployment, but they do also have some significant advantages. Foremost among these advantages is their highly accurate predictive models that can be applied across a large number of different types of problems.

Traditionally, the term neural network had been used to refer to a network or circuit of biological neurons the modern usage of the term often refers to artificial neural networks, which are composed of artificial neurons or nodes.

In general a biological neural network is composed of a group or groups of chemically connected or functionally associated neurons. A single neuron may be connected to many other neurons and the total number of neurons and connections in a network may be extensive. Connections, called synapses, are usually formed from axons to dendrites, though dendrodendritic microcircuits and other connections are possible. Apart from the electrical signaling, there are other forms of signaling that arise from neurotransmitter diffusion, which have an effect on electrical signaling. As such, neural networks are extremely complex.

Artificial intelligence and cognitive modeling try to simulate some properties of neural networks. While similar in their techniques, the former has the aim of solving particular tasks, while the latter aims to build mathematical models of biological neural systems.

7.9.1 Are Neural Networks easy to use?

A common claim for neural networks is that they are automated to a degree where the user does not need to know that much about how they work, or predictive modeling or even the database in order to use them. The implicit claim is also that most neural networks can be unleashed on your data straight out of the box without having to rearrange or modify the data very much to begin with. Just the opposite is often true. There are many important design decisions that need to be made in order to effectively use a neural network such as:

- How should the nodes in the network be connected?
- How many neuron like processing units should be used?
- When should "training" be stopped in order to avoid over fitting?

There are also many important steps required for preprocessing the data that goes into a neural network - most often there is a requirement to normalize numeric data between 0.0 and 1.0 and categorical predictors may need to be broken up into virtual predictors that are 0 or 1 for each value of the original categorical predictor. And, as always, understanding what the data in your database means and a clear definition of the business problem to be solved are essential to ensuring eventual success. The bottom line is that neural networks provide no short cuts.

7.9.2 Applying Neural Networks to Business

Neural networks are very powerful predictive modeling techniques but some of the power comes at the expense of ease of use and ease of deployment. As we will see in this section, neural networks, create very complex models that are almost always impossible to fully understand even by experts. The model itself is represented by numeric values in a complex calculation that requires all of the predictor values to be in the form of a number. The output of the neural network is also numeric and needs to be translated if the actual prediction value is categorical (e.g. predicting the demand for blue, white or black jeans for a clothing manufacturer requires that the predictor values blue, black and white for the predictor color to be converted to numbers).

Because of the complexity of these techniques much effort has been expended in trying to increase the clarity with which the model can be understood by the end user. These efforts are still in there infancy but are of tremendous importance since most data mining techniques including neural networks are being deployed against real business problems where significant investments are made based on the predictions from the models (e.g. consider trusting the predictive model from a neural network that dictates which one million customers will receive a $1 mailing). There are two ways that these shortcomings in understanding the meaning of the neural network model have been successfully addressed:

1. The neural network is package up into a complete solution such as fraud prediction. This allows the neural network to be carefully crafted for one particular application and once it has been proven successful it can be used over and over again without requiring a deep understanding of how it works.
2. The neural network is package up with expert consulting services. Here the neural network is deployed by trusted experts who have a track record of success. Either the experts are able to explain the models or they are trusted that the models do work.

7.9.3 Neural Networks for Feature Extraction

One of the important problems in all of data mining is that of determining which predictors are the most relevant and the most important in building models that are most accurate at prediction. These predictors may be used by themselves or they may be used in conjunction with other predictors to form "features". A simple example of a feature in problems that neural networks are working on is the feature of a vertical line in a computer image. The predictors, or raw input data are just the colored pixels that make up the picture. Recognizing that the predictors (pixels) can be organized in such a way as to create lines, and then using the line as the input predictor can prove to dramatically improve the accuracy of the model and decrease the time to create it.

The neural network shown in Figure 7.31. is used to extract features by requiring the network to learn to recreate the input data at the output nodes by using just 5 hidden nodes. Consider that if you were allowed 100 hidden nodes, that recreating the data for the network would be rather trivial - simply pass the input node value directly through the corresponding hidden node and on to the output node. But as there are fewer and fewer hidden nodes, that information has to be passed through the hidden layer in a more and more efficient manner since there are less hidden nodes to help pass along the information.

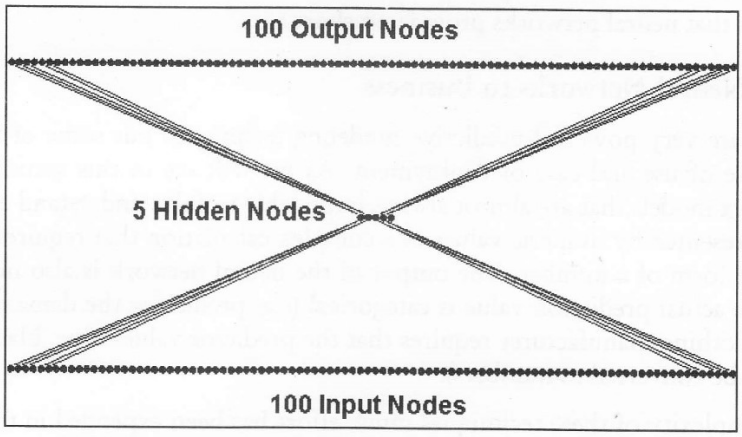


Figure 7.31

7.9.4 What does a Neural Net look like?

A neural network is loosely based on how some people believe that the human brain is organized and how it learns. Given that there are two main structures of consequence in the neural network:

- The node-which loosely corresponds to the neuron in the human brain.
- The link-which loosely corresponds to the connections between neurons (axons, dendrites and synapses) in the human brain.

In Figure 7.32 there is a drawing of a simple neural network. The round circles represent the nodes and the connecting lines represent the links. The neural network functions by accepting predictor values at the left and performing calculations on those values to produce new values in the node at the far right. The value at this node represents the prediction from the neural network model. In this case the network takes in values for predictors for age and income and predicts whether the person will default on a bank loan.

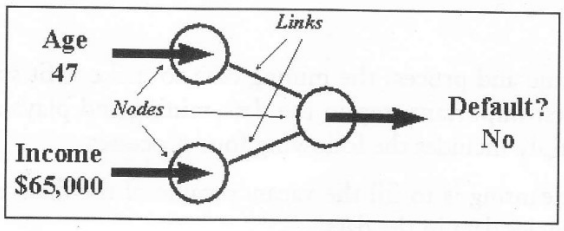


Figure 7.32

7.9.5 How is the Neural Network Model Created?

The neural network model is created by presenting it with many examples of the predictor values from records in the training set (in above example age and income are used) and the prediction value from those same records. By comparing the correct answer obtained from the training record and the predicted answer from the neural network it is possible to slowly change the behavior of the neural network by changing the values of the link weights. In some ways this is like having a grade school teacher asks questions of her student (a.k.a. the neural network) and if the answer is wrong to verbally correct the student. The greater the error the harsher the verbal correction. So that large errors are given greater attention at correction than are small errors.

For the actual neural network it is the weights of the links that actually control the prediction value for a given record. Thus the particular model that is being found by the neural network is in fact fully determined by the weights and the architectural structure of the network. For this reason it is the link weights that are modified each time an error is made.

7.9.6 Data Mining Process Based on Neural Network

Data mining process can be composed by three main phases: data preparation, data mining, expression and interpretation of the results, data mining process is the reiteration of the three phases. The details are shown in Figure 7.33

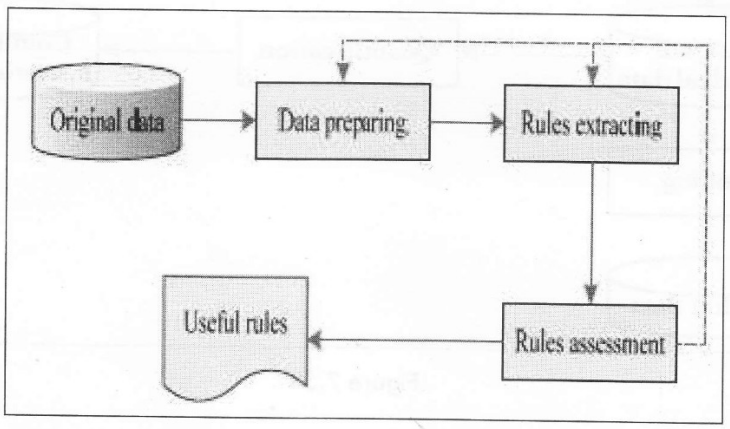


Figure 7.33

The data mining based on neural network is composed by data preparation, rules extracting and rules assessment three phases.

