

**Oracle® Data Mining**

Concepts

11g Release 1 (11.1)

**B28129-01**

July 2007

Oracle Data Mining Concepts, 11g Release 1 (11.1)

B28129-01

Copyright © 2005, 2007, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

---

---

# Contents

<b>Preface</b> .....	vii
Audience .....	vii
Documentation Accessibility .....	vii
Related Documentation .....	viii
Conventions .....	viii
<b>What's New in Oracle Data Mining?</b> .....	ix
Oracle Data Mining 11g Release 1 (11.1) New Features .....	ix
Oracle Data Mining 10g Release 2 (10.2) New Features .....	xii
<b>Part I Introductions</b>	
<b>1 What Is Data Mining?</b>	
What Is Data Mining? .....	1-1
What Can Data Mining Do and Not Do? .....	1-3
The Data Mining Process .....	1-4
<b>2 Introducing Oracle Data Mining</b>	
Data Mining in the Database Kernel .....	2-1
Data Mining Functions .....	2-2
Data Mining Algorithms .....	2-4
Data Preparation .....	2-6
How Do I Use Oracle Data Mining? .....	2-7
Where Do I Find Information About Oracle Data Mining? .....	2-10
Oracle Data Mining and Oracle Database Analytics .....	2-11
<b>3 Introducing Oracle Predictive Analytics</b>	
About Predictive Analytics .....	3-1
Oracle Spreadsheet Add-In for Predictive Analytics .....	3-2
APIs for Predictive Analytics .....	3-4
Example: PREDICT .....	3-6
Behind the Scenes .....	3-6

## Part II Mining Functions

<b>4</b>	<b>Regression</b>	
	About Regression .....	4-1
	Regression Algorithms .....	4-3
	Testing a Regression Model .....	4-3
<b>5</b>	<b>Classification</b>	
	About Classification .....	5-1
	Classification Algorithms .....	5-2
	Biassing a Classification Model .....	5-2
	Testing a Classification Model .....	5-4
<b>6</b>	<b>Anomaly Detection</b>	
	About Anomaly Detection .....	6-1
	One-Class Classification .....	6-2
	Anomaly Detection Algorithm .....	6-2
<b>7</b>	<b>Clustering</b>	
	About Clustering .....	7-1
	Clustering Algorithms .....	7-1
<b>8</b>	<b>Association Rules</b>	
	About Association Rules .....	8-1
	Association Algorithm .....	8-2
	Data for Association Models .....	8-2
<b>9</b>	<b>Feature Selection and Extraction</b>	
	Feature Extraction .....	9-1
	Attribute Importance .....	9-1
<b>Part III Algorithms</b>		
<b>10</b>	<b>Apriori</b>	
	Association Rules and Frequent Item Sets .....	10-1
	Data Preparation for Association Rules .....	10-1
<b>11</b>	<b>Decision Tree</b>	
	About Decision Tree .....	11-1
	Tuning a Decision Tree Model .....	11-4
	Data Preparation for Decision Tree .....	11-4
<b>12</b>	<b>Generalized Linear Models</b>	
	About Generalized Linear Models .....	12-1
	Tuning and Diagnostics for GLM .....	12-3

	Data Preparation for GLM.....	12-5
	Linear Regression.....	12-6
	Logistic Regression.....	12-8
<b>13</b>	<b><i>k</i>-Means</b>	
	About <i>k</i> -Means.....	13-1
	Data Preparation for <i>k</i> -Means.....	13-2
<b>14</b>	<b>Minimum Description Length</b>	
	About MDL.....	14-1
	Data Preparation for MDL.....	14-2
<b>15</b>	<b>Naive Bayes</b>	
	About Naive Bayes.....	15-1
	Tuning a Naive Bayes Model.....	15-3
	Data Preparation for Naive Bayes.....	15-3
<b>16</b>	<b>Non-Negative Matrix Factorization</b>	
	About NMF.....	16-1
	NMF for Text Mining.....	16-1
	Data Preparation for NMF.....	16-2
<b>17</b>	<b>O-Cluster</b>	
	About O-Cluster.....	17-1
	O-Cluster Scoring.....	17-2
	Data Preparation for O-Cluster.....	17-2
<b>18</b>	<b>Support Vector Machines</b>	
	About Support Vector Machines.....	18-1
	Tuning an SVM Model.....	18-3
	Data Preparation for SVM.....	18-4
	SVM Classification.....	18-4
	One-Class SVM.....	18-5
	SVM Regression.....	18-5
<b>Part IV Data Preparation</b>		
<b>19</b>	<b>Automatic and Embedded Data Preparation</b>	
	Overview.....	19-1
	Automatic Data Preparation.....	19-4
	Embedded Data Preparation.....	19-6
	Transparency.....	19-10

**Part V Mining Unstructured Data**

## 20 Text Mining

Oracle Data Mining and Oracle Text .....	20-1
What is Text Mining? .....	20-1
Oracle Data Mining Support for Text Mining .....	20-2
Oracle Support for Text Mining .....	20-4

## 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 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

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

<http://www.oracle.com/accessibility/>

### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

### TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

## Related Documentation

The documentation set for Oracle Data Mining is part of the Oracle Database 11g Release 1 (11.1) Online Documentation Library. The Oracle Data Mining documentation set consists of the following manuals:

- *Oracle Data Mining Concepts*
- *Oracle Data Mining Application Developer's Guide*
- *Oracle Data Mining Java API Reference (javadoc)*
- *Oracle Data Mining Administrator's Guide*

---

---

**Note:** Information to assist you in installing and using the Data Mining demo programs is provided in *Oracle Data Mining Administrator's Guide*.

---

---

The syntax of the PL/SQL and SQL interfaces to Oracle Data Mining are documented in the following Database manuals:

- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database SQL Language Reference*

## Conventions

The following text conventions are used in this document:

Convention	Meaning
<b>boldface</b>	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	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 the new features of Oracle Data Mining 11g Release 1 (11.1) and provides pointers to additional information. Descriptions of new features in the previous release are also retained to help users migrating to the current release.

The following topics describe the new features in Oracle Data Mining:

- [Oracle Data Mining 11g Release 1 \(11.1\) New Features](#)
- [Oracle Data Mining 10g Release 2 \(10.2\) New Features](#)

## Oracle Data Mining 11g Release 1 (11.1) New Features

- **Mining Model schema objects**

In Oracle 11g, 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 11g, 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 11g, 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**. The embedded instructions can be implemented automatically and/or manually.

**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 11g, 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 11g. 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 11g. 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 5-2

■ **Desupported Features**

- DMSYS schema
- Oracle Data Mining Scoring Engine
- In Oracle 10.2, you could use Oracle Database Configuration Assistant (DBCA) to configure the Data Mining option. In Oracle 11g, you do not need to use DBCA to configure the Data Mining option.

■ **Deprecated Features**

- Adaptive Bayes Network classification algorithm (replaced with Decision Tree)
- Basic Local Alignment Search Tool (BLAST)
- 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*.

## Enhancements to the Oracle Data Mining Java API

The Oracle Data Mining Java API (OJDM) fully supports the new features in Oracle Data Mining 11g Release 1 (11.1). This section provides a summary of the new features in the Java API. For details, see *Oracle Data Mining Java API Reference* (Javadoc).

- As described in ["Mining Model schema objects"](#) on page -ix, mining models in 11g Release 1 (11.1) are data dictionary objects in the SYS schema. System and object privileges control access to mining models.

In the Oracle Data Mining Java API, a new extension method `OraConnection.getObjectNames` is added to support listing of mining objects that can be accessed by a user. This method provides various object filtering options that applications can use as needed.

- As described in ["Automatic Data Preparation \(ADP\)"](#) on page -ix, Oracle Data Mining 11g Release 1 (11.1) supports automatic and embedded data preparation (supermodels).

In the Oracle Data Mining Java API, a new build setting extension method, `OraBuildSettings.useAutomatedDataPreparations`, is added to enable ADP. Using the new `OraBuildTask.setTransformationSequenceName`, applications can embed the transformations with the model.

- Two new GLM packages are introduced: `oracle.dmt.jdm.algorithm.glm` and `oracle.dmt.jdm.modeldetail.glm`. These packages have GLM algorithm settings and model details interfaces respectively.
- New apply content enumeration values, `probabilityLowerBound` and `probabilityUpperBound`, are added to specify probability bounds for classification apply output. The enumeration `oracle.dmt.jdm.supervised.classification.OraClassificationApplyContent` specifies these enumerations. Similarly apply contents enumeration

values `predictionLowerBound` and `predictionUpperBound` are added to specify prediction bounds for regression model apply output. In this release only GLM models support this feature.

- New static methods `addCostMatrix` and `removeCostMatrix` are added to `OraClassificationModel` to support associating a cost matrix with the model. This will greatly ease the deployment of costs along with the model.
- Mining task features are enhanced to support the building of mining process workflows. Applications can specify dependent tasks using the new `OraTask.addDependency` method. Another notable new task feature is `overwriteOutput`, which can be enabled by calling the new `OraTask.overwriteOutput` method.

With these new features, applications can easily develop mining process workflows and deploy them to the database server. These task workflows can be monitored from the client side. For usage of these methods refer to the demo programs shipped with the product (See *Oracle Data Mining Administrator's Guide* for information about the demo programs.)

- A new mining object, `oracle.dmt.jdm.OraTransformationSequence` supports the specification of user-defined transformation sequences. These can either be embedded in the mining model or managed externally. In addition, the new `OraExpressionTransform` object can be used to specify SQL expressions to be included with the model.
- New `oracle.dmt.jdm.OraProfileTask` is added to support the new predictive analytics profile functionality.
- The Oracle Data Mining Java API can be used with Oracle Database 11g Release 1 (11.1) *and* with Oracle Database 10.2. When used with a 10.2 database, only the 10.2 features are available.

**See Also:** *Oracle Data Mining Java API Reference* and *Oracle Data Mining Application Developer's Guide*

## Oracle Data Mining 10g Release 2 (10.2) New Features

- **Java Data Mining (JDM) compliant Java API**

Oracle 10g Release 2 introduced a completely new Java API for Data Mining. The API implements JSR-000073, developed through the Java Community Process (<http://jcp.org>).

The new Java API is layered on the PL/SQL API, and the two APIs are fully interoperable. The new Java API is *not* compatible with the Java API available in the previous release (Oracle 10g Release 1).

- **SQL built-in functions for Data Mining**

New built-in SQL functions support the scoring of classification, regression, clustering, and feature extraction models. Within the context of standard SQL statements, pre-created models can be applied to new data and the results returned for further processing. The Data Mining SQL functions are:

- `PREDICTION`, `PREDICTION_COST`, `PREDICTION_DETAILS`,  
`PREDICTION_PROBABILITY`, `PREDICTION_SET`
- `CLUSTER_ID`, `CLUSTER_PROBABILITY`, `CLUSTER_SET`
- `FEATURE_ID`, `FEATURE_SET`, `FEATURE_VALUE`

- **Predictive Analytics**

Predictive Analytics automates the process of data mining. Without user intervention, Predictive Analytics routines manage data preparation, algorithm selection, model building, and model scoring.

In the `DBMS_PREDICTIVE_ANALYTICS` PL/SQL package, Oracle Data Mining provides Predictive Analytics routines that calculate predictions and determine the relative influence of attributes on the prediction.

Oracle Spreadsheet Add-In for Predictive Analytics implements `DBMS_PREDICTIVE_ANALYTICS` within the context of an Excel spreadsheet. The Spreadsheet Add-In is distributed on Oracle Technology Network.

- **New and enhanced algorithms**

- The new Decision Tree algorithm generates human-understandable rules for a prediction.
- The new One-Class Support Vector Machine algorithm supports anomaly detection.
- The Support Vector Machine algorithm is enhanced with active learning for the management of large build data sets.
- Both the PL/SQL and Java APIs support the O-Cluster algorithm. In Oracle 10g Release 1, O-Cluster was only supported in the Java API.



# 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"](#)





---

---

# 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?](#)
- [What Can Data Mining Do and Not Do?](#)
- [The Data Mining Process](#)

## What Is Data Mining?

Data mining is the practice of automatically searching large stores of data for 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 and applying them to data. The notion of automatic discovery refers to the application and deployment of data mining models. The process of applying a model is also called **scoring**.

## Prediction

Many forms of data mining are predictive. For example, a data mining model might predict income based on education, buying patterns, and various demographic factors. The model would be built using a complete set of data, including income figures and all the predictive attributes. The results of a predictive model can, and should be, compared with the original data. Predictions have an associated **confidence** (How confident can I be of this prediction?) and **support** (How frequently does an association appear in the data set? ).

## 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, 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 retailer might use the model described under "[Prediction](#)" to target the customers most likely to buy a new high-end product. A car leasing agency might use the model described under "[Grouping](#)" 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 2-4.

## 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 datasets. 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 be 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.

Oracle Data Mining requires that the data be presented as a case table in single-record case format. All the data for each record (case) must be contained within a row. Most typically, the case table is a view that presents the data in the required format for mining. The actual storage format of the data mapped to the view does not matter.

**See Also:** "Attributes and the Case Table" in *Oracle Data Mining Application Developer's Guide* for more information

## 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 necessarily *causes* of an action or behavior. 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 solutions 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 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.

