

MineSet™ Enterprise Edition Reference Guide

Document Number 007-3558-002

CONTRIBUTORS

Written by Helen Vanderberg and Sandra Motroni

Illustrated by Dany Galgiani

Engineering contributions by Barry Becker, Amit Bleiweiss, Jeff Brainerd, Cliff Brunk, Eben Haber, Ara Jerahian, Andy Kar, Ed Karrels, Eser Kandogan, Alex Kozlov, Alan Norton, Peter Rathmann, Mario Schkolnick, Dan Sommerfield, Peter Welch, and Brett Zane-Ulman.

Cover design by Sarah Bolles, Sarah Bolles Design, and Dany Galgani, SGI Technical Publications

COPYRIGHT

© 2000, Silicon Graphics, Inc. All rights reserved; provided portions may be copyright in third parties, as indicated elsewhere herein. No permission is granted to copy, distribute, or create derivative works from the contents of this electronic documentation in any manner, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

LIMITED RIGHTS LEGEND

The electronic (software) version of this document was developed at private expense; if acquired under an agreement with the USA government or any contractor thereto, it is acquired as "commercial computer software" subject to the provisions of its applicable license agreement, as specified in (a) 48 CFR 12.212 of the FAR; or, if acquired for Department of Defense units, (b) 48 CFR 227-7202 of the DoD FAR Supplement; or sections succeeding thereto. Contractor/manufacturer is Silicon Graphics, Inc., 1600 Amphitheatre Pkwy 2E, Mountain View, CA 94043-1351.

TRADEMARKS AND ATTRIBUTIONS

Silicon Graphics, IRIX, and OpenGL are registered trademarks, and SGI, the Silicon Graphics logo, and MineSet are trademarks, of Silicon Graphics, Inc. INFORMIX is a registered trademark of Informix Software, Inc. Microsoft, Windows, Windows NT, Microsoft Developer Studio, jVisual Basic, Visual C++, Visual J++, Visual FoxPro, and Internet Explorer are registered trademarks, and ActiveX is a trademark, of Microsoft Corporation. Netscape and Netscape Navigator are registered trademarks of Netscape Communications Corporation. Oracle is a registered trademark, and Oracle8i, Net8, and SQL*Net are trademarks of Oracle Corporation. Sybase is a registered trademark, and SQL Server is a trademark of Sybase Inc. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd. X Window System is a trademark of the Massachusetts Institute of Technology. Linux is a registered trademark of Linus Torvalds. Java is a registered trademark of Sun Microsystems, Inc. Quattro and Paradox are registered trademarks of Corel Corporation. dBASE is a registered trademark of dBASE Inc. LIMDEP is a registered trademark or trademark of Econometric Software. Lotus 1-2-3 is a trademark of Lotus Development Corporation. S-PLUS is a registered trademark of MathSoft Inc. MATLAB is a registered trademark of The MathWorks, Inc. MINITAB is a registered trademark of Minitab Inc. Osiris is a registered trademark of Osiris Software Inc. SAS and JMP are registered trademarks of SAS Institute Inc. SPSS, SigmaPlot, and SYSTAT are trademarks or registered trademarks of SPSS Science. Stata is a registered trademark of Stata Corporation.

The Tree Visualizer is patented under United States Patents No. 5,528,735; 5,555,354; 5,671,381; and 5,861,885. The Splat Visualizer is patented under United States Patent No. 5,861,891. Patent pending for the 2D slider in the Map Visualizer, Scatter Visualizer and Splat Visualizer. Patent pending for the Evidence Visualizer. Decision Table and Splatviz animation.

Contents

Figures xiii

Tables xv

About This Guide xvii

Related Publications xvii

 Obtaining Publications xvii

Conventions xviii

Reader Comments xviii

1. Concepts and Explanations 1

Adding Columns 1

 Add Column Button 1

Aggregate 3

 Arrays 5

 Distributed Columns 6

Animation 9

Animation Control Panel 9

 Sliders Controlling Independent Dimensions 9

Animation Summary Window 11

Animation Buttons and Sliders 12

 Animation Buttons 12

 Animation Sliders 13

 Data Points and Interpolation 13

 Motion Trails 14

Apply Model 14

 Apply Model Panel 15

 Test Model Panel 16

 Fit Data to Model Panel 17

Applying a Model	17
Association Rules	19
Assoc. Tab	22
File Requirements	22
Visualizing the Association Rules	22
Association Rules Configuration	23
Association Rules Options	23
Association Rules Mappings Button	26
Association Rules Visualization	27
Sample Files for Association Rules	28
Converting Files from .ruleviz to .scatterviz	28
Automatic Binning	31
Backfitting	32
Batch Mode	33
Binning	33
Binning Options	33
Boosting	36
Brushing	37
Changing a Column Type or Name	37
Choice Point	37
Classifier	37
Classifier Name	39
Classify Tab	39
Clustering	39
Clustering by Single k-Means Method	40
Clustering by Iterative k-Means Method	41
Clustering Options	43
Attribute Weights	44
Clustering Options Dialog Box	45
Cluster Visualizer	46
File Requirements	46
Starting the Cluster Visualizer	46

Color Selection	47
Choosing Colors with the Color Chooser (Windows)	47
Choosing Colors with the Color Browser (IRIX)	49
Col. Imp.Tab	52
Column Importance	52
Finding Important Columns	52
Column Importance Differences Among Classifiers	55
Columns	56
Command-Line Operation	57
Configuration Files	57
Confusion Matrix	58
Cost Complexity	59
Cross Validation	60
Data Cleaning	60
Data Destination Pane	61
Data File Tab	61
Data Import	61
Data Transformations Panel	61
Decision Table	63
Inducing the Decision Table	63
Starting the Decision Table Visualizer	64
Discrete Label	64
Exploring Data by Mapping Columns to Axes	65
Interpreting the Decision Table	66
Decision Table Options	67
Pulldown Menus	69
View Menu	69
Nominal Order Menu	70

Decision Tree	70
Creating the Decision Tree	71
Parallelization in IRIX	72
Further Inducer Options	72
Decision Tree Options	72
Search and Filter Panels	75
Discrete Label Menu	77
Drill Through	77
Drilling Down and Drilling Up	79
Error Estimation	79
Evidence Model	83
Evidence Inducer	83
Inducing Evidence Classifiers	84
Starting the Evidence Visualizer	86
Evidence Inducer Options	87
Evidence Visualizer Menus	90
File Menu	91
Windows Systems	91
UNIX Systems	92
File Requirements	93
Filter Button	94
Filter Panel	94
Gain Ratio	95
Help (IRIX)	96
Help (Windows)	96
Histogram Visualizer	97
History	97
Holdout	97
Inducers	98
Inducer Modes in Tool Manager	99
Inducer Error Options	100
Advanced Inducer Options	100
Inducer Status Window	102

Internationalization	104
Setting the Locale on UNIX Systems	104
Extending to Other Languages and Encodings (IRIX Only)	104
Iterative k-Means	107
Laplace Correction	107
Learning Curve	107
Lift Curve	109
Loss Matrix	110
Map Visualizer	114
File Requirements for Map Visualizer	116
Starting the Map Visualizer	117
Configuring the Map Visualizer Using the Tool Manager	118
Generating .gfx and .hierarchy Files	118
Creating Sliders and Animation	119
Map Visualizer Options	119
Map Visualizer File Settings	122
Mining Tools Tab	122
Multiple Selection	123
Mutual Info	123
Naive-Bayes	123
Navigating in the Non-Tree Visualizers With Window Border Tools	124
Navigating in the Tree Visualizers With Window Border Tools	126
Nominal Order Menu	128
Normalized Mutual Info	128
Nulls	128
Option Tree	130
Inducing the Option Tree	130
File Requirements	131
Creating the Option Tree Inducer	131
Parallelization in IRIX	131
Option Tree Options	131
Parallel Computing on IRIX Systems	133
Predictability	134

Prevalence	135
Pruning	135
Random Seed	135
Record Viewer	135
Starting the Record Viewer	136
Renumbering Rows	136
Searching in the Record Viewer	136
Saving Data	137
Record Weighting	137
Regress Tab	138
Regression Trees	138
Inducing the Regression Tree	138
Continuous Label	139
Regression Tree Options	139
Error Estimation in Regressors	142
Regressor Name	142
Remove Columns	142
Return-on-Investment Curve	143
Sample File Directories	144
Saving Files	144
Scatter Visualizer	145
File Requirements	146
Starting the Scatter Visualizer	146
Configuring the Scatter Visualizer	147
Slider Creation for the Scatter Visualizer	148
Scatter Visualizer Tool Options	148
The Animation Control Panel	151
Null Handling in the Scatter Visualizer	152
Sample Configuration and Data Files	152
Selection Menu	153
Slider Creation for Mapviz, Scatterviz, Splatviz	154
Sorting Column Names	155

Splat Visualizer	155
Splat Visualizer Opacity	156
Splat Visualizer File Requirements	159
Starting the Splat Visualizer	159
Splat Visualizer Shape Options	160
Saving the Splat Visualizer Settings	161
Null Handling in the Splat Visualizer	162
Slider Creation for the Splat Visualizer	162
Animation Control Panel	163
Pulldown Menus in Splat Visualizer	165
Shape Menu	165
Sample Configuration and Data Files	166
Split Lower Bound	167
Splitting Criterion	167
Statistics Visualizer	168
How to Read Statistics Visualizer	168
Statistics Visualizer Pulldown Menus	170
Statistics Visualizer's View Menu	170
Table History Buttons	171
"Current view is" Field	171
Prev and Next Buttons	171
Tool Manager	175
Tool Manager Preferences	176
Training Set	176
Tree Visualizer	177
File Requirements	177
Starting the Tree Visualizer	178
Tree Visualizer Options	178
Saving Tree Visualizer Settings	185

Tree Visualizer Pulldown Menus	185
View Menu	185
Tree Visualizer Selection Menu	191
Tree Visualizer Display Menu	192
Tree Visualizer Go Menu	192
Help Menu	193
Null Handling in the Tree Visualizer	193
Tree Visualizer Restrictions	195
Sample Configuration and Data Files	196
Trimming Fraction	196
Uniform Range	196
Uniform Weight	196
View Menu	197
Visualization Tools	197
Warning Options	199
Web Publishing	200
Weighting	200
Year 2000 Compliance	200
A. Sample Configuration and Data Files	201
Association Rules Sample Files	202
Clustering Sample File	202
Column Importance Sample File	203

Decision Tree Sample Files	204
Churn	205
Origin of Cars	205
Predicting Gender	206
Salary Factors	208
Iris Classification	209
Mushroom Classification	210
Party Affiliation	210
Breast Cancer Diagnosis	211
Hypothyroid Diagnosis	211
Pima Diabetes Diagnosis	212
DNA Boundaries	213
Decision Table Sample Files	213
Churn	214
Origin of Cars	216
Predicting Gender	217
Salary Factors	218
Iris Classification	221
Mushroom Classification	222
Party Affiliation	223
Breast Cancer Diagnosis	224
Hypothyroid Diagnosis	225
Pima Diabetes Diagnosis	226
DNA Boundaries	227

Evidence Visualizer Sample Files	227
Churn	228
Origin of Cars	229
Predicting Gender	230
Salary Factors	231
Iris Classification	233
Mushroom Classification	233
Party Affiliation	234
Breast Cancer Diagnosis	235
Hypothyroid Diagnosis	236
Pima Diabetes Diagnosis	236
DNA Boundaries	237
Map Visualizer Sample Files	238
Option Tree Sample Files	240
Churn	241
Origin of Cars	241
Iris Classification	241
Mushroom Classification	242
Party Affiliation	242
Breast Cancer Diagnosis	243
Hypothyroid Diagnosis	243
DNA Boundaries	243
Regression Tree Sample Files	243
Churn	244
Car Mileage	244
Salary Factors	245
Iris	246
Pima	247
Scatter Visualizer Sample Files	247
Splat Visualizer Sample Files	250
Tree Visualizer Sample Files	252
Index	255

Figures

- Figure 1-1** Add Column Dialog Box 1
- Figure 1-2** Aggregate Dialog Box 4
- Figure 1-3** Animation Control Panel With Summary Window and Both Slider Controls 10
- Figure 1-4** Apply Model Dialog Box: Selecting a Classifier 15
- Figure 1-5** The Apply Model Panel 16
- Figure 1-6** Iris Dataset Misclassification, Example 1 18
- Figure 1-7** Iris Dataset Misclassification, Example 2 19
- Figure 1-8** Initial Tool Manager Window Showing Multiway Association Generation 25
- Figure 1-9** Association Rules Mappings Panel 26
- Figure 1-10** Multiple Colors Swatches 47
- Figure 1-11** Color Chooser Dialog Box 48
- Figure 1-12** HSB Pane of the Color Chooser Dialog Box 49
- Figure 1-13** RGB Pane of the Color Chooser Dialog Box 49
- Figure 1-14** Multiple Colors Swatches 50
- Figure 1-15** Color Browser (IRIX) 51
- Figure 1-16** Confusion Matrix for Iris Dataset 58
- Figure 1-17** Estimating the Classifier's Accuracy 81
- Figure 1-18** Classifier Cross-Validation (k=3) 82
- Figure 1-19** Tool Execution Sequence for Models 98
- Figure 1-20** Learning Curve 108
- Figure 1-21** Lift Curve 110
- Figure 1-22** Confusion Matrix for the Mushroom Dataset Using Defaults Settings 111
- Figure 1-23** Confusion Matrix for the Mushroom Dataset With Loss Matrix 112
- Figure 1-24** Confusion Matrix for the Mushroom Dataset With Loss Matrix Allowing Unknown Predictions 113

Figure 1-25	Sample Map Visualizer Screen Showing 1990 U.S. Population	115
Figure 1-26	Representation of a Null Value Mapped to Height (Top Middle Object) and to Color (Bottom Right Object)	130
Figure 1-27	Return on Investment Curve	143
Figure 1-28	Scatter Visualizer Drill Through Dialog	154
Figure 1-29	Shape of Opacity Function For Low and High Values of u	157
Figure 1-30	Image Where $u = 5.3$, and $u = 30$	157
Figure 1-31	Numeric Column Displayed by Statistics Visualizer	169
Figure 1-32	Discrete Column Displayed by Statistics Visualizer	170
Figure 1-33	Table History Buttons “Current view is” Field	171
Figure 1-34	View History Dialog Box (Windows)	173
Figure 1-35	View History Dialog Box (IRIX)	174
Figure 1-36	Sample Records From a Training Set	176
Figure 1-37	Tree Visualizer’s Configuration Options Dialog Box (Windows)	179
Figure 1-38	Tree Visualizer’s Configuration Options Dialog Box (IRIX)	180
Figure 1-39	Tree Visualizer’s Search Dialog Box (Windows)	186
Figure 1-40	Tree Visualizer’s Search Dialog Box (IRIX)	187
Figure 1-41	Sample Results of a Search in the Tree Visualizer	189
Figure 1-42	Representation of a Null Value Mapped to Height, Color, Disk, and Label	195
Figure A-1	Drilling Down on the Churn Dataset	215
Figure A-2	Decision Table Visualizer Using the Adult Dataset	219
Figure A-3	Closer Inspection of the Adult Dataset	220

Tables

Table 1-1	Spending Patterns Aggregated by Age and State	6
Table 1-2	Array Indexed by Age_bin	6
Table 1-3	Columns Distributed by Age_bin	7
Table 1-4	Columns Distributed by Age_bin and Salary_bin	7
Table 1-5	Results When Including Salary_bin in the Total \$ Spent Array	8
Table 1-6	Results of Making an Array by Age_bin and Salary_bin	8
Table 1-7	Results of Distributing <i>Salary_bin</i> and Indexing by <i>Age_bin</i>	8
Table 1-8	Association Rules Mappings	26
Table 1-9	Automatic Binning Options.	31
Table 1-10	Sample Default File Extensions	93
Table 1-11	Korean Font Resources	106
Table 1-12	Navigation Buttons in Non-Tree Visualizers	124
Table 1-13	Adjustment Sliders and Wheels in Non-Tree Visualizers	125
Table 1-14	Manipulating the Non-Tree Visualizer Scene	125
Table 1-15	Navigation Icons in the Tree Visualizers	126
Table 1-16	Adjustment Sliders and Wheels in the Tree Visualizers	127
Table 1-17	system Parameters	133
Table 1-18	Ages 40 to 50	163
Table 1-19	Ages 50 to 60	164
Table 1-20	Interpolation Midway Between Table 1 and Table 2	164

About This Guide

The *MineSet Enterprise Edition Reference Guide* deals with technical features and advanced capabilities of the MineSet suite of data mining and visualization tools. Current information about the MineSet product can also be found on the World Wide Web at <http://www.sgi.com/software/mineset>.

Related Publications

The following documents contain additional information that may be helpful:

- *MineSet Enterprise Edition User's Guide for the Windows NT Client*
- *MineSet Enterprise Edition Tutorial for Windows*
- *MineSet Enterprise Edition Interface Guide*
- *MineSet Enterprise Edition Tutorial for IRIX*
- *MineSet Enterprise Edition Installation Instructions*
- *MineSet Enterprise Edition User's Guide for the IRIX Client*

Obtaining Publications

To obtain SGI documentation, go to the SGI Technical Publications Library at <http://techpubs.sgi.com>.

Conventions

The following conventions are used throughout this document:

Convention	Meaning
Command	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
<i>variable</i>	Italic typeface denotes variable entries and words or concepts being defined.
user input	Fixed-space font denotes literal items that the user enters in interactive sessions. Output is shown in nonbold, fixed-space font.
[]	Brackets enclose optional portions of a command or directive line.
...	Ellipses indicate that a preceding element can be repeated.

Reader Comments

If you have comments about the technical accuracy, content, or organization of this document, please tell us. Be sure to include the title and document number of the manual with your comments. (Online, the document number is located in the front matter of the manual. In printed manuals, the document number can be found on the back cover.)

You can contact us in any of the following ways:

- Send e-mail to the following address:
techpubs@sgi.com
- Use the Feedback option on the Technical Publications Library World Wide Web page:
<http://techpubs.sgi.com>
- Contact your customer service representative and ask that an incident be filed in the SGI incident tracking system.
- Send mail to the following address:
Technical Publications
SGI
1600 Amphitheatre Pkwy., M/S 535
Mountain View, California 94043-1351
- Send a fax to the attention of “Technical Publications” at +1 650 932 0801.

We value your comments and will respond to them promptly.

Concepts and Explanations

Adding Columns

If you want to include, exclude, or sort the order of the columns of data in your dataset before sending it to a classifier or visualizer, you can do this from the Data Transformations panel of Tool Manager. The Add Column Button entry below describes the process of adding a column.

Add Column Button

You can use the *Add Column* button to create a new column whose values are computed based on a mathematical expression. For example, you could add a new column whose values are the ratio of values from two existing columns. Click *Add Column* to get a dialog box that lets you specify the new column name and expression (Figure 1-1).

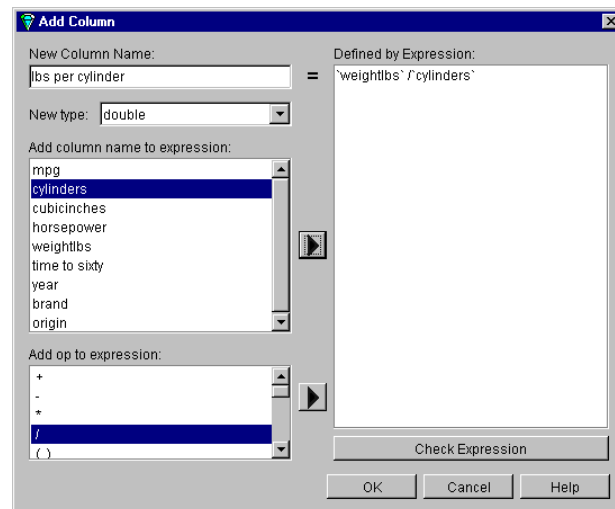


Figure 1-1 Add Column Dialog Box

The left side of this dialog box provides a field for entering the new column's name, and a pulldown menu lets you specify the column type (integer, string, floating point, and so on).

The right-hand side of the dialog contains a large text entry area where you can type in a definition of the expression. As a shortcut to typing column names and operators, scrolled lists in the lower left of the dialog display all columns in the current table and all possible operators. To insert a column name or operator into the expression, either double-click it in its scrolled list, or select it and click the arrow button to the right of the scrolled list.

The expression language used in the Filter and Add Column panels is similar to expressions in C, C++, and Java. The basic operators are the same:

+	addition
-	subtraction
*	multiplication
/	division
()	parentheses for grouping expressions
%	modulo (remainder after division)
!	logical NOT
~	logical NOT
&&	logical AND
	logical OR
^	logical exclusive OR
==	equal to
!=	not equal to
<=	less than or equal to
<	less than
>=	greater than or equal to
>	greater than
&	bitwise AND
	bitwise OR

The expression language also provides the following:

isNull()	Determines if the value in parentheses is null.
if () then () else ()	If the value in the first set of parenthesis is true, then the cell is given the value in the second set of parenthesis. Otherwise, the cell is given the value in the last set of parenthesis.
(x) ? (y) : (z)	C syntax if/then/else
divide(x, y, z)	The cell is given the value of the quotient of x divided by y, unless y is 0, in which case the value is 0.
strlen(x)	The cell is given a value equal to the number of characters in the string.
substring(x, y, z)	The cell is given a substring of string x, that starts at position y, and is of length z. Following the C, C++, and Java convention, the first character in a string has index 0.

To check the expression you have created, click the *Check Expression* button. If there is an error, a dialog box appears, indicating what the error is and where it occurred. When you click *OK*, the expression is automatically checked, and the dialog box is not removed unless the expression is correct.

The Add Column dialog box checks for type compatibility: if you have assigned a numerical expression to a string column (or vice versa), a warning message appears, and the type of the new column is automatically changed to be correct.

You can add your own user-defined functions as well. See the *MineSet Enterprise Edition Interface Guide*.

Aggregate

Using the Tool Manager's *Aggregate* button requires a basic understanding of arrays and distribution. See "Arrays" on page 5 for a basic introduction of concepts used in the aggregation feature.

You can use the Tool Manager's *Aggregate* button to create simple aggregations, make arrays, or distribute columns. Clicking this button causes the Aggregate dialog box to

appear (Figure 1-2). It shows three lists, with the columns in the current table appearing in the middle list.

To aggregate, select the name of the column, and click the left arrow button between the left and center lists. The popup menus below let you specify indexes (if the result is to be an array) and a distribution column (if the result is to be distributed).

Five check boxes at the bottom let you specify how different values are to be combined when aggregated: they can be summed or averaged or the minimum value, maximum value, or count (number of records) can be used. When aggregating number-valued columns, you can choose any combination of these options. For other types, only count is permitted. If you choose more than one option, you get more than one result. For example, selecting average and max gives you one result with average values, and another one with maximum values. The check box “Include nulls in aggregation” allows you to either ignore or include null values in the calculations.

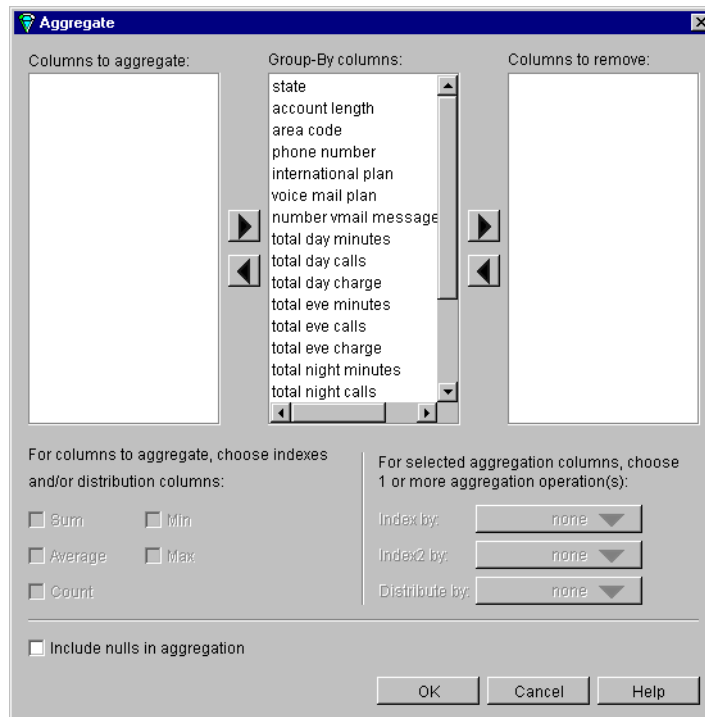


Figure 1-2 Aggregate Dialog Box

The three lists of column names are given below:

- *Columns to aggregate.*
- *Group-By columns* (the default); this keeps the columns unchanged throughout the operation. For each set of records with the same combination of values in the Group-By columns, only one record is output in the resulting table, with values in the aggregated columns summed, averaged, minimized, maximized, or counted (depending on the check boxes at the bottom of the panel).
- *Columns to remove.*

After you have finished with the additional aggregate criteria dialog box, the Current Columns text box in the Data Transformations pane of the Tool Manager window shows the new column names that result from applying these criteria.

The menus in the lower right corner of the dialog box are used to create distributed columns or array columns. These indexing and distribution menus allow you to specify an array index or distributed column. See “Arrays” on page 5 and “Distributed Columns” on page 6.

Arrays

An array is a collection of variables of a certain type, such as floating-point, integer, character string, and so on (see “Changing a Column Type or Name” on page 37 for a list of possible types). Arrays always have an index, which must be a binned column. Arrays can be one-dimensional, two dimensional, three dimensional, and so on. A one-dimensional array can be thought of as a list. A two-dimensional array can be thought of as a table. The more dimensions an array has, the more difficult it is to visualize it.

Arrays are useful for the Tree Visualizer tool; they are necessary if you want to customize sliders used in Scatter Visualizer, Splat Visualizer, and Map Visualizer displays.

For example, suppose your dataset represents dollars spent according to age group and state. To reduce the number of cells, you could bin the ages of participants into three groups: 0-20, 21-40, 41-60. The resulting table is shown in Table 1-1.

Table 1-1 Spending Patterns Aggregated by Age and State

State	Age_bin	Total \$ Spent
CA	0-20	\$50
CA	21-40	\$454
CA	41-60	\$693
NY	0-20	\$35
NY	21-40	\$541
NY	41-60	\$628

Table 1-1 shows six rows, sorted according to state. The same values can also be represented by a one-dimensional array inside a column, as shown in Table 1-2. In this representation, *Total \$ Spent* becomes an array, indexed by the binned column *Age_bin*.

Table 1-2 Array Indexed by Age_bin

State	Total \$ Spent [Age_bin]
CA	[\$50, \$454, \$693]
NY	[\$35, \$541, \$628]

Distributed Columns

Distributing columns is similar to arrays, but different in several important ways. Instead of producing a single new column holding many values, distributing produces one new column for each value of the index. For example, if the data in Table 1-2 was not made into an array, but instead distributed by *Age_bin*, the result would be as shown in Table 1-3.

Table 1-3 Columns Distributed by Age_bin

State	Total \$ Spent 0-20	Total \$ Spent 21-40	Total \$ Spent 41-60
CA	\$50	\$454	\$693
NY	\$35	\$541	\$628

This distributed example expands Table 1-1, increasing the number of columns but decreasing the number of rows.

If you have more than one binned column (for example, *Age_bin* and *Salary_bin*), you can make a two-dimensional array indexed by combinations of *Age_bin* and *Salary_bin*. Table 1-4 refines the previous example by adding the distinguishing characteristic of a *Salary_bin* column (only the rows representing California are shown). This allows you to see where, and by what income bracket the total money is spent. Use this technique to see more refined characteristics in your dataset.

Table 1-4 Columns Distributed by Age_bin and Salary_bin

State	Age_bin	Salary_bin	Total \$ Spent
CA	0-20	\$0-\$25,000	\$30
CA	0-20	\$25,001-\$50,000	\$15
CA	0-20	Over \$50,000	\$5
CA	21-40	\$0-\$25,000	\$120
CA	21-40	\$25,001-\$50,000	\$234
CA	21-40	Over \$50,000	\$100
CA	41-60	\$0-\$25,000	\$101
CA	41-60	\$25,001-\$50,000	\$290
CA	41-60	Over \$50,000	\$302

If you make Total \$ Spent an array, the same data produces the results shown in Table 1-5:

Table 1-5 Results When Including Salary_bin in the Total \$ Spent Array

State	Salary_bin	Total \$ Spent [Age_bin]
CA	\$0-\$25,000	[\$30, \$120, \$101]
CA	\$25,001-\$50,000	[\$15, \$234, \$290]
CA	Over \$50,000	[\$5, \$100, \$302]

If you make an array by both Age_bin and Salary_bin, the results are shown in Table 1-6:

Table 1-6 Results of Making an Array by Age_bin and Salary_bin

State	Total \$ Spent [Age_bin] [Salary_bin]
CA	[\$30, \$120, \$101, \$15, \$234, \$290, \$5, \$100, \$302]

Finally, if you distribute by Salary_bin and index by Age_bin, the results are shown in Table 1-7:

Table 1-7 Results of Distributing Salary_bin and Indexing by Age_bin

State	Total \$ Spent [Age_bin], Salary \$0-25,000	Total \$ Spent [Age_bin], Salary \$25,001-\$50,000	Total \$ Spent [Age_bin], Salary Over \$50,000
CA	[\$30, \$120, \$101]	[\$15, \$234, \$290]	[\$5, \$100, \$302]

The examples above had exactly one relevant value for each array element, and the distribution merely rearranged existing data values. MineSet provides several aggregation options for datasets containing more than one value to be distributed into a given output array element. The most common option is to add the values. This is useful when accumulating expenditures into budgets, for example. You also can take the minimum, maximum, and average of the total number of values, as well as count them.

When distributing values for a given dataset, it is possible that there are no values appropriate for a particular bin. In this case, for *min*, *max*, *avg*, and *sum* aggregations, DataMove fills in a value of null. For *count* aggregations, DataMove fills in a value of 0.

Animation

The Scatter Visualizer, Splat Visualizer and Map Visualizer offer the capability of animation, provided that the dataset being used has at least one slider element mapped to a column. This means you can use animation to display changes in a dataset over a dimension such as time. Typically, independent attributes, such as time or age, make the best slider dimensions, but any binned column can be used.

Animation Control Panel

The animation control panel, which appears to the right of the main visualizer window, consists of a summary window, with up to two adjacent sliders, an information field, animation buttons, the path slider, the speed slider, a data point toggle button, a synchronize sliders button, and for the Scatter Visualizer, a tube trails menu.

Sliders Controlling Independent Dimensions

The number of sliders appearing adjacent to the summary window is dependent on the dataset displayed in the Visualizer's main window. Datasets can have two, one, or no slider dimensions.

Datasets With Two Independent Dimensions

If the dataset has two dimensions of independently varying data (such as the data set in *company.scatterviz*), the controls to the right of the main window become visible (see Figure 1-3).

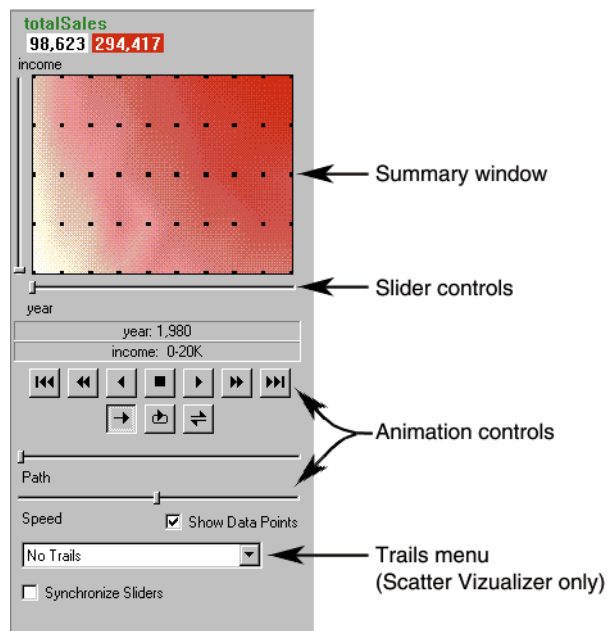


Figure 1-3 Animation Control Panel With Summary Window and Both Slider Controls

To the right of the main window are the summary window and slider controls. The summary window has a horizontal slider below it for selecting data points of the first independent dimension, and a vertical slider to the left for selecting data points of the second independent dimension. The horizontal slider's dimension is identified by a label below it. The vertical slider's dimension is identified by a label above it.

Sliders may be specified in the Tool Manager either by:

- Mapping column names to the Slider 1 and Slider 2 elements in the Data Destination pane for the Map, Scatter, or Splat Visualizers, or
- Creating one or two array columns using the Aggregation panel. These array columns are automatically used for the visualization tool animation sliders.

Animation Summary Window

The summary window shows the total values of the column mapped to the summary visual element for all possible settings of the animation sliders. That is, it shows how the summary attribute varies over the slider dimensions. The whiter the areas of the summary window, the lower the total values represented by the entities in the main window. The denser the color, the higher the values.

The summary window also shows black dots, evenly spaced across the one or two dimensions of data. These dots indicate the precise positions of the discrete datapoints. You can turn off these black dots using the Show Data Points check box.

For example, when the *company.scatterviz* file is first opened, the 2D summary window shows a color range from white (on the left) to red (on the right). White corresponds to a low sales volume; red represents a higher aggregate sales volume. In this example, the greater the density of red, the higher the total sales of life, auto, and home insurance.

An *animation path* is a path between the summary window's black dots over which the data may be animated. There are three ways to create an animation path in the summary window.

Open the file, and create an animation path in any of these ways:

- In the summary window, pick a black dot and define a starting point. Click and hold the left mouse button and drag the cursor over the window. End the path by releasing the left mouse button.
- In the summary window, define a starting point by clicking the left mouse button. Then define an endpoint by moving the cursor to another part of the window and clicking the middle mouse button. A line appears between those two points. To add more line segments, continue with repeated middle mouse clicks. This option is available only if you have a three-button mouse.
- In the summary window, define a starting point by clicking the left mouse button. Then drag one of the independent dimension sliders, drawing a straight line along this dimension. If there are two sliders, using the second slider draws a straight line along the axis controlled by this second slider.

Animation Buttons and Sliders

The VCR-like buttons and sliders (Path and Speed) below the 2D summary window let you control the animation.

Animation Buttons

Once a path is drawn in the summary window (see “Animation Summary Window” on page 11), you can use the VCR-like buttons to control animation along this path. The middle *Stop* button is highlighted in blue, indicating an initial state. Use the adjacent *Play Forward* button (to the right of *Stop*) or *Play Reverse* (to the left) to begin simple movement along the drawn path in a forward or reverse direction. (*Forward* and *Reverse* are defined by the sequence in which the path was drawn, not by the left-to-right or right-to-left movement.)

To stop and restart the animation, click the *Stop* button, then use the *Play Forward* or *Play Reverse* button again. Note that when you stop, the animation continues to the nearest discrete data point.

Adjacent to the *Play* buttons are the *Single-Step Forward* and *Single-Step Back* buttons, as well as *Forward* and *Reverse*. Clicking on one of these buttons changes the current path position to the next discrete data point.

The outside buttons are the *Fast Forward* and *Fast Reverse* buttons. Clicking one of these buttons while in *Stop* state changes the path position to the end (for *Fast Forward*) or to the beginning (for *Fast Reverse*) of the path. Clicking a *Fast* button when in *Play* state increases the animation speed.

Animation Flow

Below the Animation Buttons are the three Animation Flow buttons.

Play-once (default)—the animation moves either forward or in reverse until it reaches the end of the path, then stops.

Loop—when the animation reaches the end of the path, it automatically resets to the beginning and starts over again.

Swing—when the animation reaches the end of the path, it reverses direction and retraces its path to the other end; upon reaching that end, the animation reverses direction again, beginning the cycle again.

Animation Sliders

Sliders can be automatically or manually created through a combination of binning and aggregation. The operations do not show on the current history of Tool Manager, but they do appear in the configuration files for the tool.

Columns mapped to Slider1 and Slider2 eventually form the indices for the columns used in the animation (for example, color or size). These columns must be either numeric (int, float, double) or binned. If a column mapped to a slider is already binned, no automatic binning is needed for this column, and this column is used as an index for a slider. However, if the column is not binned, a binned column is created using the automatic binning options. (See “Binning” on page 33, and the individual tool entries for more information.)

While animation is stopped, you can move the Path slider to reset the position along the path. Note that when you use the Path slider, the cursor in the summary window moves across the drawn path, and the sliders below and to the left of the drawing area move consistently with the cursor position. Then use the *Play Forward* or *Play Reverse* button to restart the animation from the newly specified point. You can drag the Path slider to an arbitrary position between discrete data points; however, when you release the slider, the path position changes to the nearest discrete data point.

Use the Speed slider to adjust the speed of the animation along the path.

Data Points and Interpolation

As animation proceeds, the aggregated variables mapped to size, color, or axes (positions) in the Scatter, Splat, or Map Visualizer change smoothly. However, the information displayed in the Selection message box and the *Pointer is over* field show only the data values of the nearest discrete data position; they do not show interpolated data values.

The animation is produced in the following manner: assume you have data for the years 1991 and 1992, and that these correspond to the size of one entity in the Scatter Visualizer. Assume further that the size for 1991 is 20, and the size for 1992 is 40. As you move the year slider from 1991 to 1992, the size changes by being uniformly interpolated between 20 and 40. For example, midway between 1991 and 1992, the size is 30. As you approach 1992, the size approaches 40. However, you cannot stop an animation between discrete data points, and you cannot drag the Path slider to a stationary position between discrete data points.

The data points in the summary window represent the slider positions corresponding to the actual data from the data file. For example, sizes 20 and 40 are representations of actual data, but size 30 is not. In this example, there would be data points in the summary window at the slider positions corresponding to each year.

Note that not all variables are required to vary with a slider. If there are two sliders, some variables can vary with only one of the sliders, while other variables vary with both.

Motion Trails

In the Scatter Visualizer, the trail menu allows you to display the trajectory of the moving points during the animation. You can choose Line Trails, Fade-Out Trails, Tube Trails, or no trails at all. See the “Creating Animation in the Scatter and Splat Visualizers” section in the *MineSet Enterprise Edition User’s Guide for Windows* for more detail.

Apply Model

The *Apply Model* button in the Data Transformations panel of Tool Manager lets you:

- take a previously created classifier and apply it to new data.
- test a previously created classifier’s performance on the current table.
- fit the current table into a previously created classifier’s structure.

On the upper left of this dialog box (Figure 1-4) is a list of all classifiers currently available on the server. If you select a classifier, the right-hand side lists the column names and types required by that classifier. If these requirements match the current table, a message at the bottom states this. If the current table does not have all the columns required for the selected classifier, the message at the bottom states this, the columns that are missing are selected in the list on the right, and the button on the bottom is deactivated.

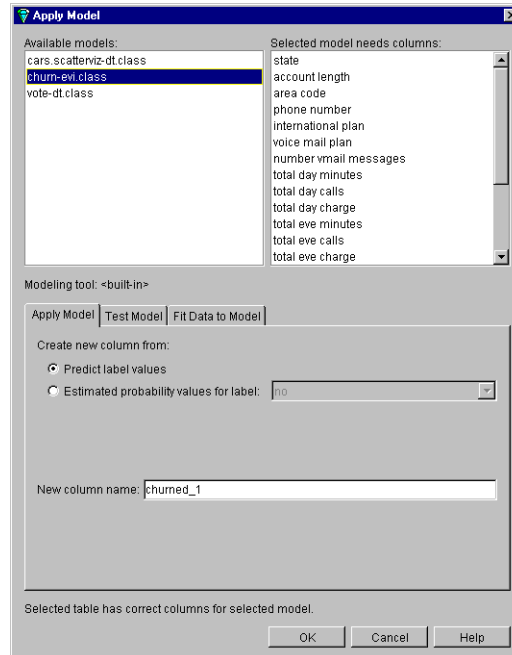


Figure 1-4 Apply Model Dialog Box: Selecting a Classifier

Apply Model Panel

The Apply Model panel is used to apply a previously created classifier to the current table, as shown in Figure 1-5. There are two modes of application for the classifier:

- *Predict label values* for the records in the current table. For example, if you created a classifier to determine churn, you can use this option to add a column that labels each customer as either likely to churn or not likely to churn.
- *Estimated probability values for label value*. Instead of using the classifier to predict the label value of each record, it is used to estimate the probability that each record has a specified label value (for example, churn = yes). Given the classifier created to determine churn, you can use this option to add a column that indicates the probability that each customer is likely to churn.

The *New column name* text field lets you specify the name of the new column.

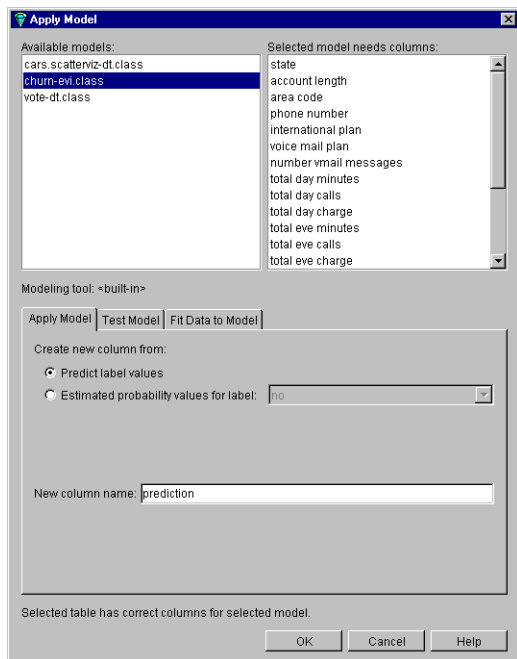


Figure 1-5 The Apply Model Panel

Test Model Panel

The Test Model panel is used to test a previously created classifier on the current table. The table must contain columns with the names and types required by the selected classifier. Unlike Apply Model, Test Model also requires the table to contain a label column with the same name and type as the label column used when building the classifier.

The Test Model panel has options that let you:

- show the confusion matrix of the classifier on the table records.
- show the lift curve of the classifier for a specified label value.
- show the ROI curve of the classifier for a specified label value.

- show a visualization of the classifier with the table used as the test-set (this is only relevant for Decision Tree and Option Tree classifiers).
- select an attribute to use as the record weight.

The text field at the bottom of the Test Model panel shows the results.

Fit Data to Model Panel

The Fit Data to Model panel is used to fit the data in the current table to a previously created classifier. This produces a new classifier with the same structure as the original one; however, the new one uses the data from the table to update the probability estimates (see “Backfitting” on page 32). Because all of the data from the table is being fit into the structure of the classifier, there is no error estimation. Fit Data to Model cannot be used on classifiers that were built using boosting. When you want to evaluate the performance of the new classifier on a separate test set (completely separate from the fit data), use Test Model.

The Fit Data to Model panel has options that let you:

- Show a visualization of the new classifier.
- Specify a name for the new classifier.
- Select an attribute to use as the record weight.

Applying a Model

After building a predictive model, you can apply it to records to predict their label. For example, if you built a classifier (one type of predictive model for discrete labels), for predicting iris type, you can apply the classifier to records containing only the descriptive attributes, and a new column is added with the predicted iris type.

To ensure data quality, after building a classifier you can apply it to the training set in order to identify records that are mislabeled by the classifier. Such records might warrant closer investigation. Perhaps they are “noise,” or they might yield special insights.

For example, suppose a Decision Tree for the iris dataset was induced using the *Classify Only* mode. If you applied the classifier to the dataset, you get a new column (iris type_1) containing the predicted labels. You can then add a column that is defined as type **int** with the expression (iris type \neq iris type_1). The new column has a 1 whenever the classifier misclassifies, and a zero when it correctly classifies. Figure 1-6 shows a Scatter Visualizer plot of the data where the new column is mapped to color with the colors set such that green is 0 (OK) and 1 is red (error). By looking at the plot, it is possible to determine where mistakes are being made.

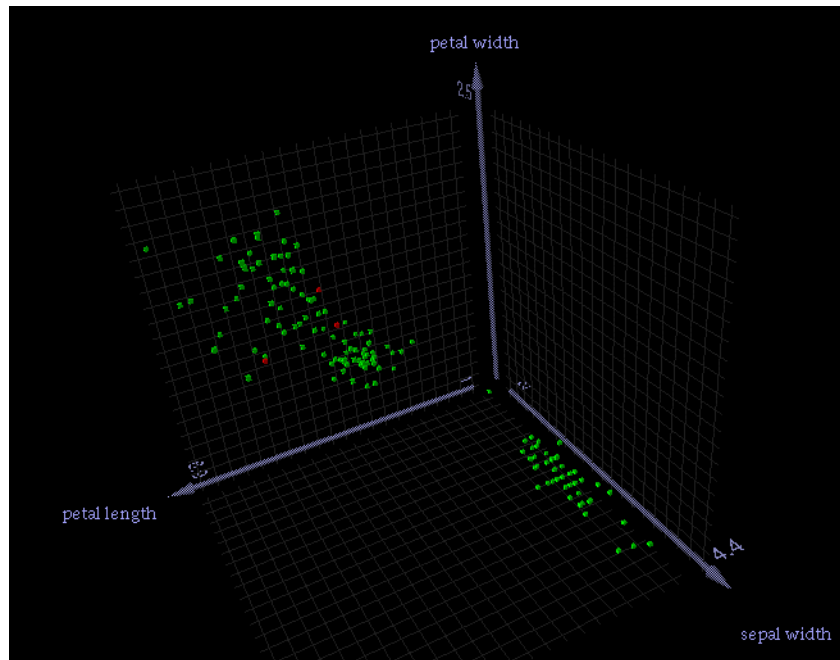


Figure 1-6 Iris Dataset Misclassification, Example 1

Another alternative is to define the new column as a float with the expression (iris type \neq iris type_1) + 0.01. The Scatter Visualizer can then be used with the original label mapped to color, and this new column mapped to size. Incorrect predictions are shown as big cubes; correct predictions are shown as small cubes (see Figure 1-7).

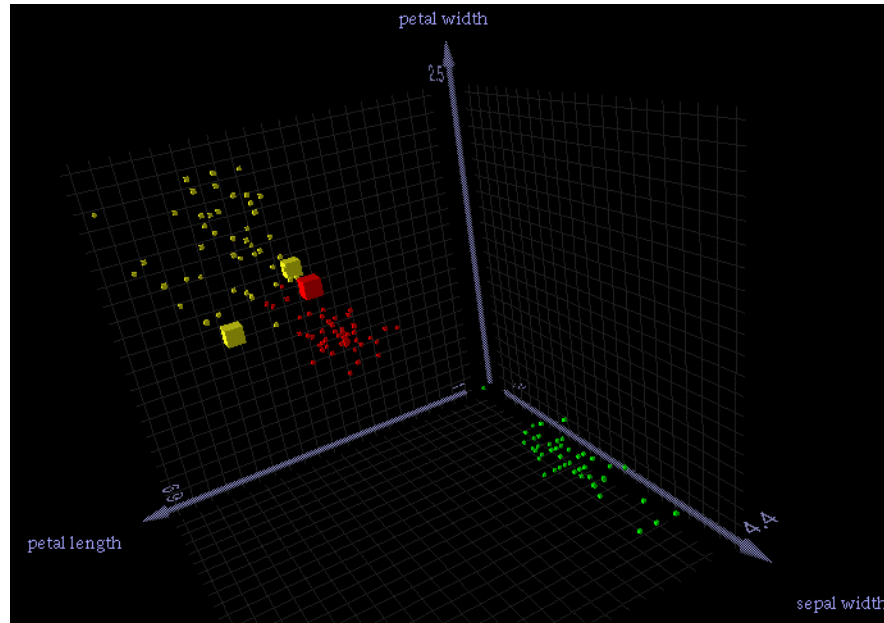


Figure 1-7 Iris Dataset Misclassification, Example 2

For further information on errors and misclassification, see “Error Estimation” on page 79.

Association Rules

Association rules let you mine data by constructing, verifying, and graphically representing models of patterns in large databases. These patterns are expressed by rules of association, which indicate how often column values occur together in a dataset. A typical application of association rules is market basket analysis, which means identifying which items tend to be found together in a shopping market basket.

Discovering and graphically displaying association rules can be relevant to many enterprises. Some examples of where association rules may generate useful associations are supermarket inventory planning, shelf planning, and attached mailing in direct marketing.

There are two steps involved in working with association rules:

1. Rules generation: the data file is processed by the Association Rules Generator, which creates a file usable by the visualizer.
2. Rules visualization: this operation displays the generated association rules.

Association Rules can generate both simple (one-to-one) and multiway rules of association. This section describes simple association rules. For a description of multiway rules, see “Multiway Rules” on page 24.

A simple association rule states that given that X is true, there is a certain probability that Y is also true. MineSet refers to X as the left-hand side (LHS) of the rule and Y as the right-hand side of the rule (RHS).

One example of applying association rules is to obtain market basket data for customer buying patterns. Here, a market basket is the set of items bought by a customer on a single visit to a store. An example rule in this context might be: “80% of the people who buy diapers also buy baby powder.” This percentage is known as the *confidence* of the rule.

In this example, “diapers” is the item on the left-hand side (LHS) of the rule, and “baby powder” is the item on the right-hand side (RHS) of the rule.

Some applications of these rules are:

- If item A appears on the RHS, the LHS can help us determine what the store should do to boost sales of this item.
- If item B appears on the LHS, the RHS can help us determine what products might be affected if the store were to discontinue item B.

The Association Rules Generator processes an input file, then generates an output file consisting of the rules. If X and Y are attributes in a record, then a rule such as:

$X \Rightarrow Y$

indicates that whenever X occurs in a record, expect Y to occur with some frequency.

The strength of the association is quantified by four numbers:

- The *support* of the rule quantifies how prevalent the rule is throughout the dataset, or how often X and Y occur together in the dataset as a fraction of the total number of records. For example, if the support is 1%, X and Y occur together in 1% of the total number of records.

You can specify a minimum support threshold for the generated rules. The default minimum support threshold is 1%. The lower the minimum support, the more rules are generated, and the slower the performance of the tool might be.

