Oracle® Data Mining

Concepts

11*g* Release 2 (11.2)

E16808-06

July 2011

Oracle Data Mining Concepts, 11*g* Release 2 (11.2)

E16808-06

Primary Author: Kathy L. Taylor

Contents

Part II Mining Functions

4 Regression

5 Classification

6 Anomaly Detection

7 Clustering

8 Association

9 Feature Selection and Extraction

Part III Algorithms

10 Apriori

11 Decision Tree

12 Generalized Linear Models

13 *k*-Means

14 Minimum Description Length

15 Naive Bayes

16 Non-Negative Matrix Factorization

17 O-Cluster

18 Support Vector Machines

Part IV Data Preparation

19 Automatic and Embedded Data Preparation

Part V Mining Unstructured Data

20 Text Mining

Glossary

Index

Preface

This manual describes the features of Oracle Data Mining, a comprehensive data mining solution within Oracle Database. It explains the data mining algorithms, and and it lays a conceptual foundation for much of the procedural information contained in other manuals. (See "Related Documentation".)

The preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documentation
- Conventions

Audience

Oracle Data Mining Concepts is intended for analysts, application developers, and data mining specialists.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documentation

The documentation set for Oracle Data Mining is part of the Oracle Database 11*g* Release 2 (11.2) Online Documentation Library. The Oracle Data Mining documentation set consists of the following:

- *Oracle Data Mining Concepts*
- *Oracle Data Mining Administrator's Guide*
- *Oracle Data Mining Application Developer's Guide*

- *Oracle Data Mining Java API Reference* (javadoc)
- *Oracle Data Mining API Reference* (virtual book)
- *Oracle Database PL/SQL Packages and Types Reference*
  - `DBMS_DATA_MINING`
  - `DBMS_DATA_MINING_TRANSFORM`
  - `DBMS_PREDICTIVE_ANALYTICS`
- *Oracle Database SQL Language Reference*
  - Data Mining Functions

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in Oracle Data Mining?

This section describes new features in Oracle Data Mining. It includes the following sections:

- Oracle Database 11g Release 2 (11.2.0.3) Oracle Data Mining
- Oracle Database 11g Release 2 (11.2.0.2) Oracle Data Mining
- Oracle Database 11g Release 1 (11.1) Oracle Data Mining

Oracle Database 11*g* Release 2 (11.2.0.3) Oracle Data Mining

- The Oracle Data Mining Java API is deprecated in this release.

  > **Note:** Oracle recommends that you not use deprecated features in new applications. Support for deprecated features is for backward compatibility only

- Oracle Data Mining supports a new release of Oracle Data Miner. The earlier release, Oracle Data Miner Classic, is still available for download on OTN, but it is no longer under active development.

  To download Oracle Data Miner 11*g* Release 2 (11.2.0.3), go to:

  `http://www.oracle.com/technetwork/database/options/odm/dataminerworkflow-168677.html`

  To download Oracle Data Miner Classic, go to:

  `http://www.oracle.com/technetwork/database/options/odm/downloads/odminer-097463.html`

Oracle Database 11*g* Release 2 (11.2.0.2) Oracle Data Mining

In Oracle Data Mining 11*g* Release 2 (11.2.0.2), you can import externally-created GLM models when they are presented as valid PMML documents. PMML is an XML-based standard for representing data mining models.

The `IMPORT_MODEL` procedure in the `DBMS_DATA_MINING` package is overloaded with syntax that supports PMML import. When invoked with this syntax, the `IMPORT_MODEL` procedure will accept a PMML document and translate the information into an Oracle Data Mining model. This includes creating and populating model tables as well as `SYS` model metadata.

External models imported in this way will be automatically enabled for Exadata scoring offload.

See Also:

Oracle Database PL/SQL Packages and Types Reference for details about `DBMS_DATA_MINING.IMPORT_MODEL`

"Data Mining in Oracle Exadata" on page 2-2

Oracle Database 11*g* Release 1 (11.1) Oracle Data Mining

- **Mining Model schema objects**

  In Oracle 11*g*, Data Mining models are implemented as data dictionary objects in the `SYS` schema. A set of new data dictionary views present mining models and their properties. New system and object privileges control access to mining model objects.

  In previous releases, Data Mining models were implemented as a collection of tables and metadata within the `DMSYS` schema. In Oracle 11*g*, the `DMSYS` schema no longer exists.

  See Also:

  Oracle Data Mining Administrator's Guide for information on privileges for accessing mining models

  Oracle Data Mining Application Developer's Guide for information on Oracle Data Mining data dictionary views

- **Automatic Data Preparation (ADP)**

  In most cases, data must be transformed using techniques such as binning, normalization, or missing value treatment before it can be mined. Data for build, test, and apply must undergo the exact same transformations.

  In previous releases, data transformation was the responsibility of the user. In Oracle Database 11*g*, the data preparation process can be automated. Algorithm-appropriate transformation instructions are embedded in the model and automatically applied to the build data and scoring data. The automatic transformations can be complemented by or replaced with user-specified transformations.

  Because they contain the instructions for their own data preparation, mining models are known as **supermodels**.

  See Also:

  Chapter 19 for information on automatic and custom data transformation for Data Mining

  Oracle Database PL/SQL Packages and Types Reference for information on `DBMS_DATA_MINING_TRANSFORM`

- **Scoping of Nested Data and Enhanced Handling of Sparse Data**

  Oracle Data Mining supports nested data types for both categorical and numerical data. Multi-record case data must be transformed to nested columns for mining.

  In Oracle Data Mining 10gR2, nested columns were processed as top-level attributes; the user was burdened with the task of ensuring that two nested columns did not contain an attribute with the same name. In Oracle Data Mining

11g, nested attributes are scoped with the column name, which relieves the user of this burden.

Handling of sparse data and missing values has been standardized across algorithms in Oracle Data Mining 11g. Data is sparse when a high percentage of the cells are empty but all the values are assumed to be known. This is the case in market basket data. When some cells are empty, and their values are not known, they are assumed to be missing at random. Oracle Data Mining assumes that missing data in a nested column is a sparse representation, and missing data in a non-nested column is assumed to be missing at random.

In Oracle Data Mining 11*g*, Decision Tree and O-Cluster algorithms do not support nested data.

> **See Also:** *Oracle Data Mining Application Developer's Guide*

- **Generalized Linear Models**

  A new algorithm, Generalized Linear Models, is introduced in Oracle 11*g*. It supports two mining functions: classification (logistic regression) and regression (linear regression).

  > **See Also:** Chapter 12, "Generalized Linear Models"

- **New SQL Data Mining Function**

  A new SQL Data Mining function, `PREDICTION_BOUNDS`, has been introduced for use with Generalized Linear Models. `PREDICTION_BOUNDS` returns the confidence bounds on predicted values (regression models) or predicted probabilities (classification).

  > **See Also:** *Oracle Data Mining Application Developer's Guide*

- **Enhanced Support for Cost-Sensitive Decision Making**

  Cost matrix support is significantly enhanced in Oracle 11*g*. A cost matrix can be added or removed from any classification model using the new procedures, `DBMS_DATA_MINING.ADD_COST_MATRIX` and `DBMS_DATA_MINING.REMOVE_COST_MATRIX`.

  The SQL Data Mining functions support new syntax for specifying an in-line cost matrix. With this new feature, cost-sensitive model results can be returned within a SQL statement even if the model does not have an associated cost matrix for scoring.

  Only Decision Tree models can be built with a cost matrix.

  > **See Also:**
  >
  > *Oracle Data Mining Application Developer's Guide*
  >
  > "Biasing a Classification Model" on page 5-5

- **Features Not Available in 11*g*** Release 1 (11.1)

  – `DMSYS` schema

  – Oracle Data Mining Scoring Engine

  – In Oracle 10.2, you could use Database Configuration Assistant (DBCA) to configure the Data Mining option. In Oracle 11*g*, you do not need to use DBCA to configure the Data Mining option.

  - Basic Local Alignment Search Tool (BLAST)

- **Features Deprecated in 11*g*** Release 1 (11.1)

  - Adaptive Bayes Network classification algorithm (replaced with Decision Tree)

  - DM_USER_MODELS view and functions that provide information about models, model signature, and model settings (for example, GET_MODEL_SETTINGS, GET_DEFAULT_SETTINGS, and GET_MODEL_SIGNATURE) are replaced by data dictionary views. See *Oracle Data Mining Application Developer's Guide*.

Part I

Introductions

Part I presents an introduction to Oracle Data Mining and Oracle predictive analytics. The first chapter is a general, high-level overview for those who are new to these technologies.

Part I contains the following chapters:

- Chapter 1, "What Is Data Mining?"
- Chapter 2, "Introducing Oracle Data Mining"
- Chapter 3, "Introducing Oracle Predictive Analytics"

1 What Is Data Mining?

This chapter provides a high-level orientation to data mining technology.

Note: Information about data mining is widely available. No matter what your level of expertise, you will be able to find helpful books and articles on data mining. Here are two web sites to help you get started:

- `http://www.kdnuggets.com/` — This site is an excellent source of information about data mining. It includes a bibliography of publications.
- `http://www.twocrows.com/` — On this site, you will find the free tutorial, *Introduction to Data Mining and Knowledge Discovery*, and other useful information about data mining.

This chapter includes the following sections:

What Is Data Mining?

Data mining is the practice of automatically searching large stores of data to discover patterns and trends that go beyond simple analysis. Data mining uses sophisticated mathematical algorithms to segment the data and evaluate the probability of future events. Data mining is also known as Knowledge Discovery in Data (KDD).

The key properties of data mining are:

- Automatic discovery of patterns
- Prediction of likely outcomes
- Creation of actionable information
- Focus on large data sets and databases

Data mining can answer questions that cannot be addressed through simple query and reporting techniques.

Automatic Discovery

Data mining is accomplished by building models. A model uses an algorithm to act on a set of data. The notion of automatic discovery refers to the execution of data mining models.

Data mining models can be used to mine the data on which they are built, but most types of models are generalizable to new data. The process of applying a model to new data is known as **scoring**.

> **See Also:** *Oracle Data Mining Application Developer's Guide* for a discussion of scoring and deployment in Oracle Data Mining

Prediction

Many forms of data mining are predictive. For example, a model might predict income based on education and other demographic factors. Predictions have an associated probability (How likely is this prediction to be true?). Prediction probabilities are also known as **confidence** (How confident can I be of this prediction?).

Some forms of predictive data mining generate **rules**, which are conditions that imply a given outcome. For example, a rule might specify that a person who has a bachelor's degree and lives in a certain neighborhood is likely to have an income greater than the regional average. Rules have an associated **support** (What percentage of the population satisfies the rule?).

Grouping

Other forms of data mining identify natural groupings in the data. For example, a model might identify the segment of the population that has an income within a specified range, that has a good driving record, and that leases a new car on a yearly basis.

Actionable Information

Data mining can derive actionable information from large volumes of data. For example, a town planner might use a model that predicts income based on demographics to develop a plan for low-income housing. A car leasing agency might use a model that identifies customer segments to design a promotion targeting high-value customers.

> **See Also:** "Data Mining Functions" on page 2-2 for an overview of predictive and descriptive data mining. A general introduction to algorithms is provided in "Data Mining Algorithms" on page 2-5.

Data Mining and Statistics

There is a great deal of overlap between data mining and statistics. In fact most of the techniques used in data mining can be placed in a statistical framework. However, data mining techniques are not the same as traditional statistical techniques.

Traditional statistical methods, in general, require a great deal of user interaction in order to validate the correctness of a model. As a result, statistical methods can be difficult to automate. Moreover, statistical methods typically do not scale well to very large data sets. Statistical methods rely on testing hypotheses or finding correlations based on smaller, representative samples of a larger population.

Data mining methods are suitable for large data sets and can be more readily automated. In fact, data mining algorithms often require large data sets for the creation of quality models.

Data Mining and OLAP

On-Line Analytical Processing (OLAP) can been defined as fast analysis of shared multidimensional data. OLAP and data mining are different but complementary activities.

OLAP supports activities such as data summarization, cost allocation, time series analysis, and what-if analysis. However, most OLAP systems do not have inductive inference capabilities beyond the support for time-series forecast. Inductive inference, the process of reaching a general conclusion from specific examples, is a characteristic of data mining. Inductive inference is also known as computational learning.

OLAP systems provide a multidimensional view of the data, including full support for hierarchies. This view of the data is a natural way to analyze businesses and organizations. Data mining, on the other hand, usually does not have a concept of dimensions and hierarchies.

Data mining and OLAP can be integrated in a number of ways. For example, data mining can be used to select the dimensions for a cube, create new values for a dimension, or create new measures for a cube. OLAP can be used to analyze data mining results at different levels of granularity.

Data Mining can help you construct more interesting and useful cubes. For example, the results of predictive data mining could be added as custom measures to a cube. Such measures might provide information such as "likely to default" or "likely to buy" for each customer. OLAP processing could then aggregate and summarize the probabilities.

Data Mining and Data Warehousing

Data can be mined whether it is stored in flat files, spreadsheets, database tables, or some other storage format. The important criteria for the data is not the storage format, but its applicability to the problem to be solved.

Proper data cleansing and preparation are very important for data mining, and a data warehouse can facilitate these activities. However, a data warehouse will be of no use if it does not contain the data you need to solve your problem.

With one exception, Oracle Data Mining requires a single-record case data presentation. This means that the data for each record (case) must be specified within a single row. The exception is the data accepted by the Apriori algorithm for the calculation of association rules. The data for association rules may be presented as transactions (market-basket data), with the data for each case specified over multiple rows.

> **See Also:** *Oracle Data Mining Application Developer's Guide* for information about data requirements for Oracle Data Mining

What Can Data Mining Do and Not Do?

Data mining is a powerful tool that can help you find patterns and relationships within your data. But data mining does not work by itself. It does not eliminate the need to know your business, to understand your data, or to understand analytical methods. Data mining discovers hidden information in your data, but it cannot tell you the value of the information to your organization.

You might already be aware of important patterns as a result of working with your data over time. Data mining can confirm or qualify such empirical observations in addition to finding new patterns that may not be immediately discernible through simple observation.

It is important to remember that the predictive relationships discovered through data mining are not *causal* relationships. For example, data mining might determine that males with incomes between $50,000 and $65,000 who subscribe to certain magazines are likely to buy a given product. You can use this information to help you develop a marketing strategy. However, you should not assume that the population identified through data mining will buy the product *because* they belong to this population.

Asking the Right Questions

Data mining does not automatically discover information without guidance. The patterns you find through data mining will be very different depending on how you formulate the problem.

To obtain meaningful results, you must learn how to ask the right questions. For example, rather than trying to learn how to "improve the response to a direct mail solicitation," you might try to find the characteristics of people who have responded to your solicitations in the past.

Understanding Your Data

To ensure meaningful data mining results, you must understand your data. Data mining algorithms are often sensitive to specific characteristics of the data: outliers (data values that are very different from the typical values in your database), irrelevant columns, columns that vary together (such as age and date of birth), data coding, and data that you choose to include or exclude.

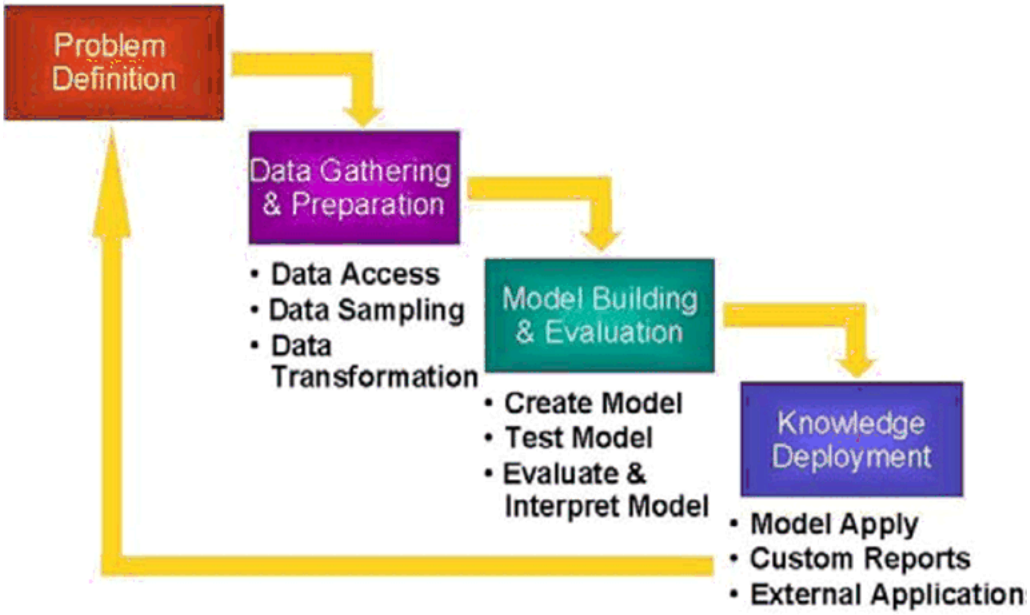
Oracle Data Mining can automatically perform much of the data preparation required by the algorithm. But some of the data preparation is typically specific to the domain or the data mining problem. At any rate, you need to understand the data that was used to build the model in order to properly interpret the results when the model is applied.

See Also: Chapter 19, "Automatic and Embedded Data Preparation" for information about data preparation in Oracle Data Mining

The Data Mining Process

Figure 1–1 illustrates the phases, and the iterative nature, of a data mining project. The process flow shows that a data mining project does not stop when a particular solution is deployed. The results of data mining trigger new business questions, which in turn can be used to develop more focused models.

Figure 1–1 The Data Mining Process



Problem Definition

This initial phase of a data mining project focuses on understanding the project objectives and requirements. Once you have specified the project from a business perspective, you can formulate it as a data mining problem and develop a preliminary implementation plan.

For example, your business problem might be: "How can I sell more of my product to customers?" You might translate this into a data mining problem such as: "Which customers are most likely to purchase the product?" A model that predicts who is most likely to purchase the product must be built on data that describes the customers who have purchased the product in the past. Before building the model, you must assemble the data that is likely to contain relationships between customers who have purchased the product and customers who have not purchased the product. Customer attributes might include age, number of children, years of residence, owners/renters, and so on.

Data Gathering and Preparation

The data understanding phase involves data collection and exploration. As you take a closer look at the data, you can determine how well it addresses the business problem. You might decide to remove some of the data or add additional data. This is also the time to identify data quality problems and to scan for patterns in the data.

The data preparation phase covers all the tasks involved in creating the case table you will use to build the model. Data preparation tasks are likely to be performed multiple times, and not in any prescribed order. Tasks include table, case, and attribute selection as well as data cleansing and transformation. For example, you might transform a `DATE_OF_BIRTH` column to `AGE`; you might insert the average income in cases where the `INCOME` column is null.

Additionally you might add new computed attributes in an effort to tease information closer to the surface of the data. For example, rather than using the purchase amount, you might create a new attribute: "Number of Times Amount Purchase Exceeds $500 in a 12 month time period." Customers who frequently make large purchases may also be related to customers who respond or don't respond to an offer.

Thoughtful data preparation can significantly improve the information that can be discovered through data mining.

Model Building and Evaluation

In this phase, you select and apply various modeling techniques and calibrate the parameters to optimal values. If the algorithm requires data transformations, you will need to step back to the previous phase to implement them (unless you are using Oracle Automatic Data Preparation, as described in Chapter 19).

In preliminary model building, it often makes sense to work with a reduced set of data (fewer rows in the case table), since the final case table might contain thousands or millions of cases.

At this stage of the project, it is time to evaluate how well the model satisfies the originally-stated business goal (phase 1). If the model is supposed to predict customers who are likely to purchase a product, does it sufficiently differentiate between the two classes? Is there sufficient lift? Are the trade-offs shown in the confusion matrix acceptable? Would the model be improved by adding text data? Should transactional data such as purchases (market-basket data) be included? Should costs associated with false positives or false negatives be incorporated into the model? (See Chapter 5 for information about classification test metrics and costs. See Chapter 8 for information about transactional data.)

Knowledge Deployment

Knowledge deployment is the use of data mining within a target environment. In the deployment phase, insight and actionable information can be derived from data.

Deployment can involve scoring (the application of models to new data), the extraction of model details (for example the rules of a decision tree), or the integration of data mining models within applications, data warehouse infrastructure, or query and reporting tools.

Because Oracle Data Mining builds and applies data mining models inside Oracle Database, the results are immediately available. BI reporting tools and dashboards can easily display the results of data mining. Additionally, Oracle Data Mining supports scoring in real time: Data can be mined and the results returned within a single database transaction. For example, a sales representative could run a model that predicts the likelihood of fraud within the context of an online sales transaction.

> **See Also:** *Oracle Data Mining Application Developer's Guide* for information about scoring and deployment in Oracle Data Mining

2 Introducing Oracle Data Mining

This chapter introduces the basics you will need to start using Oracle Data Mining.

This chapter includes the following sections:

Data Mining in the Database Kernel

Oracle Data Mining provides comprehensive, state-of-the-art data mining functionality within Oracle Database.

Oracle Data Mining is implemented in the Oracle Database kernel, and mining models are first class database objects. Oracle Data Mining processes use built-in features of Oracle Database to maximize scalability and make efficient use of system resources.

Data mining within Oracle Database offers many advantages:

- **No Data Movement**. Some data mining products require that the data be exported from a corporate database and converted to a specialized format for mining. With Oracle Data Mining, no data movement or conversion is needed. This makes the entire mining process less complex, time-consuming, and error-prone.

- **Security**. Your data is protected by the extensive security mechanisms of Oracle Database. Moreover, specific database privileges are needed for different data mining activities. Only users with the appropriate privileges can score (apply) mining models.

- **Data Preparation and Administration**. Most data must be cleansed, filtered, normalized, sampled, and transformed in various ways before it can be mined. Up to 80% of the effort in a data mining project is often devoted to data preparation. Oracle Data Mining can automatically manage key steps in the data preparation process. Additionally, Oracle Database provides extensive administrative tools for preparing and managing data.

- **Ease of Data Refresh**. Mining processes within Oracle Database have ready access to refreshed data. Oracle Data Mining can easily deliver mining results based on current data, thereby maximizing its timeliness and relevance.
- **Oracle Database Analytics**. Oracle Database offers many features for advanced analytics and business intelligence. Oracle Data Mining can easily be integrated with other analytical features of the database, such as statistical analysis and OLAP. See "Oracle Data Mining and Oracle Database Analytics" on page 2-13.
- **Oracle Technology Stack**. You can take advantage of all aspects of Oracle's technology stack to integrate data mining within a larger framework for business intelligence or scientific inquiry.
- **Domain Environment**. Data mining models have to be built, tested, validated, managed, and deployed in their appropriate application domain environments. Data mining results may need to be post-processed as part of domain specific computations (for example, calculating estimated risks and response probabilities) and then stored into permanent repositories or data warehouses. With Oracle Data Mining, the pre- and post-mining activities can all be accomplished within the same environment.
- **Application Programming Interfaces**. PL/SQL API and SQL language operators provide direct access to Oracle Data Mining functionality in Oracle Database.

Note: The Oracle Data Mining Java API is deprecated in this release.

Oracle recommends that you not use deprecated features in new applications. Support for deprecated features is for backward compatibility only

Data Mining in Oracle Exadata

Scoring refers to the process of applying a data mining model to data to generate predictions. The scoring process may require significant system resources. Vast amounts of data may be involved, and algorithmic processing may be very complex.

With Oracle Data Mining, scoring can be off-loaded to intelligent Oracle Exadata Storage Servers where processing is extremely performant.

Oracle Exadata Storage Servers combine Oracle's smart storage software and Oracle's industry-standard Sun hardware to deliver the industry's highest database storage performance. For more information about Oracle Exadata, visit the Oracle Technology Network at:

`http://www.oracle.com/us/products/database/exadata/index.htm`

Data Mining Functions

A basic understanding of data mining functions and algorithms is required for using Oracle Data Mining. This section introduces the concept of data mining functions. Algorithms are introduced in "Data Mining Algorithms" on page 2-5.

Each data mining **function** specifies a class of problems that can be modeled and solved. Data mining functions fall generally into two categories: **supervised** and **unsupervised**. Notions of supervised and unsupervised learning are derived from the science of machine learning, which has been called a sub-area of artificial intelligence.

Artificial intelligence refers to the implementation and study of systems that exhibit autonomous intelligence or behavior of their own. Machine learning deals with techniques that enable devices to learn from their own performance and modify their own functioning. Data mining applies machine learning concepts to data.

See Also: Part II, "Mining Functions" for more details about data mining functions

Supervised Data Mining

Supervised learning is also known as directed learning. The learning process is directed by a previously known dependent attribute or target. Directed data mining attempts to explain the behavior of the target as a function of a set of independent attributes or predictors.

Supervised learning generally results in **predictive** models. This is in contrast to unsupervised learning where the goal is pattern detection.

The building of a supervised model involves **training**, a process whereby the software analyzes many cases where the target value is already known. In the training process, the model "learns" the logic for making the prediction. For example, a model that seeks to identify the customers who are likely to respond to a promotion must be trained by analyzing the characteristics of many customers who are known to have responded or not responded to a promotion in the past.

Supervised Learning: Testing

Separate data sets are required for building (training) and testing some predictive models. The build data (training data) and test data must have the same column structure. Typically, one large table or view is split into two data sets: one for building the model, and the other for testing the model.

The process of applying the model to test data helps to determine whether the model, built on one chosen sample, is generalizable to other data. In particular, it helps to avoid the phenomenon of **overfitting**, which can occur when the logic of the model fits the build data too well and therefore has little predictive power.

Supervised Learning: Scoring

Apply data, also called scoring data, is the actual population to which a model is applied. For example, you might build a model that identifies the characteristics of customers who frequently buy a certain product. To obtain a list of customers who shop at a certain store and are likely to buy a related product, you might apply the model to the customer data for that store. In this case, the store customer data is the scoring data.

Most supervised learning can be applied to a population of interest. Scoring is the purpose of **classification** and **regression**, the principal supervised mining techniques.

Oracle Data Mining does not support the scoring operation for **attribute importance**, another supervised function. Models of this type are built on a population of interest to obtain information about that population; they cannot be applied to separate data. An attribute importance model returns and ranks the attributes that are most important in predicting a target value.

See Also: Table 2–1, " Oracle Data Mining Supervised Functions"

Unsupervised Data Mining

Unsupervised learning is non-directed. There is no distinction between dependent and independent attributes. There is no previously-known result to guide the algorithm in building the model.

Unsupervised learning can be used for **descriptive** purposes. It can also be used to make predictions.

Unsupervised Learning: Scoring

Although unsupervised data mining does not specify a target, most unsupervised learning can be applied to a population of interest. For example, clustering models use descriptive data mining techniques, but they can be applied to classify cases according to their cluster assignments. **Anomaly detection**, although unsupervised, is typically used to predict whether a data point is typical among a set of cases.

Oracle Data Mining supports the scoring operation for **clustering** and **feature extraction**, both unsupervised mining functions. Oracle Data Mining does not support the scoring operation for **association rules**, another unsupervised function. Association models are built on a population of interest to obtain information about that population; they cannot be applied to separate data. An association model returns rules that explain how items or events are associated with each other. The association rules are returned with statistics that can be used to rank them according to their probability.

See Also: Table 2–2, " Oracle Data Mining Unsupervised Functions"

Oracle Data Mining Functions

Oracle Data Mining supports the supervised data mining functions described in Table 2–1.

Table 2–1 Oracle Data Mining Supervised Functions

Function	Description	Sample Problem
Attribute Importance	Identifies the attributes that are most important in predicting a target attribute	Given customer response to an affinity card program, find the most significant predictors
Classification	Assigns items to discrete classes and predicts the class to which an item belongs	Given demographic data about a set of customers, predict customer response to an affinity card program
Regression	Approximates and forecasts continuous values	Given demographic and purchasing data about a set of customers, predict customers' age

Oracle Data Mining supports the unsupervised functions described in Table 2–2.

Table 2–2 Oracle Data Mining Unsupervised Functions

Function	Description	Sample Problem
Anomaly Detection (implemented through one-class classification)	Identifies items (outliers) that do not satisfy the characteristics of "normal" data	Given demographic data about a set of customers, identify customer purchasing behavior that is significantly different from the norm
Association Rules	Finds items that tend to co-occur in the data and specifies the rules that govern their co-occurrence	Find the items that tend to be purchased together and specify their relationship
Clustering	Finds natural groupings in the data	Segment demographic data into clusters and rank the probability that an individual will belong to a given cluster
Feature Extraction	Creates new attributes (features) using linear combinations of the original attribute	Given demographic data about a set of customers, group the attributes into general characteristics of the customers

See Also: Part II for details about the mining functions supported by Oracle Data Mining

Data Mining Algorithms

An algorithm is a mathematical procedure for solving a specific kind of problem. Oracle Data Mining supports at least one algorithm for each data mining function. For some functions, you can choose among several algorithms. For example, Oracle Data Mining supports four classification algorithms.