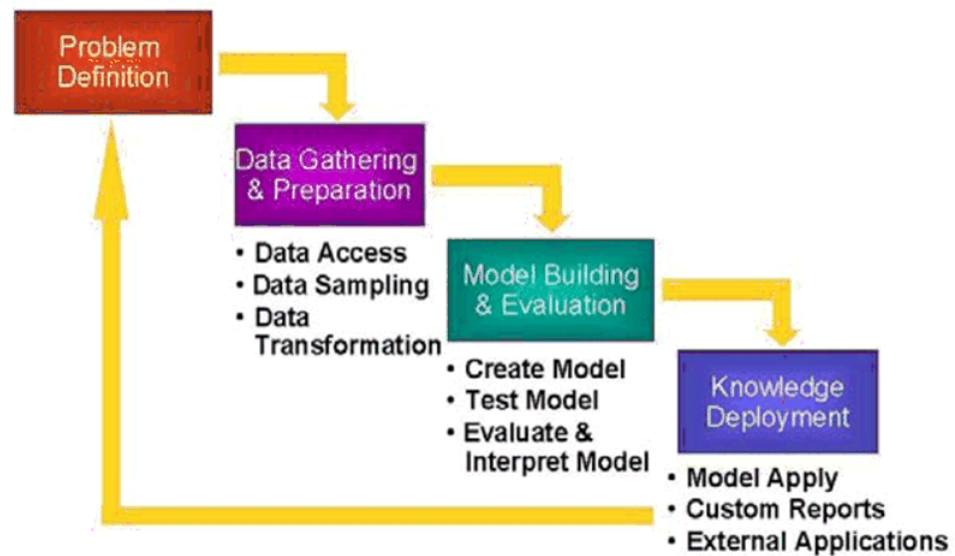
Oracle Data Mining can automatically perform much of the data preparation required by the algorithm. But some of the data preparation is typically specific to the domain or the data mining problem. At any rate, you need to understand the data that was used to build the model in order to properly interpret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

**See Also:** [Chapter 19, "Automatic and Embedded Data Preparation"](#)

## 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 and customers who have not purchased. Customer attributes might include age, number of children, own/rent, 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.

For data exploration and preliminary model building, it often makes sense to work with a reduced set of data, since the final case table might contain thousands or millions of cases.

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 Automatic Data Preparation, as described in [Chapter 19](#)).

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?

## Knowledge Deployment

Because Oracle Data Mining builds and applies data mining models inside Oracle Database, the results are immediately available to any database user with the proper privileges. Any BI reporting tool or dashboard can easily display the results of data mining.

Oracle Data Mining also supports real-time scoring. Data mining models can be applied and the results returned within a single SQL query. This enables analysis of transactional, point-of-sales data. 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:** "Scoring and Deployment" in *Oracle Data Mining Application Developer's Guide*

---

---

## 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](#)
- [Data Mining Functions](#)
- [Data Mining Algorithms](#)
- [Data Preparation](#)
- [How Do I Use Oracle Data Mining?](#)
- [Where Do I Find Information About Oracle Data Mining?](#)
- [Oracle Data Mining and Oracle Database Analytics](#)

### 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 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 and Java APIs and SQL language operators provide direct access to Oracle Data Mining functionality in Oracle Database.

**See Also:** ["Oracle Data Mining and Oracle Database Analytics"](#) on page 2-11 for a summary of additional analytics available within Oracle Database

## 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 the following section, ["Data Mining Algorithms"](#) on page 2-4.

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.

### **Build Data and Test Data for Supervised Learning**

Separate data sets are required for building (training) and testing a predictive model. 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.

### **Apply Data for Supervised Learning**

Apply data, also called scoring data, refers to 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 people who are most likely to be your best customers when you introduce a related product, you might apply the model to your customer database. In this case, the scoring data consists of your customer database.

Most supervised learning can be applied to a population of interest. Scoring is the purpose of classification and regression models. Oracle Data Mining does not support the scoring process for attribute importance models. 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. For example, a clustering model that seeks to divide a customer base into segments based on similarities in age, gender, and spending habits uses descriptive data mining techniques. However, unsupervised learning does not *imply* descriptive models. Unsupervised models can also be used to make predictions.

### **Build Data for Unsupervised Learning**

An unsupervised model returns information about the build data. A clustering model divides the data into clusters. A feature extraction model derives features from the build data. An association model returns rules about associations between items in the data. An anomaly detection model finds rare or unusual items in the data.

### **Apply Data for Unsupervised Learning**

Although unsupervised data mining does not specify a target, unsupervised models can be applied to a population of interest. For example, clustering models can be applied to classify cases according to their cluster assignments, and anomaly detection models can be applied to rank outliers. Anomaly detection, although unsupervised, is typically used to predict whether a data point is typical among a set of cases.

Most unsupervised learning can be applied to a population of interest. Oracle Data Mining supports the scoring operation for clustering and feature extraction models. Oracle Data Mining does not support the scoring operation for association models. 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 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

## 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.

---

**See Also:** [Part III, "Algorithms"](#) for more details about the algorithms supported by Oracle Data Mining.

## 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.
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 Machines (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.
<i>k</i> -Means (KM)	Clustering	<i>k</i> -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 <i>k</i> -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)	One class classification (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.

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

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 is 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.

## Supermodels

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"](#) for details.

## How Do I Use Oracle Data Mining?

Oracle Data Mining is an option to the Enterprise Edition of Oracle Database. It includes programmatic interfaces for SQL and Java and a spreadsheet add-in.

### 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. Users are free to use these routines or any other SQL-based method for defining transformations. The routines in `DBMS_DATA_MINING_TRANSFORM` are simply provided as a convenience.

Note that user-defined transformations are not required. Most transformations can be performed automatically by Oracle Data Mining.

- `DBMS_PREDICTIVE_ANALYTICS` — Contains automated data mining routines for `PREDICT`, `EXPLAIN`, and `PROFILE`.

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    => 'build_data',
    case_id_column_name => 'cust_id',
    target_column_name => 'affinity_card',
    settings_table_name => 'my_settings');
END;
/
```

## SQL Functions

The Data Mining functions are SQL language operators for the deployment of data mining models. They allow data mining 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

## 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.

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
//      (i.e., 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 (i.e., 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");
```

## 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/technology/products/bi/odm/index.html>.

### See Also:

- [Chapter 3, "Introducing Oracle Predictive Analytics"](#)
- *Oracle Data Mining Administrator's Guide* for installation instructions

## Where Do I Find Information About Oracle Data Mining?

Oracle Data Mining documentation is included in the Oracle Database online documentation library. Four manuals are dedicated to Oracle Data Mining. SQL and PL/SQL syntax for Oracle Data Mining is documented in Database manuals.

For your convenience, the Oracle Data Mining and related Oracle Database manuals are listed in [Table 2-5](#). The links in this table will take you directly to the referenced documentation.

**Table 2-5 Oracle Data Mining Documentation**

Document	Description
<i>Oracle Data Mining Concepts</i>	Overview of mining functions, algorithms, data preparation, predictive analytics, and other special features supported by Oracle Data Mining
<i>Oracle Data Mining Application Developer's Guide</i>	How to use the PL/SQL and Java APIs and the SQL operators for Data Mining
<i>Oracle Data Mining Administrator's Guide</i>	How to install and administer a database for Data Mining. How to install and use the demo programs
<i>Oracle Data Mining Java API Reference</i>	How to use the Oracle Data Mining Java API syntax (javadoc)
<i>Oracle Database PL/SQL Packages and Types Reference</i>	How to use the Oracle Data Mining PL/SQL syntax
<i>Oracle Database SQL Language Reference</i>	How to use the SQL Data Mining function (operator) syntax
<i>Oracle Database Reference</i>	How to query data dictionary views to obtain information about mining models, mining model attributes, and mining model settings

## 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/technology/products/bi/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:

- **White papers and web casts**
- **Demo programs** in SQL and Java that illustrate the Oracle Data Mining APIs
- **Oracle Spreadsheet Add-In for Predictive Analytics**, which you can download and import into Excel
- **Oracle Data Mining discussion forum** at <http://forums.oracle.com/forums/forum.jspa?forumID=55>
- **Blog on Data Mining and Analytics, with a special focus on Oracle**, at <http://oracledmt.blogspot.com/>

## Oracle Data Mining Publications

The following books, available on Amazon (<http://www.amazon.com/>), provide an excellent introduction to Oracle Data Mining. Both are based on Oracle Data Mining 10.2.

- *Java Data Mining: Strategy, Standard, and Practice*, (The Morgan Kaufmann Series in Data Management Systems), by Mark F. Hornick, Erik Marcadé, and Sunil Venkayala
- *Oracle Data Mining: Mining Gold from Your Warehouse*, (Oracle In-Focus series), by Dr. Carolyn Hamm

## Oracle Data Mining and Oracle Database Analytics

As described in "[Data Mining in the Database Kernel](#)" on page 2-1, the advantages of database analytics are considerable. 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 tools such as Oracle Discover and Oracle Portal. 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.

### **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);
```

The query in [Example 2–1](#) 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 Plus model called `tree_model`. The query uses the Oracle Text `CONTAINS` operator to search call center notes for references to Checking Plus accounts.

Some of the analytics supported by Oracle Database are described in [Table 2–6](#). Use the links in the Documentation column to find the referenced 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	Present document
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 <code>DBMS_DATA_MINING_TRANSFORM</code> package.  <code>DBMS_DATA_MINING_TRANSFORM</code> is a flexible data transformation package that includes a variety of missing value and outlier treatments, as well as binning and normalization capabilities.	<i>Oracle Database PL/SQL Packages and Types Reference</i>
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 <code>DBMS_STAT_FUNCS</code> package adds distribution fitting procedures and a summary procedure that returns descriptive statistics for a column.	<i>Oracle Database SQL Language Reference and Oracle Database PL/SQL Packages and Types Reference</i>
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 <code>SUM</code> , <code>AVERAGE</code> , <code>COUNT</code> , <code>MAX</code> , <code>MIN</code> , and many more functions.	<i>Oracle Database Data Warehousing Guide</i>
Frequent Itemsets	The <code>DBMS_FREQUENT_ITEMSET</code> supports frequent itemset counting, a mechanism for counting how often multiple events occur together. <code>DBMS_FREQUENT_ITEMSET</code> is used as a building block for the Association algorithm used by Oracle Data Mining.	<i>Oracle Database PL/SQL Packages and Types Reference</i>
Image feature extraction	Oracle Intermedia supports the extraction of image features such as color histogram, texture, and positional color. Image features can be used to characterize and analyze images.	<i>Oracle Multimedia User's Guide</i>
Linear algebra	The <code>UTL_NLA</code> package exposes a subset of the popular BLAS and LAPACK (Version 3.0) libraries for operations on vectors and matrices represented as <code>VARRAYs</code> . This package includes procedures to solve systems of linear equations, invert matrices, and compute eigenvalues and eigenvectors.	<i>Oracle Database PL/SQL Packages and Types Reference</i>

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

Analytical Feature	Description	Documentation
OLAP	<p>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.</p>	<i>Oracle OLAP User's Guide</i>
Spatial analytics	<p>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.</p> <p>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.</p>	<i>Oracle Spatial Developer's Guide</i>
Text Mining	<p>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</p>	<i>Oracle Text Application Developer's Guide</i>



---

---

## 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 Database PL/SQL Packages and Types Reference* for predictive analytics PL/SQL syntax.
- *Oracle Data Mining Java API Reference* (javadoc) for predictive analytics Java syntax.

This chapter includes the following sections:

- [About Predictive Analytics](#)
- [Oracle Spreadsheet Add-In for Predictive Analytics](#)
- [APIs for Predictive Analytics](#)
- [Example: PREDICT](#)
- [Behind the Scenes](#)

### 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. 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 3-7 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.

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

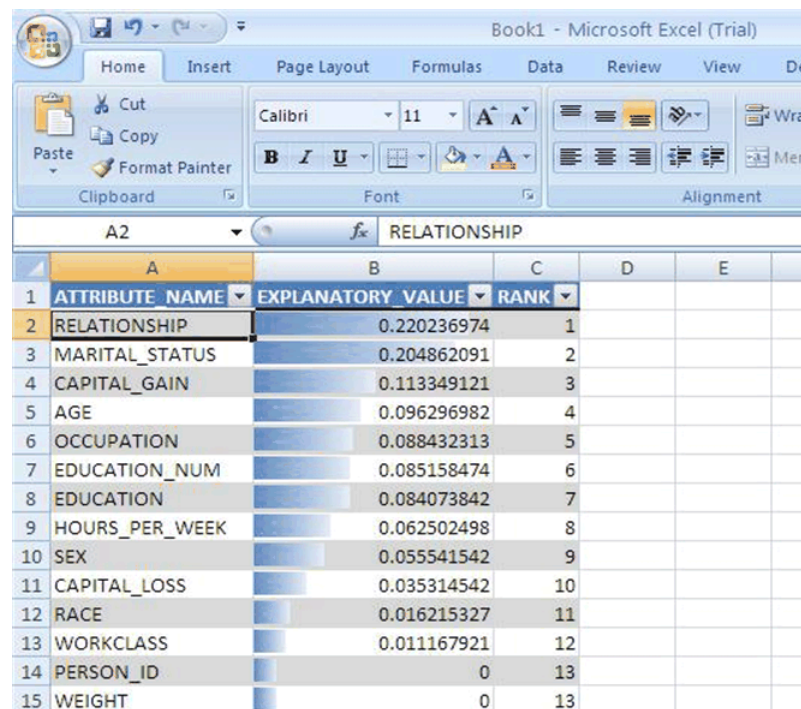
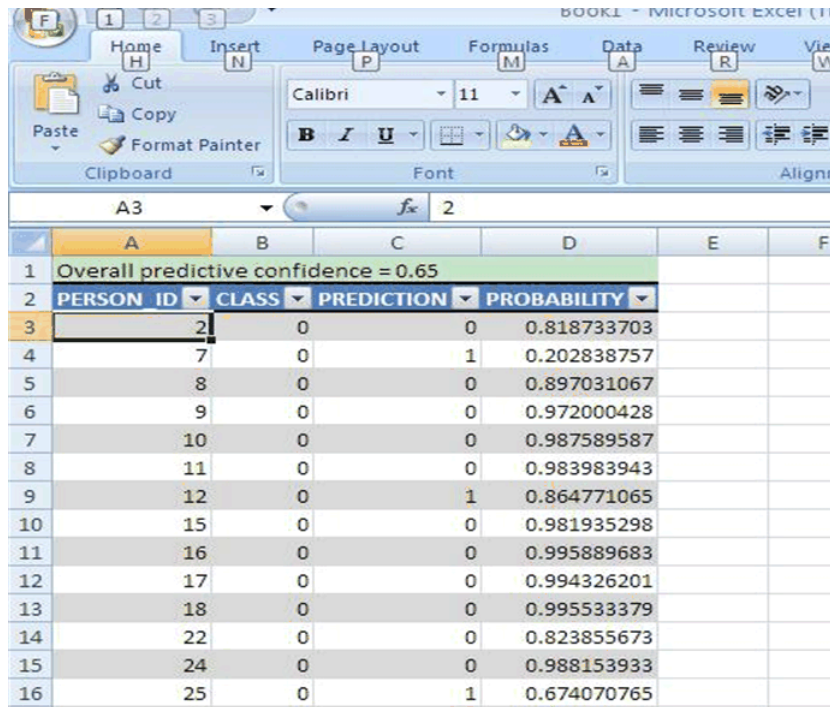


Figure 3–2 shows the PREDICT operation.

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



	A	B	C	D	E	F
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.

**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%
6	303	0	RELATIONSHIP isin ("NotinFa" "O-child" "Other-R" "Unmarr.") and CAPITAL_GAIN <= 7565.5 and EDUCATION isin ("Bach." "Masters" "PhD" "Profsc") RELATIONSHIP isin ("NotinFa" "O-child" "Other-R" "Unmarr.") and	10.9%	89.1%

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

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

## APIs for Predictive Analytics

Oracle Data Mining implements predictive analytics in the PL/SQL and Java APIs.

### Predictive Analytics in the PL/SQL API

DBMS\_PREDICTIVE\_ANALYTICS package. The following SQL DESCRIBE shows the predictive analytics procedures with their parameters.