Rules that meet a *minimum support threshold* are important for two reasons:

- A rule might have business value only if a significant fraction of records support the rule. For example, if everyone who buys caviar also buys vodka, the rule Caviar \Rightarrow Vodka has 100% confidence. However, if only a handful of people buy caviar (that is, support is very low), the rule might be of limited value to a retailer.
 - A rule might not be statistically significant if a very small number of records support the rule. The rule might be due to chance, and it would not be prudent to make decisions based on such a rule.
- The *confidence* of the rule quantifies the number of records in which both sides of the rule appear, divided by the number of records in which the LHS rule appears. For example, if the confidence is 50%, Y occurs in 50% of the records in which X occurs. Thus, knowing that X occurs in a record, the probability that Y also occurs in that record is 50%. You can also specify a minimum confidence threshold for the generated rules. The minimum confidence threshold default is 50%.
 - The *expected confidence* measures the confidence of a rule as if there were no relationship between the left and right hand sides of the rule. It is computed based on the number of records in which the RHS item appears in the dataset. So the difference between expected confidence and confidence is a measure of the change in predictive power due to the presence of the LHS item.
 - The *lift* is the ratio of confidence to expected confidence. The greater the number, the more unexpected the rule. It tells you how much additional information the LHS of a rule gives when trying to determine whether the RHS is present in any given record.

The Association Rules Generator does not report rules in which the confidence is less than the expected confidence. In other words, a rule such as $A \Rightarrow B$ is not reported if the frequency of A and B occurring together is less than the frequency of B alone.

Note: Given just Y and a rule of the form $X \Rightarrow Y$, nothing is known about X. Rules specify implications only from the LHS to the RHS.

Assoc. Tab

The Assoc. tab is the Mining Tools selection in the Data Destinations pane of Tool Manager from which you generate Association Rules. These rules, often called “market basket analysis,” can help you identify common groupings of occurrences.

File Requirements

Association Rules require the following files which the Tool Manager creates to generate the rules visualization:

- A rules file that results from running the Association Rules Generator, named the *.rules.data* file
- A schema file with the suffix *.rules.schema*, so you can view the rules in Record Viewer
- A *.rules.scatterviz* file

The *.rules* midfix is not required, but will be used whenever these files are generated by the Tool Manager.

Visualizing the Association Rules

There are several ways to start the rules visualizer:

- Use the Tool Manager to configure and start the Association Rules tool (see “Association Rules Configuration” on page 23). Once the rules are generated, the Tool Manager automatically launches the Scatter Visualizer tool.

- If you know which configuration file you want to use, double-click the icon for that file (in your file management window). This starts the Scatter Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.scatterviz* (which is always the case for configuration files created for rules using the Scatter Visualizer using the Tool Manager).
- Enter this command:
 - at the DOS command-line prompt:

```
CD file-directory  
viz [filename.scatterviz]
```
 - the UNIX shell command-line prompt:

```
scatterviz [filename.scatterviz]
```

When starting the Scatter Visualizer, you must specify the configuration file, not the data file.

Association Rules Configuration

You can configure the components of association rules using the Tool Manager which greatly simplifies the task. To configure Association Rules from the Tool Manager Data Destinations pane, select Mining Tools, and click the Assoc. (Associations) tab. Depending on the data source, you may want to bin or remove some columns to simplify your visualization.

Association Rules Options

The Association Rules dialog lets you specify several options.

Confidence

This option lets you quantify the number of records in which both sides of the rule appear, divided by the number of records in which the LHS rule appears. For example, if the confidence is 50%, *Y* occurs in 50% of the records in which *X* occurs. The minimum confidence threshold default is 50%.

Support

This option lets you quantify how prevalent the rule is throughout the dataset, or how often X and Y occur together in the dataset as a fraction of the total number of records. For example, if the support is 1%, X and Y occur together in 1% of the total number of records. You can specify a minimum support threshold for the generated rules. The default minimum support threshold is 1%.

Weight

Association rules allow for record weighting for those cases in which you want to specify that certain records are more important than others or when you want to compensate for uneven sampling. If Use Weight is not checked, then each record has a weight of one. When the box is checked, a menu allows you to choose the column which contains the weight for each record. The *Weight is attribute* box, if checked, includes the weight column in the rules found by the Association Rule Generator. If the box is unchecked, the weight column will be excluded from any rules found by the Generator. See “Record Weighting” on page 137 for a further explanation.

Multiway Rules

If you check Multiway Rules button, the Association Rule Generator generates all rules which satisfy the minimum support and confidence thresholds, including those that have more than one item in the LHS and RHS. An example of such a rule might be “beer and linguini implies potato chips and salsa and wine.”

Multiway association rules generate rules with multiple items on the LHS and/or the RHS. Figure 1-8 illustrates the Tool Manager Association panel configured for multiway rules generation. The version shown is from MineSet for Windows. The IRIX version is similar.

Figure 1-8 Initial Tool Manager Window Showing Multiway Association Generation

Multiway rules are displayed using the Record Viewer rather than the Scatter Visualizer. They are displayed with one rule per row. The first two columns of the table contain the number of items in the LHS and RHS. The next four columns contain the support, confidence, expected confidence, and lift values. The last two columns contain the LHS and RHS items. In the LHS and RHS columns, the items are separated by the word “and.” In the example rule above, the LHS contains two items and is represented as “beer and linguini.” The RHS contains three items and is represented as “potato chips and salsa and wine.”

You can limit the size of the rules generated by entering a number in the “Max total items per rule” field. This number indicates the maximum number of items that are allowed in any rule. The number of items in a rule is the sum of the number of items in the LHS and RHS. The example rule above has five items; simple rules have two items.

Note: Generating multiway rules can take a long time. Watch the status window for an indication of the number of rules generated at each iteration. If too many rules are being generated, cancel the operation and increase the minimum support or confidence thresholds, or decrease the maximum allowable number of items per rule.

Association Rules Mappings Button

Association rules let you map attributes of the rules to visual elements of the display. Specific mapping allows you to test hypotheses for greater understanding of the visualization. From the Data Destinations pane of Tool Manager, clicking on the *RuleViz Mappings* button opens the Association Rules Mappings panel shown in Figure 1-9. Each element shows only the appropriate selections in the popup menu.

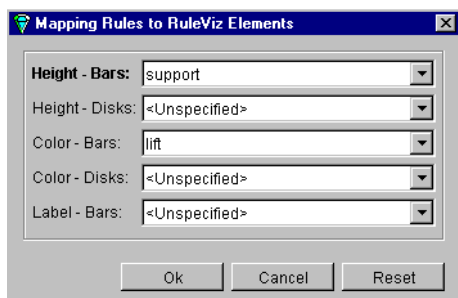


Figure 1-9 Association Rules Mappings Panel

You can map the columns from the automatically generated *.rules.data* file (support, confidence, expected confidence, and lift) to visual elements in this panel; defaults are shown in Table 1-8. Refer to “Association Rules Options” on page 23 for an explanation of the values:

Table 1-8 Association Rules Mappings

Visual Elements	Description
Height - Bars	Lets you specify what bar heights represent (mapped to support by default).
Height - Disks	Lets you specify what the disk heights represent.
Color - Bars	Lets you specify what the bar colors represent (mapped to lift by default).
Color - Disks	Lets you specify what the disk colors represent.
Labels - Bars	Labels - Bars lets you specify what the bar labels represent.

Association Rules Visualization

Association rules are displayed graphically to permit you to explore and compare the generated rules. The rules are presented on a grid landscape using the Scatter Visualizer. The left-hand side (LHS) items are on one axis, and right-hand side (RHS) items are on the other. Attributes or characteristics of a rule are displayed at the junction of its LHS and RHS item. The display can include bars, disks, and labels.

If the displayed view is too small, item labels do not appear on the sides of the axes. You can zoom in on the view using the Dolly wheel until the item labels appear. You can also view the labels for a particular rule by placing the mouse pointer over an individual bar when the mouse is in select mode. All of the details for that particular rule will be displayed in the upper left-hand corner of the view area.

A legend indicating the mapping between displayed attributes (such as bar heights and colors) and the values associated with the underlying rules (such as confidence and support) is displayed at the bottom of the main window.

Bar heights correspond to support and bar colors correspond to lift. When the scene is zoomed in enough, the LHS and RHS axes are labeled with the item names, unless this has been turned off in the configuration file.

You can change the labels as well as what the heights and colors of the bars and disks represent by modifying the configuration file using the Tool Manager (see Table 1-8), or by using an editor to change the configuration file. Color maps are automatically produced when a variable is mapped to disk or bar color. If you wish to change these default color maps, you can edit the configuration file.

If you place the cursor over a bar representing an Association Rules object, then click the left mouse button, the information appears in the Selection Window. Multiple rules may be selected by holding down the Shift key while clicking.

Drilling through is the term used to refer to the act of looking at the underlying data in a visualization. The drill through expression is determined by doing a logical “and” of selected rules. Since the columns in the original table do not match the columns in the *.rules.data* file, the rules Generator produces a special column to help construct the filter expression when a drill through is performed. This means that changing the drill through preferences panel has no effect, because a special string-valued column has already been mapped to drill through in the *.rules.scatterviz* file.

When you drill through on a rule or rules, MineSet shows all the records that satisfy the rule(s). From the visualizer Selection pulldown menu, choose Show Original Data. See “Drill Through” on page 77 entry for more information about drilling through.

Sample Files for Association Rules

The MineSet software includes sample files to demonstrate the features and capabilities of association rules. See Appendix A, “Sample Configuration and Data Files.”

Converting Files from .ruleviz to .scatterviz

Prior to MineSet 2.6, association rules were displayed in their own visualizer, and the configuration files had a slightly different format.

If you have existing *.ruleviz* files you can convert them to *.scatterviz* format by editing the existing *.ruleviz* file and saving it as a *.scatterviz* file. Example 1-1 and Example 1-2 show the differences between the *.ruleviz* and *.scatterviz* formats. Example 1-2 has embedded comments to help you with the changes. Both configuration files use the same data file.

Note: In the old *ruleviz* file format, size was called height, confidence was called predictability, and support was called prevalence.

Example 1-1 group.ruleviz

```
MineSet 2.5
input
{
    file "group.rules";
}

expressions
{
    double `pred/expected` = predictability/expected;
}
view
{
    height predictability;
    height max 10;
    height legend on;

    disk height expected;
```

```

disk height legend label "disk height: expected predictability";

color prevalence;
color colors "white" "purple";
color scale 0 10;
color legend "0%" "10%";

message "%s implies %s\npredictability: %.2f predictability/expected:
        %.2f prevalence: %.2f", LHS, RHS, predictability, `pred/expected`,
        prevalence;

options grid size 3;
options hide disk distance 600;
options hide item distance 600;
}

```

Example 1-2 group.rules.scatterviz

```

MineSet
input
{
# Rename group.rules to group.rules.data:
  file "group.rules.data";

# The schema for the rules.data file is always
# the following. Add these lines:
  int nlhs;
  int nrhs;
  float support;
  float confidence;
  float `expected confidence`;
  string LHS;
  string RHS;
}
expressions {
  float lift = confidence / `expected confidence`;
}

view
{
# This replaces height predictability:
  size confidence, scale 1.;
  size legend label "Bar Height: confidence";
}

```

```
# This replaces disk height expected:
  disk height `expected confidence`, scale 1.;
  disk height legend label "Disk Height: expected confidence";

# This replaces color prevalence:
  color support;
  color colors "white" "purple", legend label "Color: support";
  color scale 0 9;
  color legend "0%" "9%";

# Add these two axis mappings (not present in old file):
  axis RHS, max 100, orderby alpha;
  axis LHS, max 100, orderby alpha;

# Make sure the shape type is bar:
  options entity shape bar;
  options axis label size 20;

  message "%s implies %s\n support=%2.2f%%, confidence=%2.2f%%,
    expected confidence %2.2f%%, lift=%2.2f",LHS, RHS, support, confidence,
    `expected confidence`, lift;

  options grid color "#202020";

  options hide disk distance 600;
  options hide entity label distance 600;
}
```


Automatic Binning

The Tool Manager Bin Columns panel gives you the option of selecting Automatic Binning. If you select the columns you want binned and then select Automatic Binning, on Windows, or “Automatically choose number of bins” from the Automatic Thresholds tab on IRIX, MineSet will do the binning for you. This is useful when you want the program to use machine learning to suggest bins. See “Binning” on page 33 for a full description of binning options. Table 1-9 lists the available automatic binning options:

Table 1-9 Automatic Binning Options.

Option	Windows Location	IRIX Location	Description
Automatic Binning	Basic	N/A	Automatically bins the selected columns.
Auto. Choose # of bins	Advanced	Auto. Thresholds	Tells MineSet to choose the number of bins.
Group into # bins	Advanced	Auto. Thresholds	Lets you choose the number of bins.
Use approach	Advanced	Auto. Thresholds	Lets you choose between automatic, uniform range, and uniform weight.
Discrete label	Advanced	N/A	Lets you select the label for classification. If you choose the automatic approach you must select a label from the menu.
Min. Weight per bin	Advanced	N/A	Lets you enter a minimum weight per bin (or count if weighting is not set) in order to reduce the number of bins.

During automatic binning, thresholds are chosen so that the distributions of labels within different bins are as different as possible. This approach continues to create thresholds that split the range until no additional interval is considered significant. The more distinct values, the more bins are chosen (the relationship is logarithmic).

Clicking the Auto check box tells MineSet to determine the minimum weight automatically based on the total weight of the instances: the more total weight, the higher the minimum weight per bin (the relationship is logarithmic).

Backfitting

Backfitting allows you to build a structure from a training set, then backfit a large dataset to improve the probability estimates. The purpose of backfitting is to make the model's probability estimates consistent with the underlying data. Backfitting is faster than inducing the structure of a large model. When you use holdout error estimation, you leave out a portion of the data for testing. When you backfit all the data through the model structure, the error of the final model is reduced, because counts, weights and probabilities reflect the entire dataset. You can find backfitting in the Advanced Options panel of any inducer.

The model that the inducer builds has two parts:

- Structure — For decision trees and option trees, the structure is the shape of the tree. For evidence, the structure is the number of bins for every attribute, and the thresholds if the attribute is numeric.
- Probability estimates — Each particular part of the structure estimates the probability of each class occurring. These estimates are commonly based on counts of training records at different points in the structure. For decision trees, the probabilities are determined by the weight of records at the leaves. For the Evidence classifier, the probabilities are determined by the conditional probabilities for every attribute value or range. Conditional probability is shown by the rectangular charts in the Evidence Visualizer's left-hand window, which show the relative probability of each attribute value given (conditioned on) each label value.

Backfitting a model with a set of records does not alter the structure of the model, but updates the probability estimates based on the new data. Backfitting is useful for several reasons:

1. A structure can be built from a small training set, then backfitted with a large dataset to improve the probability estimates in the structure. Backfitting is a faster process than inducing the model's structure.
2. When holdout error estimation is used, a portion of the data is left out for testing. Once the model structure is induced and the error estimated, it is possible to backfit all of the data through the structure, which can reduce the error of the final model. When counts, weights, and probabilities are shown in the model's structure, they reflect all the data, not just the training set portion.

When using the drill-through function from the visualizers, you can see data corresponding to the weights shown, which reflect the whole dataset. If backfitting is not used, the weights shown represent only the training set.

You can access “Backfit test set” from the Data Destination panel’s Advanced Options button (Further Options for IRIX) for all inducers. In the Data Destination pane click Mining Tools and select the Classify or Regress tab, and choose Classifier (or Regressor) and Error mode for any inducer. The backfit checkmark is disabled when Boosting is enabled.

Batch Mode

Running MineSet in batch mode allows the Tool Manager to perform the modeling calculations without bringing up any visualizations. Batch mode can be particularly useful in projects requiring lengthy computations that need to be done frequently. For instance, the computations can be run at night so the data will be ready the next morning. For details on using batch mode, see *MineSet Enterprise Edition Interface Guide*.

Binning

Binning lets you sort the information from one or more columns into groups, creating a new column in the process (for example, one with a range of ages, 0-18, 19-25, 26-35, and so on). Binning in Tool Manager saves computation time and prepares a simpler visualization. For details on how to bin columns, see “Changing or Creating New Bins for Columns” in the *MineSet Enterprise Edition User’s Guide for Windows*.

Binning Options

The Windows Binning Options dialog box initially gives a three-way choice for any selected column: Not binned, Automatic binning, and Custom Thresholds. The IRIX version gives two choices, Automatic Thresholds and User Specified Thresholds.

Binning Notation

MineSet uses a modified interval notation for bin names:

(lower-threshold . . . upper-threshold]

The parenthesis, “(,” next to the lower threshold indicates that the threshold value is not included in the range. The square bracket, “],” indicates that the threshold value is included in the range. For example, (10.5 ... 12.6] indicates the range of values over 10.5 up to and including 12.6. If the lower threshold is omitted, the range includes all values less than and including the upper bound. For example, (... 10.5] indicates the range of values less than or equal to 10.5. If the upper threshold is omitted, the range includes all values greater than the lower bound. For example, (12.6 ...) indicates the range of values greater than 12.6.

Binning Approaches

There are three approaches to categorizing data into bins:

- *Entropy*—requires you to also select a discrete label. The thresholds are chosen so that the distribution of labels within different bins is as different as possible. This approach continues to create thresholds that split the range until no additional interval is considered significant.

The “Min weight per bin” text field lets you specify the minimum weight in any bin; this prevents the creation of bins with less weight than the number specified. No interval is split if the two resulting subintervals do not each contain at least the minimum weight you specify. By default, each record has a weight of one. In this situation, specifying the Min weight per bin is the same as specifying the minimum number of instances per bin.

Rather than specifying the minimum weight per bin, it is possible to have the algorithm set the value automatically. The check box labeled *Use Weight* causes the algorithm to calculate a value for the minimum weight per bin based on the total weight of the records: the more total weight, the higher the minimum weight per bin (the relationship is logarithmic).

- *Uniform Range*—the algorithm divides the value range into the specified number of uniformly sized subintervals. The upper and lower bounds for the extreme ranges include any values outside the ranges observed in the data. For example, if the values for an attribute are in the range 3-8, and you specify four bins, the thresholds identified are 4.25, 5.5 and 6.75, corresponding to the ranges:
 - Less than 4.25
 - Over 4.25 up to and including 5.5
 - Over 5.5 up to and including 6.75
 - Over 6.75

The upper and lower bounds for the extreme ranges include any values outside the ranges observed in the data. MineSet's notation for these bin names is:

- (... 4.25]
 - (4.25 ... 5.5]
 - (5.5 ... 6.75],
 - (6.75...]
- *Uniform Weight*—the algorithm divides the value range into the specified number of equal weight bins. Unlike Uniform Range, in which thresholds are identified that separate the value range into intervals of equal size, Uniform Weight identifies thresholds that group the records into subsets of equal weight. By default, each record has a weight of one. In this case, the Uniform Weight approach produces the specified number of bins, each containing an approximately equal number of instances.

Both Uniform Range and Uniform Weight let you specify a trimming fraction, which indicates the fraction of extreme values to be excluded from the value range prior to generating bins. The default trimming fraction is 0.05. This excludes the 5% of the records with the most extreme values (2.5% with the lowest values in the range, and 2.5% with the highest values in the range). Trimming tends to reduce the influence of outliers on the generation of thresholds.

All of the approaches let you decide whether you want to specify the number of bins or let the algorithm select the number automatically. For the Uniform Range and Uniform Weight approaches, the automatic selection of the bins is based on the number of distinct values: the more distinct values, the more bins are chosen (the relationship is logarithmic).

Typically, all of the available instances are used when identifying thresholds. When binned attributes are later used to induce a model, the error estimates tend to be overly optimistic. This is because distributional information from the test set was used to identify thresholds.

Use training set only prevents the binning approaches from looking at the records in the test set when identifying thresholds. This tends to give a more realistic estimate of the classifiers' error rate. *Use training set only* requires the user to specify the same Holdout ratio and Random seed (see "Inducer Error Options" on page 100) that are used to create the holdout set for estimating classifier error.

The Use Weight menu lets you weight the instances by any numeric attribute. Changing record weight affects both Automatic and Uniform Weight, but has no effect on the Uniform Range.

If you click *Apply*, the Tool Manager picks bin thresholds and displays them in the “Thresholds for selected column are” text field. The text field at the bottom of the Bin Columns window shows the progress of the binning algorithm and any errors that occur.

Boosting

Boosting is an algorithm that creates several different models and combines their predictions using a weighted voting scheme. Boosted models often improve accuracy by focusing the induction process on examples in the data which are harder to model than others. Boosting is enabled with the Tool Manager from the Further Inducer Options dialog box in Decision Tree and Evidence inducer modes. Select the Boost (no viz) check box.

In some cases, the most important issue in creating a model is error rate. For example, suppose you have analyzed a dataset for churn prediction to a point that you are satisfied, and are ready to create a model that will predict which of your customers are the most likely to churn. At this point, you are no longer interested in visualizing your model, since you have a reasonably good understanding of the factors that are involved. You also want to achieve the best classification accuracy possible. In this case, you might want to enable boosting.

Boosting will not always increase accuracy, but it often does. Boosted models cannot be visualized, though you can still see confusion matrices, lift curves, learning curves and ROI curves for them. Boosting is a computationally intensive process, often taking 25 times longer to run than the corresponding inducer without boosting. Backfitting does not work with boosted classifiers, because of the special way boosting weights multiple models and the records used to train them.

You can use boosting with labels that have any number of values. Boosting will not always improve the error rate of induced models; this is especially stressed for problems that have more than two label values.

Brushing

Brushing is a feature that allows you to select the same data points in more than one visualization at a time. When you select something in a visualization that supports the sending of brushing events, it will also become selected in any other visualization that is currently open and that supports the receiving of brushing events. Scatter Visualizer and Evidence visualizer support both send and receive brushing events. Tree Visualizer, Map Visualizer and Decision Table Visualizer support only the sending of brushing events.

Changing a Column Type or Name

This Tool Manger option allows you to change the type or name of a column. See the *MineSet Enterprise Edition User's Guide for Windows* for details.

Choice Point

Choice point is an advanced option shown when you click Iterative k-means from the Cluster tab of the Tool Manager Mining Tools. The choice point is a value between zero and one which guides, for instance, the selection of the number of clusters; higher choice points suggest a larger number of clusters while lower choice points suggest smaller. A choice point of 1.0 will always pick the upper boundary. In clustering, if your boundaries are one cluster and five clusters, a choice point of 0.4 might pick 2 clusters while a choice point of 0.8 will pick four. A choice point of 1.0 always picks five.

Classifier

A *classifier* predicts one attribute of a set of data, given several other attributes. A classifier is a type of model. An attribute is an inherent characteristic in the dataset. For example, if you have data on customers of a telecommunications company, a classification model can be generated to predict whether the customer will churn (leave the company) or not, given information such as whether the customer has voice mail, an international plan or not, and how much time they spend on the phone. The attribute being predicted is called the label, and the attributes used for prediction are called the descriptive attributes.

MineSet can build a classifier automatically from a training set. The training set consists of records in the data for which the label has been supplied, (see also “Training Set” on page 176). For example, you supply a database table with one column for each descriptive attribute (such as the presence of a voice mail plan, the average number of calling minutes per day), and one column for the label (churned or not). An algorithm that automatically builds a classifier from a training set is called an inducer.

When a classifier is generated, MineSet also generates a visualization that can help you understand how the classifier operates. This visualization can also provide valuable insight into the data itself. Once a classifier is generated, it can be used to classify records where the label attribute is unknown. This value is predicted by the classifier.

The classifier has two parts: structure and probability estimates

- Structure — For Decision Trees and Option Trees, the structure is the shape of the tree. For Evidence, the structure is the number of bins for every attribute and the thresholds if the attribute is numeric. The Decision Table mathematical structure is the same as the Decision Tree but the display resembles the Evidence visualization.
- Probability estimates — Each part of the structure estimates the probability of each class. These estimates are commonly based on the counts of training records at different points in the structure. For Decision Trees, the probabilities are determined by the weight of records at the leaves. For the Evidence classifier, the probabilities are determined by the conditional probabilities for every attribute value or range. Prior probability means the probability of seeing a particular class label such as *diabetes* in a randomly chosen record in the training dataset, when all other attributes are ignored, in other words, the total number of records with the class label, divided by the total number of records in the dataset. Conditional probability is the probability of a particular record falling into a class such as *age_60+* when you have already selected the label *diabetes* (that is, conditioned on selecting that label).

Note: See Appendix A in the *MineSet Enterprise Edition Interface Guide* for a list of further readings about classifiers as well as acknowledgments for the datasets used in MineSet sample files.

Classifier Name

A generated classifier is named with the prefix of the session filename, as determined in Tool Manager, and the appropriate suffix, for instance, *-dtable.class* for Decision Table classifiers, *-dt.class* for Decision Tree classifiers. By default, all classifiers are stored on the server in the *file_cache* directory, which defaults to *mineset_files*. These classifiers can be used for future classification of unlabeled records; that is, they can be used to predict the labels for unlabeled datasets (see “Apply Model” on page 14 and “Backfitting” on page 32).

Classify Tab

To access the range of MineSet classifiers, from the Data Destination pane of the Tool Manager, click the Mining Tools tab to find the Classify tab. You have a choice of four modes: Classifier and Error, Classifier Only, Estimate Error and Learning Curve. Each of these modes can be combined with any of the various inducers: Decision Tree, Option Tree, Evidence and Decision Table. For details on each inducer see the appropriate entry in this guide. For directions in using these classifiers, see the *MineSet Enterprise Edition User’s Guide for Windows*.

Clustering

The Cluster tab gives you access to MineSet’s Clustering, a useful mining tool for exploring an unfamiliar dataset. Because clustering is a descriptive mining task similar to the discovery of association rules, you do not need to designate a specific column as a label. In addition, the dataset is never split into training and test sets; clustering models are always built from and evaluated on the full dataset.

A clustering model is stored as a set of prototypical records, one per cluster. These represent a weighted average of all data in the cluster and are known as *Cluster centers* or *centroids*. Unlike standard database records, cluster centers maintain a distribution for each column in the form of summary statistics for numerical columns or histograms for categorical columns.

Clustering is run from the Mining Tools tab in the Data Destinations pane of the Tool Manager. The purpose of clustering is to determine what if any characteristics in the dataset are similar. You then look at the resulting clusters and experiment with different parameters.

All clustering in MineSet uses a combinatorial algorithm based on the k-means objective function; the algorithm forms clusters by grouping similar records together, aiming to maximize the overall similarity within each group.

To reach the clustering tool, select the Mining Tools tab in the Data Destination panel of the Tool Manager main window. From the subsequent tabs select Cluster. This will expose the main clustering panel.

You can begin the clustering process by clicking *Go* from the main Cluster panel; there are no required options. By default, you will use the single k-means clustering method to discover three clusters in the data. Once the clustering is complete, you will see an evaluation of the clustering, and then the Cluster Visualizer will appear.

The Cluster panel provides the following options:

- **Method**—Allows you to choose between the single k-means and iterative k-means methods. The default is single k-means. Both methods are described in detail below.
- **Number of clusters**—For the single k-means method only, you must specify the number of clusters to find. The default is 3.
- **Number of clusters range**—For the iterative k-means method only, you must specify lower and upper bounds on the possible number of clusters. The default is 1 ... 10.
- **Choice point**—For the iterative k-means method only, you must specify a *choice point*. This is a value between 0 and 1 which helps choose the final number of clusters. See “Clustering by Iterative k-Means Method” on page 41. The default is 0.5.

Note: Clustering is a computationally intensive operation and will take some time to complete on larger datasets, especially when running iterative k-means mode. It is best to cluster a sample of the data if your dataset has more than 10,000 records.

Clustering by Single k-Means Method

The term *k-means* refers to the objective function determining possible good clustering based on similarity between records. This method is the simplest form of clustering in MineSet. You specify the desired number of clusters, and the algorithm groups the records in the data to minimize the overall dispersion within each cluster. Dispersion refers to the cohesiveness of a cluster; the higher the dispersion, the farther each record falls from the cluster center. Technically, dispersion is measured as the root mean squared distance from each record to the center of the cluster to which it is assigned.

The algorithm itself is iterative and proceeds as follows:

1. You select, for example, five clusters to find.
2. The five cluster centers are initialized to be in random positions in the space of all records. Different choices of the random seed parameter will produce different starting positions.
3. Each record in the data is assigned to the cluster whose center is closest to it. The cluster centers are then recomputed based on the new data in each cluster.
4. If there are any records which are closer to the center of a different cluster than the one they are already in, these records are moved to the closer clusters. Then the cluster centers are recomputed based on the new data in each cluster. This step is run repeatedly until no improvement can be made.

This algorithm is guaranteed to terminate in a finite number of iterations.

Step 4 dominates the running time of clustering. Therefore, the progress window will show one progress bar for each run of step 4, combined with a note on how many iterations have been run so far. You can set a limit on the maximum number of iterations (runs of step 4) allowed before the algorithm stops. The default is 20.

Clusters are assigned sequential numerical names starting from 1. Although cluster names are represented by numbers, there is no ordering to the clusters.

Clustering by Iterative k-Means Method

The iterative k-means clustering method is a more complex extension to the single k-means clustering method, found as a method on the Cluster tab of the Tool Manager. Unlike single k-means, it does not require the specification of an exact number of clusters to create, but instead requires a lower boundary, an upper boundary, and a choice point. The algorithm will pick a number of clusters somewhere between the lower and upper boundary which is appropriate for the dataset. The choice point is a value between zero and one which guides the selection of the number of clusters; higher choice points suggest larger numbers of clusters while lower choice points suggest smaller numbers. A choice point of 1.0 will always pick the upper boundary. For example, if your boundaries are one and five clusters, a choice point of 0.4 might pick two clusters while a choice point of 0.8 will pick four. A choice point of 1.0 will always pick five.

To run iterative k-means, you select three parameters: a lower bound on the number of clusters (default 1), an upper bound on the number of clusters (default 10), and a choice point (default 0.5). Given these parameters, the algorithm proceeds as follows:

1. Use the minimum number of clusters for a run of the single k-means algorithm. This will produce an initial clustering.
2. Find the cluster with the greatest dispersion and split it in half, creating two new clusters. Half of the records in the original cluster will be distributed to one of the new clusters and the other half will be distributed to the other new cluster. Recompute the centers of the new clusters based on the data they contain.
3. If there are any records which are closer to the center of a different cluster than that they are currently in, those records are moved to the closer clusters. The cluster centers are then recomputed based on the new data in each cluster. This is the same as step 4 of the single k-means method. As in the single k-means algorithm, this step is repeated until either no records need to be moved or until it has been run the maximum number of times allowed, as determined by the maximum iterations parameter (default 20).

Steps 2 and 3 are run repeatedly until the maximum number of clusters has been reached:

This process has the effect of producing a range of clusterings between the minimum and maximum numbers of clusters.

The final clustering you are presented with is determined using the choice point parameter (default 0.5) as follows:

Each clustering is evaluated by measuring the average dispersion of each cluster. As the number of clusters increases, the average dispersion always decreases. However, it does not decrease uniformly. The choice point selects the desired degree of dispersion, measured as the proportion from the dispersion of the minimum number of clusters to the dispersion of the maximum number of clusters. The clustering with a measured dispersion closest to this value is chosen as the final clustering. A choice point of 1.0 will always pick the maximum number of clusters, and a choice point of 0.0 will always pick the minimum.

Clusters are named based on their derivation during the splitting process. The initial clustering (based on the minimum number of clusters) is named using sequential numbers, just as in single k-means. Every time a cluster is split, the two new clusters are given the name of the split cluster, but with an “A” or a “B” appended. For example, the cluster named “2-B-A” was derived from cluster 2 in the initial clustering, and was split twice.

When a clustering model is built, you will be presented with a display of statistics in the status window: This example is from the iris dataset:

```
Result of clustering:
-----
Overall root-mean-squared distance from record to centroid: 0.216 +-
0.0928 RMS distance from records to each centroid:
    Cluster 1: 0.2306 +- 0.09484
    Cluster 2: 0.1921 +- 0.09932
    Cluster 3: 0.2247 +- 0.08058
Model saved as iris.cluster
```

The numbers are measures of the dispersion of each cluster, as well as the overall excellence of fit of the clustering. Dispersions from clusterings based on different datasets or different numbers or different numbers of clusters are not immediately comparable.

Clustering Options

The k-means clustering algorithm relies on computing the distance between records and the center of a cluster. Each column in the data is treated as a separate dimension in a multi-dimensional space of all records. By default, each column in the data has an equal influence on the final distance. However, you may change the influence of each column, or attribute, by specifying *Attribute Weights* in the Clustering Advanced Options dialog box.

Attribute Weights

The attribute weight of a column is a value greater than or equal to zero, which determines the column's influence in the distance computations of the clustering algorithm. Setting an attribute's weight to 1 gives it an average influence. Setting it to 2 gives it the influence of two copies of exactly the same column. Setting it to 0 causes the column to have no effect on the clustering (the clustering proceeds as if you had removed the column from the data before running clustering). Attribute weights need not be integral and may be less than one.

Setting attribute weights begins with clicking the *Advanced Options* or *Further options* (IRIX) button in the main clustering panel. The top portion of the Advanced Options dialog box shows the current attribute weights, which all default to 1.

To change one or more weights, select the column(s) whose weights you wish to change (use Shift-click to select or deselect a range, and Control-click to select or deselect more than one value at once). Now type the new weight value into the Selected weights field.

Finally, click the *Set* button and the selected weights will change to the new value. You may select and set all weights at once using the *Select All* button.

You will need to adjust attribute weights to help discover a more understandable clustering. Guidelines for weighting are:

- String or enum (enumerated array) type columns with large numbers of values should generally get low weight (often 0). While such columns will heavily skew the clustering if used by the algorithm, they can help explain the clustering later.
- If you have detected several columns which correlate strongly, adjust the weights so that the TOTAL WEIGHT of this set of columns is 1. Otherwise these columns may excessively drive the clustering.
- Since distances between categorical (string or enum) columns tend to be greater, it is often useful to give them lower weights than real-valued columns.

Clustering Options Dialog Box

You can access the Clustering Options dialog box from the Mining Tools tab of the Tool Manager. Select the Cluster tab, then select the *Further options* button to get the Clustering Options dialog box. The options presented depend on whether you have accessed the single or iterative k-means method.

- *Attribute Weights*

Clustering gives the opportunity to weight the value of each attribute in the dataset differently. See “Weighting” on page 200 for more information about this option.

- *Distance Metric*

This metric determines the way in which distances between records and cluster centers should be measured. The default choice is Euclidean distance, which measures distance along a straight line in multidimensional space, with one dimension per column in the data. The popup menu provides an alternate choice, Manhattan. Manhattan distance is computed by adding the distance along the axis of each dimension. This metric gets its name from the way one traverses street blocks in Manhattan: it is not possible to go straight from point A to point B because travel may only proceed along paths parallel to the axes (streets).

- *Max. # Iterations*

This option sets a limit on the number of passes through the dataset the clustering algorithm may use. More specifically, it limits the number of runs of Step 4 in the single k-means algorithm. See “Clustering by Single k-Means Method” on page 40.

- *Random Seed*

Different random seeds will result in different starting points for the initial cluster centers. See “Clustering by Single k-Means Method” on page 40.

- *Use Weight*

Like most mining tools in MineSet, Clustering supports record weights. This option allows you to specify a column (which must have a numerical value) which will specify the weight of each record in the data.

- *Weight is Attribute*

If this box is selected, the designated weight column will also be used as a normal attribute by the clustering algorithm. If this box is unchecked, the designated weight column will not be used as an attribute; it will also disappear from the attribute weights section (it is given an implicit weight of zero).

Cluster Visualizer

The Cluster Visualizer presents a series of box charts and histograms. Once the clustering operation has been run, you can view the cluster centers directly using the Cluster Visualizer.

Alternatively, you can use the apply model feature (see “Apply Model” on page 14) directly on the training data to assign a cluster to each record. You can then use many of the other tools in MineSet to explore the resulting clusters.

File Requirements

The Cluster Visualizer requires the following files:

- A data file consisting of rows of tab-separated fields. This file is easily created using the Tool Manager. You can generate data files by extracting data from a source (such as a database) and formatting it specifically for use by the Cluster Visualizer. Data files have user-defined extensions (the sample files provided with the Cluster Visualizer have *.clusterviz.data* extension).
- A configuration file, describing the format of the input data and how it is to be displayed. The Tool Manager can create this file, or you can use any editor (such as jot, vi, Emacs, or your favorite text editor) to produce this file yourself.

Configuration files must have a *.clusterviz* extension. When starting the Cluster Visualizer, or when opening a file, you must specify the configuration file, not the data file.

Starting the Cluster Visualizer

There are several ways to start the Cluster Visualizer:

- Use the Tool Manager to configure and start the Cluster Visualizer. (See “Tool Manager” on page 175 for a survey of the Tool Manager’s functionality, which is common to all MineSet tools; see the *MineSet Enterprise Edition User’s Guide for Windows* for examples of using the Tool Manager with the Cluster Visualizer.)
- From the Visual Tools pulldown menu of the Tool Manger, select Cluster Visualizer. Open a configuration file by choosing File > Open.

- If you know which configuration file you want to use, double-click the icon for that configuration file. This starts the Cluster Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.clusterviz* (which is always the case for configuration files created for the Cluster Visualizer using the Tool Manager).
- From a UNIX command-line enter:

```
clusterviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don't specify a configuration file, you must use File > Open to specify one.

Color Selection

Many of the tool option dialogs have options for choosing colors. When using the Tool Manager, click the *Tool options* button (Windows) or *Further options* (IRIX) button to get the tool option dialog box.

Choosing Colors with the Color Chooser (Windows)

To display a Color Chooser, click on a color swatch or on the color list + sign in a Tool Options panel. If a list of color swatches is to be chosen, the list of swatches appears as in Figure 1-10 (these can be empty initially).



Figure 1-10 Multiple Colors Swatches

When you click on a new color swatch you will see that color reflected in the Recent and Preview areas of the Color Chooser (Figure 1-11). If you change your mind, return to the previous color swatch shown in the Recent grid. Once you have selected a color, click *OK*, and that color is added to your color list. You can add several colors without dismissing the Color Chooser.

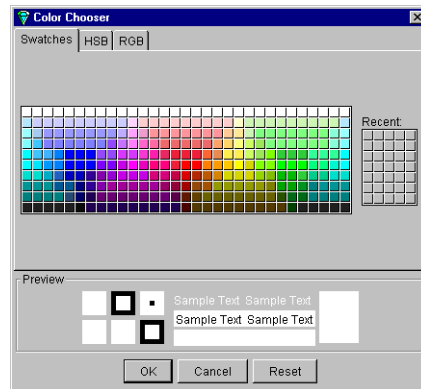


Figure 1-11 Color Chooser Dialog Box

You can also pick a color by using the HSB (hue, saturation, and brightness) or RGB (red, green, and blue) panes of the Color Chooser. In the HSB pane, click in the large colored square. A white circle will appear.

When the H radio button is selected, you can adjust the saturation and brightness by dragging the circle around in the box. To adjust the hue, move the slider next to the rainbow-colored bar.

When the S radio button is selected, you can adjust the hue and brightness by dragging the circle around in the box. To adjust the saturation, move the slider next to the colored bar.

When the B radio button is selected, you can adjust the hue and saturation by dragging the circle around in the box. To adjust the brightness, move the slider next to the colored bar.

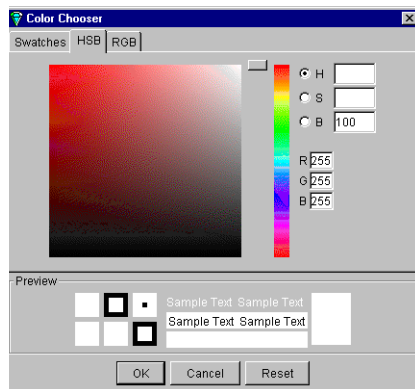


Figure 1-12 HSB Pane of the Color Chooser Dialog Box

In the RGB pane, you can use the sliders to pick specific values for the amount of red, green, and blue in your chosen color.

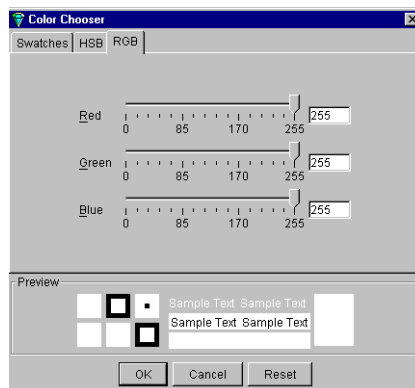


Figure 1-13 RGB Pane of the Color Chooser Dialog Box

Choosing Colors with the Color Browser (IRIX)

If the Color Browser is available, it appears on the tool option panel. MineSet has a color list chooser that uses color swatches. This section describes how to choose, apply, and change color options for the MineSet visualizers.

If only one color is to be chosen (for example a grid color), a single color swatch appears.

Clicking the swatch brings up a Color Browser that lets you change the color of that swatch (Figure 1-15).

If a list of color swatches is to be chosen, a list of swatches appears (these can be empty initially), as shown in Figure 1-14.



Figure 1-14 Multiple Colors Swatches

To edit the color, click a swatch with the left mouse button. This also selects the swatch for making changes to the colors with the buttons. If you click on the swatch with the middle mouse button, the swatch is selected, but the color chooser does not appear.

Next to the list of swatches are four buttons. First is a button labeled with a plus sign (+), which adds a new color at the end of the list. A swatch is added, and the color chooser appears, where you can select the color of that swatch. This Add button is disabled if the maximum number of colors is already in the list.

Next is a button labeled with a minus sign (-). This button deletes the selected color. It is disabled if no swatch is selected, or if the list already has the minimum number of colors.

Next to this Delete button are two buttons to shift the selected color right and left. These buttons are disabled if no swatch is selected, or if the swatch is already at the end of the list.

If there are more colors in the list than room to display them, scroll arrows are added at each end of the list. If the hardware runs out of colors, the color swatches are replaced with text labels showing the color in hexadecimal notation.

The Color Browser (Figure 1-15) appears when you click a color swatch or the add button in the Colors panel of the visualizer's Configuration Options panel.

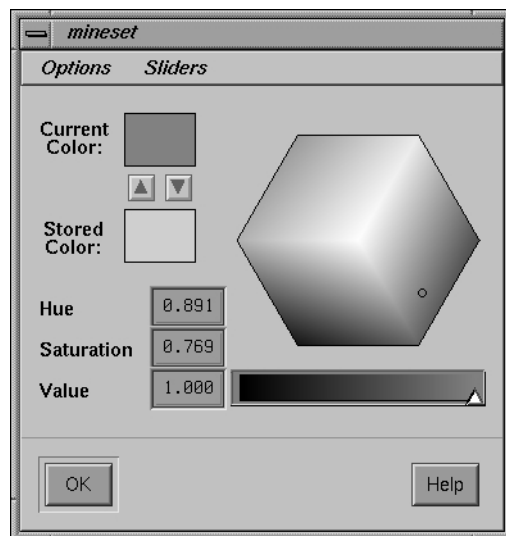


Figure 1-15 Color Browser (IRIX)

To select a color using the Color Browser:

1. Move your mouse cursor on top of the small white circle in the colored hexagon.
2. Press the left mouse button, and move your mouse around the hexagon. The color beneath the small circle appears in the rectangle next to the *Current Color* label. This rectangle acts as your color palette while you choose a color.
3. Release the mouse button when the small circle is on top of a color you want. The selected swatch immediately takes on the chosen color.

You can edit several colors without dismissing the Color Browser; clicking any color swatch in the options panel lets you edit that color in the already posted Color Browser.

Click the *OK* button when you decide on a color. The Color Browser window closes.

Col. Imp.Tab

The Col.Imp. tab of the Tool Manager gives access to the Column Importance tool. See “Column Importance” below.

Column Importance

Column Importance is run from the tab labeled Col. Imp. on the Mining Tools panel of Tool Manager. It helps you discover which are the most important columns in a dataset in discriminating different values for a label column you choose.

The difference between column importance and other data mining tools such as clustering (discussed in “Clustering” on page 39) is that with column importance, you decide which label you will use to determine the importance of columns. In clustering, the data itself shows you which are the discriminating factors. Because of the differences in representation for classification models, different attributes may be judged more important for different models. A sample file provided with MineSet, showing a case in which Column Importance might be useful is discussed in Appendix A, “Sample Configuration and Data Files.”

Finding Important Columns

With *Column Importance* you might, for example, want to find the best three columns for detecting the label *good credit risk* so you can choose them for mapping to axes the Scatter Visualizer. When you select the label and click *Go*, a popup window appears with the three columns that are the best three discriminators. A measure called “purity” (a number from 0 to 100) informs you how well the columns differentiate the different labels. Adding more columns can only increase the purity.

Purity is a measure of the skewness of the label value distribution. The cumulative purity measure is a measure of the purity of partitioning the data. The data is partitioned using columns found as important in the same way data is partitioned in a Decision Tree. Each set in the partition has its own purity measure, and the purity measure within the partition is a combination of these individual measures. For a given set in the partition, the purity is 0 if each class has equal representation, and 100 if every record is of the same class. Similarly, the cumulative purity will be 0 if each set in the partition has an equal representation of classes, and 100 if each set in the partition contains record that all have the same class.

There are two modes of Column Importance:

- Simple Mode

To invoke the Simple mode, choose a discrete label from the popup menu, and specify the number of columns you want to see, then click *Go*.

- Advanced Mode

Advanced mode lets you control the choice of columns. To enter Advanced mode, click *Advanced Mode* in the Column Importance panel. A dialog box appears that allows you to select a weight attribute and decide whether it behaves as a regular attribute for determining importance. The dialog box contains two lists of column names: The left list contains the available attributes and the right list contains attributes chosen as important (by either you or the Column Importance algorithm).

Advanced mode can work two different ways: finding several new important attributes or ranking available attributes.

- Finding Several Important Attributes

To enter this submode, click the first of the two radio buttons in the middle of the dialog (*...find [number] additional important attributes*). If you click *Go* with no further changes, the effect is the same as if you were in Simple mode, finding the specified number of important columns and automatically moving them to the right column. Near each column, the cumulative purity is given (that is, the purity of all the columns up to and including the one on the line).

Alternatively, by moving column names from the left list to the right list, you can prespecify columns that you want included and let the system add more. For example, to select the *cylinders* column and let the system find three more columns, click the *cylinders* column name, then click the right arrow between the lists.

Clicking *Go* lets you see the cumulative purity of each column, together with the previous ones in the list. A purity of 100 means that using the given columns, you can perfectly discriminate the different label values in the dataset.

– Ranking Available Attributes

Advanced mode also lets you compute the change in purity that each column would add to all those that were already marked important, that is, they are in the list on the right. For example, you might move *cylinders* to the list on the right, and then ask the system to compute the incremental improvement in purity that each column remaining in the left column would yield. The cumulative purity is computed for columns on the right (already marked important).

To enter this submode, click the second of the two radio buttons at the bottom of the dialog (*...compute improved purity for left columns, cumulative purity for right columns*). This submode permits fine control over the process. If two columns are ranked very closely, you might prefer one over the other (for example, because it is cheaper to gather, more reliable, or easier to understand).

The importance of a column depends on the columns previously marked as important. For example, while *net-income* might be a good column individually, it might not be as important together with *salary* because they are likely to be highly correlated. The best set of three columns is not necessarily composed of the columns that rank highest individually. If two columns give the income in dollars and in another currency, they are ranked equally alone; however, once one of them is chosen, the other adds no discriminatory power to the set of best features.

Column selection is useful for finding the best three axes for Scatter Visualizer and Splat Visualizer. It is also useful for finding a good discriminatory hierarchy (hierarchy that separates different label values) for the Tree Visualizer when you select the label to be the key used in the Tree Visualizer.

All floating point values (**doubles** or **floats**) are prediscretized using automatic discretization. If a column has no value given to it in the left list, the algorithm did not consider it; this is because it either had a single value (for example, when it is discretized into one interval), or the number of records that it would separate is not statistically significant.

Column Importance Differences Among Classifiers

This section describes the differences among Column Importance, the importance ranking chosen by the Evidence and Decision Table Inducers, and the splits chosen by the Decision Tree Inducer. As Column Importance uses all of the data, these descriptions assume that you are running the inducers in “Classifier Only” mode, so that the inducers are using all of the data as well.

Discretization Process

The Column Importance algorithm and the Evidence Inducer render all continuous attributes (columns) discrete using the automatic discretization algorithm (the same algorithm that is applied in automatic binning in the Tool Manager). The Decision Tree algorithm does not render attributes discrete ahead of time and finds thresholds as the tree is built.

The main advantage of doing the discretization automatically is that it discretizes the continuous range into several intervals at once, while the Decision Tree makes only binary splits.

The main advantage of the Decision Tree algorithm is that it renders subsets into discrete chunks of data (those that reach a specific node where a test is done). Thus the discretization is “local” to those records as opposed to a “global” discretization.

Importance Function

The Evidence Inducer and the Column Importance algorithm rank attributes based on “mutual information” as the purity measure, in contrast to the Decision Tree, Option Tree, and Decision Table Inducers which penalize multi-way splits. Thus if you allow Decision Tree to suggest important columns the result will prefer attributes with fewer values over attributes with many values. The default for Decision Trees can be changed to “mutual information.”

Dependence on Other Attributes

The Evidence Inducer ranks each attribute independently. If several attributes are highly correlated, they have similar ranking. If you use the Advanced mode from Column Importance, and the “...compute improved purity” option without any attributes chosen as important (that is, moved to the list on the right), the attribute ranking shown matches the sort order chosen by the Evidence Inducer.

The Column Importance algorithm, the Decision Tree Inducer, and the Decision Table Inducer all provide more powerful importance capabilities than the Evidence Inducer. All choose an importance ranking with respect to other columns.

In Column Importance, columns are judged as important relative to the set of columns in the list on the right. If two columns are highly correlated and one is chosen, the other will probably never be chosen; each column is chosen for its ability to provide more information about the label than is already present.

In the Decision Table Inducer, the Suggest button provides an importance facility similar to the Column Importance algorithm. Columns are judged as important relative to the set of columns already present in the mapping box. There are three major differences, however, between the Decision Table Inducer's suggest mode and the Column Importance algorithm. First, the Decision Table Inducer penalizes multi-way splits. Second, the Decision Table Inducer is capable of performing a more exhaustive search to find a better column ordering. Finally, the Decision Table Inducer does not report any purity scores—it merely ranks the columns.

The Decision Tree Inducer provides a more flexible importance ranking because different columns can be selected at different subtrees. For example, one column can be chosen for the left child of the root and another for the right child of the root. While this is appropriate for a Decision Tree, it is inappropriate for choosing a small set of columns to show in the Scatter Visualizer or Splat Visualizer. For these cases, the Column Importance algorithm is superior because it builds an “oblivious” Decision Tree in which every level of the tree tests the same column across the nodes. With Column Importance, a single column must be chosen for all combinations of the previously chosen columns.

Columns

See “Adding Columns” on page 1, “Remove Columns” on page 142, and “Sorting Column Names” on page 155.

Command-Line Operation

Each visualizer, as well as MineSet itself, can be started from the command line, by typing the tool name at the prompt, for example:

On Windows systems type:

```
Viz [configFile]
```

For UNIX systems type (for example):

```
scatterviz [configFile]
```

Identifying the configuration file (*configFile*) is optional, but if none is specified, the tool starts with only the File and Help menus operational. You must then specify a configuration file using File > Open.

On UNIX, the various tools use the commands: *clusterviz*, *eviviz*, *scatterviz*, *splatviz*, *statviz*, *treeviz*, and *mapviz*. Association rules uses a two-part command covered under “Association Rules” on page 19.

Configuration Files

MineSet requires two files be present for each tool:

- The data file consisting of rows of tab-separated fields, which has a *.data* extension.
- The configuration file which describes the format of the input data and how it is to be displayed. This file carries an abbreviation of the tool name as an extension, for instance *eviviz*, *scatterviz*.

You create this configuration file automatically when you use the Tool Manager to specify the options and parameters of the classifier or visualizer you plan to use. Alternatively you can create configuration files for each tool using an ascii text editor. For example, open Word Pad or similar text editor and select “Files of Type: All Documents (*.*)” You can open any *.schema* and *.data* file this way. Details and examples can be found in the *MineSet Enterprise Edition Interface Guide*.

Confusion Matrix

Confusion matrices give a more detailed picture of the errors made by a classifier. Instead of simply analyzing the number of correct and incorrect predictions, the confusion matrix shows the *type* of errors being made. Displaying a Confusion Matrix is an Advanced option for all classifiers (choose the Classify tab) under the Mining Tools tab of the Tool Manager. Figure 1-16 shows a confusion matrix for a Decision Tree that was induced on the iris dataset.

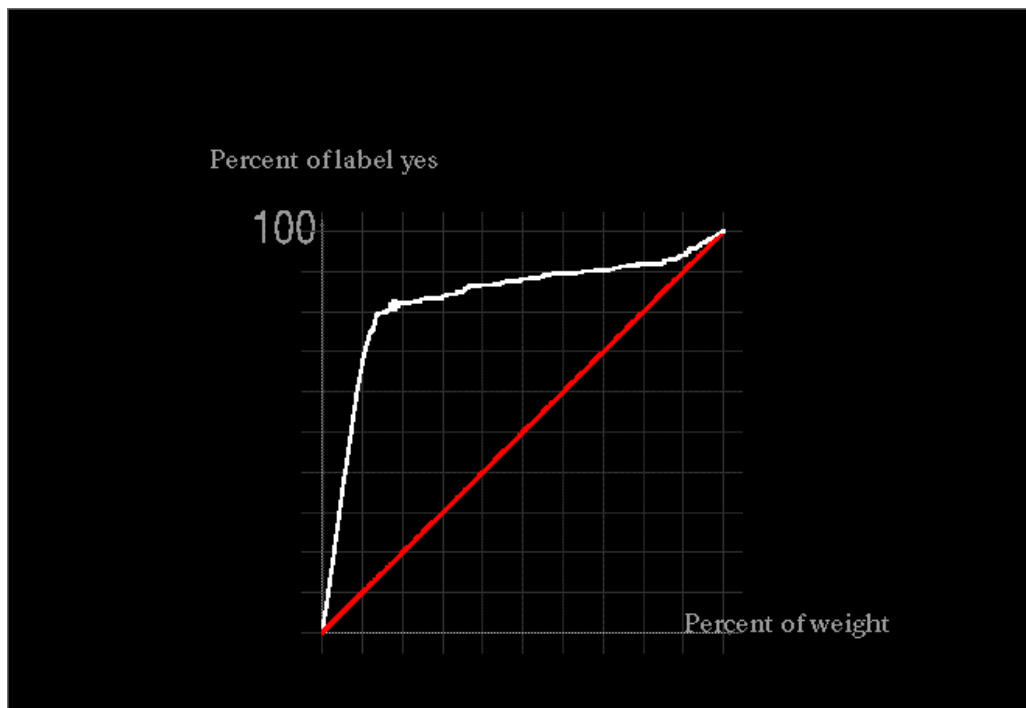


Figure 1-16 Confusion Matrix for Iris Dataset

The two axes represent:

- the class values predicted by the classifier, and
- the actual class values given in the test set (holdout set).

Entries on the diagonal are correct predictions. Entries off the diagonal indicate incorrect predictions. This representation shows that *iris-versicolor* and *iris-virginica* are frequently confused, but *iris-setosa* is always predicted correctly.

When the cost of making different types of mistakes is uneven, it is frequently useful to influence the type of errors that are being made by using a Loss Matrix (see “Loss Matrix” on page 110).

Note: The confusion matrix shows the errors made on the test set; thus, it represents the expected true distribution of errors in an actual situation if the underlying distribution of the data does not change significantly. The confusion matrix in MineSet is computed prior to backfitting and is the same whether or not backfitting is applied. (See also “Backfitting” on page 32.)

Cost Complexity

Cost complexity is an advanced pruning method option developed in CART (Classification And Regression Trees). To find this pruning method, from the Tool Manager select the Classify Mining Tools tab and choose Decision Tree. Click on the *Advanced Options* button to get the dialog box showing Pruning Options. (You can also select the Regress Mining Tools tab and follow a similar path.)

Cost complexity pruning attempts to generate optimally sized trees by trading off the error rate of the tree (its cost) and the number of leaves in the tree (its complexity). During cost complexity, pruning the training set is partitioned into a learning set and a pruning set. The learning set is used to grow a pruning tree. This tree is pruned to generate a sequence of trees with decreasing complexity. The pruning set is then used to identify the minimum cost tree in this sequence. The size of the minimum cost tree is noted. The learning and pruning sets are recombined and used to grow a tree. This tree is then pruned to the size of the minimum cost tree.

The cost complexity pruning parameter allows you to select trees smaller than the minimum cost tree. The parameter indicates the number of standard errors more costly than the minimum cost tree that you are willing to accept. Setting the parameter to zero selects the minimum cost tree; setting the parameter to 0.5 selects the minimum size tree that had an error rate no more than 0.5 standard errors worse than the minimum cost tree. Higher numbers indicate more pruning. If your data might contain noise (errors and anomalies), increase the number to create smaller trees. If the tree is pruned back to a

single node, decrease the number to decrease the amount of pruning and show more of the tree's structure.

Pruning is slower than limiting the tree height or increasing the split lower bound because a full tree is built and then pruned. Pruning, however, is done selectively, resulting in lower error rates.

Cross Validation

Cross validation is a method of estimating classifier error that splits the dataset into a certain number (k) of folds (commonly 10), and builds that same number of classifiers. The process can be repeated multiple times to increase the reliability of the estimate. The inducer is trained and tested k times; each time it is trained on all the data minus a different fold, then tested on the holdout fold.

Cross Validation is an Estimate Error Option available from any inducer. The dialog box lets you set the number of folds in cross validation and the number of times to repeat the process. You can also set the random seed to a different number or maintain the same number to be sure you always cut the data from the same point.

Data Cleaning

Data may be in a difficult location, in an obscure form, or incompatible with other existing data. Fields may be missing or no longer valid. The process of bringing order from such complexity is called data cleaning, and many times is the precursor of data mining operations. Third party extraction tools are available to assist in this process.

To convert data from other formats to the MineSet format, and to do some rudimentary data cleaning operations, select File > Import Data from the Tool Manager menu bar. For more information, see the *MineSet Enterprise Edition Interface Guide*.

Data Destination Pane

The Data Destination pane in the Tool Manager gives you these major choices: Viz Tools, Mining Tools, and Data File, all of which are located in the top level of tabs. Viz Tools provide a direct visualization of existing data to show correspondences that may provide insight. Mining Tools create models of data (and the accompanying visualization) that can then be used to predict future data. Data File is a storage option for saving manipulated data as a file.

Data File Tab

Using the Data File tab in the Data Destination pane of Tool Manager lets you save manipulated data for future use in a data file on the client or server. If you click the Data File tab, a panel appears with two toggle buttons to let you specify whether the file is to be saved on the server or your client machine. The selected name for the client file appears next to the Client check box. If you select Client, the *Choose new client file* button brings up a dialog for you to choose the name for the client file. When you select Server, you can type the server filename directly into the adjacent text field.

Note: Pathnames are not permitted for server files; all server files are stored in the DataMove cache directory.

Data Import

MineSet provides a data import facility, available from the Tool Manager File menu (Import Data). The Import Data panel allows you to choose the file you wish to import, along with a number of import options. See the *MineSet Enterprise Edition Interface Guide* for a description of these options, as well as a list of supported data formats.

Data Transformations Panel

The Data Transformations panel on the left of the main Tool Manager window lets you manipulate the tables of data in the dataset from which you will work. After you have selected a table using the File menu, the table's column headings appear in the *Current Columns* window of the Data Transformations panel. The functions of the displayed options are:

- *Remove Column*—lets you delete one or more columns that are not relevant to the current visualization or mining.
- *Bin Columns*—lets you assign each record to a group that falls within a certain range of column values. For example, an age column may be binned into the range: (0...18], (19...25], (26...35], and so on. The "(" indicates that the lower bound is not included in the range. The "]" indicates that the upper bound is included in the range (see also "Binning").
- *Aggregate*—creates new columns representing the sum, average, count, maximum, or minimum values of the selected column. Aggregate can also makes arrays from columns indexed by other binned columns. The table which results after an aggregation operation is applied usually has many fewer rows than the original table, because each row in the new table is an aggregate of several rows in the original (see "Aggregate" on page 3).
- *Filter*—lets you select a subset of the data based on an expression involving column values, for example, leave only those records in which the age is less than 20 (see "Filtering").
- *Change Types*—lets you change a column's name as well as its type (see "Change Types").
- *Add Column*—lets you add a new column based on a mathematical expression (see "Add Column"). For example, add a column "minor" based on the column "age," using the expression: "if age is less than or equal to 18 then minor becomes 1; else minor becomes 0."
- *Apply Model*—lets you use a previously created classifier to label new records, to estimate probabilities for label values, to test the classifier on new data, or to backfit data to an existing classifier (see "Apply Model").
- *Sample*—lets you select a random subset of the data. This is useful for very large data sets (see "Sampling").
- *Sort Columns*—lets you sort the columns alphabetically. This function appears as a check box on Windows systems.
- *Plugin Ops*—appears only if Plug-in APIs are available. This function gives users the ability to access plug-in operations as if they were native MineSet operations. See the MineSet home page (<http://mineset.sgi.com>) and the *MineSet Enterprise Edition Interface Guide* for more details.

Decision Table

A decision table is a model that is hierarchical in structure like a decision tree, but breaks the data down at each level using a pair of attributes rather than a single attribute. The Decision Table inducer identifies the most important attributes (columns) for classifying the data, and then the visualizer displays them graphically as a nested table of cake charts. Each cake chart in the visualization can in turn be divided into smaller cakes representing the next most important attributes. Each visualization can contain several layers representing decreasingly important attributes. See the *MineSet Enterprise Edition User's Guide for Windows* for information on using the Decision Table Visualizer.

The method of classification used by a decision table is similar to that of a decision tree. Classification is done by picking the majority class of the region in which the example is found. If a record to be classified falls in a region where no training data occurred, classification is done by picking the predominant class one level up in the table hierarchy.

Inducing the Decision Table

A Decision Table classifier can be induced, or generated automatically, from data. The data is made up of records, and a label associated with each record. (See "Inducers" on page 98.)

The automatic induction of Decision Table classifiers is a process in which record counts (or more generally record weights) are used to calculate the probabilities at every node in the table hierarchy. In the visualization, the distribution of records at each of these nodes is shown using a cake chart.

All continuous attributes are separated into discrete bins, so that class distributions in these ranges are as different as possible. The number of ranges is determined automatically. You can override automatic binning for any attribute by explicitly binning using Tool Manager.

The number of cakes in any row along one of the axes (attribute) pairs shows the number of discrete ranges produced by the inducer. If there is just one range, it means that this attribute by itself was not useful in predicting the label. Initially, the prior probabilities of the labels are displayed in the Label Probability Pane. The prior probabilities are the proportions of each class in the training set.

There are three ways to assign attributes to X and Y axes at every level in the Decision Table hierarchy: manually, automatically, and automatically with feature search.

Starting the Decision Table Visualizer

There are several ways to start the Decision Table Visualizer:

1. Run the Decision Table Inducer from the Tool Manager under the Classify tab. After the inducer builds the classifier, it automatically invokes the Decision Table Visualizer.
2. Use the Tool Manager to start the 3D Visualizer from the Visual Tools menu, then open a *.dtableviz* file (see “Tool Manager” on page 175 for details of the Tool Manager’s functions, which are common to all MineSet tools).
3. If you know what configuration file you want to use, double-click the icon for that configuration file. This starts the 3D Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.dtableviz*. (Configuration files created for the Decision Table Visualizer with Tool Manager all carry the *.dtableviz* suffix.)
4. Start the Decision Table Visualizer from the UNIX command line by entering the following command at the prompt:

```
dtableviz [filename.dtableviz]
```

Here, the filename is optional. If you don’t specify a configuration file, then you must use File > Open to select one.

Discrete Label

The Discrete Label menu provides a list of columns containing discrete values in the dataset (click on the arrow next to *Discrete Label:*). Discrete attributes (binned values, character string values, or integers) have a limited number of values. You should select a label attribute with few values—two or three values is best (see “Training Set”). If there are no discrete attributes, the menu shows No Discrete Label, and the *Go* button is disabled. You then must create a discrete attribute by binning or adding a new column, using the Tool Manager’s Data Transformations panel.

Exploring Data by Mapping Columns to Axes

You can explore the relationships between attributes in data by mapping columns to X and Y axes. Select a column name in the Current Columns panel of the Data Transformations pane of Tool Manager window, then:

- On Windows systems click on a cell in the X-list or Y-list and the pulldown menu gives you a list of appropriate choices.
- On UNIX systems, select an item in the X-axis or Y-axis windows. The first two columns mapped are shown at the top level, and subsequent mappings descend through the levels.

If you map an odd number of attributes, the last level of detail shows a single column of cakes corresponding to the values of that odd attribute. If an interaction between two attributes is suspected, map one attribute to the X axis and the other to the Y axis.

The visualization may exchange the X and Y mapping with a level in order to maintain a better aspect ratio for the overall matrix of cake charts, but it will not move attributes to different levels than the ones to which they were mapped.

If you have no idea which attribute to map to which axis, simply click the *Suggest* and *Go* button.

There are two different ways to Suggest: with or without feature subset search. In the Further inducer options panel:

- If the “Suggest using feature search” check box is not checked, MineSet runs Column Importance to find the features. Column Importance is described further in “Column Importance” on page 52.
- If the “Suggest using feature search” check box is checked, MineSet begins with Column Importance, then runs through all possible orders, beginning with the order recommended by Column Importance. An error estimate is run at every stage, and each option is explored. Predictably, this can be a tedious process, and if you become impatient you can click *Stop* at any time. The longer the feature search continues, the more accurate the resulting classifier.

Interpreting the Decision Table

The *prior probability* for each class label is depicted in the pie chart in the Label Probability Pane, on the right of the screen. The prior probability for a class label is the probability of seeing this label in the data for a randomly chosen record, ignoring all attribute values. Mathematically, this is the number of records with the class label divided by the total number of records.

The *probability distribution* for each rectangular block or cake in the Main Window on the left, shows the proportion of records in each class considering only records having that particular combination of values. These probabilities are precise for the data given.

By default, values of nominal attributes along an axis are sorted by how well they predict one of the classes. This helps identify important values. If the label is a binned attribute, the class that is the highest bin is used. If the label is nominal, then the class with the largest slice in the prior probability pie is used to determine the order. When you select a particular class, and request a sort by label probability (by choosing Nominal Order > by Label Probability), that class determines the order of the nominal values. Alternatively, the values of the nominal attributes can be sorted alphabetically or by weight. Values of binned attributes are always shown in their natural order.

When you drill down into a rectangular cake (using the right mouse button) the data represented by that cake is redisplayed in a new matrix of cakes where the attributes assigned to that level of detail are used as the axes. The cake charts sit on top of a gray base. If the base is completely covered with cakes, then every combination of values at this level is represented by the data. Often much of the base is uncovered. This shows regions where no data exists. In a very sparse dataset there are large regions that are empty. Picking or selecting a gray base has the same effect as selecting the cake that was one level up in level of detail. Drilling down on a base causes every cake sitting on top of it to expand to the next level of detail.

If an attribute has unknown (NULL) values, then the unknown values are denoted by a question mark (?). The NULL always appears as the first value if present and does not get sorted with the rest. It is possible to toggle the display of NULL values using View > Show Null Values.

Navigation varies according to whether you are running Windows or IRIX, and whether you are using a two or three-button mouse. Here a Windows two-button style is described. You can change button modes from the Preference panel on your desktop.

If a cake is selected (left mouse click) in the left pane, then the probability pie in the right pane will show a posterior probability distribution which exactly matches that shown by the cake. When more than one cake chart on the left is selected (Ctrl-left mouse click), then the pie on the right will show the probability distribution (with respect to the label) for the set of records defined by the selected cakes. It is also possible to select a whole group of cakes by selecting the gray base below them. This has the same effect as selecting the cake one level up in the hierarchy.

Classes are listed under the pie chart on the right, and are in order of slice size. The class with the largest probability is at the top. As values on the left are selected, this order changes to reflect the changing probability pie. The class that would be predicted given current selections is shown at the top. If the label is a binned attribute the order of the classes is not changed on slice size. In addition, if the label is a binned attribute, colors are assigned according to a continuous spectrum: the highest bin is red; otherwise, random colors are used.

To see more detail, move the Detail Slider beneath the Label Probability pane to the right.

Decision Table Options

Selecting Advanced Options (Windows) or Further Inducer Options (IRIX) causes the Advanced Inducer Options dialog box to appear. This dialog box consists of four panels:

- The top panel indicates the choices you made in the Tool Manager's Data Destination panel.
- The second pane from the top lets you set the loss matrix and the weight attribute. See "Loss Matrix" on page 110 and "Weighting" on page 200.
- The bottom-left panel lets you specify further Inducer Options, described below.
- The bottom-right panel lets you specify the Error Estimation Options (unless the mode you chose in the Data Destination panel was Classifier Only, in which case this area is empty). The options shown in this panel depend on the type of Error Estimation you chose (see "Applying a Model" on page 17 and "Error Estimation" on page 79).

To fine-tune the Decision Table induction algorithm, you can change the following Decision Table inducer options.

- **Maximum size**

This is particularly useful when you have a large dataset. By default, there is a limit of 10,000 nodes (these nodes correspond exactly to the cakes in the visualization). Limit the size by clicking the check box and typing a number for the limit. Limiting the number speeds up induction and limits the storage required. Although restricting the size decreases the run time, it might also increase the error rate. If the maximum size is smaller than needed to show all combinations of values, the visualization may show fewer attributes than are mapped.
- **Maximum Attributes**

This option determines how many different columns you allow to participate in the Decision Table when searching. Limiting the attributes simplifies the result and speeds its arrival. This limit affects only columns added by the *Suggest* mode. Columns added manually are not subject to the limit.
- **Minimum Weight per Bin**

The Decision Table inducer discretizes all continuous attributes. This option lets you define the minimum number of instances per bin. The automatic setting calculates this number based on the dataset size: the larger the dataset, the larger the bin size. If your dataset is very large, you might obtain more discrete ranges than you want. To reduce the number of bins, raise this value.

Drilling Down and Drilling Up

In the visualizer, you can drill down or up on the Decision Table cakes to see more or less detail (see “Drilling Down and Drilling Up” on page 79). Drill down by clicking the right mouse button on one of the cakes. This replaces the one cake with smaller cakes which show the data broken down by the pair of attributes for the next level down. If the cursor is positioned on a gray base, clicking the right mouse button drills down on all cakes on that base. If the cursor is positioned on the background, clicking drills down or up globally on all cakes. To drill up, hold the Ctrl key and click the right mouse button on the cake, base, or background (or use the middle mouse button).

Pulldown Menus

Five pulldown menus let you access additional Decision Table Visualizer functions: File, View, Nominal Order, Selection, and Help. If you start the Decision Table Visualizer without specifying a configuration file, only the File and the Help menus are available. See the File, Help and Selection entries for menu-specific details.

View Menu

The View menu lets you control certain aspects of what is shown in the Main Window and contains these options:

- Filter Panel opens the panel that allows you to filter your data according to criteria you select (see “Filter Panel” on page 94).
- Set Background Color opens the Color Chooser and lets you select a new background color (see “Color Selection” on page 47).
- Window Decoration lets you hide or show the external controls around the main window.
- Null Positions toggles the display of null values. Null values, if present, are shown as the first value, offset slightly from other non-Null values.
- Use Landscape Viewer (or Use Examiner Viewer) switches to an alternative mode of 3D navigation. Since the middle mouse button is used for navigation in the Landscape viewer, to drill up you must click the middle mouse button while holding down the Control key. This option is not supported on Windows.
- Show as Evidence (Evidence Mode on IRIX) shows conditional probabilities of each cake, rather than distributions based on the record weights, which are shown initially. This is useful if one or more of the classes are small.
- Toolbar allows you to show or hide the Tool and Status bars. This option is not available on IRIX.

Nominal Order Menu

The Nominal Order menu lets you control how values for nominal attributes are ordered and offers these choices:

- Alphabetical causes attributes with nominal values to be sorted from left to right (or bottom to top) alphabetically.
- Weight causes the values to be sorted from left to right, with those having the largest weight of records appearing toward the left.
- Label Probability (the default) causes the values of nominal attributes to be sorted by the size of the slices corresponding to one of the classes. If the label is a binned attribute, the highest bin is used by default. If the label is nominal, then whatever class has the largest slice in the prior probability pie is used by default. If a particular class is selected, and then sort by label probability is requested, the selected class is used for determining the ordering. In all cases, if there is a NULL value, it remains the first value.

Decision Tree

The Decision Tree is a predictive model. It makes predictions by using the dependent, or known, attribute values to help determine the value of the label, or unknown attribute. The task of predicting the value of a nominal value (usually character strings such as “yes” and “no”), or an attribute that can only take on a small number of values, is referred to as classification. A decision tree classifies data by predicting the label for each record. The underlying structure used for classification is a decision tree. Once the Decision Tree Inducer has built a classifier the data, the 3D Visualizer displays its structure.

The Decision Tree method shows how various attributes interact, that is, how combinations of attribute values affect the predicted label. The predicted label refers to the unknown characteristic in a given record. With the Decision Tree, how the data is distributed at subsequent nodes depends on the decision made at the previous node.

With the completed visualization, the bars at each node of the decision tree represent each label value. You can place the cursor over a bar to show the record count (or weight) and percentage for that label value. At the base of each node is shown the count (or weight) of the records that reach it.

Creating the Decision Tree

A Decision Tree classifier is induced (generated) automatically from data. You begin by logging on to the server and selecting a dataset in the normal manner.

From the Tool Manager choose the *Classify* tab, and from the *Inducer* popup menu select Decision Tree. You do not need further specifications unless you wish. Simply click the *Go* button. The Decision Tree uses the Tree Visualizer for its display. In the resulting visualization:

- Labels on the decision nodes specify the attribute that is tested at that node.
- Leaf nodes in a decision tree specify a class.
- The color of the base indicates the error estimate of the subtree.
- The vertical bars atop each node show the distribution of the classes at the node.

Pointing to a node causes the following information to be displayed:

- *Subtree weight* — The weight of the training set records in the subtree below the node pointed to. This value is mapped to the height of the base.
- *Test set error/loss* — An estimate of the subtree error (or loss if a loss matrix was given). The number after the +/- is the standard deviation of the estimate. The higher the standard deviation, the less accurate the error estimate. The error/loss estimate and the standard deviation are less reliable for leaves with few records or when the test set error is close to 0% or 100%.
- *Test set weight* — The weight of records from the test set that reached the node (number of records if weight was not set).
- *Purity* — A number from 0 to 100 indicating the skewness of the label value distribution at the node. If a node has records from a single class, the purity is 100. If the label values have the same weight, the purity is 0. The purity is computed after backfitting.

Parallelization in IRIX

If you have installed the multiprocessor version of MineSet, in Decision Trees it is possible to compute tree-based algorithms in parallel whenever a branch contains over 1000 records. (See “Parallel Computing on IRIX Systems” on page 133.) You can control the number of threads by changing the parallelization mode in the Preferences panel of Tool Manager (see “File Menu” on page 91). This option is only available on IRIX systems.

Further Inducer Options

When backfitting is enabled and Display training set as disks is checked, the vertical bars show the distribution of the training set as disks. The heights of the disks are on the same scale as the heights of the bars. You should expect the disks to appear about as high on the bars as the value used for the Holdout ratio in Further Inducer Options.

Decision Tree Options

From Tool Manager Decision Tree pane, selecting the *Advanced Options* (or *Further Options* on IRIX) causes the Classifier Advanced Options dialog box to appear. This dialog box consists of four panels:

- The top panel indicates the choices you made in the Tool Manager’s Data Destination panel.
- The second pane from the top lets you set the loss matrix and the weight attribute. See “Loss Matrix” on page 110 and “Weighting” on page 200.
- The bottom-left panel lets you specify further Inducer Options.
- The bottom-right panel lets you specify the Error Estimation Options (unless the mode you chose in the Data Destination panel was Classifier Only, in which case this area is empty). The options shown in this panel depend on the type of Error Estimation you chose (see “Applying a Model” on page 17).

To fine-tune the Decision Tree induction algorithm, you can change certain Decision Tree inducer options. The Inducer Options section of the Classifier options dialog box provides the following:

- Limit tree height by

By default, there is no limit to the height (number of levels) in the Decision Tree. Limit the height by clicking the check box and typing a number for the limit. Limiting the number of levels speeds up the induction, improves parallelization load balance, and is useful for studying the Decision Tree without the distraction of too many nodes. Restricting the size might increase the error rate. Setting this option does not affect the attributes chosen or splits made at levels before the maximum level.

- Splitting criterion

This option offers five splitting criteria selections. The definitions below are technical. For a given problem, it is difficult to know which criteria will be best. Try them all, and select the one that leads to the lowest error estimate, or to a Decision Tree you find easiest to understand.

Mutual Info is the change in purity (that is, the *entropy*) between the parent node and the weighted average of the purities of the child nodes. The weighted average is based on the number of records at each child node.

Normalized Mutual Info (the default) is the Mutual Info divided by the log (base 2) of the number of child nodes.

Gain Ratio is the Mutual Info divided by the *entropy* of the split while ignoring the label values. Normalized Mutual Info and Gain Ratio give preference to attributes with few values.

Chi-squared applies the chi-square statistical independence test to all candidate splits. It then selects the split that leads to the least independent breakdown of the label values.

Gini is the splitting criterion used in CART (Classification And Regression Trees). Like Mutual Info, Gini measures the change in purity between the parent node and the weighted average of the purities of the child nodes. Unlike Mutual Info, Gini calculates the node purity as one minus the sum of the squared label probabilities at that node.

- Split lower bound

This is a lower bound on the weight, or the number of records if weight was not set, that must be present in at least two of the node's children. The default for this option is 2. For example, if there is a three-way split in the node, at least two out of the three children must have a weight of two or more (two records or more if weight is not set). This provides another effective method of limiting the size of the Decision Tree and improving the running time.

Increasing the split lower bound tends to increase the reliability of the probability estimates, because the number of records at each leaf is larger. If you expect the data to contain a lot of noise (errors or anomalies), or if you use the tree for estimating probabilities (see Heading2: Applying a Model), increase the split lower bound to 5 or more. If your dataset is very small (< 100 records), you might want to decrease this number to 1.

- Pruning

A Decision Tree is built based on the limits imposed by Tree Height limits and Splitting criterion. Statistical tests are then made to determine when some subtrees are not significantly better than a single leaf node in which case those subtrees are pruned. There are three pruning options that let you influence pruning for decision trees: Confidence, Cost Complexity, and None.

Pruning is slower than limiting the tree height or increasing the split lower bound because a full tree is built and then pruned. Pruning, however, is done selectively, resulting in a more accurate classifier.

Confidence is the default pruning method. Higher values indicate more pruning; lower values indicate less pruning. The default confidence pruning of 0.7 indicates the recommended amount of pruning to be applied to the Decision Tree. If your data might contain noise (errors or anomalies), increase this number to create smaller trees. The lowest possible value is 0 (no pruning); there is no upper limit. With a pruning parameter of 0, a subtree is pruned only when the error rate of the single node is at least as low as that of the subtree.

Cost complexity trades off the error rate of the tree (its cost) and the number of leaves in the tree (its complexity) and allows you to select trees smaller than the minimum cost tree. This reflects the number of standard errors more than the minimum cost tree that you are willing to accept. Set the parameter to zero for the minimum cost tree; set the parameter to 0.5 for a minimum size tree with an error rate no more than 0.5 standard errors worse than the minimum cost tree. The default setting, 0, selects the minimum size tree that has the minimum cost. Higher numbers indicate more pruning. If your data might contain a lot noise (errors and anomalies),

increase the number to create smaller trees. If the tree is pruned back to a single node, decrease the number to decrease the amount of pruning and show more of the tree's structure. See "Cost Complexity" on page 59 for further details.

None performs no pruning. Although this may produce a tree that overfits the training data, resulting in higher classification error on test data, it allows you to investigate the complete structure of the decision tree.

- Allow one-off splits

Clicking this check box allows the inducer to make two-way splits on nominal attributes that have more than two values. Normally, splits on nominal attributes have as many lines as they have values (see "Creating the Decision Tree" on page 71). For example, a split on the attribute "color" might have lines for red, green, yellow, and blue. If one-off splits are enabled, then the inducer can also make splits with just two lines, for example red, and not red. One-off splits isolate exactly one of the possible values that the column can have. These kinds of splits may be useful when the data contains attributes with many possible values, some of which are exceptionally good at discriminating the label.

- Boosting

Boosting is employed to improve accuracy of classification, although it is a time-intensive process. Visualization is not performed during boosting.

The estimated error appears in the status window, as the process runs. This estimated error is the result of the algorithm repeatedly assigning new weight distributions to the training set, and inducing a classifier on the reweighted sets. See "Boosting" on page 36 for more information.

Search and Filter Panels

Select Search Panel and Filter Panel in the View menu (the Show menu on IRIX) to bring up a dialog box that lets you specify criteria to search/filter for objects. The item choices for decision trees are always the same. These are described below.

The search or filter can be restricted to specific class labels, either by selecting the values in the class list (in the top pane) or by using the class item (in the bottom pane), which allows more powerful comparison operators (such as Matches). Scroll down to get other criteria. Other items are described below:

- *Subtree weight* lets you restrict the search or filter to bars or bases (depending on the choice of the radio button bars or bases) with a given weight (number of records if weight is not set) for the subtree. For example, you can restrict the search to bars containing a weight of at least 50.
- *Test attribute* lets you restrict the search or filter to nodes labeled by the given value that the node is testing. Note that decision node labels represent the test attribute, while leaf node labels show the predicted label. For example, if you select *Test attribute contains age*, only nodes that test the value of age are considered.
- *Test value* lets you restrict the search or filter to nodes having an incoming line labeled with a value you specify.
- *Percent* lets you restrict the search or filter to bars representing a percentage of the overall weight at a node. For example, you might want to find all nodes such that a given class accounts for more than 80 percent of the weight. To do this, click the class label, and select *Percent > 80*. Setting this item is meaningless if you select bases and not bars (the value for the bases is 0).
- *Purity* lets you restrict the search or filter to nodes with a range of purity levels. For example, if you want to look at pure nodes (with one class predominant), you can select *Purity > 90*.
- *Test-set subtree weight* lets you restrict the search or filter to subtrees with a given test-set weight (number of test-set records if weight is not set).
- *Test set error/loss* lets you restrict the search or filter to nodes with a range of estimated error or loss.
- *Mean error/loss standard deviation* lets you restrict the search or filter to nodes with a range of estimated standard deviation for the test set error or loss.
- *Level* lets you restrict the search or filter to a specific level or range of levels. For example, you can search only the first five levels.

The following items and options are less useful for decision trees.

- *Hierarchy* finds all the nodes and lines that match the given value at the tail of the path from the root. It then marks the children of these nodes.
- *Treat Nulls as Zeros* is not used by the Decision Tree inducer because there are no null items generated for decision trees.

Once the search is complete, yellow spotlights highlight objects matching the search criteria. To display information about an object under a yellow spotlight, move the pointer over that spotlight; the information appears in the upper left corner, under the label "Pointer is over:." To select and zoom to an object under a yellow spotlight, left-click the spotlight; if you press the Shift key while clicking, zooming does not occur.

Once the filtering is complete, the scene shows only nodes matching the filtering criteria.

Discrete Label Menu

The Discrete Label menu on the Classify tab of the Tool Manager provides a list of possible discrete labels. Discrete attributes (binned values, character string values, or a few integers) have a limited number of values. It is wise to select a label attribute with few values; for instance, two or three. If there are no discrete attributes, the menu shows No Discrete Label, and the *Go* button is disabled. You then must create a discrete attribute by binning or adding a new column using the Tool Manager's Data Transformations panel.

Drill Through

It is often useful to see the original data that corresponds to the selections in your current visualization. This is referred to as *drill through*. By selecting objects in a visualization, and sending the underlying data to the Tool Manager you can view the original data or analyze it using another tool. All tools in MineSet have this capability, although certain tools have additional characteristics which are detailed at the end of this entry.

If you choose Selections > Drill Through the Drill Through Dialog box opens. This dialog box gives you the following choices for viewing the selected records:

- Show original data in recordviewer.
- Send to ToolManager as Filter.
- Send to ToolManager as new column.
- Send to ToolManager as SQL.

You can also choose one of the following

- Send the records to a new copy of ToolManager.
- Send all non-selected records to the Tool Manager (Complement drill through).

Only visualizations generated using the Tool Manager can be used for drill-through. Each *.schema* file for these visualizations includes a *history* section that informs the Tool Manager how that file was generated. A few special-purpose mining visualizations such as Learning Curves, Confusion Matrices, and Lift Curves, do not include a history and do not support drill-through.

When you select objects for drill through, you are implicitly specifying a filter statement based on the visualized table. If the table was transformed before visualization, the Tool Manager might have to change the filter to place it earlier in the history. For example, if the filter is based on a binned column, the Tool Manager must change it so it refers to the pre-binned column.

The filter cannot always be placed at the beginning of the history because the Tool Manager cannot adjust the filter for all operations. Specifically, if the filter refers to any column created via *Add Column*, *Aggregate*, or *Apply Model*, the filter cannot be placed earlier in the history than the operation that created the column. Furthermore, the Tool Manager can never move a filter earlier than an existing *Sample* operation, since that would dramatically change the output of the sampling.

The Show original data mode tries to truncate the history to show the records from as early a stage as possible. To prevent this mode from showing more records than were selected, however, the truncation does not remove any other existing filter operations (either user-created or the result of previous drill through).

Drilling Down and Drilling Up

Drilling down or drilling up refers to the act of clicking on an entity to see more or less detail. This is available only in the Map and Decision Table Visualizers.

Drill down using a right mouse click on an object. Drill up holding the Ctrl key and clicking the right mouse button on an object (or use the middle mouse button). Drilling up or down on the background performs the operation globally.

Error Estimation

When a classifier is built, it is useful to know how well you can expect it to perform in the future (what is the classifier's error-rate). Factors affecting classification error-rate include:

- The number of records available in the training set

Since the inducer must learn from the training set, the larger the training set, the more reliable the classifier should be; however, the larger the training set, the longer it takes the inducer to build a classifier. The improvement to the error-rate decreases as the size of the training set increases (this is a case of diminishing returns).

- The number of attributes

More attributes mean more combinations for the inducer to compute, making the problem more difficult for the inducer and requiring more time. Note that sometimes random correlations can lead the inducer astray; consequently, it might build less accurate classifiers (technically, this is known as "overfitting"). If an attribute is irrelevant to the task, remove it from the training set (this can be done using the Tool Manager).

- The information in the attributes

Sometimes there is not enough information in the attributes to correctly predict the label with a low error-rate (for example, trying to determine someone's salary based on their eye color). Adding other attributes (such as profession, hours per week, and age) might reduce the error-rate.

- The distribution of future unlabeled records

If future records come from a distribution different from that of the training set, the error-rate probably will be high. For example, if you build a classifier from a training set containing family cars, it might not be useful when attempting to classify records containing many sport cars, because the distribution of attribute values might be very different.

The two common methods for estimating the error-rate of a classifier are described below. Both of these assume that future records will be sampled from the same distribution as the training set.

- Holdout

A portion of the records (commonly two-thirds) is used as the training set, while the rest is kept as a *test set*. The inducer is shown only two-thirds of the data and builds a classifier. The test set is then classified using the induced classifier, and the error-rate or loss on this test set is the estimated error-rate or estimated loss. Figure 1-17 shows this error estimation method.

When backfitting is enabled and Display training set as disks is checked, the vertical bars show the distribution of the training set as disks. The heights of the disks are on the same scale as the heights of the bars. You should expect the disks to appear about as high on the bars as the value used for the Holdout ratio in Further Inducer Options.

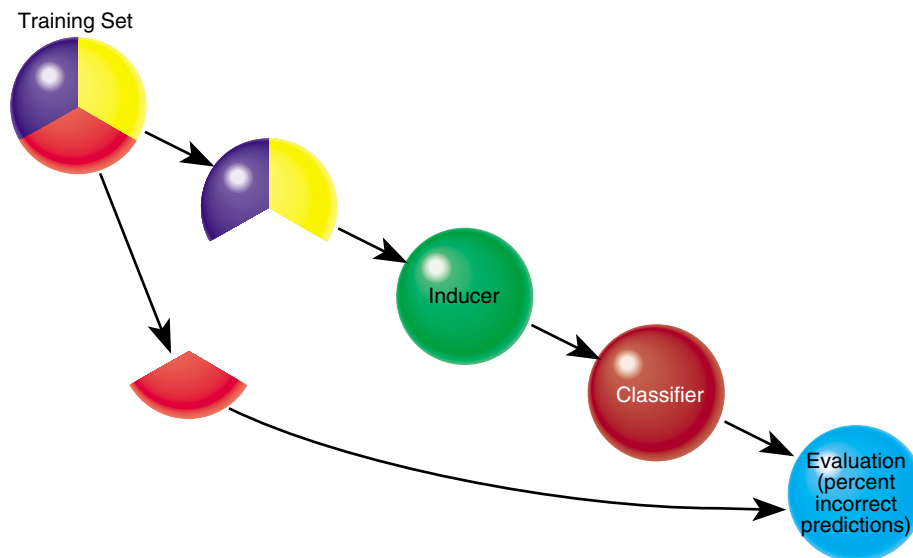


Figure 1-17 Estimating the Classifier's Accuracy

This method is fast, but since it uses only two-thirds of the data for building the classifier, it does not make efficient use of the data for learning. If all the data were used, it is possible that a more accurate classifier could be built.

- Cross-validation

The data is split into k mutually exclusive subsets (*folds*) of approximately equal size. The inducer is trained and tested k times; each time, it is trained on all the data minus a different fold, then tested on that holdout fold. The estimated error-rate is then the average of the errors obtained. Figure 1-18 shows cross-validation with $k=3$ (note that the default value is $k=10$).

Cross-validation can be repeated multiple times (t). For a t times k -fold cross-validation, $k*t$ classifiers are built and evaluated. This means the time for cross-validation is $k*t$ times longer. By default, $k=10$ and $t=1$, so cross-validation takes approximately 10 times longer than building a single classifier.

Increasing the number of repetitions (t) increases the running time and improves the error estimate and the corresponding confidence interval.

You can increase or decrease k . Reducing it to 3 or 5 shortens the running time; however, estimates are likely to be biased pessimistically because of the smaller training set sizes. You can increase k , but this is recommended only for very small datasets.

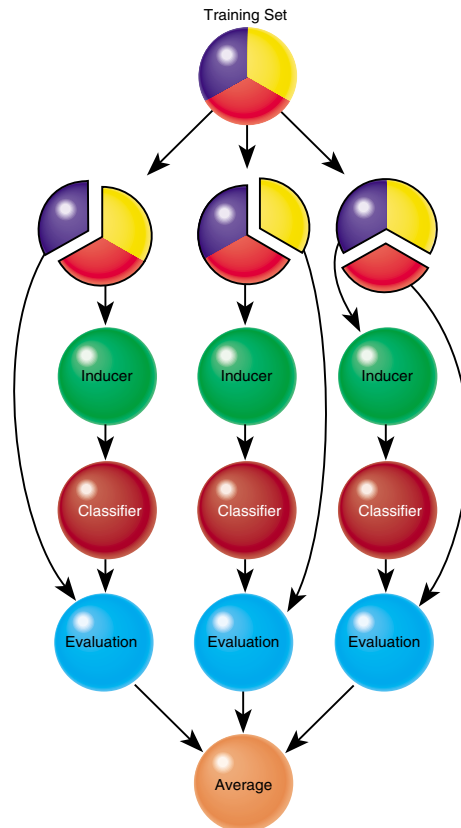


Figure 1-18 Classifier Cross-Validation ($k=3$)

Generally, a holdout estimate should be used at the exploratory stage, as well as on datasets of over 5,000 records. Cross-validation should be used for the final classifier building phase, as well as on small datasets.

Backfitting applies the full dataset to a classifier or model that has been built using only data from the training set. When using holdout error estimation, you leave out a portion of the data for testing. When you backfit all the data through the classifier structure, you can reduce the error of the final classifier, because counts, weights and probabilities reflect the entire dataset. For a full discussion, see “Backfitting” on page 32.

Evidence Model

The Evidence classifier model assigns each record in a dataset to a class. The Evidence Visualizer displays the structure of an evidence model. The visualizer can help you understand the importance of specific attribute values for classification. Also, it can be used to gain insight into how classification is done, as well as to answer “what if” questions. See the *MineSet Enterprise Edition User’s Guide for Windows* for information about how to use and interpret the Evidence Visualizer.

Evidence Inducer

The Evidence Inducer builds a model that assumes the probabilities of each attribute value are independent given the class. For example, the *iris* dataset assumes that the four attributes (sepal length, sepal width, petal length, and petal width) are independent for each class of iris (*iris-setosa*, *iris-versicolor*, and *iris-virginica*). While this simplistic model is rarely true, the model is excellent for initial explorations of data and its classification prediction performance is very good in practical applications.

Each attribute value, or range of values (for binned continuous attributes) defines exactly one chart, which, in turn, gives the conditional probabilities for each class label. To classify a given record, one computes the probability of each class by multiplying its prior probability by the appropriate conditional probability from each row in the matrix. The final product gives the relative probability for each class and the highest value is the predicted class. If an attribute has an unknown value, it is ignored. These NULL values are represented by charts that are slightly offset from the rest of the charts and are on the left when you bring up the *iris.schema* file from the examples provided with MineSet. See Appendix A, “Sample Configuration and Data Files,” for pathnames, or follow the directions given in the *MineSet Enterprise Edition User’s Guide for Windows*.

When you have the iris dataset shown with the Evidence inducer, the *prior probability* for each class label is depicted in the pie chart in the Label Probability Pane, on the right of the screen. The prior probability for a class label is the probability of seeing this label in the data for any randomly chosen record, ignoring all attribute values. Mathematically, this is the number of records with the class label divided by the total number of records. Toggle between Evidence (cakes) and Probability (pies) in the Main Window by clicking the Evidence label or by toggling the Evidence Mode in the View menu.

The *conditional probabilities*, depicted normally by cake charts in the Main Window on the left of the screen, show the relative probability of each attribute value given (conditioned on) each label value. The size of a cake slice indicates the amount of evidence the classifier adds to the prior probability after taking into account a given attribute value in a record. If the size of the slices are equal, the value is irrelevant, and the classifier adds the same amount of evidence to all classes.

Technically, the slice of the chart represents the normalized conditional probability of an attribute value A , given the class label L . The conditional probability, $P(A | L)$, is the probability that a random record chosen only from records with label L takes the value A . Under the default settings, the probability is computed based on record weights. For example, $P(0.75 < \text{petal width} < 1.65 | \text{iris-versicolor})$ is 91.6, because there are 36 records with label iris-versicolor, and 33 of them have a petal width in this range.

The height of the gray base in bars/evidence mode shows the evidence for the selected label. This overall evidence value is computed by $-\log[1 - P(L) / \sum(P(L_i))]$ for evidence for, and by $-\log[P(L) / \sum(P(L_i))]$ for evidence against. The subtract min evidence toggle affects the base the same as the small bars.

Importance is a measure of predictive power with respect to a label. The Main Window provides valuable insight not only into the importance of each attribute value affecting the class value, but also into the importance of specific attribute values.

Inducing Evidence Classifiers

The automatic induction of Evidence classifiers is a process whereby counts (or weights) are used to calculate the probabilities. Evidence classifiers are automatically induced (generated) from data. The data, which is made up of records and a label associated with each record, is called the *training set*.

The probabilities are generated using the following method:

1. All continuous attributes are discretized (binned), such that class distributions in these ranges are as different as possible. The number of ranges is determined automatically, which invokes parallelization so that attributes are binned in parallel, if the multiprocessor version of MineSet is installed on your system. The automatic binning for any attribute may be overridden by explicitly performing the binning operation in Tool Manager.
2. The prior probabilities are the proportions of each class in the training set.
3. The conditional probabilities are the probabilities of each attribute value conditioned on each class label in the training set. (The charts show them normalized for each attribute value.)

The number of charts in a row is the number of discrete ranges produced by the inducer. If there is just one range, it means that this attribute by itself was not useful in predicting the label. Initially, the prior probabilities of the labels are displayed in the Label Probability Pane.

Each attribute value, or range of values, (for binned continuous attributes), defines exactly one chart, which, in turn, gives the conditional probabilities for each class label.

- Laplace Correction

This biases the probabilities towards the average, thus avoiding extreme numbers (such as 0 and 1). If the Laplace correction is checked in the Evidence Inducer *Advanced Options* (or *Further Inducer Options* on IRIX systems) dialog box, and the factor is left empty or set to 0, an automatic Laplace correction is applied, using a heuristic that applies a factor of $1/\text{training-set-weight}$. See also “Laplace Correction” on page 107.

- Minimum Weight per Bin

The Evidence Inducer discretizes all continuous attributes. This option lets you define the minimum number of instances per bin. The automatic setting has a heuristic that sets this number based on the dataset size: the larger the dataset, the greater the minimum number of records allowed in the bin, and the smaller the width of the bin, in general. If your dataset is very large, you might obtain more discrete ranges than you want. To reduce the number of bins, raise this value.

- Automatic column selection

This applies a process that chooses only those columns that help prediction the most. Because extra columns can degrade the predictive accuracy of the evidence classifier, this process searches for a good subset of the columns automatically. Only those columns found to be useful are used. This process can take a long time, especially if there are many columns. It is useful for eliminating highly correlated columns that could degrade accuracy. Automatic column selection and Boosting cannot both be enabled, as together they would take far too long to complete.

Automatic column selection conducts a search for the best set of columns that reduce the error of the classifier. The selection of these columns is done by estimating the error of different attribute sets using the wrapper approach. Each feature subset is evaluated by estimating the classifier's error using cross-validation. Columns are added or removed based on the error estimates using a best-first search mechanism. In the default mode, the search begins with an empty set of features. By selecting the *Backwards* option, the search starts with the full set of options; this is slower, since larger models are initially built (for more information, see the *MineSet Enterprise Edition Interface Guide*).

Starting the Evidence Visualizer

There are several ways to start the Evidence Visualizer:

- Run the Evidence Inducer from the Tool Manager under the Classify tab. After the inducer builds the classifier, it automatically invokes the Evidence Visualizer. See below for details about using the Tool Manager in conjunction with the Evidence Visualizer.
- Use the Tool Manager to start the 3D Visualizer from the Visual Tools menu.
- If you know what configuration file you want to use, double-click the diamond-shaped icon for that configuration file. This starts the 3D Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.eviviz* (which is always the case for configuration files created for the Evidence Visualizer via the Tool Manager).
- Start the Evidence Visualizer from the UNIX command line by entering:

```
eviviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don't specify a configuration file, you must use File > Open to specify one.

UNIX Options for Invoking the Evidence Visualizer

The `-quiet` option eliminates the dialogs that popup to indicate progress. You can enable this option permanently by adding the line

```
*minesetQuiet:TRUE
```

to your `.Xdefaults` file. See also “Warning Options” on page 199.

On Windows, this option is available from the 3D Visualizer Preferences panel.

Evidence Inducer Options

Selecting Advanced Options (Further Inducer Options on IRIX) causes the Inducer Options dialog box to appear. This dialog box consists of three panels:

- The top panel shows the choices you made in the Tool Manager’s Data Destination panel. The type of Error Estimation is determined by the model.
- The bottom-left panel lets you specify further Inducer Options (described below).
- The bottom-right panel lets you specify the Error Estimation Options (unless the mode you chose in the Data Destination panel was Classifier Only, in which case this area is empty). The options shown in this panel depend on the type of Error Estimation you chose (see “Error Estimation” on page 79).

If a Loss Matrix has been specified in the Advanced Inducer options, a button labelled *Use Loss Matrix* appears to the lower right of the probability pie. When selected (the default) the Loss Matrix is used to adjust the probabilities shown. The largest slice shown is the class that takes into account the Loss Matrix. To see what the probabilities would be without using the Loss Matrix, deselect the *Use Loss Matrix* button. A gray slice indicates the presence of a column edited to predict null. If the gray slice is the largest slice, the classifier predicts null. For further details, see “Loss Matrix” on page 110.

Selecting Items in the Evidence Visualizer Main Window

In select mode, the cursor appears as an arrow. You can highlight an object (either a pie chart or a bar) by moving the cursor over that object. Information about that object then appears above the Main Window. The information is displayed as long as the cursor is over the object.

- If the object is a pie chart, then the message takes this format:

```
<attribute name>: <value or range>
weight = <weight>
```

Here, *weight* is the total weight of the data points that fall in that range or have that value for that attribute. The chart height is proportional to this number. Unless record weighting is used, the weight shows record counts.