Data Preparation

Data preparation is to define and process the mining data to make it fit specific data mining method. Data preparation is the first important step in the data mining and plays a decisive role in the entire data mining process. It mainly includes the following four processes.

- *Data cleaning:* Data cleansing is to fill the vacancy value of the data, eliminate the noise data and correct the inconsistencies data in the data.
- *Data option:* Data option is to select the data arrange and row used in this mining.
- *Data preprocessing:* Data preprocessing is to enhanced process the clean data which has been selected.
- *Data expression:* Data expression is to transform the data after preprocessing into the form which can be accepted by the data mining algorithm based on neural network. The data mining based on neural network can only handle numerical data, so it is need to transform the sign data into numerical data. The simplest method is to establish a table with one-to-one correspondence between the sign data and the numerical data. The other more complex approach is to adopt appropriate Hash function to generate a unique numerical data according to given string. Although there are many data types in relational database, but they all basically can be simply come down to sign data, discrete numerical data and serial numerical data three logical data types. Figure 7.34 gives the conversion of the three data types.

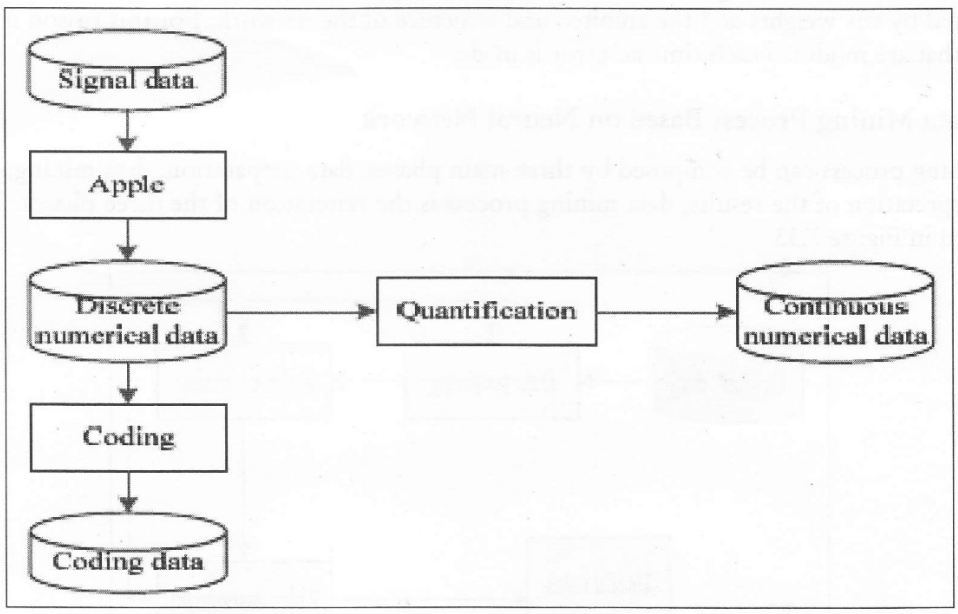


Figure 7.34

Rules Extracting

There are many methods to extract rules, in which the most commonly used methods are LRE method, black-box method, the method of extracting fuzzy rules, the method of extracting rules from recursive network, the algorithm of binary input and output rules extracting (BIO-RE), partial rules extracting algorithm (Partial-RE) and full rules extracting algorithm (Full-RE).

Rules Assessment

Although the objective of rules assessment depends on each specific application, but, in general terms, the rules can be assessed in accordance with the following objectives.

Find the optimal sequence of extracting rules, making it obtains the best results in the given data set;

- Test the accuracy of the rules extracted;
- Detect how much knowledge in the neural network has not been extracted;
- Detect the inconsistency between the extracted rules and the trained neural network.

7.9.7 Neural Network Models

There are two main types of neural network models: supervised neural networks such as the multi-layer perceptron or radial basis functions, and unsupervised neural networks such as Kohonen feature maps. A supervised neural network uses training and testing data to build a model. The data involves historical data sets containing input variables, or data fields, which correspond to an output. The training data is what the neural network uses to "learn" how to predict the known output, and the testing data is used for validation. The aim is for the neural networks to predict the output for any record given the input variables only.

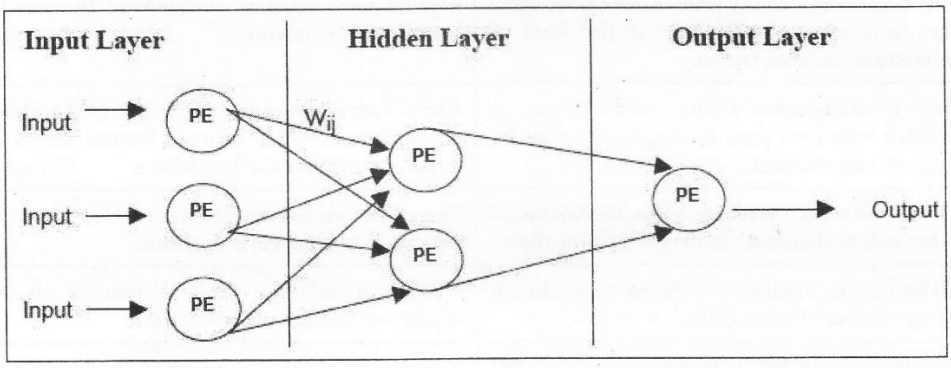


Figure 7.35

One of the simplest Feed Forward Neural Networks (FFNN), consists of three layers: an input layer, hidden layer and output layer. In each layer there are one or more Processing Elements (PEs). PEs are meant to simulate the neurons in the brain and this is why they are often referred to as neurons or nodes. A PE receives inputs from either the outside world or the previous layer. There are connections between the PEs in each layer that have a weight (parameter) associated with them. This weight is adjusted during training. Information only travels in the forward direction through the network – there are no feedback loops.

The simplified process for training a FFNN is as follows:

- Input data is presented to the network and propagated through the network until it reaches the output layer. This forward process produces a predicted output.
- The predicted output is subtracted from the actual output and an error value for the networks is calculated.

- The neural network then uses supervised learning, which in most cases is backpropagation, to train the network. Backpropagation is a learning algorithm for adjusting the weights. It starts with the weights between the output layer PE's and the last hidden layer PE's and works backwards through the network.
- Once backpropagation has finished, the forward process starts again, and this cycle is continued until the error between predicted and actual outputs is minimised.

Neural networks are becoming very popular with data mining practitioners, particularly in medical research, finance and marketing. This is because they have proven their predictive power through comparison with other statistical techniques using real data sets.

Advantages of Neural Networks	Disadvantages of Neural Networks
High Accuracy: Neural networks are able to approximate complex non-linear mappings.	*Poor Transparency:* Neural networks operate as "black boxes".
Noise Tolerance: Neural networks are very flexible with respect to incomplete, missing and noisy data.	*Trial-and-error design:* The selection of the hidden nodes and training parameters is heuristic.
Independence from prior assumptions: Neural networks do not make a priori assumptions about the distribution of the data, or the form of interactions between factors.	*Data hungry:* Estimating the network weights requires large amounts of data, and this can be very Computer intensive.
Ease of maintenance: Neural networks can be updated with fresh data, making them useful for dynamic environments.	*Over-fitting:* If too many weights are used without regularisation, Neural network become useless in terms of generalisation to new data.
Neural network overcome some limitations of other statistical methods while generalizing them.	There is no explicit set of rules to select the most suitable Neural network algorithm.
Hidden nodes, in supervised Neural networks can be regarded as latent variables.	Neural networks are totally dependent on the quality and amount of data available.
Neural networks can be implemented in parallel hardware.	Neural networks may converge to local minima in the error surface.
Neural networks performance can be highly automated, minimizing human involvement.	Neural networks lack classical statistical properties. Confidence intervals and hypothesis testing are not available.
Neural networks are especially suited to tackling problems in non-conservative domains.	Neural network techniques are still rapidly evolving and they are not yet robust.