Each data mining model is produced by a specific algorithm. Some data mining problems can best be solved by using more than one algorithm. This necessitates the development of more than one model. For example, you might first use a feature extraction model to create an optimized set of predictors, then a classification model to make a prediction on the results.

Note: You can be successful at data mining without understanding the inner workings of each algorithm. However, it is important to understand the general characteristics of the algorithms and their suitability for different kinds of applications.

Oracle Data Mining Supervised Algorithms

Oracle Data Mining supports the supervised data mining algorithms described in Table 2–3. The algorithm abbreviations are used throughout this manual.

Table 2–3 Oracle Data Mining Algorithms for Supervised Functions

Algorithm	Function	Description
Decision Tree (DT)	Classification	Decision trees extract predictive information in the form of human-understandable rules. The rules are if-then-else expressions; they explain the decisions that lead to the prediction.
Generalized Linear Models (GLM)	Classification and Regression	GLM implements logistic regression for classification of binary targets and linear regression for continuous targets. GLM classification supports confidence bounds for prediction probabilities. GLM regression supports confidence bounds for predictions.

Table 2–3 (Cont.) Oracle Data Mining Algorithms for Supervised Functions

Algorithm	Function	Description
Minimum Description Length (MDL)	Attribute Importance	MDL is an information theoretic model selection principle. MDL assumes that the simplest, most compact representation of data is the best and most probable explanation of the data.
Naive Bayes (NB)	Classification	Naive Bayes makes predictions using Bayes' Theorem, which derives the probability of a prediction from the underlying evidence, as observed in the data.
Support Vector Machine (SVM)	Classification and Regression	Distinct versions of SVM use different kernel functions to handle different types of data sets. Linear and Gaussian (nonlinear) kernels are supported. SVM classification attempts to separate the target classes with the widest possible margin. SVM regression tries to find a continuous function such that the maximum number of data points lie within an epsilon-wide tube around it.

Oracle Data Mining Unsupervised Algorithms

Oracle Data Mining supports the unsupervised data mining algorithms described in Table 2–4. The algorithm abbreviations are used throughout this manual.

Table 2–4 Oracle Data Mining Algorithms for Unsupervised Functions

Algorithm	Function	Description
Apriori (AP)	Association	Apriori performs market basket analysis by discovering co-occurring items (frequent itemsets) within a set. Apriori finds rules with support greater than a specified minimum support and confidence greater than a specified minimum confidence.
k-Means (KM)	Clustering	*k*-Means is a distance-based clustering algorithm that partitions the data into a predetermined number of clusters. Each cluster has a centroid (center of gravity). Cases (individuals within the population) that are in a cluster are close to the centroid. Oracle Data Mining supports an enhanced version of *k*-Means. It goes beyond the classical implementation by defining a hierarchical parent-child relationship of clusters.
Non-Negative Matrix Factorization (NMF)	Feature Extraction	NMF generates new attributes using linear combinations of the original attributes. The coefficients of the linear combinations are non-negative. During model apply, an NMF model maps the original data into the new set of attributes (features) discovered by the model.
One Class Support Vector Machine (One-Class SVM)	Anomaly Detection	One-class SVM builds a profile of one class and when applied, flags cases that are somehow different from that profile. This allows for the detection of rare cases that are not necessarily related to each other.
Orthogonal Partitioning Clustering (O-Cluster or OC)	Clustering	O-Cluster creates a hierarchical, grid-based clustering model. The algorithm creates clusters that define dense areas in the attribute space. A sensitivity parameter defines the baseline density level.

See Also: Part III for details about the algorithms supported by Oracle Data Mining

Data Preparation

Data for mining must exist within a single table or view. The information for each case (record) must be stored in a separate row.

A unique capability of Oracle Data Mining is its support for dimensioned data (for example, star schemas) through nested table transformations. Additionally, Oracle Data Mining can mine unstructured data.

See Also:

Oracle Data Mining Application Developer's Guide to learn how to construct a table or view for data mining

Chapter 20, "Text Mining" for information about mining unstructured data

Proper preparation of the data is a key factor in any data mining project. The data must be properly cleansed to eliminate inconsistencies and support the needs of the mining application. Additionally, most algorithms require some form of data transformation, such as binning or normalization.

The data mining development process may require several data sets. One data set may needed for building (training) the model; a separate data set may be used for scoring. Classification models should also have a test data set. Each of these data sets must be prepared in exactly the same way.

Automatic Data Preparation

Oracle Data Mining supports automatic and embedded data transformation, which can significantly reduce the time and effort involved in developing a data mining model. In **Automatic Data Preparation (ADP)** mode, the model itself transforms the build data according to the requirements of the algorithm. The transformation instructions are embedded in the model and reused whenever the model is applied.

You can choose to add your own transformations to those performed automatically by Oracle Data Mining. These are embedded along with the automatic transformation instructions and reused with them whenever the model is applied. In this case, you only have to specify your transformations once — for the build data. The model itself will transform the data appropriately when it is applied.

Mining models are known as **supermodels**, because they contain the instructions for their own data preparation.

See Also: Chapter 19, "Automatic and Embedded Data Preparation"

Model Transparency

Oracle Data Mining provides a high degree of model transparency. All models support a set of **model details** that you can query to observe the effect of the algorithm on the training data. Additionally, some algorithms produce **rules**, which express the logic used by the model. Clustering algorithms, as well as Decision Tree and Association Rules, all produce rules.

The data used internally by the model often does not look like the data used to build the model. This is because the model transforms much of the data prior to algorithmic processing. Model details reverse these transformations and present the training data as it is understood by a user. Similarly, any transformations applied to the target of a supervised model are reversed before the results are returned to the user.

See Also:

Oracle Data Mining Application Developer's Guide for a summary of information returned in model details

Chapters on O-Cluster, k-Means, Decision Tree, and Apriori in Part III, "Algorithms"

How Do I Use Oracle Data Mining?

Oracle Data Mining supports programmatic interfaces for PL/SQL, SQL, and R.

Note: The Oracle Data Mining Java API is deprecated in this release.

Certain models that are PMML-compliant can be imported into Oracle Database for scoring.

The Oracle Data Miner graphical user interface provides access to the full range of in-database capabilities of Oracle Data Mining. A spreadsheet add-in provides access to predictive analytics that are powered by Oracle Data Mining.

Oracle Data Miner

Oracle Data Miner is the graphical user interface to Oracle Data Mining. Oracle Data Miner, an extension to Oracle SQL Developer 3.0, uses a work flow paradigm to perform data mining tasks.

You can use Oracle Data Miner to explore data, build and evaluate multiple mining models, and apply the models to new data. By building work flows, you can capture and document the methodology you use to perform a range of mining tasks. You can save and share work flows.

See Also: *Oracle Data Mining Administrator's Guide*

Note: To download Oracle Data Miner 11*g* Release 2 (11.2.0.2), go to:

`http://www.oracle.com/technetwork/database/options/odm/dataminerworkflow-168677.html`

The previous release of Oracle Data Miner is no longer under active development, but it is still available for download on the Oracle Technology Network. To download the earlier release, go to:

`http://www.oracle.com/technetwork/database/options/odm/downloads/odminer-097463.html`

PL/SQL Packages

The Oracle Data Mining PL/SQL API is implemented in the following PL/SQL packages:

- `DBMS_DATA_MINING` — Contains routines for building, testing, and applying data mining models.
- `DBMS_DATA_MINING_TRANSFORM` — Contains routines for transforming the data sets prior to building or applying a model. You can use these routines, adaptations of these routines, or any other SQL-based method for implementing transformations.

  Note that user-defined transformations may not be required. With ADP, Oracle Data Mining can automatically perform most transformations required by a given algorithm.
- `DBMS_PREDICTIVE_ANALYTICS` — Contains automated data mining routines for `PREDICT`, `EXPLAIN`, and `PROFILE` operations.

The following example shows the PL/SQL routine for creating an SVM classification model called `my_model`. The algorithm is specified in a settings table called `my_settings`. The algorithm must be specified as a setting because Naive Bayes, not SVM, is the default classifier.

```
CREATE TABLE my_settings(
  setting_name  VARCHAR2(30),
  setting_value VARCHAR2(4000));

BEGIN
    INSERT INTO my_settings VALUES
        (dbms_data_mining.algo_name,
         dbms_data_mining.algo_support_vector_machines);
    COMMIT;
END;
/

BEGIN
  DBMS_DATA_MINING.CREATE_MODEL(
    model_name          => 'my_model',
    mining_function     => dbms_data_mining.classification,
    data_table_name     => 'mining_data_build',
    case_id_column_name => 'cust_id',
    target_column_name  => 'affinity_card',
    settings_table_name => 'my_settings');
END;
/
```

See Also: *Oracle Database PL/SQL Packages and Types Reference*

SQL Scoring Functions

A family of SQL language operators supports the deployment of data mining models. These operators allow scoring to be easily incorporated into SQL queries, and thus into SQL-based applications.

The following example illustrates the Data Mining `PREDICTION_PROBABILITY` operator. The operator applies the classification model `nb_sh_clas_sample` to the data set `mining_data_apply_v`.

```
SELECT cust_id, prob
  FROM (SELECT cust_id,
          PREDICTION_PROBABILITY (nb_sh_clas_sample, 1 USING *) prob
          FROM  mining_data_apply_v
          WHERE cust_id < 100011)
  ORDER BY cust_id;
```

The `SELECT` statement returns ten customers, listed by customer ID, along with the likelihood that they will accept (1) an affinity card.

```
   CUST_ID       PROB
---------- ----------
    100001 .025622714
    100002 .090424232
    100003 .028064789
    100004 .048458859
    100005 .989335775
    100006 .000151844
    100007  .05749942
    100008 .108750373
    100009 .538512886
```

```
100010 .186426058
```

See Also: *Oracle Database SQL Language Reference*

R-ODM

The R Interface to Oracle Data Mining (R-ODM) provides access to Oracle Data Mining from the R programming environment. R-ODM consists of a set of function wrappers written in source R language. These functions pass data and parameters from the R environment to Oracle Database through an ODBC connection.

The R-ODM interface is completely external to Oracle Database. It does not use or expose any Oracle product code. R-ODM is packaged as a standard R source package and is distributed freely as part of the Comprehensive R Archive Network (CRAN).

See Also:

http://www.oracle.com/technetwork/database/options/odm/odm-r-integration-089013.htmlfor more information about R-ODM

http://www.r-project.org for information about the R environment, R packages, and CRAN

PMML Import

Using the PL/SQL API, you can import a GLM model represented in Predictive Model Markup Language (PMML) into an Oracle database.This functionality is available starting with Oracle Database 11g Release 2 (11.2.0.2) Data Mining.

PMML is an XML-based standard specified by the Data Mining Group (http://www.dmg.org). Applications that are PMML-compliant can deploy PMML-compliant models that were created by any vendor. Oracle Data Mining supports the core features of PMML 3.1 for regression models.

See Also:

Oracle Data Mining Administrator's Guide for information about exporting and importing data mining models

http://www.dmg.org/faq.html for information about PMML

Oracle Spreadsheet Add-In for Predictive Analytics

Predictive Analytics automates the data mining process with routines for PREDICT, EXPLAIN, and PROFILE. The Oracle Spreadsheet Add-In for Predictive Analytics implements these routines for Microsoft Excel.You can use the Spreadsheet Add-In to analyze Excel data or data that resides in an Oracle database.

You can download the Spreadsheet Add-In, including a readme file, from the Oracle Technology Network.

http://www.oracle.com/technetwork/database/options/odm/index.html

See Also: Chapter 3, "Introducing Oracle Predictive Analytics"

Java API

The Oracle Data Mining Java API is an Oracle implementation of the JDM standard Java API for data mining (JSR-73). The Java API is layered on the PL/SQL API, and the two APIs are fully interoperable.

> **Note:** The Oracle Data Mining Java API is deprecated in this release.
>
> Oracle recommends that you not use deprecated features in new applications. Support for deprecated features is for backward compatibility only

The following code fragment creates a Decision Tree model that models customer affinity card response patterns and applies this model to predict new customers' affinity card responses.

```
//Step-1: Create connection to a database with the Data Mining option
 OraConnectionFactory m_dmeConnFactory = new OraConnectionFactory();
 ConnectionSpec connSpec = m_dmeConnFactory.getConnectionSpec();
 connSpec.setURI("jdbc:oracle:thin:@<hostName>:<port>:<sid>");
 connSpec.setName("<user name>");
 connSpec.setPassword("password");
 m_dmeConn = m_dmeConnFactory.getConnection(connSpec);

//Step-2: Create object factories
 PhysicalDataSetFactory m_pdsFactory =
   (PhysicalDataSetFactory)m_dmeConn.getFactory(
      "javax.datamining.data.PhysicalDataSet");
 PhysicalAttributeFactory m_paFactory =
   (PhysicalAttributeFactory)m_dmeConn.getFactory(
      "javax.datamining.data.PhysicalAttribute");
 TreeSettingsFactory m_treeFactory =
   (TreeSettingsFactory)m_dmeConn.getFactory(
      "javax.datamining.algorithm.tree.TreeSettings");
 ClassificationSettingsFactory m_clasFactory =
   (ClassificationSettingsFactory)m_dmeConn.getFactory(
      "javax.datamining.supervised.classification.ClassificationSettings");
BuildTaskFactory m_buildFactory =
   (BuildTaskFactory)m_dmeConn.getFactory(
      "javax.datamining.task.BuildTask");
ClassificationApplySettingsFactory m_applySettingsFactory =
   (ClassificationApplySettingsFactory)m_dmeConn.getFactory(
     "javax.datamining.supervised.classification.ClassificationApplySettings");
DataSetApplyTaskFactory m_dsApplyFactory =
   (DataSetApplyTaskFactory)m_dmeConn.getFactory(
      "javax.datamining.task.apply.DataSetApplyTask");
ClassificationApplySettingsFactory m_applySettingsFactory =
   (ClassificationApplySettingsFactory)m_dmeConn.getFactory(
     "javax.datamining.supervised.classification.ClassificationApplySettings");

//Step-3: Create and save model build task input objects
//        (training data, build settings)
    //Create & save model input data specification (PhysicalDataSet)
    PhysicalDataSet buildData =
      m_pdsFactory.create("MINING_DATA_BUILD_V", false);
    PhysicalAttribute pa =
      m_paFactory.create("CUST_ID", AttributeDataType.integerType,
                         PhysicalAttributeRole.caseId);
    buildData.addAttribute(pa);
    m_dmeConn.saveObject("treeBuildData_jdm", buildData, true);
    //Create & save Mining Function Settings
    ClassificationSettings buildSettings = m_clasFactory.create();
    TreeSettings treeAlgo = m_treeFactory.create();
    buildSettings.setAlgorithmSettings(treeAlgo);
    buildSettings.setTargetAttributeName("AFFINITY_CARD");
    m_dmeConn.saveObject("treeBuildSettings_jdm", buildSettings, true);

//Step-4: Create and save model build task
 BuildTask buildTask =
    m_buildFactory.create("treeBuildData_jdm", "treeBuildSettings_jdm",
                            "treeModel_jdm");
 m_dmeConn.saveObject("treeBuildTask_jdm", buildTask, true);

//Step-5: Create and save model apply task input objects (apply settings)
  //Create & save PhysicalDataSpecification
    PhysicalDataSet applyData =
      m_pdsFactory.create("MINING_DATA_APPLY_V", false);
    PhysicalAttribute pa =
      m_paFactory.create("CUST_ID", AttributeDataType.integerType,
                         PhysicalAttributeRole.caseId);
    applyData.addAttribute(pa);
    m_dmeConn.saveObject("treeApplyData_jdm", applyData, true);
    //Create & save ClassificationApplySettings
    ClassificationApplySettings clasAS = m_applySettingsFactory.create();

//Step-6: Create and save model apply task with build task as dependent
    DataSetApplyTask applyTask =
      m_dsApplyFactory.create("treeApplyData_jdm", "treeModel_jdm",
                              "treeApplySettings_jdm",
                              "TREE_APPLY_OUTPUT_JDM");
    ((OraTask)applyTask).addDependency("treeBuildTask_jdm");
    m_dmeConn.saveObject("treeApplyTask_jdm", applyTask, true);

//Step-7: Execute build task which executes build task and then after
//        successful completion triggers the execution of its dependent
//        task(s). In this example, there is only one dependent task.
m_dmeConn.execute("treeBuildTask_jdm");
```

See Also: *Oracle Data Mining Application Developer's Guide*

Where Do I Find Information About Oracle Data Mining?

Oracle Data Mining documentation is included in the Oracle Database Online Documentation Library.

For your convenience, the Oracle Data Mining and related Oracle Database manuals are listed in Table 2–5.

Table 2–5 Oracle Data Mining Documentation

Document	Description
Oracle Data Mining Concepts	This manual. An overview of mining functions, algorithms, data preparation, predictive analytics, and other special features supported by Oracle Data Mining
Oracle Data Mining Application Developer's Guide	How to use the programmatic interfaces to Oracle Data Mining.
Oracle Data Mining Administrator's Guide	How to install and administer a database for Data Mining. How to install and use the demo programs
Oracle Database PL/SQL Packages and Types Reference	How to use the Data Mining PL/SQL API syntax
Oracle Database SQL Language Reference	How to use the Data Mining SQL function syntax
Oracle Database Reference	How to query data dictionary views to obtain information about mining models, mining model attributes, and mining model settings

See Also: Oracle Database online documentation library at `http://www.oracle.com/pls/db112/homepage`

Oracle Data Mining Resources on the Oracle Technology Network

The Oracle Technology Network (OTN) is easily accessible and provides a wealth of information. You can visit the Oracle Data Mining home page at:

`http://www.oracle.com/technetwork/database/options/odm/index.html`

This site provides news and discussion forums as well as tools and educational materials for download. On this site, you will find:

- **Oracle Data Miner** (download)
- **Oracle Spreadsheet Add-In for Predictive Analytics** (download)
- **Sample code** (download)
- **R-ODM information**
- **White papers and web casts**
- **Oracle Data Mining discussion forum**
- **Blogs on data mining and analytics in the Oracle Database**

Oracle Data Mining and Oracle Database Analytics

As described in "Data Mining in the Database Kernel" on page 2-1, in-database analytics offer significant advantages. When analytical capabilities are implemented where the data is stored, the data does not have to be exported to an external server for analysis. The results of analysis do not need to be imported; they reside in the database where they can be easily accessed, refreshed, and combined with other data.

Along with data mining and predictive analytics, Oracle Database supports a wide array of analytical features. Since these features are part of a common server it is possible to combine them efficiently. The results of analytical processing can be integrated with Oracle Business Intelligence Suite Enterprise Edition and other BI tools and applications. Taken as a whole, these features make the Oracle Database a powerful platform for developing analytical applications.

The possibilities for combining different analytics are virtually limitless. Example 2–1 shows data mining and text processing within a single SQL query. The query selects all customers who have a high propensity to attrite (> 80% chance), are valuable customers (customer value rating > 90), and have had a recent conversation with customer services regarding a Checking Plus account. The propensity to attrite information is computed using a Data Mining model called `tree_model`. The query uses the Oracle Text `CONTAINS` operator to search call center notes for references to Checking Plus accounts.

Example 2–1 Combine Oracle Data Mining and Oracle Text in a SQL Query

```
SELECT A.cust_name, A.contact_info
  FROM customers A
 WHERE PREDICTION_PROBABILITY(tree_model,
            'attrite' USING A.*) > 0.8
   AND A.cust_value > 90
   AND A.cust_id IN
       (SELECT B.cust_id
          FROM call_center B
         WHERE B.call_date BETWEEN '01-Jan-2005'
                               AND '30-Jun-2005'
         AND CONTAINS(B.notes, 'Checking Plus', 1) > 0);
```

Some of the analytics supported by Oracle Database are described in Table 2–6. Use the links in the Documentation column to find related documentation.

Table 2–6 Overview of Analytics in Oracle Database

Analytical Feature	Description	Documentation
Data Mining	Oracle Data Mining implements complex algorithms that sift through large volumes of data to find hidden information. Data Mining models discover patterns, predict probable outcomes, identify key predictors, and find other kinds of valuable information	*Oracle Data Mining Concepts* (this manual)
Complex data transformations	Data transformation is a key aspect of analytical applications and ETL (extract, transform, and load). You can use SQL expressions to implement data transformations, or you can use the `DBMS_DATA_MINING_TRANSFORM` package. `DBMS_DATA_MINING_TRANSFORM` is a flexible data transformation package that includes a variety of missing value and outlier treatments, as well as binning and normalization capabilities.	*Oracle Database PL/SQL Packages and Types Reference*
Statistical functions	Oracle Database provides a long list of SQL statistical functions with support for: hypothesis testing (such as t-test, F-test), correlation computation (such as pearson correlation), cross-tab statistics, and descriptive statistics (such as median and mode). The `DBMS_STAT_FUNCS` package adds distribution fitting procedures and a summary procedure that returns descriptive statistics for a column.	*Oracle Database SQL Language Reference* and *Oracle Database PL/SQL Packages and Types Reference*
Window and analytic SQL functions	Oracle Database supports analytic and windowing functions for computing cumulative, moving, and centered aggregates. With windowing aggregate functions, you can calculate moving and cumulative versions of `SUM`, `AVERAGE`, `COUNT`, `MAX`, `MIN`, and many more functions.	*Oracle Database Data Warehousing Guide*
Frequent Itemsets	The `DBMS_FREQUENT_ITEMSET` supports frequent itemset counting, a mechanism for counting how often multiple events occur together. `DBMS_FREQUENT_ITEMSET` is used as a building block for the Association Rules algorithm used by Oracle Data Mining.	*Oracle Database PL/SQL Packages and Types Reference*
Linear algebra	The `UTL_NLA` package exposes a subset of the popular `BLAS` and `LAPACK` (Version 3.0) libraries for operations on vectors and matrices represented as VARRAYs. This package includes procedures to solve systems of linear equations, invert matrices, and compute eigenvalues and eigenvectors.	*Oracle Database PL/SQL Packages and Types Reference*

Table 2–6 (Cont.) Overview of Analytics in Oracle Database

Analytical Feature	Description	Documentation
OLAP	Oracle OLAP supports multidimensional analysis and can be used to improve performance of multidimensional queries. Oracle OLAP provides functionality previously found only in specialized OLAP databases. Moving beyond drill-downs and roll-ups, Oracle OLAP also supports time-series analysis, modeling, and forecasting.	*Oracle OLAP User's Guide*
Spatial analytics	Oracle Spatial provides advanced spatial features to support high-end GIS and LBS solutions. Oracle Spatial's analysis and mining capabilities include functions for binning, detection of regional patterns, spatial correlation, colocation mining, and spatial clustering. Oracle Spatial also includes support for topology and network data models and analytics. The topology data model of Oracle Spatial allows one to work with data about nodes, edges, and faces in a topology. It includes network analysis functions for computing shortest path, minimum cost spanning tree, nearest-neighbors analysis, traveling salesman problem, among others.	*Oracle Spatial Developer's Guide*
Text Mining	Oracle Text uses standard SQL to index, search, and analyze text and documents stored in the Oracle database, in files, and on the web. It also supports automatic classification and clustering of document collections. Many of these analytical features are layered on top of ODM functionality	*Oracle Text Application Developer's Guide*

3

Introducing Oracle Predictive Analytics

This chapter presents an overview of Oracle Data Mining predictive analytics, an automated form of predictive data mining.

> **See Also:**
>
> *Oracle Data Mining Administrator's Guide* for installation instructions
>
> *Oracle Database PL/SQL Packages and Types Reference* for predictive analytics syntax in PL/SQL

This chapter includes the following sections:

About Predictive Analytics

Predictive Analytics is a technology that captures data mining processes in simple routines. Sometimes called "one-click data mining," predictive analytics simplifies and automates the data mining process.

Predictive analytics develops profiles, discovers the factors that lead to certain outcomes, predicts the most likely outcomes, and identifies a degree of confidence in the predictions.

Predictive Analytics and Data Mining

Predictive analytics uses data mining technology, but knowledge of data mining is not needed to use predictive analytics.

You can use predictive analytics simply by specifying an operation to perform on your data. You do not need to create or use mining models or understand the mining functions and algorithms summarized in Chapter 2 of this manual.

How Does it Work?

The predictive analytics routines analyze the input data and create mining models. These models are trained and tested and then used to generate the results returned to the user. The models and supporting objects are not preserved after the operation completes.

When you use data mining technology directly, you create a model or use a model created by someone else. Usually, you apply the model to new data (different from the data used to train and test the model). Predictive analytics routines apply the model to the same data used for training and testing.

> **See Also:** "Behind the Scenes" on page 3-6 to gain insight into the inner workings of Oracle predictive analytic

Predictive Analytics Operations

Oracle Data Mining predictive analytics operations are described in Table 3–1.

Table 3–1 Oracle Predictive Analytics Operations

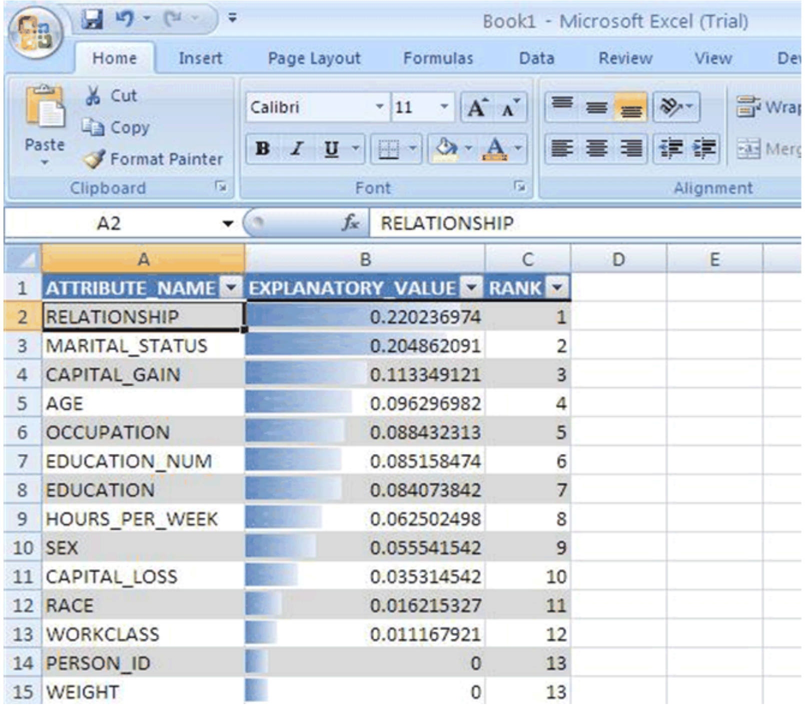
Operation	Description
`EXPLAIN`	Explains how the individual attributes affect the variation of values in a target column
`PREDICT`	For each case, predicts the values in a target column
`PROFILE`	Creates a set of rules for cases that imply the same target value

Oracle Spreadsheet Add-In for Predictive Analytics

The Oracle Spreadsheet Add-In for Predictive Analytics provides predictive analytics operations within a Microsoft Excel spreadsheet. You can analyze Excel data or data that resides in an Oracle database.

Figure 3–1 shows the `EXPLAIN` operation using Microsoft Excel 7.0. `EXPLAIN` shows the predictors of a given target ranked in descending order of importance. In this example, `RELATIONSHIP` is the most important predictor, and `MARTIAL STATUS` is the second most important predictor .