```
SQL> describe dbms_predictive_analytics
```



PROCEDURE <b>EXPLAIN</b>			
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 <b>PREDICT</b>			
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 <b>PROFILE</b>			
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

## Predictive Analytics in the Java API

The Oracle Data Mining Java API defines predictive analytics tasks that use the `DBMS_PREDICTIVE_ANALYTICS` package.

The create methods that are defined under `oracle.dmt.jdm.task.OraPredictiveAnalyticsTaskFactory` class are shown as follows:

```
public OraPredictTask createPredictTask(String inputDataURI, String caseID,
                                         String targetColumn,
                                         String predictionResultTableName)
    throws JDMException;

public OraExplainTask createExplainTask(String inputDataURI,
                                         String explainColumn,
                                         String explainResultTableName)
    throws JDMException;

public OraProfileTask createProfileTask(String inputDataURI,
                                         String targetAttributeName,
                                         String profileResultTableName)
    throws JDMException;
```

### Example: Use OraProfileTask to Create Profile Results

```
//Step-1: Create OraPredictiveAnalyticsTaskFactory
OraPredictiveAnalyticsTaskFactory m_paFactory =
    (DataSetApplyTaskFactory)m_dmeConn.getFactory(
        "oracle.dmt.jdm.task.OraPredictiveAnalyticsTask");

//Step-2: Create, save and execute OraProfileTask. After
OraProfileTask m_profileTask =
    m_paFactory.createProfileTask("MINING_DATA_BUILD_V", "AFFINITY_CARD",
        "PROFILE_OUTPUT_JDM");
```