- If the object is a bar, then the message takes this format:

```
(<attribute> = <value>) ==> Prob(<selected label>) = x% [low%-high%]
Evidence=z
<selected label> ==> Prob(<attribute> = <value>) = y% [low%-high%]
weight = <weight>
```

Here, x is the probability that a record has the selected label given that it has the highlighted attribute value. The bracketed range, [low%-high%] gives the 95% confidence interval. Similarly, $y\%$ is the probability that a record has the highlighted attribute value given the selected label. The height of the bar shows evidence, not probability. The amount of evidence, z , is directly related to the bar heights. Evidence can be summed in order to determine which class is predicted (unlike probability, which must be multiplied). *Weight* is the weight of data points having that value.

Technically, *evidence for* is defined as

$$-\log \left[1 - \frac{P(A|L)}{\sum_{i=1}^N P(A|L_i)} \right]$$

while *evidence against* is defined as

$$-\log \left[\frac{P(A|L)}{\sum_{i=1}^N P(A|L_i)} \right]$$

A is the attribute value, L is the selected label value, and N is the number of label values. When computing the bar heights, a very small number is added inside the brackets of the above expressions to prevent the bars from becoming infinitely tall. The word “for” or “against” in the Main Window has a box around it to indicate that it may be clicked on. Click on the box to toggle the representation.

The height of the gray rectangular base (on which the bars stand) represents the amount of evidence contributed by the prior probability. For example, if the label is car cylinders, there are very few three cylinder cars, so the base is low when *evidence for* is showing, and high when *evidence against* is showing. You can add to this height the height of individual bars that are on top.

Evidence for can be useful in determining which values are the most helpful in predicting a particular label value.

The amount of evidence (bar height) is not derived directly from either probability shown while highlighting. Instead, the evidence depends on the conditional probability relative to the other probabilities for all the other label values according to the equation above.

You can also select any number of values from an attribute row by clicking the left mouse button while the cursor is over one of the attribute values. The large pie chart in the Label Probability Pane on the right changes to reflect items you select; it now shows the posterior probability, given the attribute values just selected. The classes remain ordered, so the one corresponding to the largest slice is at the top of the list on the right. The Evidence Visualizer arrives at the new posterior probability distribution by multiplying the conditional probabilities for each attribute together, then multiplying this result by the prior probability and normalizing to one.

This multiplication corresponds to a conditional independence assumption. When this assumption is violated, and several attributes are instantiated, the probabilities of the predicted class are likely to be wrong, (even though the final classification is correct). The estimated error shown in the Status window when you run the inducer can help you determine how reasonable this assumption is. If the error rate/loss is low, the assumption is reasonably robust in the domain.

When a particular label has been selected in the Label Probability Pane, the Main Window shows bars rather than cakes or pies for each value of an attribute. The title over the bars reads “Evidence For.” The box around the “For” indicates that it can be selected.

Clicking the “For” in the “Evidence For” title toggles it to display “Against”. As a result, the bar heights change to show evidence against the label.

Selecting bars has the same effect on the large probability pie in the Label Probability Pane to the right as did selecting cakes or pies. The bar height indicates the amount of evidence for or against the selected label contributed by that selected value. Since log probabilities are used to represent evidence, the bar heights are added to accumulate evidence (whereas probabilities must be multiplied).

Evidence Visualizer Menus

Five pulldown menus let you access additional Evidence Visualizer functions: File, View, Selections, Nominal Order and Help. If you start the Evidence Visualizer without specifying a configuration file, only the File and the Help menus are available. See the File and Help entries for menu-specific details.

View Menu

The View menu lets you control certain aspects of what is shown in the Evidence Visualizer pane. The common menu items are detailed in “View Menu” on page 185. Depending on your platform, the menu also contains some additional options:

- *Show as Evidence* toggles the display of Evidence. If checked, the scene on the left shows evidence. If unchecked, the scene shows probabilities.
- *Sort By Importance* lets you display the attributes sorted according to their usefulness in classifying with respect to the chosen label. If this option is turned off, then the attributes will appear in the same order they did under “Current Columns” in the Tool Manager.
- *Subtract Minimum Evidence* applies only when a label has been selected and the bars are shown. With this option on (the default), the height that is the minimum over all the label values is subtracted. This amount may be different for each value of each attribute, but for a given attribute value, the amount subtracted is constant across label values. Activating this option magnifies small differences by subtracting the least common denominator among all the label values.
- *Use Laplace Correction* toggles the use of Laplace correction. If a specific Laplace correction was specified in the further inducer options dialog box, then that value is used, otherwise a default value is provided.
- *Use Landscape Viewer* (or *Use Examiner Viewer*) switches to an alternative mode of 3D navigation.

Nominal Order Menu

The Nominal Order menu lets you control how values for nominal attributes are ordered. The choices are detailed in “Nominal Order Menu” on page 128.

Selection Menu

The Selection menu allows drill through to the underlying data. To do a drill through, first select some combination of values and/or a class, then chose one of the two methods of drilling-through to the underlying records. This menu is similar for several tools, see “Selection Menu” on page 153 for menu item explanations.

File Menu

The File Menu for most of the visualizers is similar and can contain a number of options:

Windows Systems

- *Open* loads and opens a configuration file, displaying it in the main window. Previously displayed data is discarded. Use *Open* to view a new dataset, or to view the same dataset after changing its configuration.
- *Reopen* reopens the currently opened file. This can be used after the configuration or data file has been updated.
- *Save Image* saves the state of the current visualizer window into an image file. You can choose whether to save the entire window, including any legends, or just the main scene with the graphical objects. You can also choose the Offscreen Render option, which allows you to save the image in bitmapped (.bmp), JPEG, Post Script (.ps), or Tagged Image File Format (.tif) format.
- *Print Image* outputs the state of the current window to a printer. You can specify the output printer using the Print dialog panel (default is your system's default printer) and, like the *Save Image* dialog, choose whether to print the entire window or just the main scene window.
- *Print Preview* displays the potential print output.
- *Print Setup* opens a dialog panel to setup your print output (default is your system's default printer).

- *Publish on the Web* prints the current file to a Web-compatible file.
- *Preferences* starts the Preferences dialog to specify mouse mappings and to toggle warnings on the execution of commands. Sound effects and spin animation as well as default visualizer font sizes are specified here.
- *Start Tool Manager* starts the Tool Manager (if not already running), and restores it to the state it was in when the Tree Visualizer was invoked.
- *Recent File* opens any recent file no matter which visualizer was used. The menu lists the last four files that were opened.
- *Exit* closes all windows and exits the application.

UNIX Systems

- *Open* loads and opens a configuration file, displaying it in the main window. Previously displayed data is discarded. Use *Open* to view a new dataset, or to view the same dataset after changing its configuration.
- *Reopen* reopens the currently opened file. This can be used after the configuration or data file has been updated.
- *Save As* saves the state of the current visualizer window into an image file. The user specifies both the file name (for Tree Visualizer the default is *treeviz.rgb*), format (default is *rgb*), and whether to save the entire window, including any legends, or just the main scene with the graphical objects (default is the full window).
- *Print Image* outputs the state of the current Visualizer window to a printer. You can specify the output printer using the Print dialog panel (default is your system's default printer) and, like the *Save As* dialog, choose whether to print the entire window or just the main scene window.
- *Publish on the Web* saves the visualization as a *.mtr* file, suitable for publishing on the web.
- *Start Tool Manager* starts the Tool Manager (if not already running), and restores it to the state it was in when the Tree Visualizer was invoked.
- *Exit* closes all windows and exits the application.

File Requirements

Most MineSet visual tools require at least two files in addition to a configuration file: the *.data* file and the *.schema* file, which are created automatically when you access the tools with the Tool Manager.

The *.data* file consists of rows of tab-separated fields. You can also generate data files by extracting data from a source (such as an Oracle, INFORMIX, or Sybase database) and formatting it using your favorite text editor (such as WordPad, jot, vi, or Emacs). See the *MineSet Enterprise Edition Interface Guide* for the required file format. Table 1-10 gives the types of file extensions created for each visual tool when you use the Tool Manager.

Table 1-10 Sample Default File Extensions

Tool	Data File Extension	Schema File Extension	Configuration File Extension
Association Rules	<i>.rules.data</i>	<i>.rules.schema</i>	<i>.rules.scatterviz</i>
Cluster Visualizer	none	none	<i>.clusterviz</i>
Decision Table Visualizer	<i>.dtableviz.data</i>	none	<i>.dtableviz</i>
Evidence Inducer	none	none	<i>.evidviz</i>
Map Visualizer	<i>.mapviz.data</i>	<i>.mapviz.schema</i>	<i>.mapviz</i>
Record Viewer	<i>.data</i>	<i>.schema</i>	none
Scatter Visualizer	<i>.scatterviz.data</i>	<i>.scatterviz.schema</i>	<i>.scatterviz</i>
Splat Visualizer	<i>.splatviz.data</i>	<i>.splatviz.schema</i>	<i>.splatviz</i>
Statistics Visualizer	none	none	<i>.statviz</i>
Tree Visualizer	<i>.treeviz.data</i>	<i>.treeviz.schema</i>	<i>.treeviz</i>

When starting a visualizer or when opening a file, you must specify the configuration file, not the data or schema file. The Record Viewer, however, opens any *.schema* files.

Data files can have user-defined extensions (the sample files provided have a *.data* extension).

Filter Button

This button on the Data Transformation pane of Tool Manager lets you filter the data via a mathematical expression. The resulting table includes only records for which the expression is true (or, if numerical, non-zero). When you click *Filter*, the Filter dialog box appears.

This dialog box lets you select column names and operators on the left to build an expression on the right. For a complete description of the expression definition language, see “Adding Columns” on page 1.

Filter Panel

The Filter Panel is accessible from the View pulldown menu. This section describes the options. The Scatter, Splat, Map, and Decision Table Visualizers use this panel. The Tree Visualizer has a similar panel. The Evidence Visualizer has its own variant.

Filter Panel brings up a panel which lets you reduce the number of entities displayed in the main viewing area, based on one or more criteria. You can use the filter panel to fine-tune the display, emphasize specific information, or simply shrink the amount of information displayed.

Scale to Filter (only in the Scatter Visualizer) lets you specify whether the heights of the graphical objects are scaled across the entire dataset or just across the filtered data.

The filter panel has two sections. The top part lets you filter based on string variables. To select all values of a variable, click *Set All*. To clear the current selections, click *Clear*. To select a value, click it. To deselect a value, simply click it again.

The bottom part lets you filter based on the values of both string and numeric variables. Only variables whose values do not change as you navigate the slider can be used in filtering. Scrolling the pane reveals the hierarchy (Contains, Equals, Matches, IsNull).

To filter numeric values, enter the value and select a relational operation (=, !=, >, <, >=, <=). To filter alphanumeric values, enter the string. You can use any of three types of string comparisons:

- Contains indicates that it contains the appropriate string. For example, California contains the strings Cal and forn.
- Equals requires the strings to match exactly.
- Matches allows wildcards:
 - An asterisk (*) represents any number of characters.
 - A question mark (?) represents one character.
 - Square braces ([]) enclose a list of characters to match.

For example, California matches Cal*, Cal?ornia, and Cal[a-z]ornia.

In some cases (usually associated with binning in the Tool Manager), an option menu of values appears, instead of a text field. To ignore that variable, select *Ignored* in the Option menu. You can use relational operators (such as >=) with these options. This means that the specified value as well as subsequent ones are selected.

In addition to numeric and string comparison operations, you can specify `Is Null`, which is true if the value is null.

To the right of each field is an additional option menu that lets you specify “And” or “Or” options. For example, you could specify “sales > 20 And < 40.” You can have any number of And or Or clauses for a given variable, but cannot mix And and Or in a single variable.

Click the *Apply* button to start filtering. If you press *Enter* while the panel is active, filtering starts automatically.

Gain Ratio

Gain Ratio is a Decision Tree splitting criteria consisting of the Mutual Info divided by the entropy (change in purity) of the split made while ignoring the label values. Refer to “Decision Tree” on page 70 for a discussion of the Decision Tree splitting criteria.

Help (IRIX)

Besides the F1 or Shift-F1 keyboard key combination, the Help menu in each tool provides access to five help functions:

- *Click for Help* turns the cursor into a question mark. Placing this cursor over an object in the visualizer's main window and clicking the mouse causes a help screen to appear; this screen contains information about that object. Closing the help window restores the cursor to its arrow form and deselects the help function. The keyboard shortcut for this function is Shift+F1. (Note that it also is possible to place the arrow cursor over an object and press the F1 function key to access a help screen about that object in some tools.)
- *Overview* provides a brief summary of the major functions of a tool, including how to open a file and how to interact with the resulting view.
- *Index* provides an index of the complete help system. This option is currently disabled.
- *Keys & Shortcuts* provides the keyboard shortcuts for all of the visualizer's functions that have accelerator keys.
- *Product Information* brings up a screen with the version number and copyright notice for the visualizer.

Help (Windows)

Besides the F1 key, the Help menu in each tool provides access to all of the MineSet Enterprise Edition manuals. *Product Information* brings up a screen with the version number and copyright notice for the visualizer.

Histogram Visualizer

The Histogram Visualizer automatically bins all of the continuous-type columns in the data and sends the result to the Statistics Visualizer. See “Statistics Visualizer” on page 168. You can set the following Histogram Visualizer options:

- You can pick the number of bins or allow MineSet to do it for you.
- You can set the trimming fraction. The trimming fraction indicates the fraction of extreme values to be excluded from the value range prior to generating bins. The default trimming fraction is 0.05. This excludes the 5% of the instances with the most extreme values (2.5% with the lowest values in the range and 2.5% with the highest values in the range). Trimming tends to reduce the influence of outliers on the generation of thresholds.

History

You can get the history of previous operations in the current session by using the *Table History* buttons in Tool Manager. See “Table History Buttons” on page 171. You can also save your work using the File pulldown menu. You can also save the data file after completing your transformations by clicking the Data File tab in the Tool Manager Data Destinations pane and then saving the file. When you resume work, load the appropriate session or file you want to work with and the history is there.

Holdout

Holdout refers to the process of training a model on a sample of the dataset so that the model may be tested on the remaining portion. This procedure assesses the model’s error rate based on data which was not used to train the model. The fraction of data which is used for training is referred to in MineSet as the Holdout Ratio.

Inducers

An inducer is an algorithm that builds a predictive model from a *training set*, which consists of records with labels. The training set is part of the dataset used by the inducer to “learn” how to construct the model. Once the model is built, its structure can be visualized or used to classify unlabeled records. Running inducers can be a CPU- and I/O-intensive process. For this reason, the MineSet inducers run on the MineSet server, rather than on the MineSet client (see Figure 1-19).

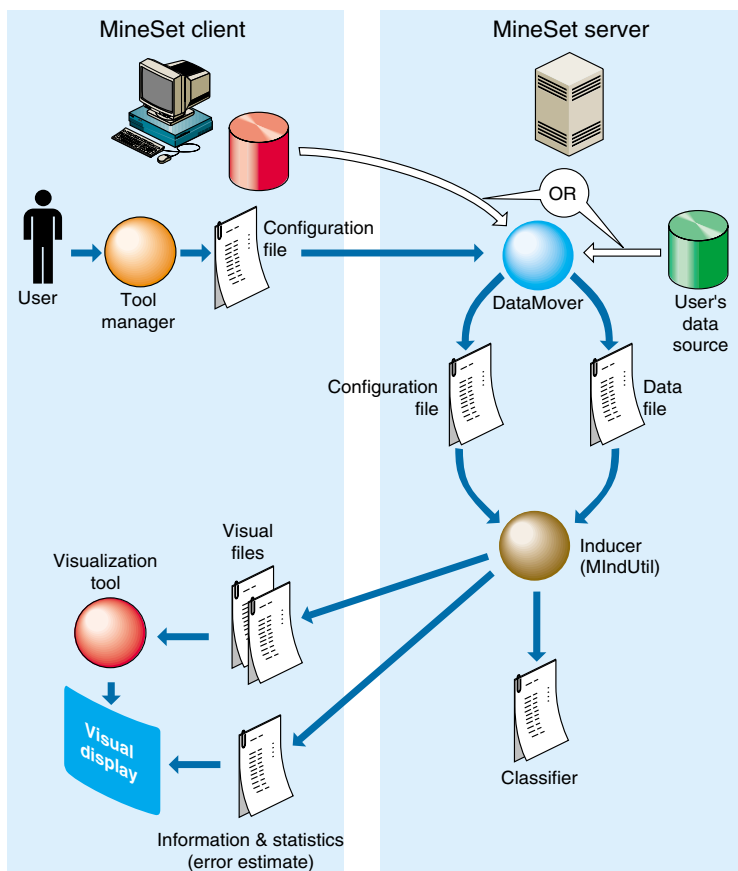


Figure 1-19 Tool Execution Sequence for Models

Inducers require a training set, which is a table with attributes or characteristics, one of which is designated as the label. Once a model is built, it can predict this label for new records. These new records must be in a table that has all the attributes used by the model, with the same name and type as they were in the training set. The table need not contain the label attribute. If it exists, it is ignored during prediction.

Inducer Modes in Tool Manager

There are four modes for running an inducer.

- Classifier and Error (or Regressor and Error)
- Classifier Only (or Regressor Only)
- Estimate Error
- Learning Curve

The *Classifier and Error* mode (or *Regressor and Error* mode) uses a holdout method to build a model: a random portion of the data is used for training (commonly two-thirds) and the rest for testing. This holdout proportion can be set in *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems (see “Error Estimation” on page 79). This method is the default mode and is recommended for initial explorations. It is fast and provides an error estimate.

The *Classifier Only* (or *Regressor Only*) mode uses all the data to build the model. There is no error estimation. Use this mode when there is little data or when you build the final model.

The *Estimate Error* mode assesses the error of a model that would be built if all the data were used (as with *Classifier Only* or *Regressor Only* mode). Estimate Error uses cross-validation, resulting in long running times. (See “Cross Validation” on page 60.) Use this method when there is little data. The induced model is exactly the same as the one induced by the *Classifier Only* or *Regressor Only* mode.

The *Learning Curve* mode assesses the effect of training set size on the error rate of a classifier (see “Learning Curve” on page 107).

Inducer Error Options

The following options are available to fine tune the error estimation for the inducers. The Error Options available to you depend on the inducer mode you have chosen.

In both *Classifier & Error* (or *Regressor and Error*) and *Estimate Error*, you can set a random seed that determines how the data is split into training and testing sets. Changing the random seed causes a different split of the data into training and test sets. If the error estimate varies appreciably, the induction process is not stable.

In *Classifier & Error* (or *Regressor and Error*) you can set the Holdout Ratio of records to keep as the training set. This defaults to 0.666667 (two-thirds). The rest of the records are used for assessing the error.

In *Estimate Error* you can set the number of folds in cross validation and the number of times to repeat the process. See “Cross Validation” on page 60.

Advanced Inducer Options

MineSet supports several advanced options for all inducers. You can take into account the cost of making certain kind of mistakes, or compensate for a non-uniform sampling process (in which some parts of the true population are sampled more heavily than others). Another option lets you create more complicated models which may have better accuracy, at the expense of added compute time.

- **Backfitting**—The *Backfit test set* option is a check box that can be found under *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems for all inducers when using Classifier & Error mode. The backfit check box is disabled when Boosting is enabled. See “Backfitting” on page 32.
- **Confusion Matrix**—The *Display Confusion Matrix* option is a check box under *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems for all classification inducers when using Classifier & Error mode. See “Confusion Matrix” on page 58.
- **Return-on-Investment Curve**—A Return-on-Investment (ROI) curve is similar to a Lift Curve, but displays accuracy in terms of loss rather than in terms of error; taking into account the Loss Matrix used. The *Display ROI Curve* option is a check box under *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems for all classification inducers. An ROI curve requires a label value to

be chosen. An ROI curve is then generated and displayed for that label value. See “Return-on-Investment Curve” on page 143.

- **Lift Curve**—A lift curve is a graph that shows the difference between a random ordering of records and that created by the classifier in describing a particular label value. See “Lift Curve” on page 109.
- **Loss Matrix**—A Loss Matrix lets you change or reweight the cost of making certain kinds of errors. Loss matrices, together with confusion matrices help mitigate the cost of errors. The *Use Loss Matrix* option is a check box near the top of the dialog box under *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems for all inducers. See “Loss Matrix” on page 110.
- **Weight Setting**—Record weighting lets you give each record a weight; a subpopulation that was sampled twice as frequently might get a weight of 0.5, while the rest of the population is given a weight of 1. The *Use Weight* option is a check box near the top of the dialog box under *Advanced Options* on Windows systems, or *Further Inducer Options* on IRIX systems for all inducers. Choose the column for the weight. The *Weight is Attribute* option determines whether the inducer can use this attribute for modeling purposes or not. In certain cases where the weight is a result of a stratified sample that is part of the experimental design, the model should not be given access to the weight column as it is not a property of the real-world entity.
- **Learning Curve**—A learning curve is a graph that shows the error of the model generated by an inducer as a function of the number of records used to create it. Typically, the more records used to generate the model, the lower its error. See “Learning Curve” on page 107.

Setting Special Options

- Attributes of type arrays are always ignored.
- Dates are considered strings. Unless there are few dates, such attributes are usually ignored because of the limit on discrete attributes. You should bin dates before running an inducer.

Inducer Status Window

After you press *Go!* in the Data Destination panel, the Status Window at the bottom of the Tool Manager's main window shows the algorithm's progress and displays specific information for the induced model. For example, for Decision Trees it shows the number of nodes, the number of leaves, and the depth of the Decision Tree. This information is saved automatically on your workstation under the session file name with a *.out* extension.

- For Classifier & Error (or Regressor & Error), the first series of dots represent reading the file, then information about the classifier build progress is shown, then the test set classification progress is shown.
- For Classifier Only (or Regressor only) mode, there is no test set classification phase.
- For Estimate Error, the times and folds are shown.
- For Learning curves, each average point on the x-axis will be described on a line and each run for that average point will be represented by a dot.

Classifier and Error Mode Status Window Detail

When you select the Classifier and Error mode (or Regressor only), the Status window shows:

- The random seed used to split the data into training and test sets.
- The number of records used for generating the model.
- The number of records used for evaluating the resulting model.
- The number of correct and incorrect predictions made.
- For classifiers, the average normalized mean squared error represents the accuracy of the probability estimates. For each test record, the mean squared error is the square of one minus the probability estimate for the correct label value, plus the sum of the squares of the probability estimates for the other (incorrect) label values. The normalized mean squared error is half the mean squared error, and is a value between zero and one. The average normalized mean squared error is the normalized mean squared error averaged over all the records in the test set by their appropriate weights (weighted average).

- For classifiers, the classification error, which is the percent of incorrect predictions.
- For classifiers, both the average mean squared error and the classification error show the standard deviation of the mean and the confidence interval for the mean. This is the range you can expect from the classifier if the data comes from the same distribution. For error estimates (not losses), a more accurate formula than the usual two-standard deviation rule is used.

Estimate Error Mode Status Window Detail

When you have selected the Estimate Error mode, the Status window contains the following information for classifiers:

- The number of cross-validation folds and times.
- The random seed.
- The estimated accuracy with standard deviation.
- The 95% confidence interval for the estimated accuracy.

For regressors the following information is shown:

- The estimated mean square error and estimated mean absolute error.
- The 95% confidence interval for the above accuracy metrics.

Setting Special Options

- Attributes of type arrays are always ignored.
- Dates are considered strings. Unless there are few dates, such attributes are usually ignored because of the limit on discrete attributes. You should bin dates before running an inducer.

Internationalization

On Windows systems, setting the locale is done through the desktop Control Panel element “Regional Settings.” This section deals with setting the locale on IRIX systems.

MineSet supports international datasets. Text labels in the graphical interface appear in English, but you can view multibyte column names and data values in the language corresponding to the data encoding. MineSet automatically supports EUC encoding for Japanese, Chinese, and Korean, provided the corresponding language product is installed. For other languages and encodings, see “Extending to Other Languages and Encodings (IRIX Only)” on page 104.

Setting the Locale on UNIX Systems

The locale and fonts for the language you are using must be present on both the client and the server system, as well as any system used for remote display. To see a list of locales installed on your system, enter the following command at a shell prompt:

```
locale -a
```

To set the locale, set the environment variable LANG to the appropriate locale from the list generated by the command above. For example, to set the locale to Japanese, EUC encoding, using csh, enter the following command:

```
setenv LANG ja_JP.EUC
```

Then simply invoke MineSet from the same shell. To permanently set the locale for all applications, consult your UNIX documentation.

Extending to Other Languages and Encodings (IRIX Only)

For MineSet to run in a locale other than those included in the installation, copy the resource files to the appropriate directory and modify them. MineSet visualization tools use Open Inventor with both 2D and 3D fonts. For text to appear properly, you must have Type III (often called CID outline) fonts installed.

Resource files are included in the installation for the following locales:

- ja_JP.EUC
- ko_KR.euc

- zh_CN.ugb
- zh_TW.ucns

To run MineSet in locale *locale_name* (see “Setting the Locale on UNIX Systems” on page 104 for how to list your installed locales):

1. Install MineSet as usual.
2. Log in as root.
3. Copy the following resource files from */usr/lib/X11/app-defaults* to */usr/lib/X11/locale_name/app-defaults*:
 - *Clusterviz*
 - *Dtableviz*
 - *Eviz*
 - *Mapviz*
 - *Mineset*
 - *Scatterviz*
 - *Splatviz*
 - *Statviz*
 - *Treeviz*
4. Edit the resource files in */usr/lib/X11/locale_name/app-defaults*. You will need to know the resource names and the specifications for the fonts you want to use (see Table 1-11 for an example).
5. Set the locale to *locale_name* and invoke MineSet.

Example 1-3 Resource File Changes for Korean

The changes needed for Korean are given in Table 1-11. The fonts listed came from the lists in the following files:

- */usr/lib/X11/fonts/ps2xlf_d_map.korean*
- */usr/lib/X11/fonts/ps2xlf_d_map.korean.outline*

Table 1-11 Korean Font Resources

Files	English Resources (some lines are wrapped)	Korean Resources (some lines are wrapped)
Clusterviz, Statviz	titleFont: screen12	titleFont: screen12,-ksg-mj-medium-r-normal--14-130-75-75-c-140-ksc5601.1987-0
Clusterviz, Statviz	gradationsFont: screen11	gradationsFont: screen11,-ksg-mj-medium-r-normal--12-110-75-75-c-120-ksc5601.1987-0
Clusterviz, Statviz	balloonFont: screen11	balloonFont: screen11,-ksg-mj-medium-r-normal--12-110-75-75-c-120-ksc5601.1987-0
Clusterviz, Statviz	xFontEncoding: ISO8859-1	xFontEncoding: ksc5601.1987-0
Dtableviz, Eviviz, Mapviz, Scatterviz, Splatviz, Treeviz	myDefaultFont: Helvetica-Narrow	myDefaultFont: Helvetica-Narrow;Gungso-Regular--KSC-H
Mineset	zoom2*fontList: -*-*-medium-r-*-*-6-*-*-*-*-*-* zoom3*fontList: -*-*-medium-r-*-*-8-*-*-*-*-*-* zoom4*fontList: -*-*-medium-r-*-*-10-*-*-*-*-*-* zoom5*fontList: -*-*-medium-r-*-*-12-*-*-*-*-*-* zoom6*fontList: -*-*-medium-r-*-*-14-*-*-*-*-*-* zoom7*fontList: -*-*-medium-r-*-*-16-*-*-*-*-*-* zoom8*fontList: -*-*-medium-r-*-*-24-*-*-*-*-*-*	zoom2*fontList: -*-*-medium-r-*-*-6-*-*-*-*-*-*;-ksg-*-*-medium-*--12-*: zoom3*fontList: -*-*-medium-r-*-*-8-*-*-*-*-*-*;-ksg-*-*-medium-*--12-*: zoom4*fontList: -*-*-medium-r-*-*-10-*-*-*-*-*-*;-ksg-*-*-medium-*--14-*: zoom5*fontList: -*-*-medium-r-*-*-12-*-*-*-*-*-*;-ksg-*-*-medium-*--14-*: zoom6*fontList: -*-*-medium-r-*-*-14-*-*-*-*-*-*;-ksg-*-*-medium-*--18-*: zoom7*fontList: -*-*-medium-r-*-*-16-*-*-*-*-*-*;-ksg-*-*-medium-*--24-*: zoom8*fontList: -*-*-medium-r-*-*-24-*-*-*-*-*-*;-ksg-*-*-medium-*--24-*:

Iterative k-Means

Iterative k-means is a method used by MineSet's clustering tool. This method will automatically choose the number of clusters within a user-selected range; no specific number of clusters is needed. The algorithm chooses the number of clusters by generating several candidate clusterings and choosing the best one using a combination of the dispersion metric and a user-selectable choice point.

Laplace Correction

The Evidence Visualizer uses an induction process whereby counts (or weights) are used to calculate probabilities. Laplace correction avoids extremes of probabilities (zeroes and ones).

To find the Laplace correction function, from the Tool Manager Mining Tools click the Classify tab and choose Inducer: > Evidence. When you click the Advance Options button the resulting dialog box shows the Laplace check box under "Evidence Options."

The Laplace correction may be used in an example in which we may prefer not to assign a probability of 1 to the event "a patient tested positive for AIDS has a deadly disease." Assigning a probability close to 1 (but not 1), allows for errors or unrepresentative samples. Laplace correction biases the Evidence Visualizer's probabilities towards the average, thus avoiding the extremes of 0 and 1.

This means every chart in the Evidence Main Window has a non-zero slice for each class. The fewer the records in a bin, the more it is changed towards the average. If the Laplace correction is checked, and the factor is left empty or set to 0, an automatic Laplace correction is applied, using a heuristic that applies a factor of $1/\text{training-set-weight}$.

Learning Curve

A learning curve is a graph that shows the error of the model generated by an inducer as a function of the number of records used to create the model. Typically, the more records used to generate the mode, the lower its error.

The learning curve is created by generating the specified number of models for each of the points on the curve. Each model is generated using a random sample of the records, and its error is estimated using the rest of the records (those not used for training).

Generating a learning curve takes a significant amount of CPU time. If t_i is the time to train an inducer on training set i (where i ranges from 1 to the number of points), and there are k runs per point, the total time is

$$k * \sum_i t_i$$

Increasing the number of runs per point increases the running time proportionally, but improves the estimate of the average. The default value of the number of runs is 3.

The Scatter Visualizer's filter panel can be used to filter some of the data types shown (average points, confidence intervals, interpolated points, or actual trials). For example, you might want to remove the data points for the trials and confidence intervals and show only the averages and interpolated points.

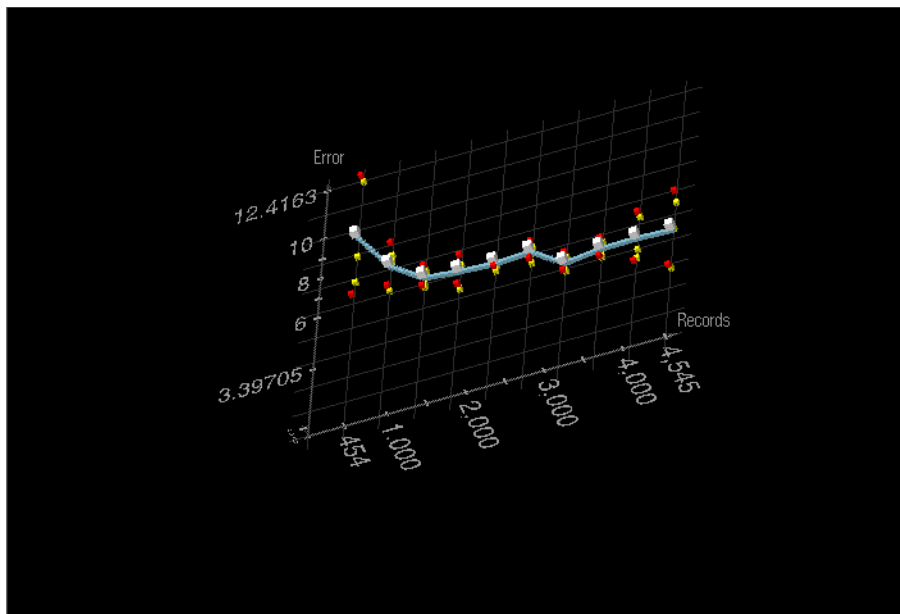


Figure 1-20 Learning Curve

Learning Curve is a mode in the Classify menu (or the Regressor menu) of the Mining Tools tab. It can be used with any of the inducers. When the Learning Curve mode is selected, the *Advanced Options* dialog box (or *Further Options* dialog box) lets you specify *Learning Curve Options* on the right of the dialog box, including:

- the number of points in the learning curve,
- the number of runs per point, and
- the number of records to use at the start and end points.

The number of records to use at each intermediate point is calculated automatically.

The number of points in the learning curve must be specified; also, it must be greater than or equal to 1. The number of records for the starting and ending points can be specified to allow generating a learning curve for a specific range of the training set. If either of these options are left blank, they are calculated automatically based on the number of points in the learning curve and the total number of records in the training set. This default covers the entire range of the training set. For instance, assume a file containing 80,000 records. If you specified 3 points in the learning curve, the algorithm generates points at 20,000, 40,000 and 60,000 records. Often it is useful to “zoom in” on a smaller range. For example, a learning curve might be generated only for a range of 1000 to 10,000 records.

Lift Curve

A lift curve graphs the difference between a random ordering of records, and the order created by the classifier when describing a particular label value. For example, you created a model to predict which customers are likely to churn, and now want to target those customers *before* they churn. The lift curve helps accomplish this goal.

A lift curve is a plot in which the X axis shows the number of records from 0 to 100% and the Y axis shows the number of records corresponding to customers who have a given label value (*Churn=yes* in our case). Two curves are shown on the graph Figure 1-21.

The lower curve (red) shows the number of customers expected to churn given a random ordering of the records. The upper curve (white) shows the percentage of customers who churn when placed in order according to the classifier's score (probability estimate) for each record. Records representing customers that the classifier identifies as most likely to churn appear first; those less likely to churn appear last. The advantage that the classifier ordering provides can be seen by the difference between the classifier curve and the random curve.

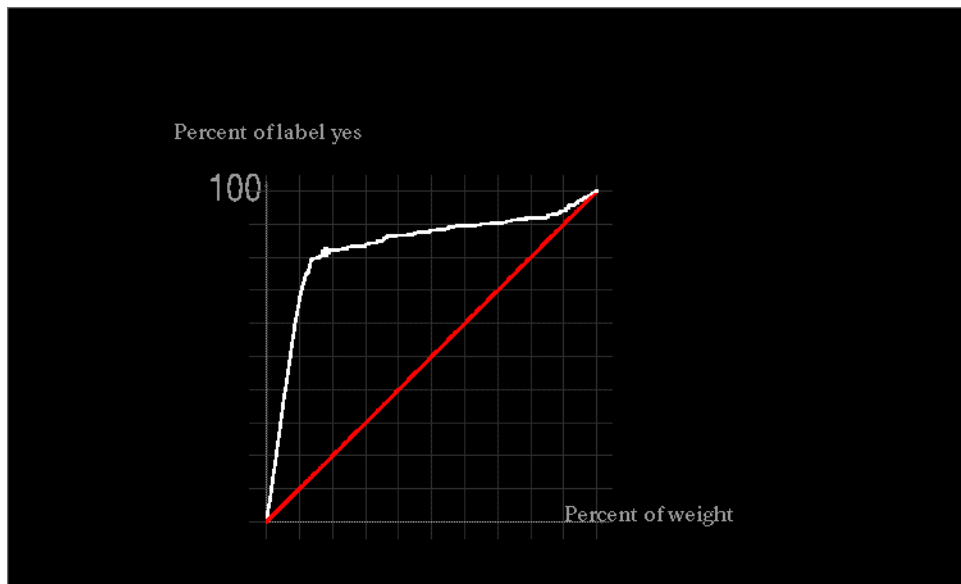


Figure 1-21 Lift Curve

In building this lift curve, a selected classifier is applied to the test set. A specified segment of the dataset is used for training, then the induced classifier run on the remainder of the dataset. The *Display Lift Curve* option is a check box under *Advanced Options* (or *Further Options*) for all inducers when using *Classifier & Error* mode. A Lift Curve requires a label value to be chosen. A lift curve is generated and displayed for that label value.

Loss Matrix

The purpose of a loss matrix is to control which types of errors the classifier will make. In many situations, some errors are costlier than others. A good example comes from the mushroom dataset, in which mushrooms are classified as edible or poisonous. Misclassifying an edible mushroom as poisonous can cost a few dollars (the price of the uneaten mushroom). Misclassifying a poisonous mushroom as edible can cost thousands (the price of a hospital stay). Classifiers use a loss matrix to avoid such costly errors.

The loss matrix is used most effectively in conjunction with a confusion matrix. The confusion matrix identifies the types and quantities of errors made by the classifier. If the

confusion matrix indicates that the classifier is making a large number of costly errors, giving costly errors a high weight in the loss matrix may be the best way to improve the classifier's performance. The Use Loss Matrix check box under Advanced Options (Further Options on IRIX) activates the loss matrix. It is available for all classification inducers.

The process is demonstrated in the following example: Figure 1-22 shows a confusion matrix for the *mushroom* dataset using the Decision Tree Inducer when only a ratio of 0.1 (10%) was used for a training set.

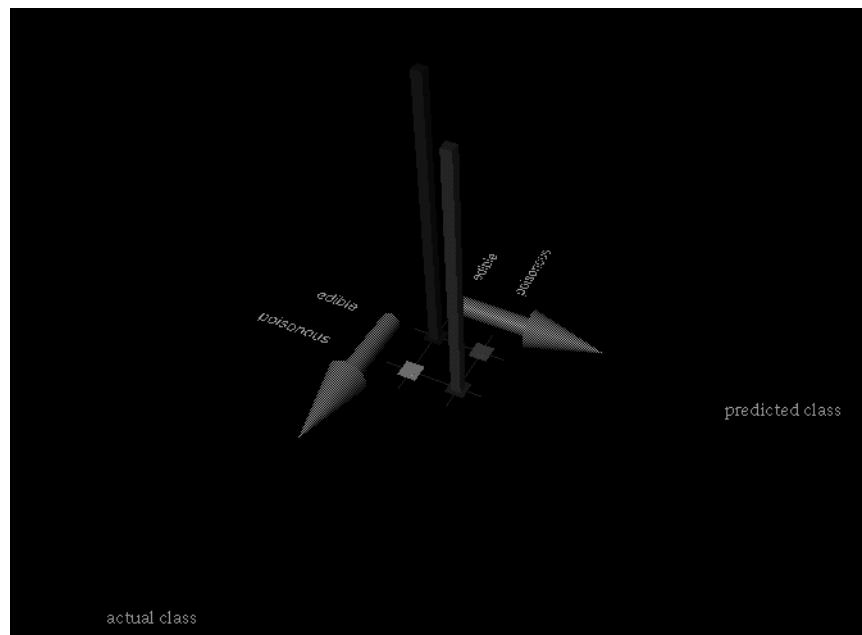


Figure 1-22 Confusion Matrix for the Mushroom Dataset Using Defaults Settings

Eight records, representing poisonous mushrooms, were classified as edible (0.1%); 15 records, representing edible mushrooms, were classified as poisonous (0.2%); 3793 edible mushrooms and 3496 poisonous mushrooms were correctly classified. While the error-rate for the classifier is only 0.31% (less than one percent), our estimated loss would be $\$10000 * 8 + \$2 * 15 = \$80,030$ or worse.

Figure 1-23 shows a confusion matrix for the same dataset, but with the Decision Tree Inducer run using a loss matrix representing the above costs. The new classifier is very

conservative and makes no mistakes in classifying a poisonous mushroom as edible; but it makes 1558 mistakes (1543+8) in classifying edible mushrooms as poisonous. The total estimated loss we would incur is thus $\$10000 * 0 + \$2 * 1558 = \$3116$, only 3% of the cost of the classifier that did not take losses into account.

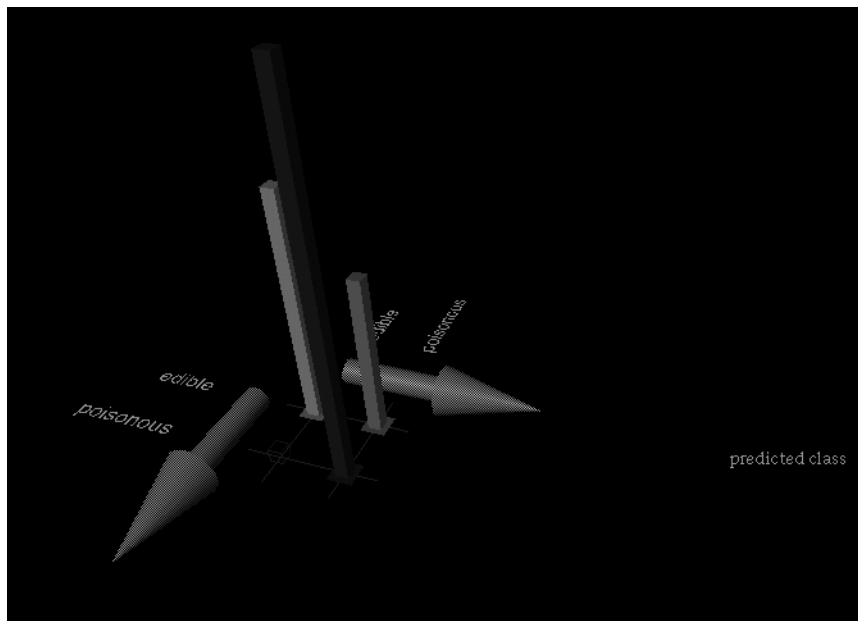


Figure 1-23 Confusion Matrix for the Mushroom Dataset With Loss Matrix

Loss matrices also allow predicting unknown (null values), which are shown as question marks (?). For example, suppose it costs us \$1 to ask an outside expert whether a mushroom is poisonous or edible. In that case, some classifications result in an unknown prediction. Running the Decision Tree Inducer yields the confusion matrix shown in Figure 1-24 where there are 1551 unknowns, and only 15 edible mushrooms are classified as poisonous. The overall cost is thus $\$10000*0 + \$1 * 1551 + \$2 * 15 = \1581 .

If a Loss Matrix has been specified in the Evidence Visualizer's *Advanced Options* (or *Further Inducer Options* on IRIX systems), a button labelled *Use Loss Matrix* appears to the lower right of the probability pie. When selected (the default) the Loss Matrix is used to adjust the probabilities shown. The largest slice shown is the class that takes into account the Loss Matrix. To see what the probabilities would be without using the Loss Matrix, deselect the *Use Loss Matrix* button. When using the Loss Matrix, a gray slice may be present because there was a column for predicting null when you edited it. If the gray slice is the largest slice, the classifier predicts null.

Map Visualizer

The Map Visualizer shows you data displayed over a three-dimensional landscape of bar chart shapes. It is useful when data has a geographical context, although it can be useful in creating any spatially-related visualization.

Data items are associated with graphical bar chart objects in the visual landscape. However, the objects have recognizable shapes and positions. The landscape can consist of a collection of these objects, each with individual heights and colors (see Figure 1-25). Besides dynamically navigating through this landscape, you can drill up and down to aggregate or see increased granularity, as well as use animation to see how the data changes across one or two independent dimensions.

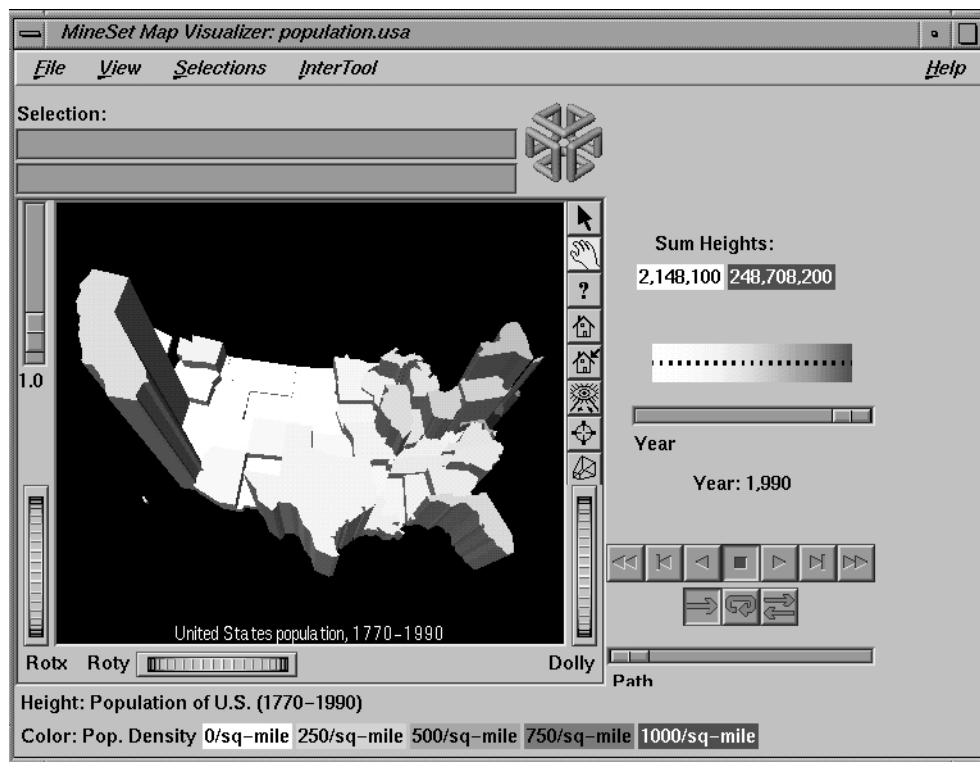


Figure 1-25 Sample Map Visualizer Screen Showing 1990 U.S. Population

The landscape can also consist of a flat plane of these geographical objects drawn as simple outlines, with “bar chart” cylinders placed at specific locations.

Another landscape possibility is lines with endpoints at specific point locations, all with individual widths and colors. Lines have width and color properties, instead of the height and color properties of the arbitrarily shaped objects and cylinders.

File Requirements for Map Visualizer

The Map Visualizer requires data, gfx, hierarchy and configuration files:

- The data file consists of rows of tab-separated fields. Typically, the Tool Manager creates this file. You can also generate this file without using the Tool Manager (for the required file format, see the *MineSet Enterprise Edition Interface Guide*, “Creating Data, Configuration, Hierarchy, and GFX Files for the Map Visualizer” chapter).

Data files are the result of extracting raw data from a source (such as an Oracle, INFORMIX, or Sybase database) and formatting it specifically for use by the Map Visualizer. Data files have user-defined extensions (the sample files provided with the Map Visualizer have a *.data* extension).

- The gfx file consists of a description of the shapes and locations of the 1-, 2-, or 3-dimensional objects to be displayed.

Gfx files must have a *.gfx* extension. MineSet includes various *.gfx* files, including the United States to the granularity of counties, telephone area codes, and postal zip codes, as well as Canada to the granularity of provinces. You can also manually generate *.gfx* files (see the *MineSet Enterprise Edition Interface Guide*, “Creating Data, Configuration, Hierarchy, and GFX Files for the Map Visualizer” chapter) for the required file format).

- The hierarchy file consists of a description of
 - the column names of the various graphical objects to be displayed
 - the filenames of the *.gfx* files that describe the locations and shapes of the graphical objects
 - an optional description of the hierarchical relationship of the graphical objects, enabling you to see more or less detail, called drill-down and drill-up functions.

A hierarchy file can define the relationship between states and regions comprising multiple states, allowing population levels to be shown at both the individual state level as well as at regional levels. The *gfx_files/usa.state.gfx* file, for example, describes the shapes of the 50 United States; the *gfx_files/usa.state.hierarchy* file describes the hierarchy grouping individual states into regions, regions into East-West areas, and the East-West areas into an aggregated United States.

For more information, see the *MineSet Enterprise Edition Interface Guide*, “Creating Data, Configuration, Hierarchy, and GFX Files for the Map Visualizer” chapter.

- The configuration file describes the format of the input data and how these are to be displayed. Typically, this file is created using the Tool Manager. You also can use your favorite text editor (such as WordPad, jot, vi, or Emacs) to produce this file without using the Tool Manager (see the *MineSet Enterprise Edition Interface Guide*, “Creating Data, Configuration, Hierarchy, and GFX Files for the Map Visualizer” chapter).

Configuration files should have a *.mapviz* extension in order to be listed when selecting the Open option from the File pulldown menu. When starting the Map Visualizer, or when opening a file, specify the configuration file, not the data file.