Figure 3–1 EXPLAIN in Oracle Spreadsheet Add-In for Predictive Analytics

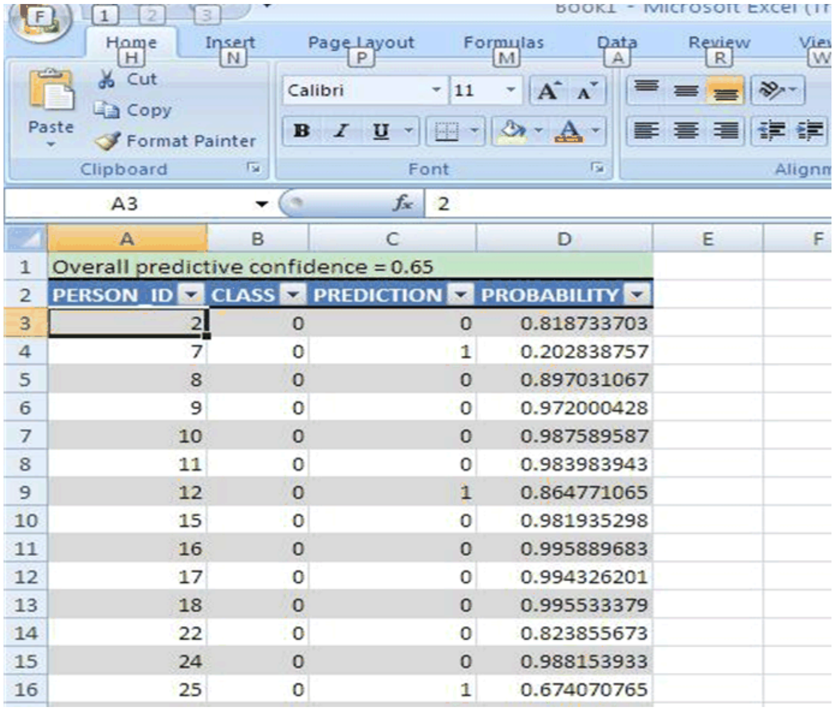


A2 RELATIONSHIP

	A	B	C
1	ATTRIBUTE_NAME	EXPLANATORY_VALUE	RANK
2	RELATIONSHIP	0.220236974	1
3	MARITAL_STATUS	0.204862091	2
4	CAPITAL_GAIN	0.113349121	3
5	AGE	0.096296982	4
6	OCCUPATION	0.088432313	5
7	EDUCATION_NUM	0.085158474	6
8	EDUCATION	0.084073842	7
9	HOURS_PER_WEEK	0.062502498	8
10	SEX	0.055541542	9
11	CAPITAL_LOSS	0.035314542	10
12	RACE	0.016215327	11
13	WORKCLASS	0.011167921	12
14	PERSON_ID	0	13
15	WEIGHT	0	13

Figure 3–2 shows the `PREDICT` operation for a binary target. `PREDICT` shows the actual and predicted classification for each case. It includes the probability of each prediction and the overall predictive confidence for the data set.

Figure 3–2 PREDICT in Oracle Spreadsheet Add-In for Predictive Analytics



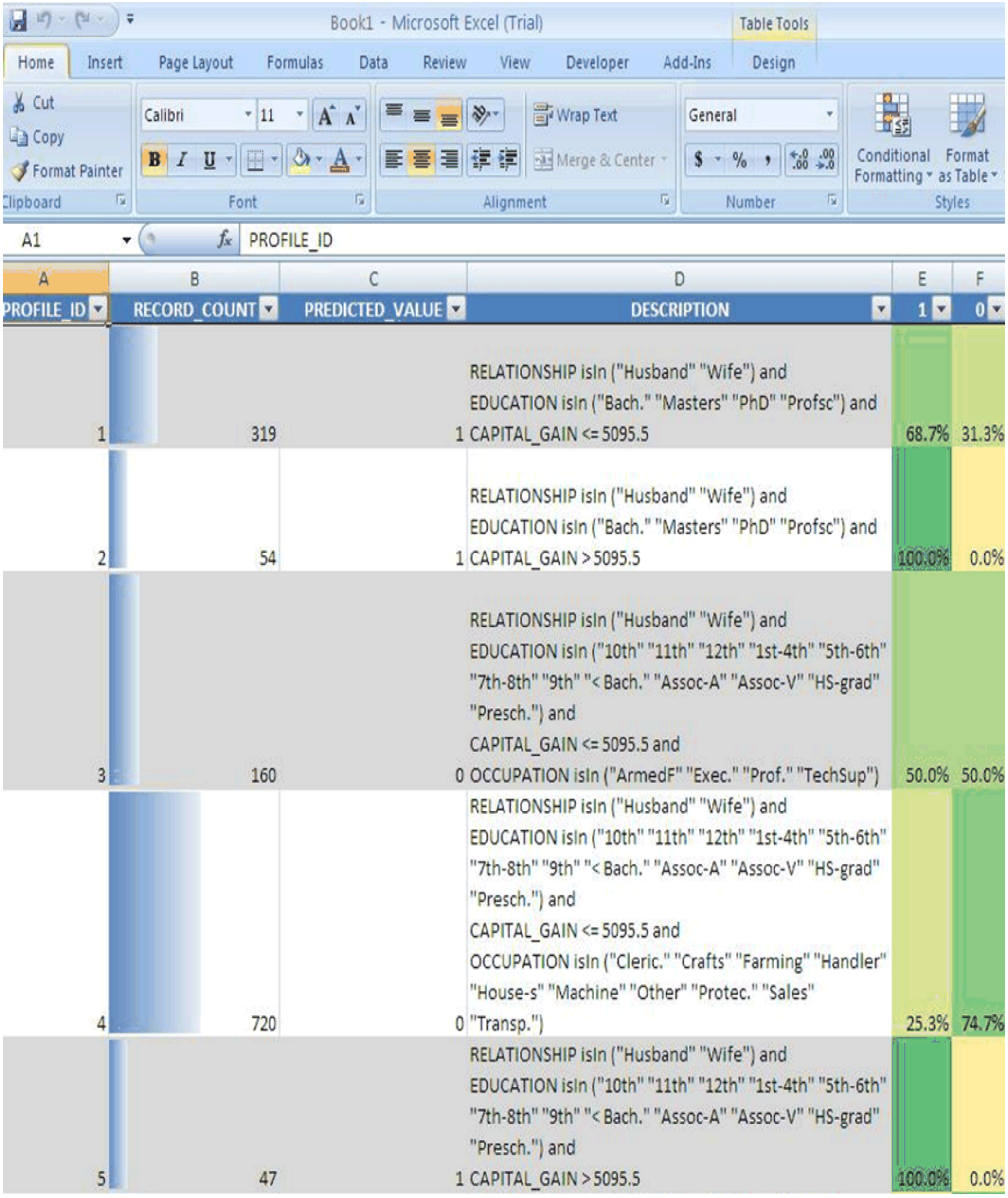
A3 2

	A	B	C	D
1	Overall predictive confidence = 0.65			
2	PERSON_ID	CLASS	PREDICTION	PROBABILITY
3	2	0	0	0.818733703
4	7	0	1	0.202838757
5	8	0	0	0.897031067
6	9	0	0	0.972000428
7	10	0	0	0.987589587
8	11	0	0	0.983983943
9	12	0	1	0.864771065
10	15	0	0	0.981935298
11	16	0	0	0.995889683
12	17	0	0	0.994326201
13	18	0	0	0.995533379
14	22	0	0	0.823855673
15	24	0	0	0.988153933
16	25	0	1	0.674070765

Figure 3–3 shows the `PROFILE` operation. This example shows five profiles for a binary classification problem. Each profile includes a rule, the number of cases to

which it applies, and a score distribution. Profile 1 describes 319 cases. Its members are husbands or wives with bachelors, masters, Ph.D., or professional degrees; they have capital gains <= 5095.5. The probability of a positive prediction for this group is 68.7%; the probability of a negative prediction is 31.3%.

Figure 3–3 PROFILE in Oracle Spreadsheet Add-In for Predictive Analytics



PROFILE_ID	RECORD_COUNT	PREDICTED_VALUE	DESCRIPTION	1	0
1	319	1	RELATIONSHIP isIn ("Husband" "Wife") and EDUCATION isIn ("Bach." "Masters" "PhD" "Profsc") and CAPITAL_GAIN <= 5095.5	68.7%	31.3%
2	54	1	RELATIONSHIP isIn ("Husband" "Wife") and EDUCATION isIn ("Bach." "Masters" "PhD" "Profsc") and CAPITAL_GAIN > 5095.5	100.0%	0.0%
3	160	0	RELATIONSHIP isIn ("Husband" "Wife") and EDUCATION isIn ("10th" "11th" "12th" "1st-4th" "5th-6th" "7th-8th" "9th" "< Bach." "Assoc-A" "Assoc-V" "HS-grad" "Presch.") and CAPITAL_GAIN <= 5095.5 and OCCUPATION isIn ("ArmedF" "Exec." "Prof." "TechSup")	50.0%	50.0%
4	720	0	RELATIONSHIP isIn ("Husband" "Wife") and EDUCATION isIn ("10th" "11th" "12th" "1st-4th" "5th-6th" "7th-8th" "9th" "< Bach." "Assoc-A" "Assoc-V" "HS-grad" "Presch.") and CAPITAL_GAIN <= 5095.5 and OCCUPATION isIn ("Cleric." "Crafts" "Farming" "Handler" "House-s" "Machine" "Other" "Protec." "Sales" "Transp.")	25.3%	74.7%
5	47	1	RELATIONSHIP isIn ("Husband" "Wife") and EDUCATION isIn ("10th" "11th" "12th" "1st-4th" "5th-6th" "7th-8th" "9th" "< Bach." "Assoc-A" "Assoc-V" "HS-grad" "Presch.") and CAPITAL_GAIN > 5095.5	100.0%	0.0%

You can download the latest version of the Spreadsheet Add-In from the Oracle Technology Network.

http://www.oracle.com/technology/products/bi/odm/

DBMS_PREDICTIVE_ANALYTICS

Oracle Data Mining implements predictive analytics in the DBMS_PREDICTIVE_ANALYTICS PL/SQL package. The following SQL DESCRIBE statement shows the predictive analytics procedures with their parameters.

```
SQL> describe dbms_predictive_analytics

PROCEDURE EXPLAIN
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 DATA_TABLE_NAME                VARCHAR2                IN
 EXPLAIN_COLUMN_NAME            VARCHAR2                IN
 RESULT_TABLE_NAME              VARCHAR2                IN
 DATA_SCHEMA_NAME               VARCHAR2                IN     DEFAULT

PROCEDURE PREDICT
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 ACCURACY                       NUMBER                  OUT
 DATA_TABLE_NAME                VARCHAR2                IN
 CASE_ID_COLUMN_NAME            VARCHAR2                IN
 TARGET_COLUMN_NAME             VARCHAR2                IN
 RESULT_TABLE_NAME              VARCHAR2                IN
 DATA_SCHEMA_NAME               VARCHAR2                IN     DEFAULT

PROCEDURE PROFILE
 Argument Name                  Type                    In/Out Default?
 ------------------------------ ----------------------- ------ --------
 DATA_TABLE_NAME                VARCHAR2                IN
 TARGET_COLUMN_NAME             VARCHAR2                IN
 RESULT_TABLE_NAME              VARCHAR2                IN
 DATA_SCHEMA_NAME               VARCHAR2                IN     DEFAULT
```

Example: PREDICT

Example 3–1 shows how a simple PREDICT operation can be used to find the customers most likely to increase spending if given an affinity card.

The customer data, including current affinity card usage and other information such as gender, education, age, and household size, is stored in a view called MINING_DATA_APPLY_V. The results of the PREDICT operation are written to a table named p_result_tbl.

The PREDICT operation calculates both the prediction and the accuracy of the prediction. Accuracy, also known as predictive confidence, is a measure of the improvement over predictions that would be generated by a naive model. In the case of classification, a naive model would always guess the most common class. In Example 3–1, the improvement is almost 50%.

Example 3–1 Predict Customers Most Likely to Increase Spending with an Affinity Card

```
DECLARE
p_accuracy NUMBER(10,9);
BEGIN
  DBMS_PREDICTIVE_ANALYTICS.PREDICT(
       accuracy                 => p_accuracy,
       data_table_name          =>'mining_data_apply_v',
       case_id_column_name      =>'cust_id',
       target_column_name       =>'affinity_card',
       result_table_name       =>'p_result_tbl');
  DBMS_OUTPUT.PUT_LINE('Accuracy: ' || p_accuracy);
END;
/

Accuracy: .492433267
```

The following query returns the gender and average age of customers most likely to respond favorably to an affinity card.

```
SELECT cust_gender, COUNT(*) as cnt, ROUND(AVG(age)) as avg_age
             FROM mining_data_apply_v a, p_result_tbl b
      WHERE a.cust_id = b.cust_id
         AND b.prediction = 1
      GROUP BY a.cust_gender
      ORDER BY a.cust_gender;

C        CNT    AVG_AGE
- ---------- ----------
F         90         45
M        443         45
```

Behind the Scenes

This section provides some high-level information about the inner workings of Oracle predictive analytics. If you know something about data mining, you will find this information to be straight-forward and easy to understand. If you are unfamiliar with data mining, you can skip this section. You do not need to know this information to use predictive analytics.

> **See Also:** Chapter 2 for an overview of model functions and algorithms

EXPLAIN

EXPLAIN creates an attribute importance model. Attribute importance uses the Minimum Description Length algorithm to determine the relative importance of attributes in predicting a target value. EXPLAIN returns a list of attributes ranked in relative order of their impact on the prediction. This information is derived from the model details for the attribute importance model.

Attribute importance models are not scored against new data. They simply return information (model details) about the data you provide.

Attribute importance is described in "Feature Selection" on page 9-2.

PREDICT

PREDICT creates a Support Vector Machine (SVM) model for classification or regression.

PREDICT creates a Receiver Operating Characteristic (ROC) curve to analyze the per-case accuracy of the predictions. PREDICT optimizes the probability threshold for binary classification models. The probability threshold is the probability that the model uses to make a positive prediction. The default is 50%.

Accuracy

PREDICT returns a value indicating the accuracy, or predictive confidence, of the prediction. The accuracy is the improvement gained over a naive prediction. For a categorical target, a naive prediction would be the most common class, for a numerical target it would be the mean. For example, if a categorical target can have values small, medium, or large, and small is predicted more often than medium or large, a naive model would return small for all cases. Predictive analytics uses the accuracy of a naive model as the baseline accuracy.

The accuracy metric returned by PREDICT is a measure of improved maximum average accuracy versus a naive model's maximum average accuracy. Maximum average accuracy is the average per-class accuracy achieved at a specific probability threshold that is greater than the accuracy achieved at all other possible thresholds.

SVM is described in Chapter 18.

PROFILE

PROFILE creates a Decision Tree model to identify the characteristic of the attributes that predict a common target. For example, if the data has a categorical target with values small, medium, or large, PROFILE would describe how certain attributes typically predict each size.

The Decision Tree algorithm creates rules that describe the decisions that affect the prediction. The rules, expressed in XML as if-then-else statements, are returned in the model details. PROFILE returns XML that is derived from the model details generated by the algorithm.

Decision Tree is described in Chapter 11.

Part II

Mining Functions

In Part II, you will learn about the mining functions supported by Oracle Data Mining. Mining functions represent a class of mining problems that can be solved using data mining algorithms. When creating a data mining model, you must first specify the mining function then choose an appropriate algorithm to implement the function if one is not provided by default. Oracle Data Mining algorithms are described in Part III.

> **Note on Terminology:** The term *mining function* has no relationship to a *SQL language function*.
>
> Oracle Data Mining supports a family of SQL language functions that serve as operators for the deployment of mining models. See *Oracle Database SQL Language Reference*.

Part II contains the following chapters:

4

Regression

This chapter describes regression, the supervised mining function for predicting a continuous, numerical target.

> **See Also:** "Supervised Data Mining" on page 2-3

This chapter includes the following topics:

- About Regression
- Testing a Regression Model
- Regression Algorithms

About Regression

Regression is a data mining function that predicts numeric values along a continuum. Profit, sales, mortgage rates, house values, square footage, temperature, or distance could all be predicted using regression techniques. For example, a regression model could be used to predict the value of a house based on location, number of rooms, lot size, and other factors.

A regression task begins with a data set in which the target values are known. For example, a regression model that predicts house values could be developed based on observed data for many houses over a period of time. In addition to the value, the data might track the age of the house, square footage, number of rooms, taxes, school district, proximity to shopping centers, and so on. House value would be the target, the other attributes would be the predictors, and the data for each house would constitute a case.

In the model build (training) process, a regression algorithm estimates the value of the target as a function of the predictors for each case in the build data. These relationships between predictors and target are summarized in a model, which can then be applied to a different data set in which the target values are unknown.

Regression models are tested by computing various statistics that measure the difference between the predicted values and the expected values. The historical data for a regression project is typically divided into two data sets: one for building the model, the other for testing the model. See "Testing a Regression Model" on page 4-4.

Regression modeling has many applications in trend analysis, business planning, marketing, financial forecasting, time series prediction, biomedical and drug response modeling, and environmental modeling.

How Does Regression Work?

You do not need to understand the mathematics used in regression analysis to develop and use quality regression models for data mining. However, it is helpful to understand a few basic concepts.

Regression analysis seeks to determine the values of parameters for a function that cause the function to best fit a set of data observations that you provide. The following equation expresses these relationships in symbols. It shows that regression is the process of estimating the value of a continuous target (y) as a function (F) of one or more predictors ($\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_n$), a set of parameters (θ_1 , θ_2 , ..., θ_n), and a measure of error (e).

$$y = F(\mathbf{x},\theta) + e$$

The predictors can be understood as independent variables and the target as a dependent variable. The error, also called the **residual**, is the difference between the expected and predicted value of the dependent variable. The regression parameters are also known as **regression coefficients**. (See "Regression Coefficients" on page 4-3.)

The process of training a regression model involves finding the parameter values that minimize a measure of the error, for example, the sum of squared errors.

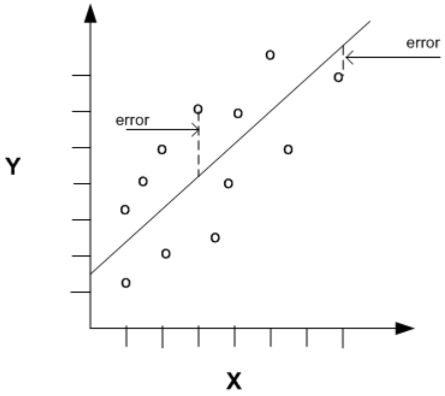
There are different families of regression functions and different ways of measuring the error.

Linear Regression

A linear regression technique can be used if the relationship between the predictors and the target can be approximated with a straight line.

Regression with a single predictor is the easiest to visualize. Simple linear regression with a single predictor is shown in Figure 4–1.

Figure 4–1 Linear Regression With a Single Predictor



Linear regression with a single predictor can be expressed with the following equation.

$$y = \theta_2\mathbf{x} + \theta_1 + e$$

The regression parameters in simple linear regression are:

- The **slope** of the line (θ_2) — the angle between a data point and the regression line
- The ***y* intercept** (θ_1) — the point where $\mathbf{x}$ crosses the y axis ($\mathbf{x} = 0$)

Multivariate Linear Regression

The term **multivariate linear regression** refers to linear regression with two or more predictors ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_n$). When multiple predictors are used, the regression line cannot be visualized in two-dimensional space. However, the line can be computed simply by expanding the equation for single-predictor linear regression to include the parameters for each of the predictors.

$$y = \theta_1 + \theta_2\mathbf{x}_1 + \theta_3\mathbf{x}_2 + \ldots.. \theta_n \mathbf{x}_{n-1} + e$$

Regression Coefficients

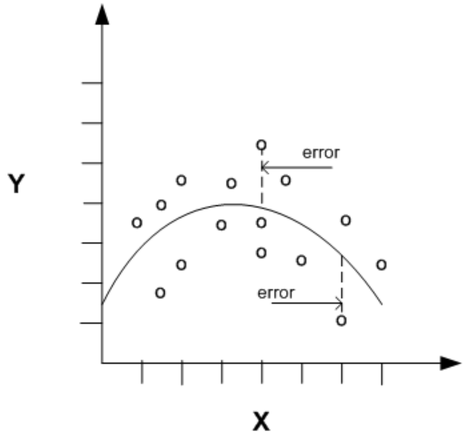
In multivariate linear regression, the regression parameters are often referred to as coefficients. When you build a multivariate linear regression model, the algorithm computes a coefficient for each of the predictors used by the model. The coefficient is a measure of the impact of the predictor **x** on the target *y*. Numerous statistics are available for analyzing the regression coefficients to evaluate how well the regression line fits the data. ("Regression Statistics" on page 4-4.)

Nonlinear Regression

Often the relationship between **x** and ***y*** cannot be approximated with a straight line. In this case, a nonlinear regression technique may be used. Alternatively, the data could be preprocessed to make the relationship linear.

Nonlinear regression models define ***y*** as a function of **x** using an equation that is more complicated than the linear regression equation. In Figure 4–2, **x** and ***y*** have a nonlinear relationship.

Figure 4–2 Nonlinear Regression With a SIngle Predictor



Multivariate Nonlinear Regression

The term **multivariate nonlinear regression** refers to nonlinear regression with two or more predictors ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_n$). When multiple predictors are used, the nonlinear relationship cannot be visualized in two-dimensional space.

Confidence Bounds

A regression model predicts a numeric target value for each case in the scoring data. In addition to the predictions, some regression algorithms can identify confidence bounds, which are the upper and lower boundaries of an interval in which the predicted value is likely to lie.

When a model is built to make predictions with a given confidence, the confidence interval will be produced along with the predictions. For example, a model might predict the value of a house to be $500,000 with a 95% confidence that the value will be between $475,000 and $525,000.

Testing a Regression Model

A regression model is tested by applying it to test data with known target values and comparing the predicted values with the known values.

The test data must be compatible with the data used to build the model and must be prepared in the same way that the build data was prepared. Typically the build data and test data come from the same historical data set. A percentage of the records is used to build the model; the remaining records are used to test the model.

Test metrics are used to assess how accurately the model predicts these known values. If the model performs well and meets the business requirements, it can then be applied to new data to predict the future.

Regression Statistics

The Root Mean Squared Error and the Mean Absolute Error are commonly used statistics for evaluating the overall quality of a regression model. Different statistics may also be available depending on the regression methods used by the algorithm.

Root Mean Squared Error

The Root Mean Squared Error (RMSE) is the square root of the average squared distance of a data point from the fitted line.

This SQL expression calculates the RMSE.

```
SQRT(AVG((predicted_value - actual_value) * (predicted_value - actual_value)))
```

This formula shows the RMSE in mathematical symbols. The large sigma character represents summation; *j* represents the current predictor, and *n* represents the number of predictors.

$$\mathbf{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Mean Absolute Error

The Mean Absolute Error (MAE) is the average of the absolute value of the residuals (error). The MAE is very similar to the RMSE but is less sensitive to large errors.

This SQL expression calculates the MAE.

```
AVG(ABS(predicted_value - actual_value))
```

This formula shows the MAE in mathematical symbols. The large sigma character represents summation; *j* represents the current predictor, and *n* represents the number of predictors.

$$\mathbf{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

Regression Algorithms

Oracle Data Mining supports two algorithms for regression. Both algorithms are particularly suited for mining data sets that have very high dimensionality (many attributes), including transactional and unstructured data.

- **Generalized Linear Models (GLM)**

  GLM is a popular statistical technique for linear modeling. Oracle Data Mining implements GLM for regression and for binary classification.

  GLM provides extensive coefficient statistics and model statistics, as well as row diagnostics. GLM also supports confidence bounds.

- **Support Vector Machines (SVM)**

  SVM is a powerful, state-of-the-art algorithm for linear and nonlinear regression. Oracle Data Mining implements SVM for regression and other mining functions.

  SVM regression supports two kernels: the Gaussian kernel for nonlinear regression, and the linear kernel for linear regression. SVM also supports active learning.

See Also:

Chapter 12, "Generalized Linear Models"

Chapter 18, "Support Vector Machines"

5 Classification

This chapter describes classification, the supervised mining function for predicting a categorical target.

See Also: "Supervised Data Mining" on page 2-3

This chapter includes the following topics:

About Classification

Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify loan applicants as low, medium, or high credit risks.

A classification task begins with a data set in which the class assignments are known. For example, a classification model that predicts credit risk could be developed based on observed data for many loan applicants over a period of time. In addition to the historical credit rating, the data might track employment history, home ownership or rental, years of residence, number and type of investments, and so on. Credit rating would be the target, the other attributes would be the predictors, and the data for each customer would constitute a case.

Classifications are discrete and do not imply order. Continuous, floating-point values would indicate a numerical, rather than a categorical, target. A predictive model with a numerical target uses a regression algorithm, not a classification algorithm.

The simplest type of classification problem is binary classification. In binary classification, the target attribute has only two possible values: for example, high credit rating or low credit rating. Multiclass targets have more than two values: for example, low, medium, high, or unknown credit rating.

In the model build (training) process, a classification algorithm finds relationships between the values of the predictors and the values of the target. Different classification algorithms use different techniques for finding relationships. These relationships are summarized in a model, which can then be applied to a different data set in which the class assignments are unknown.

Classification models are tested by comparing the predicted values to known target values in a set of test data. The historical data for a classification project is typically divided into two data sets: one for building the model; the other for testing the model. See "Testing a Classification Model" on page 5-2.

Scoring a classification model results in class assignments and probabilities for each case. For example, a model that classifies customers as low, medium, or high value would also predict the probability of each classification for each customer.

Classification has many applications in customer segmentation, business modeling, marketing, credit analysis, and biomedical and drug response modeling.

Testing a Classification Model

A classification model is tested by applying it to test data with known target values and comparing the predicted values with the known values.

The test data must be compatible with the data used to build the model and must be prepared in the same way that the build data was prepared. Typically the build data and test data come from the same historical data set. A percentage of the records is used to build the model; the remaining records are used to test the model.

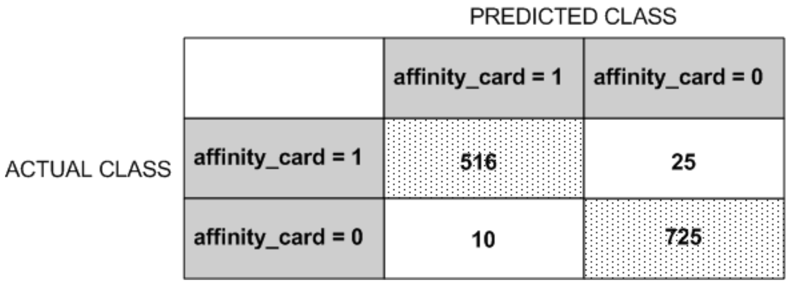
Test metrics are used to assess how accurately the model predicts the known values. If the model performs well and meets the business requirements, it can then be applied to new data to predict the future.

Confusion Matrix

A confusion matrix displays the number of correct and incorrect predictions made by the model compared with the actual classifications in the test data. The matrix is *n*-by-*n*, where *n* is the number of classes.

Figure 5–1 shows a confusion matrix for a binary classification model. The rows present the number of actual classifications in the test data. The columns present the number of predicted classifications made by the model.

Figure 5–1 Confusion Matrix for a Binary Classification Model



		PREDICTED CLASS	
		affinity_card = 1	affinity_card = 0
ACTUAL CLASS	affinity_card = 1	516	25
	affinity_card = 0	10	725

In this example, the model correctly predicted the positive class for `affinity_card` 516 times and incorrectly predicted it 25 times. The model correctly predicted the negative class for `affinity_card` 725 times and incorrectly predicted it 10 times. The following can be computed from this confusion matrix:

- The model made 1241 **correct predictions** (516 + 725).
- The model made 35 **incorrect predictions** (25 + 10).
- There are 1276 **total scored cases** (516 + 25 + 10 + 725).
- The **error rate** is 35/1276 = 0.0274.

- The **overall accuracy rate** is 1241/1276 = 0.9725.

Lift

Lift measures the degree to which the predictions of a classification model are better than randomly-generated predictions. Lift applies to binary classification only, and it requires the designation of a positive class. (See "Positive and Negative Classes" on page 5-5.) If the model itself does not have a binary target, you can compute lift by designating one class as positive and combining all the other classes together as one negative class.

Numerous statistics can be calculated to support the notion of lift. Basically, lift can be understood as a ratio of two percentages: the percentage of correct positive classifications made by the model to the percentage of actual positive classifications in the test data. For example, if 40% of the customers in a marketing survey have responded favorably (the positive classification) to a promotional campaign in the past and the model accurately predicts 75% of them, the lift would be obtained by dividing .75 by .40. The resulting lift would be 1.875.

Lift is computed against quantiles that each contain the same number of cases. The data is divided into quantiles after it is scored. It is ranked by probability of the positive class from highest to lowest, so that the highest concentration of positive predictions is in the top quantiles. A typical number of quantiles is 10.

Lift is commonly used to measure the performance of response models in marketing applications. The purpose of a response model is to identify segments of the population with potentially high concentrations of positive responders to a marketing campaign. Lift reveals how much of the population must be solicited to obtain the highest percentage of potential responders.