```
m_dmeConn.saveObject("profileTask_jdm", m_profileTask, true);
m_dmeConn.execute("profileTask_jdm");
```

## 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 "[Attribute Importance](#)" on page 9-1.

## PREDICT

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

PREDICT creates a Receiver Operating Characteristics (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 characteristics 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 specific data mining techniques. An algorithm appropriate to the mining function must be used to create a mining model. 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 applying mining models in SQL. See "Scoring and Deployment" in *Oracle Data Mining Application Developer's Guide*.

---

Part II contains the following chapters:

- [Chapter 4, "Regression"](#)
- [Chapter 5, "Classification"](#)
- [Chapter 6, "Anomaly Detection"](#)
- [Chapter 7, "Clustering"](#)
- [Chapter 8, "Association Rules"](#)
- [Chapter 9, "Feature Selection and Extraction"](#)



---

---

# Regression

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

**See Also:** ["Supervised Data Mining"](#) on page 2-2

This chapter includes the following topics:

- [About Regression](#)
- [Regression Algorithms](#)
- [Testing a Regression Model](#)

## About Regression

Regression is a data mining function that predicts a number. Age, weight, distance, temperature, income, or sales could all be predicted using regression techniques. For example, a regression model could be used to predict children's height, given their age, weight, 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 children's height could be developed based on observed data for many children over a period of time. The data might track age, height, weight, developmental milestones, family history, and so on. Height would be the target, the other attributes would be the predictors, and the data for each child 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. See ["Testing a Regression Model"](#) on page 4-3.

## Common Applications of Regression

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 quality regression models for data mining. However, it is helpful to understand a few basic concepts.

The goal of regression analysis is 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 ( $x_1, x_2, \dots, x_n$ ), a set of parameters ( $\theta_1, \theta_2, \dots, \theta_n$ ), and a measure of error ( $e$ ).

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

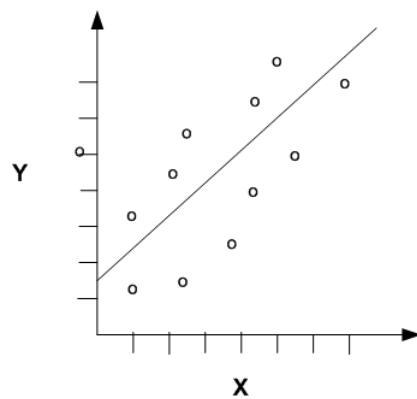
The process of training a regression model involves finding the best parameter values for the function that minimize a measure of the error, for example, the sum of squared errors.

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

### Linear Regression

The simplest form of regression to visualize is linear regression with a single predictor. A linear regression technique can be used if the relationship between  $x$  and  $y$  can be approximated with a straight line, as shown in [Figure 4-1](#).

**Figure 4-1** Linear Relationship Between  $x$  and  $y$



In a linear regression scenario with a single predictor ( $y = \theta_2x + \theta_1$ ), the regression parameters (also called coefficients) are:

The **slope** of the line ( $\theta_2$ ) — the angle between a data point and the regression line and

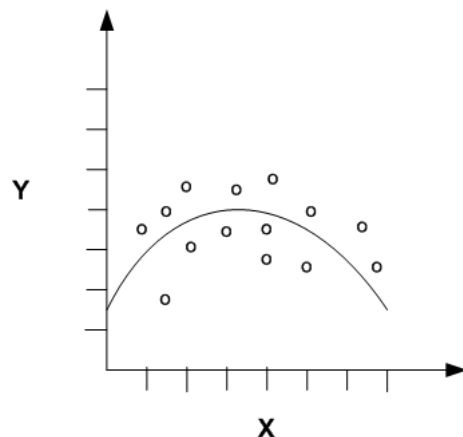
The **y intercept** ( $\theta_1$ ) — the point where  $x$  crosses the  $y$  axis ( $x = 0$ )

### 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.

In [Figure 4-2](#),  $x$  and  $y$  have a nonlinear relationship. Oracle Data Mining supports nonlinear regression via the gaussian kernel of SVM. (See "[Kernel-Based Learning](#)" on page 18-2.)



**Figure 4–2 Nonlinear Relationship Between  $x$  and  $y$** **Multivariate Regression**

Multivariate regression refers to regression with multiple predictors ( $x_1, x_2, \dots, x_n$ ). For purposes of illustration, [Figure 4–1](#) and [Figure 4–2](#) show regression with a single predictor. Multivariate regression is also referred to as **multiple regression**.

**Regression Algorithms**

Oracle Data Mining provides the following algorithms for regression:

- **Generalized Linear Models**

Generalized Linear Models (GLM) is a popular statistical technique for linear modeling. Oracle Data Mining implements GLM for regression and classification. See [Chapter 12, "Generalized Linear Models"](#)

- **Support Vector Machines**

Support Vector Machines (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. See [Chapter 18, "Support Vector Machines"](#)

---



---

**Note:** Both GLM and SVM, as implemented by Oracle Data Mining, are particularly suited for mining data that includes many predictors (wide data).

---



---

**Testing a Regression Model**

The Root Mean Squared Error and the Mean Absolute Error are 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. [Figure 4–3](#) shows the formula for the RMSE.

**Figure 4–3 Root Mean Squared Error**

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

This SQL expression calculates the RMSE.

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

## Mean Absolute Error

The Mean Absolute Error (MAE) is the average of the absolute value of the residuals. The MAE is very similar to the RMSE but is less sensitive to large errors. [Figure 4–4](#) shows the formula for the MAE.

**Figure 4–4 Mean Absolute Error**

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

This SQL expression calculates the MAE.

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

---

---

# Classification

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

**See Also:** ["Supervised Data Mining"](#) on page 2-2

This chapter includes the following topics:

- [About Classification](#)
- [Classification Algorithms](#)
- [Biasing a Classification Model](#)
- [Testing a Classification Model](#)

## 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.

A classification task begins with a data set in which the class assignments are known for each case. The classes are the values of the target. The classes are distinct and do not exist in an ordered relationship to each other. Ordered 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.

For example, customers might be classified as either users or non-users of a loyalty card. The predictors would be the attributes of the customers: age, gender, address, products purchased, and so on. The target would be yes or no (whether or not the customer used a loyalty card).

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. See ["Testing a Classification Model"](#) on page 5-4.

## Binary and Multiclass Targets

The simplest type of classification problem is binary classification. In binary classification, the target attribute has only two possible values: for example, good credit risk or poor credit risk.

Multiclass targets have more than two values: for example, occupations such as engineer, teacher, or lawyer.

## Common Applications of Classification

Classification is used in customer segmentation, business modeling, credit analysis, and many other applications. For example, a credit card company might use a classification model to predict which customers are likely to pay their entire credit card balance every month. A medical researcher might use a classification model to predict patient response to a drug.

## 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**  
Generalized Linear Models (GLM) is a popular statistical technique for linear modeling. Oracle Data Mining implements GLM in logistic regression for binary classification. See [Chapter 12, "Generalized Linear Models"](#).
- **Support Vector Machines**  
Support Vector Machines (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 (See ["Transparency"](#) on page 19-10). 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.

## Biassing a Classification Model

Cost and benefit matrices and prior probabilities are methods for biasing a classification model.

**See Also:** *Oracle Data Mining Application Developer's Guide* to learn how to implement costs, benefits, and priors using the APIs

## Cost/Benefit Matrix

In a classification problem, it is often important to specify the cost or benefit associated with correct or incorrect classifications. Doing so can be valuable when the cost of different misclassifications varies significantly.

You can create a cost matrix to bias the model to minimize cost or maximize benefit. The cost/benefit matrix is taken into consideration when the model is scored.

For example, suppose the problem is to predict whether a customer will respond to a promotional mailing. The target has two categories: YES (the customer responds) and NO (the customer does not respond). Suppose a positive response to the promotion generates \$500 and that it costs \$5 to do the mailing. After building the model, you compare the model predictions with actual data held aside for testing. At this point, you can evaluate the relative cost of different misclassifications.

- *If the model predicts YES and the actual value is YES, the cost of misclassification is \$0.*
- *If the model predicts YES and the actual value is NO, the cost of misclassification is \$5.*
- *If the model predicts NO and the actual value is YES, the cost of misclassification is \$495.*
- *If the model predicts NO and the actual value is NO, the cost is \$0.*

Table 5–1 shows these relationships summarized in a cost matrix table.

**Table 5–1 Cost Matrix**

Actual Target Value	Predicted Target Value	Cost
YES	YES	0
NO	YES	5
YES	NO	495
NO	NO	0

The cost matrix shown in Table 5–1 shows that the cost of misclassifying a non-responder (sending the mailing to someone who does not respond) is only \$5.00, the cost of the mailing. However, the cost of misclassifying a responder (*not* sending the mailing to someone who *would have* responded) is \$495.00, because you will lose \$500.00 while only saving the cost of the mailing.

Using the same costs shown in Table 5–1, you can approach the relative value of outcomes from a benefits perspective. When you correctly predict a YES (a responder), the benefit is \$495. When you correctly predict a NO (a non-responder), the benefit is \$5.00 since you can avoid sending out the mailing. As the goal is to find the lowest cost solution, benefits would be represented as negative numbers in the matrix, as shown in Table 5–2.

**Table 5–2 Cost/Benefit Matrix**

Actual Target Value	Predicted Target Value	Cost/Benefit
YES	YES	-495
NO	YES	0
YES	NO	0
NO	NO	-5

**See Also:** "Cost-Sensitive Decision Making" in *Oracle Data Mining Application Developer's Guide*

## Priors

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

---

**Note:** Priors are not used by the Decision Tree algorithm or logistic regression.

SVM classification uses priors as weights. See "[Class Weights](#)" on page 18-5.

---

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 a model.

To correct for unrealistic distributions in the training data, you can specify priors for the model build process. [Table 5-3](#) shows a sample priors table that specifies a prior probability of 25% for a target value of 0 and 75% for a target of 1. This means that the ratio of 0 to 1 in the actual population is typically about 1 to 3.

**Table 5-3 Priors Table**

Target Value	Prior Probability
0	25
1	75

---

**Note:** The model should be tested against data that has the actual target values.

---

## 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.

## Confusion Matrix

A confusion matrix summarizes the types of errors that a classification model is likely to make. The confusion matrix is calculated by applying the model to test data in which the target values are already known. These target values are compared with the predicted target values.

A confusion matrix is a square with  $n$  dimensions, where  $n$  is the number of target classes. For example, a multiclass classification model with the target values `small`,

medium, and large would have a three-by-three confusion matrix. A binary classification model has a two-by-two confusion matrix.

The rows of a confusion matrix identify the known target values. The columns indicate the predicted values.

Figure 5–1 shows a confusion matrix for a binary classification model. The target values are either `buyer` or `non-buyer`. In this example, the model correctly predicted a buyer 516 times and incorrectly predicted a buyer 10 times. The model correctly predicted a non-buyer 725 times and incorrectly predicted a non-buyer 25 times.

**Figure 5–1 Sample Confusion Matrix**

		Predicted	
		Buyer	Non-Buyer
Actual	Buyer	516	25
	Non-Buyer	10	725

The following can be computed from this confusion matrix:

- The model made 1241 correct predictions ( $516 + 725$ ).
- The model made 35 incorrect predictions ( $25 + 10$ ).
- The model scored 1276 cases ( $1241+35$ ).
- The error rate is  $35/1276 = 0.0274$ .
- The accuracy rate is  $1241/1276 = 0.9725$ .

## Lift

Lift measures the concentration of positive predictions within segments of the population and specifies the improvement over the rate of positive predictions in the population as a whole.

Lift is commonly used to measure the performance of targeting models in marketing applications. The purpose of a targeting model is to identify segments of the population with potentially high concentrations of positive responders to a marketing campaign. Lift is the ratio of positive responders in a segment to the positive responders in the population as a whole. For example, if a population has a predicted response rate of 20%, but one segment of the population has a predicted response rate of 60%, then the lift of that segment is 3 ( $60\%/20\%$ ).

The notion of lift implies a binary target: either a responder or not a responder, either yes or no. Lift can be computed for multiclass targets by designating a preferred positive class and combining all other target class values, effectively turning a multiclass target into a binary target.

The calculation of lift begins by applying the model to test data in which the target values are already known. Then the predicted results are sorted in order of probability, from highest to lowest predictive confidence. The ranked list is divided into quantiles (equal parts). The default number of quantiles is 10.

Oracle Data Mining computes the following lift statistics:

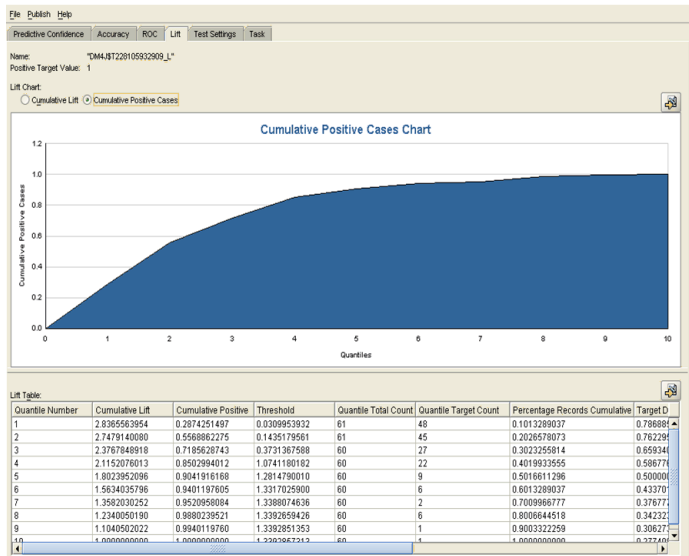
- **Probability threshold** for a quantile  $n$  is the minimum probability for the positive target to be included in this quantile or any preceding quantiles (quantiles  $n-1$ ,

$n-2, \dots, 1$ ). 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.

- **Cumulative gain** for a quantile 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 test 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.

The sample lift chart in Figure 5–2 shows that the cumulative lift for the top 30% is 2.37. The next column indicates that over 71% of all likely positive responses are found in the top 3 quantiles.

**Figure 5–2 Sample Lift Chart**



## Receiver Operating Characteristic (ROC)

ROC is a method for experimenting with changes in the probability threshold and observing the resulting effect on the predictive power of the model.

ROC curves are similar to lift charts in that they provide a means of comparison between individual models and determine thresholds which yield a high proportion of



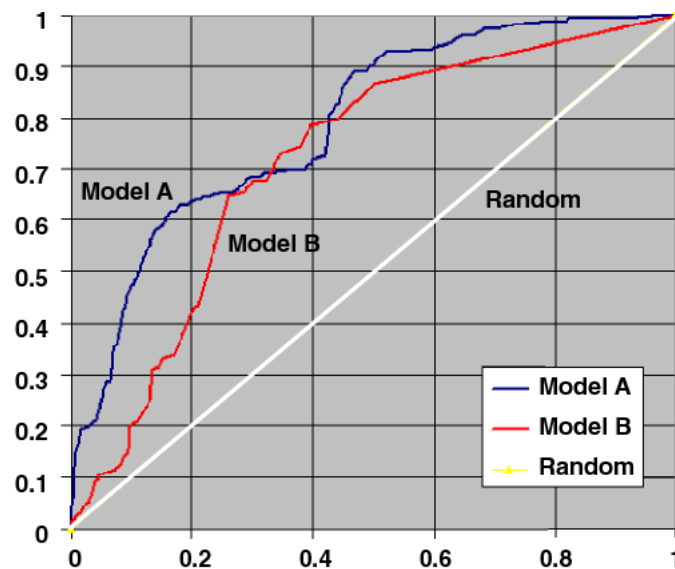
positive hits. ROC was originally used in signal detection theory to gauge the true hit versus false alarm ratio when sending signals over a noisy channel.

The horizontal axis of an ROC graph measures the false positive rate as a percentage. The vertical axis shows the true positive rate. The top left hand corner is the optimal location in an ROC curve, indicating high TP (true-positive) rate versus low FP (false-positive) rate.

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).

In the example graph in [Figure 5–3](#), Model A clearly has a higher AUC for the entire data set. However, if a false positive rate of 40% is acceptable, Model B is better suited, since it achieves a better error true positive rate at that false positive rate.

**Figure 5–3 Receiver Operating Characteristics Curves**



Besides model selection the ROC also helps to determine a threshold value to achieve an acceptable trade-off between hit (true positives) rate and false alarm (false positives) rate. By selecting a point on the curve for a given model a given trade-off is achieved. This threshold can then be used as a post-processing parameter for achieving the desired performance with respect to the error rates. Data Mining models by default use a threshold of 0.5.

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 (`true_positive_fraction`) and different false alarm rates (`false_positive_fraction`).
- **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. ( $\text{true positives} / (\text{true positives} + \text{false negatives})$ )
- **False positive fraction:** False alarm rate. ( $\text{false positives} / (\text{false positives} + \text{true negatives})$ )

---

---

## 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 2-3.

This chapter contains the following sections:

- [About Anomaly Detection](#)
- [One-Class Classification](#)
- [Anomaly Detection Algorithm](#)

### About Anomaly Detection

The goal of anomaly detection is to identify cases that are unusual. Anomaly detection is an important tool for detecting fraud, network intrusion, and other rare events that are significant but 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 counterexamples.

- 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.

### Counter-examples

Counter-examples are cases that do not fall within a given class. Sometimes examples are easy to find, but counter-examples are either hard to specify or expensive to collect. For example, in text document classification, it is easy to classify a document under a given topic. However, the universe of documents not belonging to this topic can be very large and it may not be feasible to provide counter-examples

### 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 \$60,000 and a mean household income of \$70,000, but one or two

households might have an income of \$10,000. 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 training data. Each case can be ranked according to the probability that it is either typical or atypical.

## 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 counterexamples 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 the training data. Any 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 is difficult, and 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 Algorithm

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

**See Also:**

- [Chapter 18, "Support Vector Machines"](#)
- ["SVM Classification"](#) on page 18-4
- ["One-Class SVM"](#) on page 18-5

---

---

# Clustering

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

**See Also:** ["Unsupervised Data Mining"](#) on page 2-3

This chapter includes the following topics:

- [About Clustering](#)
- [Clustering Algorithms](#)

## About Clustering

A cluster is a collection of data objects that are similar in some sense to one another. Clustering analysis identifies clusters in the data.

A good clustering method produces high-quality clusters to ensure that the inter-cluster similarity is low and the intra-cluster similarity is high; in other words, members of a cluster are more like each other than they are like members of a different cluster.

Clustering is useful for exploring data. If there are many cases and no obvious natural groupings, clustering data mining algorithms can be used to find natural groupings. Clustering can also serve as a useful data-preprocessing step to identify homogeneous groups on which to build supervised models.

Clustering models are different from supervised models in that the outcome of the process is not guided by a known result, that is, there is no target attribute. Clustering models focus on the intrinsic structure, relations, and interconnectedness of the data. Clustering models are built using optimization criteria that favor high intra-cluster and low inter-cluster similarity. The model can then be used to assign cluster identifiers to data points.

In Oracle Data Mining, a cluster is characterized by its centroid, attribute histograms, and the cluster's place in the model's hierarchical tree. A centroid represents the most typical case in a cluster. For numerical clusters, the centroid is the mean. For categorical clusters, the centroid is the mode.

## Clustering Algorithms

Oracle Data Mining performs hierarchical clustering using an enhanced version of the *k*-means algorithm and Orthogonal Partitioning Clustering algorithm (O-Cluster), an Oracle proprietary algorithm.

The clusters discovered by these algorithms are used to create rules that capture the main characteristics of the data assigned to each cluster. The rules represent the bounding boxes that envelop the data in the clusters discovered by the clustering algorithm. The antecedent of each rule describes the clustering bounding box. The consequent encodes the cluster ID for the cluster described by the rule. For example, for a data set with two attributes: AGE and HEIGHT, the following rule represents most of the data assigned to cluster 10:

If AGE >= 25 and AGE <= 40 and HEIGHT >= 5.0ft and HEIGHT <= 5.5ft then  
CLUSTER = 10

The clusters are also used to generate a Bayesian probability model, which is used during scoring for assigning data points to clusters.

The main characteristics of the enhanced *k*-means and O-Cluster algorithms are summarized in [Table 7-1](#).

**Table 7-1 Clustering Algorithms Compared**

Feature	Enhanced <i>k</i> -Means	O-Cluster
Clustering methodology	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
Recommended data preparation	Normalization	Equi-width binning after clipping

**See Also:**

[Chapter 17, "O-Cluster"](#)

[Chapter 13, "k-Means"](#)

---

---

## Association Rules

This chapter describes the association mining function used for market basket analysis. Association, also known as **association rules**, is an unsupervised mining function.

**See Also:** ["Unsupervised Data Mining"](#) on page 2-3

This chapter contains the following topics:

- [About Association Rules](#)
- [Association Algorithm](#)
- [Data for Association Models](#)

### About Association Rules

An Association model is often used for market basket analysis, which attempts to discover relationships or correlations in a set of items. Market basket analysis is widely used in data analysis for direct marketing, catalog design, and other business decision-making processes. A typical association rule of this kind asserts that, for example, "70% of the people who buy spaghetti, wine, and sauce also buy garlic bread."

Association models capture the co-occurrence of items or events in large volumes of customer transaction data. Because of progress in bar-code technology, it is now possible for retail organizations to collect and store massive amounts of sales data. Association models were initially defined for such sales data, even though they are applicable in several other applications. Finding association rules is valuable for cross-marketing and mail-order promotions, but there are other applications as well: catalog design, add-on sales, store layout, customer segmentation, web page personalization, and target marketing.

Traditionally, association models are used to discover business trends by analyzing customer transactions. However, they can also be used effectively to predict Web page accesses for personalization. For example, assume that after mining the Web access log, Company X discovered an association rule "A and B implies C," with 80% confidence, where A, B, and C are Web page accesses. If a user has visited pages A and B, there is an 80% chance that he/she will visit page C in the same session. Page C may or may not have a direct link from A or B. This information can be used to create a dynamic link to page C from pages A or B so that the user can "click-through" to page C directly. This kind of information is particularly valuable for a Web server supporting an e-commerce site to link the different product pages dynamically, based on the customer interaction.

There are several properties of association models that can be calculated. Oracle Data Mining calculates the following two properties related to rules:

- **Support:** Support of a rule is a measure of how frequently the items involved in it occur together. Using probability notation, support (A implies B) =  $P(A, B)$ .
- **Confidence:** Confidence of a rule is the conditional probability of B given A; confidence (A implies B) =  $P(B \text{ given } A)$ .

These statistical measures can be used to rank the rules and hence the predictions.

## Difficult Cases for Associations

The Apriori algorithm works by iteratively enumerating item sets of increasing lengths subject to the minimum support threshold. Since state-of-the-art algorithms for associations work by iterative enumeration, association rules algorithms do *not* handle the following cases efficiently:

- Finding associations involving rare events
- Finding associations in data sets that are dense and that have a large number of attributes.

### Finding Associations Involving Rare Events

Association mining 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 item sets, especially in cases with large number of items. That could increase the execution time significantly.

Therefore, association rule mining is not recommended for finding associations involving rare events in problem domains with a large number of items.

One option is to use classification models in such problem domains.

## Association Algorithm

Oracle Data Mining uses the Apriori algorithm for association models.

**See Also:** [Chapter 10, "Apriori"](#)

## Data for Association Models

Association models are designed to use **sparse data**. Sparse data is data for which only a small fraction of the attributes are nonzero or non-null in any given row. Examples of sparse data include market basket and text mining data. For example, in a market basket problem, there might be 1,000 products in the company's catalog, and the average size of a basket (the collection of items that a customer purchases in a typical transaction) might be 20 products. In this example, a transaction/case/record has on average 20 out of 1000 attributes that are not null. This implies that the fraction of nonzero attributes on the table (or the density) is 20/1000, or 2%. This density is typical for market basket and text processing problems. Data that has a significantly higher density can require extremely large amounts of temporary space to build associations.