Starting the Map Visualizer

There are several ways to start the Map Visualizer:

- Use the Tool Manager to configure and start the Map Visualizer. See the *MineSet Enterprise Edition User's Guide for Windows* first for details on most of the Tool Manager's functionality, which is common to all MineSet tools.
- If you know what configuration file you want to use, double-click the icon for that configuration file. This starts the Map Visualizer and automatically loads the configuration file you specified. This only works if the configuration filename ends in *.mapviz* (which is always the case for configuration files created for the Map Visualizer using the Tool Manager).
- Start the Map Visualizer from the UNIX command-line by entering:

```
mapviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don't specify a configuration file, you must use File > Open to specify one.

Options for Invoking the Map Visualizer

On UNIX systems you can use the **-quiet** option to eliminate the dialogs that popup to indicate progress. You can enable this option permanently by adding the line

```
*minesetQuiet:TRUE
```

to your *.Xdefaults* file.

You can also set a warnexecute statement, see the “Warning Options” on page 199 entry. Windows users may set these options from the File > Preferences menu.

Configuring the Map Visualizer Using the Tool Manager

This section describes how the Map Visualizer can be configured using the Tool Manager. Although the Tool Manager greatly simplifies the task of configuring the Map Visualizer, you can construct a configuration file manually for this tool using a text editor (see the “Creating Data, Configuration, Hierarchy and GFX Files for the Map Visualizer” chapter in the *MineSet Enterprise Edition Interface Guide*).

Generating .gfx and .hierarchy Files

To use the Map Visualizer, you must provide the application with two files that define the graphical objects to be displayed:

- One or more *.gfx* files, which define the shapes of the graphical objects displayed.
- A *.hierarchy* file, which describes the relationship of multiple, interrelated map (*.gfx*) files.

These files are not created by the Tool Manager; they must already exist as part of MineSet or they must be created by the user. Windows users find them in the directory in which MineSet is installed under *\mapviz.gfx_files*. UNIX users find them in the */usr/lib/MineSet/mapviz/gfx_files* directory. If you decide to create your own, instructions can be found in “Creating Data, Configuration, Hierarchy and GFX Files for the Map Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

The *.gfx* and *.hierarchy* files that are part of the MineSet package include:

- the individual states of the United States
- the areas covered by the individual counties of the United States
- the areas covered by the individual five-digit ZIP codes of the United States
- the areas covered by the telephone area codes of the United States
- the individual provinces and territories of Canada
- the individual states of Mexico
- the individual states and territories of Australia
- the individual countries of Western and Central Europe
- regional subdivisions of both France and The Netherlands

The Map Visualizer requires a data file with

- One column indicating geographical objects (for example, states). Each row in this column must indicate a unique geographical object (staying with the example, this means one row for each state).
- At least one column with numeric values mapped (using arithmetic expressions) to the heights and/or colors of each geographic bar. These columns can be scalar, a 1D array, or a 2D array. If the column is an array, a slider must be used to select specific data points for this mapping to heights and colors.

If both heights and colors are mapped to 1D or 2D arrays, the arrays must have the same indexes (see “Creating Data, Configuration, Hierarchy and GFX Files for the Map Visualizer” in the *MineSet Enterprise Edition Interface Guide*).

Creating Sliders and Animation

See “Animation” on page 9.

Map Visualizer Options

Clicking the *Tool Options* button displays a dialog box that lets you change some of the Map Visualizer options from their default values.

The following sections describe the buttons and fields of the Map Visualizer’s Options dialog box.

Geography

The *Entities File* specifies a *.hierarchy* file to be used for the representation of the geographical “entity” objects, in the Map Visualizer's main window.

The *Outlines File* specifies outline objects to draw, which appear as a flat plane on which the 3-D entity objects are placed.

The *Find File* button lets you browse your files to find the *.hierarchy* file to be used.

The *Entities File* and *Outlines File* fields are optional. If the Entities File is not supplied, then the Map Visualizer creates graphical entity objects consisting of simple rectangles that are arbitrarily sized and placed in the scene.

Height

This section specifies an initial height *Scale* value (default is 1.0) and whether to display a height legend at the bottom of the Map Visualizer window.

Color

To use these Color options, you must have mapped a column to the **Color - Bars* requirement of the Data Destination panel. See “Color Selection” on page 47 for a more detailed explanation of how to choose and change colors.

Color List—You can specify the color list using the + button next to the color list label. This brings up a color editor that lets you specify a color to be added to the list.

Mapping—You can specify whether the color change that is shown in the graphic display is *Continuous* or *Discrete*. If you choose *Continuous*, the color values shift gradually between the colors entered in the “Color List” field as a function of the values that are mapped to those colors in the “Mapping” field.

The field to the right of the popup button lets you enter specific values to which the colors are mapped. You must have the same number of values in this field as there are colors entered in the “Color list to use” field.

You can enter as many colors into this field as necessary for your display. If the number of values in the column that maps to **Color - Bars* exceeds the number of distinct colors you have chosen, the Map Visualizer adds an appropriate number of randomly chosen colors at runtime. For more on choosing colors, see “Color Selection” on page 47.

Legend On—lets you determine whether a color legend is displayed or hidden.

Normalize On—lets you determine whether the Map Visualizer automatically scales the colors between the color column's minimum and maximum values (this is called color normalization), as opposed to you manually specifying threshold values. When *Normalize On* is enabled, the threshold values must lie within the range 0 to 100, representing a percentage of the color column's minimum to maximum numeric range.

Sliders

See “Slider Creation for Mapviz, Scatterviz, Splatviz” on page 154.

Message Field

This lets you specify the message displayed when an entity is selected. For a listing and description of format types that can be entered in this field, see the “Message Statement” section in “Creating Data Configuration, Hierarchy, and .gfx Files for the Map Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

This lets you specify a string that appears at the bottom of the Map Visualizer main window. This string must be enclosed in double-quotes.

Execute Field

This option lets you type in a command that is executed when double-clicking on an entity. The format is similar to the message statement. If no execute statement appears, double-clicking has no effect.

For a detailed description of the Execute field, see the “Creating Data, Configuration, Hierarchy and GFX Files for the Map Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

Resetting the Tool Options

If, after making changes to the Tool Options dialog box, you want to reset the values of all options to their default values, click the *Reset* button.

Accepting the Tool Options

Once you have finished making changes to the Tool Options dialog box, click *OK* to return the Tool Manager’s main screen.

Map Visualizer File Settings

The Tool Manager stores information for the Map Visualizer in several files, all sharing the same prefix:

- `<prefix>.mapviz.data` contains data.
- `<prefix>.mapviz.schema` describes the data file.
- `<prefix>.mapviz` contains information needed by the Map Visualizer.
- `<prefix>.mineset` contains all the information needed to create the other files.

To specify a prefix, use the *Save ...* menu option in the File menu of the Tool Manager's main window. If you do not specify a prefix, it is based on the data source.

When you use the *Invoke Tool* button, the `.data`, `.schema`, and `.mapviz` files are updated, if necessary.

Mining Tools Tab

From the Data Destination pane of Tool Manager, clicking the Mining Tools tab exposes the following tabs:

- Assoc.—Association Rules
- Cluster—Clustering
- Classify—with the modes: Classify, Classifier & Error, Estimate Error, and Learning Curve. Inducers available from this tab are: Decision Tree, Option Tree, Evidence, and Decision Table
- Regress—Regression with the modes: Regressor & Error, Regressor Only, Estimate Error, and Learning Curve
- Col.Imp.—Column Importance
- Any plug-ins such as ACpro

Multiple Selection

In most of the tools, selection is done using Shift-left mouse click. Clicking the left mouse button on an object without pressing Shift, selects the object under the cursor while deselecting all other previously selected objects. Holding down Shift while clicking the left mouse button toggles the selection of that object without affecting any other selections. (The Splat Visualizer has a different interface, described in the “Splat Visualizer” entry.)

When you select an item, a message describing that item appears in the tool's main window; by default, the visual tools only show information on the last object selected. A separate Record Viewer window displays a table, which shows the values for all selections. In the Tree and Map Visualizers, choose the Selection > Show Values to see this.

If a message has been set for the particular tool, that message also appears in the table. Columns in this table can be resized by dragging the separators between the columns. You also can click on a value to display the complete text of that value at the top of the table.

Mutual Info

Mutual Info is a Decision Tree splitting criterion that governs the way the classifier makes its decisions. Mutual Info is the change in purity (that is, the *entropy*) between the parent node and the weighted average of the purities of the child nodes. The weighted average is based on the number of records at each child node. See “Decision Tree” on page 70 for discussion of other splitting criteria.

Naive-Bayes

The Evidence Inducer is sometimes called Naive-Bayes or Simple Bayes. It builds a model that assumes the probabilities of each attribute value are independent given the class. For example, for the *iris* dataset that the four attributes (sepal length, sepal width, petal length, and petal width) are independent for each class of iris (iris setosa, iris versicolor, and iris virginica). While this simplistic assumption is rarely true, the model is excellent for initial explorations of data and its classification prediction performance is very good in practical applications.

Navigating in the Non-Tree Visualizers With Window Border Tools

This section consists of three tables that serve as a quick reference for the Evidence, Decision Table, Map, Scatter, and Splat Visualizer navigation controls. Table 1-12 describes the navigation buttons.

Table 1-12 Navigation Buttons in Non-Tree Visualizers











Button	Name	Action
	Pick	Changes the program to pick mode (an arrow). In pick mode, you can highlight (brush over) or select (click) elements of the chart.
	Grasp	Changes the program to grasp mode (a hand). In grasp mode, you can move the chart around in the window: <ul style="list-style-type: none"> — To rotate chart, hold down left mouse button and move mouse. — To move chart in window, hold down left and right mouse buttons (or use middle mouse button if your system is configured for a three-button mouse) and move mouse.
	Home	Returns the chart to the size and position designated as the home view. By default this is the size and position of the chart when the visualizer is first invoked. You can change the home position by using the set home icon.
	Set home	Sets a new home view for the chart. Use this when you want to save a certain view or position.
	View All	Moves the chart to a position where it is centered and all of it is visible in the window.
	Zoom	Moves the point you select to the middle of the pane and zooms to it. When the mouse cursor becomes a targeting sight, move it to the spot you want to see more clearly, then press and hold the left and right mouse buttons (or use the middle mouse button if your system is configured for a three-button mouse).
	3D	Toggles the 3D perspective.
	Top View	Changes the chart to a top view (Scatter and Splat Visualizers only).
	Front View	Changes the chart to a front view (Scatter and Splat Visualizers only).
	Side View	Changes the chart to a side view (Scatter and Splat Visualizers only).

Table 1-13 describes the adjustment sliders and wheels in the non-tree visualizers.

Table 1-13 Adjustment Sliders and Wheels in Non-Tree Visualizers

Slider or Wheel	Action
Height slider (upper left)	Raises or lowers cake, pie, or bar heights to emphasize differences.
Detail slider (Evidence and Decision Table Visualizers only)	Filters out less important attributes.
% Weight Threshold slider (Evidence and Decision Table Visualizers only)	Filters out attribute values with record weights less than a specified percentage of the total weight of records in the dataset, up to 2%.
Rotx wheel	Rotates chart around X axis.
Roty wheel	Rotates chart around Y axis.
Dolly wheel	Zooms chart in and out.

Table 1-14 lists several manipulations you can to perform on the charts in the non-tree visualizers.

Table 1-14 Manipulating the Non-Tree Visualizer Scene

Action	Slider or Wheel	Mouse or Keyboard Action
Toggle between Select and Grasp mode	N/A	Press Esc key or the navigation buttons.
Move the scene	N/A	In grasp mode, click and hold the right mouse button. Move the cursor in the direction you want to move the chart.
Raise or lower cake, pie, or bar heights to emphasize differences	Height slider (upper left)	N/A
Rotate the scene around X axis	Rotx wheel	In grasp mode, click and hold the left mouse button. Move cursor in the direction you want to rotate the chart.
Rotate the scene around Y axis	Roty wheel	In grasp mode, click and hold the left mouse button. Move the cursor in the direction you want to rotate the chart.

Table 1-14 (continued) Manipulating the Non-Tree Visualizer Scene

Action	Slider or Wheel	Mouse or Keyboard Action
Zoom the scene in and out	Dolly wheel	In grasp mode, click and hold the left and right mouse buttons (or middle mouse button). Move the mouse down to zoom in and up to zoom out.
Drill down through levels of detail (Decision Table and Map Visualizers only)	N/A	Put the mouse arrow over a specific chart (or the background for all charts) and click the right mouse button.
Drill up through levels of detail (Decision Table and Map Visualizers only)	N/A	Put the mouse arrow over a specific chart (or the background for all charts) and Ctrl-click the right mouse button (or click the middle mouse button).

Navigating in the Tree Visualizers With Window Border Tools

The Tree Visualizer display is best thought of as though you are viewing the scene through a camera. To change the view, you change the position of the camera (the viewpoint). This section consists of two tables that serve as a quick reference for the Tree, Decision Tree, Option Tree, and Regression Tree Visualizer controls. Table 1-15 describes the navigation buttons.

Table 1-15 Navigation Icons in the Tree Visualizers







icon	Action
	Returns the chart to the size and position designated as the home view. By default this is the size and position of the chart when the visualizer is first invoked. You can change the home position by using the next icon.
	Sets a new home view for the chart. Use this to save a certain view or position.
	Moves the chart to a position where it is centered and all of it is visible in the window.
	Undoes the previous move (like the Back button on a Web browser).
	Redoes a move that has been undone (like the Forward button on a Web browser).
	Moves one node closer to root of tree.

Table 1-15 (continued) Navigation Icons in the Tree Visualizers






icon	Action
	Moves one node or bar to left.
	Moves one node or bar to right.
	Moves one node down the tree on the left path.
	Moves one node down the tree on the right path.
	Pops up a menu of possible paths from the current node.

Table 1-16 describes the adjustment sliders and wheels in the tree visualizers.

Table 1-16 Adjustment Sliders and Wheels in the Tree Visualizers

Slider or Wheel	Action
Height slider (upper left)	Raises or lowers bar heights to emphasize differences
H wheel	Moves viewpoint up and down
Tilt wheel	Changes the up and down tilt of the camera
Side-to-side wheel (<-->)	Moves the viewpoint from side-to-side
Dolly wheel	Moves the camera backwards and forwards

Nominal Order Menu

The Nominal Order menu on the Evidence Visualizer and Decision Table lets you control how values for nominal (named) attributes are ordered, and offers these choices:

- Alphabetical causes attributes with nominal values to be sorted from left to right (or bottom to top) alphabetically.
- Weight causes the values to be sorted from left to right, with those having the largest weight of records appearing toward the left.
- Label Probability (the default) causes the values of nominal attributes to be sorted by the size of the slices corresponding to one of the classes. If the label is a binned attribute, the highest bin is used by default. If the label is nominal, then whatever class has the largest slice in the prior probability pie is used by default. If a particular class is selected, and then sort by label probability is requested, the selected class is used for determining the ordering. In all cases, if there is a NULL value, it remains the first value.

Normalized Mutual Info

The Decision Tree tool uses Normalized Mutual Info as the default splitting criteria. Technically, this is the Mutual Info divided by the log (base 2) of the number of child nodes. Refer to “Decision Tree” on page 70 for a discussion of the Decision Tree splitting criteria.

Nulls

Some visualizers, such as Splat Visualizer and Map Visualizer use special representations when fields with unknown data values, or nulls, are mapped to visual attributes. (For a discussion of null values, see the *MineSet Enterprise Edition Interface Guide* chapter “Nulls in MineSet.”) When every record in a bin has a null value for the column mapped to color, the resulting color for that splat is gray. If one or more records in the aggregate have non-null values for the column mapped to colors, then that value is (or those values are) used to compute the color. While the sum of a value and null is null, the average of a value and null is the value (that is, $\text{value} + \text{Null} = \text{Null}$; $\text{avg}(\text{val}, \text{Null}) = \text{val}$).

When a null value is displayed in the various visualizer display window areas: Pick Window, Selection Window or “Pointer is Over” area, it is shown as a question mark (?).

For numeric columns containing nulls which are mapped to axes, there is a special null position below the range defined by the axis. This is to help show that the null value is discontinuous with the other values. The null positions for numeric axes can be turned off using the Show Null Positions option under the View Menu. For string-valued columns mapped to axes, nulls (represented by a ‘?’) are treated as just another value.

In the Map Visualizer, nulls can occur when any of the following is true:

- The database or data file contains a null.
- The Tool Manager is used to make an array based on bins and no data falls into a specific bin. For example, if there is no data for the 30-40-year-old population, that bin is null.
- The Tool Manager is used to make an array and the null enum option is specified. In this case, an extra array element is created to represent the aggregation of all the values for which the bin value is null. The Tool Manager assigns the question mark (?) character to this extra bin. To view the values of this bin, move the corresponding slider to its left-most position. If there are no data for that null bin, the values associated with it are null as well, and the Map Visualizer represents the corresponding graphical object(s) as a “null object.”
- Expressions and aggregations of nulls can generate nulls (see Nulls in MineSet).
- The Map Visualizer uses special representations when a null value is mapped to a visual attribute. A null height results in a dark grey object with zero height; a null color results in an object with appropriate height (as defined by the value mapped to height), but with a dark gray color.

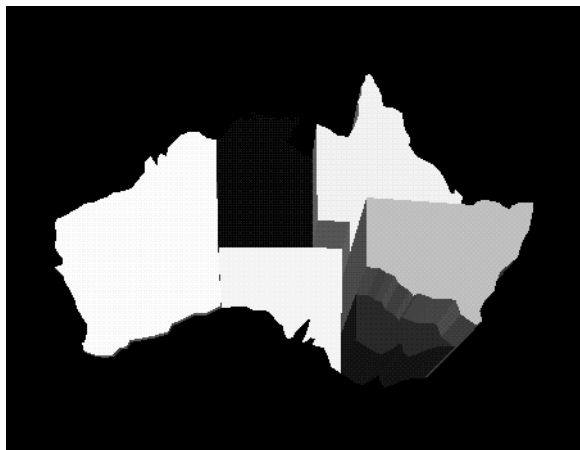


Figure 1-26 Representation of a Null Value Mapped to Height (Top Middle Object) and to Color (Bottom Right Object)

When selecting an object with a null value, a question mark (?) is shown in the selection field. Nulls mapped in Map Visualizer are shown in Figure 1-26 in which a null value has been mapped to Height and Color.

Option Tree

The Option Tree is a predictive model. It makes predictions by using the dependent, or known, attribute values to help determine the value of the label, or unknown attribute. An Option Tree starts classifying data by assigning each record to a class. The underlying structure used for classification is a Decision Tree, as described in “Decision Tree” on page 70. Option Trees extend a regular Decision Tree model with *Option Nodes*. An Option Node shows several options that can be chosen at a decision node in the tree. Option Trees tend to be much larger and more complex than ordinary decision trees.

Inducing the Option Tree

An Option Tree model is induced (generated) automatically from data. The data, which is made up of records and a label associated with each record, is called the *training set*.

File Requirements

The Option Tree inducer requires a training set. Files are generated by extracting data from a source (such as a MineSet ASCII or binary file, or a table in an Oracle, INFORMIX, or Sybase database). To apply the generated classifier, you should have a dataset of records with the attributes used by the classifier, except that the label need not be present.

Creating the Option Tree Inducer

An Option Tree classifier is induced (generated) automatically from data. You begin by logging on to the server and selecting a dataset in the normal manner.

From the Tool Manager choose the *Classify* tab, and from the *Inducer* popup menu select Option Tree. You do not need further specifications unless you wish. Simply click the Go button. The Option Tree uses the Tree Visualizer for its display.

Parallelization in IRIX

If you have installed the multiprocessor version of MineSet, in Decision Trees it is possible to compute tree-based algorithms in parallel whenever a branch contains over 1000 records. (See “Parallel Computing on IRIX Systems” on page 133.) You can control the number of threads by changing the parallelization mode in the Preferences panel of Tool Manager (see “The File Menu”). This option is only available on IRIX systems.

Option Tree Options

Selecting Advanced Options (Further Classifier Options on IRIX) causes the inducer options dialog box to appear. This dialog box consists of four panels:

- The top panel indicates the choices you made in the Tool Manager’s Data Destinations panel.
- The second pane from the top lets you set the loss matrix and the weight attribute. See “Loss Matrix” on page 110 and “Record Weighting” on page 137.

- The bottom-left panel lets you specify further Inducer Options (described below).
- The bottom-right panel lets you specify the Error Estimation Options (unless the mode you chose in the Data Destinations panel was Classifier Only, in which case this area is empty). The options shown in this panel depend on the type of Error Estimation you chose (see “Error Estimation” on page 79).

To fine-tune the Option Tree induction algorithm, you can change any of the options for Decision Trees described in “Decision Tree” on page 70. In addition, the following additional options are provided.

- Max # root options
This integer, which defaults to 5, restricts the maximum number of options that may be created at the root. The inducer might not allow the full number of options to appear because the other attributes might be inferior.
- Decrease
This integer, which defaults to 2, defines the amount by which the maximum number of options decreases at every level. With the default of 5 for *Max # root* options, it implies that there are at most three options ($5-2=3$) for the second level of decision nodes. The third level of decision nodes would be restricted to a single option ($3-2=1$). Levels further down would be similarly restricted to a single option.
- Min fitness ratio
This ratio determines when to exclude attributes as options. When the inducer gives a fitness score to each attribute, it chooses the best attribute as well as other attributes that might also be good as options. The fitness ratio determines how good those other options must be, to be chosen. A ratio value of f implies that to be considered an option, an attribute must rank at least $(1-f)*b$, where b is the score for the best attribute. A fitness ratio of 1 picks all the attributes (so the limiting options described above are reached if there are attributes on which to split). A fitness ratio of 0 causes a regular Decision Tree to be created (no option nodes). The default value is 0.9.

The time required to induce an Option Tree is closely related to the number of option nodes created. Because option nodes usually are created near the top (where they are most useful for both comprehensibility and error reduction), a good approximation for the time to induce an Option Tree is the number of options created that have no children options times the time to build a Decision Tree. Under the default setting, the root node can have up to five options, and each child can have up to three options. The total options then can be up to 15 (3 times 5). If Max # root options is increased to 6, the number of

options then is limited by 48 ($6 \cdot 4 \cdot 2$); if it is increased to 7, the number of options is then limited by 105 ($7 \cdot 5 \cdot 3$). Keeping the Max # root options to 5, but changing the decrease to 1, limits the options by 120 ($5 \cdot 4 \cdot 3 \cdot 2$). The expected induction time for the last example, thus, is two orders of magnitude longer than for a regular Decision Tree. Decreasing the Min fitness ratio option usually results in fewer options than the limiting factor, thus reducing induction time.

Parallel Computing on IRIX Systems

Parallelization is offered through the multiprocessor version of MineSet for IRIX systems. It allows compute-intensive tasks to be performed in parallel. To run the 64-bit version, install the parallel server and choose parallel version from the Tool Manager Preferences menu.

DataMove is a process that runs on the server, and provides access to databases and data stored in flat files, and transforms data for the mining and visualization tools.

Large memory (64-bit) is supported on IRIX 6.4 and later releases. If you have IRIX 6.2, you can still use the 32-bit data mining utility, but you must upgrade to IRIX 6.5 in order to obtain 64-bit support and p-threads. To get the full advantage of 64-bit addressing you may also need to change the **systune** resource parameters, depending on your system configuration.

The **systune** parameters determine the default limits on the available system resources. Table 1-17 lists the **systune** parameter values recommended on IRIX (for more details see the `systune(1M)` reference page).

Table 1-17 systune Parameters

Parameter	Definition	Recommended Value
<code>rlimit_pthread_cur</code>	Current limit on the number of threads	1024
<code>rlimit_rss_cur</code>	Current limit on memory usage	The amount of physical memory on your machine

Table 1-17 (continued) systune Parameters

Parameter	Definition	Recommended Value
rlimit_vmem_cur	Current limit on virtual memory usage	The size of the logical swap space on your machine or about twice the physical memory
rlimit_nofile_cur	Current limit on number of open files	1024 or the limit on the number of threads

Note: You must reboot your machine after setting the new parameters.

If you have installed the multiprocessor version of MineSet, it is possible to compute tree-based algorithms in parallel whenever a branch contains over 1000 records. Each node on the tree estimates the best possible split on the corresponding level, and these tasks are performed in parallel. The maximum number of threads a program can spawn is determined automatically by default. You can control the number of threads by changing the parallelization mode in the Preferences panel of Tool Manager.

Parallelization may cause memory fragmentation, causing the largest data sets that can be computed in parallel to be smaller than the largest data sets that can be computed on a single processor.

Predictability

When working with Association Rules, if you visualize a grid with right-hand side (RHS) and left-hand side (LHS) as two axes, predictability describes the fraction of items occurring in the RHS out of all items with LHS, or the prevalence divided by the frequency of occurrence of LHS items (see “Association Rules Visualization” on page 27 for an explanation of RHS and LHS). It indicates how often X and Y occur together, stated as a fraction of the number of records in which X occurs. For example, if the predictability is 50%, X and Y occur together in 50% of the records in which X occurs. Therefore, if you know that X occurs in a record, you can expect that 50% of the time Y occurs in that record. For example, if the record shows a 50% predictability in the correlation between men buying baby diapers and cigarettes, then 50% of men buying baby diapers will also buy cigarettes.

Prevalence

When working with Association Rules, if you visualize a grid with right-hand side (RHS) and left-hand side (LHS) as two axes, prevalence describes the frequency of items occurring in LHS and RHS together (see “Association Rules Visualization” on page 27 for an explanation of RHS and LHS). Prevalence quantifies how often X and Y occur together in the file as a fraction of the total number of records. For example, if the prevalence in 1%, X and Y occur together is 1% of the total number of records.

Pruning

Pruning is a Decision Tree inducer option that may be changed to affect tree height and Split Lower Bounds. The default pruning factor is 0.7. Higher numbers indicate more pruning. See “Decision Tree” on page 70 for a full discussion.

Random Seed

A random seed may be specified in several option dialog boxes in MineSet. It refers to the starting point of a random number generating algorithm. The random number generated is then used in the sampling of the dataset. By using the same random seed each time you take a sampling, for example, you can be sure of working with the same selection of data each time you refine your model. If you change the random seed, you get a different selection from the basic dataset.

Record Viewer

The Record Viewer allows you to view your data directly. This gives you the opportunity to get familiar with the columns and the data values within them. The Record Viewer also allows you to:

- Manipulate the columns by resizing, rearranging, or hiding them
- Sort or filter your data by the values in any given column
- Sort by multiple columns (Windows only)
- Renumber the rows after sorting or filtering

- Search for a given value
- Save your manipulated file in a number of formats

Starting the Record Viewer

There are several ways to start the Record Viewer:

- From the Tool Manager:
 - Click the Viz Tools tab in the Data Destinations panel.
 - Click the Records tab (Windows) or choose the Record Viewer from the Tool menu (IRIX).
- From the Tool Manager Visual Tools menu, choose the Record Viewer. Then use the Record Viewer File menu to open a file.
- Double-click on a .schema file icon (Windows only).
- UNIX users can enter this command at the shell command-line prompt:

```
recordview [ filename ]
```

Renumbering Rows

The Record Viewer allows you to renumber the rows at any point. If you do this after sorting or filtering, the renumbering cannot be undone. To go back to your original data, you must reopen the file.

To renumber, choose View > Renumber rows.

Searching in the Record Viewer

The Record Viewer allows you to search for a value in your data. To open the Search panel, choose View > Search panel. To search, type in the value, highlight the columns you want to search in, and click *Find Next* or *Find Previous*.

Saving Data

The Record Viewer allows you to save your data, including any changes to the data that you may have made. You can save your file using either File > Save or File > Save As.

If you use Save, your file is saved under the original name and format. If this is the first time you are saving the file, it is saved in MineSet binary format. Save As brings up the Save data screen, where you can enter the desired filename and the type of format you wish.

With Save As, you can save your data in four formats: binary, ASCII, HTML, or text. When you save in binary or ASCII format, both the data file and a schema file are saved. HTML format saves the file as an HTML table. Text format saves the file in tab-delimited form, with the column titles as the first row.

Record Weighting

In certain experimental designs, portions of the true population are sampled more frequently than others. For example, while you might want a 1% sample of some population, a small minority that is already 0.1% of the population results in a 0.001% sample, which might be too small (for instance, you might get two people). Record weighting lets you give each record a weight; thus, a subpopulation that was sampled twice as frequently might get a weight of 0.5, while the rest of the population is given a weight of 1.

As another example, a phone company stores all fraudulent phone calls in the dataset, while storing only a small fraction of non-fraudulent calls. By using record weighting, it is possible give each record its true portion of the population.

Finally, some datasets are already aggregated, and the records have a natural “count” associated with them (for example, statistics about cities in the U.S. usually have an associated count of the population). This count attribute can be mapped to weight, which is equivalent to replicating each record by the number of counts.

The semantics of record weighting is that a record weight of 2 is equivalent to two records with a record weight of 1. Floating point weights are allowed. See also “Weighting” on page 200.

Regress Tab

The Regress tab allows you to access MineSet's regressor. From the Data Destinations pane of Tool Manager, click the Mining Tools tab to reveal the Regress tab.

Regression Trees

A regressor is a predictive model that performs regression. Regression is the task of predicting a continuous label value, given a set of descriptive attributes. Regression and classification are similar, the difference being that in classification the predicted label can take on only a small number of discrete values.

When a regressor is generated, MineSet also generates a visualization. This visualization can help you understand the regressor and how it makes predictions. In addition, it can provide valuable insight into the data itself. Once generated, a regressor can be used to predict the label value for unlabeled records.

Inducing the Regression Tree

A Regression Tree regressor is induced (generated) from data automatically using an algorithm called an inducer. If you make any selection but Regressor Only the data is segmented to produce a training set. The training set consists of records in the database for which the continuous label is known. For example, you could supply a database table with columns (such as age, education, occupation, hours worked per week, and so forth), and one column containing descriptive attributes (gross income). You begin by logging on to the server and selecting a dataset in the normal manner.

Windows users:

From the Tool Manager, under Mining Tools choose the *Regress* tab.

IRIX users:

From the Tool Manager choose the *Regress* tab, and the *Inducer* menu automatically shows Regression Tree.

You do not need further specifications unless you wish. The choices are exactly the same as the Decision Tree except the Label value is Continuous instead of Discrete. Simply click the *Invoke Tool* or *Go* button. The Regression Tree uses the Tree Visualizer for its display.

Continuous Label

The Continuous Label menu provides a list of possible continuous labels. This list includes all attributes that take on numeric values. Select the label attribute you wish to model. For instance, to generate a regressor for predicting gross income, select “gross income”. If there are no continuous attributes, the menu shows *No Continuous Label*, and the *Go* button is disabled. Regressors can only be generated for continuous attributes. If the dataset does not contain a continuous attribute you may add a new continuous column using the Tool Manager’s Data Transformations panel.

Regression Tree Options

Selecting *Advanced Options (Further Inducer Options)* on IRIX displays the Regressor Options dialog box to appear. This dialog box consists of four panels:

- The top panel indicates the choices you made in the Tool Manager’s Data Destinations panel.
- The second panel from the top lets specify a Loss Matrix, see “Loss Matrix” on page 110, and set the weight attribute, see “Weighting” on page 200.
- The bottom-left panel lets you specify other Inducer Options (described below).
- The bottom-right panel lets you specify the Error Estimation Options (unless the mode you chose in the Data Destinations panel was Classifier Only, in which case this area is empty). The options shown in this panel depend on the type of Error Estimation you chose (see also “Error Estimation” on page 79).

To fine-tune the Regression Tree induction algorithm, you can change the following Regression Tree Inducer options.

- Limit tree height by

By default, there is no limit to the height (number of levels) in the Regression Tree. You can limit the height by clicking the check box and typing a number for the limit. Limiting the number of levels speeds up the induction and is useful for studying the Regression Tree without the distraction of too many nodes. Restricting the size decreases the run time, but may increase the error rate. Setting this option does not affect the attributes chosen at levels before the maximum level.

- Splitting criterion

This option allows you to specify which criterion will be used to select among competing attribute splits during tree induction. For regression trees, MineSet supports four splitting criteria:

- Variance

This causes the Regression Tree Inducer to choose splits that minimize the within-node variance at each point in the tree. When generating a leaf, the prediction at the leaf is the mean of the label values of the records reaching that leaf. Variance is the squared difference between each of the values in the set, divided by the total number of elements or values. This mode is most commonly used statistically.

- Absolute Deviation

This causes the inducer to choose splits that minimize the absolute deviation within the node at each point in the tree. When generating a leaf, the prediction at the leaf is the median of the label values of the records reaching that leaf.

- Normalized Variance

This is a splitting criterion just like variance, but which is biased against making multiway splits.

- Normalized Absolute Deviation

This is a splitting criterion just like absolute deviation, but which is biased against making multiway splits.

For a given problem, it is difficult to know which criterion will be best. Try each, and select the one that leads to the lowest error estimate, or the Regression Tree you find easiest to understand.

- Split lower bound

This is a lower bound on the weight (or the number of records if weight was not set) that must be present in at least two of the node's children. The default for this option is 5. For example, if there is a three-way split in the node, at least two out of the three children must have a weight of five or more. This provides another method of limiting the size of the Regression Tree.

Increasing the split lower bound tends to increase the reliability of the probability estimates, because the number of records at each leaf is larger. It also creates smaller trees and decreases the induction time. If you expect the data to contain noise (errors or anomalies), increase the split lower bound. If your dataset is very small (< 100 records), you might want to decrease the split lower bound.

- Cost Complexity Pruning

Cost complexity pruning attempts to generate optimally sized trees by trading off the error rate of the tree (its cost) and the number of leaves in the tree (its complexity). During cost complexity pruning, the training set is partitioned into a learning set and a pruning set. The learning set is used to grow a pruning tree. This tree is pruned to generate a sequence of trees with decreasing complexity. The pruning set is then used to identify the minimum cost tree in this sequence. The size of the minimum cost tree is noted. The learning and pruning sets are recombined and used to grow a tree. This tree is then pruned to the size of the minimum cost tree.

The cost complexity pruning parameter allows you to select trees smaller than the minimum cost tree. The parameter indicates the number of standard errors more costly than the minimum cost tree that you are willing to accept. Setting the parameter to zero selects the minimum cost tree; setting the parameter to 0.5 selects the minimum size tree that had an error rate no more than 0.5 standard errors worse than the minimum cost tree. The default setting, 0, selects the minimum size tree that has an error rate no more than one standard error worse than the minimum cost tree. Higher numbers indicate more pruning. If your data is likely to contain noise (errors and anomalies), increase the number to create smaller trees. If the tree is pruned back to a single node, decrease the number to decrease the amount of pruning and show more of the trees structure.

Pruning is slower than limiting the tree height or increasing the split lower bound because a full tree is built and then pruned. Pruning, however, is done selectively, resulting in a more accurate regressor.

Error Estimation in Regressors

When evaluating a classifier the natural metric is error (the number of examples for which the classifier predicts the wrong label). When a loss matrix is supplied, different types of misclassification errors may have different associated costs. In this situation loss is a natural measure.

For regression, where the task is to predict a real value there is no single natural evaluation metric. The two measures that are frequently used are *mean squared error* and *mean absolute error*. In Mean Squared Error, the mean of the squared difference between the predicted label value and the actual label value is used. In mean absolute error, the mean of the absolute value of the difference between the predicted label value and the actual label value is used.

Regressor Name

The generated regressor is named with the prefix of the session filename (as determined in Tool Manager) and the suffix *-rt.regress*, for example *churn-rt.regress*. By default, all regressors are stored on the server in the *file_cache* directory, which defaults to *MineSet Files* on Windows and *mineset_files* on IRIX. Windows systems store the visualizations in the current working directory as a 3D bar chart icon. The files have a *-rt.treeviz* suffix.

These regressors can be used to predict the labels for unlabeled datasets. To apply a stored regressor, you need to select a dataset of records with the attributes used by the regressor. The regressor will predict a new continuous label value for each record. See “Apply Model” on page 14 and “Backfitting” on page 32.

Remove Columns

When you need to simplify your visualization or model, you can remove columns from your current work without removing them from the dataset. From the Data Transformations pane of the Tool Manager, select a column from the Current columns text pane and click the *Remove Column* button. This removes the column from your calculations without removing it from the dataset.

Return-on-Investment Curve

A Return-on-Investment (ROI) curve is similar to a Lift Curve, but displays accuracy in terms of loss rather than in terms of error; taking into account the Loss Matrix used.

The ROI curve is configurable with the Tool Manager from any inducer's Advanced Options panel, as a check box under the Error Estimation Options section. It is not available for Regressors.

The points in an ROI curve are ordered by the expected loss for each record, were they to be labeled by the chosen label value. Similarly, the height of each point in the curve indicates the cumulative profit (inverse loss), rather than the cumulative accuracy (inverse error) of all records up to this point.

The expected loss is computed by multiplying entries in the Loss Matrix, under the chosen labels column (see "Loss Matrix" on page 110) by the probabilities assigned to the corresponding classes, for the classes by the classifier. Hence, if the classifier is very sure about its prediction, the expected loss will be low, and the record will appear near the left side of the ROI curve.

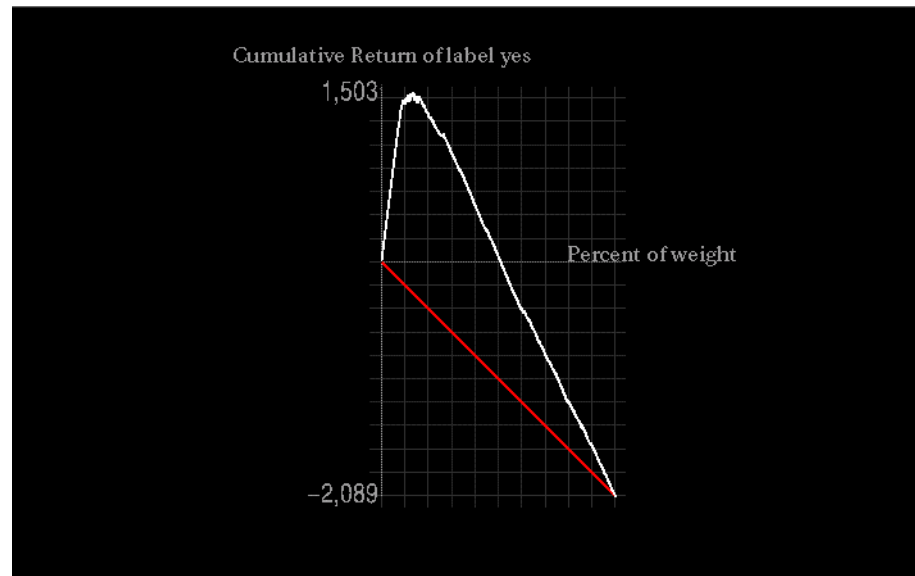


Figure 1-27 Return on Investment Curve

The idea behind the ROI curve is that the user will take an action for each individual record in the dataset. That action will be the one associated with the chosen label value. For example, in the churn dataset, the action associated with the label Yes, might be to send that person some marketing material. This might stop the person from churning; but the action is costly if done indiscriminately. The peak of the ROI curve shows approximately how much money would have been saved on the test set, if the classifier was used to predict whether or not to send the mailing to a particular person.

Special care needs to be taken when filling out a Loss Matrix for use with an ROI Curve. The column under a certain predicted label determines the resulting ROI curve for that label value. The entries in this column need to represent the expected gain or loss for taking the action associated with that label value, on all of the possible classes. For example, the entry under the column “prediction yes” in churn, under the row “actual value no”, may contain the value 2 to indicate that the cost of mailing a brochure (the action associated with “yes”) to someone who was not going to churn, is \$2. On the other hand, the entry under the column yes, row yes, may have a value of -10 to indicate that a customer was prevented from churning, saving the company \$10 over the cost of the mailing.

Sample File Directories

MineSet provides a range of sample files for users to work with. These are covered in detail in the previous section. Directories vary according to platform.

Windows users find the example files in the directory in which MineSet was installed, under *\examples*.

UNIX users find the files in */usr/lib/MineSet/examples*.

Saving Files

You can save the session you are working on using the Tool Manager’s File pulldown menu, or save particular transformations by clicking on the Data File tab in the Tool Manager’s Data Destination pane. When you return to work, load the session or file from Tool Manager’s File pulldown menu and the file and history resumes from there.

Scatter Visualizer

The Scatter Visualizer lets you visually analyze relationships among several variables, either statically or by animation. It is particularly useful for seeing individual data points when you do not have a large number of records (less than 50,000), or if you have aggregated a large number of records into a small set of distinct aggregates. If your dataset has a very large number of records consider using the Splat Visualizer. Analysis in the Scatter Visualizer is done using:

- a three-dimensional landscape
- an animation control panel that includes a two-dimensional slider
- graphical objects, called *entities*, that can be animated in the three-dimensional landscape. The position, color, and size of these entities can change during animation.

The Scatter Visualizer lets you visualize your data by mapping each record, or row, in the dataset to an entity in the three-dimensional landscape. Variables in the data can be mapped to the sizes, colors, and positions of the entities. Also, you can map one or two numeric variables to the sliders in the animation control panel. If the variables mapped to sizes, colors, or positions of the entities depend on the variables mapped to sliders, the sliders can be used to drive an animation. For example, the data might represent the sales of several companies over time. If the time variable is mapped to a slider and the sales variable is mapped to size, then the entities grow or shrink as the time slider is animated.

After you create a visualization of your data, the Scatter Visualizer lets you analyze the data in various ways. The animation control panel lets you trace animation paths in one or two dimensions. By playing back the path you created, you can watch the size, color, and motion of the entities for trends or anomalies. In the three-dimensional landscape, you can orient the display to emphasize particular dimensions or a point of view. The Scatter Visualizer lets you scale the values of variables to give them greater emphasis. Also, you can filter the display to show only those entities meeting certain criteria.

File Requirements

The Scatter Visualizer requires the following files:

- A data file, consisting of rows of tab-separated fields. This file is easily created using the Tool Manager. If you are generating this file yourself, see the *MineSet Enterprise Edition Interface Guide*, “Creating Data and Configuration Files for the Scatter Visualizer” for the required file format.

You can generate data files by extracting data from a source (such as a database) and formatting it specifically for use by the Scatter Visualizer. Data files have user-defined extensions (the sample files provided with the Scatter Visualizer have a *.data* extension).

- A configuration file, describing the format of the input data and how it is to be displayed. The Tool Manager can create this file, or you can use your favorite text editor (such as WordPad, jot, vi, Emacs) to produce this file yourself (see *MineSet Enterprise Edition Interface Guide*, “Creating Data and Configuration Files for the Scatter Visualizer”).

Configuration files must have a *.scatterviz* extension. When starting the Scatter Visualizer, or when opening a file, you must specify the configuration file, not the data file.

Starting the Scatter Visualizer

There are several ways to start the Scatter Visualizer:

- Run the Scatter Visualizer from the Tool Manager under the Viz Tools tab. See “Configuring the Scatter Visualizer” on page 147 for details about using the Tool Manager in conjunction with the Scatter Visualizer.
- Use the Tool Manager to start the 3D Visualizer from the Visual Tools menu. (See the “Tool Manager” entry in this book and the *MineSet Enterprise Edition User’s Guide for Windows* for details of the Tool Manager’s functions, which are common to all MineSet tools.)

- If you know which configuration file you want to use, double-click the icon for that configuration file. This starts the Scatter Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.scatterviz* (which is always the case for configuration files created for the Scatter Visualizer using the Tool Manager).
- Start the Scatter Visualizer from the UNIX command line by entering:

```
scatterviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don't specify a configuration file, you must use File > Open to specify one.

Configuring the Scatter Visualizer

- Using Tool Manager On Windows Systems:
In the Data Destinations pane of the Tool Manager, click on the Viz Tools tab and select Scatter. The Scatter Visualizer panel shows the various elements that you can map to columns. Select a column from the popup menu to the right of each text field. The selections available from each popup menu are limited to those column types that are appropriate.
- Using Tool Manager on IRIX Systems
In the Data Destinations pane of the Tool Manager, click on the Viz Tools tab; from the Tools popup menu, choose Scatter Visualizer. The displayed visual elements showing asterisks require mapping. Select a column from the Current Columns and map to the elements in the right pane.
- Constructing a Configuration File with a Text Editor
Although the Tool Manager greatly simplifies the task of configuring the Scatter Visualizer, you can also construct a configuration file manually for this tool using a text editor. See the *MineSet Enterprise Edition Interface Guide*, "Creating Data and Configuration Files for the Scatter Visualizer" chapter.

Slider Creation for the Scatter Visualizer

Creating sliders for the Scatter Visualizer is useful when any column in the dataset varies independently, such as when data changes over a period of time. Sliders are a necessary precursor to animation and can be created manually or automatically. See the *MineSet Enterprise Edition User's Guide for Windows* for more information.

Scatter Visualizer Tool Options

To change various options in Scatter Visualizer, return to the Tool Manager's Data Destinations pane, and select the Viz Tools tab; choose Scatter Visualizer. At this point, clicking the *Tool Options* button causes a new dialog box to be displayed. This lets you change some of the Scatter Visualizer options from their default values.

The Scatter Visualizer's Options dialog box has four basic options blocks:

- Entities
- Sliders
- Axes
- Other

Entity Options

This option block lets you specify characteristics for entities controlling the appearance of the Scatter Visualizer's graphical display:

- *Entity Legend On*—lets you determine whether the entity legend is displayed or hidden.
- *Entity Size*—lets you scale the entity to a max size, a scale size, or a default (no adjustment). You also can specify whether the legend for entity size is displayed or hidden.
- *Entity Colors*—lets you control the colors in which entities are displayed. You can:
 - specify the list of colors to use
 - specify the kind of mapping
 - map the list of colors to a list of values
 - specify whether the legend for color is displayed or hidden

- map colors to entities
- *Entity Shape*—lets you choose a visual representation for the entities: cubes, bars, spheres or diamonds.
- *Entity Label Color* lets you modify a label color by clicking on it. This causes the Color Choose dialog box to appear, which lets you implement your color changes.
- *Entity Label Size* controls the size of the entity labels. A smaller number decreases the size, a larger one increases it.

To use Colors options, you must have mapped a column to the Entity-color requirement of the Data Destination panel. See “Color Selection” on page 47 for a more detailed explanation of how to choose and change colors.

Color List lets you specify the color list using the + button next to the color list label. This brings up a color editor that lets you specify a color to be added to the list.

Color Mapping let you specify whether the color change that is shown in the graphic display is *Continuous* or *Discrete*. If you choose *Continuous*, the color values shift gradually between the colors entered in the *Color List* field as a function of the values that are mapped to those colors in the *Color Mapping* field.

The field to the right of the popup button lets you enter specific values for mapping the colors. If you do not specify any mapping values, the range of values in the color variable is used. See “Color Selection” on page 47 for more information on choosing colors.

Summary Options

The summary slider allows animation over one or two additional variables. Every position on this slider has a color corresponding to the aggregate value of the variable mapped to Summary. Summary options let you specify what color to use for the variable shown in the Summary window. You can also specify whether the summary legend, which indicates what the values are, is displayed or hidden. For more on animation, see “Animation” on page 9.

If you have an array of values, you can specify an X or Y slider. The popup buttons next to these options provide a list of available keys, and let you specify which to use as sliders.

Slider Options

The Slider options control how the slider mappings are interpreted. For details see “Slider Creation for the Scatter Visualizer” on page 148.

Axis Options

The Axis options let you specify the following, for each axis:

- A label (if you leave this box blank, the Scatter Visualizer defaults to using the column names for each axis).
- A color
- A size type for each axis (this can be *Max Size*, *Scale Size*, or *No Adjustment*).
 - *Max Size* lets you specify that an axis is scaled independently to a specified size. If one axis has a *Max Size* that is twice as large as the other, it will be twice as long, regardless of the data values. This option is most useful when comparing axes that are in different units (for example, comparing income to age). This option has no effect on non-numeric data.
 - *Scale Size* lets you specify that the axis is scaled based on its maximum value. If two axes have the same *Scale Size*, but one has a maximum that is twice the value of the other, the former will be twice as long as the latter. This option is useful for comparing axes with the same units (for example, income vs. expenses). This option does affect the size of non-numeric axes.
 - *No Adjust* is equivalent to a *Scale Size* of 1.0.
- A size value
- Whether the axis should be extended to include the value 0.

Other Options

The Other Options, at the bottom of the dialog box, include the following fields:

- *Message* lets you specify the message displayed when an entity is selected. For a listing and description of format types that can be entered in this field, see the “Message Statement” section in “Creating Data and Configuration Files for the Scatter Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

- *Execute* lets you type in a UNIX command that is executed when double-clicking on an entity. The format is similar to the message statement. If no execute statement appears, double-clicking has no effect. For a detailed description of the Execute field, see “Execute Statement” in “Creating Data and Configuration Files for the Scatter Visualizer” in the *MineSet Enterprise Edition Interface Guide*.
- *Hide Label Distance* controls the distance at which entity labels become invisible. Smaller distances might improve performance, but the labels disappear more quickly. The higher the number, the greater the distance at which labels are hidden.
- *Axis Label Size* controls the size of the axis labels. A smaller number decreases the size, a larger one increases it.
- *Grid (X, Y, Z) Size* lets you specify the spacing between grid lines for the respective axis. A smaller number decreases the size, a larger one increases it. If zero (0) is specified, then no grid lines are drawn.
- *Grid Color* lets you modify a grid color by clicking on it. This causes the Color Chooser dialog box to appear, which lets you implement your color changes.

Resetting the Tool Options

If you want to revert all options to their default values, click the *Reset* button.

Saving the New Tool Options

Once you have finished making changes to the Tool Options dialog box, click *OK* to return to the Tool Manager’s main screen.

The Animation Control Panel

The animation control panel, which appears to the right of the main window, consists of a summary window, with up to two adjacent sliders, an information field, animation buttons, and animation sliders. See “Animation” on page 9.

Null Handling in the Scatter Visualizer

The Scatter Visualizer uses special representations when fields with unknown data values, or nulls, are mapped to visual attributes. (For a discussion of null values, see the *MineSet Enterprise Edition Interface Guide* “Nulls in MineSet” chapter.) When a null value is mapped to an entity’s size, the entity is drawn as the outline of a cube. When a null value is mapped to an entity’s color, it is drawn in dark grey. When a null value is displayed in the Selection Window or “Pointer is Over” area, it is shown as a question mark (?).

If a null value is mapped to the x , y , or z position of an entity, the result depends on the Show Entities with Null Positions option under the View Menu (see “View Menu” on page 197). If the option is set, the entities with null positions on an axis are shown just below the range of the corresponding axis. If the option is not set, the entities with null positions are not shown.

Sample Configuration and Data Files

Sample data and configuration files are provided to demonstrate the Scatter Visualizer’s features and capabilities. Detailed descriptions of these files are in Appendix A, “Sample Configuration and Data Files.”

Selection Menu

The Selection menu is similar for all tools; functions not available for a particular tool do not appear on the menu. The Selection menu lets you drill through to the underlying data. To perform a drill-through, first select one or more entities or splats, then choose one of the two methods of drilling through to the underlying records.

- *Create Box Selection* creates a 3-D box selector that can be stretched and translated to select regions of the volume. While active, a table in Record Viewer format is opened showing information about all of the aggregated data that is represented by the entities within it. Closing this window removes the bounding box, but not its selections. Any entities within the selection box or selected using Shift-click are shown in the table window. To translate the selection box, click on one of the faces with the left mouse button, and drag it in the desired direction. Holding the Shift key while dragging constrains the motion to the axis to which the drag motion is closest. To change the extent of the selection box, drag one of the gray scale tabs in the desired direction. Trying to resize or translate beyond the bounds of the volume is not permitted. The gray scale tabs constantly resize to maintain constant screen size. If at any time they appear too big, you can zoom in closer, and they reduce their size relative to the box.
- *Show Values* pops up a table showing the entities in the selection.
- *Drill Through* opens the Drill Through Dialog box (Figure 1-28). This dialog box gives you the following choices for viewing the selected records:
 - Show original data in recordviewer.
 - Send to ToolManager as Filter.
 - Send to ToolManager as new column.
 - Send to ToolManager as SQL.

You can also choose one of the following

- Send the records to a new copy of ToolManager.
- Send all non-selected records to the Tool Manager (Complement drill through).