Lift Statistics

Oracle Data Mining computes the following lift statistics:

- **Probability threshold** for a quantile *n* is the minimum probability for the positive target to be included in this quantile or any preceding quantiles (quantiles *n*-1, *n*-2,..., 1). If a cost matrix is used, a cost threshold is reported instead. The cost threshold is the maximum cost for the positive target to be included in this quantile or any of the preceding quantiles. (See "Costs" on page 5-5.)
- **Cumulative gain** is the ratio of the cumulative number of positive targets to the total number of positive targets.
- **Target density** of a quantile is the number of true positive instances in that quantile divided by the total number of instances in the quantile.
- **Cumulative target density** for quantile *n* is the target density computed over the first *n* quantiles.
- **Quantile lift** is the ratio of target density for the quantile to the target density over all the test data.
- **Cumulative percentage of records** for a quantile is the percentage of all cases represented by the first *n* quantiles, starting at the end that is most confidently positive, up to and including the given quantile.
- **Cumulative number of targets** for quantile *n* is the number of true positive instances in the first *n* quantiles.
- **Cumulative number of nontargets** is the number of actually negative instances in the first *n* quantiles.

- **Cumulative lift** for a quantile is the ratio of the cumulative target density to the target density over all the test data.

Receiver Operating Characteristic (ROC)

ROC is another metric for comparing predicted and actual target values in a classification model. ROC, like lift, applies to binary classification and requires the designation of a positive class. (See "Positive and Negative Classes" on page 5-5.)

You can use ROC to gain insight into the decision-making ability of the model. How likely is the model to accurately predict the negative or the positive class?

ROC measures the impact of changes in the **probability threshold**. The probability threshold is the decision point used by the model for classification. The default probability threshold for binary classification is .5. When the probability of a prediction is 50% or more, the model predicts that class. When the probability is less than 50%, the other class is predicted. (In multiclass classification, the predicted class is the one predicted with the highest probability.)

The ROC Curve

ROC can be plotted as a curve on an X-Y axis. The **false positive rate** is placed on the X axis. The **true positive rate** is placed on the Y axis.

The top left corner is the optimal location on an ROC graph, indicating a high true positive rate and a low false positive rate.

Area Under the Curve

The area under the ROC curve (AUC) measures the discriminating ability of a binary classification model. The larger the AUC, the higher the likelihood that an actual positive case will be assigned a higher probability of being positive than an actual negative case. The AUC measure is especially useful for data sets with unbalanced target distribution (one target class dominates the other).

ROC and Model Bias

Changes in the probability threshold affect the predictions made by the model. For instance, if the threshold for predicting the positive class is changed from .5 to.6, fewer positive predictions will be made. This will affect the distribution of values in the confusion matrix: the number of true and false positives and true and false negatives will all be different.

The ROC curve for a model represents all the possible combinations of values in its confusion matrix. You can use ROC to find the probability thresholds that yield the highest overall accuracy or the highest per-class accuracy. For example, if it is important to you to accurately predict the positive class, but you don't care about prediction errors for the negative class, you could lower the threshold for the positive class. This would bias the model in favor of the positive class.

A cost matrix is a convenient mechanism for changing the probability thresholds for model scoring.

See Also: "Costs" on page 5-5

ROC Statistics

Oracle Data Mining computes the following ROC statistics:

- **Probability threshold:** The minimum predicted positive class probability resulting in a positive class prediction. Different threshold values result in different hit rates and different false alarm rates.
- **True negatives:** Negative cases in the test data with predicted probabilities strictly less than the probability threshold (correctly predicted).
- **True positives:** Positive cases in the test data with predicted probabilities greater than or equal to the probability threshold (correctly predicted).
- **False negatives:** Positive cases in the test data with predicted probabilities strictly less than the probability threshold (incorrectly predicted).
- **False positives:** Negative cases in the test data with predicted probabilities greater than or equal to the probability threshold (incorrectly predicted).
- **True positive fraction**: Hit rate. (true positives/(true positives + false negatives))
- **False positive fraction:** False alarm rate. (false positives**/**(false positives + true negatives))

Biasing a Classification Model

Costs, prior probabilities, and class weights are methods for biasing classification models.

Costs

A cost matrix is a mechanism for influencing the decision making of a model. A cost matrix can cause the model to minimize costly misclassifications. It can also cause the model to maximize beneficial accurate classifications.

For example, if a model classifies a customer with poor credit as low risk, this error is costly. A cost matrix could bias the model to avoid this type of error. The cost matrix might also be used to bias the model in favor of the correct classification of customers who have the worst credit history.

ROC is a useful metric for evaluating how a model behaves with different probability thresholds. You can use ROC to help you find optimal costs for a given classifier given different usage scenarios. You can use this information to create cost matrices to influence the deployment of the model.

Costs Versus Accuracy

Like a confusion matrix, a cost matrix is an *n*-by-*n* matrix, where *n* is the number of classes. Both confusion matrices and cost matrices include each possible combination of actual and predicted results based on a given set of test data.

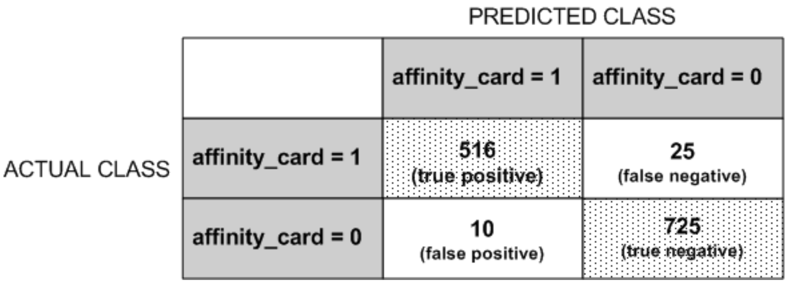
A confusion matrix is used to measure accuracy, the ratio of correct predictions to the total number of predictions. A cost matrix is used to specify the relative importance of accuracy for different predictions. In most business applications, it is important to consider costs in addition to accuracy when evaluating model quality. (See "Confusion Matrix" on page 5-2.)

Positive and Negative Classes

The positive class is the class that you care the most about. Designation of a positive class is required for computing lift and ROC. (See "Lift" on page 5-3 and "Receiver Operating Characteristic (ROC)" on page 5-4).

In the confusion matrix in Figure 5–2, the value 1 is designated as the positive class. This means that the creator of the model has determined that it is more important to accurately predict customers who will increase spending with an affinity card (affinity_card=1) than to accurately predict non-responders (affinity_card=0). If you give affinity cards to some customers who are not likely to use them, there is little loss to the company since the cost of the cards is low. However, if you overlook the customers who are likely to respond, you miss the opportunity to increase your revenue.

Figure 5–2 Positive and Negative Predictions



		PREDICTED CLASS	
		affinity_card = 1	affinity_card = 0
ACTUAL CLASS	affinity_card = 1	516 (true positive)	25 (false negative)
	affinity_card = 0	10 (false positive)	725 (true negative)

The true and false positive rates in this confusion matrix are:

- False positive rate — 10/(10 + 725) =.01
- True positive rate — 516/(516 + 25) =.95

Assigning Costs and Benefits

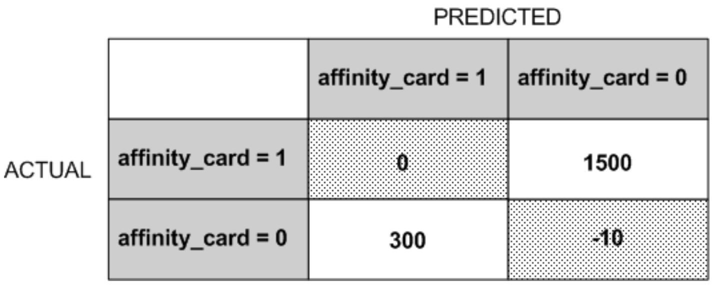
In a cost matrix, positive numbers (costs) can be used to influence negative outcomes. Since negative costs are interpreted as benefits, negative numbers (benefits) can be used to influence positive outcomes.

Suppose you have calculated that it costs your business $1500 when you do not give an affinity card to a customer who would increase spending. Using the model with the confusion matrix shown in Figure 5–2, each false negative (misclassification of a responder) would cost $1500. Misclassifying a non-responder is less expensive to your business. You figure that each false positive (misclassification of a non-responder) would only cost $300.

You want to keep these costs in mind when you design a promotion campaign. You estimate that it will cost $10 to include a customer in the promotion. For this reason, you associate a benefit of $10 with each true negative prediction, because you can simply eliminate those customers from your promotion. Each customer that you eliminate represents a savings of $10. In your cost matrix, you would specify this benefit as -10, a negative cost.

Figure 5–3 shows how you would represent these costs and benefits in a cost matrix.

Figure 5–3 Cost Matrix



PREDICTED

		affinity_card = 1	affinity_card = 0
ACTUAL	affinity_card = 1	0	1500
	affinity_card = 0	300	-10

With Oracle Data Mining you can specify costs to influence the scoring of any classification model. Decision Tree models can also use a cost matrix to influence the model build.

Priors

With Bayesian models, you can specify prior probabilities to offset differences in distribution between the build data and the real population (scoring data).

In many problems, one target value dominates in frequency. For example, the positive responses for a telephone marketing campaign may be 2% or less, and the occurrence of fraud in credit card transactions may be less than 1%. A classification model built on historic data of this type may not observe enough of the rare class to be able to distinguish the characteristics of the two classes; the result could be a model that when applied to new data predicts the frequent class for every case. While such a model may be highly accurate, it may not be very useful. This illustrates that it is not a good idea to rely solely on accuracy when judging the quality of a classification model.

To correct for unrealistic distributions in the training data, you can specify priors for the model build process. Other approaches to compensating for data distribution issues include stratified sampling and anomaly detection. (See Chapter 6.)

Classification Algorithms

Oracle Data Mining provides the following algorithms for classification:

- **Decision Tree**

  Decision trees automatically generate rules, which are conditional statements that reveal the logic used to build the tree. See Chapter 11, "Decision Tree".

- **Naive Bayes**

  Naive Bayes uses Bayes' Theorem, a formula that calculates a probability by counting the frequency of values and combinations of values in the historical data. See Chapter 15, "Naive Bayes".

- **Generalized Linear Models (GLM)**

  GLM is a popular statistical technique for linear modeling. Oracle Data Mining implements GLM for binary classification and for regression.

  GLM provides extensive coefficient statistics and model statistics, as well as row diagnostics. GLM also supports confidence bounds.

- **Support Vector Machine**

Support Vector Machine (SVM) is a powerful, state-of-the-art algorithm based on linear and nonlinear regression. Oracle Data Mining implements SVM for binary and multiclass classification. See Chapter 18, "Support Vector Machines".

The nature of the data determines which classification algorithm will provide the best solution to a given problem. The algorithm can differ with respect to accuracy, time to completion, and transparency. In practice, it sometimes makes sense to develop several models for each algorithm, select the best model for each algorithm, and then choose the best of those for deployment.

6

Anomaly Detection

This chapter describes anomaly detection, an unsupervised mining function for detecting rare cases in the data.

See Also: "Unsupervised Data Mining" on page 2-4

Reference:

Campos, M.M., Milenova, B.L., Yarmus, J.S., "Creation and Deployment of Data Mining-Based Intrusion Detection Systems in Oracle Database 10*g*"

`http://www.oracle.com/technology/products/bi/odm/`

This chapter contains the following sections:

About Anomaly Detection

The goal of anomaly detection is to identify cases that are unusual within data that is seemingly homogeneous. Anomaly detection is an important tool for detecting fraud, network intrusion, and other rare events that may have great significance but are hard to find.

Anomaly detection can be used to solve problems like the following:

- A law enforcement agency compiles data about illegal activities, but nothing about legitimate activities. How can suspicious activity be flagged?

  The law enforcement data is all of one class. There are no counter-examples.

- An insurance agency processes millions of insurance claims, knowing that a very small number are fraudulent. How can the fraudulent claims be identified?

  The claims data contains very few counter-examples. They are outliers.

One-Class Classification

Anomaly detection is a form of classification. See "About Classification" on page 5-1 for an overview of the classification mining function.

Anomaly detection is implemented as one-class classification, because only one class is represented in the training data. An anomaly detection model predicts whether a data point is typical for a given distribution or not. An atypical data point can be either an outlier or an example of a previously unseen class.

Normally, a classification model must be trained on data that includes both examples and counter-examples for each class so that the model can learn to distinguish between them. For example, a model that predicts side effects of a medication should be trained on data that includes a wide range of responses to the medication.

A one-class classifier develops a profile that generally describes a typical case in the training data. Deviation from the profile is identified as an anomaly. One-class classifiers are sometimes referred to as positive security models, because they seek to identify "good" behaviors and assume that all other behaviors are bad.

Note: Solving a one-class classification problem can be difficult. The accuracy of one-class classifiers cannot usually match the accuracy of standard classifiers built with meaningful counterexamples.

The goal of anomaly detection is to provide some useful information where no information was previously attainable. However, if there are enough of the "rare" cases so that stratified sampling could produce a training set with enough counterexamples for a standard classification model, then that would generally be a better solution.

Anomaly Detection for Single-Class Data

In single-class data, all the cases have the same classification. Counter-examples, instances of another class, may be hard to specify or expensive to collect. For instance, in text document classification, it may be easy to classify a document under a given topic. However, the universe of documents outside of this topic may be very large and diverse. Thus it may not be feasible to specify other types of documents as counter-examples.

Anomaly detection could be used to find unusual instances of a particular type of document.

Anomaly Detection for Finding Outliers

Outliers are cases that are unusual because they fall outside the distribution that is considered normal for the data. For example, census data might show a median household income of $70,000 and a mean household income of $80,000, but one or two households might have an income of $200,000. These cases would probably be identified as outliers.

The distance from the center of a normal distribution indicates how typical a given point is with respect to the distribution of the data. Each case can be ranked according to the probability that it is either typical or atypical.

The presence of outliers can have a deleterious effect on many forms of data mining. Anomaly detection can be used to identify outliers before mining the data.

Anomaly Detection Algorithm

Oracle Data Mining supports One-Class Support Vector Machine (SVM) for anomaly detection. When used for anomaly detection, SVM classification does not use a target.

See Also: "One-Class SVM" on page 18-5

7

Clustering

This chapter describes clustering, the unsupervised mining function for discovering natural groupings in the data.

See Also: "Unsupervised Data Mining" on page 2-4

This chapter includes the following topics:

- About Clustering
- Hierarchical Clustering
- Clustering Algorithms

About Clustering

Clustering analysis finds clusters of data objects that are similar in some sense to one another. The members of a cluster are more like each other than they are like members of other clusters. The goal of clustering analysis is to find high-quality clusters such that the inter-cluster similarity is low and the intra-cluster similarity is high.

Clustering, like classification, is used to segment the data. Unlike classification, clustering models segment data into groups that were not previously defined. Classification models segment data by assigning it to previously-defined classes, which are specified in a target. Clustering models do not use a target.

Clustering is useful for exploring data. If there are many cases and no obvious groupings, clustering algorithms can be used to find natural groupings.

Clustering can serve as a useful data-preprocessing step to identify homogeneous groups on which to build supervised models.

Clustering can also be used for anomaly detection. Once the data has been segmented into clusters, you might find that some cases do not fit well into any clusters. These cases are anomalies or outliers.

How are Clusters Computed?

There are several different approaches to the computation of clusters. Oracle Data Mining supports distance-based and grid-based clustering:

- **Distance-based** — This type of clustering uses a distance metric to determine similarity between data objects. The distance metric measures the distance between actual cases in the cluster and the prototypical case for the cluster. The prototypical case is known as the **centroid**.

Oracle Data Mining supports an enhanced version of *k*-Means, a distance-based clustering algorithm.

- **Grid-based** — This type of clustering divides the input space into hyper-rectangular cells, discards the low-density cells, and then combines adjacent high-density cells to form clusters.

  Oracle Data Mining supports Orthogonal Partitioning Clustering (O-Cluster), a proprietary grid-based clustering algorithm.

Reference:

Campos, M.M., Milenova, B.L., "O-Cluster: Scalable Clustering of Large High Dimensional Data Sets", Oracle Data Mining Technologies, 10 Van De Graaff Drive, Burlington, MA 01803.

`http://www.oracle.com/technology/products/bi/odm/`

Scoring New Data

Oracle Data Mining supports the scoring operation for clustering. The clusters discovered by the algorithm are used to generate a Bayesian probability model that can be used to score new data.

Hierarchical Clustering

The clustering algorithms supported by Oracle Data Mining perform hierarchical clustering. The leaf clusters are the final clusters generated by the algorithm. Clusters higher up in the hierarchy are intermediate clusters.

Rules

Rules describe the data in each cluster. A rule is a conditional statement that captures the logic used to split a parent cluster into child clusters. A rule describes the conditions for a case to be assigned with some probability to a cluster.

Support and Confidence

Support and **confidence** are metrics that describe the relationships between clustering rules and cases. Support is the percentage of cases for which the rule holds. Confidence is the probability that a case described by this rule will actually be assigned to the cluster.

Evaluating a Clustering Model

Since known classes are not used in clustering, the interpretation of clusters can present difficulties. How do you know if the clusters can reliably be used for business decision making?

Oracle Data Mining clustering models support a high degree of model transparency. You can evaluate the model by examining information generated by the clustering algorithm: for example, the centroid of a distance-based cluster. Moreover, because the clustering process is hierarchical, you can evaluate the rules and other information related to each cluster's position in the hierarchy.

Clustering Algorithms

Oracle Data Mining supports two clustering algorithms: *k*-Means and O-Cluster. The main characteristics of the two algorithms are compared in Table 7–1.

Table 7–1 Clustering Algorithms Compared

Feature	*k*-Means	O-Cluster
Clustering methodolgy	Distance-based	Grid-based
Number of cases	Handles data sets of any size	More appropriate for data sets that have more than 500 cases. Handles large tables through active sampling
Number of attributes	More appropriate for data sets with a low number of attributes	More appropriate for data sets with a high number of attributes
Number of clusters	User-specified	Automatically determined
Hierarchical clustering	Yes	Yes
Probabilistic cluster assignment	Yes	Yes

See Also:

Chapter 13, "k-Means"

Chapter 17, "O-Cluster"

8

Association

This chapter describes association, the unsupervised mining function for discovering association rules.

See Also: "Unsupervised Data Mining" on page 2-4

This chapter contains the following topics:

- About Association
- Transactional Data
- Association Algorithm

About Association

Association is a data mining function that discovers the probability of the co-occurrence of items in a collection. The relationships between co-occurring items are expressed as **association rules**.

Association Rules

The results of an association model are the rules that identify patterns of association within the data. Oracle Data Mining does not support the scoring operation for association modeling.

Association rules are ranked by these metrics:

Support — How often do these items occur together in the data?
Confidence — How likely are these items to occur together in the data?

Market-Basket Analysis

Association rules are often used to analyze sales transactions. For example, it might be noted that customers who buy cereal at the grocery store often buy milk at the same time. In fact, association analysis might find that 85% of the checkout sessions that include cereal also include milk. This relationship could be formulated as the following rule.

```
Cereal implies milk with 85% confidence
```

This application of association modeling is called **market-basket analysis**. It is valuable for direct marketing, sales promotions, and for discovering business trends. Market-basket analysis can also be used effectively for store layout, catalog design, and cross-sell.

Association Rules and eCommerce

Association modeling has important applications in other domains as well. For example, in e-commerce applications, association rules may be used for Web page personalization. An association model might find that a user who visits pages A and B is 70% likely to also visit page C in the same session. Based on this rule, a dynamic link could be created for users who are likely to be interested in page C. The association rule could be expressed as follows.

```
A and B imply C with 70% confidence
```

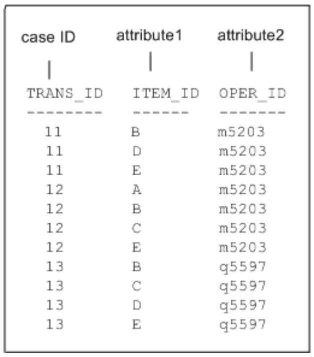
See Also: "Confidence" on page 10-6

Transactional Data

Unlike other data mining functions, association is transaction-based. In transaction processing, a case includes a collection of items such as the contents of a market basket at the checkout counter. The collection of items in the transaction is an attribute of the transaction. Other attributes might be a timestamp or user ID associated with the transaction.

Transactional data, also known as **market-basket data**, is said to be in **multi-record case** format because a set of records (rows) constitute a case. For example, in Figure 8–1, case 11 is made up of three rows while cases 12 and 13 are each made up of four rows.

Figure 8–1 Transactional Data



```
case ID     attribute1    attribute2
   |            |             |
TRANS_ID    ITEM_ID   OPER_ID
--------    ------    -------
   11       B         m5203
   11       D         m5203
   11       E         m5203
   12       A         m5203
   12       B         m5203
   12       C         m5203
   12       E         m5203
   13       B         q5597
   13       C         q5597
   13       D         q5597
   13       E         q5597
```

Non transactional data is said to be in **single-record case** format because a single record (row) constitutes a case. In Oracle Data Mining, association models can be built using either transactional or non transactional data. If the data is non transactional, it must be transformed to a nested column before association mining activities can be performed.

See Also:

Oracle Data Mining Application Developer's Guide

"Data Preparation for Apriori" on page 10-2

Association Algorithm

Oracle Data Mining uses the Apriori algorithm to calculate association rules for items in frequent itemsets.

See Also: Chapter 10, "Apriori"

9

Feature Selection and Extraction

This chapter describes the feature selection and extraction mining functions. Oracle Data Mining supports a supervised form of feature selection and an unsupervised form of feature extraction.

> **See Also:**
>
> "Supervised Data Mining" on page 2-3
>
> "Unsupervised Data Mining" on page 2-4

This chapter contains the following sections:

- Finding the Best Attributes
- Feature Selection
- Feature Extraction
- Feature Selection and Extraction Algorithms

Finding the Best Attributes

Sometimes too much information can reduce the effectiveness of data mining. Some of the columns of data attributes assembled for building and testing a model may not contribute meaningful information to the model. Some may actually detract from the quality and accuracy of the model.

For example, you might collect a great deal of data about a given population because you want to predict the likelihood of a certain illness within this group. Some of this information, perhaps much of it, will have little or no effect on susceptibility to the illness. Attributes such as the number of cars per household may have no effect whatsoever.

Irrelevant attributes add noise to the data and affect model accuracy. Noise increases the size of the model and the time and system resources needed for model building and scoring.

Moreover, data sets with many attributes may contain groups of attributes that are correlated. These attributes may actually be measuring the same underlying feature. Their presence together in the build data can skew the logic of the algorithm and affect the accuracy of the model.

Wide data (many attributes) generally presents processing challenges for data mining algorithms. Model attributes are the dimensions of the processing space used by the algorithm. The higher the dimensionality of the processing space, the higher the computation cost involved in algorithmic processing.

To minimize the effects of noise, correlation, and high dimensionality, some form of dimension reduction is sometimes a desirable preprocessing step for data mining. Feature selection and extraction are two approaches to dimension reduction.

- **Feature selection** — Selecting the most relevant attributes
- **Feature extraction** — Combining attributes into a new reduced set of features

Feature Selection

Oracle Data Mining supports feature selection in the **attribute importance** mining function. Attribute importance is a supervised function that ranks attributes according to their significance in predicting a target.

Finding the most significant predictors is the goal of some data mining projects. For example, a model might seek to find the principal characteristics of clients who pose a high credit risk.

Attribute importance is also useful as a preprocessing step in classification modeling, especially for models that use Naive Bayes or Support Vector Machine. The Decision Tree algorithm includes components that rank attributes as part of the model build.

Oracle Data Mining does not support the scoring operation for attribute importance. The results of attribute importance are the attributes of the build data ranked according to their predictive influence. The ranking and the measure of importance can be used for selecting attributes.

Feature Extraction

Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction actually transforms the attributes. The transformed attributes, or **features**, are linear combinations of the original attributes.

The feature extraction process results in a much smaller and richer set of attributes. The maximum number of features may be user-specified or determined by the algorithm. By default, it is determined by the algorithm.

Models built on extracted features may be of higher quality, because the data is described by fewer, more meaningful attributes.

Feature extraction projects a data set with higher dimensionality onto a smaller number of dimensions. As such it is useful for data visualization, since a complex data set can be effectively visualized when it is reduced to two or three dimensions.

Some applications of feature extraction are latent semantic analysis, data compression, data decomposition and projection, and pattern recognition. Feature extraction can also be used to enhance the speed and effectiveness of supervised learning.

Feature extraction can be used to extract the themes of a document collection, where documents are represented by a set of key words and their frequencies. Each theme (feature) is represented by a combination of keywords. The documents in the collection can then be expressed in terms of the discovered themes.

See Also: Chapter 20, "Text Mining"

Feature Selection and Extraction Algorithms

Oracle Data Mining uses the Minimum Description Length (MDL) algorithm for feature selection (attribute importance).

See Also: Chapter 14, "Minimum Description Length"

Oracle Data Mining uses the Non-Negative Matrix Factorization (NMF) algorithm for feature extraction.

See Also: Chapter 16, "Non-Negative Matrix Factorization"

See *Oracle Data Mining Application Developer's Guide* for information about feature extraction for text mining.

Part III

Algorithms

Part III provides basic conceptual information to help you understand the algorithms supported by Oracle Data Mining. In cases where more than one algorithm is available for a given mining function, this information in these chapters should help you make the most appropriate choice. Also, if you have a general understanding of the workings of an algorithm, you will be better prepared to optimize its use with tuning parameters and data preparation.

Part III contains the following chapters:

10

Apriori

This chapter describes Apriori, the algorithm used by Oracle Data Mining for calculating association rules.

See Also: Chapter 8, "Association"

This chapter contains the following topics:

About Apriori

An association mining problem can be decomposed into two subproblems:

- Find all combinations of items in a set of transactions that occur with a specified minimum frequency. These combinations are called **frequent itemsets**.
- Calculate rules that express the probable co-occurrence of items within frequent itemsets. (See "Example: Calculating Rules from Frequent Itemsets" on page 10-4.)

Apriori calculates the probability of an item being present in a frequent itemset, given that another item or items is present.

Association rule mining is not recommended for finding associations involving rare events in problem domains with a large number of items. Apriori discovers patterns with frequency above the minimum support threshold. Therefore, in order to find associations involving rare events, the algorithm must run with very low minimum support values. However, doing so could potentially explode the number of enumerated itemsets, especially in cases with a large number of items. This could increase the execution time significantly. Classification or anomaly detection may be more suitable for discovering rare events when the data has a high number of attributes.

Association Rules and Frequent Itemsets

The Apriori algorithm calculates rules that express probabilistic relationships between items in frequent itemsets For example, a rule derived from frequent itemsets containing A, B, and C might state that if A and B are included in a transaction, then C is likely to also be included.

An association rule states that an item or group of items implies the presence of another item with some probability. Unlike decision tree rules, which predict a target, association rules simply express correlation.

Antecedent and Consequent

The IF component of an association rule is known as the **antecedent**. The THEN component is known as the **consequent**. The antecedent and the consequent are disjoint; they have no items in common.

Oracle Data Mining supports association rules that have one or more items in the antecedent and a single item in the consequent.

Confidence

Rules have an associated , which is the conditional probability that the consequent will occur given the occurrence of the antecedent. The minimum confidence for rules can be specified by the user.

Data Preparation for Apriori

Association models are designed to use transactional data. In transactional data, there is a one-to-many relationship between the case identifier and the values for each case. Each case ID/value pair is specified in a separate record (row).

Native Transactional Data and Star Schemas

Transactional data may be stored in native transactional format, with a non-unique case ID column and a values column, or it may be stored in some other configuration, such as a star schema. If the data is not stored in native transactional format, it must be transformed to a nested column for processing by the Apriori algorithm.

See Also:

"Transactional Data" on page 8-2

Oracle Data Mining Application Developer's Guide for details about transforming transactional data to nested columns

Items and Collections

In transactional data, a collection of items is associated with each case. The collection could theoretically include all possible members of the collection. For example, all products could theoretically be purchased in a single market-basket transaction. However, in actuality, only a tiny subset of all possible items are present in a given transaction; the items in the market-basket represent only a small fraction of the items available for sale in the store.

Sparse Data

Missing items in a collection indicate **sparsity**. Missing items may be present with a null value, or they may simply be missing.

Nulls in transactional data are assumed to represent values that are known but not present in the transaction. For example, three items out of hundreds of possible items might be purchased in a single transaction. The items that were not purchased are known but not present in the transaction.

Oracle Data Mining assumes sparsity in transactional data. The Apriori algorithm is optimized for processing sparse data.

See Also: *Oracle Data Mining Application Developer's Guide* for information about missing value treatment

Note: Apriori is not affected by Automatic Data Preparation.

Calculating Association Rules

The first step in association analysis is the enumeration of **itemsets**. An itemset is any combination of two or more items in a transaction.