Association models treat NULL values as sparse data. The algorithm does not handle missing values. If the data is not sparse and the NULL values are indeed missing at



random, you should perform missing data imputation (that is, treat the missing values) and substitute non-null values for the NULL value.

The presence of outliers, when external equal-width binning is used, makes most of the data concentrate in a few bins (a single bin in extreme cases). As a result, the ability of the model to detect differences in numerical attributes may be significantly lessened. For example, a numerical attribute such as income may have all the data belonging to a single bin except for one entry (the outlier) that belongs to a different bin. As a result, there won't be any rules reflecting different levels of income. All rules containing income will only reflect the range in the single bin; this range is basically the income range for the whole population. In cases like this, use a clipping transformation to handle outliers.



---

---

## Feature Selection and Extraction

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

**See Also:**

["Supervised Data Mining"](#) on page 2-2

["Unsupervised Data Mining"](#) on page 2-3

This chapter contains the following sections:

- [Feature Extraction](#)
- [Attribute Importance](#)

### Feature Extraction

Feature Extraction creates a set of features based on the original data. A **feature** is a combination of attributes that is of special interest and captures important characteristics of the data. It becomes a new attribute. Typically, there are far fewer features than there are original attributes.

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.

For example, 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.

### Feature Extraction Algorithm

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

**See Also:** [Chapter 16, "Non-Negative Matrix Factorization"](#)

### Attribute Importance

Attribute Importance provides an automated solution for improving the speed and possibly the accuracy of classification models built on data tables with a large number of attributes.

The time required to build classification models increases with the number of attributes. Attribute Importance identifies a proper subset of the attributes that are most relevant to predicting the target. Model building can proceed using the selected attributes only.

Using fewer attributes does not necessarily result in lost predictive accuracy. Using too many attributes (especially those that are "noise") can affect the model and degrade its performance and accuracy. Mining using the smallest number of attributes can save significant computing time and may build better models.

The programming interfaces for Attribute Importance permit the user to specify a number or percentage of attributes to use; alternatively the user can specify a cutoff point.

## Data Preparation for Attribute Importance

Attribute importance models typically benefit from binning, to minimize the effect of outliers. 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.

## Attribute Importance Algorithm

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

**See Also:** [Chapter 14, "Minimum Description Length"](#)

# 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:

- [Chapter 10, "Apriori"](#)
- [Chapter 11, "Decision Tree"](#)
- [Chapter 12, "Generalized Linear Models"](#)
- [Chapter 13, "k-Means"](#)
- [Chapter 14, "Minimum Description Length"](#)
- [Chapter 15, "Naive Bayes"](#)
- [Chapter 16, "Non-Negative Matrix Factorization"](#)
- [Chapter 17, "O-Cluster"](#)
- [Chapter 18, "Support Vector Machines"](#)



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

**See Also:** [Chapter 8, "Association Rules"](#)

This chapter contains the following topics:

- [Association Rules and Frequent Item Sets](#)
- [Data Preparation for Association Rules](#)

## Association Rules and Frequent Item Sets

Associations are calculated using the Apriori algorithm. The association mining problem can be decomposed into two subproblems:

- Find all combinations of items, called frequent itemsets, whose support is greater than the specified minimum support.
- Use the frequent itemsets to generate the desired rules. Oracle Data Mining association supports single consequent rules only (for example, "ABC implies D").

The number of frequent itemsets is controlled by the minimum support parameters. The number of rules generated is controlled by the number of frequent itemsets and the confidence parameter. If the confidence parameter is set too high, there may be frequent itemsets in the association model but no rules.

## Data Preparation for Association Rules

When Apriori uses equi-width binning, outliers cause most of the data to concentrate in a few bins, sometimes a single bin. As a result, the discriminating power of these algorithms can be significantly reduced.

Similarly, an association model might have all the values of a numerical attribute concentrated in a single bin, except for one value (the outlier) that belongs to a different bin. If, for example, this attribute is income, there will not be any rules reflecting different levels of income. All rules containing income will only reflect the range in the single bin; this range is basically the income range for the whole population.

Similarly, an association model might have all the values of a numerical attribute concentrated in a single bin, except for one value (the outlier) that belongs to a different bin. If, for example, this attribute is income, there will not be any rules reflecting different levels of income. All rules containing income will only reflect the

range in the single bin; this range is basically the income range for the whole population.



---

---

## Decision Tree

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

**See:** [Chapter 5, "Classification"](#)

This chapter contains the following topics:

- [About Decision Tree](#)
- [Tuning a Decision Tree Model](#)
- [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 accuracy of a prediction is the only thing that really matters. It may not be important to know how the model works. 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.

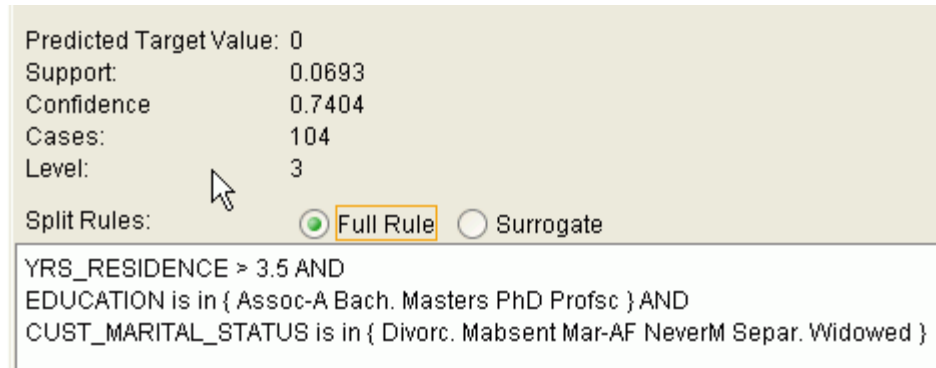
### Decision Tree Rules

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**



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).

## 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**

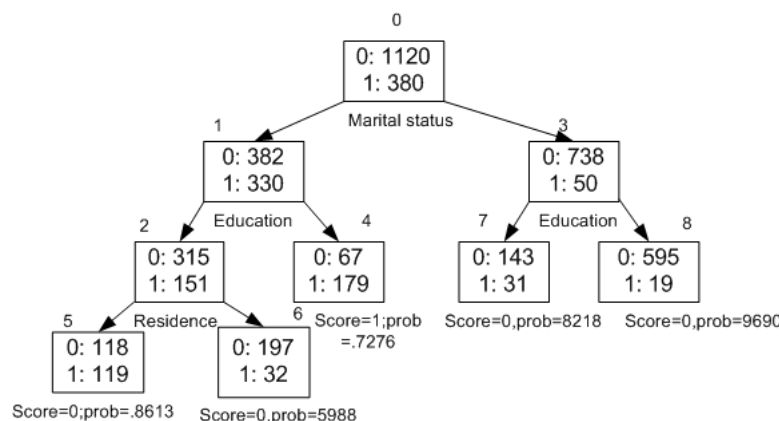


Figure 11–2 is a decision tree with nine nodes (and nine corresponding rules). The target attribute is binary: 1 if the customer will increase spending, 0 if the customer will not increase spending. The first split in the tree is based on the `CUST_MARITAL_STATUS` attribute. The root of the tree (node 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 assesses 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 "[Cost/Benefit Matrix](#)" on page 5-3 and "[Priors](#)" on page 5-4.

### 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.

### 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 at <http://www.dmg.org>.

### Tuning a Decision Tree Model

The Decision Tree algorithm is implemented with reasonable defaults for splitting and termination criteria. It is unlikely that you will need to use any of the build settings that are supported for Decision Tree. The settings are described as follows.

Settings to specify the homogeneity metric for finding the optimal split condition:

- `TREE_IMPURITY_METRIC` can be either gini or entropy. The default is gini.

Settings to control the growth of the tree:

- `TREE_TERM_MAX_DEPTH` specifies the maximum depth of the tree, from root to leaf inclusive. The default is 7.
- `TREE_TERM_MINPCT_MODE` specifies the minimum number of cases required in a child node, expressed as a percentage of the rows in the training data. The default is .05%.
- `TREE_TERM_MINPCT_SPLIT` specifies the minimum number of cases required in a node in order for a further split to be possible. Expressed as a percentage of all the rows in the training data. The default is 1%.
- `TREE_TERM_MINREC_MODE` specifies the minimum number of cases required in a child node. Default is 10.
- `TREE_TERM_MINREC_SPLIT` specifies the minimum number of cases required in a node in order for a further split to be possible. Default is 20.

**See:** *Oracle Database PL/SQL Packages and Types Reference* for specifics.

### Data Preparation for Decision Tree

The Decision Tree algorithm manages its own data preparation internally. It does not require pretreatment of the data.

The Decision Tree algorithm ignores the `PREP_AUTO` setting, which can be used to enable or disable Automatic Data Preparation.

---

---

**Note:** The Decision Tree algorithm does not support nested tables. All attributes must have simple character or numeric data types.

---

---



---

---

## 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 19-10

### 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 over-ride 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 Corporation 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 Corporation 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 Bayesian information criterion
- Estimated mean square error of the prediction
- Hocking  $S_p$  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

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.

### 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 and 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





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](#)
- [Data Preparation for k-Means](#)

## About k-Means

The *k*-Means algorithm is 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. The distance metric is either Euclidean, Cosine, or Fast Cosine distance. Data points are assigned to the nearest cluster according to the distance metric used.

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

- The algorithm builds models in a hierarchical manner. The algorithm builds a model top down 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 centroid of the inner nodes in the hierarchy are updated to reflect changes as the tree evolves. The whole tree is returned.
- The algorithm grows the tree one node at a time (unbalanced approach). Based on a user setting available in either of the programming interfaces, the node with the largest variance is split to increase the size of the tree until the desired number of clusters is reached.
- The algorithm provides probabilistic scoring and assignment of data to clusters.
- The algorithm returns, for each cluster, a centroid (cluster prototype), histograms (one 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 or 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.

## Scoring k-Means Clustering Models

The clusters discovered by enhanced *k*-Means are used to generate a Bayesian probability model that is then used during scoring (model apply) for assigning data points to clusters. The *k*-means algorithm can be interpreted as a mixture model where the mixture components are spherical multivariate normal distributions with the same variance for all components.

## Data Preparation for k-Means

The Oracle Data Mining implementation of *k*-Means supports both categorical and numerical data.

For numerical attributes, data normalization is recommended.

For the *k*-Means algorithm, NULL values indicate sparse data. Missing values are not automatically handled. If the data is not sparse and the values are indeed missing at random, you should perform missing data imputation (that is, perform some kind of missing values treatment) and substitute a non-NULL value for the NULL value. One simple way to treat missing values is to use the mean for numerical attributes and the mode for categorical attributes. If you do not treat missing values, the algorithm will not handle the data correctly.

Outliers with equi-width binning can prevent k-Means from creating clusters that are different in content. The clusters may have very similar centroids, histograms, and rules.

---

---

## Minimum Description Length

This chapter describes Minimum Description Length, the supervised technique 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. MDL assumes that the simplest, most compact representation of data is the best and most probable explanation of the data. The MDL principle is used to build Oracle Data Mining attribute importance models.

MDL considers each attribute as a simple predictive model of the target class. These single predictor models are compared and ranked with respect to the MDL metric (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.

With MDL, the model selection problem is treated as a communication problem. There is a sender, a receiver, and data to be transmitted. For classification models, the data to be transmitted is a model and the sequence of target class values in the training data.

Attribute importance uses a two-part code to transmit the 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,  $n_i$  ( $i=1, \dots, k$ ) rows per value, there are  $C_i$ , the combination of  $j-1$  things taken  $n_i-1$  at time possible conditional probabilities. The size of the preamble in bits can be shown to be  $\text{Sum}(\log_2(C_i))$ , 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  $\text{Sum}(\log_2(p_i))$ , 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

When these algorithms use equi-width binning, outliers cause most of the data to concentrate in a few bins, sometimes a single bin. As a result, the discriminating power of these algorithms can be significantly reduced.

Similarly, an association model might have all the values of a numerical attribute concentrated in a single bin, except for one value (the outlier) that belongs to a different bin. If, for example, this attribute is income, there will not be any rules reflecting different levels of income. All rules containing income will only reflect the range in the single bin; this range is basically the income range for the whole population.

---

---

## Naive Bayes

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

**See:** [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.

---

---

$$\text{Bayes' Theorem: } \text{Prob}(B \text{ given } A) = \text{Prob}(A \text{ and } B) / \text{Prob}(A)$$

---

---

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:

$$\text{Prob}(A \text{ and } B) = 25\%$$

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

$$\text{Prob}(A) = 75\%$$

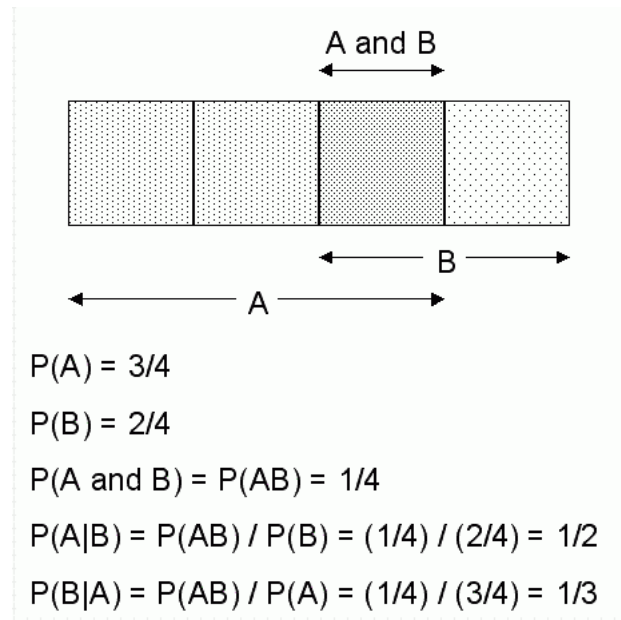
Bayes' Theorem would predict that 33% of customers under 21 are likely to increase spending (25/75).

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**



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 .

## 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: `NABS_PAIRWISE_THRESHOLD` and `NABS_SINGLETON_THRESHOLD`. The default thresholds work well for most models, so you will not generally need to adjust them.

- `NABS_PAIRWISE_THRESHOLD` — The minimum percentage of pairwise occurrences required for including a predictor in the model.
- `NABS_SINGLETON_THRESHOLD` — The minimum percentage of singleton occurrences required for including a predictor in the model.

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

## Data Preparation for Naive Bayes

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).

You can use the Automatic Data Preparation (ADP) feature of Oracle Data Mining to perform the binning. ADP uses a technique called **supervised binning** for Naive Bayes models. This technique uses decision trees to create the optimal bin boundaries.

When NB uses equi-width binning, outliers cause most of the data to concentrate in a few bins, sometimes a single bin. As a result, the discriminating power of these algorithms can be significantly reduced.





---

---

## Non-Negative Matrix Factorization

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

**See Also:** [Chapter 9, "Feature Selection and Extraction"](#)

This chapter contains the following topics:

- [Feature Extraction](#)
- [NMF for Text Mining](#)
- [Data Preparation for NMF](#)

### About NMF

Non-Negative Matrix Factorization (NMF) is 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.

---

---

**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).

---

---

NMF decomposes a data matrix  $V$  into the product of two lower rank matrices  $W$  and  $H$  so that  $V$  is approximately equal to  $W$  times  $H$ . NMF uses an iterative procedure to modify the initial values of  $W$  and  $H$  so that the product approaches  $V$ . The procedure terminates when the approximation error converges or the specified number of iterations is reached.

Each feature is a linear combination of the original attribute set; the coefficients of these 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.

### NMF for Text Mining

Text mining involves extracting information from unstructured data. Typically, text data is high-dimensional and sparse. Unsupervised algorithms like Principal Components Analysis (PCA), Singular Value Decomposition (SVD), and NMF involve

factoring the document-term matrix based on different constraints. One widely used approach for text mining is latent semantic analysis.

NMF focuses on reducing dimensionality. By comparing the vectors for two adjoining segments of text in a high-dimensional semantic space, NMF provides a characterization of the degree of semantic relatedness between the segments. NMF is less complex than PCA and can be applied to sparse data. NMF-based latent semantic analysis is an attractive alternative to SVD approaches due to the additive non-negative nature of the solution and the reduced computational complexity and resource requirements.

## Data Preparation for NMF

The presence of outliers can significantly impact NMF models. Use a clipping transformation before you bin or normalize the table to avoid the problems caused by outliers for these algorithms.

NMF may benefit from normalization.

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.

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., "O-Cluster: Scalable Clustering of Large High Dimensional Data Sets", Oracle Data Mining Technologies, Copyright © 2002 Oracle Corporation.

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

---

---

This chapter contains the following topics

- [About O-Cluster](#)
- [O-Cluster Scoring](#)
- [Data Preparation for O-Cluster](#)

## About O-Cluster

The O-Cluster algorithm 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. The resulting clusters define dense areas in the attribute space. The clusters are described by intervals along the attribute axes and the corresponding centroids and histograms. A parameter called *sensitivity* defines a baseline density level. Only areas with peak density above this baseline level can be identified as clusters.

The *k*-means algorithm tessellates the space even when natural clusters may not exist. For example, if there is a region of uniform density, *k*-Means tessellates it into *n* clusters (where *n* is specified by the user). O-Cluster separates areas of high density by placing cutting planes through areas of low density. O-Cluster needs multi-modal histograms (peaks and valleys). If an area has projections with uniform or monotonically changing density, O-Cluster does not partition it.

## O-Cluster Scoring

The clusters discovered by O-Cluster are used to generate a Bayesian probability model that is then used during scoring (model apply) for assigning data points to clusters. 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.

## Data Preparation 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.

Because O-Cluster may stop the model build before it reads all of the data, it is highly recommended that the data be randomized.

The use of Oracle Data Mining's equi-width binning transformation with automated estimation of the required number of bins is highly recommended.

Binary attributes should be declared as categorical.

The presence of outliers can significantly impact either type of clustering model. Use a clipping transformation before you bin or normalize the table to avoid the problems caused by outliers.

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.

Categorical data is mapped to numerical values.

---

---

## 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:**

Campos, M.M., Milenova, B.L., Yarmus, J.S., "SVM in Oracle Database 10g: 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 the same functional form as 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 flexibility, scalability, and speed 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 is often 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 towards 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 a linear kernel and a Gaussian (nonlinear) kernel.

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 attributes. 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**

Configure....	Setting Name	Description
Kernel	SVMS_KERNEL_FUNCTION	<p>Linear or Gaussian. The algorithm automatically uses the kernel function that is most appropriate to the data.</p> <p>SVM uses the linear kernel when there are many attributes (more than 1000) in the training data, otherwise it uses the Gaussian kernel. See "<a href="#">Kernel-Based Learning</a>" on page 18-2.</p> <p>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 "<a href="#">Data Preparation for SVM</a>" on page 18-4.</p>
Standard deviation for Gaussian kernel	SVMS_STD_DEV	<p>Controls the spread of the Gaussian kernel function.</p> <p>SVM uses a data-driven approach to find a standard deviation value that is on the same scale as distances between typical cases.</p>
Cache size for Gaussian kernel	SVMS_KERNEL_CACHE_SIZE	Amount of memory allocated to the Gaussian kernel cache that is maintained in memory to improve model build time.
Active learning	SVMS_ACTIVE_LEARNING	<p>Whether or not to use active learning. This setting is especially important for nonlinear (Gaussian) SVM models.</p> <p>By default, active learning is enabled. See "<a href="#">Active Learning</a>" on page 18-3.</p>

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

Configure....	Setting Name	Description
Complexity factor	SVMS_ COMPLEXITY_ FACTOR	Regularization setting that balances the complexity of the model against model robustness to achieve good generalization on new data.
Convergence tolerance	SVMS_ CONVERGENCE_ TOLERANCE	The criteria for completing the model training process.
Epsilon factor for regression	SVMS_EPSILON	Regularization setting for regression, similar to complexity factor. Epsilon specifies the allowable residuals, or noise, in the data.
Outliers for anomaly detection	SVMS_OUTLIER_ RATE	The expected outlier rate in anomaly detection.

**See:** *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 "explodes" categorical data into a set of binary attributes, one per category value.

SVM transforms a binary character column to two numeric attributes, one for each of the binary values. For example, a character column for marital status with values `married` or `single` would be transformed to two numeric attributes. Each of the new attributes would have the value 0 or 1 to indicate either single or married for each case.

## 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 handles 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 Corporation 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.

The Oracle Data Mining APIs use priors to specify class weights for SVM. To use priors in training a model, you create a priors table and specify its name as a build setting for the model.

Priors are associated with probabilistic models to correct for biased sampling procedures. SVM uses priors as a weight vector that biases optimization and favors one class over another.

**See Also:** "Priors" on page 5-4.

## 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.

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 most anomaly detection problem.

## 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:

- [Chapter 19, "Automatic and Embedded Data Preparation"](#)



---

## 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](#)
- [Automatic Data Preparation](#)
- [Embedded Data Preparation](#)
- [Transparency](#)

### Overview

The quality of a model depends to a large extent on the quality of the data used to build (train) it. A large proportion 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** — Transformation instructions specified during the model build are embedded in the model, used during the model build, and used again when the model is applied. If you specify transformation instructions, you only have to specify them once.
- **Automatic Data Preparation (ADP)** — Oracle Data Mining has 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 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. See "[Embedded Data Preparation](#)" on page 19-6.
- **Automatic management of missing values and sparse data** — Oracle Data Mining uses consistent methodology across mining algorithms to handle sparsity and missing values. See *Oracle Data Mining Application Developer's Guide*.

- **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 when a supervised model is scored, thus the results of scoring are in the same units as the units of the original target. See "[Transparency](#)" on page 19-10.

## 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:** *Oracle Data Mining Application Developer's Guide* for further details on nested data and other issues involved in creating the case table.

## 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:** *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 19-5.

## 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.

### Enabling Automatic Data Preparation

ADP is a model setting. You can enable ADP by specifying `PREP_AUTO` in the settings table for the model. By default, ADP is not enabled.

The settings table is a user-created table with two columns: `SETTING_NAME` and `SETTING_VALUE`. To enable ADP, set `PREP_AUTO` to `PREP_AUTO_ON`; to disable ADP, set `PREP_AUTO` to `PREP_AUTO_OFF`.

[Example 19-1](#) enables ADP in a settings table called `SETTINGS_TBL`.

#### **Example 19-1 Turn on the ADP Setting**