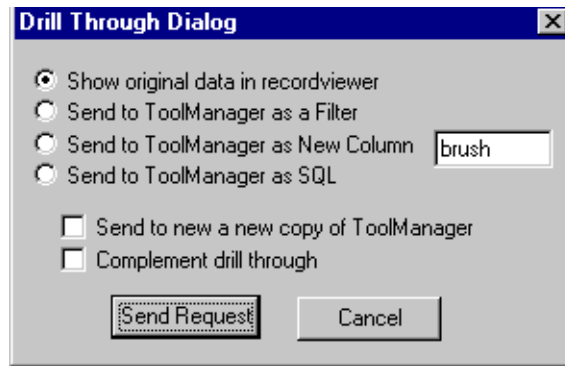


Figure 1-28 Scatter Visualizer Drill Through Dialog

- *Drill Through Columns* brings up a panel that lets you select which columns are used in drill through. Unlike other visual tools, there are no specific columns in the data that are designated as the key to the data. It is impossible for the Scatter Visualizer to determine which columns the user desires in the drill-through expression. For example, you might have cars data with brand, model, and weight. Perhaps you want to drill through to the original data, and specify that brand and model should be considered, but weight should not. By default, all columns that have been mapped to graphical requirements are considered significant on drill-through. The others are not, but may be made so by highlighting them in the Preferences dialog box.

For further details on drilling through, see “Drill Through” on page 77.

Slider Creation for Mapviz, Scatterviz, Splatviz

You can map a column to a slider to show how a value changes according to a specific criteria. Columns can be mapped to sliders if that column is numeric (int, float, double) or binned. If the column is already binned it will have `_bin` after the name. Column type is noted after the name of the column in the Current Columns field, for example `total day calls - double`.

You can create sliders for the Map, Scatter, and Splat Visualizers automatically by mapping any numeric column to either or both sliders from the *Current Columns* in the Data Transformations pane of Tool Manager. See “Animation Buttons and Sliders” on page 12.

Sorting Column Names

You can sort column names in alphabetical order for ease of reference, without changing the dataset. From the Data Transformations pane of the Tool Manager, click the *Show columns sorted* check box (Windows), or the *Sort Column Names* button (IRIX).

Splat Visualizer

The Splat Visualizer lets you visually analyze relationships among several variables, either statically or by animation. It is particularly appropriate for datasets with large numbers of records. Choose the Scatter Visualizer if you want to see individual data points and do not have a large number of records. Data analysis is done using

- a three-dimensional landscape
- an animation control panel that includes a two-dimensional slider
- graphical objects, called *splats*, which represent aggregates of datapoints. Color and opacity (but not position or size) of the splats can change during animation.

The Splat Visualizer lets you visualize your data by mapping columns to axes, sliders, color, and opacity. The resulting three-dimensional landscape can be thought of as an approximation to a scatterplot in which every datapoint is drawn separately. It is not truly a scatterplot, because datapoints that are close together (fall in the same bin) are aggregated and drawn as a single splat.

Each numeric column that is mapped to an axis or slider first must be binned. If this binning step is skipped, the Tool Manager does it using automatic uniform binning (see “Binning” on page 33). String columns can be mapped directly to axes. Any numeric column can be mapped to a color. The color of a splat is derived by averaging the value of the column mapped to color for all the data points that fall in a bin. The opacity of a splat is based on a weighting of the number of datapoints that fall in a bin. If nothing is mapped to opacity, record counts are used to determine it. The interactivity of the resulting visualization is independent of the number of data points represented; it depends only on the number of bins in the axis dimensions.

If your dataset is very large, aggregate explicitly in the Tool Manager. This causes the server to perform the processing, rather than having the entire dataset sent to the client and aggregated there. See “Aggregate” on page 3.

Up to two numeric columns can be mapped to the sliders in the animation control panel. The splats change their color and opacity during animation as the sliders in the animation panel are moved from point to point along the slider's path. Unlike the Scatter Visualizer, neither the position nor the size of the splats change; they are at fixed, uniformly spaced positions. Only their color and opacity change, which can give the illusion of actual movement.

If a string column is mapped onto an axis, binning is defined to be the distinct values of that column. The order of the values along a string axis is automatically determined by sorting the distinct values by the average aggregate value of the column mapped to color. Looking at the color changes along a string-valued axis lets you see how well that column correlates with the column mapped to color. If no color is present, then the opacity variable is used to determine the order.

Splat Visualizer Opacity

The column mapped to opacity should be record count or a column used to weight records. A splat's opacity, α , is based on this column according to the following relation:

$$\alpha = 1 - e^{-u \cdot weight}$$

where *weight* is the column mapped to opacity (or the record count if no such column was mapped to opacity). The shape of this function is such that the opacity asymptotically approaches 1 (totally opaque) as the value of *weight* becomes large. The variable *u* is what is scaled when you adjust the opacity scale slider on the left of the main window. Figure 1-29 shows the shape of this function for low and high values of *u*. Figure 1-30 shows the same visualization with low and high values of *u*.

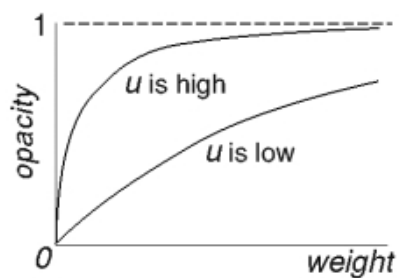


Figure 1-29 Shape of Opacity Function For Low and High Values of u

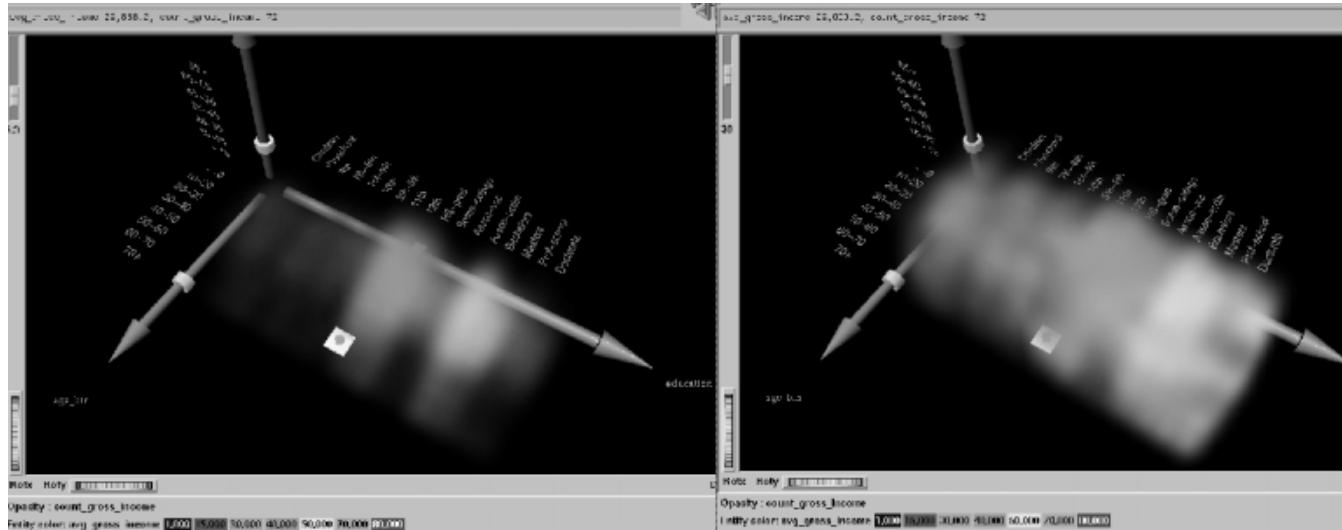


Figure 1-30 Image Where $u = 5.3$, and $u = 30$

If nothing is mapped to opacity, the Splat Visualizer generates a column of ones to produce record counts when aggregating. This means all records are weighted equally. A sum aggregation is done on this column, and an average aggregation is done on the column mapped to color while grouping by all the axis and slider columns. All other columns are unnecessary and removed. You do not need to map anything to opacity unless you want each record to be weighted by something other than 1.

See the *MineSet Enterprise Edition User's Guide for Windows* for details about processing in the Tool Manager rather than the client.

When you invoke the tool, all the processing is done on the server, and that the datafile, *adult94.splatviz.data*, contains rows that are aggregates of rows in the original data.

In some cases, you might have a column by which you want to weight the records. For example, if you have a dataset for which one column was population and another was average salary (which you want to map to color), you can map population to opacity, and average salary to color; then have the Splat Visualizer do the aggregation. This aggregation groups-by the axis and slider columns, so that it sum aggregates the opacity column (which, in this case, is population). The new column is called `sum_population`. The `average_salary` column is revised, so that it is still average salary, but weighted by each row's population. In this way, the average salary column still shows the average salary for all the people it represents.

Alternatively, if you want to avoid client-side processing and storage because of the size of your dataset, you can perform the same aggregation in Tool Manager by doing the following:

1. Create a new column, defining `temp = population * avg_income`.
2. Perform an aggregation: group-by axis and slider columns, sum aggregate population, and sum aggregate temp.
3. create a new column, defining `avg_salary = sum_temp / sum_population`
This creates the weighted average.
4. Now you can map `sum_population` to opacity, and `avg_salary` to color.

Note that these steps are the ones automatically taken by the Splat Visualizer if you do not explicitly do them in the Tool Manager. However it is more efficient if you perform them in Tool Manager so that the work is performed on the server, resulting in a considerably smaller file being retrieved to the client.

Splat Visualizer File Requirements

The Splat Visualizer requires the following files:

- A data file, consisting of rows of tab-separated fields. This file is easily created using the Tool Manager. If you are generating this file yourself, see “Creating Data and Configuration Files for the Splat Visualizer” in the *MineSet Enterprise Edition Interface Guide* for the required file format.

You can generate data files by extracting data from a source (such as a database) and formatting it specifically for use by the Splat Visualizer. Data files have user-defined extensions (the sample files provided with the Splat Visualizer have a *.data* extension).

- A configuration file, describing the format of the input data and how it is to be displayed. The Tool Manager can create this file, or you can use your favorite text editor to produce this file yourself (see the “Creating Data and Configuration Files for the Splat Visualizer” chapter in the *MineSet Enterprise Edition Interface Guide*).

Configuration files must have a *.splatviz* extension. When starting the Splat Visualizer, or when opening a file, you must specify the configuration file, not the data file.

Starting the Splat Visualizer

There are several ways to start the Splat Visualizer:

- Use the Tool Manager to configure and start the 3D Visualizer. This is detailed in the *MineSet Enterprise Edition User’s Guide for Windows*.
- From the Visual Tools pulldown menu of the Tool Manger, select Splat Visualizer. Open a configuration file by choosing File > Open.
- If you know what configuration file you want to use, double-click the icon for that configuration file. This starts the 3D Visualizer and automatically loads the configuration file you specified. This works only if the configuration filename ends in *.splatviz* (which is always the case for configuration files created for the Splat Visualizer via the Tool Manager).
- At the UNIX command-line prompt, type:

```
splatviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don’t specify a configuration file, you must use File > Open to specify one.

UNIX Options for Invoking the Splat Visualizer

The `-quiet` option eliminates the dialogs that pop up to indicate progress. You can enable this option permanently by adding the line:

```
*minesetQuiet:TRUE
```

to your `.Xdefaults` file.

Windows users achieve the same effect through the File > Preferences menu.

Splat Visualizer Shape Options

The Splat option lets you specify a number of characteristics for the Splats that the Splat Visualizer then graphically displays.

- *Splat Colors*—lets you control the colors used for the splats. You can
 - specify the list of colors to use
 - specify the kind of mapping
 - map the list of colors to a list of values
- *Splat Shape*—lets you choose one of the following methods for drawing splats: linear, gaussian, texture, sphere, cube, or diamond. See “Shape Menu” on page 165 for a further explanation of each of these.

To use these Colors options, you must have mapped a column to the color requirement of the Data Destination panel. If nothing is entered in the color list, the default colormap is used. The default colormap is a continuous spectrum from blue (lowest value) to red (highest value). See “Color Selection” on page 47 for a more detailed explanation of how to choose and change colors.

Color list—You can specify the color list using the + button next to the color list label. This brings up a color editor that lets you specify a color to be added to the list.

Color mapping—You can specify whether the color change that is shown in the graphic display is *Continuous* or *Discrete*. If you choose *Continuous*, the color values shift gradually between the colors entered in the *Color list* field as a function of the values that are mapped to those colors in the *Color Mapping* field.

The field to the right of the popup button lets you enter specific values for mapping the colors. If you do not specify any mapping values, the range of values in the color column is used. For more information about choosing colors see “Color Selection” on page 47.

Summary Options

Summary options let you specify what color to use for the Summary window. This is only applicable if you have mapped a column to the summary.

Other Options

The Other Options, at the bottom of the dialog box, include the following fields:

- *Hide Label Distance* — controls the distance at which axis labels (for string valued axes) become invisible. Increase this number to make the labels appear at further distances. The higher the number, the greater the distance at which labels are hidden.
- *Axis Label Size* — this controls the size of the axis labels. A smaller number decreases the size, a larger one increases it.
- *Grid Color* — lets you modify a grid color by clicking on it. This causes the Color Chooser dialog box to appear, which lets you implement your color changes.
- *Grid (X, Y, Z) Size* — lets you specify the spacing between grid lines for the respective axis. A smaller number decreases the size, a larger one increases it. If the size is set to 0, there are no grid lines in that dimension.

Resetting the Tool Options

Clicking the *Reset Options* button resets the values of all options to their default values.

Saving the Splat Visualizer Settings

When you press *Invoke Tool*, The Tool Manager stores information for the Splat Visualizer in several files, all sharing the same prefix:

- `<prefix>.splatviz.data` contains data.
- `<prefix>.splatviz.schema` describes the data file.
- `<prefix>.splatviz` contains information required by the Splat Visualizer.

To save the entire session along with the current tool options, use one of these menu options from the Tool Manager File menu:

- *Save Current Session...* where the default prefix is based on the data source
- *Save Current Session As...* to specify your own prefix

The saved file is *<prefix>.mineset*, and contains all the information needed to return MineSet to its current state.

When you use *Invoke Tool*, the *.data*, *.schema*, and *.splatviz* files are updated, if necessary.

Null Handling in the Splat Visualizer

The Splat Visualizer uses special representations when fields with unknown data values, or nulls, are mapped to visual attributes. (For a discussion of null values, see the *MineSet Enterprise Edition Interface Guide* “Nulls in MineSet.”) When every record in a bin has a null value for the column mapped to color, the resulting color for that splat is gray. If one or more records in the aggregate have non-null values for the column mapped to colors, then that value is (or those values are) used to compute the color. While the sum of a value and null is null, the average of a value and null is the value (that is, $\text{value} + \text{Null} = \text{Null}$; $\text{avg}(\text{val}, \text{Null}) = \text{val}$).

When a null value is displayed textually, it is shown as a question mark (?). (The Selection Window and “Pointer is Over” areas are discussed in their own sections.)

For numeric columns containing nulls which are mapped to axes, there is a special null position below the range defined by the axis. This is to help show that the null value is discontinuous with the other values. The null positions for numeric axes can be turned off using the Show Null Positions option under the View Menu. For string-valued columns mapped to axes, nulls (represented by a ‘?’) are treated as just another value.

Slider Creation for the Splat Visualizer

The number of sliders appearing adjacent to the summary window next to the main window is dependent upon the slider mappings specified in the configuration file. Any sliders appear, together with their identifying labels, according to whether the dataset has two, one or slider dimensions mapped.

Columns mapped to Slider1 and Slider2 eventually form the indices for the sliders. These columns must be either numeric (int, float, double) or binned. If a column mapped to a slider is already binned, no automatic binning is needed for this column, and this column is used as an index for a slider. However, if the column is not binned, a binned column is created using automatic uniform binning. (See “Binning” on page 33 for more information.) The column used in forming the automatic bins is deleted from the current table.

Animation Control Panel

The animation control panel, which appears to the right of the main window, consists of a summary window, with up to two adjacent sliders, an information field, animation buttons, and animation sliders. See “Animation” on page 9.

At each distinct slider position (indicated by a black dot in the summary window) the scene corresponds to a table of data in memory. For interpolation on a one dimensional slider, two adjacent tables are merged, then aggregated using the spatial columns as unique keys. The weight value (later mapped to opacity) of each splat is interpolated (0 weight is assumed if one of the tables lacks a particular row) as the slider moves from one binned position to another. The average value used for splat color is also interpolated, but weighted by the weight.

Example 1-4 Interpolation Process

This example describes technical details of the interpolation process. Suppose we want to show an image that represents an interpolation between the tables for the 40-50 year-olds and the 50-60 year-olds on the external slider. Let Table 1-18 and Table 1-19 be the tables for age=40-50 and age=50-60, respectively, for the two slider positions.

Table 1-18 Ages 40 to 50

education	occupation	hours_worked	income	weight
HS-grad	Exec-Man.	15-25	25000	2
HS-grad	Mach-op	15-25	30000	1
Masters	Technician	25-35	35000	3

Table 1-19 Ages 50 to 60

education	occupation	hours_worked	income	weight
HS-grad	Exec-Man.	15-25	70000	1
Vocational	Mach-op	35-45	40000	2

This is how the Splat Visualizer performs the interpolation. For Table 1-18, a new weight column equal to $(1-t)weight$ and a new weighted value column equal to $(1-t)(weight)(value)$ are added. For Table 2, a new weight column equal to $t(weight)$, and a new weighted value column equal to $t(weight)(value)$ are added. The two tables are merged together.

The merged table is aggregated using the spatial axes columns as keys, and sum aggregating the two new columns. This ensures that no two rows have the same binned values for all the spatial axes. Finally, divide the summed value by the summed weight to get the interpolated values. In this case, the interpolated values are for *income*. If $t=.5$, the resulting table would be Table 1-20.

Table 1-20 Interpolation Midway Between Table 1 and Table 2

education	occupation	hours_worked	income	weight
HS-grad	Exec-Man.	15-25	40000	1.5
HS-grad	Mach-op	15-25	30000	.5
Masters	Technician	25-35	35000	1.5
Vocational	Mach-op	35-45	40000	1

If the external query slider has two dimensions, bilinear interpolation is used.

This census dataset contains nearly 150,000 rows. The purpose of the external slider is to allow navigation through, and show summary info for additional dimensions in the data. The red regions represent places where the summary value is high; white shows areas where it is low. When the slider is positioned over a black point, the image shows uninterpolated data. One can trace a path on the slider and animate it using the VCR control panel below the slider.

To show how animation is produced, assume you have data for 8 years, 1990-1997 (that is, eight data points in the summary window). Begin by examining how one splat changes as the slider is moved from one year to the next. Assume that in 1990 a splat at a given position has value of 20 (to be mapped to color) and a weight of 2, meaning it represents 2 records). Assume further that in 1991 that same splat has a value of 40 and a weight of 200.

The splat in year 1991 is much more opaque than the one in 1990 because it represents an aggregation of many more records (or of much more heavily weighted records). As you move the year slider from 1990 to 1991, the weight changes by being linearly interpolated between 2 and 200.

The value is computed by taking an average of the two values weighted by records weights. For example, midway between 1990 and 1991, the weight is 101, and the value is $((1-.5)*2*20+.5*200*40)/((1-.5)*2+.5*200) = 39.8$. As you approach 1992, the size approaches 40.

You cannot stop an animation between discrete data points, and you cannot drag the Path slider to a stationary position between discrete data points. The data points in the summary window represent the slider positions corresponding to the actual data from the data file. For example, values 20 and 40 represent aggregations of actual data, but the value 39.8 does not.

Pulldown Menus in Splat Visualizer

Five pulldown menus let you access additional Splat Visualizer functions. These are labeled File, View, Selection, Shape, and Help. These are described in "File Menu" on page 91, "View Menu" on page 197, "Selection Menu" on page 153, and "Help (IRIX)" on page 96 entries.

Shape Menu

Splats are used in this tool to model clouds of small points (see Lee Westover, "Footprint Evaluation for Volume Rendering" in *Proceedings of SIGGRAPH '90*, Vol. 24, No. 4, pages 367-376).

The Shape menu lets you change the method for drawing the splats. You can choose to exchange accuracy for interactivity. Texture splats are the most accurate representation of ideal Gaussian density that is approximated in every approach. Since most computers

support hardware-assisted texturing well, the texture splat is usually the best choice. Among SGI platforms, only the Indy or earlier systems are restricted to the slower software implementation. The three splat types are:

- *Linear* draws a small set of triangles to give a linear approximation to a Gaussian splat.
- *Gaussian* draws a large set of triangles to approximate a Gaussian splat.
- *Texture* uses a texture mapped rectangle to give the most accurate representation. This can be very slow on machines that don't support hardware-assisted texture mapping.

Alternatively, the following opaque primitives are allowed.

- *Sphere* draws an opaque sphere, the radius for which varies with the cube root of the weight (or weight).
- *Cube* draws a cube the width of which varies with the cube root of the count (weight).
- *Diamond* draws a wire frame triangle whose size varies with the square root of the count (weight).

Sample Configuration and Data Files

There are sample data and configuration files provided with the MineSet product to demonstrate the Splat Visualizer's features and capabilities. A detailed description of each file is in Appendix A, "Sample Configuration and Data Files."

Split Lower Bound

Split Lower Bound is an option for refining the Decision Tree Inducer and Regression Tree Inducer. Increasing it tends to produce smaller trees, but may impair accuracy.

Split Lower Bound is a lower boundary on the weight (normally the number of records if weight was not set) that must be present in at least two of the node's children. The default for this option is 2. For example, if there is a three-way split in the node, at least two out of the three children must have a weight of two or more (two records or more if weight is not set). This provides another method of limiting the size of the Decision Tree.

Increasing the split lower bound tends to increase the reliability of the probability estimates, because the number of records at each leaf is larger. It also creates smaller trees and decreases the induction time. If you expect the data to contain noise (errors or anomalies), or if you use the tree for estimating probabilities (see "Apply Model" on page 14), increase the split lower bound to 5 or more. If your dataset is very small (< 100 records), you might want to decrease this number to 1. See "Decision Tree" on page 70.

Splitting Criterion

This option offers three splitting criteria selections in Decision Tree. The definitions below are technical. For a given problem, it is difficult to know which criteria will be best. Try them all, and select the one that leads to the lowest error estimate, or to a Decision Tree you find easiest to understand.

Mutual Info is the change in purity (that is, the *entropy*) between the parent node and the weighted average of the purities of the child nodes. The weighted average is based on the number of records at each child node.

Normalized Mutual Info (the default) is the Mutual Info divided by the log (base 2) of the number of child nodes.

Gain Ratio is the Mutual Info divided by the *entropy* of the split while ignoring the label values.

Normalized Mutual Info and *Gain Ratio* give preference to attributes with few values.

The options offered for Regressors determines how columns are selected in the regression tree.

Variance determines how columns are selected in the regression tree. Variance selects the column producing the split that minimizes the within-node variance. Choosing Variance will produce a regression tree with mean predictors at the leaves.

Absolute Deviation selects the column producing the split that minimizes the within-node absolute deviation. Choosing Absolute Deviation will produce a regression tree with median predictors at the leaves.

Normalized Variance (the default) is variance divided by the log (base 2) of the number of child nodes.

Normalized Absolute Deviation is absolute deviation divided by the log (base 2) of the number of child nodes.

Statistics Visualizer

The Statistics Visualizer, accessible from the Viz Tools tab in the Data Destinations pane of the Tool Manager, shows a window with a series of small panels, one for each column listed in the *Current Columns* pane of Tool Manager. Only a restricted number of column panels can be shown at a time, so use the side scroll bars, or stretch the Statistics Visualizer window horizontally or vertically to see more column panels.

With the Statistics Visualizer you can see certain statistics, based on the number of records in the dataset given to Tool Manager. The format of the column panel varies according to the column type, and the number of distinct values that exist for that column. Columns are generally divided into two types: *numeric* and *discrete*, shown as box plots and histograms, respectively.

How to Read Statistics Visualizer

The box plots are drawn from *numeric* columns, each numeric column is made up of either integer, float, double or date values. Each box plot panel shows statistics about data from a single column, including the minimum, maximum, mean, median, and two quartiles (25th and 75th percentiles) of these numeric values. These values are shown as lines across a vertical bar in graduated shades of green, and the standard deviation of the population is shown as a +/- value. The quartiles are shown whenever there are fewer than 50,000 distinct values, (see Figure 1-31). If there are more than 50,000 distinct values in the column, the statistics are shown as a gray vertical bar.

The mean is the number found by adding the data in a column, then dividing by the number of records. The median is the middle number when numbers in a given column are arranged in order of size. The standard deviation is a measure of the dispersion of the data in a column.

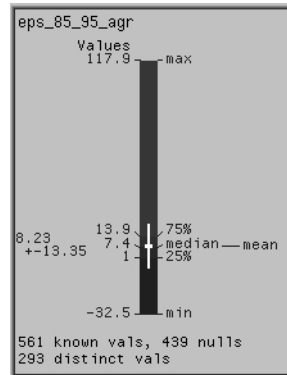


Figure 1-31 Numeric Column Displayed by Statistics Visualizer

Histograms are drawn for *discrete* (or *nominal*) columns, each of which has non-numeric (string, bin, or enum) values (see Figure 1-32). The discrete column panel shows up to 100 distinct values, as well as a histogram of the number of instances of this distinct value. The default ordering of the discrete rows is by decreasing count, but you can use the View pulldown menu to select an alternative sorting. If there are 100 or fewer distinct categories, then the column panel also contains the count of distinct values.

Histograms are used whenever discrete values are to be shown, for example, yes/no values, or state names. Each box, whether a box plot or a histogram, shows the number of records in the data set, as “total vals”, and the number of distinct records represented in this particular box, as “distinct vals.”

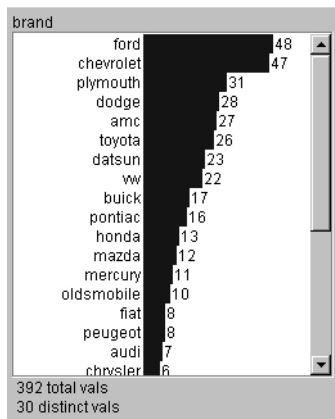


Figure 1-32 Discrete Column Displayed by Statistics Visualizer

If you started the Statistics Visualizer from the icon, the main window allows only the File and Help pulldown menus to be used. For the main window to show all menus and controls, open a *.statviz* file. Use File > Open () to see a list of configuration files.

Statistics Visualizer Pulldown Menus

Three pulldown menus let you access additional Statistics Visualizer functions. These are labeled File, View, and Help. If you start the Statistics Visualizer without specifying a configuration file, only the File and the Help menus are available. See the File and Help entries.

Statistics Visualizer's View Menu

The View pulldown menu in Statistics Visualizer sorts the histograms and box plots:

- *Sort Nominals By Count* specifies that the nominal (discrete) columns show the histogram of values that is ordered by decreasing per-value counts.
- *Sort Nominals By Name* specifies that the values be ordered alphabetically.

Table History Buttons

MineSet allows you to apply a sequence of operations to transform a data table. This sequence of transformations is recorded, and a specific transformation can be identified using the two *Table History* buttons at the bottom of the Data Transformation panel of the Tool Manager. Using these buttons you can see this sequence of steps, and go back if you made a mistake. When you click the left arrow button, the columns window shows the table as it appeared at an earlier step. Clicking the right arrow button returns the table to its current state.

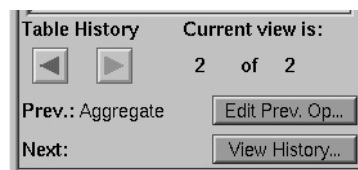


Figure 1-33 Table History Buttons “Current view is” Field

“Current view is” Field

To the right of the history buttons is the information field *Current view is*, which counts the transformations you’ve made and indicates which step you are viewing. The two integers in this field indicate which step in the transformation sequence you are looking at, out of the total number of steps that exist. For example, if you’ve made two changes, you can view the original table (1 of 3), the table after the first change (2 of 3), or the table after the second change (3 of 3).

Prev and Next Buttons

As you go back and forth using the *Table History* buttons to view earlier transformations, the *Prev* and *Next* fields (under the arrow buttons) help you keep track of where you are in the history of the table. For any table you view, the *Prev:* field tells you what the previous transformation was, and the *Next:* field tells you the next transformation.

Edit Prev. Op. Button

The *Edit Prev. Op.* button allows you to edit the operation shown in the *Prev.* field. (This button is not active when *Current view is: 1 of some number*, because that is the original table, with no previous changes.) When you click the *Edit Prev. Op.* button, the dialog box for the previous operation comes up, and you can make changes to that transformation. For example, if the previous transformation was binning columns, when you click *Edit Prev. Op.*, the *Bin Columns* dialog box appears.

By changing a previous transformation, you could affect transformations you set up subsequent to the current one. For example, if you delete a column that you used in a subsequent binning operation, that binning operation becomes invalid. The *Edit History* button can help you avoid such problems.

Delete Ops. to End Button (Windows Only)

Whenever you back up in the history, Tool Manager's Data Destination pane (on the right side of the main window) becomes disabled (greyed out) because you are no longer at the end of the history. For the Data Destination pane to become enabled again, you must either go forward to the end of the history, or click the *Delete Ops. to End* button to delete all of the operations that you applied after the current view.

History of Operations Tab

When you click the *History of Operations* Tab (*View History* button on IRIX), the panels showing the current columns and data destination are replaced by a panel showing you the complete history of the *Data Transformation* table (Figure 1-34). Each version of the table appears as a box containing a list of the columns, linked by a box (indicating the operation performed on the table) to the next version of it.

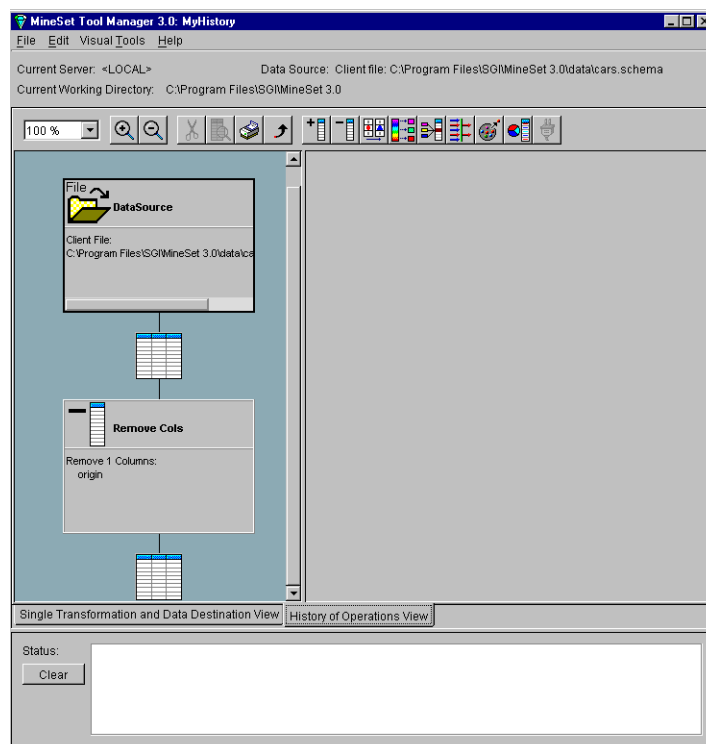


Figure 1-34 View History Dialog Box (Windows)

The icons on the toolbar of the Windows version allow you to examine and perform various operations to the dataset with the Tool Manager. Clicking on an operation in the left pane, then clicking the *Single Transformation and Data Destination View* tab (*View Single Ops/Dest* button on IRIX) it presents the Tool Manager at the selected stage of operations (see Figure 1-34 and Figure 1-35). Clicking the *History of Operations View* tab (*View History* button on IRIX) returns the previous view.

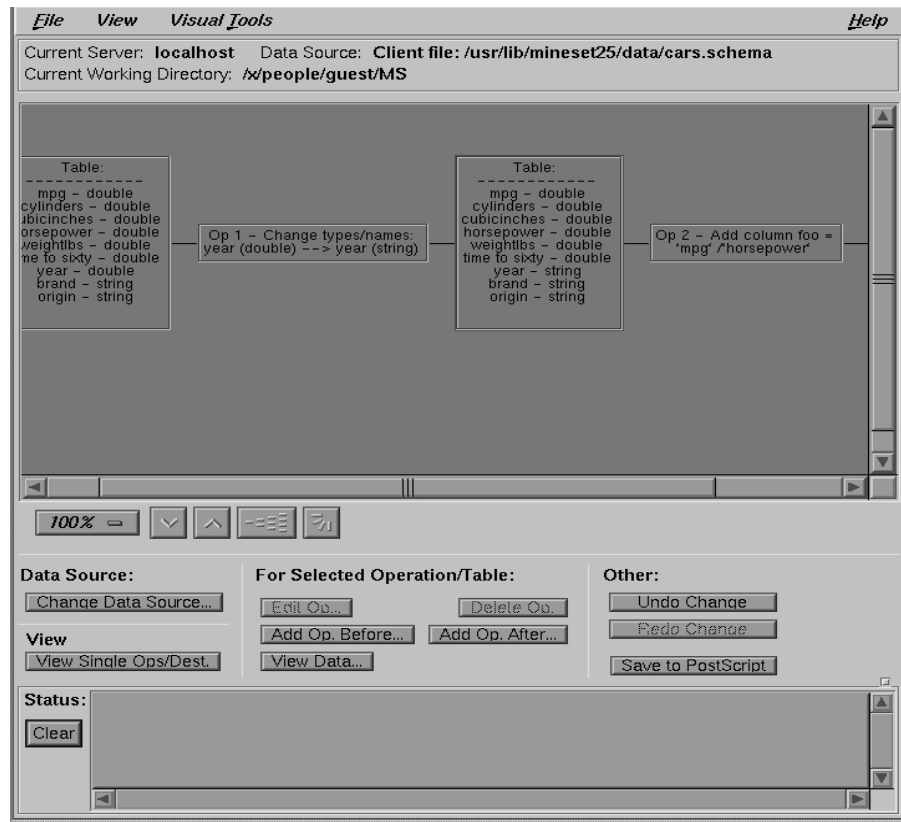


Figure 1-35 View History Dialog Box (IRIX)

As with *Edit Prev. Op*, changing one operation usually affects (sometimes invalidates) subsequent operations in the history. You can select a specific operation to edit, add, or view. The View History dialog warns you when changes affect the history, shows you the new history.

The row of buttons beneath the diagram window in the IRIX version of the View History panel allows you to change the size and orientation of the diagram.

Tool Manager

Tool Manager is the graphical user interface for specifying the configuration file, data file and tools to be used. General operation of the Tool Manager is discussed in this section. Refer to the *MineSet Enterprise Edition User's Guide for Windows* for directions on how to use it.

Tool Manager runs on your MineSet client. The process typically follows this path:

1. The Tool Manager opens a connection to the DataMover, which runs on the MineSet server. In some cases this may be the same as your client workstation, and in others is a separate machine.
2. The Tool Manager lets you specify
 - the database and table, or a binary or ASCII flat file containing the data on either the client or the server
 - which mining or visualization tools are to be applied
 - how that data is to be displayed, through tool options
 - a session file to save the history of your work

Information retrieved via the DataMover is used to guide this interaction. As a result, the Tool Manager generates a configuration file. This file contains the user-defined parameters that determine the execution of the following steps.

3. The Tool Manager transmits a copy of the configuration file from step 2 to the DataMover. The DataMover processes the file by
 - accessing the database or flat file
 - performing the specified data transformations
 - running the mining tools when requested
 - generating the visualization files when requested

These visualization files consist of your data in a specific format readable by the MineSet tool. Then a copy of these visualization files is transferred to the MineSet client.

4. The Tool Manager invokes the appropriate MineSet visualization tool.
5. The tool accesses the visualization files and displays the data.
6. If you generated a model, that model can be applied to additional data.

Tool Manager Preferences

The Tool Manager Preferences dialog box allows you to set the following options:

- *Automatically restore session on startup* allows you to return to your last session when you log into MineSet. MineSet saves the history, and opens the file in the same state that you left it.
- *Use binary data files* tells MineSet to use binary files, which tend to decrease processing time.
- *Maximum Attribute Values* lets you set the cut-off number of values for your dataset. Any columns having more unique values will not be used in the computations.
- *Parallelization* lets you set options for parallel processing on IRIX.

Training Set

A training set is a table containing attributes, one of which is designated as the class label. The label is the attribute for which you are trying to generate a predictive model.

The goal in this example is to predict the type of an iris flower (iris-setosa, iris-versicolor, or iris-virginica) given as descriptive attributes its sepal length, sepal width, petal length and petal width. Figure 1-36 shows several records from a sample training set.

	Descriptive Attributes				Label
	sepal length	sepal width	petal length	petal width	iris type
Record 1	5.1	3.5	1.4	0.2	Iris-setosa
Record 2	5.9	3	5.1	1.8	Iris-virginica
Record 3	6.5	2.8	4.6	1.5	Iris-versicolor
⋮	6.3	2.9	5.6	1.8	Iris-virginica
⋮	6.5	3	5.8	2.2	Iris-virginica

Figure 1-36 Sample Records From a Training Set

Once a model is built, it can predict the label value for new records. These new records must be in a table that has all the attributes used by the model with the same name and type as they were in the training set. The table need not contain the label attribute. If it exists, it is ignored during prediction.

Tree Visualizer

The Tree Visualizer is a graphical interface that displays data as a three-dimensional “landscape.” It presents your data as hierarchical blocks (nodes) and bars with disks through which you can dynamically navigate, viewing part, or all, of the dataset. Using the Tree Visualizer is detailed in the *MineSet Enterprise Edition User’s Guide for Windows*.

The Tree Visualizer displays quantitative and relational characteristics of your data by showing them as hierarchically connected nodes. Each node contains bars whose height, color and disk correspond to aggregations of data values. The edges (displayed as lines) connecting nodes show the relationship of one set of data to its subsets.

Values in subgroups can be summed and displayed automatically in the next higher level. The base under the bars can provide information about the aggregate value of all the bars. Bars representing negative values are shown below the top of the base. You can see negative value bars more clearly by disabling the base height (see “Tree Visualizer Display Menu” on page 192, or the *MineSet Enterprise Edition Interface Guide*, “Creating Data and Configuration Files for the Tree Visualizer”).

File Requirements

The Tree Visualizer requires the following files:

- A *data* file consisting of rows of tab-separated fields. This file is easily created using the Tool Manager as described in the *MineSet Enterprise Edition User’s Guide for Windows*. If you are generating this file yourself, see *MineSet Enterprise Edition Interface Guide*, “Creating Data and Configuration Files for the Tree Visualizer” for the required file format.

Data files have user-defined extensions (the sample files provided with the Tree Visualizer have a *.data* extension).

- A *configuration* file describing the format of the input data and how these are converted to a hierarchy. This file also is easily created using the Tools Manager, as described in the *MineSet Enterprise Edition User’s Guide for Windows*. You also can use your favorite text editor (such as Word, jot, vi, or Emacs) to produce this file see *MineSet Enterprise Edition Interface Guide*, “Creating Data and Configuration Files for the Tree Visualizer”).

Configuration files must have a *.treeviz* extension. When starting the Tree Visualizer, or when opening a file, specify the configuration file, not the data file.

Starting the Tree Visualizer

There are several ways to start the Tree Visualizer:

- Use the Tool Manager, as described in *MineSet Enterprise Edition User's Guide for Windows*.
- From the Visual Tools pulldown menu of the Tool Manger, select Tree Visualizer. Open a configuration file by choosing File > Open.
- If you know what configuration file you want to use, double-click the icon for that file. This starts the 3D Visualizer and automatically loads the file you specified. This only works if the filename ends in *.treeviz* (which is always the case for configuration files created for the Tree Visualizer via the Tool Manager).

- From the UNIX command line enter:

```
treeviz [configFile]
```

configFile is optional and specifies the name of the configuration file to use. If you don't specify a configuration file, you must use File > Open to specify one.

You can enable a warning system and suppress dialog boxes on invoking a tool, see "Warning Options" on page 199.

Tree Visualizer Options

Clicking the *Tool Options* button causes a new dialog box to be displayed (Figure 1-37 and Figure 1-38). This lets you change some of the Tree Visualizer options from their default values.

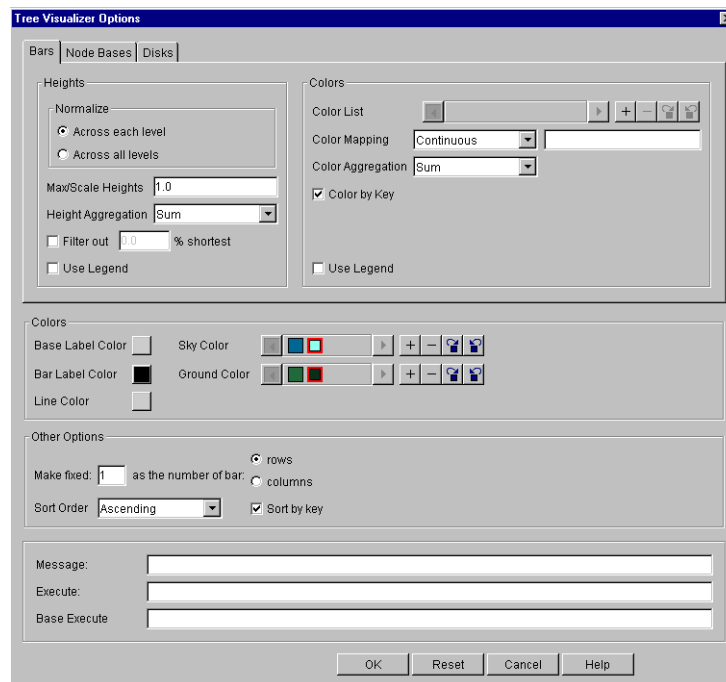


Figure 1-37 Tree Visualizer’s Configuration Options Dialog Box (Windows)

In the Windows version, the top of the dialog box shows three tabs: *Bars*, *Node Bases* and *Disks*. Each of these tabs reveals a series of options that allow you to configure the details of your visualization. To understand about choosing colors, refer to “Color Selection” on page 47. To specify the Heights portion of each dialog box, refer to “Normalize Heights” on page 180 next.

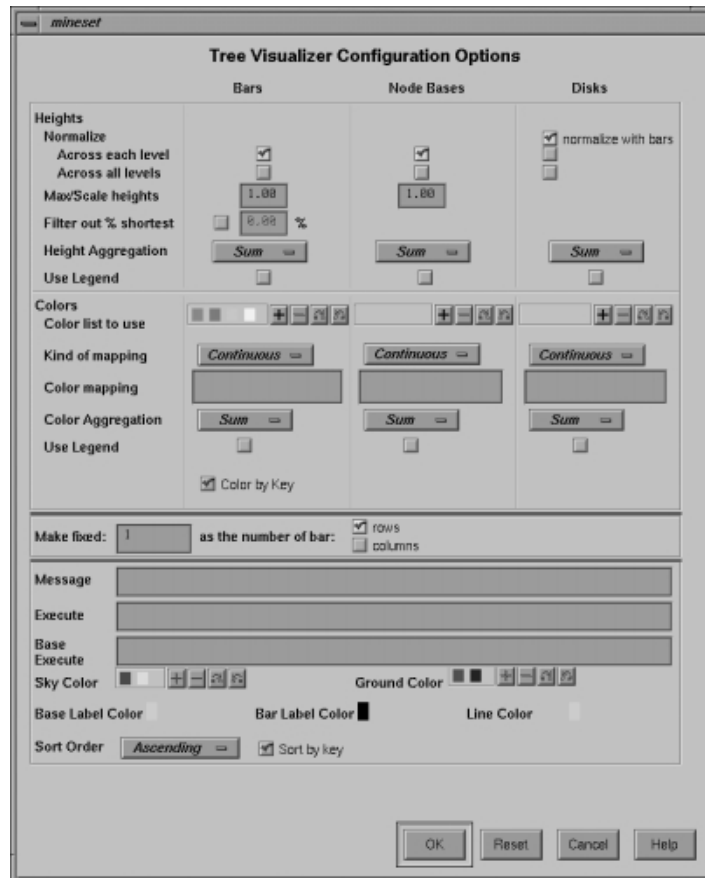


Figure 1-38 Tree Visualizer's Configuration Options Dialog Box (IRIX)

In the IRIX version, the top of the dialog box has three columns: *Bars*, *Node Bases*, and *Disks*. All the options are accessible from this one dialog box.

Normalize Heights

This option lets you normalize heights across each level of the hierarchy (or across all levels) of bars, node bases, and disks. Normalizing the heights determines the maximum value of the height variable; it normalizes all values relative to that height. Thus, if the maximum value is 30.0, and the maximum bar height was set to 1.0 (in arbitrary units), a value of 15.0 would be mapped to a value of 0.5.

Normalizing across each level independently normalizes each level of the hierarchy. This option is most useful if data has been summed up the hierarchy, and prevents the top level of the hierarchy from dwarfing items at the lowest level. Normalizing across all levels normalizes everything together, regardless of the level in the hierarchy. If neither box is checked for bars, no normalization takes place.

Node Bases are normalized independently of Bars. If no boxes are checked, the same normalization method used for bars is used for node bases, although the values are normalized independently.

If disks are present and *normalize with bars* is checked, the disks are normalized in conjunction with the bars: a disk and a bar representing the same value have the same height. If one of the other normalize boxes is checked in the Disks column, disks are normalized independently of the bars: the highest disk and the tallest bar have the same height, regardless of the actual values represented by them.

Max/Scale Heights

This option lets you specify the height of the tallest bars and node bases. The default is 1.0 (in arbitrary units). If after looking at the view, you see that the heights are too low or too high, use this field to adjust them. For example, entering 2 in the field causes all bars to be doubled in height; entering .5 makes all bars half as big.

If normalization was specified, this value represents the height of the tallest bar or base. If normalization was not specified, all values are scaled by this amount. The latter can be useful when comparing views of two different datasets.

Filter out % shortest

This option lets you filter out nodes containing only short bars. First, the tallest bar in the scene is calculated (if heights are normalized by level, then the tallest bar in each level). Then only those nodes that contain at least one bar that is the appropriate percentage of the tallest bar are shown. For example, if you enter 5% in this field, then only those nodes containing at least one bar that is at least 5% of the height of the tallest bar are shown. (Also shown are ancestors of such bars). This option is intended as a coarse way to filter out small, uninteresting nodes. It is not intended as an exact mechanism of identifying specific nodes of a certain value. Use of this option can accelerate the rendering of slow, complex scenes, or reduce clutter resulting from many bars near zero height.

Although small nodes are filtered out, they are nonetheless counted in any cumulation up the hierarchy.

Height Aggregation

By default, the height of the bars of the parent node is the sum of the height of all the bars of the children; however, these heights can be average, max, min, count, or any of the values that appear. This aggregation can be used for the values of the bar heights, base heights, and disk heights.

Colors

This set of options lets you

- specify the list of colors to use
- specify the kind of mapping
- map colors to bars, node bases, and disks

To use these Colors options, you must have mapped a column to the Color - Bar, Color - Disk, or Color - Base requirements of the Data Destination panel. See “Color Selection” on page 47 for a more detailed explanation of how to choose and change colors.

Color list to use lets you specify the color list using the + button next to the color list label. This brings up a color editor that lets you specify a color to be added to the list.

Kind of mapping lets you specify whether the color change that is shown in the graphic display is *Continuous* or *Discrete*. If you choose *Continuous*, the color values (of the bars, node bases, or disks) shift gradually between the colors entered in the Color list to use field as a function of the values that are mapped to those colors in the *Color mapping* field. If you choose *Discrete*, the colors change only at the specified boundaries.

Color mapping lets you specify values to which the colors are mapped.

Color Aggregation

By default, the values of the colors of the bars of the parent node are the sum of the values of all the bars of the children; however, these colors can be average, max, min, or any of the values that appear. This aggregation can be used for the values of the bar colors, base node colors, and disk colors.

Color by Key

This option lets you automatically color the bars by their key value. This option is ignored if another coloring was specified. If you specify no color list, or specify insufficient colors, additional colors are chosen at random. If extra colors are specified, they are ignored.

Make Fixed

By default, this option places all bars across one row. This option allows changing the number of rows or columns. If neither rows nor columns are selected, or the number is set to 0, then neither rows nor columns are fixed, and the closest approximation to a square is displayed.

Message

This option lets you type in any message you want. The message statement specifies the message displayed when the pointer is moved over an object or when an object is selected. By default, the same message is used for the base as for the bars. If no message is specified, a default message containing the names and values of all the columns is used.

The format of the message must match the type of data being used:

- Strings must use %s.
- Ints must use integer formats (like %d).
- Floats and doubles must use floating-point formats (like %f).

For a detailed description of the message field, see “Message Statements” in the chapter “Creating Data and Configuration Files for the Tree Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

Execute and Base Execute

These options let you type in a command that is executed when double-clicking on a bar or base. If only the Execute field is filled in, it applies to both bars and bases. If both are filled in, Execute applies to bars, and Base Execute applies to bases. The format is similar to the message statement. If no execute statement appears, double-clicking has no effect.

For a detailed description of the Execute field, see “Execute Statement” in the chapter “Creating Data and Configuration Files for the Tree Visualizer” in the *MineSet Enterprise Edition Interface Guide*.

Sky Color

You can specify either one or two colors. If only one color is specified, the sky is solid. If two colors are specified, the sky is shaded between the colors. When specifying two colors, the first color is for the top of the sky, the second for the bottom.

Ground Color

You can specify either one or two colors. If only one color is specified, the ground is solid. If two colors are specified, the ground is shaded between the colors. For the ground, the first color is for the far horizon, the second is for the near ground.

Base Label Color

You can specify the color of the labels on the front of the bases.

Bar Label Color

You can specify the color of the labels on the front of the bars.

Line Color

You can specify the color of the lines connecting the bases.

Sort Order

If you select the *Sort by Key* check box, the nodes in the display are in sorted order. The menu next to the check box lets you specify whether to sort in ascending or descending order.

Resetting the Tool Options

If, after you have made changes to the Tool Options dialog box, you want to reset the values of all options to their default values, click the *Reset Options* button.

Saving the New Tool Options

Once you have finished making changes to the Tool Options dialog box, click *OK* to return to the Tool Manager's main screen.

Saving Tree Visualizer Settings

The Tool Manager stores information for the Tree Visualizer in several files, all sharing the same prefix:

- `<prefix>.treeviz.data` contains data.
- `<prefix>.treeviz.schema` describes the data file.
- `<prefix>.treeviz` contains information needed by the Tree Visualizer.
- `<prefix>.mineset` contains all the information needed to create the other files.

To specify a prefix, use the *Save Current Session As ...* menu option in the File menu of the Tool Manager's main window. If you do not specify a prefix, it is based on the data source.

When you use the *Invoke Tool* button, the `.data`, `.schema`, and `.treeviz` files are updated, if necessary.

Tree Visualizer Pulldown Menus

You can access all of the Tree Visualizer's functions using the five pulldown menus labeled File, Show, Display, Go, and Help. The File menus is the same for most MineSet tools, see "File Menu" on page 91.

View Menu

The View menu (Show menu on IRIX systems) contains four options: Overview, Search Panel, Filter Panel and Marks Panel. Each of these options brings up another dialog box for interacting with the data.

The Search Panel

Select *Search* in the View (or Show) menu to bring up a dialog box that lets you specify criteria to search for objects (Figure 1-39 and Figure 1-40).

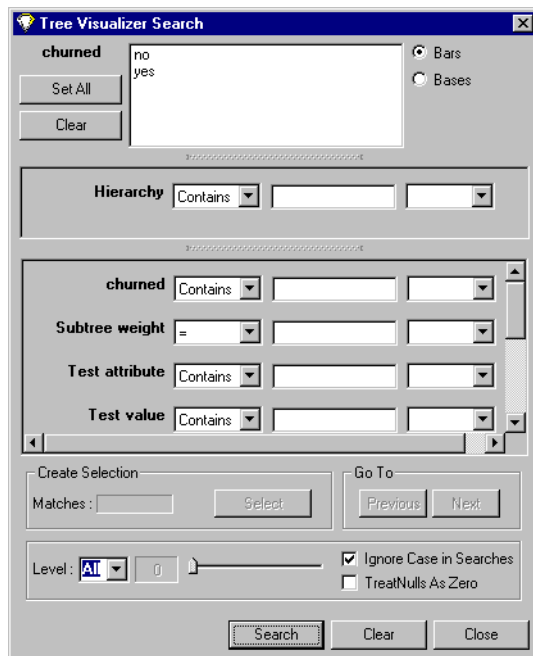


Figure 1-39 Tree Visualizer’s Search Dialog Box (Windows)

You can specify the parts of the hierarchy to be searched. By default, the whole hierarchy is searched. To limit the levels searched, select a relational operator (such as \leq) from the option menu that lets you specify the operand for the level. Then use the Level slider to select the level to be searched. Level 0 is the root of the hierarchy, level 1 is the level below that, and so forth. To search the root and the two levels below that, for example, choose ≤ 2 .

You can also choose whether to search the bars or the bases.

The Hierarchy field lets you specify nodes to search. Below the Hierarchy field are fields that let you specify search criteria for individual columns (defined in the Current Columns: window of the Tool Manager’s Table Processing pane).

To specify whether a search is case-sensitive, click the *Ignore Case In Searches* check box, in the Search panel. For example, if this toggle is on (a check mark appears on that button), the string “hello” is the same as “Hello.”

The check box labeled *Treat Nulls as Zeros* defaults to off, in which case, comparisons involving nulls cannot return TRUE in a search. If it is on, nulls are treated as equal to zero.

When searching through bars, the default is that all bars are searched. To search only a specific list of bars, you must select them. The *Set All* button turns on all bars; this is useful if most of the bars are to be searched, and only a few are to be turned off. The *Clear* button turns off all bars. If no bar is selected, the bar list is ignored, and all bars are searched.

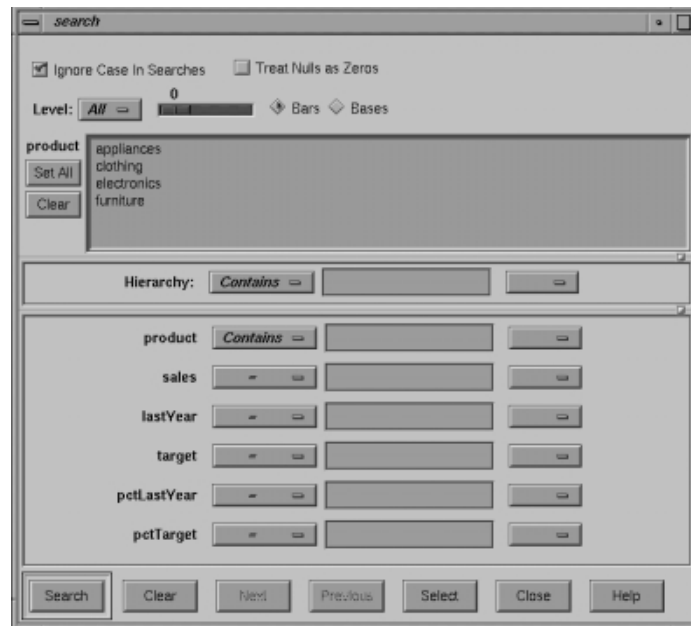


Figure 1-40 Tree Visualizer’s Search Dialog Box (IRIX)

To search for numeric values, enter the value, and select a relational operation (=, !=, >, <, >=, <=). To search for alphanumeric values, enter the string for which you want to search. You can use any of three types of string comparisons:

- “Contains” indicates that it contains the appropriate string. For example, California contains the strings Cal and forn.
- “Equals” requires the strings to match exactly.
- “Matches” allows wildcards:
 - An asterisk (*) represents any number of characters.
 - A question mark (?) represents one character.
 - Square braces ([]) enclose a list of characters to match.

For example, California matches Cal*, Cal?fornia, and Cal[a-z]fornia.

In some cases (usually associated with binning in the Tool Manager), an option menu of values appears, instead of a text field. To ignore that variable, select Ignored in the Option menu. You can use relational operators (such as >=) with these options. This means that the specified value as well as subsequent ones are selected.

In addition to numeric and string comparison operations, you can specify `Is Null`, which is true if the value is null.

To the right of each search field is an additional option menu that lets you specify “And” or “Or” options. For example, you could specify “sales > 20 And < 40.” You can have any number of And or Or clauses for a given column, but cannot mix And and Or in a single column.

If different levels of the hierarchy are keyed by different types of data (for example, the top level is selected by strings, while the second level is selected by integers), then the “Hierarchy” search field is treated as a string and provides string operations, not number operations.

If the *Ignore Case In Searches* check box is checked, the comparisons of all string searches are case-insensitive.

The buttons across the bottom of the Search panel include some or all of the following:

- *Search* causes the search to be started. This button is automatically activated if the Enter key is pressed and the panel is active.
- *Clear* turns off all search spotlights and erases the values from the search fields.
- *Next* selects and zooms to the next matched object, in left-to-right order. After the last matched object is selected, clicking *Next* returns the view to the Home position. *Next* is valid only after a search that has found matches.
- *Previous* selects and zooms in the opposite order from that of the *Next* button.
- *Select* causes all objects that matched the search criteria to be selected. The Selection menu can then interact with these objects.
- *Close* closes the search window and turns off the search spotlights. If the Search panel is reopened, it is in the same state as it was before the last *Close*; clicking *Search* again repeats the last search.

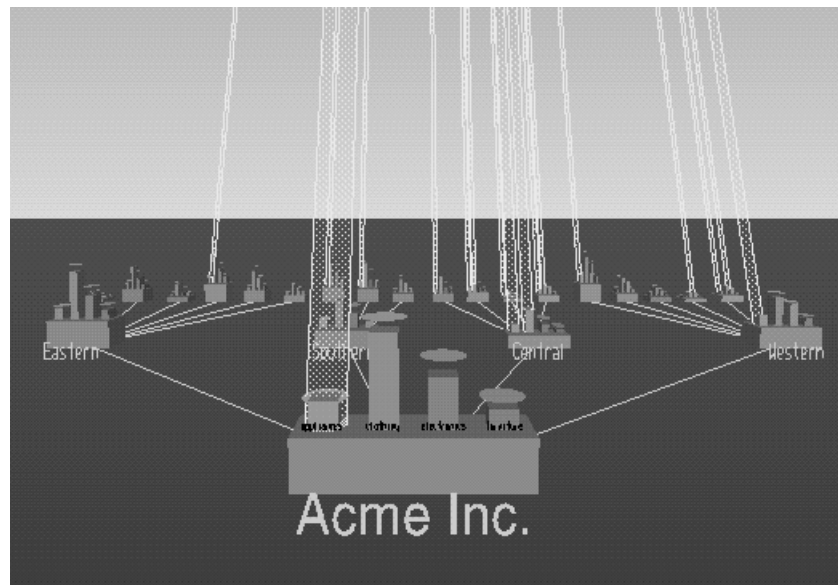


Figure 1-41 Sample Results of a Search in the Tree Visualizer

Once the search is complete, yellow spotlights highlight objects matching the search criteria (see Figure 1-41). To display information about an object under a yellow spotlight, move the pointer over that spotlight; the information appears in the upper left corner, under the label *Pointer is over:*. To select and zoom to an object under a yellow spotlight, left-click the spotlight; if you press the Control key while clicking, zooming does not occur.

Filter Panel

The Filter panel filters out selected information, thus fine-tuning the displayed hierarchy. You can use the Filter panel to emphasize specific information, or to shrink the amount of data for better performance. The Filter panel is similar for most MineSet tools, see “Filter Panel” on page 94.

The fields in the Tree Visualizer Filter dialog box follow the same conventions as that used in the Search dialog box. If the *Ignore Case In Filters* check box is checked, the comparisons of all string filters are case-insensitive.

If a node does not meet the filter criteria, has no bars that meet the criteria, and has no children that meet the criteria, the node is not shown. There can be, however, cases in which a specific object meets the filter criteria, but its ancestors up the tree do not. Also, other bars in the same node might not meet the criteria. Since position is important in interpreting context, it might not be good to eliminate those bars. Consequently, you are given an option of selecting one of three radio buttons that control how these objects should be drawn: *Solid*, *Outline*, and *Hidden*. Note, however, that if objects are drawn in a less solid form due to the Display Zeros or Display Null menu, they are displayed appropriately. For example, if Nulls are to be hidden, they are always hidden, regardless of the filter criteria.

The exception to this is when filtering to specific bars. In such a case, the other bars are eliminated and don’t take up space, regardless of the radio button settings.

The Height Filter slider lets you filter out those nodes containing only short bars. The size of a value is shown as a percentage of the maximum height. First, the tallest bar in the scene is calculated (if heights are normalized by level, then the tallest bar in each level). Then only those nodes that contain at least one bar that is the appropriate percentage of the tallest bar are shown.

For example, if you enter 5% in this field, then only those nodes containing at least one bar that is at least 5% of the height of the tallest bar are shown. (Also shown are ancestors of such bars). This option is intended as a coarse way to filter out small, uninteresting nodes. It is not intended as an exact mechanism of identifying specific nodes of a certain value; use the search panel for that purpose. Use of this option can accelerate the rendering of slow, complex scenes, or reduce clutter resulting from many bars near zero height. You can also set this filtering option in the configuration file by using the Height Filter command.

Although small nodes are filtered out, they are nonetheless counted in any cumulation up the hierarchy.

The Depth slider, which is under the Height Filter slider, lets you display the hierarchy so that only a given number of levels are displayed at any given time. When you are at the top of the hierarchy, only the number of hierarchical levels specified by the slider is seen. The nodes in the rows are arranged to optimize their visibility. When navigating to nodes lower in the hierarchy, additional rows are made visible automatically. The nodes above them automatically adjust their locations to accommodate the newly added nodes; thus, some nodes might seem to move. Note that the overview shows all nodes in the hierarchy, not just the top nodes; thus, the layout of the overview might not match the layout of the main view. The X in the overview approximates the corresponding location in the main view; there is no exact mapping between the two layouts.

To start filtering, click the *Filter* button. If the *Enter* key is pressed while the panel is active, filtering automatically starts. To close the panel, click the *Close* button.

Tree Visualizer Selection Menu

The Selection menu lets you drill through to the underlying data. This menu has five items.

- *Show Values* displays a table (Record Viewer) of the values for all selected objects.
- *Drill Through* opens the Drill Through Dialog, which gives you the option to view the selected data in the Record Viewer or send the data to the Tool Manager as a filter, a new column, or as SQL. You can send the records to a new copy of Tool Manager or you can send all non-selected records to the Tool Manager (complement drill through).
- *Normalize Subtree* determines the maximum height of the elements in the subtree, and normalizes all values relative to that height.

For further details on drilling through, see “Drill Through” on page 77.

Tree Visualizer Display Menu

The Tree Visualizer's Display menu lets you control several display parameters.

- *Base Heights* is a check box that lets you turn the heights of the bases on and off. To see negative numbers, or to make it easier to compare the bar heights, turn this option off. Turning it on provides summary information about all the bars. The initial value of this toggle can be changed with the “base height” statement in the configuration file.
- *Mark Flags* is a toggle option that lets you turn on or off the flags representing marks (see the *MineSet Enterprise Edition User's Guide for Windows*).
- *Zeros* is a submenu that controls how objects with zero height are displayed. By default, they are shown like other objects: a solid cube of height zero (a plane). The submenu lets you specify them to be displayed as outlines (appearing as a hollow square), or to be hidden completely (not drawn). The initial value of this of this can be changed using the “zero” option in the configuration file.
- *Nulls* is a submenu that controls how objects of null height are displayed. It has the same options as the zero menu; however, the default for null options is to display the objects as an outline. The initial value can be changed using the “null” option in the configuration file.

Tree Visualizer Go Menu

The Go menu duplicates the functions of the buttons on the upper right-hand side of the main window. It also identifies keyboard shortcuts for some functions.

- *Home* takes you to a designated location. By default, this location is the initial view point of the scene. Initially, this location is the first viewpoint shown after invoking the Tree Visualizer and specifying a configuration file. If you have been working with the Tree Visualizer and have clicked the *Set Home* menu item, then clicking *Home* returns you to the viewpoint that was current when you last clicked *Set Home*. The keyboard shortcut for this function is Control+H.
- *Set Home* changes the Home location to your current location. Clicking the *Home* menu item then returns you to the viewpoint that was current when you last clicked *Set Home*.
- *View All* shows the whole hierarchy, keeping the tilt of the camera. To get an overhead view of the scene, tilt the camera to point straight down, then click the *View All* menu item.

- *Go Back* lets you return to the previous location. If you have just started the Tree Visualizer and have not moved from the home view, this menu item is grayed out. The keyboard shortcut for this function is Control+B.
- *Go Forward* lets you proceed to the location from which you clicked the *Go Back* menu item. If you have not clicked the *Go Back* menu item, the *Go Forward* menu item is grayed out. The keyboard shortcut for this function is Control+R.
- *Parent* is active only when an object is selected. If a bar is selected, clicking this menu item selects the base containing the bar. If a base is selected, clicking this menu item moves up the hierarchy to the parent node. Once the root node has been reached (highest level of the hierarchy), the *Parent* menu is grayed out. The keyboard shortcut for this function is Control+U.
- *Move Left* lets you select the next sibling to the left. If a bar is selected, the bar to the left of it is selected. If a base is selected, then, if the parent has another child to the left, that is selected. This button is grayed out if nothing is selected, or if the current selection has no sibling to the left.
- *Move Right* lets you select the next sibling to the right. If a bar is selected, the bar to the right of it is selected. If a base is selected, then, if the parent has another child to the right, that is selected. This button is grayed out if nothing is selected, or if the current selection has no sibling to the right.
- *First Child* lets you select the first child of the current node. This button is grayed out if there is no selection, if a bar is selected, or if the current selection has no children.
- *Last Child* lets you select the last child of the current node. This button is grayed out if there is no selection, if a bar is selected, or if the current selection has no children.

Help Menu

The Help menu is the same for all MineSet tools, see “Help (IRIX)” on page 96.

Null Handling in the Tree Visualizer

Nulls represent unknown data (see “Nulls in MineSet” in the *MineSet Enterprise Edition Interface Guide*).

In the Tree Visualizer, nulls can occur in the following cases:

- The database or data file contains a null value.

- The skipMissing option is not present in the configuration file (see skipMissing in “Creating Data and Configuration Files for the Tree Visualizer” in the *MineSet Enterprise Edition Interface Guide*) and data is present for the key value in one node of the hierarchy, but not in another. For example, in a representation of state budgets, if there is no record for state income tax for Texas, Texas would have an income tax of null. This is different than for the case where there is a record showing 0 as the income tax for Texas, in which case it would show a tax of 0.
- When the Tool Manager is used to make an array based on bins and no data falls into a specific bin, the value for that bin is null. For example, if there is no data for 30-40 year olds, that bin is null.
- When making an array in the Tool Manager and the null enum option is specified, an extra array entry, corresponding to the first bar in each bar chart, is created to represent the aggregation of all the values where the bin value is null. (This bar is labeled with a question mark (?), representing null. If there is no data for that null bin, the values associated with it are null as well.
Note: if all values throughout the data associated with the null bin are null, the Tree Visualizer ignores the null bin and does not display it.
- Expressions and aggregations of nulls can generate nulls.

When a null value is mapped to a visual attribute, special representations are used in the Tree Visualizer. If null is mapped to height, the object is normally drawn in outline mode (although this is configurable through the Display menu (see “Tree Visualizer Display Menu” on page 192) or the configuration file (see “Null” in “Creating Data and Configuration Files for the Tree Visualizer” in the *MineSet Enterprise Edition Interface Guide*). For a bar or a base, this looks like an empty square. (It does not look like a cube, since it has no height.) For a disk, it looks like a circle. If a null value is mapped to a color, it is drawn in a dark grey (see Figure 1-42).

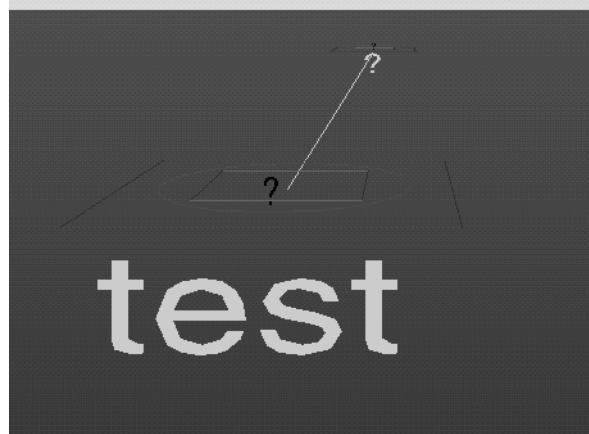


Figure 1-42 Representation of a Null Value Mapped to Height, Color, Disk, and Label

When selecting an object with a null value, it is shown as a question mark (?) in the selection field.

Tree Visualizer Restrictions

For the Tree Visualizer, the Tool Manager does not support the following:

- Non-aggregated hierarchies where the data is displayed directly without aggregating it.
- Real-time monitoring.
- A number of very rarely used options (skip missing, overview, shrinkage, root label, speed, climb speed, leaf margin, root leaf margin, leaf edge margin, initial position, initial angle, bar label size, base label size, and lod).
- Variable-length arrays.
- Expressions computed after creating the hierarchy. For example, if you are computing a percentage, the percentage must be computed after the hierarchy aggregation takes place, since it is not possible to aggregate the percentages.

Sample Configuration and Data Files

The provided sample configuration and data files demonstrate the Tree Visualizer's features and capabilities. Examples of the Tree Visualizer used to visualize Decision, Option, or Regression trees can be found in Appendix A, "Sample Configuration and Data Files."

Trimming Fraction

The trimming fraction is an Advanced Binning operation that allows you to exclude extreme values (called outliers) from the dataset before you start the binning operation. The default trimming fraction is 0.05. This excludes the 5% of the instances with the most extreme values (2.5% with the lowest values in the range, and 2.5% with the highest values in the range). The purpose of trimming is to reduce the influence of outliers on the generation of thresholds.

Uniform Range

Uniform range is a selection used in automatic binning of data in which the value range is divided into uniformly sized subintervals. See "Binning" on page 33 for a full discussion on how this is applied using the Tool Manager's *Bin Columns* button.

Uniform Weight

Uniform weight is a selection used in automatic binning of data in which the value range is divided into the specified number of equal weight bins, so that each bin contains the same number of instances which can result in equal weight if the weight is set to 1 per instance. See "Binning" on page 33 for a full discussion on how this is applied using the Tool Manager's *Bin Columns* button.

View Menu

The View menu lets you control various display options, and these are similar for most visualizers. Depending on your platform, the menu contains some or all of the following options:

- *Filter Panel* lets you reduce the number of entities displayed in the main viewing area, based on one or more criteria. You can use the filter panel to fine-tune the display, emphasize specific information, or simply shrink the amount of information displayed. The Set Landscape to Filter checkbox, which appears in the lower right of the filter panel, lets you specify whether the landscape in the main window covers the entire dataset or just the filtered data. See “Filter Panel” on page 94 to fill out this dialog box, or linger the mouse pointer over a field and press Shift F1.
- *Set Background Color* brings up a color chooser to let you specify a new background color.
- *Show Window Decoration* lets you hide or show the external controls around the display window.
- *Null Positions* toggles the display of null values.
- *Show Animation Panel* lets you show or hide the animation control panel. This menu item is disabled for datasets with no independent dimension.

Visualization Tools

This section provides an overview of the visualization tools accessible through MineSet’s Tool Manager that let you view your data using different visual metaphors:

- **Cluster Visualizer** displays statistics about the clusters or groups and places these statistics side-by-side with those for the entire data set, so that you can see which features make each cluster unique.
- **Decision Table Visualizer** lets you see data from a column in hierarchical levels. For example, you can examine the profitability of a business according to product class, geography, sales promotions and sales-representative compensation plan. Data is distributed two attributes at a time, so you can drill down to further pairs of attributes at each level.

- **Evidence Visualizer** lets you see how knowing the value of a particular attribute helps predict the probability of a given label. For example, if in the iris dataset you looked at the attribute “sepal length is 5.45...5.85”, you could see that the probability of the label being iris-setosa is 86.54%. The Evidence Visualizer shows how various attributes contribute to the decisions generated when the Evidence Inducer builds a model, and allows “what-if” analysis.
- **Histogram Visualizer** automatically bins all of the continuous-type columns in the data and sends the result to the Statistics Visualizer to be displayed as histograms.
- **Map Visualizer** lets you visualize data relationships that exist across geographically meaningful areas. For example, you can visualize different areas of a country, showing the relative impact of a marketing program. The Map Visualizer’s drill-down capabilities let you focus on designated regions and perform a more detailed analysis in smaller geographical elements.
- **Record Viewer** lets you view the data in rows and columns resembling a spreadsheet.
- **Scatter Visualizer shows data points in one-, two-, or three-dimensional space.** Additional attributes can be mapped to size, color, and shape. Finally, two additional attributes may be mapped to sliders, allowing animation and fly-throughs, for a total of eight dimensions. When you move the slider the display changes to reflect the changes in the independent variables. Scatter Visualizer is also used to display Association Rules.
- **Splat Visualizer** has many of the same features as the Scatter Visualizer with the distinction that density of data is shown using opacity. When large amounts of data need rendering, and plotting each individual point is inefficient, this tool is appropriate.
- **Statistics Visualizer** computes and displays summary information for the current dataset (maximum, minimum, median, standard deviation, distinct values, and quartiles).
- **Tree Visualizer** is useful in analyzing data with hierarchical relationships. The interactive “fly-through” approach allows you to examine data relationships at different hierarchical levels. For example, the Tree Visualizer can be used to examine a company’s product line, graphically displaying each product’s contribution to the company’s total revenue. Each branch of the hierarchy displays information at increasing levels of detail, displaying revenues according to product lines and, eventually, individual products.

The Tree Visualizer is also used to view the resulting models of the Decision Tree and Option Tree Classifiers, and the Regression Tree Regressor; with each decision being represented by a separate node in the tree.

Warning Options

On UNIX, when operating from the command-line, there are two MineSet options that affect how any tool is invoked:

- **-warnexecute** indicates that if you attempt to execute a command specified in an execute statement, a warning is displayed and you are given the option to execute the command or not. This is intended for an insecure environment, such as files obtained from the Web, and is used automatically when commands are executed using mtr files. (See the *MineSet Enterprise Edition Interface Guide* for an explanation of mtr files.)

You can enable this option permanently—

- on UNIX systems by adding:

```
*minesetWarnExecute:TRUE
```

to your *.Xdefaults* file, or

- on Windows systems by changing the value of:

```
MINESET_WARN_EXECUTE
```

in the registry. You can do this using the File Preferences dialog.

- **-quiet** eliminates the dialogs that popup to indicate progress. You can enable this option permanently by adding the line

```
*minesetQuiet:TRUE
```

to your *.Xdefaults* file.

On Windows, these options are available in the Preferences dialog box of the 3D Visualizer (available from the File menu).

Web Publishing

MineSet Web extensions allow visualizing files and data generated by MineSet software over the Web. The MineSet *.mtr* extension lets you place MineSet configuration, schema and data files into an archive file, which can be embedded in a web page as an HTML tag. When this page is loaded in Internet Explorer, the MineSet visual tool is launched within the browser's window. The machine that the browser is running on must have the MineSet client software installed.

For examples and installation instructions, refer to the *MineSet Enterprise Edition Interface Guide*.

Weighting

Often weight represents actual record count, but when a dataset is unevenly sampled, some records may be given more importance by weighting them, that is, by increasing their importance. Weighting records can be done from the Tool Manager Data Destination pane by selecting Mining Tools. For any inducer click the *Further Options* (or *Advanced Mode*) button to get a dialog box which has a Use Weight check box. The semantics of second weighting is that a record weight of 2 is equivalent to two records with a record weight of 1. Floating point weights are allowed. See also "Record Weighting" on page 137.

Year 2000 Compliance

MineSet supports Y2K-compliant dates. In the U.S. locale, dates may be entered in the form MM/DD/YY or MM/DD/YYYY. MineSet follows the X/Open standard for two-digit years: two-digit years greater than 68 are assumed to be the years 1969 to 1999, and two-digit years less than or equal to 68 are assumed to be the years 2000 to 2068.

In European locales, dates may be entered in the form DD/MM/YY or DD/MM/YYYY, with the same handling of two-digit years as above.

In either locale, if you enter a two-digit year, it is automatically expanded to a four-digit year in the display.

Sample Configuration and Data Files

Several sample configuration and data files are available with MineSet to demonstrate the capabilities of various tools. In this section sample files for each tool are detailed and explained. The entries are arranged in alphabetical order by tool.

- Windows users find the files in which MineSet was installed, under *\examples*.
- IRIX users find the files in */usr/lib/MineSet/examples*.

The file descriptions are:

- “Association Rules Sample Files” on page 202.
- “Clustering Sample File” on page 202.
- “Column Importance Sample File” on page 203.
- “Decision Tree Sample Files” on page 204.
- “Decision Table Sample Files” on page 213.
- “Evidence Visualizer Sample Files” on page 227.
- “Map Visualizer Sample Files” on page 238.
- “Option Tree Sample Files” on page 240.
- “Regression Tree Sample Files” on page 243.
- “Scatter Visualizer Sample Files” on page 247.
- “Splat Visualizer Sample Files” on page 250.
- “Tree Visualizer Sample Files” on page 252.

Association Rules Sample Files

The following sample data and configuration files are provided to visualize Association Rules based on prepared datasets. Some of these files correspond to hierarchical datasets. Rules files contain the generated rules obtained by running the Association Rules Generator. The files containing the rules should, by convention, have a *.rules.data* extension. Each configuration file specifies how the corresponding rules file is displayed. Configuration files must have a *.scatterviz* extension. The files mentioned in this subsection are in the Windows directory in which MineSet was installed, under *\examples*, or the IRIX directory */usr/lib/MineSet/scatterviz/examples*.

- *group.rules.data* and *group.rules.scatterviz*
These files provide the generated rules and configuration specifications for product groups, such as bread and baked goods, dairy milk, and carbonated beverages.
- *category.rules.data* and *category.rules.scatterviz*
These files provide the generated rules and configuration specifications for product categories within product groups, such as refrigerated or non-refrigerated milk.
- *adult94.rules.data* and *adult94.rules.scatterviz*
These files provide the generated rules and configuration specifications for a census dataset, showing associations between marital status, education level, age, income, and other variables.
- *germanCredit.rules.data* and *germanCredit.rules.scatterviz*
These files provide the generated rules and configuration specifications for a credit dataset from Germany, showing associations between credit history, employment, savings, and other variables.
- *cars.rules.data* and *cars.rules.scatterviz*
These files provide generated rules and configurations specifications for the cars dataset, showing associations among the various attributes.

Clustering Sample File

The following example shows a case in which clustering might be useful. This example is associated with a sample dataset provided with MineSet. It shows how to work with the Clustering mining tool, and explains the different outcomes and options.

The cars dataset is relatively simple, dealing with familiar concepts of horsepower, vehicle weight, and time required to reach 60 mph.

When the Cluster Visualizer first appears, the attributes are arranged from top to bottom according to how useful they are for differentiating among all clusters.

If you select cluster 1, that cluster then controls the priority ordering of attributes represented by the bar charts and histograms. The order of attributes in other clusters will also be based on cluster 1. For example, if you click on cluster 1, the attribute sequence is cylinders, weight, then miles per gallon. The change in ordering will only appear in the visualization, not the basic dataset. You can compare the same row across the other clusters to see how that attribute differs from cluster to cluster. When you select cluster 2, you see a different order of attributes at a lower level. In this case, origin is most important, then cylinder, then horsepower, then miles per gallon.

Column Importance Sample File

The following example shows a case in which Column Importance might be useful. This example is associated with a sample dataset provided with MineSet. It shows how to work with the Column Importance mining tool, and explains the different outcomes and options.

When customers change their phone carrier from one telecommunications company to another, this is termed “churning.” This is a common problem in the telecommunications industry. The files *churn.schema* and *churn.data* were used to generate this example. Windows users can find them in the directory in which MineSet was installed, under *\data*. IRIX users can find them in */usr/lib/MineSet/data*.

Running the simple Column Importance mode yields the following three attributes:

- Total Day Minutes.
- Number of customer service calls.
- State.

By running “compute improved purity” from the advanced mode, you can see that Total Day Charge and Total Day Minutes have the same purity ranking (48.67). By moving one of them to the right (for example, Total Day Minutes) and rerunning Compute Improved Purity, you can see that there is no value to the other (Total Day Charge). These two attributes are highly correlated.

Looking at the attributes when Total Day Minutes is on the right, we can see that the following are good:

- International plan (4.1)
- Number of Customer Service Calls (8.1).
- State (4.7)

You can choose to move International Plan to the right, because this information is readily available and easy to measure.

The other two attributes (Number of Customer Service Calls and State) remain highly important (in fact, their importance increases), so they are apparently not correlated with the International Plan.

By looking at the importance of attributes this way, you can determine which ones can be substituted with others that are equally good (or almost as good), but are easier to measure or understand. By looking at the purity, you can determine how much the additional attributes help. For example, in the above scenario, state significantly improves the purity. In the *iris* dataset, the third attribute chosen (*sepal length*) raises the purity only slightly higher. The simpler, two-dimensional scatterplot will give nearly as much information as a three-dimensional one.

Decision Tree Sample Files

The following examples illustrate cases in which the Decision Tree inducer can be useful. Each of these examples is associated with a sample data file provided with MineSet. By running the inducer, you can generate the *-dt.treeviz* files described below.

The data files can be loaded into MineSet by opening the corresponding *.schema* file from the *data* directory, (for example *churn.schema*). The classifier visualization files, which have a *-dt.treeviz* extension, can be opened from the *examples* directory.

- Windows users find these files in the *data* and *examples* directories of the directory in which MineSet was installed.
- IRIX users find these files in the *data* and *examples* directories of */usr/lib/MineSet/treeviz/examples*.

Churn

When customers change their phone carrier from one telecommunications company to another, this is termed “churning.” This is a common problem in the telecommunications industry. The file in the *examples* directory, *churn-dt.treeviz*, shows a Decision Tree classifier induced for this problem. The file was generated by running the inducer on the file in the *data* directory, *churn.schema*, with the label set to churn (yes, no). The file given is fictitious, but based on patterns found in real data.

In this tree the root split is on the amount of time the customers talk during the day (total day minutes). Customers who talk more than 264 minutes per day churn at a significantly higher rate than those who don’t (60% versus 11%). These also are probably the most profitable customers.

The left subtree represents customers who talk less than 264 minutes per day. They have a churn rate of 11%; but if they make more than three customer service calls, the churn rate increases to 49%.

The right subtree represents customers who talk over 264 minutes per day. They have a churn rate of 59%; but if they have a voice-mail plan, the rate decreases to 9.3%. If they do not have a voice-mail plan, the churn rate is almost 75%.

Origin of Cars

The *cars* dataset contains information about different models of cars from the 1970s and early 1980s. Attributes include weight, acceleration, and miles per gallon (mpg). The file from the *examples* directory, *cars-dt.treeviz*, shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on the file from the *data* directory, *cars.schema*, with the label set to origin (Japan, U.S., Europe). If you have a dataset of car attributes, you might want to know what characterizes cars of different origins.

- Windows users find these files in the directory in which MineSet was installed, under *examples\cars-dt.treeviz* and *\data\cars.schema*
- IRIX users find these files in */usr/lib/MineSet/treeviz/examples/cars-dt.treeviz* and */usr/lib/MineSet/data/cars.schema*

Note that in the tree the left split is on brand. The root split is not brand because the Decision Tree inducer penalizes multi-way splits; and the split on cubic_inches was deemed a better discriminator. You can use the Tool Manager Remove Column transformation to hide the brand, thus making the problem more interesting.

In the Decision Tree, you can see that cubic inches is an excellent discriminator for U.S.-made cars. Cars with large engines (>169.5 cubic inches) are all made in the U.S., but smaller cars are made everywhere. By choosing Selections > Show Original Data, you can see that the one car with a big engine that was not made in the US is a Mercedes. Note that in this tree, the root node (that is, the entire training dataset) has many more U.S. cars (62.50%), yet after a single split on the cubic inches attribute, it is more difficult to predict the origin of cars with small engines. The purity of the root is 16.2 showing that there is one class (U.S., in this case) that is dominant. The right node (cubic inches > 169.5) has purity 96.81, indicating that we have identified a very pure subpopulation (almost all cars with large engines were made in the U.S). Indeed, the error rate for the right subtree is estimated at 0% (green base). The left node from the root has purity 0.23 and a much higher error rate of 31.25% (orange base). This subproblem is much harder than the original one: the number of records for each class is approximately the same.

Predicting Gender

The *adult* dataset contains information about working adults. This dataset was extracted from the U.S. Census Bureau. It contains data about people older than 16, with a gross income of more than \$100 per year who work at least one hour a week. You might want to know how to characterize males and females. The file *adult-sex-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on *adult.schema*, with the label set to *sex*. This dataset contains almost 50,000 records; so running the Decision Tree Inducer can take several minutes when you run this on your workstation.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\adult-sex-dt.treeviz` and `\data\adult.schema`
- IRIX users find these files in `/usr/lib/MineSet/treeviz/examples/adult-sex-dt.treeviz` and `/usr/lib/MineSet/data/adult.schema`

The resulting visualization provides the following insights:

- Relationship is a giveaway attributes for some values. Husbands usually are male. (Interestingly, there is one husband that is a female, showing data quality problems at the Census Bureau, which does not recognize same-sex marriages.) Similarly, if the person is a wife, the person is usually a female, except for three records that show otherwise.

To make the problem more interesting, remove the relationship attribute and generate a new Decision Tree. In this case:

- The most important attribute is marital status.
- From the height of the bases, most people are either divorced, married to a civilian spouse, or never married. Few are married with spouse absent, separated, married to armed-forces spouse, or widowed.
- The distribution at the root shows more males in this dataset. (This dataset contains information about working adults and is not representative of the entire population.)
- The left-most node contains divorced working adults. We can see that the distribution is more balanced than at the root (60% female, 40% male). The second node contains married working adults. We can see that 89% are males. The third node contains working adults that have never married. Their numbers are approximately equal to those in the divorced group, with slightly more males. The right-most node contains working widowed adults, of which 81% are females (probably because of their higher life expectancy). The term “widowed” refers to anyone who has lost a spouse.

If you want to target working females for a new product, you can use the search panel to identify segments that have a large population of females. You can do this by choosing

- sex matches female (click female on the top portion of the window)
- subtree weight > 1000
- percent > 80

Three yellow spotlights show the matching nodes. Since two are on one path, look at the node closest to the root (on the right). The paths translate into the rules