Itemsets

The maximum number of items in an itemset is user-specified. If the maximum is two, all the item pairs will be counted. If the maximum is greater than two, all the item pairs, all the item triples, and all the item combinations up to the specified maximum will be counted.

Example 10–1 Sample Transactional Data

```
TRANS_ID   ITEM_ID
---------  -------------------
11         B
11         D
11         E
12         A
12         B
12         C
12         E
13         B
13         C
13         D
13         E
```

Table 10–1 shows the itemsets derived from the transactions shown in Example 10–1, assuming that maximum number of items in an itemset is set to 3.

Table 10–1 Itemsets

Transaction	Itemsets
11	(B,D) (B,E) (D,E) (B,D,E)
12	(A,B) (A,C) (A,E) (B,C) (B,E) (C,E) (A,B,C) (A,B,E) (A,C,E) (B,C,E)
13	(B,C) (B,D) (B,E) (C,D) (C,E) (D,E) (B,C,D) (B,C,E) (B,D,E) (C,D,E)

Frequent Itemsets

Association rules are calculated from itemsets. If rules are generated from all possible itemsets, there may be a very high number of rules and the rules may not be very meaningful. Also, the model may take a long time to build. Typically it is desirable to only generate rules from itemsets that are well-represented in the data. **Frequent itemsets** are those that occur with a minimum frequency specified by the user.

The minimum frequent itemset **support** is a user-specified percentage that limits the number of itemsets used for association rules. An itemset must appear in at least this percentage of all the transactions if it is to be used as a basis for rules.

Table 10–2 shows the itemsets from Table 10–1 that are frequent itemsets with support > 66%.

Table 10–2 Frequent Itemsets

Frequent Itemset	Transactions	Support
(B,C)	2 of 3	67%
(B,D)	2 of 3	67%
(B,E)	3 of 3	100%
(C,E)	2 of 3	67%
(D,E)	2 of 3	67%
(B,C,E)	2 of 3	67%
(B,D,E)	2 of 3	67%

See Also: Chapter 10, "Apriori" for information about the calculation of association rules

Example: Calculating Rules from Frequent Itemsets

Table 10–4 and Table 10–4 show the itemsets and frequent itemsets that were calculated in Chapter 8. The frequent itemsets are the itemsets that occur with a minimum support of 67%; at least 2 of the 3 transactions must include the itemset.

Table 10–3 Itemsets

Transaction	Itemsets
11	(B,D) (B,E) (D,E) (B,D,E)
12	(A,B) (A,C) (A,E) (B,C) (B,E) (C,E) (A,B,C) (A,B,E) (A,C,E) (B,C,E)
13	(B,C) (B,D) (B,E) (C,D) (C,E) (D,E) (B,C,D) (B,C,E) (B,D,E) (C,D,E)

Table 10–4 Frequent Itemsets with Minimum Support 67%

Itemset	Transactions	Support
(B,C)	12 and 13	67%
(B,D)	11 and 13	67%
(B,E)	11, 12, and 13	100%
(C,E)	12 and 13	67%
(D,E)	11 and 13	67%
(B,C,E)	12 and 13	67%
(B,D,E)	11 and 13	67%

A rule expresses a conditional probability. Confidence in a rule is calculated by dividing the probability of the items occurring together by the probability of the occurrence of the antecedent.

For example, if B (antecedent) is present, what is the chance that C (consequent) will also be present? What is the confidence for the rule "IF B, THEN C"?

As shown in Table 10–3:

- All 3 transactions include B (3/3 or 100%)
- Only 2 transactions include both B and C (2/3 or 67%)
- Therefore, the confidence of the rule "IF B, THEN C" is 67/100 or 67%.

Table 10–5, shows the rules that could be derived from the frequent itemsets in Table 10–4.

Table 10–5 Frequent Itemsets and Rules

Frequent Itemset	Rules	prob(antecedent and consequent) / prob(antecedent)	Confidence
(B,C)	(If B then C) (If C then B)	67/100 67/67	67% 100%
(B,D)	(If B then D) (If D then B)	67/100 67/67	67% 100%
(B,E)	(If B then E) (If E then B)	100/100 100/100	100% 100%
(C,E)	(If C then E) (If E then C)	67/67 67/100	100% 67%
(D,E)	(If D then E) I(f E then D)	67/67 67/100	100% 67%
(B,C,E)	(If B and C then E) (If B and E then C) (If C and E then B)	67/67 67/100 67/67	100% 67% 100%
(B,D,E)	(If B and D then E) (If B and E then D) (If D and E then B)	67/67 67/100 67/67	100% 67% 100%

If the minimum confidence is 70%, ten rules will be generated for these frequent itemsets. If the minimum confidence is 60%, sixteen rules will be generated

> **Tip:** Increase the minimum confidence if you want to decrease the build time for the model and generate fewer rules.

Evaluating Association Rules

Minimum support and confidence are used to influence the build of an association model. Support and confidence are also the primary metrics for evaluating the quality of the rules generated by the model. Additionally, Oracle Data Mining supports lift for association rules. These statistical measures can be used to rank the rules and hence the usefulness of the predictions.

Support

The support of a rule indicates how frequently the items in the rule occur together. For example, cereal and milk might appear together in 40% of the transactions. If so, the following two rules would each have a support of 40%.

```
cereal implies milk
milk implies cereal
```

Support is the ratio of transactions that include all the items in the antecedent and consequent to the number of total transactions.

Support can be expressed in probability notation as follows.

```
support(A implies B) = P(A, B)
```

See Also: "Frequent Itemsets" on page 10-3

Confidence

The confidence of a rule indicates the probability of both the antecedent and the consequent appearing in the same transaction. Confidence is the conditional probability of the consequent given the antecedent. For example, cereal might appear in 50 transactions; 40 of the 50 might also include milk. The rule confidence would be:

```
cereal implies milk with 80% confidence
```

Confidence is the ratio of the rule support to the number of transactions that include the antecedent.

Confidence can be expressed in probability notation as follows.

```
confidence (A implies B) = P (B/A), which is equal to P(A, B) / P(A)
```

See Also: "Confidence" on page 10-2 and "Example: Calculating Rules from Frequent Itemsets" on page 10-4

Lift

Both support and confidence must be used to determine if a rule is valid. However, there are times when both of these measures may be high, and yet still produce a rule that is not useful. For example:

```
Convenience store customers who buy orange juice also buy milk with
a 75% confidence.
The combination of milk and orange juice has a support of 30%.
```

This at first sounds like an excellent rule, and in most cases, it would be. It has high confidence and high support. However, what if convenience store customers in general buy milk 90% of the time? In that case, orange juice customers are actually *less* likely to buy milk than customers in general.

A third measure is needed to evaluate the quality of the rule. Lift indicates the strength of a rule over the random co-occurrence of the antecedent and the consequent, given their individual support. It provides information about the improvement, the increase in probability of the consequent given the antecedent. Lift is defined as follows.

```
(Rule Support) /(Support(Antecedent) * Support(Consequent))
```

This can also be defined as the confidence of the combination of items divided by the support of the consequent. So in our milk example, assuming that 40% of the customers buy orange juice, the improvement would be:

```
30% / (40% * 90%)
```

which is 0.83 – an improvement of less than 1.

Any rule with an improvement of less than 1 does not indicate a real cross-selling opportunity, no matter how high its support and confidence, because it actually offers less ability to predict a purchase than does random chance.

> **Tip:** Decrease the maximum rule length if you want to decrease the build time for the model and generate simpler rules.

> **Tip:** Increase the minimum support if you want to decrease the build time for the model and generate fewer rules.

11

Decision Tree

This chapter describes Decision Tree, one of the classification algorithms supported by Oracle Data Mining.

See Also: Chapter 5, "Classification"

This chapter contains the following topics:

- About Decision Tree
- Growing a Decision Tree
- Tuning the Decision Tree Algorithm
- Data Preparation for Decision Tree

About Decision Tree

The Decision Tree algorithm, like Naive Bayes, is based on conditional probabilities. Unlike Naive Bayes, decision trees generate **rules**. A rule is a conditional statement that can easily be understood by humans and easily used within a database to identify a set of records.

In some applications of data mining, the reason for predicting one outcome or another may not be important in evaluating the overall quality of a model. In others, the ability to explain the reason for a decision can be crucial. For example, a Marketing professional would need complete descriptions of customer segments in order to launch a successful marketing campaign. The Decision Tree algorithm is ideal for this type of application.

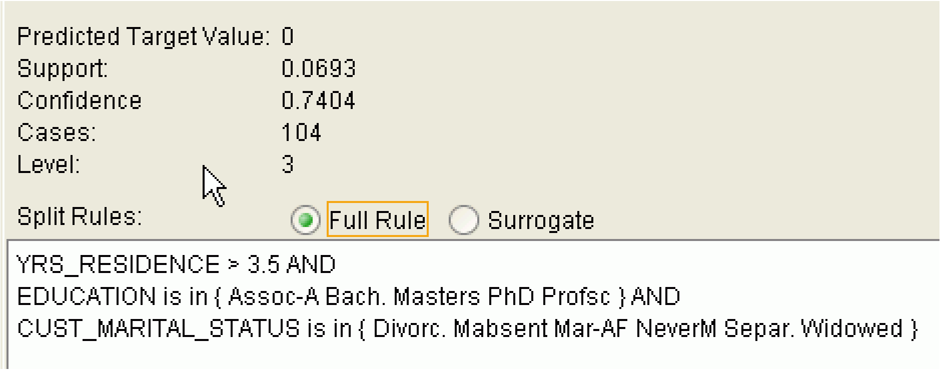
Decision Tree Rules

Oracle Data Mining supports several algorithms that provide rules. In addition to decision trees, clustering algorithms (described in Chapter 7) provide rules that describe the conditions shared by the members of a cluster, and association rules (described in Chapter 8) provide rules that describe associations between attributes.

Rules provide **model transparency**, a window on the inner workings of the model. Rules show the basis for the model's predictions. Oracle Data Mining supports a high level of model transparency. While some algorithms provide rules, *all* algorithms provide **model details**. You can examine model details to determine how the algorithm handles the attributes internally, including transformations and reverse transformations. Transparency is discussed in the context of data preparation in Chapter 19 and in the context of model building in *Oracle Data Mining Application Developer's Guide*.

Figure 11–1 shows a rule generated by a Decision Tree model. This rule comes from a decision tree that predicts the probability that customers will increase spending if given a loyalty card. A target value of 0 means not likely to increase spending; 1 means likely to increase spending.

Figure 11–1 Sample Decision Tree Rule



```
Predicted Target Value:  0
Support:                 0.0693
Confidence               0.7404
Cases:                   104
Level:                   3
```

Split Rules: ● Full Rule ○ Surrogate

```
YRS_RESIDENCE > 3.5 AND
EDUCATION is in { Assoc-A Bach. Masters PhD Profsc } AND
CUST_MARITAL_STATUS is in { Divorc. Mabsent Mar-AF NeverM Separ. Widowed }
```

The rule shown in Figure 11–1 represents the conditional statement:

```
IF
          (current residence > 3.5 and has college degree and is single)
THEN
          predicted target value = 0
```

This rule is a full rule. A surrogate rule is a related attribute that can be used at apply time if the attribute needed for the split is missing.

Confidence and Support

Confidence and support are properties of rules. These statistical measures can be used to rank the rules and hence the predictions.

Support: The number of records in the training data set that satisfy the rule.

Confidence: The likelihood of the predicted outcome, given that the rule has been satisfied.

For example, consider a list of 1000 customers (1000 cases). Out of all the customers, 100 satisfy a given rule. Of these 100, 75 are likely to increase spending, and 25 are not likely to increase spending. The **support of the rule** is 100/1000 (10%). The **confidence of the prediction** (likely to increase spending) for the cases that satisfy the rule is 75/100 (75%).

Advantages of Decision Trees

The Decision Tree algorithm produces accurate and interpretable models with relatively little user intervention. The algorithm can be used for both binary and multiclass classification problems.

The algorithm is fast, both at build time and apply time. The build process for Decision Tree is parallelized. (Scoring can be parallelized irrespective of the algorithm.)

Decision tree scoring is especially fast. The tree structure, created in the model build, is used for a series of simple tests, (typically 2-7). Each test is based on a single predictor. It is a membership test: either IN or NOT IN a list of values (categorical predictor); or LESS THAN or EQUAL TO some value (numeric predictor).

XML for Decision Tree Models

You can generate XML representing a decision tree model; the generated XML satisfies the definition specified in the Data Mining Group Predictive Model Markup Language (PMML) version 2.1 specification. The specification is available `http://www.dmg.org`.

Growing a Decision Tree

A decision tree predicts a target value by asking a sequence of questions. At a given stage in the sequence, the question that is asked depends upon the answers to the previous questions. The goal is to ask questions that, taken together, uniquely identify specific target values. Graphically, this process forms a tree structure.

Figure 11–2 Sample Decision Tree

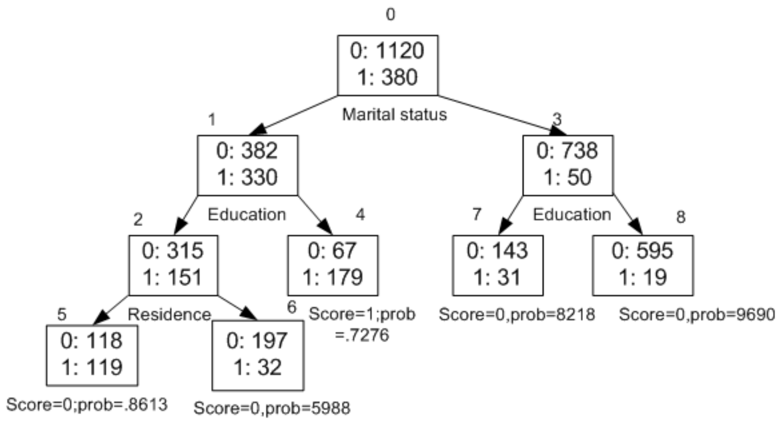


Figure 11–2 is a decision tree with nine nodes (and nine corresponding rules). The target attribute is binary: 1 if the customer will increase spending, 0 if the customer will not increase spending. The first split in the tree is based on the `CUST_MARITAL_STATUS` attribute. The root of the tree (node 0) is split into nodes 1 and 3. Married customers are in node 1; single customers are in node 3.

The rule associated with node 1 is:

```
Node 1 recordCount=712,0 Count=382, 1 Count=330
CUST_MARITAL_STATUS isIN  "Married",surrogate:HOUSEHOLD_SIZE isIn "3""4-5"
```

Node 1 has 712 records (cases). In all 712 cases, the `CUST_MARITAL_STATUS` attribute indicates that the customer is married. Of these, 382 have a target of 0 (not likely to increase spending), and 330 have a target of 1 (likely to increase spending).

Splitting

During the training process, the Decision Tree algorithm must repeatedly find the most efficient way to split a set of cases (records) into two child nodes. Oracle Data Mining offers two homogeneity metrics, **gini** and **entropy**, for calculating the splits. The default metric is gini.

Homogeneity metrics asses the quality of alternative split conditions and select the one that results in the most homogeneous child nodes. Homogeneity is also called **purity**; it refers to the degree to which the resulting child nodes are made up of cases with the same target value. The objective is to maximize the purity in the child nodes. For example, if the target can be either yes or no (will or will not increase spending), the objective is to produce nodes where most of the cases will increase spending or most of the cases will not increase spending.

Cost Matrix

All classification algorithms, including Decision Tree, support a cost-benefit matrix at apply time. You can use the same cost matrix for building and scoring a Decision Tree model, or you can specify a different cost/benefit matrix for scoring.

See "Costs" on page 5-5 and "Priors" on page 5-7.

Preventing Over-Fitting

In principle, Decision Tree algorithms can grow each branch of the tree just deeply enough to perfectly classify the training examples. While this is sometimes a reasonable strategy, in fact it can lead to difficulties when there is noise in the data, or when the number of training examples is too small to produce a representative sample of the true target function. In either of these cases, this simple algorithm can produce trees that over-fit the training examples. Over-fit is a condition where a model is able to accurately predict the data used to create the model, but does poorly on new data presented to it.

To prevent over-fitting, Oracle Data Mining supports automatic **pruning** and configurable **limit conditions** that control tree growth. Limit conditions prevent further splits once the conditions have been satisfied. Pruning removes branches that have insignificant predictive power.

Tuning the Decision Tree Algorithm

The Decision Tree algorithm is implemented with reasonable defaults for splitting and termination criteria. However several build settings are available for fine tuning.

You can specify a homogeneity metric for finding the optimal split condition for a tree. The default metric is gini. The entropy metric is also available.

Settings for controlling the growth of the tree are also available. You can specify the maximum depth of the tree, the minimum number of cases required in a child node, the minimum number of cases required in a node in order for a further split to be possible, the minimum number of cases in a child node, and the minimum number of cases required in a node in order for a further split to be possible.

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for details

Data Preparation for Decision Tree

The Decision Tree algorithm manages its own data preparation internally. It does not require pretreatment of the data. Decision Tree is not affected by Automatic Data Preparation.

Decision Tree interprets missing values as missing at random. The algorithm does not support nested tables and thus does not support sparse data.

See Also:

Chapter 19, "Automatic and Embedded Data Preparation"

Oracle Data Mining Application Developer's Guide for information about nested data and missing value treatment

12

Generalized Linear Models

This chapter describes Generalized Linear Models (GLM), a statistical technique for linear modeling. Oracle Data Mining supports GLM for both regression and classification mining functions.

See Also: Chapter 4, "Regression" and Chapter 5, "Classification"

This chapter includes the following topics:

- About Generalized Linear Models
- Tuning and Diagnostics for GLM
- Data Preparation for GLM
- Linear Regression
- Logistic Regression

About Generalized Linear Models

Generalized Linear Models (GLM) include and extend the class of linear models described in "Linear Regression" on page 4-2.

Linear models make a set of restrictive assumptions, most importantly, that the target (dependent variable *y*) is normally distributed conditioned on the value of predictors with a constant variance regardless of the predicted response value. The advantage of linear models and their restrictions include computational simplicity, an interpretable model form, and the ability to compute certain diagnostic information about the quality of the fit.

Generalized linear models relax these restrictions, which are often violated in practice. For example, binary (yes/no or 0/1) responses do not have same variance across classes. Furthermore, the sum of terms in a linear model typically can have very large ranges encompassing very negative and very positive values. For the binary response example, we would like the response to be a probability in the range [0,1].

Generalized linear models accommodate responses that violate the linear model assumptions through two mechanisms: a link function and a variance function. The link function transforms the target range to potentially -infinity to +infinity so that the simple form of linear models can be maintained. The variance function expresses the variance as a function of the predicted response, thereby accommodating responses with non-constant variances (such as the binary responses).

Oracle Data Mining includes two of the most popular members of the GLM family of models with their most popular link and variance functions:

- **Linear regression** with the identity link and variance function equal to the constant 1 (constant variance over the range of response values). See "Linear Regression" on page 12-6.
- **Logistic regression** with the logit link and binomial variance functions. See "Logistic Regression" on page 12-8.

GLM in Oracle Data Mining

GLM is a **parametric** modeling technique. Parametric models make assumptions about the distribution of the data. When the assumptions are met, parametric models can be more efficient than non-parametric models.

The challenge in developing models of this type involves assessing the extent to which the assumptions are met. For this reason, quality diagnostics are key to developing quality parametric models.

Interpretability and Transparency

Oracle Data Mining GLM models are easy to interpret. Each model build generates many statistics and diagnostics. Transparency is also a key feature: model details describe key characteristics of the coefficients, and global details provide high-level statistics.

See Also:

"Tuning and Diagnostics for GLM" on page 12-3

"Transparency" on page 19-8

Wide Data

Oracle Data Mining GLM is uniquely suited for handling wide data. The algorithm can build and score quality models that use a virtually limitless number of predictors (attributes). The only constraints are those imposed by system resources.

Confidence Bounds

GLM has the ability to predict confidence bounds. In addition to predicting a best estimate and a probability (classification only) for each row, GLM identifies an interval wherein the prediction (regression) or probability (classification) will lie. The width of the interval depends upon the precision of the model and a user-specified confidence level.

The confidence level is a measure of how sure the model is that the true value will lie within a confidence interval computed by the model. A popular choice for confidence level is 95%. For example, a model might predict that an employee's income is $125K, and that you can be 95% sure that it lies between $90K and $160K. Oracle Data Mining supports 95% confidence by default, but that value is configurable.

Note: Confidence bounds are returned with the coefficient statistics. You can also use the `PREDICTION_BOUNDS` SQL function to obtain the confidence bounds of a model prediction. See *Oracle Database SQL Language Reference*.

Ridge Regression

The best regression models are those in which the predictors correlate highly with the target, but there is very little correlation between the predictors themselves.

Multicollinearity is the term used to describe multivariate regression with correlated predictors.

Ridge regression is a technique that compensates for multicollinearity. Oracle Data Mining supports ridge regression for both regression and classification mining functions. The algorithm automatically uses ridge if it detects singularity (exact multicollinearity) in the data.

Information about singularity is returned in the global model details. See "Global Model Statistics for Linear Regression" on page 12-7 and "Global Model Statistics for Logistic Regression" on page 12-8.

Build Settings for Ridge Regression

You can choose to explicitly enable ridge regression by specifying the GLMS_RIDGE_REGRESSION setting. If you explicitly enable ridge, you can use the system-generated ridge parameter or you can supply your own. If ridge is used automatically, the ridge parameter is also calculated automatically.

The build settings for ridge are summarized as follows:

- GLMS_RIDGE_REGRESSION — Whether or not to override the automatic choice made by the algorithm regarding ridge regression
- GLMS_RIDGE_VALUE — The value of the ridge parameter, used only if you specifically enable ridge regression.
- GLMS_VIF_FOR_RIDGE — Whether or not to produce Variance Inflation Factor (VIF) statistics when ridge is being used for linear regression.

See Also: *Oracle Database PL/SQL Packages and Types Reference*

Ridge and Confidence Bounds

Confidence bounds are not supported by models built with ridge regression. See "Confidence Bounds" on page 12-2.

Ridge and Variance Inflation Factor for Linear Regression

GLM produces Variance Inflation Factor (VIF) statistics for linear regression models, unless they were built with ridge. You can explicitly request VIF with ridge by specifying the GLMS_VIF_FOR_RIDGE setting. The algorithm will produce VIF with ridge only if enough system resources are available.

Ridge and Data Preparation

When ridge regression is enabled, different data preparation is likely to produce different results in terms of model coefficients and diagnostics. Oracle recommends that you enable Automatic Data Preparation for GLM models, especially when ridge regression is being used. See "Data Preparation for GLM" on page 12-5.

Tuning and Diagnostics for GLM

The process of developing a GLM model typically involves a number of model builds. Each build generates many statistics that you can evaluate to determine the quality of your model. Depending on these diagnostics, you may want to try changing the model settings or making other modifications.

Build Settings

You can use build settings to specify:

- **Coefficient confidence** — The GLMS_CONF_LEVEL setting indicates the degree of certainty that the true coefficient lies within the confidence bounds computed by the model. The default confidence is.95.
- **Row weights** — The ODMS_ROW_WEIGHT_COLUMN_NAME setting identifies a column that contains a weighting factor for the rows.
- **Row diagnostics** — The GLMS_DIAGNOSTICS_TABLE_NAME setting identifies a table to contain row-level diagnostics.

Additional build settings are available to:

- Control the use of ridge regression, as described in "Ridge Regression" on page 12-2.
- Specify the handling of missing values in the training data, as described in "Data Preparation for GLM" on page 12-5.
- Specify the target value to be used as a reference in a logistic regression model, as described in "Logistic Regression" on page 12-8.

> **See:** *Oracle Database PL/SQL Packages and Types Reference* for details about GLM settings

Diagnostics

GLM models generate many metrics to help you evaluate the quality of the model.

Coefficient Statistics

The same set of statistics is returned for both linear and logistic regression, but statistics that do not apply to the mining function are returned as NULL. The coefficient statistics are described in "Coefficient Statistics for Linear Regression" on page 12-6 and "Coefficient Statistics for Logistic Regression" on page 12-8.

Coefficient statistics are returned by the GET_MODEL_DETAILS_GLM function in DBMS_DATA_MINING.

Global Model Statistics

Separate high-level statistics describing the model as a whole, are returned for linear and logistic regression. When ridge regression is enabled, fewer global details are returned (See "Ridge Regression" on page 12-2). The global model statistics are described in "Global Model Statistics for Linear Regression" on page 12-7 and "Global Model Statistics for Logistic Regression" on page 12-8.

Global statistics are returned by the GET_MODEL_DETAILS_GLOBAL function in DBMS_DATA_MINING.

Row Diagnostics

You can configure GLM models to generate per-row statistics by specifying the name of a diagnostics table in the build setting GLMS_DIAGNOSTICS_TABLE_NAME. The row diagnostics are described in "Row Diagnostics for Linear Regression" on page 12-7 and "Row Diagnostics for Logistic Regression" on page 12-9.

GLM requires a case ID to generate row diagnostics. If you provide the name of a diagnostic table but the data does not include a case ID column, an exception is raised.

Data Preparation for GLM

Automatic Data Preparation (ADP) implements suitable data transformations for both linear and logistic regression.

Note: Oracle recommends that you use Automatic Data Preparation with GLM.

See Also: Chapter 19, "Automatic and Embedded Data Preparation"

Data Preparation for Linear Regression

When ADP is enabled, the build data are standardized using a widely used correlation transformation (Netter, et. al, 1990). The data are first centered by subtracting the attribute means from the attribute values for each observation. Then the data are scaled by dividing each attribute value in an observation by the square root of the sum of squares per attribute across all observations. This transformation is applied to both numeric and categorical attributes.

Prior to standardization, categorical attributes are exploded into N-1 columns where N is the attribute cardinality. The most frequent value (mode) is omitted during the explosion transformation. In the case of highest frequency ties, the attribute values are sorted alpha-numerically in ascending order, and the first value on the list is omitted during the explosion. This explosion transformation occurs whether or not ADP is enabled.

In the case of high cardinality categorical attributes, the described transformations (explosion followed by standardization) can increase the build data size because the resulting data representation is dense. To reduce memory, disk space, and processing requirements, an alternative approach needs to be used. For large datasets where the estimated internal dense representation would require more than 1Gb of disk space, categorical attributes are not standardized. Under these circumstances, the VIF statistic should be used with caution.

Reference: Neter, J., Wasserman, W., and Kutner, M.H., "Applied Statistical Models", Richard D. Irwin, Inc., Burr Ridge, IL, 1990.

See Also:

- "Ridge and Data Preparation" on page 12-3
- Chapter 19, "Automatic and Embedded Data Preparation"

Data Preparation for Logistic Regression

Categorical attributes are exploded into *N*-1 columns where *N* is the attribute cardinality. The most frequent value (mode) is omitted during the explosion transformation. In the case of highest frequency ties, the attribute values are sorted alpha-numerically in ascending order and the first value on the list is omitted during the explosion. This explosion transformation occurs whether or not ADP is enabled.

When ADP is enabled, numerical attributes are standardized by scaling the attribute values by a measure of attribute variability. This measure of variability is computed as the standard deviation per attribute with respect to the origin (not the mean) (Marquardt, 1980).

Reference: Marquardt, D.W., "A Critique of Some Ridge Regression Methods: Comment", Journal of the American Statistical Association, Vol. 75, No. 369 , 1980, pp. 87-91.

Missing Values

When building or applying a model, Oracle Data Mining automatically replaces missing values of numerical attributes with the mean and missing values of categorical attributes with the mode.

You can configure a GLM model to override the default treatment of missing values. With the `ODMS_MISSING_VALUE_TREATMENT` setting, you can cause the algorithm to delete rows in the training data that have missing values instead of replacing them with the mean or the mode. However, when the model is applied, Oracle Data Mining will perform the usual mean/mode missing value replacement. As a result, statistics generated from scoring may not match the statistics generated from building the model.

If you want to delete rows with missing values in the scoring the model, you must perform the transformation explicitly. To make build and apply statistics match, you must remove the rows with NULLs from the scoring data before performing the apply operation. You can do this by creating a view.

```
CREATE VIEW viewname AS SELECT * from tablename
     WHERE column_name1 is NOT NULL
     AND   column_name2 is NOT NULL
     AND   column_name3 is NOT NULL .....
```

Note: In Oracle Data Mining, missing values in nested data indicate sparsity, not values missing at random.

The value `ODMS_MISSING_VALUE_DELETE_ROW` is only valid for tables without nested columns. If this value is used with nested data, an exception is raised.