```
BEGIN
  INSERT into settings_tbl(
    dbms_data_mining.prep_auto,
    dbms_data_mining.prep_auto_on);
  commit;
END;
/
```

[Example 19-2](#) uses this settings table to enable ADP for a model called `CLAS_MODEL`.

#### **Example 19-2 Enable ADP for a Model**

```
BEGIN
  dbms_data_mining.create_model(
    model_name          => 'clas_model',
    mining_function     => dbms_data_mining.classification,
    data_table          => 'my_data',
    case_id_column_name => 'case_id',
    target_column_name  => 'class',
    settings_table_name => 'settings_tbl');
END;
/
```

---



---

**Note:** By default, ADP is *not* enabled. To use ADP, you must explicitly set `PREP_AUTO` in the settings table for the model.

---



---

### Overview of Algorithm-Specific Transformations

Binning, normalization, and outlier treatment are transformations that are commonly needed by data mining algorithms. These transformation techniques are introduced in this section and described in more detail in "[Embedded Data Preparation](#)" on page 19-6.



## 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).

---

---

See "[Binning](#)" on page 19-9 for information on binning methods supported by DBMS\_DATA\_MINING\_TRANSFORM.

## 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.

---

---

See "[Normalization](#)" on page 19-9 for information on normalization methods supported by DBMS\_DATA\_MINING\_TRANSFORM.

## 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.

See "[Outlier Treatment](#)" on page 19-10 for information on methods for outlier treatment in DBMS\_DATA\_MINING\_TRANSFORM.

## Algorithms and ADP

[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);
```

## Transformation Lists and ADP

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 you disable ADP (accept the default) and you specify a transformation list, your custom transformations are embedded in the model. No automatic transformations are performed.

If you disable ADP (accept 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.

## Creating a Transformation List

A transformation list consists of a set of **attribute transformation expressions**. Each one specifies the transformation for a single attribute.

You can use the `STACK` routines in `DBMS_DATA_MINING_TRANSFORM` to assemble the attribute transformation expressions into a transformation list. A transformation list can specify transformations for any number of attributes.