7.9.8 Architecture of Neural Networks

Feed Forward Networks

Feed-forward ANNs (Figure 7.36) allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down.

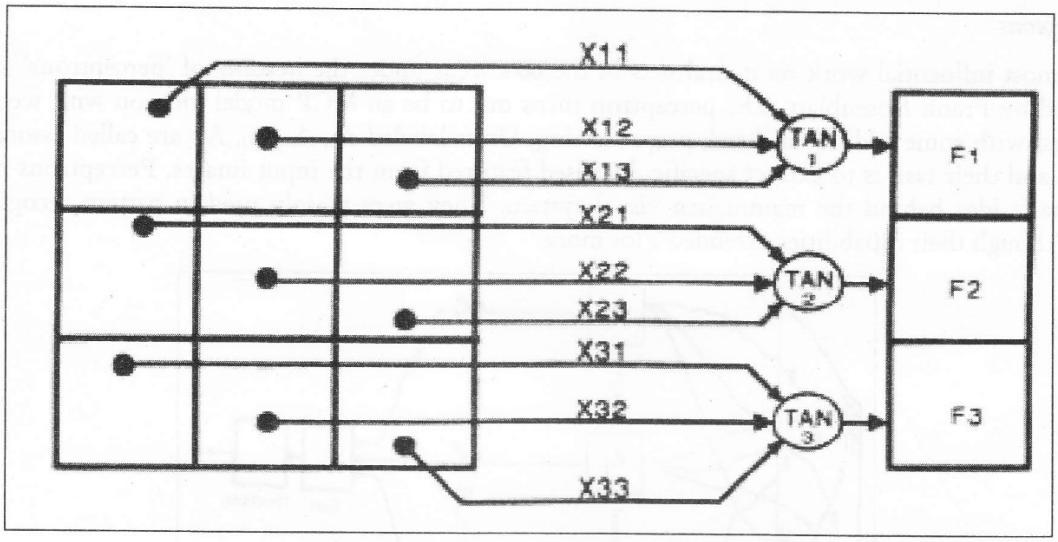


Figure 7.36

Feed Back Networks

Feedback networks can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations.

Network Layer

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "**input**" units is connected to a layer of "**hidden**" units, which is connected to a layer of "**output**" units.

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.
- The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organisation, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organisations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering.

Perceptons

The most influential work on neural nets in the 60's went under the heading of 'perceptrons' a term coined by Frank Rosenblatt. The perceptron turns out to be an MCP model (neuron with weighted inputs) with some additional, fixed, preprocessing. Units labeled A_1, A_2, A_j, Ap are called association units and their task is to extract specific, localised featured from the input images. Perceptrons mimic the basic idea behind the mammalian visual system. They were mainly used in pattern recognition even though their capabilities extended a lot more.

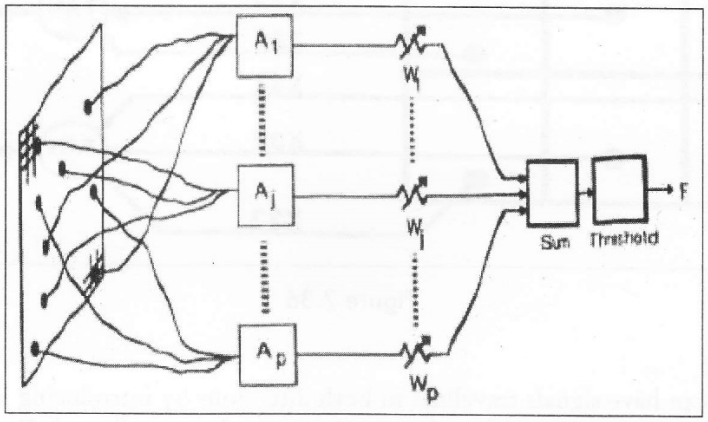


Figure 7.37

In 1969 Minsky and Papert wrote a book in which they described the limitations of single layer Perceptrons. The impact that the book had was tremendous and caused a lot of neural network researchers to loose their interest. The book was very well written and showed mathematically that *single layer* perceptrons could not do some basic pattern recognition operations like determining the parity of a shape or determining whether a shape is connected or not. What they did not realised, until the 80's, is that given the appropriate training, multilevel perceptrons can do these operations.

7.9.9 Neural Networks Category

Fixed Neural Networks

In which the weights cannot be changed, i.e. dW/dt=0. In such networks, the weights are fixed a priori according to the problem to solve.

Adaptive Networks

Which are able to change their weights, i.e. dW/dt not= 0.

All learning methods used for adaptive neural networks can be classified into two major categories:

1. ***Supervised Learning:*** Which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning. An important issue concerning supervised learning is the problem of error convergence, i.e. the minimization of error between the desired and computed unit values. The aim is to determine a set of weights, which minimizes the error. One

well-known method, which is common to many learning paradigms, is the Least Mean Square (LMS) convergence.

2. ***Unsupervised Learning:*** The behaviour of an ANN (Artificial Neural Network) depends on both the weights and the input-output function (transfer function) that is specified for the units. This function typically falls into one of three categories:
   - ❖ Linear (or ramp)
   - ❖ Threshold
   - ❖ Sigmoid

For ***linear units***, the output activity is proportional to the total weighted output.

For ***threshold units***, the output is set at one of two levels, depending on whether the total input is greater than or less than some threshold value.

For ***sigmoid units***, the output varies continuously but not linearly as the input changes. Sigmoid units bear a greater resemblance to real neurones than do linear or threshold units, but all three must be considered rough approximations.

To make a neural network that performs some specific task, we must choose how the units are connected to one another and we must set the weights on the connections appropriately. The connections determine whether it is possible for one unit to influence another. The weights specify the strength of the influence.

7.9.10 Applications of Neural Networks

Neural Networks in Practice

Given this description of neural networks and how they work, what real world applications are they suited for? Neural networks have broad applicability to real world business problems. In fact, they have already been successfully applied in many industries.

Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

- Sales forecasting
- Industrial process control
- Customer research
- Data validation
- Risk management
- Target marketing

But to give you some more specific examples; ANN are also used in the following specific paradigms: recognition of speakers in communications; diagnosis of hepatitis; recovery of telecommunications from faulty software; interpretation of multimeaning Chinese words; undersea mine detection; texture analysis; three-dimensional object recognition; hand-written word recognition; and facial recognition.

Neural Network in Medicine

Artificial Neural Networks (ANN) are currently a 'hot' research area in medicine and it is believed that they will receive extensive application to biomedical systems in the next few years. At the moment, the research is mostly on modeling parts of the human body and recognising diseases from various scans (e.g. cardiograms, CAT scans, ultrasonic scans, etc.).

Neural networks are ideal in recognising diseases using scans since there is no need to provide a specific algorithm on how to identify the disease. Neural networks learn by example so the details of how to recognise the disease are not needed. What is needed is a set of examples that are representative of all the variations of the disease. The quantity of examples is not as important as the 'quantity'. The examples need to be selected very carefully if the system is to perform reliably and efficiently.

Electronic Noses

ANNs are used experimentally to implement electronic noses. Electronic noses have several potential applications in telemedicine. Telemedicine is the practice of medicine over long distances via a communication link. The electronic nose would identify odours in the remote surgical environment. These identified odours would then be electronically transmitted to another site where a door generation system would recreate them. Because the sense of smell can be an important sense to the surgeon, telesmell would enhance telepresent surgery.

Neural Network in Business

Business is a diverted field with several general areas of specialization such as accounting or financial analysis. Almost any neural network application would fit into one business area or financial analysis.

There is some potential for using neural networks for business purposes, including resource allocation and scheduling. There is also a strong potential for using neural networks for database mining, that is, searching for patterns implicit within the explicitly stored information in databases. Most of the funded work in this area is classified as proprietary. Thus, it is not possible to report on the full extent of the work going on. Most work is applying neural networks, such as the Hopfield-Tank network for optimization and scheduling.