Linear Regression

Linear regression is the GLM regression algorithm supported by Oracle Data Mining. The algorithm assumes no target transformation and constant variance over the range of target values.

Coefficient Statistics for Linear Regression

GLM regression models generate the following coefficient statistics:

- Linear coefficient estimate
- Standard error of the coefficient estimate
- t-value of the coefficient estimate
- Probability of the t-value
- Variance Inflation Factor (VIF)
- Standardized estimate of the coefficient
- Lower and upper confidence bounds of the coefficient

Global Model Statistics for Linear Regression

GLM regression models generate the following statistics that describe the model as a whole:

- Model degrees of freedom
- Model sum of squares
- Model mean square
- Model *F* statistic
- Model *F* value probability
- Error degrees of freedom
- Error sum of squares
- Error mean square
- Corrected total degrees of freedom
- Corrected total sum of squares
- Root mean square error
- Dependent mean
- Coefficient of variation
- R-Square
- Adjusted R-Square
- Akaike's information criterion
- Schwarz's Baysian information criterion
- Estimated mean square error of the prediction
- Hocking Sp statistic
- JP statistic (the final prediction error)
- Number of parameters (the number of coefficients, including the intercept)
- Number of rows
- Whether or not the model converged
- Whether or not a covariance matrix was computed

Row Diagnostics for Linear Regression

For linear regression, the diagnostics table has the columns described in Table 12–1. All the columns are `NUMBER`, except the `CASE_ID` column, which preserves the type from the training data.

Table 12–1 Diagnostics Table for GLM Regression Models

Column	Description
`CASE_ID`	Value of the case ID column
`TARGET_VALUE`	Value of the target column
`PREDICTED_VALUE`	Value predicted by the model for the target
`HAT`	Value of the diagonal element of the hat matrix

Table 12–1 (Cont.) Diagnostics Table for GLM Regression Models

Column	Description
RESIDUAL	Measure of error
STD_ERR_RESIDUAL	Standard error of the residual
STUDENTIZED_RESIDUAL	Studentized residual
PRED_RES	Predicted residual
COOKS_D	Cook's D influence statistic

Logistic Regression

Binary logistic regression is the GLM classification algorithm supported by Oracle Data Mining. The algorithm uses the logit link function and the binomial variance function.

Reference Class

You can use the build setting `GLMS_REFERENCE_CLASS_NAME` to specify the target value to be used as a reference in a binary logistic regression model. Probabilities will be produced for the other (non-reference) class. By default, the algorithm chooses the value with the highest prevalence. If there are ties, the attributes are sorted alpha-numerically in ascending order.

Class Weights

You can use the build setting `CLAS_WEIGHTS_TABLE_NAME` to specify the name of a class weights table. Class weights influence the weighting of target classes during the model build.

Coefficient Statistics for Logistic Regression

GLM classification models generate the following coefficient statistics:

- Name of the predictor
- Coefficient estimate
- Standard error of the coefficient estimate
- Wald chi-square value of the coefficient estimate
- Probability of the Wald chi-square value
- Standardized estimate of the coefficient
- Lower and upper confidence bounds of the coefficient
- Exponentiated coefficient
- Exponentiated coefficient for the upper and lower confidence bounds of the coefficient

Global Model Statistics for Logistic Regression

GLM classification models generate the following statistics that describe the model as a whole:

- Akaike's criterion for the fit of the intercept only model

- Akaike's criterion for the fit of the intercept and the covariates (predictors) model
- Schwarz's criterion for the fit of the intercept only model
- Schwarz's criterion for the fit of the intercept and the covariates (predictors) model
- -2 log likelihood of the intercept only model
- -2 log likelihood of the model
- Likelihood ratio degrees of freedom
- Likelihood ratio chi-square probability value
- Pseudo R-square Cox an Snell
- Pseudo R-square Nagelkerke
- Dependent mean
- Percent of correct predictions
- Percent of incorrect predictions
- Percent of ties (probability for two cases is the same)
- Number of parameters (the number of coefficients, including the intercept)
- Number of rows
- Whether or not the model converged
- Whether or not a covariance matrix was computed.

Row Diagnostics for Logistic Regression

For logistic regression, the diagnostics table has the columns described in Table 12–2. All the columns are NUMBER, except the CASE_ID and TARGET_VALUE columns, which preserve the type from the training data.

Table 12–2 Row Diagnostics Table for Logistic Regression

Column	Description
CASE_ID	Value of the case ID column
TARGET_VALUE	Value of the target value
TARGET_VALUE_PROB	Probability associated with the target value
HAT	Value of the diagonal element of the hat matrix
WORKING_RESIDUAL	Residual with respect to the adjusted dependent variable
PEARSON_RESIDUAL	The raw residual scaled by the estimated standard deviation of the target
DEVIANCE_RESIDUAL	Contribution to the overall goodness of fit of the model
C	Confidence interval displacement diagnostic
CBAR	Confidence interval displacement diagnostic
DIFDEV	Change in the deviance due to deleting an individual observation
DIFCHISQ	Change in the Pearson chi-square

13

k-Means

This chapter describes the enhanced k-Means clustering algorithm supported by Oracle Data Mining.

> **See Also:** Chapter 7, "Clustering"

This chapter includes the following topics:

- About k-Means
- Tuning the k-Means Algorithm
- Data Preparation for k-Means

About *k*-Means

The *k*-Means algorithm is a distance-based clustering algorithm that partitions the data into a specified number of clusters.

Distance-based algorithms rely on a distance function to measure the similarity between cases. Cases are assigned to the nearest cluster according to the distance function used.

Oracle Data Mining Enhanced *k*-Means

Oracle Data Mining implements an enhanced version of the *k*-Means algorithm with the following features:

- **Distance function** — The algorithm supports Euclidean, Cosine, and Fast Cosine distance functions. The default is Euclidean.
- **Hierarchical model build** —The algorithm builds a model in a top-down hierarchical manner, using binary splits and refinement of all nodes at the end. In this sense, the algorithm is similar to the bisecting *k*-Means algorithm. The centroids of the inner nodes in the hierarchy are updated to reflect changes as the tree evolves. The whole tree is returned.
- **Tree growth** — The algorithm uses a specified split criterion to grow the tree one node at a time until a specified maximum number of clusters is reached, or until the number of distinct cases is reached. The split criterion may be the variance or the cluster size. By default the split criterion is the variance.
- **Cluster properties** — For each cluster, the algorithm returns the centroid, a histogram for each attribute, and a rule describing the hyperbox that encloses the majority of the data assigned to the cluster. The centroid reports the mode for categorical attributes and the mean and variance for numerical attributes.

This approach to *k*-Means avoids the need for building multiple *k*-Means models and provides clustering results that are consistently superior to the traditional *k*-Means.

Centroid

The **centroid** represents the most typical case in a cluster. For example, in a data set of customer ages and incomes, the centroid of each cluster would be a customer of average age and average income in that cluster. The centroid is a prototype. It does not necessarily describe any given case assigned to the cluster.

The attribute values for the centroid are the mean of the numerical attributes and the mode of the categorical attributes.

Scoring

The clusters discovered by *k*-Means are used to generate a Bayesian probability model that can be used to score new data.

Tuning the *k*-Means Algorithm

The Oracle Data Mining enhanced *k*-Means algorithm supports several build-time settings. All the settings have default values. There is no reason to override the defaults unless you want to influence the behavior of the algorithm in some specific way.

You can configure *k*-Means by specifying any of the following:

- Number of clusters
- Growth factor for memory allocated to hold clusters
- Convergence tolerance
- Distance Function. The default distance function is Euclidean.
- Split criterion. The default criterion is the variance.
- Number of iterations for building the cluster tree.
- The fraction of attribute values that must be non-null in order for an attribute to be included in the rule description for a cluster. Setting the parameter value too high in data with missing values can result in very short or even empty rules.
- Number of histogram bins. The bin boundaries for each attribute are computed globally on the entire training data set. The binning method is equi-width. All attributes have the same number of bins with the exception of attributes with a single value that have only one bin.

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for details about the build settings for *k*-Means

Data Preparation for *k*-Means

Normalization is typically required by the *k*-Means algorithm. Automatic Data Preparation performs outlier-sensitive normalization for *k*-Means. If you do not use ADP, you should normalize numeric attributes before creating or applying the model.

When there are missing values in columns with simple data types (not nested), *k*-Means interprets them as missing at random. The algorithm replaces missing categorical values with the mode and missing numerical values with the mean.

When there are missing values in nested columns, *k*-Means interprets them as sparse. The algorithm replaces sparse numerical data with zeros and sparse categorical data with zero vectors.

See Also:

Oracle Database PL/SQL Packages and Types Reference for details about normalization routines

Chapter 19 for information about automatic and embedded data transformation in Oracle Data Mining

Oracle Data Mining Application Developer's Guide for information about support for nested columns and missing data in Oracle Data Mining

14

Minimum Description Length

This chapter describes Minimum Description Length, the supervised technique used by Oracle Data Mining for calculating attribute importance.

See Also: Chapter 9, "Feature Selection and Extraction"

This chapter includes the following topics:

- About MDL
- Data Preparation for MDL

About MDL

Minimum Description Length (MDL) is an information theoretic model selection principle. It is an important concept in information theory (the study of the quantification of information) and in learning theory (the study of the capacity for generalization based on empirical data).

MDL assumes that the simplest, most compact representation of the data is the best and most probable explanation of the data. The MDL principle is used to build Oracle Data Mining attribute importance models.

Compression and Entropy

Data compression is the process of encoding information using fewer **bits** than the original representation would use. The MDL Principle is based on the notion that the shortest description of the data is the most probable. In typical instantiations of this principle, a model is used to compress the data by reducing the uncertainty (entropy) as discussed below. The description of the data includes a description of the model and the data as described by the model.

Entropy is a measure of uncertainty. It quantifies the uncertainty in a random variable as the information required to specify its value. **Information** in this sense is defined as the number of yes/no questions known as **bits** (encoded as 0 or 1) that must be answered for a complete specification. Thus, the information depends upon the number of values that variable can assume.

For example, if the variable represents the sex of an individual, then the number of possible values is two: female and male. If the variable represents the salary of individuals expressed in whole dollar amounts, it may have values in the range $0-$10B, or billions of unique values. Clearly it will take more information to specify an exact salary than to specify an individual's sex.

Values of a Random Variable: Statistical Distribution

Information (the number of bits) depends on the statistical distribution of the values of the variable as well as the number of values of the variable. If we are judicious in the choice of Yes/No questions, the amount of information for salary specification may not be as much as it first appears. Most people do not have billion dollar salaries. If most people have salaries in the range $32000-$64000, then most of the time, we would require only 15 questions to discover their salary, rather than the 30 required, if every salary from $0-$1000000000 were equally likely. In the former example, if the persons were known to be pregnant, then their sex is known to be female. There is no uncertainty, no Yes/No questions need be asked. The entropy is 0.

Values of a Random Variable: Significant Predictors

Suppose that for some random variable there is a predictor that when its values are known reduces the uncertainty of the random variable. For example, knowing whether a person is pregnant or not, reduces the uncertainty of the random variable sex-of-individual. This predictor seems like a valuable feature to include in a model. How about name? Imagine that if you knew the name of the person, you would also know the person's sex. If so, the name predictor would seemingly reduce the uncertainty to zero. However, if names are unique, then what was gained? Is the person named Sally? Is the person named George?... We would have as many Yes/No predictors in the name model as there are people. Therefore, specifying the name model would require as many bits as specifying the sex of each person.

Total Entropy

For a random variable, X, the **total entropy** is defined as minus the Probability(X) multiplied by the log to the base 2 of the Probability(X). This can be shown to be the variable's most efficient encoding.

Model Size

MDL takes into consideration the size of the model as well as the reduction in uncertainty due to using the model. Both model size and entropy are measured in bits. For our purposes, both numeric and categorical predictors are binned. Thus the size of each single predictor model is the number of predictor bins. The uncertainty is reduced to the within-bin target distribution.

Model Selection

MDL considers each attribute as a simple predictive model of the target class. **Model selection** refers to the process of comparing and ranking the single-predictor models.

MDL uses a communication model for solving the model selection problem. In the communication model there is a sender, a receiver, and data to be transmitted.

These single predictor models are compared and ranked with respect to the MDL metric, which is the relative compression in bits. MDL penalizes model complexity to avoid over-fit. It is a principled approach that takes into account the complexity of the predictors (as models) to make the comparisons fair.

The MDL Metric

Attribute importance uses a two-part code as the metric for transmitting each unit of data. The first part (preamble) transmits the model. The parameters of the model are the target probabilities associated with each value of the prediction.

For a target with *j* values and a predictor with *k* values, n_i (*i*= 1,..., k) rows per value, there are C_i, the combination of *j*-1 things taken n_i-1 at a time possible conditional probabilities. The size of the preamble in bits can be shown to be Sum($\log_2(C_i)$), where the sum is taken over *k*. Computations like this represent the penalties associated with each single prediction model. The second part of the code transmits the target values using the model.

It is well known that the most compact encoding of a sequence is the encoding that best matches the probability of the symbols (target class values). Thus, the model that assigns the highest probability to the sequence has the smallest target class value transmission cost. In bits this is the Sum($\log_2(p_i)$), where the p_i are the predicted probabilities for row $_i$ associated with the model.

The predictor rank is the position in the list of associated description lengths, smallest first.

Data Preparation for MDL

Automatic Data Preparation performs supervised binning for MDL. Supervised binning uses decision trees to create the optimal bin boundaries. Both categorical and numerical attributes are binned.

MDL handles missing values naturally as missing at random. The algorithm replaces sparse numerical data with zeros and sparse categorical data with zero vectors. Missing values in nested columns are interpreted as sparse. Missing values in columns with simple data types are interpreted as missing at random.

If you choose to manage your own data preparation, keep in mind that MDL usually benefits from binning. However, the discriminating power of an attribute importance model can be significantly reduced when there are outliers in the data and external equal-width binning is used. This technique can cause most of the data to concentrate in a few bins (a single bin in extreme cases). In this case, quantile binning is a better solution.

See Also:

Chapter 19, "Automatic and Embedded Data Preparation"

Oracle Data Mining Application Developer's Guide for information about nested data and missing values

15

Naive Bayes

This chapter describes Naive Bayes, one of the classification algorithms supported by Oracle Data Mining.

See Also: Chapter 5, "Classification"

This chapter contains the following topics:

- About Naive Bayes
- Tuning a Naive Bayes Model
- Data Preparation for Naive Bayes

About Naive Bayes

The Naive Bayes algorithm is based on conditional probabilities. It uses Bayes' Theorem, a formula that calculates a probability by counting the frequency of values and combinations of values in the historical data.

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. If `B` represents the dependent event and `A` represents the prior event, Bayes' theorem can be stated as follows.

Bayes' Theorem: Prob(B given A) = Prob(A and B)/Prob(A)

To calculate the probability of `B` given `A`, the algorithm counts the number of cases where `A` and `B` occur together and divides it by the number of cases where `A` occurs alone.

Example 15–1 Use Bayes' Theorem to Predict an Increase in Spending

Suppose you want to determine the likelihood that a customer under 21 will increase spending. In this case, the prior condition (`A`) would be "under 21," and the dependent condition (`B`) would be "increase spending."

If there are 100 customers in the training data and 25 of them are customers under 21 who have increased spending, then:

Prob(A and B) = 25%

If 75 of the 100 customers are under 21, then:

Prob(A) = 75%

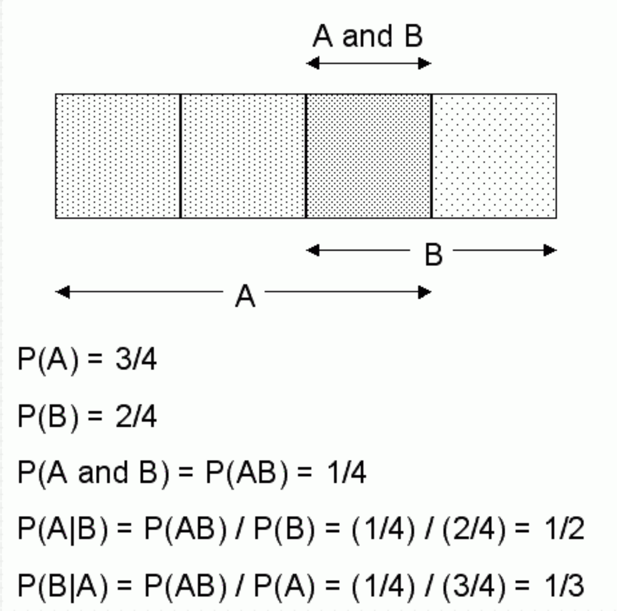
Bayes' Theorem would predict that 33% of customers under 21 are likely to increase spending (25/75).

The cases where both conditions occur together are referred to as **pairwise**. In Example 15–1, 25% of all cases are pairwise.

The cases where only the prior event occurs are referred to as **singleton**. In Example 15–1, 75% of all cases are singleton.

A visual representation of the conditional relationships used in Bayes' Theorem is shown in Figure 15–1.

Figure 15–1 Conditional Probabilities in Bayes' Theorem



$P(A) = 3/4$

$P(B) = 2/4$

$P(A \text{ and } B) = P(AB) = 1/4$

$P(A|B) = P(AB) / P(B) = (1/4) / (2/4) = 1/2$

$P(B|A) = P(AB) / P(A) = (1/4) / (3/4) = 1/3$

For purposes of illustration, Example 15–1 and Figure 15–1 show a dependent event based on a single independent event. In reality, the Naive Bayes algorithm must usually take many independent events into account. In Example 15–1, factors such as income, education, gender, and store location might be considered in addition to age.

Naive Bayes makes the assumption that each predictor is conditionally independent of the others. For a given target value, the distribution of each predictor is independent of the other predictors. In practice, this assumption of independence, even when violated, does not degrade the model's predictive accuracy significantly, and makes the difference between a fast, computationally feasible algorithm and an intractable one.

Sometimes the distribution of a given predictor is clearly not representative of the larger population. For example, there might be only a few customers under 21 in the training data, but in fact there are many customers in this age group in the wider customer base. To compensate for this, you can specify **prior probabilities** when training the model. See "Priors" on page 5-7.

Advantages of Naive Bayes

The Naive Bayes algorithm affords fast, highly scalable model building and scoring. It scales linearly with the number of predictors and rows. The build process for Naive Bayes is parallelized. (Scoring can be parallelized irrespective of the algorithm.)

Naive Bayes can be used for both binary and multiclass classification problems.

Tuning a Naive Bayes Model

Naive Bayes calculates a probability by dividing the percentage of pairwise occurrences by the percentage of singleton occurrences. If these percentages are very small for a given predictor, they probably will not contribute to the effectiveness of the model. Occurrences below a certain threshold can usually be ignored.

Two build settings are available for adjusting the probability thresholds. You can specify:

- the minimum percentage of pairwise occurrences required for including a predictor in the model
- the minimum percentage of singleton occurrences required for including a predictor in the model

The default thresholds work well for most models, so you will not generally need to adjust these settings.

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for details about algorithm settings for Naive Bayes

Data Preparation for Naive Bayes

Automatic Data Preparation performs supervised binning for Naive Bayes. Supervised binning uses decision trees to create the optimal bin boundaries. Both categorical and numerical attributes are binned.

Naive Bayes handles missing values naturally as missing at random. The algorithm replaces sparse numerical data with zeros and sparse categorical data with zero vectors. Missing values in nested columns are interpreted as sparse. Missing values in columns with simple data types are interpreted as missing at random.

If you choose to manage your own data preparation, keep in mind that Naive Bayes usually requires binning. Naive Bayes relies on counting techniques to calculate probabilities. Columns should be binned to reduce the cardinality as appropriate. Numerical data can be binned into ranges of values (for example, low, medium, and high), and categorical data can be binned into meta-classes (for example, regions instead of cities). Equi-width binning is not recommended, since outliers will cause most of the data to concentrate in a few bins, sometimes a single bin. As a result, the discriminating power of the algorithms will be significantly reduced

> **See Also:**
>
> Chapter 19, "Automatic and Embedded Data Preparation"
>
> *Oracle Data Mining Application Developer's Guide* for information about nested columns and missing data

16

Non-Negative Matrix Factorization

This chapter describes Non-Negative Matrix Factorization (NMF), the unsupervised algorithm used by Oracle Data Mining for feature extraction.

See Also: Chapter 9, "Feature Selection and Extraction"

Note: Non-Negative Matrix Factorization (NMF) is described in the paper "Learning the Parts of Objects by Non-Negative Matrix Factorization" by D. D. Lee and H. S. Seung in *Nature* (401, pages 788-791, 1999).

This chapter contains the following topics:

- About NMF
- Tuning the NMF Algorithm
- Data Preparation for NMF

About NMF

Non-Negative Matrix Factorization is a state of the art feature extraction algorithm. NMF is useful when there are many attributes and the attributes are ambiguous or have weak predictability. By combining attributes, NMF can produce meaningful patterns, topics, or themes.

Each feature created by NMF is a linear combination of the original attribute set. Each feature has a set of coefficients, which are a measure of the weight of each attribute on the feature. There is a separate coefficient for each numerical attribute and for each distinct value of each categorical attribute. The coefficients are all non-negative.

Matrix Factorization

Non-Negative Matrix Factorization uses techniques from multivariate analysis and linear algebra. It decomposes the data as a matrix M into the product of two lower ranking matrices W and H. The sub-matrix W contains the NMF basis; the sub-matrix H contains the associated coefficients (weights).

The algorithm iteratively modifies of the values of W and H so that their product approaches M. The technique preserves much of the structure of the original data and guarantees that both basis and weights are non-negative. The algorithm terminates when the approximation error converges or a specified number of iterations is reached.

The NMF algorithm must be initialized with a seed to indicate the starting point for the iterations. Because of the high dimensionality of the processing space and the fact that there is no global minimization algorithm, the appropriate initialization can be critical in obtaining meaningful results. Oracle Data Mining uses a random seed that initializes the values of W and H based on a uniform distribution. This approach works well in most cases.

Scoring with NMF

NMF can be used as a dimensionality reduction pre-processing step in classification, regression, clustering, and other mining tasks. Scoring an NMF model produces data projections in the new feature space. The magnitude of a projection indicates how strongly a record maps to a feature.

Text Mining with NMF

NMF is especially well-suited for text mining. In a text document, the same word can occur in different places with different meanings. For example, "hike" can be applied to the outdoors or to interest rates. By combining attributes, NMF introduces context, which is essential for explanatory power:

"hike" + "mountain" -> "outdoor sports"
"hike" + "interest" -> "interest rates"

See Also: "Text Feature Extraction" on page 20-4

Tuning the NMF Algorithm

Oracle Data Mining supports five configurable parameters for NMF. All of them have default values which will be appropriate for most applications of the algorithm. The NMF settings are:

- Number of features. By default, the number of features is determined by the algorithm.
- Convergence tolerance. The default is .05.
- Number of iterations. The default is 50.
- Random seed. The default is -1.
- Non-negative scoring. You can specify whether negative numbers should be allowed in scoring results. By default they are allowed.

See Also: *Oracle Database PL/SQL Packages and Types Reference* for information about model settings

Data Preparation for NMF

Automatic Data Preparation normalizes numerical attributes for NMF.

When there are missing values in columns with simple data types (not nested), NMF interprets them as missing at random. The algorithm replaces missing categorical values with the mode and missing numerical values with the mean.

When there are missing values in nested columns, NMF interprets them as sparse. The algorithm replaces sparse numerical data with zeros and sparse categorical data with zero vectors.

If you choose to manage your own data preparation, keep in mind that outliers can significantly impact NMF. Use a clipping transformation before binning or normalizing. NMF typically benefits from normalization. However, outliers with min-max normalization cause poor matrix factorization. To improve the matrix factorization, you need to decrease the error tolerance. This in turn leads to longer build times.

See Also:

Chapter 19, "Automatic and Embedded Data Preparation"

Oracle Data Mining Application Developer's Guide for information about nested columns and missing data

17

O-Cluster

This chapter describes Orthogonal Partitioning Clustering (O-Cluster), an Oracle-proprietary clustering algorithm.

See Also: Chapter 7, "Clustering"

Reference:

Campos, M.M., Milenova, B.L., "Clustering Large Databases with Numeric and Nominal Values Using Orthogonal Projections", Oracle Data Mining Technologies, Oracle Corporation.

`http://www.oracle.com/technology/products/bi/odm/`

This chapter contains the following topics:

- About O-Cluster
- Tuning the O-Cluster Algorithm
- Data Preparation for O-Cluster

About O-Cluster

O-Cluster is a fast, scalable grid-based clustering algorithm well-suited for mining large, high-dimensional data sets. The algorithm can produce high quality clusters without relying on user-defined parameters.

The objective of O-Cluster is to identify areas of high density in the data and separate the dense areas into clusters. It uses axis-parallel uni-dimensional (orthogonal) data projections to identify the areas of density. The algorithm looks for splitting points that result in distinct clusters that do not overlap and are balanced in size.

O-Cluster operates recursively by creating a binary tree hierarchy. The number of leaf clusters is determined automatically. The algorithm can be configured to limit the maximum number of clusters.

Partitioning Strategy

Partitioning strategy refers to the process of discovering areas of density in the attribute histograms. The process differs for numerical and categorical data. When both are present in the data, the algorithm performs the searches separately and then compares the results.

In choosing a partition, the algorithm balances two objectives: finding well separated clusters, and creating clusters that are balanced in size. The following paragraphs detail how partitions for numerical and categorical attributes are identified.

Partitioning Numerical Attributes

To find the best valid cutting plane, O-Cluster searches the attribute histograms for bins of low density (valleys) between bins of high density (peaks). O-Cluster attempts to find a pair of peaks with a valley between them where the difference between the peak and valley histogram counts is statistically significant.

A **sensitivity** level parameter specifies the lowest density that may be considered a peak. Sensitivity is an optional parameter for numeric data. It may be used to filter the splitting point candidates.

Partitioning Categorical Attributes

Categorical values do not have an intrinsic order associated with them. Therefore it is impossible to apply the notion of histogram peaks and valleys that is used to partition numerical values.

Instead the counts of individual values form a histogram. Bins with large counts are interpreted as regions with high density. The clustering objective is to separate these high-density areas and effectively decrease the entropy (randomness) of the data.

O-Cluster identifies the histogram with highest entropy along the individual projections. Entropy is measured as the number of bins above **sensitivity** level. O-Cluster places the two largest bins into separate partitions, thereby creating a splitting predicate. The remainder of the bins are assigned randomly to the two resulting partitions.

Active Sampling

The O-Cluster algorithm operates on a data buffer of a limited size. It uses an active sampling mechanism to handle data sets that do not fit into memory.

After processing an initial random sample, O-Cluster identifies cases that are of no further interest. Such cases belong to *frozen* partitions where further splitting is highly unlikely. These cases are replaced with examples from *ambiguous* regions where further information (additional cases) is needed to find good splitting planes and continue partitioning. A partition is considered ambiguous if a valid split can only be found at a lower confidence level.

Cases associated with frozen partitions are marked for deletion from the buffer. They are replaced with cases belonging to ambiguous partitions. The histograms of the ambiguous partitions are updated and splitting points are reevaluated.

Process Flow

The O-Cluster algorithm evaluates possible splitting points for all projections in a partition, selects the best one, and splits the data into two new partitions. The algorithm proceeds by searching for good cutting planes inside the newly created partitions. Thus O-Cluster creates a binary tree structure that divides the input space into rectangular regions with no overlaps or gaps.

The main processing stages are:

1. Load the buffer. Assign all cases from the initial buffer to a single active root partition.

2. Compute histograms along the orthogonal uni-dimensional projections for each active partition.
3. Find the best splitting points for active partitions.
4. Flag ambiguous and frozen partitions.
5. When a valid separator exists, split the active partition into two new active partitions and start over at step 2.
6. Reload the buffer after all recursive partitioning on the current buffer is completed. Continue loading the buffer until either the buffer is filled again, or the end of the data set is reached, or until the number of cases is equal to the data buffer size.

> **Note:** O-Cluster requires at most one pass through the data

Scoring

The clusters discovered by O-Cluster are used to generate a Bayesian probability model that can be used to score new data. The generated probability model is a mixture model where the mixture components are represented by a product of independent normal distributions for numerical attributes and multinomial distributions for categorical attributes.

Tuning the O-Cluster Algorithm

The O-Cluster algorithm supports two build-time settings. Both settings have default values. There is no reason to override the defaults unless you want to influence the behavior of the algorithm in some specific way.

You can configure O-Cluster by specifying any of the following:

- **Buffer size** — Size of the processing buffer. (See Active Sampling.)
- **Sensitivity factor** — A fraction that specifies the peak density required for separating a new cluster. (See Partitioning Strategy.)