An attribute transformation expression has the fields described in [Table 19-2](#).

**Table 19-2 Components of an Attribute Transformation Expression**

Field Name	Data Type	Description
<code>attribute_name</code>	<code>VARCHAR2(30)</code>	Name of the column in the build data. If the column is not nested, this is also the complete attribute name. If the column is nested, the full attribute name is: <code>attribute_name.attribute_subname</code>
<code>attribute_subname</code>	<code>VARCHAR2(4000)</code>	Individual attribute within a nested column. If the column is not nested, the attribute subname is null.
<code>expression</code>	<code>EXPRESSION_REC</code>	A SQL expression that specifies how to transform the attribute. This expression is applied to the attribute when it is used internally by the model.
<code>reverse_expression</code>	<code>EXPRESSION_REC</code>	A SQL expression that specifies how to reverse the transformation. This expression is applied to the attribute when it is visible to a user: in the model details and in the target of a supervised model. The reverse expression supports model transparency.
<code>attribute_spec</code>	<code>VARCHAR2(4000)</code>	Either null or "NOPREP". You can set the attribute spec to NOPREP to prevent automatic preparation of this attribute when ADP is on. When ADP is off, NOPREP is ignored. NOPREP cannot be used for an individual subname of a nested attribute. If NOPREP is specified for an individual subname when ADP is on, an error is generated.

The `expression` and `reverse_expression` fields can potentially be very long (over 32K).

[Example 19-3](#) shows a transformation expression for an attribute called `INCOME`. The attribute subname is null, because `INCOME` is not a nested column. Internally, the model uses a log representation of `INCOME`, but the user sees the attribute in its original form in model details and in the results of scoring, if `INCOME` is the target of a supervised model.

**Example 19-3 An Attribute Transformation Expression**

```
( 'INCOME',
  NULL,
  '(log(10, income) - 4.3)/0.7',
  'power(10, 0.7*income + 4.3)', NULL)
```

**Transforming a Nested Attribute**

You can apply the same transformation expression to all the attributes in a nested column, or you can specify separate transformations for individual nested attributes.

If you separately transform some of the nested attributes, you can provide a default transformation expression for all the remaining attributes in the nested column. The default specification has `NULL` in the column field and the name of the nested column in the subattribute field.

For example, the following transformation list specifies the transformation expressions for two nested attributes, `subname1` and `subname2`, in the nested column `nested_coll`.

```
{ nested_coll, subname1, (VALUE - (-1.5))/20, VALUE*20 + (-1.5), NULL }
{ nested_coll, subname2, NULL, NULL, NULL }
{ NULL, nested_coll, VALUE/10, VALUE*10, NULL }
```

The remaining attributes in `nested_coll` are divided by 10 for use within the model, then multiplied by 10 for viewing by a user.

Note that the value of the nested attribute in the transformation and reverse transformation expressions is a constant, `VALUE`.

**See Also:**

- *Oracle Data Mining Application Developer's Guide* for information about attributes.
- *Oracle Database PL/SQL Packages and Types Reference* for details about the stack interface, transformation expressions, and transformation lists.

**Oracle Data Mining Transformation Routines**

Oracle Data Mining provides routines that implement various transformation techniques in the `DBMS_DATA_MINING_TRANSFORM` package. Some of these transformation techniques are summarized in this section.

You can use the routines in `DBMS_DATA_MINING_TRANSFORM`, or can write your own SQL, or use some combination of the two to create your own transformation lists.

**See Also:** *Oracle Database PL/SQL Packages and Types Reference* for information on `DBMS_DATA_MINING_TRANSFORM`.

## Binning

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 19–3 lists the binning techniques provided by Oracle Data Mining.

**Table 19–3 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 "Outlier Treatment" on page 19-10.)
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

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 "Outlier Treatment" on page 19-10.)

**Table 19–4 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 = $\max\{\text{abs}(\text{max}), \text{abs}(\text{min})\}$ .

**Table 19–4 (Cont.) Normalization Methods in DBMS\_DATA\_MINING\_TRANSFORM**

Transformation	Description
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.

### 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 19–5 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.
Winsorizing	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 provides a `GET_MODEL_DETAILS` function for each algorithm. These functions return descriptions of the categorical and numerical attributes used internally by the model.

Model details support **transparency**. Because of transparency, you can obtain meaningful information about a model and gain insight into the way it works.

Transparency ensures that predictions generated by the model are expressed in the original format; any transformations used internally by the algorithm are reversed when the model is applied. For example, if a numerical target is normalized during model build, the predictions in the scoring data are denormalized.

## Model Details and the Build Data

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.

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 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. This is an important aspect of transparency.

The `GET_MODEL_TRANSFORMATIONS` function can be used to obtain the embedded transformations associated with a model.

## Reverse Transformations

In user-specified embedded transformations, the reverse transformation expression should be provided by the user. When ADP is enabled, the reversal is performed automatically. In many cases, this is a straight-forward process.

SVM and NMF are a bit more complicated than the other algorithms in regards to interpretability of results. They both have a set of coefficients, which are used in combination with the transformed attributes. These coefficients are relevant to the data on the transformed scale, not the original data scale.

### Altering the Reverse Transformation Expression

The `ALTER_REVERSE_EXPRESSION` procedure can be used to change the reverse transformation generated by ADP for an attribute. You can use this function to improve readability of model details, specify labels for clusters generated by clustering models, or label the output of one-class SVM.

**See Also:** *Oracle Database PL/SQL Packages and Types Reference* for information about `GET_MODEL_DETAILS`, `GET_MODEL_TRANSFORMATIONS`, and `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:

- [Chapter 20, "Text Mining"](#)



This chapter explains how you can use Oracle Data Mining to mine text.

This chapter includes the following topics:

- [Oracle Data Mining and Oracle Text](#)
- [What is Text Mining?](#)
- [Oracle Data Mining Support for Text Mining](#)
- [Oracle Support for Text Mining](#)

## Oracle Data Mining and Oracle Text

Oracle Data Mining is an option of the Enterprise Edition of Oracle Database. Oracle Text is a separate product that is part of the base functionality offered by Oracle Database. Oracle Text uses internal components of Oracle Data Mining to provide some data mining capabilities.

To use Oracle Text and its data mining capabilities, you do not need to license the Data Mining option. If you wish to use Oracle Data Mining, then a license for the Data Mining option is required.

The support for text data in Oracle Data Mining is different from that provided by Oracle Text. Oracle Text is dedicated to text document processing. Oracle Data Mining allows the combination of text (unstructured) columns and non-text (categorical and numerical) columns of data as input for clustering, classification, and feature extraction.

Oracle Text is described in the *Oracle Text Reference* and the *Oracle Text Application Developer's Guide*.

[Table 20-1](#) summarizes how DBMS\_DATA\_MINING, the Oracle Data Mining Java interface, and Oracle Text support text mining.

*Oracle Data Mining Application Developer's Guide* provides information that helps you develop text mining applications using the PL/SQL and Java interfaces. Oracle Data Mining Administrator's Guide contains descriptions of the sample text mining programs included with Oracle Data Mining.

## What is Text Mining?

Text mining is 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.

Some of the applications for text mining include:

- Create and manage taxonomies
- Classify or categorize documents
- Integrate search capabilities with classification and clustering of documents returned from a search
- Extract topics automatically
- Extract features for subsequent mining

## Document Classification

Document classification, also known as document categorization, is the process of assigning documents to categories (for example, themes or subjects). A particular document may fit into two or more different categories. This type of classification can often be represented as a multi target classification problem where a supervised model is built for each category.

## Combining Text and Structured Data

In some classes of problems, text is combined with structured data. For example, patient records or other clinical records usually contain both structured data (temperature, blood pressure) and unstructured data (physician's notes). In such a case, you can use Oracle Data Mining to perform mining on the structured data, the unstructured data, or both the structured and unstructured data combined.

Oracle Data Mining supports mining one or more columns of text data. A column of text data must have data type CLOB, BLOB, BFILE, LONG, VARCHAR2, XMLType, CHAR, RAW, or LONG RAW. Before text columns can be used in mining, the features of the text columns must be extracted into a nested table. Before you can extract features, you must create a text index for the columns containing text using Oracle Text.

The sample programs distributed with Oracle Data Mining include examples of text mining. For information about the sample programs, see the *Oracle Data Mining Administrator's Guide*.

## Oracle Data Mining Support for Text Mining

Oracle Data Mining provides infrastructure for developing data mining applications suitable for addressing a variety of business problems involving text. Among these, the following specific technologies provide key elements for addressing problems that require text mining:

- Classification
- Clustering
- Feature extraction
- Association
- Regression
- Anomaly Detection

The technologies that are most used in text mining are classification, clustering, and feature extraction.

## Classification and Text Mining

A large number of document classification applications fall into one of the following:

- Assigning multiple labels to a document. Oracle Data Mining does not support this case.
- Assigning a document to one of many labels. For example, automatically assigning a mail message to a folder and spam filtering. This application requires multi-class classification.

The Support Vector Machine (SVM) algorithm provides powerful classifiers that have been used successfully in document classification applications. SVM can deal with thousands of features and is easy to train with small or large amounts of data. SVM is known to work well with text data. For more information about SVM, see [Chapter 5](#).

## Clustering and Text Mining

Clustering is used frequently in text mining; the main applications of clustering in text mining are

- Taxonomy generation
- Topic extraction
- Grouping the hits returned by a search engine

Clustering can also be used to group textual information with other indications from business databases to provide novel insights.

The current release of Oracle Data Mining supports clustering text features using both the PL/SQL and Java interfaces.

The *k*-Means clustering algorithm, described in , supports mining text columns.

## Feature Extraction and Text Mining

There are two kinds of text mining problems for which feature extraction is useful:

- Extract features from actual text. Oracle Text is designed to solve this kind of problem. Oracle Data Mining also supports feature extraction from text. Most text mining is focused on this problem.
- Extract semantic features or higher-level features from the basic features uncovered when features are extracted from actual text. Statistical techniques, such as single value decomposition (SVD) and non-negative matrix factorization (NMF), are important in solving this kind of problem. Higher-order features can greatly improve the quality of information retrieval, classification, and clustering tasks.

NMF has been found to provide superior text retrieval when compared to SVD and other traditional decomposition methods. NMF takes as input a term-document matrix and generates a set of topics that represent weighted sets of co-occurring terms. The discovered topics form a basis that provides an efficient representation of the original documents. For more information about NMF, see [Chapter 9, "Feature Selection and Extraction"](#).

## Association and Text Mining

Association models can be used to uncover the semantic meaning of words. For example, suppose that the word *sheep* co-occurs with words like *sleep*, *fence*, *chew*, *grass*, *meadow*, *farmer*, and *shear*. An association model would include rules connecting

sheep with these concepts. Inspection of the rules would provide context for *sheep* in the document collection. Such associations can improve information retrieval engines. For more information about association models, see [Chapter 8, "Association Rules"](#).

## Regression and Text Mining

Regression is most often used in problems that combine text with other types of data. For more information about regression, see .

## Anomaly Detection and Text Mining

One-Class Support Vector Machine can be used to detect or identify novel or anomalous patterns. For more information, see .

## Oracle Support for Text Mining

[Table 20–1](#) summarizes how Oracle Data Mining (both the Java and PL/SQL interfaces) and Oracle Text support text mining functions.

**Table 20–1 Text Mining Comparison**

Feature	Oracle Data Mining	Oracle Text
Association	Text data only or text and non-text data	No support
Clustering	<i>k</i> -Means supports text only or text and non-text data	<i>k</i> -means algorithm supports text only
Attribute importance	Text data only or text and non-text data	No support
Regression	SVM supports text data only or text and non-text data	No support
Classification	SVM and Naive Bayes support text only or text and non-text data Support for assigning documents to one of many labels	SVM and decision trees support text only Support for assigning documents to one of many labels and also for assigning documents to multiple labels at the same time
Anomaly detection	One-Class SVM supports text only or text and non-text data.	No support
Feature extraction (basic features)	The Java API supports the feature extraction process that transforms a text column to a nested table. The PL/SQL API requires the use of Oracle Text procedures to perform extraction. Oracle Data Mining allows the same degree of control as Oracle Text	Feature extraction is done internally; the results are not exposed
Feature extraction (higher order features)	NMF supports either text or text and non-text data	No support
Record apply	No support for record apply	Supports record apply for classification
Support for text columns	Features extracted from a column of type CLOB, BLOB, BFILE. LONG, VARCHAR2, XMLType, CHAR, RAW, LONG RAW using an appropriate transformation	Supports table columns of type CLOB, BLOB, BFILE. LONG, VARCHAR2, XMLType, CHAR, RAW, LONG RAW

---

---

# 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.

## **apply settings**

In the Java API, an object used to specify the kind of output desired from applying a model to data. This output may include predicted values, associated probabilities, key values, and other data.

## **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  $A \rightarrow B$ , which means that the presence of item set A implies the presence of item set B with certain support and confidence. The support of the rule is the ratio of the number of transactions where the item sets 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 item sets A and B are present to the number of transactions where item set 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*.

### **build settings**

In the Java API, an object that captures the high-level specifications used to build a model. Build settings must specify a mining function; they may specify an algorithm. Oracle Data Mining supports the following mining functions: classification, regression, association, attribute importance, and clustering. [anomaly detection](#) is supported through the classification mining function.