Instant Physician

An application developed in the mid-1980s called the "instant physician" trained an auto associative memory neural network to store a large number of medical records, each of which includes information on symptoms, diagnosis, and treatment for a particular case. After training, the net can be presented with input consisting of a set of symptoms; it will then find the full stored pattern that represents the "best" diagnosis and treatment.

7.10 RULE INDUCTION

Rule induction is one of the major forms of data mining and is perhaps the most common form of knowledge discovery in unsupervised learning systems. It is also perhaps the form of data mining that most closely resembles the process that most people think about when they think about data mining, namely "mining" for gold through a vast database. The gold in this case would be a rule that is interesting - that tells you something about your database that you didn't already know and probably weren't able to explicitly articulate (aside from saying "show me things that are interesting").

Rule induction on a data base can be a massive undertaking where all possible patterns are systematically pulled out of the data and then an accuracy and significance are added to them that tell the user how strong the pattern is and how likely it is to occur again. In general these rules are relatively simple such as for a market basket database of items scanned in a consumer market basket you might find interesting correlations in your database such as:

- If bagels are purchased then cream cheese is purchased 90% of the time and this pattern occurs in 3% of all shopping baskets.
- If live plants are purchased from a hardware store then plant fertilizer is purchased 60% of the time and these two items are bought together in 6% of the shopping baskets.

Automating the process of culling the most interesting rules and of combing the recommendations of a variety of rules are well handled by many of the commercially available rule induction systems on the market today and is also an area of active research.

Applying Rule Induction to Business

Rule induction systems are highly automated and are probably the best of data mining techniques for exposing all possible predictive patterns in a database. They can be modified to for use in prediction problems but the algorithms for combining evidence from a variety of rules comes more from rules of thumbs and practical experience.

In comparing data mining techniques along an axis of explanation neural networks would be at one extreme of the data mining algorithms and rule induction systems at the other end. Rule induction systems when used for prediction on the other hand are like having a committee of trusted advisors each with a slightly different opinion as to what to do but relatively well grounded reasoning and a good explanation for why it should be done.

The business value of rule induction techniques reflects the highly automated way in which the rules are created which makes it easy to use the system but also that this approach can suffer from an overabundance of interesting patterns which can make it complicated in order to make a prediction that is directly tied to Return on Investment (ROI).

What is Rule?

In rule induction systems the rule itself is of a simple form of "if this and this and this then this". For example a rule that a supermarket might find in their data collected from scanners would be: "if pickles are purchased then ketchup is purchased'. Or

- If paper plates then plastic forks
- If dip then potato chips
- If salsa then tortilla chips

In order for the rules to be useful there are two pieces of information that must be supplied as well as the actual rule:

- Accuracy - How often is the rule correct?
- Coverage - How often does the rule apply?

Just because the pattern in the database is expressed as rule does not mean that it is true all the time. Thus just like in other data mining algorithms it is important to recognize and make explicit the uncertainty in the rule. This is what the accuracy of the rule means.

In some cases accuracy is called the confidence of the rule and coverage is called the support. Accuracy and coverage appear to be the preferred ways of naming these two measurements.

Rule	Accuracy	Coverage
If breakfast cereal purchased then milk purchased.	85%	20%

The rules themselves consist of two halves. The left hand side is called the antecedent and the right hand side is called the consequent. The antecedent can consist of just one condition or multiple conditions which must all be true in order for the consequent to be true at the given accuracy.

Discovery

The claim to fame of these ruled induction systems is much more so for knowledge discovers in unsupervised learning systems than it is for prediction. These systems provide both a very detailed view of the data where significant patterns that only occur a small portion of the time and only can be found when looking at the detail data as well as a broad overview of the data where some systems seek to deliver to the user an overall view of the patterns contained n the database. These systems thus display a nice combination of both micro and macro views:

1. *Macro Level:* Patterns that cover many situations are provided to the user that can be used very often and with great confidence and can also be used to summarize the database.
2. *Micro Level:* Strong rules that cover only a very few situations can still be retrieved by the system and proposed to the end user. These may be valuable if the situations that are covered are highly valuable (maybe they only apply to the most profitable customers) or represent a small but growing subpopulation which may indicate a market shift or the emergence of a new competitor (e.g. customers are only being lost in one particular area of the country where a new competitor is emerging).

How to Evaluate the Rule

One way to look at accuracy and coverage is to see how they relate so some simple statistics and how they can be represented graphically. From statistics coverage is simply the a priori probability of the antecedent and the consequent occurring at the same time. The accuracy is just the probability of the consequent conditional on the precedent. So, for instance the if we were looking at the following database of super market basket scanner data we would need the following information in order to calculate the accuracy and coverage for a simple rule (let's say milk purchase implies eggs purchased).

Example:

T = 100 = Total number of shopping baskets in the database.

E = 30 = Number of baskets with eggs in them.

M = 40 = Number of baskets with milk in them.

B = 20 = Number of baskets with both eggs and milk in them.

Accuracy is then just the number of baskets with eggs and milk in them divided by the number of baskets with milk in them. In this case that would be 20/40 = 50%. The coverage would be the number of baskets with milk in them divided by the total number of baskets. This would be 40/100 = 40%. This can be seen graphically in Figure 7.38.

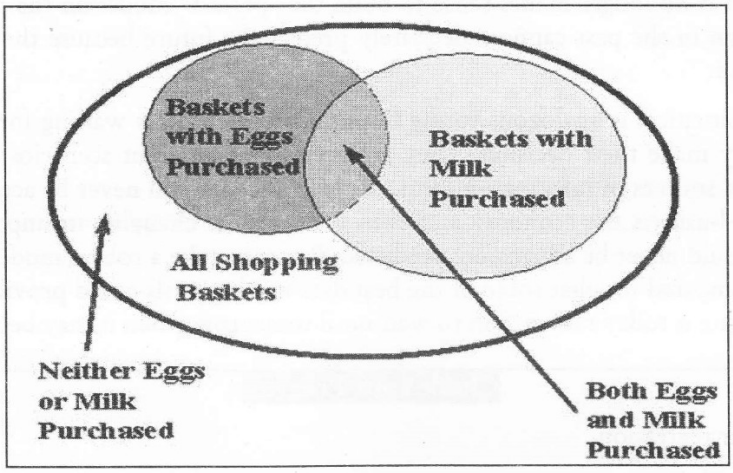


Figure 7.38

Graphically the total number of shopping baskets can be represented in a space and the number of baskets containing eggs or milk can be represented by the area of a circle. The coverage of the rule "If Milk then Eggs" is just the relative size of the circle corresponding to milk. The accuracy is the relative size of the overlap between the two to the circle representing milk purchased.

Notice that we haven't used E the number of baskets with eggs in these calculations. One way that eggs could be used would be to calculate the expected number of baskets with eggs and milk in them based on the independence of the events. This would give us some sense of how unlikely and how special the event is that 20% of the baskets have both eggs and milk in them. Remember from the statistics section that if two events are independent (have no effect on one another) that the product of their individual probabilities of occurrence should equal the probability of the occurrence of them both together.

If the purchase of eggs and milk were independent of each other one would expect that 0.3 × 0.4 = 0.12 or 12% of the time we would see shopping baskets with both eggs and milk in them. The fact that this combination of products occurs 20% of the time is out of the ordinary if these events were independent. That is to say there is a good chance that the purchase of one affects the other and the degree to which this is the case could be calculated through statistical tests and hypothesis testing.

7.11 WHICH TECHNIQUE AND WHEN?

Clearly one of the hardest things to do when deciding to implement a data mining system is to determine which technique to use when. When are neural networks appropriate and when are decision trees appropriate? When is data mining appropriate at all as opposed to just working with relational databases and reporting? When would just using OLAP and a multidimensional database be appropriate?

Some of the criteria that are important in determining the technique to be used are determined by trial and error. There are definite differences in the types of problems that are most conducive to each technique but the reality of real world data and the dynamic way in which markets, customers and hence the data that represents them is formed means that the data is constantly changing. These dynamics mean that it no longer makes sense to build the "perfect" model on the historical data since whatever was known in the past cannot adequately predict the future because the future is so unlike what has gone before.