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for details about the build settings for O-Cluster

Data Preparation for O-Cluster

Automatic Data Preparation bins numerical attributes for O-Cluster. It uses a specialized form of equi-width binning that computes the number of bins per attribute automatically. Numerical columns with all nulls or a single value are removed.

O-Cluster handles missing values naturally as missing at random. The algorithm does not support nested columns and thus does not support sparse data.

> **See Also:**
>
> Chapter 19, "Automatic and Embedded Data Preparation"
>
> *Oracle Data Mining Application Developer's Guide* for information about nested columns and missing data

User-Specified Data Preparation for O-Cluster

Keep the following in mind if you choose to prepare the data for O-Cluster:

- O-Cluster does not necessarily use all the input data when it builds a model. It reads the data in batches (the default batch size is 50000). It will only read another batch if it believes, based on statistical tests, that there may still exist clusters that it has not yet uncovered.
- Binary attributes should be declared as categorical.
- Automatic equi-width binning is highly recommended. The bin identifiers are expected to be positive consecutive integers starting at 1. See *Oracle Database PL/SQL Packages and Types Reference* for an example.
- The presence of outliers can significantly impact clustering algorithms. Use a clipping transformation before binning or normalizing. Outliers with equi-width binning can prevent O-Cluster from detecting clusters. As a result, the whole population appears to fall within a single cluster.

18

Support Vector Machines

This chapter describes Support Vector Machines, a powerful algorithm based on statistical learning theory. Support Vector Machines is implemented by Oracle Data Mining for classification, regression, and anomaly detection.

See Also:

Chapter 5, "Classification"

Chapter 4, "Regression"

Chapter 6, "Anomaly Detection"

Reference:

Milenova, B.L., Yarmus, J.S., Campos, M.M., "SVM in Oracle Database 10*g*: Removing the Barriers to Widespread Adoption of Support Vector Machines", Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005.

`http://www.oracle.com/technology/products/bi/odm/`

This chapter contains the following sections:

- About Support Vector Machines
- Tuning an SVM Model
- Data Preparation for SVM
- SVM Classification
- One-Class SVM
- SVM Regression

About Support Vector Machines

Support Vector Machines (SVM) is a powerful, state-of-the-art algorithm with strong theoretical foundations based on the Vapnik-Chervonenkis theory. SVM has strong **regularization** properties. Regularization refers to the generalization of the model to new data.

Advantages of SVM

SVM models have similar functional form to neural networks and radial basis functions, both popular data mining techniques. However, neither of these algorithms has the well-founded theoretical approach to regularization that forms the basis of SVM. The quality of generalization and ease of training of SVM is far beyond the capacities of these more traditional methods.

SVM can model complex, real-world problems such as text and image classification, hand-writing recognition, and bioinformatics and biosequence analysis.

SVM performs well on data sets that have many attributes, even if there are very few cases on which to train the model. There is no upper limit on the number of attributes; the only constraints are those imposed by hardware. Traditional neural nets do not perform well under these circumstances.

Advantages of SVM in Oracle Data Mining

Oracle Data Mining has its own proprietary implementation of SVM, which exploits the many benefits of the algorithm while compensating for some of the limitations inherent in the SVM framework. Oracle Data Mining SVM provides the scalability and usability that are needed in a production quality data mining system.

Usability

Usability is a major enhancement, because SVM has often been viewed as a tool for experts. The algorithm typically requires data preparation, tuning, and optimization. Oracle Data Mining minimizes these requirements. You do not need to be an expert to build a quality SVM model in Oracle Data Mining. For example:

- Data preparation is not required in most cases. (See "Data Preparation for SVM" on page 18-4.)
- Default tuning parameters are generally adequate. (See "Tuning an SVM Model" on page 18-3.)

Scalability

When dealing with very large data sets, sampling is often required. However, sampling is not required with Oracle Data Mining SVM, because the algorithm itself uses stratified sampling to reduce the size of the training data as needed.

Oracle Data Mining SVM is highly optimized. It builds a model incrementally by optimizing small working sets toward a global solution. The model is trained until convergence on the current working set, then the model adapts to the new data. The process continues iteratively until the convergence conditions are met. The Gaussian kernel uses caching techniques to manage the working sets. See "Kernel-Based Learning" on page 18-2.

Oracle Data Mining SVM supports **active learning**, an optimization method that builds a smaller, more compact model while reducing the time and memory resources required for training the model. See "Active Learning" on page 18-3.

Kernel-Based Learning

SVM is a kernel-based algorithm. A **kernel** is a function that transforms the input data to a high-dimensional space where the problem is solved. Kernel functions can be linear or nonlinear.

Oracle Data Mining supports linear and Gaussian (nonlinear) kernels.

In Oracle Data Mining, the **linear kernel** function reduces to a linear equation on the original attributes in the training data. A linear kernel works well when there are many attributes in the training data.

The **Gaussian kernel** transforms each case in the training data to a point in an *n*-dimensional space, where *n* is the number of cases. The algorithm attempts to separate the points into subsets with homogeneous target values. The Gaussian kernel uses nonlinear separators, but within the kernel space it constructs a linear equation.

Active Learning

Active learning is an optimization method for controlling model growth and reducing model build time. Without active learning, SVM models grow as the size of the build data set increases, which effectively limits SVM models to small and medium size training sets (less than 100,000 cases). Active learning provides a way to overcome this restriction. With active learning, SVM models can be built on very large training sets.

Active learning forces the SVM algorithm to restrict learning to the most informative training examples and not to attempt to use the entire body of data. In most cases, the resulting models have predictive accuracy comparable to that of a standard (exact) SVM model.

Active learning provides a significant improvement in both linear and Gaussian SVM models, whether for classification, regression, or anomaly detection. However, active learning is especially advantageous for the Gaussian kernel, because nonlinear models can otherwise grow to be very large and can place considerable demands on memory and other system resources.

Tuning an SVM Model

SVM has built-in mechanisms that automatically choose appropriate settings based on the data. You may need to override the system-determined settings for some domains.

The build settings described in Table 18–1 are available for configuring SVM models. Settings pertain to regression, classification, and anomaly detection unless otherwise specified.

Table 18–1 Build Settings for Support Vector Machines

Setting Name	Configures....	Description
`SVMS_KERNEL_FUNCTION`	Kernel	Linear or Gaussian. The algorithm automatically uses the kernel function that is most appropriate to the data. SVM uses the linear kernel when there are many attributes (more than 100) in the training data, otherwise it uses the Gaussian kernel. See "Kernel-Based Learning" on page 18-2. The number of attributes does not correspond to the number of columns in the training data. SVM explodes categorical attributes to binary, numeric attributes. In addition, Oracle Data Mining interprets each row in a nested column as a separate attribute. See "Data Preparation for SVM" on page 18-4.
`SVMS_STD_DEV`	Standard deviation for Gaussian kernel	Controls the spread of the Gaussian kernel function. SVM uses a data-driven approach to find a standard deviation value that is on the same scale as distances between typical cases.
`SVMS_KERNEL_CACHE_SIZE`	Cache size for Gaussian kernel	Amount of memory allocated to the Gaussian kernel cache maintained in memory to improve model build time. The default cache size is 50 MB.

Table 18–1 (Cont.) Build Settings for Support Vector Machines

Setting Name	Configures....	Description
SVMS_ACTIVE_LEARNING	Active learning	Whether or not to use active learning. This setting is especially important for nonlinear (Gaussian) SVM models. By default, active learning is enabled. See "Active Learning" on page 18-3.
SVMS_COMPLEXITY_FACTOR	Complexity factor	Regularization setting that balances the complexity of the model against model robustness to achieve good generalization on new data. SVM uses a data-driven approach to finding the complexity factor.
SVMS_CONVERGENCE_TOLERANCE	Convergence tolerance	The criterion for completing the model training process. The default is 0.001.
SVMS_EPSILON	Epsilon factor for regression	Regularization setting for regression, similar to complexity factor. Epsilon specifies the allowable residuals, or noise, in the data.
SVMS_OUTLIER_RATE	Outliers for anomaly detection	The expected outlier rate in anomaly detection. The default rate is 0.1.

See Also: *Oracle Database PL/SQL Packages and Types Reference* for details about SVM settings

Data Preparation for SVM

The SVM algorithm operates natively on numeric attributes. The algorithm automatically "explodes" categorical data into a set of binary attributes, one per category value. For example, a character column for marital status with values married or single would be transformed to two numeric attributes: married and single. The new attributes could have the value 1 (true) or 0 (false).

When there are missing values in columns with simple data types (not nested), SVM interprets them as missing at random. The algorithm automatically replaces missing categorical values with the mode and missing numerical values with the mean.

When there are missing values in nested columns, SVM interprets them as sparse. The algorithm automatically replaces sparse numerical data with zeros and sparse categorical data with zero vectors.

Normalization

SVM requires the normalization of numeric input. Normalization places the values of numeric attributes on the same scale and prevents attributes with a large original scale from biasing the solution. Normalization also minimizes the likelihood of overflows and underflows. Furthermore, normalization brings the numerical attributes to the same scale (0,1) as the exploded categorical data.

SVM and Automatic Data Preparation

The SVM algorithm automatically handles missing value treatment and the transformation of categorical data, but normalization and outlier detection must be handled by ADP or prepared manually. ADP performs min-max normalization for SVM.

Note: Oracle recommends that you use Automatic Data Preparation with SVM. The transformations performed by ADP are appropriate for most models.

See Chapter 19, "Automatic and Embedded Data Preparation".

SVM Classification

SVM classification is based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. SVM finds the vectors ("support vectors") that define the separators giving the widest separation of classes.

SVM classification supports both binary and multiclass targets.

Class Weights

In SVM classification, weights are a biasing mechanism for specifying the relative importance of target values (classes).

SVM models are automatically initialized to achieve the best average prediction across all classes. However, if the training data does not represent a realistic distribution, you can bias the model to compensate for class values that are under-represented. If you increase the weight for a class, the percent of correct predictions for that class should increase.

See Also: "Priors" on page 5-7

One-Class SVM

Oracle Data Mining uses SVM as the one-class classifier for anomaly detection. When SVM is used for anomaly detection, it has the classification mining function but no target.

One-class SVM models, when applied, produce a prediction and a probability for each case in the scoring data. If the prediction is 1, the case is considered typical. If the prediction is 0, the case is considered anomalous. This behavior reflects the fact that the model is trained with normal data.

You can specify the percentage of the data that you expect to be anomalous with the `SVMS_OUTLIER_RATE` build setting. If you have some knowledge that the number of "suspicious" cases is a certain percentage of your population, then you can set the outlier rate to that percentage. The model will identify approximately that many "rare" cases when applied to the general population. The default is 10%, which is probably high for many anomaly detection problems.

SVM Regression

SVM uses an epsilon-insensitive loss function to solve regression problems.

SVM regression tries to find a continuous function such that the maximum number of data points lie within the epsilon-wide insensitivity tube. Predictions falling within epsilon distance of the true target value are not interpreted as errors.

The epsilon factor is a regularization setting for SVM regression. It balances the margin of error with model robustness to achieve the best generalization to new data. See Table 18–1 for descriptions of build settings for SVM.

Part IV

Data Preparation

In Part IV, you will learn about the automatic and embedded data transformation features supported by Oracle Data Mining.

Part IV contains the following chapter:

19

Automatic and Embedded Data Preparation

This chapter explains how to use features of Oracle Data Mining to prepare data for mining.

This chapter contains the following sections:

Overview

The quality of a model depends to a large extent on the quality of the data used to build (train) it. Much of the time spent in any given data mining project is devoted to data preparation. The data must be carefully inspected, cleansed, and transformed, and algorithm-appropriate data preparation methods must be applied.

The process of data preparation is further complicated by the fact that any data to which a model is applied, whether for testing or for scoring, must undergo the same transformations as the data used to train the model.

Oracle Data Mining offers several features that significantly simplify the process of data preparation.

- **Embedded data preparation** — The transformations used in training the model are embedded in the model and automatically executed whenever the model is applied to new data. If you specify transformation for the model, you only have to specify them once.
- **Automatic Data Preparation (ADP)** — Oracle Data Mining supports an automated data preparation mode. When ADP is active, Oracle Data Mining automatically performs the data transformations required by the algorithm. The transformation instructions are embedded in the model along with any user-specified transformation instructions.
- **Tools for custom data preparation** — Oracle Data Mining provides a PL/SQL package of transformation routines that you can use to build your own transformation instructions. You can use these transformation instructions along with ADP or instead of ADP.
- **Automatic management of missing values and sparse data** — Oracle Data Mining uses consistent methodology across mining algorithms to handle sparsity and missing values.

- **Transparency** — Oracle Data Mining provides model details, which are a view of the categorical and numerical attributes internal to the model. This insight into the inner details of the model is possible because of reverse transformations, which map the transformed attribute values to a form that can be interpreted by a user. Where possible, attribute values are reversed to the original column values. Reverse transformations are also applied to the target of a supervised model, thus the results of scoring are in the same units as the units of the original target.

The Case Table

The first step in preparing data for mining is the creation of a **case table**. If all the data resides in a single table and all the information for each case (record) is included in a single row (single-record case), this process is already taken care of.

If the data resides in several tables, creating the data source involves the creation of a view. For the sake of simplicity, the term "case table" refers to either a table or a view.

When the data source includes transactional data (multi-record case), it must be aggregated to the case level, using nested columns when desired. In transactional data, the information for each case is contained in multiple rows. An example is sales data in a star schema when mining at the product level. Sales is stored in many rows for a single product (the case) since the product is sold in many stores to many customers over a period of time.

Once you have created a case table that contains all the pertinent data, you should cleanse the data of any inconsistent formats within columns. Pay special attention to such items as phone numbers, zip codes, and dates, as described in the following section.

See Also: *Oracle Data Mining Application Developer's Guide* for details

Data Type Conversion

Oracle Data Mining supports a limited number of column data types. Numeric data is interpreted as numerical attributes and character data is interpreted as categorical attributes.

You must convert the data type of a column if its type is not supported by Oracle Data Mining or if its type will cause Oracle Data Mining to interpret it incorrectly. For example, zip codes identify different postal zones; they do not imply order. If the zip codes are stored in a numeric column, it will be interpreted as a numerical attribute. You must convert the data type so that the column data can be used as a categorical attribute by the model. You can do this using the `TO_CHAR` function to convert the digits 1-9 and the `LPAD` function to retain the leading 0, if there is one.

```
LPAD(TO_CHAR(ZIPCODE),5,'0')
```

Date Data

The Oracle Data Mining APIs do not support `DATE` and `TIMESTAMP` data. Date columns must be converted to simple numeric or character data types for data mining.

In most cases, `DATE` and `TIMESTAMP` should be converted to `NUMBER`, but you should evaluate each case individually. A `TIMESTAMP` column should generally be converted to a number since it represents a unique point in time.

Alternatively, a column of dates in a table of annual sales data might indicate the month when a product was sold. This `DATE` column would be converted to `VARCHAR2` and treated as a categorical. You can use the `TO_CHAR` function to convert a `DATE` data type to `VARCHAR2`.

You can convert dates to numbers by selecting a starting date and subtracting it from each date value. Another approach would be to parse the date and distribute its components over several columns. This approach is used by DBMS_PREDICTIVE_ANALYTICS, which *does* support DATE and TIMESTAMP data types.

See Also:

Oracle Database SQL Language Reference for information on data type conversion

Oracle Database PL/SQL Packages and Types Reference for information about date data types supported by DBMS_PREDICTIVE_ANALYTICS

Text Transformation

You can use Oracle Data Mining to mine text. Columns of text in the case table can be mined once they have undergone the proper transformation.

The text column must be in a table, not a view. The transformation process uses several features of Oracle Text; it treats the text in each row of the table as a separate document. Each document is transformed to a set of text tokens known as **terms**, which have a numeric value and a text label. The text column is transformed to a nested column of DM_NESTED_NUMERICALS.

See Also: *Oracle Data Mining Application Developer's Guide* for details

Business and Domain-Sensitive Transformations

Some transformations are dictated by the definition of the business problem. For example, you might want to build a model to predict high-revenue customers. Since your revenue data for current customers is in dollars you need to define what "high-revenue" means. Using some formula that you have developed from past experience, you might recode the revenue attribute into ranges Low, Medium, and High before building the model.

Another common business transformation is the conversion of date information into elapsed time. For example, date of birth might be converted to age.

In some cases, the data might need to be transformed to minimize an unwanted interpretation by the model. An example is logarithmic transformations. You might take the log of a numerical attribute when the values fall within a very wide range. For instance, commissions might range from a few hundred to a million. Converting to the log scale would minimize the skewing effect on the model.

Domain knowledge can be very important in deciding how to prepare the data. For example, some algorithms might produce unreliable results if the data contains values that fall far outside of the normal range. In some cases, these values represent errors or abnormalities. In others, they provide meaningful information. See "Outlier Treatment" on page 19-4.

Automatic Data Preparation

Most algorithms require some form of data transformation. During the model training process, Oracle Data Mining can automatically perform the transformations required by the algorithm. You can choose to supplement the automatic transformations with additional transformations of your own, or you can choose to manage all the transformations yourself.

In calculating automatic transformations, Oracle Data Mining uses heuristics that address the common requirements of a given algorithm. This process results in reasonable model quality in most cases.

Binning, normalization, and outlier treatment are transformations that are commonly needed by data mining algorithms.

Binning

Binning, also called discretization, is a technique for reducing the cardinality of continuous and discrete data. Binning groups related values together in bins to reduce the number of distinct values.

Binning can improve resource utilization and model build response time dramatically without significant loss in model quality. Binning can improve model quality by strengthening the relationship between attributes.

Note: Binning is the primary transformation required by **Naive-Bayes** and **Attribute Importance** algorithms. In Oracle Data Mining, the Decision Tree algorithm implements its own form of binning (supervised binning).

Normalization

Normalization is the most common technique for reducing the range of numerical data. Most normalization methods map the range of a single variable to another range (often 0,1).

Note: Normalization is the primary transformation required by **Support Vector Machine** (one-class, classification, and regression), **Non-Negative Matrix Factorization**, and ***k*-Means** algorithms.

Outlier Treatment

A value is considered an outlier if it deviates significantly from most other values in the column. The presence of outliers can have a skewing effect on the data and can interfere with the effectiveness of transformations such as normalization or binning.

Outlier treatment methods such as trimming or clipping can be implemented to minimize the effect of outliers.

Outliers may represent problematic data, for example a bad reading due to the abnormal condition of an instrument. However, in some cases, especially in the business arena, outliers may be perfectly valid. For example, in census data, the earnings for some of the richest individuals may vary significantly from the general population. This information should not be treated as an outlier, since it is an important part of the data. Domain knowledge is usually needed to determine outlier handling.

Transformations With Automatic Data Preparation

Table 19–1 shows how ADP prepares the data for each algorithm.

Note: Many algorithms incorporate some form of data preparation. For example, algorithms that operate natively on numeric attributes explode each non-numeric input column into a set of numerical attributes.

Transformations encapsulated within the algorithm are transparent to the user and occur independently of ADP.

Also, the handling of nested data, sparsity, and missing values is standard across algorithms and occurs independently of ADP. (See *Oracle Data Mining Application Developer's Guide*.)

Table 19–1 Oracle Data Mining Algorithms With ADP

Algorithm	Mining Function	Treatment by ADP
Naive Bayes	Classification	All attributes are binned with supervised binning.
Decision Tree	Classification	The ADP setting has no effect on Decision Tree. Data preparation is handled by the algorithm.
GLM	Classification and Regression	Numerical attributes are normalized.
SVM	Classification, Anomaly Detection, and Regression	Numerical attributes are normalized.
k-Means	Clustering	Numerical attributes are normalized with outlier-sensitive normalization.
O-Cluster	Clustering	Numerical attributes are binned with a specialized form of equi-width binning, which computes the number of bins per attribute automatically. Numerical columns with all nulls or a single value are removed.
MDL	Attribute Importance	All attributes are binned with supervised binning..
Apriori	Association Rules	The ADP setting has no effect on association rules.
NMF	Feature Extraction	Numerical attributes are normalized.

See Also: The chapters on the individual algorithms in Part III for more information about algorithm-specific data preparation

Embedded Data Preparation

Transformations can be embedded in a model automatically by ADP or they can be embedded as a result of user-specified transformation instructions. To specify your own embedded transformations, create a transformation list and pass it to DBMS_DATA_MINING.CREATE_MODEL.

```
PROCEDURE create_model(
                  model_name           IN VARCHAR2,
                  mining_function      IN VARCHAR2,
                  data_table_name      IN VARCHAR2,
                  case_id_column_name  IN VARCHAR2,
                  target_column_name   IN VARCHAR2 DEFAULT NULL,
                  settings_table_name  IN VARCHAR2 DEFAULT NULL,
                  data_schema_name     IN VARCHAR2 DEFAULT NULL,
                  settings_schema_name IN VARCHAR2 DEFAULT NULL,
                  xform_list           IN TRANSFORM_LIST DEFAULT NULL);
```

See Also: For details about transformation lists, see *Oracle Database PL/SQL Packages and Types Reference*

Transformations Lists and Automatic Data Preparation

If you enable ADP and you specify a transformation list, the transformation list is embedded with the automatic, system-generated transformations. The transformation list is executed before the automatic transformations.

If you enable ADP and do not specify a transformation list, only the automatic transformations are embedded in the model.

If ADP is disabled (the default) and you specify a transformation list, your custom transformations are embedded in the model. No automatic transformations are performed.

If ADP is disabled (the default) and you do not specify a transformation list, no transformations will be embedded in the model. You will have to transform the build, test, and scoring data sets yourself. You must take care to apply the same transformations to each data set. This method of data preparation was required in previous releases of Oracle Data Mining.

Oracle Data Mining Transformation Routines

Oracle Data Mining provides routines that implement various transformation techniques in the `DBMS_DATA_MINING_TRANSFORM` package. Details about the package are in *Oracle Database PL/SQL Packages and Types Reference*.

Binning Routines

A number of factors go into deciding a binning strategy. Having fewer values typically leads to a more compact model and one that builds faster, but it can also lead to some loss in accuracy.

Model quality can improve significantly with well-chosen bin boundaries. For example, an appropriate way to bin ages might be to separate them into groups of interest, such as children 0-13, teenagers 13-19, youth 19-24, working adults 24-35, and so on.

Table 19–2 lists the binning techniques provided by Oracle Data Mining.

Table 19–2 Binning Methods in DBMS_DATA_MINING_TRANSFORM

Binning Method	Description
Top-N Most Frequent Items	You can use this technique to bin categorical attributes. You specify the number of bins. The value that occurs most frequently is labeled as the first bin, the value that appears with the next frequency is labeled as the second bin, and so on. All remaining values are in an additional bin.
Supervised Binning	Supervised binning is a form of intelligent binning, where bin boundaries are derived from important characteristics of the data. Supervised binning builds a single-predictor decision tree to find the interesting bin boundaries with respect to a target. It can be used for numerical or categorical attributes.
Equi-Width Binning	You can use equi-width binning for numerical attributes. The range of values is computed by subtracting the minimum value from the maximum value, then the range of values is divided into equal intervals. You can specify the number of bins or it can be calculated automatically. Equi-width binning should usually be used with outlier treatment. (See "Routines for Outlier Treatment" on page 19-7.)

Table 19–2 (Cont.) Binning Methods in DBMS_DATA_MINING_TRANSFORM

Binning Method	Description
Quantile Binning	Quantile binning is a numerical binning technique. Quantiles are computed using the SQL analytic function NTILE. The bin boundaries are based on the minimum values for each quantile. Bins with equal left and right boundaries are collapsed, possibly resulting in fewer bins than requested.

Normalization Routines

Most normalization methods map the range of a single attribute to another range, typically 0 to 1 or -1 to +1.

Normalization is very sensitive to outliers. Without outlier treatment, most values will be mapped to a tiny range, resulting in a significant loss of information. (See"Routines for Outlier Treatment" on page 19-7.)

Table 19–3 Normalization Methods in DBMS_DATA_MINING_TRANSFORM

Transformation	Description
Min-Max Normalization	This technique computes the normalization of an attribute using the minimum and maximum values. The shift is the minimum value, and the scale is the difference between the maximum and minimum values.
Scale Normalization	This normalization technique also uses the minimum and maximum values. For scale normalization, shift = 0, and scale = max{abs(max), abs(min)}.
Z-Score Normalization	This technique computes the normalization of an attribute using the mean and the standard deviation. Shift is the mean, and scale is the standard deviation.

Routines for Outlier Treatment

Outliers are extreme values, typically several standard deviations from the mean. To minimize the effect of outliers, you can Winsorize or trim the data.

Winsorizing involves setting the tail values of an attribute to some specified value. For example, for a 90% Winsorization, the bottom 5% of values are set equal to the minimum value in the 5th percentile, while the upper 5% of values are set equal to the maximum value in the 95th percentile.

Trimming sets the tail values to NULL. The algorithm treats them as missing values.

Outliers affect the different algorithms in different ways. In general, outliers cause distortion with equi-width binning and min-max normalization.

Table 19–4 Outlier Treatment Methods in DBMS_DATA_MINING_TRANSFORM

Transformation	Description
Trimming	This technique trims the outliers in numeric columns by sorting the non-null values, computing the tail values based on some fraction, and replacing the tail values with nulls.
Windsorizing	This technique trims the outliers in numeric columns by sorting the non-null values, computing the tail values based on some fraction, and replacing the tail values with some specified value.

Transparency

Oracle Data Mining support for model transparency ensures that information returned by the model is expressed in a format that is similar to or the same as the format of the data that was used to train the model. Internal transformations are reversed in the model details and in the predictions generated by supervised models.

Internal Transformations

Some of the attributes used by the model correspond to columns in the build data. However, because of logic specific to the algorithm, nested data, and transformations, many attributes do not correspond to columns.

For example, a nested column in the training data is not interpreted as an attribute by the model. During the model build, Oracle Data Mining explodes nested columns, and each row (an attribute name/value pair) becomes an attribute.

Some algorithms, for example SVM and GLM, only operate on numeric attributes. Any non-numeric column in the build data is exploded into binary numerical attributes, one for each distinct value in the column (SVM). GLM does not generate a new attribute for the most frequent value in the original column. These binary attributes are set to one only if the column value for the case is equal to the value associated with the binary attribute.

Algorithms that generate coefficients present challenges in regards to interpretability of results. Examples are SVN and NMF. These algorithms produce coefficients that are used in combination with the transformed attributes. The coefficients are relevant to the data on the transformed scale, not the original data scale.

Algorithms do not necessarily use all the columns in the training data. Some columns might be deemed unnecessary or harmful to the quality of the model. These columns are not used as attributes.

For all these reasons, the attributes listed in the model details might not resemble the columns of data used to train the model. However, attributes that undergo embedded transformations, whether initiated by ADP or by a user-specified transformation list, appear in the model details in their pre-transformed state, as close as possible to the original column values. Although the attributes are transformed when they are used by the model, they are visible in the model details in a form that can be interpreted by a user.

See Also: The following in *Oracle Database PL/SQL Packages and Types Reference* :

```
GET_MODEL_DETAILS
GET_MODEL_TRANSFORMATIONS
ALTER_REVERSE_EXPRESSION
```

Part V

Mining Unstructured Data

In Part V, you will learn how to use Oracle Data Mining to mine text and other forms of unstructured data.

Part V contains the following chapters:

20
Text Mining

This chapter includes the following topics:

About Unstructured Data

Data mining algorithms act on numerical and categorical data stored in relational databases or spreadsheets. Numerical data has a type such as `INTEGER`, `DECIMAL`, or `FLOAT`. Categorical data has a type such as `CHAR` or `VARCHAR2`.

What if you want to mine data items that are not numericals or categoricals? There are many examples: web pages, document libraries, PowerPoint presentations, product specifications, emails, sound files, and digital images to name a few. What if you want to mine the information stored in long character strings, such as product descriptions, comment fields in reports, or call center notes?

Data that cannot be meaningfully interpreted as numerical or categorical is considered unstructured for purposes of data mining. It has been estimated that as much as 85% of enterprise data falls into this category. Extracting meaningful information from this unstructured data can be critical to the success of a business.

How Oracle Data Mining Supports Unstructured Data

Unstructured data may be binary objects, such as image or audio files, or text objects, which are language-based. Oracle Data Mining supports text objects.

The case table for Data Mining may include one or more columns of text (see "Mixed Data" on page 20-2), which can be designated as attributes. A text column cannot be used as a target. The case table itself must be a relational table; it cannot be created as a view.