### **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.

### **category**

In the Java interface, corresponds to a distinct value of a categorical attribute. Categories may have character or numeric values.

### **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.

### **connection**

In the Java API, an object used to connect to the data mining server in an Oracle database to perform data mining tasks.

### **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 machines](#).

**data mining**

The process of discovering hidden, previously unknown, and usable information from a large amount of data. This information is represented in a compact form, often referred to as a *model*.

**data mining algorithm**

A specific technique or procedure for producing a data mining model. An algorithm uses a specific model representation and may support one or more [mining functions](#). The algorithms in the Oracle Data Mining programming interfaces are [naive bayes](#), [support vector machine](#), and [decision tree](#) for classification; [support vector machine](#) 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. The data mining server is the Oracle Data Mining implementation of the Data Mining Engine (DME) in the Java API.

**data set**

In general, a collection of data. A data set is a collection of [cases](#).

**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](#).

## **JDM**

See [Java Data Mining](#).

## **Java Data Mining**

A Pure Java API that facilitates development of data mining-enabled applications. Java Data Mining (JDM) supports common data mining operations, as well as the creation, persistence, access, and maintenance of metadata supporting mining activities. JDM is described in the Java Community Process Specification JSR-73. The Java interface to Oracle Data Mining is an extension of JDM.

---

### **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.

### **MDL**

See [minimum description length](#).

### **min-max normalization**

Normalize each attribute using the transformation  $x_{\text{new}} = (x_{\text{old}} - \text{min}) / (\text{max} - \text{min})$ .

### **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.

### **mining task**

See [task](#).

### **missing value**

A data value that is missing because it was not measured (that is, has a null value), not answered, was unknown, or was lost. 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.

### **model**

In the Java API, the object resulting from the application of an algorithm to data, as specified in a [build settings](#) object. See also [mining model](#).

### **multi-record case**

Each case in the data is stored as multiple records in a table with columns `sequenceID`, `attribute_name`, and `value`. Also known as [transactional format](#). 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 table**

An unordered set of data elements, all of the same data type. It has a single column, and the type of that column is a built-in type or an object type. If an object type, the table can also be viewed as a multicolumn table, with a column for each attribute of the object type. Nested tables can contain other nested tables.

### **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  $x_{\text{new}} = (x_{\text{old}} - \text{shift}) / \text{scale}$ . 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.

### **physical data set**

In the Java API, an object that identifies data to be used as input to data mining. The data referenced by a *physical data set* object can be used in model build, model apply (scoring), lift computation, statistical analysis, and similar operations. The physical data specification includes information about the format of the data and the roles that the data columns play. In the Oracle Data Mining Java interface, the physical data set must be a table or view within the database instance specified in the connection object.

### **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 includes transformations to perform common data preparation functions (binning, normalization, and so on)

---

### **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 provides the Support Vector Machine algorithm 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 each attribute using the transformation  $x_{\text{new}} = (x - \min) / \max(\text{abs}(\max), \text{abs}(\min))$ .

---

**schema**

Database schema, that is, a collection of database objects, including logical structures such as tables, views, sequences, stored procedures, synonyms, indexes, clusters, and database links.

**score**

Scoring data means applying a data mining model to data to generate predictions. See [apply](#).

**settings**

See [algorithm settings](#) and [build settings](#).

**single-record case**

Each case in the data is stored as one record (row) in a table. Contrast with [multi-record case](#).

**sparse data**

Data for which only a small fraction of the attributes are nonzero or non-null in any given case. Market basket data and text mining data are often sparse.

**split**

Divide a data set into several disjoint subsets. For example, in a classification problem, a data set is often divided into 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 (ADP) 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](#).

## **test metrics**

In the Java API, an object results from the testing of a supervised model with test data. The test metrics computed depend on the mining function. For classification, the accuracy, confusion-matrix, lift, and receiver-operating characteristics can be computed to access the model. Similarly for regression, R-squared and RMS errors can be computed. Test metrics can also be computed using the PL/SQL interface.

## **task**

In the Java API, an object that represents all the information needed to perform a mining operation, such as build, test, apply, import, and export. A task is executed using the execute method of a connection object.

## **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 format**

Each case in the data is stored as multiple records in a table with columns sequenceID, attribute\_name, and value. Also known as [multi-record case](#). Compare with [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 the mean and standard deviation computed from the data. Normalize each attribute using the transformation  $x_{\text{new}} = (x - \text{mean}) / \text{standard deviation}$ .

---

---

# Index

## A

---

ABN  
    *See* deprecated features, xi  
accuracy, 3-7  
active learning, 18-3  
Adaptive Bayes Network  
    *See* deprecated features  
ADD\_COST\_MATRIX, x  
ADP  
    *See* Automatic Data Preparation  
AI  
    *See* attribute importance, 9-1  
algorithms  
    Apriori, 10-1  
    Decision Tree, 11-1  
    Generalized Linear Models, 12-1  
    *k*-Means, 13-1  
    Minimum Description Length, 14-1  
    Naive Bayes, 15-1  
    Non-Negative Matrix Factorization, 16-1  
    O-Cluster, 17-1  
    supervised, 2-5  
    Support Vector Machines, 18-1  
    unsupervised, 2-5  
ALTER\_REVERSE\_EXPRESSION, 19-11  
anomaly detection, 2-4, 6-1  
Apriori, 2-6  
association models, 8-1  
    algorithm, 10-1  
    confidence, 8-2  
    data preparation, 8-2  
    rare events, 8-2  
    sparse data, 8-2  
    support, 8-2  
    text mining, 20-3  
association rules  
    *See* association models  
attribute importance, 2-3, 2-4, 9-1  
Automatic Data Preparation, ix, 1-6, 2-6, 19-1

## B

---

Bayes' Theorem, 5-2, 15-1  
binary target, 5-2  
binning, ix

## BLAST

*See* deprecated features, xi

## C

---

case table, 1-3, 19-2  
centroid, 7-1  
classification, 2-4, 5-1  
    costs, 5-3  
    one class, 6-2  
    text mining, 20-3  
clustering, 2-4, 7-1, 13-1  
    text mining, 20-3  
confidence, 1-2, 8-2  
confidence bounds, 12-2  
confusion matrix, 1-6, 5-4  
cost/benefit matrix, 5-3, 11-3  
costs, x, 5-3  
counter-examples, 6-1  
CREATE\_MODEL, 2-7  
cube, 1-3

## D

---

data  
    dimensioned, 1-2, 2-6  
    market basket, x, 1-6  
    missing values, x  
    multi-record case, x  
    single record case, 1-3  
    sparse, x, 8-2  
    unstructured, 2-6  
data dictionary views, ix, xi  
data mining process, 1-4  
data preparation, 1-4, 1-5, 2-6, 19-1  
    association models, 8-2  
    clustering, 7-1  
    missing values, 19-1  
data types, 19-2  
data warehouse, 1-3  
date data, 19-2  
DBMS\_DATA\_MINING, 2-7  
DBMS\_DATA\_MINING\_TRANSFORM, 2-7, 2-12  
DBMS\_FREQUENT\_ITEMSET, 2-12  
DBMS\_PREDICTIVE\_ANALYTICS, 2-7, 3-4  
DBMS\_STAT\_FUNCS, 2-12

Decision Tree, x, 2-5, 5-2, 5-4  
demo programs, 2-11  
deployment, 1-6  
deprecated features, xi  
desupported features, xi  
distance-based clustering models, 13-1  
DM\_USER\_MODELS  
    *See* deprecated features  
DMSYS schema  
    *See* deprecated features, xi  
documentation, 2-10

## E

---

embedded data preparation, ix, 2-6, 19-1  
equal-width binning, 8-3  
Excel, 3-2  
EXPLAIN, 2-7, 3-2, 3-5, 3-7

## F

---

feature, 9-1  
feature extraction, 2-4, 9-1  
    Oracle Text, 20-3  
    text, 20-3  
    text mining, 16-1

## G

---

generalization to new data, 18-1  
Generalized Linear Models, x, 2-5, 4-3, 5-2  
GET\_DEFAULT\_SETTINGS  
    *See* deprecated features  
GET\_MODEL\_DETAILS, 19-10  
GET\_MODEL\_SETTINGS  
    *See* deprecated features  
GET\_MODEL\_SIGNATURE  
    *See* deprecated features  
GET\_MODEL\_TRANSFORMATIONS, 19-11  
GLM  
    *See* Generalized Linear Models  
grid-based clustering models, 17-1

## J

---

Java API, xi, 2-8, 3-5

## K

---

KDD, 1-1  
kernel, 2-1  
*k*-Means, 2-6, 7-2, 13-1, 13-2  
Knowledge-Discovery in Databases  
    *See* KDD

## L

---

lift, 1-6, 5-5  
linear regression, x, 4-2, 4-3  
logistic regression, x, 5-2, 5-4

## M

---

market basket analysis, 8-1  
market-basket data, 1-6  
MDL  
    *See* Minimum Description Length  
Mean Absolute Error, 4-4  
Minimum Description Length, 2-5, 14-1  
mining functions, 2-4  
mining model schema objects, ix  
missing value treatment, ix, 19-1  
mixture model, 13-2  
models  
    association, 8-1  
    classification, 5-1  
    clustering, 7-1  
    supervised, 4-1, 5-1  
    unsupervised, 7-1  
multiclass target, 5-2  
multicollinearity, 12-3  
multidimensional data, 1-2, 2-13  
multiple regression, 4-3  
multivariate regression, 4-3

## N

---

Naive Bayes, 2-5, 5-2  
nested data, x, 2-6  
neural networks, 18-1  
NMF  
    *See* Non-Negative Matrix Factorization  
nonlinear regression, 4-2  
Non-Negative Matrix Factorization, 2-6  
    data preparation, 16-2  
    text, 20-3  
    text mining, 16-1  
normalization, ix

## O

---

O-Cluster, x, 2-6, 7-2, 17-2  
OLAP, 1-2  
one-class classification, 6-2  
One-Class SVM, 2-6, 6-2, 18-5  
Oracle Data Mining discussion forum, 2-11  
Oracle Data Mining Scoring Engine  
    *See* deprecated features  
Oracle Database analytics, 2-11  
Oracle Database Configuration Assistant, xi  
Oracle Database statistical functions, 2-12  
Oracle OLAP, 2-13  
Oracle Spatial, 2-13  
Oracle Spreadsheet Add-In for Predictive  
    Analytics, 3-2  
Oracle Text, 2-11, 2-13  
Orthogonal Partitioning Clustering  
    *See* O-Cluster  
outliers, 6-1, 8-3

## P

---

PL/SQL API, 2-7  
PREDICT, 2-7, 3-2, 3-3, 3-5, 3-6, 3-7  
PREDICTION\_BOUNDS, x  
PREDICTION\_PROBABILITY, 2-8  
predictive analytics, 3-1  
    Java API, 3-5  
    PL/SQL API, 3-4  
    *See also* EXPLAIN  
    *See also* PREDICT  
    *See also* PROFILE  
    Spreadsheet Add-In, 3-2  
prior probabilities, 5-4  
PROFILE, 2-7, 3-2, 3-3, 3-5, 3-7

## R

---

radial basis functions, 18-1  
rare events  
    association models, 8-2  
Receiver Operating Characteristic, 5-6  
regression, 2-4, 4-1  
    text mining, 20-4  
regression coefficients, 4-2  
regularization, 18-1  
REMOVE\_COST\_MATRIX, x  
reverse transformations, 19-11  
ridge regression, 12-3  
ROC, 5-6  
Root Mean Squared Error, 4-3  
rules  
    association model, 8-1  
    decision trees, 11-1  
    PROFILE, 3-7

## S

---

scoping of attribute name, x  
scoring, 1-2, 1-6, 13-2  
    O-Cluster, 17-2  
Scoring Engine  
    *See* deprecated features  
single-record case data, 1-3  
singularity, 12-3  
slope, 4-2  
sparse data, 8-2, 19-1  
Spreadsheet Add-In, 3-2  
SQL data mining functions, 2-8  
star schema, 2-6  
statistical functions, 2-12  
statistics, 1-2  
supermodel, ix, 2-6, 7  
support, 1-2, 8-2  
Support Vector Machine, 2-5, 5-2  
    active learning, 18-3  
    classification, 5-4  
        text, 20-3  
    One-Class, 20-4  
    regression, 4-3  
    text, 20-4

text mining, 20-4

## T

---

text features, 20-1  
text mining, 16-1  
    association models, 20-3  
    classification, 20-3  
    clustering, 20-3  
    feature extraction, 16-1, 20-3  
    Non-Negative Matrix Factorization, 16-1  
    Oracle support, 20-4  
    regression, 20-4  
    support (table), 20-4  
    Support Vector Machine, 20-3  
timestamp data, 19-2  
transactional data, 1-6  
transformations, ix, 2-6, 2-7, 19-1  
transparency, 11-1, 12-2, 19-2, 19-7, 19-10

## U

---

unstructured data, 2-6  
unsupervised models, 7-1  
UTL\_NLA, 2-12

## V

---

Vapnik's theory, 4-3

## W

---

white papers, 2-11  
wide data, 4-3

## X

---

XML  
    Decision Tree, 11-4  
    PROFILE, 3-7

## Y

---

y intercept, 4-2