In some ways this situation is analogous to the business person who is waiting for all information to come in before they make their decision. They are trying out different scenarios, different formulae and researching new sources of information. But this is a task that will never be accomplished - at least in part because the business the economy and even the world is changing in unpredictable and even chaotic ways that could never be adequately predicted. Better to take a robust model that perhaps is an under-performer compared to what some of the best data mining tools could provide with a great deal of analysis and execute it today rather than to wait until tomorrow when it may be too late.

Check Your Progress

1. Define linear regression.
2. Define clustering.
3. State whether the following statements are true or false:
   (i) Statistical techniques are not data mining.
   (ii) The decision tree technology cannot be used for exploration of the dataset and business problem.
4. Fill in the blanks:
   (i) Neural networks are very powerful..... modeling techniques
   (ii) Clustering is the method by which like records are..... together.

7.12 LET US SUM UP

A cross-functional team was formed that included clinical, financial, and technical expertise. Business problems were formulated at different levels of abstraction in order to identify financial opportunities. We had a set of tools that we developed and bought, and we tried executing these tools in various combinations with one another of data mining techniques. We eventually came up with a repeatable methodology. We discovered that we can apply different data mining techniques to find optimized regions and patterns, respectively, by performing complex algorithmic steps. And then after running those tools, we could take a region and points outside the region and compare them, using OLAP, which allows us to compare the region and outer region attribute by attribute. In other words, standard OLAP reports can be run in which each dimension or attribute (payer, admission source, patient age, physician perspectives, marital status, and so on) can be described. If any of the reports describing an attribute is interesting, then a set of follow-up, detailed OLAP reports can be run.

We showed how to apply the methodology to an analysis of business data. We explained each step and the results of running each step; we illustrated how this method helped turn data into knowledge.

Results were presented in a graphically appealing way that helped users determine how to use the information to make decisions.

7.13 KEYWORDS

Online Analytical Processing (OLAP): Departmental processing for the data mart environment.

Induction: Organize a group.

Parameter: An elementary data value used as a criterion for qualification, usually of data searches or in the control of modules.

7.14 QUESTIONS FOR DISCUSSION

1. Explain the meaning of the term:
   (i) Induction
   (ii) Rule
   (iii) Clustering
2. What do you mean by decision tree?
3. Explain the neural networks.
4. What is linear regression?
5. Differentiate between Clustering and Nearest Neighbor Technique.
6. Discuss how Neural Networks and Decision trees are applied to Business.
7. Discuss statistical techniques.

Check Your Progress: Model Answers

1. The simplest form of regression is simple linear regression that just contains one predictor and a prediction.
2. Clustering is the method by which like records are grouped together.
3. (i) True (ii) False
4. (i) Predictive, (ii) grouped

7.15 SUGGESTED READINGS

David L. Olson, *Introduction to Business Data Mining*, McGraw Hill, 2007.

Michael J.A. Berry, *Data Mining Techniques*, Gordon Linoff.

UNIT V

LESSON

8

MINING ASSOCIATION RULES

CONTENTS

8.0 AIMS AND OBJECTIVES

After studying this lesson, you will be able to:

- Explain association rule mining
- Describe market basket analysis
- Discuss apriori algorithm
- State mining multidimensional association rules from relational databases and data warehouses
- Explain association rules technique

8.1 INTRODUCTION

Association is the discovery of association relationships or correlations among a set of **large item sets.** They are often expressed in the rule form showing attribute-value conditions that occur frequently together in a given set of data. An association rule in the form of $X \rightarrow Y$ is interpreted as 'database tuples that satisfy X are likely to satisfy Y'. Association analysis is widely used in transaction data analysis for direct marketing, catalog design, and other business decision-making process. Substantial research has been performed recently on association analysis with efficient algorithms proposed, including the level-wise Apriori search, mining multiple-level, multi-dimensional associations, mining associations for numerical, categorical, and interval data, meta-pattern directed or constraint-based mining, and mining correlations.

Association analysis is based on the *association rules.* It studies the frequency of items occurring together in transactional databases, and based on a threshold called *support*, identifies the frequent item sets. Another threshold, *confidence*, which is the conditional probability than an item appears in a transaction when another item appears, is used to pinpoint association rules. Association analysis is commonly used for market basket analysis. For example, it could be useful for the OurVideoStore manager to know what movies are often rented together or if there is a relationship between renting a certain type of movies and buying popcorn or pop. The discovered association rules are of the form: $P \rightarrow Q$ [s, c], where P and Q are conjunctions of attribute value-pairs, and s (for support) is the probability that P and Q appear together in a transaction and c (for confidence) is the conditional probability that Q appears in a transaction when P is present. For example, the hypothetic association rule

Rent Type (X, "game") $\wedge$ Age(X,"13-19") $\rightarrow$ Buys(X, "pop") [s=2% ,c=55%]

would indicate that 2% of the transactions considered are of customers aged between 13 and 19 who are renting a game and buying a pop, and that there is a certainty of 55% that teenage customers, who rent a game, also buy pop.

This lesson gives basic introduction to association rule mining, which is a technique to find out some rare but useful and interesting relationships among a large set of data items. These relationships may be very useful in sales and marketing.

8.2 ASSOCIATION RULE MINING

Association rule mining finds interesting association or correlation relationships among a large set of data items. With massive amounts of data continuously being collected and stored in databases, many industries are becoming interested in mining association rules from their databases. A typical example of association rule mining is market basket analysis which is discussed later in this lesson. This process analyzes customer's buying habits by finding associations between the different items that customers place in their "shopping baskets". The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For instance, if customers are buying milk, they are likely to buy bread (and what kind of bread) also on the same trip to the supermarket. Such information can lead to increased sales by helping retailers to do selective marketing and plan their shelf space. For instance, placing milk and bread within close proximity may further encourage the sale of these items together within single visits to the store.

8.2.1 Basic Concepts

Let $I=\{i_1,i_2,\ldots, i_m\}$ be a set of items. Let D, the task relevant data, be a set of database transactions where each transaction T is a set of items such that $T \subseteq I$. Each transaction is associated with an identifier, called TID. Let A be a set of items. A transaction T is said to contain A if and only if $A \subseteq T$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subset I$, $B \subset I$ and $A \cap B = \phi$. The rule $A \Rightarrow B$ holds in the transaction set D with support s, where s is the percentage of transactions in D that contain $A \cup B$. The rule $A \Rightarrow B$ has confidence c in the transaction set D if c is the percentage of transactions in D containing A which also contain B. That is,

$$support(A \Rightarrow B) = Prob\{A \cup B\}$$

$$confidence(A \cup B) = Prob\{B|A\}.$$

Rules that satisfy both a minimum support threshold *(min-sup)* and a minimum confidence threshold *(mm.conf)* are called strong.

A set of items is referred to as an itemset. An itemset that contains k items is a k-itemset. The set *{computer, financial-management-software}* is a 2-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the frequency or support count of the itemset. An itemset satisfies minimum support if the occurrence frequency of the itemset is greater than or equal to the product of *man's* and the total number of transactions in D. If an itemset satisfies minimum support, then it is a frequent itemset. The set of frequent k-itemsets is commonly denoted by L_k [2].

Association rule mining for large databases is a two-step process:

Step 1: Find all frequent itemsets. By definition, each of these itemsets will occur at least as frequently as a pre-determined minimum support count.

Step 2: Generate strong association rules from the frequent itemsets. By definition, these rules must satisfy minimum support and minimum confidence.

Additional interestingness measures can be applied, if desired. The second step is the easier of the two. The overall performance of mining association rules is determined by the first step.

8.2.2 Association Rule Mining: A Road Map

There are many kinds of association rules. Market basket analysis is also just one form of *association rule mining.* Association rules can be classified in various ways, based on the following criteria:

1. ***Based on the types of values handled in the rule:*** If a rule concerns associations between the presence or absence of items, it is a Boolean association rule. For example,

   Rule: buys (x, milk) → buys (x, bread) [25%, 60.0%] is a Boolean association rule obtained from market basket analysis.

   If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule. In these rules, quantitative values for items or attributes are partitioned into intervals. For example,

   Rule: age(X, "30 - 34") a *income(X,* "42A' - 48A'") ⇒ *buys(X, "high resolution TV")* is an example of a quantitative association rule.