Text must undergo a transformation process before it can be mined. Once the data has been properly transformed, the case table can be used for building, testing, or scoring data mining models. Most Oracle Data Mining algorithms support text. (See "Text Mining Algorithms" on page 20-2.)

Mixed Data

Much of today's enterprise information includes both structured and unstructured content related to a given item of interest. Customer account data may include text fields that describe support calls and other interactions with the customer. Insurance claim data may include a claim status description, supporting documents, email correspondence, and other information. It is often essential that analytic applications evaluate the structured information together with the related unstructured information.

Oracle Data Mining offers this capability. You can use Oracle Data Mining to mine data sets that contain regular relational information (numeric and character columns), as well as one or more text columns.

Text Data Types

Oracle Data Mining supports text columns that have any of the data types shown in Table 20–1.

Table 20–1 Data Types for Text Columns

Data Type	Description
`BFILE`	Locator to a large binary file stored outside the database
`BLOB`	Binary large object
`CHAR`	Fixed length character string
`CLOB`	Character large object
`LONG`	Long variable length character string
`LONG RAW`	Long variable length raw binary data
`RAW`	Raw binary data
`VARCHAR2`	Variable length character string
`XMLTYPE`	XML data

See Also: *Oracle Database SQL Language Reference* for information about Oracle data types

Text Mining Algorithms

The Oracle Data Mining algorithms shown in Table 20–2 can be used for text mining.

Table 20–2 Oracle Data Mining Algorithms that Support Text

Algorithm	Mining Function
Naive Bayes	Classification
Generalized Linear Models	Classification, Regression
Support Vector Machine	Classification, Regression, Anomaly Detection
k-Means	Clustering
Non-Negative Matrix Factorization	Feature Extraction
Apriori	Association Rules
Minimum Description Length	Attribute Importance

Oracle Data Mining supports text with all mining functions. As shown in Table 20–2, at least one algorithm per mining function has text mining capability.

Classification, clustering, and feature extraction have important applications in pure text mining. Other functions, such as regression and anomaly detection, are more suited for mining mixed data (both structured and unstructured).

Text Classification

Text classification is the process of categorizing documents: for example, by subject or author. Most document classification applications use either multi-class classification or multi-target classification.

Multi-Class Document Classification

In multi-class document classification, each document is assigned a probability for each category, and the probabilities add to 1. For example, if the categories are `economics`, `math`, and `physics`, document A might be 20% likely to be `economics`, 50% likely to be `math`, and 30% likely to be `physics`.

This approach to document classification is supported by Oracle Data Mining and by Oracle Text.

Multi-Target Document Classification

In multi-target document classification, each document is assigned a probability for either being in a category or not being in a category, and the probabilities for each category add to 1. Given categories `economics`, `math`, and `physics`, document A might be classified as: 30% likely to be `economics` and 70% likely not to be `economics`; 65% likely to be `math` and 35% likely not to be `math`; 40% likely to be `physics` and 60% likely not to be `physics`.

In multi-target document classification, each category is a separate binary target. Each document is scored for each target.

This approach to document classification is supported by Oracle Text but not by Oracle Data Mining. However, you can obtain similar results by building a single binary classification model for each category and then scoring all the models separately in a single SQL scoring query.

See Also: "Oracle Data Mining and Oracle Text" on page 20-5

Document Classification Algorithms

Oracle Data Mining supports three classification algorithms that are well suited to text mining applications. Both can easily process thousands of text features (see "Preparing Text for Mining" on page 20-5 for information about text features), and both are easy to train with small or large amounts of data. The algorithms are:

- Support Vector Machine (SVM), described in Chapter 18
- Logistic Regression (GLM), described in Chapter 12
- Naive Bayes (NB), described in Chapter 15

See Also: Chapter 5, "Classification"

Text Clustering

The main applications of clustering in text mining are:

- Simple clustering. This refers to the creation of clusters of text features (see "Preparing Text for Mining" on page 20-5 for information about text features). For example: grouping the hits returned by a search engine.
- Taxonomy generation. This refers to the generation of hierarchical groupings. For example: a cluster that includes text about car manufacturers is the parent of child clusters that include text about car models.
- Topic extraction. This refers to the extraction of the most typical features of a group. For example: the most typical characteristics of documents in each document topic.

The Oracle Data Mining enhanced *k*-Means clustering algorithm, described in Chapter 13, supports text mining.

See Also: Chapter 7, "Clustering"

Text Feature Extraction

Feature extraction is central to text mining. Feature extraction is used for text transformation at two different stages in the text mining process:

1. A feature extraction process must be performed on text documents before they can be mined. This preprocessing step transforms text documents into small units of text called **features** or **terms**.

   See Also: "Preparing Text for Mining" on page 20-5.

2. The text transformation process generates large numbers (potentially many thousands) of text features from a text document. Oracle Data Mining algorithms treat each feature as a separate attribute. Thus text data may present a huge number of attributes, many of which provide little significant information for training a supervised model or building an unsupervised model.

   Oracle Data Mining supports the Non-Negative Matrix Factorization (NMF) algorithm for feature extraction. You can create an NMF model to consolidate the text attributes derived from the case table and generate a reduced set of more meaningful attributes. The results can be far more effective for use in classification, clustering, or other types of data mining models. See Chapter 16 for information on NMF.

   See Also: Chapter 9, "Feature Selection and Extraction"

Text Association

Association models can be used to uncover the semantic meaning of words. For example, suppose that the word `account` co-occurs with words like `customer`, `explanation`, `churn`, `story`, `region`, `debit`, and `memo`. An association model would produce rules connecting `account` with these concepts. Inspection of the rules would provide context for `account` in the document collection. Such associations can improve information retrieval engines.

Oracle Data Mining supports Apriori for association. See Chapter 10 for information on Apriori.

See Also: Chapter 8, "Association"

Text Attribute Importance

Attribute importance can be used to find terms that distinguish the values of a target column. Attribute importance ranks the relative importance of the terms in predicting the target. For example, certain words and phrases might distinguish the writing style of one writer from another.

Oracle Data Mining supports Minimum Description Length (MDL) for attribute importance. See Chapter 14 for information on MDL.

> **See Also:** Chapter 9, "Feature Selection and Extraction"

Preparing Text for Mining

Before text can be mined, it must undergo a special preprocessing step known as **term extraction**, also called **feature extraction**. This process breaks the text down into units (terms) that can be mined. Text terms may be keywords or other document-derived features.

The Oracle Data Miner graphical tool performs term extraction transparently when you create or apply a text mining model. You can use a set of Oracle Text table functions to extract terms for text mining with the PL/SQL API, as described in *Oracle Data Mining Application Developer's Guide*.

> **See Also:** *Oracle Data Mining Administrator's Guide* for information about sample term extraction code provided with the Oracle Data Mining sample programs

The term extraction process uses Oracle Text routines to transform a text column into a nested column. Each term is the name of a nested attribute. The value of each nested attribute is a number that uniquely identifies the term. Thus each term derived from the text is used as a separate numerical attribute by the data mining algorithm.

All Oracle Data Mining algorithms that support nested data can be used for text mining. These algorithms are listed in Table 20–2

> **See Also:**
>
> "Oracle Data Mining and Oracle Text" on page 20-5
>
> *Oracle Data Mining Application Developer's Guide* for information about nested data

Oracle Data Mining and Oracle Text

Oracle Text is a technology included in the base functionality offered by Oracle Database. Oracle Text uses internal components of Oracle Data Mining to provide some data mining capabilities.

Oracle Data Mining is an option of the Enterprise Edition of Oracle Database. To use Oracle Data Mining, you must have a license for the Data Mining option. To use Oracle Text and its data mining capabilities, you do not need to license the Data Mining option.

Oracle Text consists of a set of PL/SQL packages and related data structures that support text query and document classification. Oracle Text routines can be used to:

- Query document collections, such as web sites and online libraries
- Query document catalogs, such author and publisher descriptions

- Perform document classification and clustering

The primary functional differences between Oracle Data Mining and Oracle Text can be summarized as follows:

- Oracle Data Mining supports the mining of mixed data, as described in "Mixed Data" on page 20-2. Oracle Text mining capabilities only support text; they do not support mixed structured and unstructured data.
- Oracle Data Mining supports mining more than one text column at once. Oracle Text routines operate on a single column.
- Oracle Data Mining supports text with all data mining functions. Oracle Text has limited support for data mining. The differences are summarized in Table 20–3.
- Oracle Data Mining and Oracle Text both support text columns with any of the data types listed in Table 20–1. Oracle Data Mining requires text feature extraction transformation prior to mining, as described in "Preparing Text for Mining" on page 20-5. Oracle Text operates on native text; it performs text feature extraction internally.
- Oracle Data Mining and Oracle Text both support document classification, as described in "Text Classification" on page 20-3. Oracle Data Mining supports multi-class classification. Oracle Text supports multi-class and multi-target classification.

Table 20–3 Mining Functions: Oracle Data Mining and Oracle Text

Mining Function	Oracle Data Mining	Oracle Text
Anomaly detection	Text or mixed data can be mined using One-Class SVM	No support
Association	Text or mixed data can be mined using Apriori	No support
Attribute importance	Text or mixed data can be mined using MDL	No support
Classification	Text or mixed data can be mined using SVM, GLM, or Naive Bayes	Text can be mined using SVM, decision trees, or user-defined rules
Clustering	Text or mixed data can be mined using *k*-Means	Text can be mined using *k*-Means
Feature extraction	Text or mixed data can be mined using NMF	No support
Regression	Text or mixed data can be mined using SVM or GLM	No support

See Also:

Oracle Text Application Developer's Guide

Oracle Text Reference

Glossary

active learning

A feature of the **Support Vector Machine** algorithm that provides a way to deal with large training data sets.

ADP

See **Automatic Data Transformation**,

aggregation

The process of consolidating data values into a smaller number of values. For example, sales data could be collected on a daily basis and then be totalled to the week level.

algorithm

A sequence of steps for solving a problem. See **data mining algorithm**. The Oracle Data Mining programmatic interfaces support the following algorithms: **MDL**, **Apriori**, **Decision Tree**, **k-Means**, **Naive Bayes**, **GLM**, **O-Cluster**, and **Support Vector Machine**.

algorithm settings

The settings that specify algorithm-specific behavior for model building.

anomaly detection

The detection of outliers or atypical cases. To build an anomaly detection model using the Data Mining programmatic interfaces, specify classification as the mining function, SVM as the algorithm, and pass a `NULL` or empty string as the target column name.

apply

The data mining operation that scores data, that is, uses the model with new data to predict results.

Apriori

Uses frequent itemsets to calculate associations.

association

A machine learning technique that identifies relationships among items.

association rules

A mining function that captures co-occurrence of items among transactions. A typical rule is an implication of the form A -> B, which means that the presence of itemset A implies the presence of itemset B with certain support and confidence. The support of the rule is the ratio of the number of transactions where the itemsets A and B are present to the total number of transactions. The confidence of the rule is the ratio of the number of transactions where the itemsets A and B are present to the number of transactions where itemset A is present. Oracle Data Mining uses the Apriori algorithm for association models.

attribute

An attribute is a predictor in a predictive model or an item of descriptive information in a descriptive model. **Data attributes** are the columns used to build a model. Data attributes undergo transformations so that they can be used as categoricals or numericals by the model. Categoricals and numericals are **model attributes**. See also **target**.

attribute importance

A mining function providing a measure of the importance of an attribute in predicting a specified target. The measure of different attributes of a training data table enables users to select the attributes that are found to be most relevant to a mining model. A smaller set of attributes results in a faster model build; the resulting model could be more accurate. Oracle Data Mining uses the **Minimum Description Length** to discover important attributes. Sometimes referred to as *feature selection* or *key fields*.

Automatic Data Transformation

Mining models can be created in Automatic Data Preparation (ADP) mode. ADP transforms the build data according to the requirements of the algorithm, embeds the transformation instructions in the model, and uses the instructions to transform the test or scoring data when the model is applied.

binning

See **discretization**.

build data

Data used to build (train) a model. Also called *training data*.

case

All the data collected about a specific transaction or related set of values. A data set is a collection of cases. Cases are also called *records* or *examples*. In the simplest situation, a case corresponds to a row in a table.

case table

A table or view in single-record case format. All the data for each case is contained in a single row. The case table may include a case ID column that holds a unique identifier for each row. Mining data must be presented as a case table.

categorical attribute

An attribute whose values correspond to discrete categories. For example, *state* is a categorical attribute with discrete values (CA, NY, MA). Categorical attributes are either non-ordered (nominal) like state or gender, or ordered (ordinal) such as high, medium, or low temperatures.

centroid

See **cluster centroid**.

classification

A mining function for predicting categorical target values for new records using a model built from records with known target values. Oracle Data Mining supports the following algorithms for classification: Naive Bayes, Decision Tree, and Support Vector Machines.

clipping

See **trimming**.

cluster centroid

The vector that encodes, for each attribute, either the mean (if the attribute is numerical) or the mode (if the attribute is categorical) of the cases in the training data assigned to a cluster. A cluster centroid is often referred to as "the centroid."

clustering

A mining function for finding naturally occurring groupings in data. More precisely, given a set of data points, each having a set of attributes, and a similarity measure among them, clustering is the process of grouping the data points into different clusters such that data points in the same cluster are more similar to one another and data points in different clusters are less similar to one another. Oracle Data Mining supports two algorithms for clustering, **k-Means** and **Orthogonal Partitioning Clustering**.

confusion matrix

Measures the correctness of predictions made by a model from a test task. The row indexes of a confusion matrix correspond to *actual values* observed and provided in the test data. The column indexes correspond to *predicted values* produced by applying the model to the test data. For any pair of actual/predicted indexes, the value indicates the number of records classified in that pairing.

When predicted value equals actual value, the model produces correct predictions. All other entries indicate errors.

cost matrix

An *n* by *n* table that defines the cost associated with a prediction versus the actual value. A cost matrix is typically used in classification models, where *n* is the number of distinct values in the target, and the columns and rows are labeled with target values. The rows are the actual values; the columns are the predicted values.

counterexample

Negative instance of a target. Counterexamples are required for classification models, except for **one-class Support Vector Machine**s.

data mining

Data mining is the practice of automatically searching large stores of data to discover patterns and trends that go beyond simple analysis. Data mining uses sophisticated mathematical algorithms to segment the data and evaluate the probability of future events. Data mining is also known as *Knowledge Discovery in Data* (KDD).

A data mining *model* implements a data mining algorithm to solve a given type of problem for a given set of data.

data mining algorithm

A specific technique or procedure for producing a data mining model. An algorithm uses a specific data representation and a specific **mining function**.

The algorithms in the Oracle Data Mining programming interfaces are **Naive Bayes**, **Support Vector Machine**, **Generalized Linear Model**, and **Decision Tree** for classification; **Support Vector Machine** and **Generalized Linear Model** for regression; **k-Means** and **O-Cluster** for clustering; **Minimum Description Length** for attribute importance; **Non-Negative Matrix Factorization** for feature extraction; **Apriori** for associations, and **one-class Support Vector Machine** for anomaly detection.

data mining server

The component of the Oracle database that implements the data mining engine and persistent metadata repository. You must connect to a data mining server before performing data mining tasks.

data set

In general, a collection of data. A data set is a collection of **case**s.

descriptive model

A descriptive model helps in understanding underlying processes or behavior. For example, an association model describes consumer behavior. See also **mining model**.

discretization

Discretization groups related values together under a single value (or bin). This reduces the number of distinct values in a column. Fewer bins result in models that build faster. Many Oracle Data Mining algorithms (for example NB) may benefit from input data that is *discretized* prior to model building, testing, computing lift, and applying (scoring). Different algorithms may require different types of binning. Oracle Data Mining includes transformations that perform **top N frequency binning** for categorical attributes and **equi-width binning** and **quantile binning** for numerical attributes.

distance-based (clustering algorithm)

Distance-based algorithms rely on a distance metric (function) to measure the similarity between data points. Data points are assigned to the nearest cluster according to the distance metric used.

Decision Tree

A decision tree is a representation of a classification system or supervised model. The tree is structured as a sequence of questions; the answers to the questions trace a path down the tree to a leaf, which yields the prediction.

Decision trees are a way of representing a series of questions that lead to a class or value. The top node of a decision tree is called the root node; terminal nodes are called leaf nodes. Decision trees are grown through an iterative splitting of data into discrete groups, where the goal is to maximize the distance between groups at each split.

An important characteristic of the decision tree models is that they are transparent; that is, there are rules that explain the classification.

See also **rule**.

DMS

See **data mining server**.

equi-width binning

Equi-width binning determines bins for numerical attributes by dividing the range of values into a specified number of bins of equal size.

explode

For a **categorical attribute**, replace a multi-value categorical column with several binary categorical columns. To explode the attribute, create a new binary column for each distinct value that the attribute takes on. In the new columns, 1 indicates that the value of the attribute takes on the value of the column; 0, that it does not. For example, suppose that a categorical attribute takes on the values {1, 2, 3}. To explode this attribute, create three new columns, `col_1`, `col_2`, and `col_3`. If the attribute takes on the value 1, the value in `col_1` is 1; the values in the other two columns is 0.

feature

A combination of attributes in the data that is of special interest and that captures important characteristics of the data. See **feature extraction**.

See also **text feature**.

feature extraction

Creates a new set of features by decomposing the original data. Feature extraction lets you describe the data with a number of features that is usually far smaller than the number of original attributes. See also **Non-Negative Matrix Factorization**.

Generalized Linear Model

A statistical technique for linear modeling. Generalized linear models (GLM) include and extend the class of simple linear models. Oracle Data Mining supports logistic regression for GLM classification and linear regression for GLM regression.

GLM

See **Generalized Linear Model**.

k-Means

A distance-based clustering algorithm that partitions the data into a predetermined number of clusters (provided there are enough distinct cases). Distance-based algorithms rely on a distance metric (function) to measure the similarity between data points. Data points are assigned to the nearest cluster according to the distance metric used. Oracle Data Mining provides an enhanced version of *k*-Means.

lift

A measure of how much better prediction results are using a model than could be obtained by chance. For example, suppose that 2% of the customers mailed a catalog make a purchase; suppose also that when you use a model to select catalog recipients, 10% make a purchase. Then the lift for the model is 10/2 or 5. Lift may also be used as a measure to compare different data mining models. Since lift is computed using a data table with actual outcomes, lift compares how well a model performs with respect to this data on predicted outcomes. Lift indicates how well the model improved the predictions over a random selection given actual results. Lift allows a user to infer how a model will perform on new data.

lineage

The sequence of transformations performed on a data set during the data preparation phase of the model build process.

linear regression

The GLM regression algorithm supported by Oracle Data Mining.

logistic regression

The GLM classification algorithm supported by Oracle Data Mining.

MDL

See **Minimum Description Length**.

min-max normalization

Normalize numerical attributes using this transformation:

```
x_new = (x_old-min) / (max-min)
```

Minimum Description Length

Given a sample of data and an effective enumeration of the appropriate alternative theories to explain the data, the best theory is the one that minimizes the sum of

- The length, in bits, of the description of the theory
- The length, in bits, of the data when encoded with the help of the theory

This principle is used to select the attributes that most influence target value discrimination in **attribute importance**.

mining function

A major subdomain of data mining that shares common high level characteristics. The Oracle Data Mining programming interfaces support the following mining functions: **classification**, **regression**, **attribute importance**, **feature extraction**, and **clustering**. In both programming interfaces, anomaly detection is supported as classification.

mining model

An important function of data mining is the production of a model. A model can be a **supervised model** or an **unsupervised model**. Technically, a mining model is the result of building a model from mining settings. The representation of the model is specific to the algorithm specified by the user or selected by the DMS. A model can be used for direct inspection, for example, to examine the rules produced from an association model, or to score data.

mining object

Mining tasks, models, settings, and their components.

mining result

The end product(s) of a mining task. For example, a build task produces a mining model; a test task produces a test result.

missing value

A data value that is missing at random. It could be missing because it is unavailable, unknown, or because it was lost. Oracle Data Mining interprets missing values in columns with simple data types (not nested) as missing at random. Oracle Data Mining interprets missing values in nested columns as sparse.

Data mining algorithms vary in the way they treat missing values. There are several typical ways to treat them: ignore them, omit any records containing missing values, replace missing values with the mode or mean, or infer missing values from existing values. See also **sparse data**.

model

See **mining model**.

multi-record case

Each case in the data table is stored in multiple rows. Also known as **transactional data**. See also **single-record case**.

Naive Bayes

An algorithm for classification that is based on Bayes's theorem. Naive Bayes makes the assumption that each attribute is conditionally independent of the others: given a particular value of the target, the distribution of each predictor is independent of the other predictors.

nested data

Oracle Data Mining supports **transactional data** in nested columns of name/value pairs. Multidimensional data that expresses a one-to-many relationship can be loaded into a nested column and mined along with single-record case data in a case table.

NMF

See **Non-Negative Matrix Factorization**.

Non-Negative Matrix Factorization

A feature extraction algorithm that decomposes multivariate data by creating a user-defined number of features, which results in a reduced representation of the original data.

normalization

Normalization consists of transforming numerical values into a specific range, such as [–1.0,1.0] or [0.0,1.0] such that `x_new = (x_old-shift)/scale`. Normalization applies only to numerical attributes. Oracle Data Mining provides transformations that perform **min-max normalization**, **scale normalization**, and **z-score normalization**.

numerical attribute

An attribute whose values are numbers. The numeric value can be either an integer or a real number. Numerical attribute values can be manipulated as continuous values. See also **categorical attribute**.

O-Cluster

See **Orthogonal Partitioning Clustering**.

one-class Support Vector Machine

The version of the **Support Vector Machine** model used to solve **anomaly detection** problems. The Oracle Data Mining programmatic interfaces implement the one-class algorithm as classification.

Orthogonal Partitioning Clustering

An Oracle proprietary clustering algorithm that creates a hierarchical grid-based clustering model, that is, it creates axis-parallel (orthogonal) partitions in the input attribute space. The algorithm operates recursively. The resulting hierarchical structure represents an irregular grid that tessellates the attribute space into clusters.

outlier

A data value that does not come from the typical population of data; in other words, extreme values. In a normal distribution, outliers are typically at least 3 standard deviations from the mean.

positive target value

In binary classification problems, you may designate one of the two classes (target values) as positive, the other as negative. When Oracle Data Mining computes a model's lift, it calculates the density of positive target values among a set of test instances for which the model predicts positive values with a given degree of confidence.

predictive model

A predictive model is an equation or set of rules that makes it possible to predict an unseen or unmeasured value (the dependent variable or output) from other, known values (independent variables or input). The form of the equation or rules is suggested by mining data collected from the process under study. Some training or estimation technique is used to estimate the parameters of the equation or rules. A predictive model is a **supervised model**.

predictor

An attribute used as input to a supervised model or algorithm to build a model.

prepared data

Data that is suitable for model building using a specified algorithm. Data preparation often accounts for much of the time spent in a data mining project. Oracle Data Mining supports transformations binning, normalization, and missing value treatment. Oracle Data Mining can automatically perform algorithm-appropriate transformations when **Automatic Data Transformation** is enabled.

prior probabilities

The set of prior probabilities specifies the distribution of examples of the various classes in the original source data. Also referred to as *priors*, these could be different from the distribution observed in the data set provided for model build.

priors

See **prior probabilities**.

quantile binning

A numerical attribute is divided into bins such that each bin contains approximately the same number of cases.

random sample

A sample in which every element of the data set has an equal chance of being selected.

recode

Literally "change or rearrange the code." Recoding can be useful in many instances in data mining. Here are some examples:

- Missing values treatment: Missing values may be indicated by something other than `NULL`, such as "0000" or "9999" or "NA" or some other string. One way to treat the missing value is to recode, for example, "0000" to `NULL`. Then the Oracle Data Mining algorithms and the database recognize the value as missing.
- Change data type of variable: For example, change "Y" or "Yes" to 1 and "N" or "No" to 0.
- Establish a cutoff value: For example, recode all incomes less than $20,000 to the same value.

- Group items: For example, group individual US states into regions. The "New England region" might consist of ME, VT, NH, MA, CT, and RI; to implement this, recode the five states to, say, NE (for New England).

record

See **case**.

regression

A data mining function for predicting continuous target values for new records using a model built from records with known target values. Oracle Data Mining supports **linear regression** (GLM) and **Support Vector Machine** algorithms for regression.

rule

An expression of the general form *if X, then Y*. An output of certain algorithms, such as clustering, association, and decision tree. The predicate *X* may be a compound predicate.

sample

See **random sample**.

scale normalization

Normalize numerical attributes using this transformation:

```
x_new = (x_old - 0) / (max(abs(max),abs(min)))
```

schema

A collection of objects in an Oracle database, including logical structures such as tables, views, sequences, stored procedures, synonyms, indexes, clusters, and database links. A schema is associated with a specific database user.

score

Scoring data means applying a data mining model to data to generate predictions.

settings

See **algorithm settings**.

single-record case

Each case in the data table is stored in one row. Contrast with **multi-record case**.

sparse data

Data for which only a small fraction of the attributes are non-zero or non-null in any given case. Market basket data and text mining data are typically sparse. Oracle Data Mining interprets nested data as sparse. See also **missing value**.

split

Divide a data set into several disjoint subsets. For example, in a classification problem, a data set is often divided in to a training data set and a test data set.

stratified sample

Divide the data set into disjoint subsets (strata) and then take a random sample from each of the subsets. This technique is used when the distribution of target values is skewed greatly. For example, response to a marketing campaign may have a positive target value 1% of the time or less. A stratified sample provides the data mining algorithms with enough positive examples to learn the factors that differentiate positive from negative target values. See also **random sample**.

supermodel

Mining models that contain instructions for their own data preparation. Oracle Data Mining provides **Automatic Data Transformation** and embedded data transformation, which together provide support for supermodels.

supervised learning

See **supervised model**.

supervised model

A data mining model that is built using a known dependent variable, also referred to as the target. Classification and regression techniques are examples of supervised mining. See **unsupervised model**. Also referred to as **predictive model**.

Support Vector Machine

An algorithm that uses machine learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. Support vector machines can make predictions with sparse data, that is, in domains that have a large number of predictor columns and relatively few rows, as is the case with bioinformatics data. Support vector machine can be used for classification, regression, and anomaly detection.

SVM

See **Support Vector Machine**.

table

The basic unit of data storage in an Oracle database. Table data is stored in rows and columns.

target

In supervised learning, the identified attribute that is to be predicted. Sometimes called *target value* or *target attribute*. See also **attribute**.

text feature

A combination of words that captures important attributes of a document or class of documents. Text features are usually keywords, frequencies of words, or other document-derived features. A document typically contains a large number of words and a much smaller number of features.

text mining

Conventional data mining done using text features. Text features are usually keywords, frequencies of words, or other document-derived features. Once you derive text features, you mine them just as you would any other data. Both Oracle Data Mining and Oracle Text support text mining.

top N frequency binning

This type of binning bins categorical attributes. The bin definition for each attribute is computed based on the occurrence frequency of values that are computed from the data. The user specifies a particular number of bins, say N. Each of the bins bin_1,..., bin_N corresponds to the values with top frequencies. The bin bin_N+1 corresponds to all remaining values.

training data

See **build data**.

transactional data

The data for one case is contained in several rows. An example is market basket data, in which a case represents one basket that contains multiple items. Oracle Data Mining supports transactional data in nested columns of attribute name/value pairs. See also **nested data**, **multi-record case**, and **single-record case**.

transformation

A function applied to data resulting in a new representation of the data. For example, discretization and normalization are transformations on data.

trimming

A technique used for dealing with outliers. Trimming removes values in the tails of a distribution in the sense that trimmed values are ignored in further computations. This is achieved by setting the tails to `NULL`.

unstructured data

Images, audio, video, geospatial mapping data, and documents or text data are collectively known as unstructured data. Oracle Data Mining supports the mining of unstructured text data.

unsupervised learning

See **unsupervised model**.

unsupervised model

A data mining model built without the guidance (supervision) of a known, correct result. In supervised learning, this correct result is provided in the **target** attribute. Unsupervised learning has no such target attribute. Clustering and association are examples of unsupervised mining functions. See **supervised model**.

view

A view takes the output of a query and treats it as a table. Therefore, a view can be thought of as a stored query or a virtual table. You can use views in most places where a table can be used.

winsorizing

A way of dealing with outliers. Winsorizing involves setting the tail values of an particular attribute to some specified value. For example, for a 90% Winsorization, the bottom 5% of values are set equal to the minimum value in the 6th percentile, while the upper 5% are set equal to the maximum value in the 95th percentile.

z-score normalization

Normalize numerical attributes using this transformation:

```
x_new = (x_old-mean) / standard_deviation
```

Index