   Note that the quantitative attributes, *age* and *income,* have been discretized.

2. ***Based on the dimensions of data involved in the rule:*** If the items or attributes in an association rule each reference only one dimension, then it is a single-dimensional association rule. For example,

   Rule: buys (x, milk) ⇒ buys (x, bread) [25%, 60.0%]

   Is also the example of single-dimensional association rule since it refers to only one dimension, i.e., *buys.*

   If a rule references two or more dimensions, such as the dimensions *buys, time-of Jransaction,* and *customer-category,* then it is a multidimensional association rule.

   Rule: age(X, "30 - 34") A *income(X,* "42A' - 48A'") ⇒ *buys(X, "high resolution TV")* is also considered a multidimensional association rule since it involves three dimensions,*age, income,* and *buys.*

3. ***Based on the levels of abstractions involved in the rule set:*** Some methods for association rule mining can find rules at differing levels of abstraction. For example, suppose that a set of association rules mined as below:

   Rule A: age(X, "30 - 34") ⇒ *buys(X, "laptop computer")*

   Rule B: age(X, "30 - 34") ⇒ *buys(X, "computer")*

   In the above rules, Rule A and Rule B, the items bought are referenced at different levels of abstraction. (That is, *"computer''* is a higher level abstraction of *"laptop computer").* We refer to the rule set mined as consisting of multilevel association rules. If, instead, the rules within a given set do not reference items or attributes at different levels of abstraction, then the set contains single-level association rules.

4. ***Based on the nature of the association involved in the rule:*** Association mining can be extended to correlation analysis, where the absence or presence of correlated items can be identified.

8.3 MINING SINGLE-DIMENSIONAL ASSOCIATION RULES FROM DATABASES

8.3.1 Parallel and Distributed Algorithms

These algoritnms are based on the kernel that employs the well known Apriori algorithm which is discussed below.

Apriori Algorithm: Finding Frequent Item Sets

Apriori is an influential algorithm for mining frequent item sets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses *prior knowledge* of frequent item set properties. Apriori employs an iterative approach known as a *level-wise* search, where k-item sets are used to explore *(k+1)-item sets.*

The following algorithm Frequent Item sets is used to generate all frequent item sets in a given database D. This is the Apriori algorithm. The parameter db size is the total number of tuples in the database.

Algorithm - Frequent Item Sets

```
begin
Input: D: data set; minsupp: minimum support;
```

```
Output: L: frequent item sets;
let frequent item set L ← {};
let frontier set F ← {{}};
while F ≠ {} do begin
-make a pass over the database D
let candidate set C ← {};
for all database tuples t do
for all item sets f in F do
if it contains f then begin
let Cf ← candidate itemsets that are extensions of f and contained in t;
for all itemsets cf in Cf do
if cf ∈ C then
cf .count ← cf .count + 1;
else
cf .count ← 0; C ← C ∪ {cf };
end
-consolidate
let F ← {};
for all itemsets c in C do begin
if c.count/dbsize > minsupp then
L ← L ∪ c;
if c should be used as a frontier in the next pass then
F ← F ∪ c;
end
end
end
```

The Apriori algorithm makes multiple passes over a given database. The *frontier set* for a pass consists of those item sets that are extended during the pass. In each pass, the support for certain item sets is measured. These item sets, referred to *candidate item sets*, are derived from the tuples in the database and the item sets contained in the frontier set.

Associated with each item set is a counter that stores the number of transactions in which the corresponding item set has appeared. This counter is initialized to zero when an item set is created.

Initially, the frontier set consists of only one element, which is an empty set. At the end of a pass, the support for a candidate item set is compared to *minsupp* to determine whether it is a frequent itemset. At the same time, it is determined whether this item set should be added to the frontier set for the next pass. The algorithm terminates when the frontier set becomes empty. The support count for the itemset is preserved when an item set is added to the frequent/frontier set.

However, in a given large database, the Apriori algorithm *Frequent Item sets* used for identifying frequent item sets, involves a search with little heuristic information in a space with an exponential amount of items and possible item sets. This algorithm may suffer from large computational overheads when the number of frequent item sets is very large.

Table 8.1: A Transaction Database

TID	Items				
100	A		C	D	
200		B	C		E
300	A	B	C		E
400		B			E

For example, suppose there are 1000 items in a given large database, the average number of items in each transaction is 6. Then there are almost 10^{15} possible item sets to be counted in the database. To illustrate the use of the Apriori algorithm, we use the data in Table 8.1 where *minsupp* = 50%.

Firstly, the 1-itemsets *{A}*, *{B}*, *{C}*, *{D}*, and *{E}* are generated as candidates at the first pass over the dataset, where *A.count* = 2, *B.count* = 3, *C.count* = 3, *D.count* = 1, and *E.count* = 3. Because *minsupp* = 50% and *dbsize* = 4, *{A}*, *{B}*, *{C}*, and *{E}* are frequent itemsets.

Table 8.2: Frequent 1-itemsets in the dataset in Table 8.1

Itemsets	Frequency	> *minsupp*
{A}	2	y
{B}	3	y
{C}	3	y
{E}	3	y

Secondly, the frequent 1-itemsets {A}, {B}, {C}, and {E} are appended into the frontier set F, and the second pass begins over the dataset to search for 2-itemset candidates. Each such candidate is a subset of F. The 2-itemset candidates are *{A,B}*, *{A,C}*, *{A,E}*, *{B,C}*, *{B,E}*, and *{C,E}*, where $A \cup B.count = 1$, $A \cup C.count = 2$, $A \cup E.count = 1$, $B \cup C.count = 2$, $B \cup E.count = 3$, and $C \cup E.count = 2$.$A \cup C$, $B \cup C$, $B \cup E$ and $C \cup E$ are frequent itemsets. They are listed in Table 8.3

Table 8.3: Frequent 2-itemsets in the dataset in Table 8.1

Itemsets	Frequency	> *minsupp*
{A,C}	2	y
{B,C}	2	y
{B,E}	3	y
{C,E}	2	y

Thirdly, the frequent 1-itemsets and 2-itemsets are appended into the frontier set *F*, and the third pass begins over the dataset to search for 3-itemset candidates. Frequent 3-itemsets are listed in Table 8.4.

Table 8.4: Frequent 3-itemsets in the dataset in Table 8.1

Itemsets	Frequency	> *minsupp*
{B,C,E}	2	y

Fourthly, the frequent 1-itemsets, 2-itemset, and 3-itemsets are appended into the frontier set *F*, and the fourth pass begins over the dataset to search for 4-itemset candidates. There is no frequent 4-itemset, and the algorithm is ended.

8.3.2 Generating Association Rules from Frequent Item Sets

Once the frequent item sets from transactions in a database D have been found, it is straightforward to generate strong association rules from them (where *strong* association rules satisfy both minimum support and minimum confidence). This can be done using the following equation for confidence, where the conditional probability is expressed in terms of itemset support:

$$\text{Confidence}\ (A \Rightarrow B) = \text{Prob}(B|A) = \frac{\text{support}(A \cup B)}{\text{support}(A)}$$

where, *support*$(A \cup B)$ is the number of transactions containing the item sets $A \cup B$, and *support*(A) is the number of transactions containing the item set A.

Based on this equation, association rules can be generated as follows.

- For each frequent item set, /, generate all non-empty subsets of /.
- For every non-empty subset s, of /, output the rule "$s \Rightarrow (l - s)$" if $\frac{\text{support}(l)}{\text{support}(s)} \geq$ min_*conf*, where *min_.conf* is the minimum confidence threshold.

Since the rules are generated from frequent item sets, then each one automatically satisfies minimum support. Frequent item sets can be stored ahead of time in hash tables along with their counts so that they can be accessed quickly.

8.3.3 Variations of the Apriori Algorithm

Many variations of the Apriori algorithm have been proposed. A number of these variations are enumerated below. Methods 1 to 6 focus on improving the efficiency of the original algorithm, while methods 7 and 8 consider transactions over time:

1. ***A Hash-based Technique:*** Hashing item set counts. A hash-based technique can be used to reduce the size of the candidate k-item sets, C_k, for $k > 1$.
2. ***Scan Reduction:*** Reducing the number of database scans. Recall that in the Apriori algorithm, one scan is required to determine *Lk for each Ck*- A scan reduction technique reduces the total number of scans required by doing extra work in some scans.
3. ***Transaction Reduction:*** Reducing the number of transactions scanned in future iterations. A transaction which does not contain any frequent fc-item sets cannot contain any frequent $(k + 1)$-item sets. Therefore, such a transaction can be marked or removed from further consideration since subsequent scans of the database for j-item sets, where $j > k$, will not require it.
4. ***Partitioning:*** Partitioning the data to find candidate item sets. A partitioning technique can be used which requires just two database scans to mine the frequent item sets It consists of two phases. In Phase I, the algorithm subdivides the transactions of D into n non-overlapping partitions. In Phase II, a second scan of D is conducted in which the actual support of each candidate is assessed in order to determine the global frequent item sets. Partition size and the number of partitions are set so that each partition can fit into main memory and therefore be read only once in each phase.
5. ***Sampling:*** Mining on a subset of the given data. The basic idea of the sampling approach is to pick a random sample S of the given data D, and then search for frequent item sets in S instead D. In

this way, we trade off some degree of accuracy against efficiency. The sample size of S is such that the search for frequent item sets in S can be done in main memory, and so, only one scan of the transactions in S is required overall. The sampling approach is especially beneficial when efficiency is of utmost importance, such as in computationally intensive applications that must be run on a very frequent basis.

6. ***Dynamic item set counting:*** Adding candidate item sets at different points during a scan. A dynamic item set counting technique was proposed in which the database is partitioned into blocks marked by start points. In this variation, new candidate item sets can be added at any start point, unlike in Apriori, which determines new candidate item sets only immediately prior to each complete database scan. The technique is dynamic in that it estimates the support of all of the item sets that have been counted so far, adding new candidate item sets if all of their subsets are estimated to be frequent. The resulting algorithm requires two database scans.

7. ***Calendric market basket analysis:*** Finding item sets that are frequent in a set of user-defined time intervals. Calendric market basket analysis uses transaction time stamps to define subsets of the given database. An item set that does not satisfy minimum support may be considered frequent with respect to a subset of the database which satisfies user-specified time constraints.

8. ***Sequential patterns:*** Finding sequences of transactions associated over time. The goal of sequential pattern analysis is to find sequences of item sets that many customers have purchased in roughly the same order. A transaction sequence is said to contain an item set sequence if each item set is contained in one transaction, and the following condition is satisfied: If the ith item set in the item set sequence is contained in transaction j in the transaction sequence, then the $(i + 1)$th item set in the item set sequence is contained in a transaction numbered greater than j. The support of an item set sequence is the percentage of transaction sequences that contain it.

8.4 MINING MULTI-DIMENSIONAL ASSOCIATION RULES FROM RELATIONAL DATABASES AND DATA WAREHOUSES

8.4.1 Multi-dimensional Association Rules

The Boolean, single-dimensional association rules consider only one predicate and only the presence/absence of an item. In contrast, multi-dimensional association rules consider multiple predicates, and quantitative association rules consider items (attributes) that may assume a range of values instead of just two values.

The motivation for multidimensional association rules stems from data warehouses, which store multidimensional data. In this case, items correspond to features that describe different dimensions, and thus are associated with different predicates. An example of multidimensional rule is shown below.

Rule: age(X, "30 - 34") A income(X, "42A' - 48A'") ⇒ buys(X, "high resolution TV")

The above rule is considered a multidimensional association rule since it involves three dimensions, *age, income,* and *buys.*

Such a rule is referred to as an interdimension association rule, since none of the predicates is repeated. In contrast, the hybrid-dimension association rule incorporates some of the predicates multiple times (see the following example).

Example: An association rule that describes graduate students major(x, Computer Engineering) AND takes_course(x, Advanced Data Analysis and Decision Making) AND takes_course(x, Data Mining) ⇒ level(x, PhD) [0.9%, 90%]

Some of the data attributes are discrete (categorical) and continuous, in contrast to Boolean attributes, which are used in generic association mining. All attribute types can be categorized as nominal, in which case there is no ordering between their values, and ordinal, in which case some ordering exists. An example discrete ordinal attribute is *age*, while an example discrete nominal attribute is *major*. The continuous attributes must be preprocessed before being used for association mining. The preprocessing boils down to discretization, which divides the entire range of the attribute values into subintervals and associates a discrete value with each of the intervals. For instance, *age* can be divided into subintervals [0, 9], [9, 18], [18, 30], [30, 65], [65, 120], and these intervals can be associated with the following discrete values: *child*, *teenager*, *young_adult*, *senior*. The discretization can be performed manually, based on an associated attribute hierarchy and/or an expert's knowledge. In the case of continuous ordinal attributes, discretization can also be performed automatically. A quantitative association rule uses discrete and/or discretized continuous items (attributes).

The multidimensional and quantitative association rules can be generated using the same algorithms as the Boolean, single-dimensional rules. These include the Apriori algorithm and its modifications, such as hashing, partitioning, sampling, etc. The main difference is in how the input transactional data, is prepared. Namely, the continuous attributes must be discretized and the algorithm must search through all relevant attributes (attribute-value pairs) together, instead of searching just one attribute (predicate).

8.4.2 Techniques for Mining Multi-dimensional Association Rules

Techniques for mining multidimensional association rules can be categorized according to three basic approaches regarding the treatment of quantitative (continuous-valued) attributes:

1. In the first approach, *quantitative attributes are discretized using predefined concept hierarchies.* This discretization occurs prior to mining. For instance, a concept hierarchy for *income* may be used to replace the original numeric values of this attribute by ranges, such as *"0-20K"*, *"21-30K"*, *"31-40K"*, and so on. Here, discretization is *static* and predetermined. The discretized numeric attributes, with their range values, can then be treated as categorical attributes (where each range is considered a category). We refer to this as mining multidimensional association rules using static discretization of quantitative attributes.
2. In the second approach, *quantitative attributes are discretized into "bins" based on the distribution of the data.* These bins may be further combined during the mining process. The discretization process is *dynamic* and established so as to satisfy some mining criteria, such as maximizing the confidence of the rules mined. Because this strategy treats the numeric attribute values as quantities rather than as predefined ranges or categories, association rules mined from this approach are also referred to as quantitative association rules.
3. In the third approach, *quantitative attributes are discretized so as to capture the semantic meaning of such interval data.* This dynamic discretization procedure considers the distance between data points. Hence, such quantitative association rules are also referred to as distancebased association rules.

8.5 MEASURING THE QUALITY OF RULES

Association rule mining finds interesting association or correlation relationships among a large set of data items. With massive amounts of data continuously being collected and stored in databases, many industries are becoming interested in mining association rules from their databases. A typical example of association rule mining is market basket analysis. This process analyzes customer's buying habits by finding associations between the different items that customers place in their "shopping baskets". The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For instance, if customers are buying milk, they are likely to buy bread (and what kind of bread) also on the same trip to the supermarket. Such information can lead to increased sales by helping retailers to do selective marketing and plan their shelf space. For instance, placing milk and bread within close proximity may further encourage the sale of these items together within single visits to the store.

Market Basket Analysis: A Motivating Example for Association Rule Mining

Suppose, as manager of a computer shop, you would like to learn more about the buying habits of your customers. Specifically, you will be interested to know the groups or sets of items customers are likely to purchase on a given trip to the shop. For this you have to perform market basket analysis on the retail data of customer transactions at your store. The results may be used to plan marketing or advertising strategies, as well as catalog design. For instance, market basket analysis may help managers design different store layouts. In one strategy, items that are frequently purchased together can be placed in close proximity in order to further encourage the sale of such items together. If customers who purchase computers also tend to buy financial management software at the same time, then placing the hardware display close to the software display may help to increase the sales of both of these items. Similarly, market basket analysis can also help retailers to plan which items to put on sale at reduced prices. If customers tend to purchase computers and printers together, then having a sale on printers may encourage the sale of printers as *well as* computers.

Continuing our example of market-basket analysis, we represent each product in the shop as a Boolean variable, which represents whether an item is present or absent. Each customer's basket is represented as a Boolean vector, denoting which items are purchased. The vectors are analyzed to find which products are frequently bought together (by different customers), i.e., associated with each other. These co-occurrences are represented in the form of association rules:

$$\text{LHS} \rightarrow \text{RHS [support, confidence]}$$

Where, the Left-hand Side (LHS) implies the Right-hand Side (RHS), with a given value of support and confidence.

Support and confidence are used to **measure the quality of a given rule**, in terms of its usefulness (strength) and certainty. Support tells how many examples (transactions) from a data set that was used to generate the rule include items from both LHS and RHS. Confidence expresses how many examples (transactions) that include items from LHS also include items from RHS. Measured values are most often expressed as percentages. An association rule is considered interesting if it satisfies minimum values of confidence and support, which are to be specified by the user (domain expert). Typically, association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold. Such thresholds can be set by users or domain experts.

Example: To understand the support and confidence

We present a small example database in Table 8.5 to illustrate the support and confidence measures. There are five transactions in the database, so the support of each itemset is measured relative to 5.

Table 8.5

Transaction ID	Items
T_1	A, C, D
T_2	B, E
T_3	A, B, C, E
T_4	B, E
T_5	B, D, F

Table 8.6 shows the supports of the 6 items in the database. The first column lists the item, the second lists the number of transactions in which the item appears, and the third column lists the support of the item.

Table 8.6

Item	Number of transactions	Support sup (X)
A	2	40%
B	4	80%
C	2	40%
D	2	40%
E	3	60%
F	1	20%

Table 8.7 shows the support for some item sets derived from the database. For example, the item set {A, B} in the second row appears in only one transaction, transaction T_3, which gives it a support of 20%.

Table 8.7

Item set	Number of Transactions	Support sup (X)
A, C	2	40%
A, B	1	20%
B, D	1	20%
C, D	1	20%
A, B, C	1	20%
A, B, E	1	20%
A, C, D	1	20%
B, D, F	1	20%
A, B, C, E	1	20%

Table 8.8 shows the confidence measures of several association rules derived from the item sets in Table 8.7 The confidence of 100% for the rule A C means that in every transaction in which A appears, C also appears. The confidence of this rule can be calculated by dividing the number of transactions in which the item set {A, C} appears, which is 2 (see Table 8.7), by the number of transactions in which the item A appears, also 2 (Table 8.6).

Table 8.8

Association Rule	Confidence conf (X Y)
A → C	100%
A → B	50%
B → D	25%
A, B → C	100%
A, C → B	50%
B, E → A	33%

The following examples are used to illustrate the concepts.

Example: An association rule that describes customers who buy milk and bread.

buys (x, milk) → buys (x, bread) [25%, 60.0%]

The rule shows that customers who buy milk also buy bread. The direction of the association, from left to right, shows that buying milk "triggers" buying bread. These items are bought together in 25% of store purchases (transactions), and 60% of the baskets that include milk also include bread.

Example: An association rule describing graduate students might read as follows:

Major (x, Computer Engineering) AND takes_course(x, Advanced Data Analysis and Decision

Making) → level (x, PhD) [1%, 75%]

The rule has two items in the LHS and one item in the RHS. Association rules can include multiple items in their LHS. The rule in this example states that students who major in Computer Engineering and who take Advanced Data Analysis and Decision Making course are at the Ph.D. level with 1% support and 75% confidence. Again, the support shows that 1% of all students in the database satisfy this association. At the same time, among those who major in Computer Engineering and who take Advanced Data Analysis and Decision Making courses, 75% are the Ph.D. students.

Check Your Progress

Fill in the blanks:

1. ..... is the discovery of association relationships or correlations among a set of items.
2. Association rule mining finds interesting association or correlation relationships among a ..... set of data items.
3. A typical example of association rule mining is ..... .
4. ..... and ..... are used to measure the quality of a given rule, in terms of its usefulness (strength) and certainty.
5. An association rule is an implication of the form ....., where $A \subset I$, $B \subset I$ and $A \cap B = \phi$.
6. ..... is an influential algorithm for mining frequent item sets for Boolean association rules.

Contd...

7. The name of the algorithm is based on the fact that the algorithm uses ..... knowledge of frequent item set properties.
8. The Apriori algorithm makes ..... passes over a given database.
9. ..... market basket analysis is used to find item sets that are frequent in a set of user-defined time intervals.
10. A ..... association rule uses discrete and/or discretized continuous items (attributes).

8.6 LET US SUM UP

Association is the discovery of association relationships or correlations among a set of items. They are often expressed in the rule form showing attribute-value conditions that occur frequently together in a given set of data. An association rule in the form of $X \rightarrow Y$ is interpreted as 'database tuples that satisfy X are likely to satisfy Y'.

Apriori is an influential algorithm for mining frequent item sets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses *prior knowledge* of frequent item set properties. Many variations of the Apriori algorithm have been proposed as: A hash-based technique, Scan reduction, Transaction reduction, Partitioning, Sampling, Dynamic item set counting, Calendric market basket analysis and Sequential patterns.

The motivation for multidimensional association rules stems from data warehouses, which store multidimensional data. In this case, items correspond to features that describe different dimensions, and thus are associated with different predicates.

Techniques for mining multidimensional association rules can be categorized according to three basic approaches regarding the treatment of quantitative (continuous-valued) attributes.

Association rule mining finds interesting association or correlation relationships among a large set of data items. With massive amounts of data continuously being collected and stored in databases, many industries are becoming interested in mining association rules from their databases. A typical example of association rule mining is market basket analysis. Support and confidence are used to measure the quality of a given rule, in terms of its usefulness (strength) and certainty. Support tells how many examples (transactions) from a data set that was used to generate the rule include items from both LHS and RHS. Confidence expresses how many examples (transactions) that include items from LHS also include items from RHS. Measured values are most often expressed as percentages. An association rule is considered interesting if it satisfies minimum values of confidence and support, which are to be specified by the user (domain expert). Typically, association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold. Such thresholds can be set by users or domain experts.

8.7 KEYWORDS

Item Set: A set of items is referred to as an item set.

k-item Set: An item set that contains k items is a k-item set.

Boolean Association Rule: If a rule concerns associations between the presence or absence of items, it is a Boolean association rule.

Quantitative Association Rule: If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule.

Apriori Algorithm: Apriori is an influential algorithm for mining frequent item sets for Boolean association rules.

Association: Association is the discovery of association relationships or correlations among a set of items.

Association Rule Mining: Association rule mining finds interesting association or correlation relationships among a large set of data items.

Market Basket Analysis: A typical example of association rule mining is market basket analysis. This process analyzes customer's buying habits by finding associations between the different items that customers place in their "shopping baskets".

Support and Confidence: These terms are used ed to measure the quality of a given rule, in terms of its usefulness (strength) and certainty.

8.8 QUESTIONS FOR DISCUSSION

1. How do you find frequent item sets with the help of apriori algorithm?
2. How do you generate association rules from frequent item sets?
3. Discuss the following variations of apriori algorithm:
   (i) Hash-based technique (ii) Transaction reduction
   (iii) Sampling (iv) Market basket analysis
4. Write a short note on "Mining multi-dimensional association rules from relational databases and data warehouses".
5. Discuss the techniques for mining multi-dimensional association rules.

Check Your Progress: Model Answers

1. Association	2. Large	3. Market basket analysis
4. Support, confidence	5. $A \Rightarrow B$	6. Apriori
7. Prior	8. Multiple	9. Calendric
10. Quantitative		

8.9 SUGGESTED READINGS

Jiawei Han, Micheline Kamber, *Data Mining - Concepts and Techniques*, Morgan Kaufmann Publishers, First Edition, 2003.

Michael J. A. Berry, Gordon S Linoff, *Data Mining Techniques*, Wiley Publishing Inc, Second Edition, 2004.

Alex Berson, Stephen J. Smith, *Data warehousing, Data Mining & OLAP*, Tata McGraw Hill, Publications, 2004.

Sushmita Mitra, Tinku Acharya, *Data Mining - Multimedia, Soft Computing and Bioinformatics*, John Wiley & Sons, 2003.

Sam Anohory, Dennis Murray, *Data Warehousing in the Real World*, Addison Wesley, First Edition, 2000.