```
marital status = Widowed implies that 81.23% are female
marital status = Divorced and occupation =
    administrative clerical implies that 87.67% are female
```

In this training set, 1233 (widowed) and 1045 (divorced and occupation) females satisfy these rules out of 16,192 at the root. This simple segment contains over 14% of the working women in the dataset.

Salary Factors

If you have a dataset of working adults, you might want to find out what factors affect salary. You might then divide the records into two classes: those adults earning under \$50,000 a year, and those earning more. Each record then has an attribute with one of two values: “– 50,000” and “50,000+”. You can run a MineSet classifier to help determine what factors influence salary. The examples file *adult-salary-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on the data file *adult.schema* with *gross_income* binned at the user-specified threshold of 50000 and the label set to *gross_income_bin*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\adult-salary-dt.treeviz` and `\data\adult.schema`
- IRIX users find these files in `/usr/lib/MineSet/treeviz/examples/adult-salary-dt.treeviz` and `/usr/lib/MineSet/data/adult.schema`

The resulting visualization provides the following insights:

- The root, which represents the entire training set, shows 76.07% of the working adults earn under \$50,000.
- Age is the most important factor. Only 3.07% of the people under 27 years old earn more than \$50,000. The base color is green, indicating a very accurate rule (about 3% error rate).
- Education is an important factor for predicting salary for people over 27 years old. The Census Bureau assigns education levels to each person. The Decision Tree classifier splits on 12.5; the level 13 matches a Bachelor’s degree. People with a Bachelor’s degree or higher, go right to the node where about 55% earn over \$50,000.
- Of the segment that is older than 27 years and well educated, relationship is an important predictor of salary. For those persons that are married, chances of earning \$50,000 or more increase to 73% for husbands and 75% for wives. (However, the node containing wives has a small base, indicating that few females match this rule.) If the person in this group is not married, chances of earning \$50,000 or more decrease to 27% for males and 25% for females.

Iris Classification

In this dataset, each record describes four characteristics of iris flowers: petal width, petal length, sepal width, and sepal length. Each iris was further classified into the types *iris-setosa*, *iris-versicolor*, or *iris-virginica*. The goal is to understand what characterizes each iris type.

Before running a classifier, click the Importance tab in the Tool Manager's Classifiers tab; then click *Go*. You obtain a ranking of the importance of the features: `petal_width`, `petal_length`, and `sepal_length`. You can map these to the axes in the Scatter Visualizer, with the `iris_type` mapped to the color, and see the clusters.

The file *iris-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on *iris.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\iris-dt.treeviz` and `\data\iris.schema`
- IRIX users find these files in `/usr/lib/MineSet/treeviz/examples/iris-dt.treeviz` and `/usr/lib/MineSet/data/iris.schema`

Running the Tree Visualizer, you can see that the root has 6% error rate, even though the purity is very low (0). The purity measures the skewness of the distribution, and, at the root, the distribution is perfectly uniform: 50 records for each label value. The left branch (`petal-length <= 2.6` inches) goes to a green node (zero error) containing only *iris-setosas*. The other branches are also quickly able to separate the classes using another test on the `petal_width`. The path `petal-length > 2.6` and `petal-width <= 1.65` and `petal-length > 5` ends with an impure leaf containing 4 records. There are three records of type *iris-virginica* and one of *iris-versicolor*. The Decision Tree did not split this node because it was deemed insignificant (by default, every split must contain two children with at least a weight of two). The node color is also black, indicating that no test instances reach this node, so we do not have an estimated error rate for it.

To summarize: the flowers with petal length ≤ 2.6 inches are predicted as *iris-setosa*, those with petal length > 2.6 inches and ≤ 5 inches and petal width ≤ 1.65 inches are predicted as *iris-versicolor*, and those with a petal length > 2.6 inches and a petal width > 1.65 or petal length > 5 inches and petal width ≤ 1.65 are predicted as *iris-virginica*.

Because the Decision Tree makes binary splits on continuous attributes while Column Importance discretizes the data, the root split of the tree is different from the first attribute in column importance (see "Column Importance" on page 52 for more details).

Mushroom Classification

The file *mushroom-dt.treeviz* shows the Decision Tree classifier induced for the classification of mushrooms. This file was generated by running the inducer on *mushroom.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\mushroom-dt.treeviz` and `\data\mushroom.schema`
- IRIX users find these files in `/usr/lib/MineSet/treeviz/examples/mushroom-dt.treeviz` and `/usr/lib/MineSet/data/mushroom.schema`

The goal is to understand which mushrooms are edible and which are poisonous, given this dataset. There are over 8000 records in this set; thus, running this inducer might take several seconds.

Each mushroom has many characteristics, including cap color, bruises, and odor. If you build a Decision Tree classifier, you can see that using only the odor attribute lets you determine in 50% of the cases whether the mushroom is poisonous or edible. If the mushroom has no odor, there is a 3.4% chance it is poisonous. The next attribute to look at is the shape of the stalk. If it tapers, the mushroom is edible; but if it enlarges, there is a 11.6% chance the mushroom is poisonous. There are 1032 mushrooms that reach this node. You can follow the tree down further nodes to see what other attributes to consider.

Party Affiliation

This dataset consists of voting records. The goal is to identify the party a congress person belongs to given data about key votes. The dataset includes votes for each member of the U.S. House of Representatives on the 16 key votes identified by the *Congressional Quarterly Almanac (CQA)*. The CQA lists nine types of votes: voted for, paired for, and announced for (these three are simplified to yes); voted against, paired against, and announced against (these three are simplified to no); voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three are simplified to an unknown disposition).

Before running a classifier, look at the 16 votes to see if you can perceive which features are important. Then run the Decision Tree classifier.

The file *vote-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on *vote.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\vote-dt.treeviz* and *\data\vote.schema*
- IRIX users find these files in */usr/lib/MineSet/treeviz/examples/vote-dt.treeviz* and */usr/lib/MineSet/data/vote.schema*

Breast Cancer Diagnosis

The breast cancer dataset contains information about women undergoing breast cancer diagnosis. Each record is a patient with attributes such as cell size, clump thickness, and marginal adhesion. The final attribute is whether the diagnosis is malignant or benign. The file *breast-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducers on *breast.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\breast-dt.treeviz* and *\data\breast.schema*
- IRIX users find these files in */usr/lib/MineSet/treeviz/examples/breast-dt.treeviz* and */usr/lib/MineSet/data/breast.schema*

The Decision Tree shows that uniformity of cell size is a very strong discriminatory attribute. While the root distribution is about 65% versus 35% (purity is 7.07), the two children of the root are much more skewed, with the left node having an error rate of only 1.29%. The root alone is an excellent discriminator: if you limit the tree height to a single level, the error rate is 7.3%.

Hypothyroid Diagnosis

The hypothyroid diseases dataset is similar to the one for breast cancer, except that we are trying to predict hypothyroidism rather than cancer. The file *hypothyroid-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on *hypothyroid.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\hypothyroid-dt.treeviz* and *\data\hypothyroid.schema*
- IRIX users find these files in */usr/lib/MineSet/treeviz/examples/hypothyroid-dt.treeviz* and */usr/lib/MineSet/data/hypothyroid.schema*

There are 3163 records in this dataset and most of them do not have hypothyroid (95.23%). This means that one can predict “negative” and be correct most of the time. However, we are worried about those people that have hypothyroidism, yet the model predicts to be healthy. The false negatives are very important. By selecting a confusion matrix from Further Inducer Options, you’ll see that there are five patients with hypothyroidism who were misclassified.

Looking at the Decision Tree, you can see that the root node is green (highly accurate). The single attribute on *fti* at the root shows that it is relatively easy to identify many of the negative diagnoses. People with high *fti* are 99.7% negative, and all those where the value is unknown are also negative (perhaps the doctor decided not to measure this attribute because something else was obvious), but the rest (218 people) are difficult to diagnose cases. We started with 3163 records, but only 218 are really “interesting” to mine because it was very easy to determine the classification of most cases. In this example most of the data is uninteresting and you want to concentrate on a small part quickly. Of the 218 people, you can see that about 66% are positive and 34% negative.

As you move down the tree, increase the height scale (slider on the top left of the visualizer) to see the different heights. The node that catches most of the people with hypothyroidism has the conditions “*fti* <= 64.5 and *tsh* > 5.95.” It contains 140 of the 151 records that have hypothyroidism.

Pima Diabetes Diagnosis

This dataset is a diagnosis problem for diabetes using statistics gathered from a Native American tribe in Phoenix, Arizona. The task is to determine whether a patient has diabetes, given some medical attributes, such as blood pressure, body mass, glucose level, and age.

The file *pima-dt.treeviz* shows the Decision Tree classifier induced for this problem. This file was generated by running the inducer on *pima.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\pima-dt.treeviz` and `\data\pima.schema`
- IRIX users find these files in `/usr/lib/MineSet/treeviz/examples/pima-dt.treeviz` and `/usr/lib/MineSet/data/pima.schema`

DNA Boundaries

There are 3,186 records in this DNA dataset. The domain is drawn from the field of molecular biology. Splice junctions are points on a DNA sequence at which “superfluous” DNA is removed during protein creation. The task is to recognize exon/intron boundaries, referred to as EI sites; intron/exon boundaries, referred to as IE sites; or neither. The IE borders are referred to as “acceptors” and the EI borders are “donors.” The records were originally taken from GenBank 64.1 (*genbank.bio.net*). The attributes provide a window of 60 nucleotides. The classification is the middle point of the window, thus providing 30 nucleotides at each side of the junction.

In this example, the root of the Decision Tree shows the distribution of the three classes. By pointing to the bars, you can see that the composition is about 24% exon/intron, 24% intron/exon, and 52% none. The “left_01” in front of the root node indicates that this is an important attribute to look at first. The “left_01” notation refers to the first nucleotide found to the left of the splice junction in question. The choices of attribute values for this first nucleotide (and all nucleotides in general) are the “A”, “G”, “T”, and “C” nucleotides. If the “left_01” nucleotide is a “G”, then the “G” branch is taken and followed to the next node, where the distribution now shows that such a nucleotide is more likely to be an “exon/intron” or an “intron/exon” than at the root: the distribution is 34% for “exon/intron,” 42% for “intron/exon”, and 24% for “none.” If the “left_01” nucleotide is an “A”, “T”, or “C”, then the corresponding “A”, “T”, or “C” branch is taken instead and in all three cases, the probability of “none” increases dramatically (87%, 87%, and 95% respectively). This testing and branching process is repeated until the final node with the predicted class (“exon/intron”, “intron/exon”, or “none”) is reached.

For this dataset, the Evidence Classifier is more appropriate than a Decision Tree due to the probabilistic nature of this domain. This can be verified by comparing the estimated error rates.

Decision Table Sample Files

The following examples show cases in which the Decision Table can be useful. Each of these examples is associated with a sample data file provided with MineSet. By running the Decision Table inducer (with Suggest Using Feature Search turned on in the Further Inducer Options), you can generate the *-dtab.dtableviz* and *-dtab.dtableviz.data* files described below.

Note: The data (*.data*) and accompanying schema (*.schema*) files are located in the *data* directory on the client workstation. The classifier visualization files, which have a *-dtab.dtableviz* extension, reside on the client workstation in the *examples* directory. To load a data file into MineSet, open the *.schema* file.

- Windows users find these files in the directory in which MineSet was installed, under *\examples* and *\data*
- IRIX users find these files in */usr/lib/MineSet/treeviz/examples* and */usr/lib/MineSet/data*

Churn

Churn is when a customer leaves one company for another. This example shows what causes customer churn for a telephone company. The files *churn.schema* and *churn.data* were used to generate this example. Windows users can find them in *Program Files\SGI\MineSet\data*. IRIX users can find them in */usr/lib/MineSet/data*.

The file *churn-dtab.dtableviz* shows the structure of the classifier induced using the attribute *churned* as the label. The error rate for this classifier is 5.5%. Of the records, 14.3% represent customers who churned. The two attributes selected for the first level of detail were number of customer service calls and total day charge. By looking at the distribution over these two attributes, you can see that churn increases as total day charge increases, except when the total day charge is less than 29.75. Then the churn is high if the number of service calls is more than 3. About 3/4 of the records have total day charge less than 38 and 3 or fewer customer service calls.

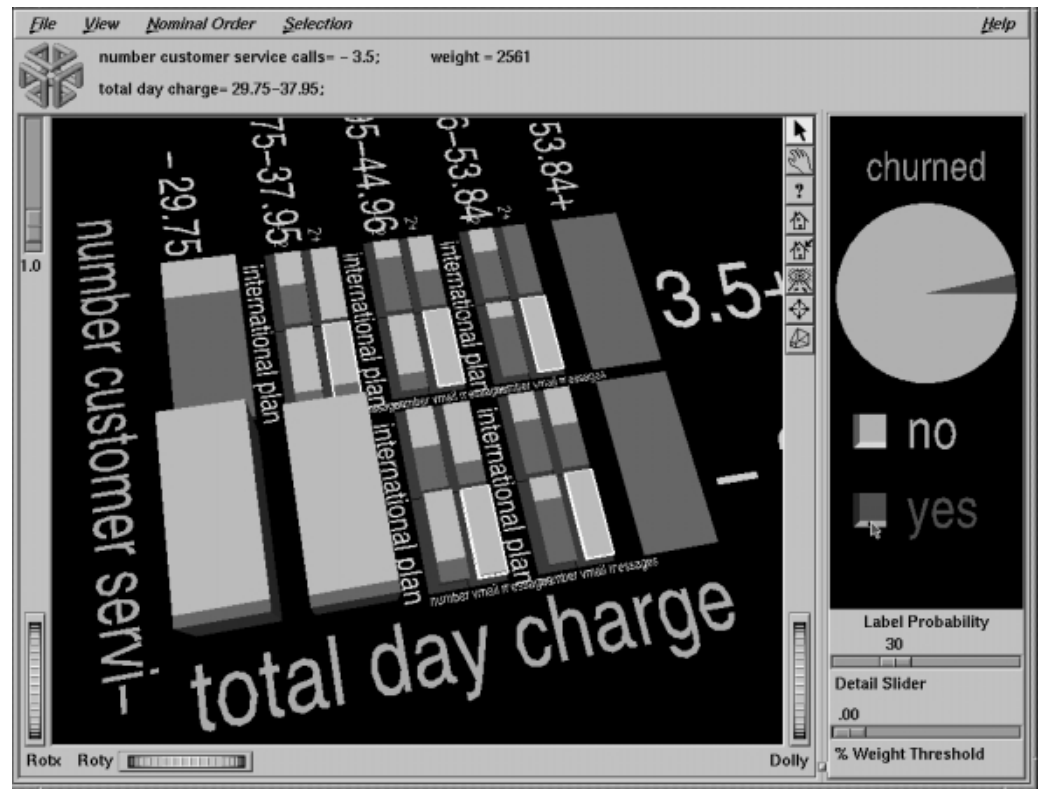


Figure A-1 Drilling Down on the Churn Dataset

Begin to drill down on regions where it's not clear when a customer churns. Figure A-1 shows drilling down on all cakes in which there was not a clear majority class. The next attributes considered are international plan and number of vmail messages. Among those with heavy day charge, it appears clear that having the international plan and having few voice mail messages correlates well with customer churn. Selecting the lower right cake in each of the drill-down regions, and then brushing with the mouse across the box next to "churned=yes" in the probability pane on the right, shows that only 3.4 percent of the customers in the selected regions churned.

Now drill down further on the cake in the upper left of each previous drill-down region. Doing so shows a very similar distribution for each. The consistent pattern shows: high total evening charge and high total international charge correlate well with churn. You can even drill-down another level to see the effect of total international calls, but doing so leaves so few records from which to draw conclusions, you could not be confident of a prediction made based on this sample. If you are interested in how total international calls affects churn and how it correlates with other variables, return to the Tool Manager and explicitly map total international calls to a higher level in the hierarchy, and rerun the decision table.

Although “state” is fairly well correlated with churn, it was not selected because the algorithm has a built-in preference for variables with few values. This prevents the algorithm from selecting attributes like social security number which uniquely identify each record, thus yielding high training set accuracy, but are not useful for classifying future unlabeled data.

Origin of Cars

The *cars* dataset contains information about different models of cars from the 1970s and early 1980s. Attributes include weight, acceleration, and miles per gallon (mpg). The file *cars-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this dataset. This file was generated by running the inducer on *cars.schema* with the label set to “origin” (Japan, U.S., Europe).

- Windows users find these files in the directory in which MineSet was installed, under `\examples\cars-dtab.dtableviz` and `\data\cars.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/dtableviz/cars-dtab.dtableviz` and `/usr/lib/MineSet/data/cars.schema`

Since the brand attribute uniquely determines the origin, the structure of the classifier is extremely simple. The only two attributes shown are brand and cylinders. It is interesting to see which brands tend toward high or low cylinder types. For example, there are 21 different models of Mazda, and 18 models of Honda, but they all have 5 or fewer cylinders. Conversely, Cadillac only makes cars with six or more cylinders.

This example could probably be made more interesting by first removing the brand attribute. Another useful transformation might be to convert cylinders to string so each unique cylinder value is shown, rather than a bin. Alternatively, one can create additional levels of detail beyond brand and cylinder, by mapping them explicitly.

Predicting Gender

The *adult* dataset contains information about working adults. This dataset was extracted from the U.S. Census Bureau. It contains data about people older than 16, with a gross income of more than \$100 per year who work at least one hour a week. You might want to know how to characterize males and females.

The file *adult-sex-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *adult.schema*, with the label set to *sex*, after removing the relationship column (which would have made the classifier trivial). To make it easier to see the distribution of records for each combination of values, you can scale the cake heights using the scale slider on the left.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\adult-sex-dtab.dtableviz` and `\data\adult-sex.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/dtableviz/adult-sex-dtab.dtableviz` and `/usr/lib/MineSet/data/adult-sex.schema`

In the Decision Table Visualizer, the Label Probability Pane shows that the prior probability of working males is higher than that of females. The Evidence Visualizer showed us that marital status and occupation are very important attributes for determining gender, however, it did not show us the dependencies between these two attributes. The top level shows several interactions. For example, most people with occupation *craft repair* are married civilian spouses (more specifically 98.6 of them are husbands), while most people with occupation *“Other-service”* have *“Never-married”* (48% male).

At first it may seem odd that most of the people with *“marital_status = Married-civilian-spouse”* are male, but once you consider that this data was probably gathered from tax returns, it seems reasonable that the wives of these males are not working, but filing jointly with their husbands.

The divorce rate seems highest among those with *“occupation = Admin-clerical”*. Those in *“Other-service”* also have high divorce rates, but they seem to prefer separation, as the number who are separated is even greater than that of *“Admin-clerical.”*

Suppose you wanted to find out the probability of being female given that a person is *“widowed”* and has *“occupation=Adm-clerical”*. In the evidence visualizer one can get an approximate answer (94.7% female) for this by clicking on these two attribute values. Here we can get the exact answer by clicking the left mouse on the cake at the intersection of these two values (95.2% female).

Drill down to the lowest level on the cake for “Married-civilian-spouse” and “Occupation= Unknown.” There is a pattern evident here more than any of the other combinations of occupation and marital status. For this cake, the younger members tend to be women, and the older members tend to be men.

Salary Factors

For a dataset of working adults, you might want to find out what factors affect salary. First bin gross income into two bins, those that earn less than 50,000, and those earning over 50,000. You can run a MineSet classifier to help determine what factors influence salary. The file *adult-salary-dtab.dtableviz* shows the Decision Table classifier induced for this problem. This file was generated by running the inducer on *adult.schema* with *gross income* divided into five bins using user-specified thresholds.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\adult-salary-dtab.dtableviz* and *\data\adult-salary.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/adult-salary-dtab.dtableviz* and */usr/lib/MineSet/data/adult-salary.schema*

Since the label is numeric, a continuous spectrum is used to assign colors to each class. Also the classes in the probability pane on the right are not sorted by slice size because they have a numeric order. Red is assigned to the highest bin (50,000+).

The two attributes chosen at the top of the hierarchy are relationship and “education_num.” The attribute education_num is not particularly useful because it is simply an enumeration of the different educations possible, not years of education, as you might think. However there is an approximate correlation. Replace this column with education if you prefer to see the actual string values. If you simply remove the column, education_num, and rerun using feature search, the algorithm may not pick education at the top of the hierarchy because it has so many values.

The order of the attributes in this model were selected automatically to increase accuracy. Often a model in which domain knowledge is used to perform the mappings can give a more useful visualization. Such a model is provided by *adult-salary3-dtab.dtableviz*, and shown in Figure A-2. Here the salary has first been binned into 3 ranges (20,000, and 60,000 are the thresholds). The attributes mapped at level one are: relationship and sex; level two has education and occupation; and level 3 has hrswk (hours worked per week) and age.

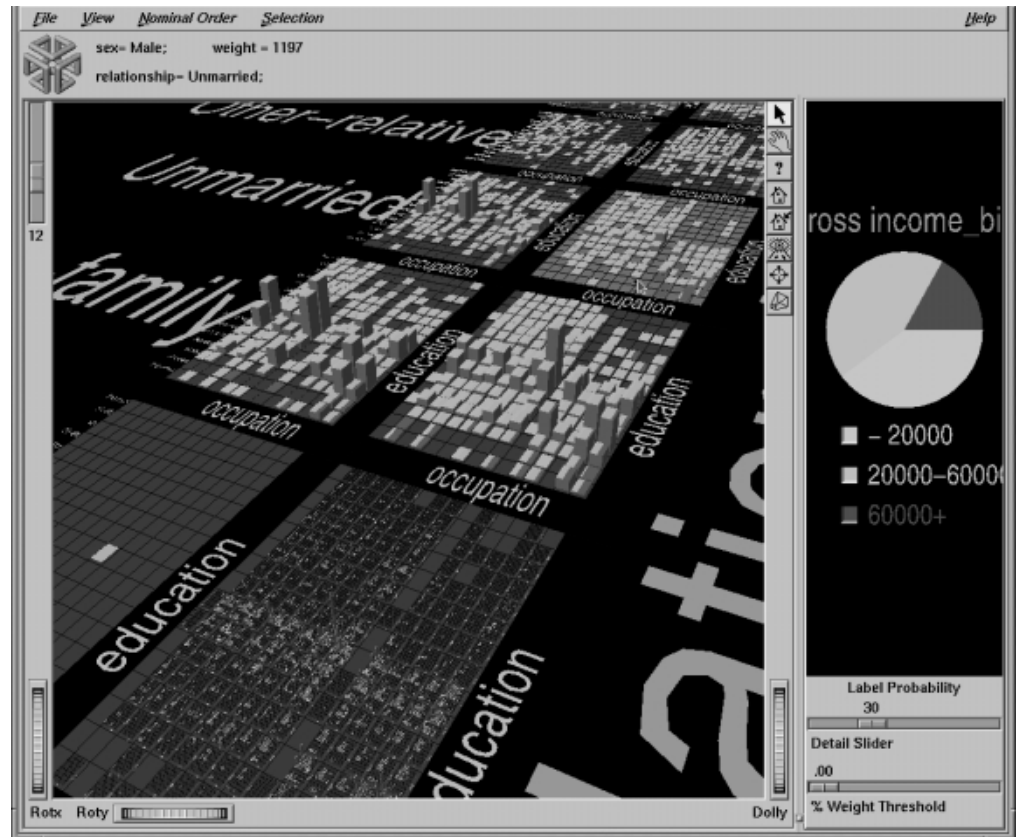


Figure A-2 Decision Table Visualizer Using the Adult Dataset

At the top level we know “relationship” and “sex” have a strong correlation. Of course we expect all husbands to be male, and wives to be female, but we can see right away this is not the case. By picking on the cake for male wives we see that there are 3 of them and all their salaries fall in the 20,000-60,000 range. Drilling down on these cakes reveals more information about these anomalous records. You may wish to select these cakes (using the left mouse button) and drill through to the underlying data so you can find the values of all the other fields.

Click the right mouse on the background; this will drill down globally to the next level. Every cake is now replaced with a matrix for every combination of education and occupation. The ordering is the same for every matrix, and overall the ordering is by correlation with income. If you choose Nominal Order > By Weight from the pulldown menu, the overall ordering will be by record weights. The most prevalent occupations and education levels will appear in the lower left corner of each matrix. “High school grads” is the most prevalent education level, and “Professional-specialty” is the most common occupation, but there are not many high school graduates whose occupation is professional specialty.

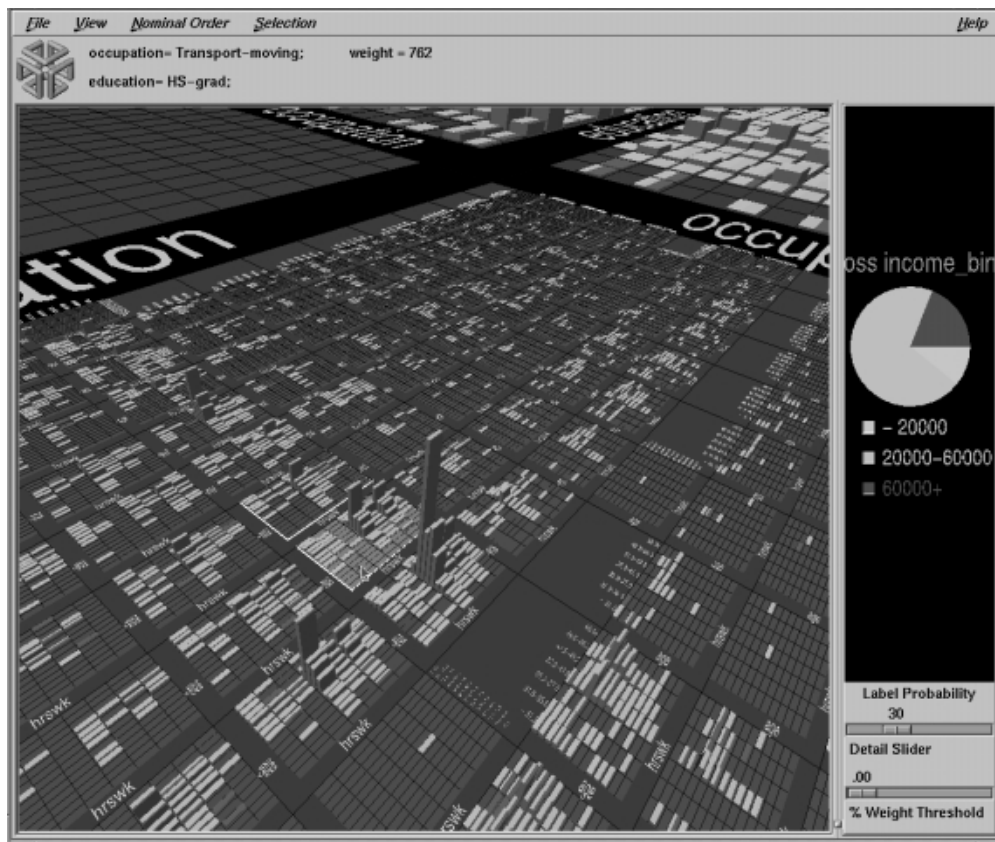


Figure A-3 Closer Inspection of the Adult Dataset

Returning to ordering nominals by income, you can see distinct distributions for each combination of sex and relationship. There is not much difference between males and females whose relationship is not-in-family. The difference between unmarried males and females, however, is very pronounced. (See Figure A-3). There is a very distinctive cluster of red cakes in the lower left of the male matrix that does not exist in the female matrix. By scaling up the heights somewhat, you will notice that the female matrix has obvious spikes at occupation = "admin clerical" and "other service." No such spikes are visible in the corresponding male matrix.

Click with the right mouse button on the most populous cake (male husbands). This operation may take a minute to perform because the visualizer needs to construct all the geometry for the next level for this cake. The geometry is constructed on demand because the time needed to create it all at the beginning would be excessive, and wasteful since the user rarely explores many of the high detail regions. If you right click on the background by mistake you may be forced to wait a very long time if the amount of detail at the next level is very long - as it is in this case. If the drill-down will take a long time, a progress bar with a cancel button is displayed.

Consider the many age by hrswk distributions that are displayed for every combination of occupation and education. The first surprising fact is that, in spite of the many hundreds of cakes shown, there is a single spike at that accounts for 2.5% of all husbands! (See Figure A-3) If you had to pick characteristics for a typical husband, it would be reasonable to say he is a HS-grad doing craft-repair, aged between 41 and 59 and working between 38 and 41 hours a week.

Compare the salary distributions for husbands who are HS-grads in sales with those who are HS-grads and executive managers. Although the distribution of age and hours worked is similar, the probability of being in the income greater than 60,000 class is 34% for this group of managers, compared with 27% for the salesmen. To see these probabilities shown at the top, first click the button next to the 60000+ income class in the label probability pane on the right, then pick cakes on the left.

Iris Classification

In this dataset, each record describes four characteristics of iris flowers: petal width, petal length, sepal width, and sepal length. Each iris was further classified into the types iris-setosa, iris-versicolor, or iris-virginica. The goal is to understand what characterizes each iris type.

Before running a classifier, click the Column Importance tab in the Tool Manager's Classifiers tab; then click Suggest then Go. You obtain a ranking of the importance of the features: petal width, petal length, and sepal length. You can map these to the axes in the Scatter Visualizer, with the iris type mapped to the color and see the clusters.

The file *iris-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *iris.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\iris-dtab.dtableviz* and *\data\iris.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/iris-dtab.dtableviz* and */usr/lib/MineSet/data/iris.schema*

In the Decision Table Visualizer, we can see that petal width is an excellent discriminatory attribute. When you add sepal width you see that all instances of iris versicolor appear in the "sepal width < 3.05" bin for those records which have petal width of between 0.75 and 1.65.

Drill down on the three cakes which are not 100% pure. The top two cakes each contain a single instance of iris-versicolor which prevent them from being pure. For the "sepal width < 3.05" cake it is very difficult to isolate the anomalous iris-versicolor. For that particular cake, however, the iris versicolor is isolated by using petal length.

Mushroom Classification

The file *mushroom-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *mushroom.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\mushroom-dtab.dtableviz* and *\data\mushroom.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/mushroom-dtab.dtableviz* and */usr/lib/MineSet/data/mushroom.schema*

The goal is to understand which mushrooms are edible and which ones are poisonous, given this dataset. There are over 8000 records in this set; thus, running this inducer might take several minutes. Note that under the default mode of the one-third holdout for accuracy estimation, a third of the records are kept for testing.

Each mushroom has many characteristics, including cap-color, bruises, and odor. In the Decision Table Visualizer odor and stalk-shape appear at the top level. Note that odor alone does an excellent job of discriminating edibility. Only when there is no odor and the stalk-shape is “enlarging” is there any ambiguity. So naturally we drill down on this lone cake. Now we see just the records with these 2 values broken down by their values for bruises and gill-size. Notice the interaction between gill-size and bruises. This interaction is difficult to discern using any other classifier.

Since all the attributes in this dataset are nominal, all the values are sorted by how well they predict edibility. You might want to order the values alphabetically or by weight (prevalence). To do this, select the appropriate method from the nominal order menu. If you considered either bruises or gill_size alone you would not be able to predict large classes of completely edible or poisonous mushrooms, but by considering them together, we see that if there are no bruises and the gill-size is broad, then all 814 mushrooms of this type are edible. Conversely, if there are bruises and the gill-size is narrow, then all 11 mushrooms of this type are poisonous. To disambiguate the other two cases we would have to drill down further.

In the Decision Table Visualizer, move the % Weight Threshold slider to the right. Eventually those with musty odor will be deleted from the scene. The reason for this is that there are fewer than 1% of the records labeled “odor=musty.”

Party Affiliation

This dataset consists of voting records. The goal is to identify the party to which a congress person belongs given data about key votes. The dataset includes votes for each member of the U.S. House of Representatives on the 16 key votes identified by the *Congressional Quarterly Almanac* (CQA). The CQA lists nine types of votes: voted for, paired for, and announced for (these three are simplified to yes), voted against, paired against, and announced against (these three are simplified to no), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three are simplified to an unknown disposition).

Before running a classifier, look at the 16 votes to see if you can perceive which features are important. Then run the Decision Table Visualizer. For this dataset, you may wish to order the values alphabetically, so that all “yes” votes appear in the upper right and “no” votes appear in the lower left.

At the top level we see “synfuels corporation cutback” and “physician fee freeze.” There is a fascinating relationship between these variables that would be next to impossible to point out with any other model. All but three who voted against physicians was a democrat. Nearly every democrat that voted against physicians also voted against the synfuels cutback (only 3 of 206 did not fit this pattern). Surprisingly, all but five republicans that voted for the physicians, voted against the synfuels cutback. This odd relationship between these very different issues hints that these bills may have been connected in ways that would require further investigation.

Most of the cakes at the top level are nearly pure except for the middle one (which contains only 6 records) and the one where the representatives voted yes on both issues. Here we can drill-down another level to discriminate the political affiliation of the representatives in this group. Doing so uses “anti-satellite test ban” and “adoption of the budget resolution” to further discriminate among the 55 representatives in this group.

The file *vote-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *vote.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\vote-dtab.dtableviz* and *\data\vote.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/vote-dtab.dtableviz* and */usr/lib/MineSet/data/vote.schema*

Breast Cancer Diagnosis

The breast cancer dataset contains information about women undergoing breast cancer diagnosis. Each record represents a patient with attributes such as cell size, clump thickness, and marginal adhesion. The final attribute is whether the diagnosis is malignant or benign. The file *breast-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *breast.schema*.

- Windows users find these files in the directory in which MineSet was installed under *\examples\breast-dtab.dtableviz* and *\data\breast.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/breast-dtab.dtableviz* and */usr/lib/MineSet/data/breast.schema*

In the Decision Table Visualizer, mitosis and uniformity of cell shape are shown at the top of the hierarchy. If both attributes have low values at the same time the given sample is 99.2% likely to be benign. On the other hand, 100% of the training records that had high values for both, were malignant.

Drill-down on the four cakes that are not so pure. Now marginal adhesion and bare-nuclei are used to discriminate. There are far fewer records in each cake at this level; as a result there is more noise, and trends are more difficult to detect. High values for both marginal-adhesion and bare-nuclei seem to contribute to malignancy, but its uncertain. Note that the first value of bare-nuclei is null. The distributions for these null cakes are more suspect than others so you may wish to hide them by unchecking View > Show Nulls.

If you drill down globally two more levels, you can note a few interesting features. The cakes get very small, and there are large regions of the multi-dimensional space which are empty. There are a few tiny regions where many records are clustered. There is one huge spike (100% benign) where all the values are low. This spike alone accounts for about 20% of the data.

Hypothyroid Diagnosis

The hypothyroidism dataset is similar to the one for breast cancer. The file *hypothyroid-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *hypothyroid.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\hypothyroid-dtab.dtableviz` and `\data\hypothyroid.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/dtableviz/hypothyroid-dtab.dtableviz` and `/usr/lib/MineSet/data/hypothyroid.schema`

There are 3,163 records in this dataset and most of them do not have hypothyroidism (95.45%). This means that one can predict “negative” and be correct most of the time. However, we are worried about those people that have hypothyroidism, yet the model predicts to be healthy. The false negatives are very important.

This is a case where you might want to adjust the loss matrix to skew the posterior probability toward predicting hypothyroidism in order to avoid false negatives. There might be a high cost associated with predicting that someone is healthy when they actually have the disease; predicting them sick when they are actually healthy means they merely have to take a more accurate test or a treatment they do not need.

Using the Decision Table Visualizer on this dataset, we note:

- If *fti* is null then *tbg* is not null and *t3* is almost always null (drilling down-one more level).
- Except for two instances, the only time *t4u* is null is when *fti* is null.
- Hypothyroidism is more prevalent for lower values of *fti*.

Pima Diabetes Diagnosis

This dataset is a diagnosis problem for diabetes using statistics gathered from an Indian tribe in Phoenix Arizona. The task is to determine whether a patient has diabetes, given some medical attributes, such as blood pressure, body mass, glucose level, and age.

The file *pima-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *pima.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\pima-dtab.dtableviz* and *\data\pima.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/pima-dtab.dtableviz* and */usr/lib/MineSet/data/pima.schema*

Using the Decision Table Visualizer we note:

- At the second level of detail you can see that few women under 28 have more than six pregnancies (only four do). You may wish to drill-through to these records to get all the information available for these four.
- High values of plasma glucose, body mass, indicate a greater likelihood of diabetes.

DNA Boundaries

The file *dna-dtab.dtableviz* shows the structure of the Decision Table Classifier induced for this problem. This file was generated by running the inducer on *dna.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\dna-dtab.dtableviz* and *\data\dna.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/dtableviz/dna-dtab.dtableviz* and */usr/lib/MineSet/data/dna.schema*

There are 3,186 records in this DNA dataset. The domain is drawn from the field of molecular biology. Splice junctions are points on a DNA sequence at which “superfluous” DNA is removed during protein creation. The task is to recognize exon/intron boundaries, referred to as EI sites; intron/exon boundaries, referred to as IE sites; or neither. The IE borders are referred to as “acceptors” and the EI borders are “donors.” The records were originally taken from GenBank 64.1 (*genbank.bio.net*). The attributes provide a window of 60 nucleotides. The classification is the middle point of the window, thus providing 30 nucleotides at each side of the junction.

From the Decision Table Visualizer, you can see a surprising pattern not nearly as evident in any other classifier model. At the top level there is a pronounced interaction between *left_01* and *right_02*. Exon/intron is only present if *right_02* = T. Intron/exon is only present if *left_01* = G. For other values of *left_01* and *right_01* there are very few splice junctions.

Drill down globally to the next level (*left_02* and *right_01*). Among the records where *right_02* = T and *left_01* = G we see a pattern which is consistent with the patterns along each edge.

Evidence Visualizer Sample Files

The following examples show cases in which classifiers might be useful. Each of these examples is associated with a sample dataset provided with MineSet. By running the inducer, you can generate the *.eviviz* files described below.

The data files can be loaded into MineSet by opening the corresponding *.schema* file from the *data* directory, (for example *churn.schema*). The classifier visualization files, which have a *.eviviz* extension, can be opened from the *examples* directory.

Churn

Churn is when a customer leaves one company for another. This example shows what causes customer churn for a telephone company.

The files *churn.schema* and *churn.data* were used to generate this example. To load a data file into MineSet, open the *.schema* file.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\churn.data` and `\data\churn.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/churn.data` and `/usr/lib/MineSet/data/churn.schema`

The file *churn.eviviz* shows the structure of the classifier induced using the attribute *churned* as the label. The error rate for this classifier is 12%. 14.1% of the records represent customers who churned. The two most important attributes, total day minutes and total day charge, are clearly correlated. If you run the inducer after selecting Automatic Feature Selection from the Further Inducer Options, the error-rate drops to 10.5% using only 4 attributes (total day charge, number of service calls, voice mail plan, and number of voice mail messages). All 29 customers who had a total day charge above 53.78 churned.

A high number of customer service calls is a predictor of churn. Many customer service calls might indicate frustration in using a complicated equipment or receiving unreliable service. Customers with the International plan are also more likely to churn. The people in some states were much more likely to churn than those in others; for example, California and New Jersey have the most churn, Virginia the least. To see just those states that have more than 2% of the total number of records, slide the % Weights Threshold slider all the way to the right. This eliminates most of the values for state from the display. If you also select Nominal Order > Weight, then the state with the most records, West Virginia (WV), is left-most. Many of the attributes (at the bottom of the list) are not useful in discriminating churn. Note that day charge is a great predictor, but night charge is not.

Origin of Cars

The *cars* dataset contains information about different models of cars from the 1970s and early 1980s. Attributes include *weight*, *acceleration*, and miles per gallon (*mpg*). The file *cars.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *cars.schema* with the label set to origin (Japan, U.S., Europe) and the cylinders column changed to type string. The cylinders were changed to type string in order to see all values and avoid the automatic discretization.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\cars-eviviz` and `\data\cars.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/cars.eviviz` and `/usr/lib/MineSet/data/cars.schema`

If you have a dataset of car attributes, you might want to know what characterizes cars of different origins. From the distribution of label values in the pie on the right we can see that most cars in this dataset were made in the U.S. (62.5%) and a smaller number in Japan (20.2%) and Europe (17.3%). Clearly brand is the best predictor of origin, since each brand is associated with only one country of origin. For this reason, it has the highest importance and is at the top of the list. By looking at the height of the pies, it can be seen that many cars have four cylinders, most weigh less than 3000 lbs and most can reach 60 miles per hour in less than 20 seconds but more than 13.

Look at the distribution of slices for individual attribute values. If a car has an engine size >169 cubic inches, it is almost certainly made in the U.S.; it certainly was not made in Japan. Other pies show that U.S. cars generally have six or eight cylinders, low miles per gallon, high horsepower (over 134), heavy weight (over 2981 lbs), and fast acceleration. Japanese cars have better gas mileage, three or four cylinders (and a few six cylinders), and smaller engines. If you click “Europe” in the Label Probability Pane, you can see bars representing evidence for a car being European. For example, five cylinders strongly indicates that a car is European. The height of the corresponding pie, however, shows that there were only three cars with five cylinders in the data. If a car’s mileage is good, there is much evidence for it being European. If a car’s mileage is less than 41, then there is an 83% chance that it’s European. If a car is European, there is only a 10.4% chance that its mileage is better than 41 mpg. But only 2% of Japanese cars—and no U.S. cars—have mpg in this range, so Europe gets the most evidence.

Suppose you wanted to predict where a car came from knowing only that it got 40 mpg and weighed 3000 lbs. Select the appropriate pies (or bars): mpg=30.95-41.15 and weightlbs=2981.5+. The resulting probability distribution on the right shows 84% U.S., 16% European. There is no possibility it is Japanese because there were no Japanese cars in the training set with weightlbs>2981.5. If you run the inducer again with Laplace correction turned on (with a value of .5), you get a different answer: 16% chance for European, 82% chance for U.S., and a 2% chance for Japanese. This is because Laplace correction prevents any slice in the cake charts from going completely to zero. Certainly, there is no fundamental reason why the Japanese could not make a car that weighs more than 2981lbs; hence, when the probabilities (pies) are multiplied together, the possibility of predicting a Japanese car is not eliminated.

Predicting Gender

The *adult* dataset contains information about working adults. This dataset was extracted from the U.S. Census Bureau. It contains data about people older than 16, with a gross income of more than \$100 per year who work at least one hour a week. You might want to know how to characterize males and females. The file *adult-sex.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *adult.schema*, with the label set to sex, after removing the relationship column (which would have made the classifier trivial).

- Windows users find these files in the directory in which MineSet was installed, under `\examples\adult-sex.eviviz` and `\data\adult-sex.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/adult-sex.eviviz` and `/usr/lib/MineSet/data/adult.schema`

In the Evidence Visualizer, the Label Probability Pane shows that the prior probability of working males is higher than that of females.

- Marital status is the most important predictor of gender. If a worker is a married-civilian-spouse there is a greater probability of being male. A worker who is widowed and working, however, is much more likely to be female.
- The second attribute listed shows occupation. Study this to learn which occupations are popular with a particular gender. The various occupations are listed from left to right in order of decreasing male dominance: Armed forces (100%), Craft-repair (95%), Transport-moving (95%), and Farming-fishing (94%). Female trades are Private-house-service (94%) and Adm-clerical (67%). By clicking on the button next to "Female" in the Label Probability Pane, and then moving the mouse over occupation = Adm-clerical, one can see that 23% of females have an Adm-clerical

job. Conversely, given that one's job is Adm-clerical, there is a 67% chance that the gender is Female.

Suppose you wanted to find out the probability of being female given that a person is widowed and has occupation = Adm-clerical. This can be done by clicking on the values and reading 95% from the text at the top when you move the mouse over the box next to "Female" (in select mode).

- If the working class is either self-employed-incorporated or self-employed-not-incorporated, the probability that the person is a male is higher. Conversely, if the working class is state-gov, the conditional probability that the person is a female is higher, but the posterior probability (after taking into account the prior probability) is not higher (click it and look at the posterior probability on the right). The size of the female slice increased by selecting state-gov, but not so much that it would lead you to predict that a person was female, given only that they worked for the state.

By rotating the view, you can see that most people work in private industry by looking at the height of the charts.

- By looking at the gross-income attribute, you can see that the higher the income range, the higher the probability of being male.
- Education generally does not indicate much about gender, except for doctorate degrees, where you are more likely to find males.
- Different occupations have different distributions for males and females.
- The race attribute shows that African-Americans have a higher percentage of females working than the percentage of other races in the conditional probability. Click the value to see that the posterior is about equal between males and females.
- Males in this dataset work more hours per week than do females.

Salary Factors

If you have a dataset of working adults, you might want to find out what factors affect salary. First bin gross_income into five bins, with thresholds at 10,000, 20,000, 30,000, and 60,000. Each record then has an attribute with one of five values. You can run a MineSet classifier to help determine what factors influence salary. The file *adult-salary.eviz* shows the Evidence classifier induced for this problem. This file was generated by running the inducer on *adult.schema* with gross_income divided into five bins using user-specified thresholds.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\adult-salary.eviviz` and `\data\adult.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/adult-salary.eviviz` and `/usr/lib/MineSet/data/adult.schema`

The attributes in the Evidence Visualizer are ranked by importance; thus, relationship, marital status, age, occupation, education, hours per week, and sex are considered most important. Since the label is numeric, a continuous spectrum is used to assign colors to each class. Red is assigned to the highest bin (60,000+). The class labels are listed in the Label Probability Pane according to slice size. As you click on values in the Main Window, the order of the class labels changes to keep the label for the largest predicted class at the top.

- *Relationship* shows that husbands and wives are likely to make more money than unmarried workers or workers not in a family. Wives have slightly higher incomes than husbands.
- *Marital status* shows that most people are married (the second chart from the left is tall). Married workers earn more money than unmarried people.
- *Age* shows that age is a crucial factor. Until the age of 61, when many people retire, the probability of making over \$50,000 increases as workers get older.
- Different occupations yield different probabilities. Executive and professional jobs raise the evidence for making over \$60,000 per year.
- *Education* is an important factor. When considering just education, the highest evidence for earning over \$60,000 is given to workers whose educational level includes a masters or doctoral degree, or matriculation from professional schools.
- Hours per week show that the more hours worked, the higher the evidence for earning more money.
- *Sex* shows that being a female gives evidence for making less than \$60,000 per year.
- Adjust the *Percent Weights* slider to remove values of `native_country`, education and occupation with low weights are removed.

Iris Classification

In this dataset, each record describes four characteristics of iris flowers: petal width, petal length, sepal width, and sepal length. Each iris was further classified into the types *iris-setosa*, *iris-versicolor*, or *iris-virginica*. The goal is to understand what characterizes each iris type.

Before running a classifier, click the Column Importance tab in the Tool Manager's Classifiers tab; then click *Go*. You obtain a ranking of the importance of the features: petal width, petal length, and sepal length. You can map these to the axes in the Scatter Visualizer, with the *iris_type* mapped to the color and see a natural clustering.

The file *iris.eviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *iris.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\iris.eviz* and *\data\iris.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/eviz/iris.eviz* and */usr/lib/MineSet/data/iris.schema*

In the Evidence Visualizer, we can see that petal length and petal width are excellent discriminatory attributes, while sepal length and sepal width are not as good. Move the importance threshold slider to the right to see that the sepal-based attributes disappear first.

Mushroom Classification

The file *mushroom.eviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *mushroom.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\mushroom.eviz* and *\data\mushroom.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/eviz/mushroom.eviz* and */usr/lib/MineSet/data/mushroom.schema*

The goal is to understand which mushrooms are edible and which ones are poisonous, given this dataset. There are over 8000 records in this set; thus, running this inducer might take several seconds. Note that under the default mode of the one-third holdout for accuracy estimation, a third of the records are kept for testing.

Each mushroom has many characteristics, including cap color, bruises, and odor. By default the Evidence Visualizer orders attributes by importance (that is, usefulness in predicting the label). Odor and spore print color appear at the top of the list because the distributions in the cake charts is most different from value to value for these attributes. Since all the attributes in this dataset are nominal, all the values are sorted from left to right by how well they predict edibility. You might want to order the values alphabetically or by weight (prevalence). To do this, select the appropriate method from the nominal order menu. You can see a characterization of poisonous mushrooms by changing the pointer to an arrow (click the arrow icon at the top right of the main screen or press the Esc key), then clicking the button by that class label in the right pane. High bars are associated with values that indicate the mushrooms are poisonous.

In the Evidence Visualizer, move the Detail slider to the right. The attributes with the lowest importance are removed from the scene. The most important attribute by far is odor, as its importance is 92; all other attributes have importance less than 48. Almost all values are good discriminators, but if there is no odor (none), then there is a mix of both classes. The Evidence Visualizer lets you see specific values that might be critical, even if the attribute itself is not always important. For example, `stalk_color_below_ring` is not a good discriminatory attribute because most of the time it takes on the value white. White offers no predictive power because there are equal amounts of edible and poisonous mushrooms with this value. When `stalk_color_below_ring` takes the value gray or buff, it provides excellent discrimination, but there are very few mushrooms with these values.

Party Affiliation

This dataset consists of voting records. The goal is to identify the party a congress person belongs to given data about key votes. The dataset includes votes for each member of the U.S. House of Representatives on the 16 key votes identified by the *Congressional Quarterly Almanac* (CQA). The CQA lists nine types of votes: voted for, paired for, and announced for (these three are simplified to yes), voted against, paired against, and announced against (these three are simplified to no), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three are simplified to an unknown disposition).

Before running a classifier, look at the 16 votes to see if you can perceive which features are important. Then run the Evidence Visualizer. For this dataset, you might want to order the values alphabetically, so that all no votes are on the left, undecided is in the middle, and yes is on the right.

Some issues clearly define one's party affiliation. Democrats tended to vote for a physician fee freeze and aid for El Salvador, while Republicans voted for adoption of a budget resolution and aid to the Contras in Nicaragua.

Immigration was an issue not split along party lines; nevertheless, politicians had strong positions on it because only 7 out of the 235 were undecided on this issue.

The file *vote.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *vote.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\vote.eviviz* and *\data\vote.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/eviviz/vote.eviviz* and */usr/lib/MineSet/data/vote.schema*

Breast Cancer Diagnosis

The breast cancer dataset contains information about women undergoing breast cancer diagnosis. Each record represents a patient with attributes such as cell size, clump thickness, and marginal adhesion. The final attribute is whether the diagnosis is malignant or benign. The file *breast.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *breast.schema*.

- Windows users find these files in the directory in which MineSet was installed, under *\examples\breast.eviviz* and *\data\breast.schema*
- IRIX users find these files in */usr/lib/MineSet/examples/eviviz/breast.eviviz* and */usr/lib/MineSet/data/breast.schema*

In the Evidence Visualizer, you can see that *sample_code_number* was discretized into one range that is equally split, meaning that it does not indicate whether the breast cancer is benign or malignant.

Hypothyroid Diagnosis

The hypothyroidism dataset is similar to the one for breast cancer. The file *hypothyroid.eviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *hypothyroid.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\hypothyroid.eviz` and `\data\hypothyroid.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviz/hypothyroid.eviz` and `/usr/lib/MineSet/data/hypothyroid.schema`

There are 3,163 records in this dataset and most of them do not have hypothyroidism (95.45%). This means that one can predict “negative” and be correct most of the time. However, we are worried about those people that have hypothyroidism, yet the model predicts to be healthy. The false negatives are very important.

Look at the cake chart for *tsh* between 6.35 and 27.5. It shows much evidence for hypothyroidism. When you click on it, however, the posterior probability pie on the right still predicts “negative” because the prior probability for “negative” was so great.

This is a case where you might want to adjust the Loss Matrix to skew the posterior probability toward predicting hypothyroidism in order to avoid false negatives. There might be a high cost associated with predicting that someone is healthy when they actually have the disease; predicting them sick when they are actually healthy means they take a more accurate test or a treatment they do not need.

In the Evidence Visualizer, you can see that *fti* is very important. The first two ranges (besides the unknown) give a lot of evidence for hypothyroidism.

Pima Diabetes Diagnosis

This dataset is a diagnosis problem for diabetes using statistics gathered from an Indian tribe in Phoenix Arizona. The task is to determine whether a patient has diabetes, given some medical attributes, such as blood pressure, body mass, glucose level, and age.

The file *pima.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *pima.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\pima.eviviz` and `\data\pima.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/pima.eviviz` and `/usr/lib/MineSet/data/pima.schema`

In the Evidence Visualizer, you can see that many attributes are irrelevant by themselves. As plasma_glucose increases, the probability of having diabetes increases. The number of pregnancies is also a good indicator when it is high (above 6), as is age (above 27).

DNA Boundaries

The file *dna.eviviz* shows the structure of the Evidence Classifier induced for this problem. This file was generated by running the inducer on *dna.schema*.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\dna.eviviz` and `\data\dna.schema`
- IRIX users find these files in `/usr/lib/MineSet/examples/eviviz/dna.eviviz` and `/usr/lib/MineSet/data/dna.schema`

There are 3,186 records in this DNA dataset. The domain is drawn from the field of molecular biology. Splice junctions are points on a DNA sequence at which “superfluous” DNA is removed during protein creation. The task is to recognize exon/intron boundaries, referred to as EI sites; intron/exon boundaries, referred to as IE sites; or neither. The IE borders are referred to as “acceptors” and the EI borders are “donors.” The records were originally taken from GenBank 64.1 (*genbank.bio.net*). The attributes provide a window of 60 nucleotides. The classification is the middle point of the window, thus providing 30 nucleotides at each side of the junction.

From the Evidence Visualizer, you can see that attributes near the center are chosen as very important. Attributes further away from the splice junction are less important.

If you click and select the charts in the left pane corresponding to “left_01: G” and “left_02: A”, then the pie chart in the label probability pane on the right will change to show the probability distribution of each class as predicted by the evidence classifier. Given these two values, the pie chart shows that the evidence model built assigns the highest probability to “intron/exon”, followed by “exon/intron” and “none”.

The accuracy improves slightly if you invoke automatic feature selection, although running time increases dramatically (sometimes hours). In such cases, run feature selection once, and continue mining only with the chosen features.

Map Visualizer Sample Files

The provided sample configuration and data files demonstrate the Map Visualizer's features and capabilities.

- Windows users find these files in the directory in which MineSet was installed, under `\examples\mapviz`. The `.gfx` and `.hierarchy` files can be found in `\config\mapviz`.
- IRIX users find these files in `/usr/lib/MineSet/examples/mapviz`. The `.hierarchy` and `.gfx` can be found in `/usr/lib/MineSet/mapviz/gfx_files`.

- *blocks.mapviz, blocks.data, blocks.gfx, and blocks.hierarchy*

This simple example shows four adjacent blocks. The height and color of each block varies based on the underlying data in `blocks.data`. You can drill up using the middle mouse button (see the section) to see the upper pair and the lower pair of blocks aggregate; then drill up again to see these upper and lower blocks aggregate into a single block. You can drill down using the right mouse button to see the objects of finer granularity reappear.

- *population.australia.mapviz, population.australia.data, australia.states.gfx, and australia.states.hierarchy*

The data file contains one row for each Australian state and territory. Each row contains three tab-separated items: a keyword name for the state or territory, the population value, and the size of the territory.

This sample graphically displays the 1991 population and population density of the Australian states and territories. Heights of the graphical objects represent the relative population; color represents the relative population density. A legend at the bottom of the display describes the color range and the associated values.

- *population.canada.mapviz, population.canada.data, canada.provinces.gfx, and canada.provinces.hierarchy*

The data file contains one row for each Canadian province and territory. In this example, each row contains 13 blank-separated values (one for each decade between 1871 and 1991).

This sample graphically displays the population and population density of the Canadian provinces and territories from 1871 to 1991, in 10-year increments. The animation control panel lets you dynamically view the datasets across a range of time. Animation operation is explained in “Animation Control Panel” on page 9.

- *population.europe.mapviz*, *population.europe.data*, *europe.countries.hierarchy*, and *europe.countries.gfx*
When graphically displayed, this shows the 1992 population and population density of countries in Western and Central Europe.
- *population.usa.mapviz*, *population.usa.data*, *usa.state.gfx*, and *usa.state.hierarchy*
When graphically displayed, this shows the population and population density of the United States from 1770 to 1990. The animation controls let you dynamically view population and density changes across time.
- *population.usa.city.mapviz*, *population.usa.city.data*, *usa.state.gfx*, *usa.state.hierarchy*, and *usa.city.gfx* and *usa.city.hierarchy*
The *usa.state.gfx* file specifies the United States, which is displayed as a background. The *usa.city.gfx* file specifies the location of the cities on this background. The *.data* file specifies the population of each city.

This sample graphically displays the population of the 48 largest U.S. cities from 1950 to 1990. No data has been mapped to the colors. The animation controls let you dynamically view changes across time.

- *perhouse.perage.mapviz*, *perhouse.perage.data*, *usa.state.gfx*, and *usa.state.hierarchy*
This sample graphically displays consumer household spending data from July-August 1988 to May-June 1991. Color is mapped to the gender of the spending household member; height represents the average dollar amount spent per household for a given time period and age group. This data has two independent dimensions: time and age. The highest spending is indicated in the summary window by the areas with the greatest color density, namely “May-June 1989 (Age: 30-39)” and “May-June 1990 (Age: 30-39).”
- *telecom.mapviz*, *telecom.data*, *usa.city.lines.gfx*, *usa.city.lines.hierarchy*, *usa.state.gfx*, and *usa.state.hierarchy*
This sample graphically displays a flat map with arched lines on it. These lines connect two endpoints. The lines can have variable width and color. In this example, the widths and colors are random; however, they could relate to the volume and duration of the connections between the endpoints.

- *fasta.m.data*, *fasta.m.mapviz*, *fasta.m.gfx*, and *fasta.m.hierarchy*

The data file for this example contains the partial results of a full biological sequence comparison between two complete genomes (courtesy of Dr. Tom Flores, European Bioinformatics Institute). When graphically displayed, scientists can quickly identify and locate the regions of similarity between the two genomes. The ability to display such large amounts of information in a visual data exploration method such as this could be extended to include much more information about the individual genomes. Scientists could explore this data more easily and thereby perhaps better understand the function and purpose of the similar genetic sequences.

In this example, the “map” is the circular-shaped genome of a biological organism called *Mycoplasma genitalium* (MG). The MG genome is divided into 500 equal segments, each representing a 1000-nucleotide sequence in the genome. The slider selects one of the segments of the second genome, called *Haemophilus influenzae* (HI), for cross-comparison between the two genomes. The Summary Window in the Animation Control Panel indicates which segments show the greatest similarities, and you can move the slider to examine those particular segments of interest. The bar heights and colors on the “map” therefore indicate the relative similarity of each MG segment to each HI segment, where higher bars correspond to greater measures of similarity. This similarity is measured by the “Reciprocal Evaluates,” which ranges from 0.0 to 1.0.

Option Tree Sample Files

The following examples show cases in which the Option Tree inducer can be useful. Each of these examples is associated with a sample data file provided with MineSet. By running the inducer, you can generate the *-odt.treeviz* files described below. The text describing the scenario and goal for each task is described in Tree Visualizer Sample Files. Here we describe the specific advantages and disadvantages of Option Trees for several of the example datasets.

Note: The data files, which have a *.schema* extension, are located in the *data* directory on the client workstation. The classifier visualization files, which have a *-odt.treeviz* extension, reside on the client workstation in the *examples* directory. To load a data file into MineSet, open the *.schema* file.

- Windows users find these files in the directory in which MineSet was installed, under *\examples* and *\data*
- IRIX users find these files in */usr/lib/MineSet/examples/treeviz* and */usr/lib/MineSet/data*

Churn

The Option Tree for this dataset shows that total day charge, total day minutes, and customer service calls are all good attributes for the root: they all have approximately the same estimated error rate. You can choose to fly down to one subtree or another, based on your preferences and understanding of the data. Note that while the right subtree starts with customer service calls, the second test is on total daily charge or total daily minutes (as the root's left option). However, because a split already occurred on an attribute, the thresholds are different.

Origin of Cars

The Option Tree for this dataset shows several good attributes for the root, including: cubic inches, cylinders, weight lbs, mpg, and brand. Note that the root has a lower estimated error rate than any of the children.

Iris Classification

This is an example where Option Trees seem to be performing worse than Decision Trees. The root for the Decision Tree shows 6% error and the root for the Option Tree shows 8% error, so it seems that Option Trees perform worse. However:

- The standard deviation of the error estimate is fairly high: 3.88% and 3.39%. A rule of thumb in statistics is that if the difference is less than two standard deviations, the difference is not statistically significant at the 95% confidence level. A difference of 2% is not larger than even a single standard deviation; hence, the classifier error rates are probably not statistically different at the 95% confidence level

- For small files (Iris has 150 records), different random seeds give different results. For example, changing the seed to 3 improves the Option Tree classifier's error from 8% to 4% without changing the Decision Tree classifier's error rate (remember to reset the seed). This does *not* imply that a more accurate classifier has been generated, rather that the error estimate is not stable. Because only 50 records are used for testing, each mistake is 2%. The difference between 4% and 8% is making two more mistakes.
- For small files (Iris has 150 records), use the "Estimate Error" option in MineSet. It results in better estimates that have narrower confidence intervals. When you run this mode, the status window shows that the Decision Tree classifier has an estimated error of 4.67% +/- 1.73%, and the Option Tree classifier has an estimated error of 4.00% +/- 1.61%. The difference is not significant in this case either, but the Option Tree is slightly superior.
- Even if the error rate is higher for Option Trees, they might be (and usually are) better at assigning probability estimates. For this dataset, the estimated mean squared error for Decision Trees is 3.94; for Option Trees it is 3.67 (although the difference is not significant at the 95% confidence level).

Mushroom Classification

The Option Tree for this dataset shows that all five options chosen at the root have zero error rate estimates. Looking at the result, you might prefer the left option (bruises) because it is as accurate but is easier to measure than odor (the root test of the induced Decision Tree). You might want to remove odor and gill size, then build a regular Decision Tree that turns out to be just as accurate (0% estimated error rate).

Note, however, that removal of a root option to have a sibling option selected by the Decision Tree might not necessarily result in the same accurate classifier that is shown in the Option Tree. The removed attribute might have been used lower down in the tree. For example, removing brand from the cars dataset significantly increases the error rate, even though four out of five options do not use it at the root.

Party Affiliation

This dataset behaves very similarly to the Iris dataset. The Option Tree has the same error rate as the Decision Tree. Under "Estimate error," the cross-validated estimate shows that it is slightly better than the Decision Tree (but not significantly so at the 95% level) both on error rate and on mean squared error.

Breast Cancer Diagnosis

The error rate for Option Trees is slightly lower than that for Decision Trees, both for *Classifier & Error* and for *Estimate Error*; however, the difference is not significant (at 95%).

Hypothyroid Diagnosis

The error rates for this dataset are very low (less than 1%), but this is because most people who were tested for hypothyroid (95%) did not suffer from it. If we use a loss matrix that attempts to avoid false negatives (by penalizing by 100 a prediction of negative when the actual value is hypothyroid), we can see that the loss for Option Trees is significantly lower than that of Decision Trees: 182 versus 523 (total), or 0.17 versus 0.5 (per record). This difference is significant at the 95% confidence level.

DNA Boundaries

For this dataset, the Option Tree is slightly more accurate than the Decision Tree; however, looking at the root options, you might notice that it chooses *left 1,2*, and *right 1,2,5*. Given the background knowledge that attributes closer to the boundary can be more important, you might want to exclude the option split on *right 5*. After updating the maximum number of root options to 4 (down from 5), the error rate increases from 5.65% to 6.59%. This might be surprising, given that the root no longer uses *right 5* as an option; another effect of changing the number of root options from 5 to 4 was to also reduce the number of options that appear further down the tree (because of the decrease parameter). This caused the individual error rates for each of the other 4 subtrees to increase. Still, the option tree's error rate is significantly better (at the 95% confidence level) than the Decision Tree error rate of 7.06% +/- 0.79%.

Regression Tree Sample Files

The following examples show cases in which regression might be useful and highlight some of the capabilities of the Regression Tree Inducer. Each of these examples is associated with a sample data file provided with MineSet. By running the inducer, you can generate the *-rt.regress files* described below.

Note: The data files, which have a *.schema* extension, are located in the *data* directory on the client workstation; and the regressor visualization files, with a *-rt.treeviz* extension, reside on the client workstation in *examples* directory. To load a data file into MineSet, open the *.schema* file.

- Windows users find these files in the directory in which MineSet was installed, under *\examples* and *\data*
- IRIX users find these files in */usr/lib/MineSet/examples/treeviz* and */usr/lib/MineSet/data*

Churn

The *churn* dataset contains generated information on the calling patterns of a telecommunication company's customers. In the classification examples, this dataset is used to determine which factors lead a customer to churn, or leave the company for one of its competitors. In this regression example, we will try to determine what factors influence how much the company charges each customer per day.

The file *churn-rt.treeviz* shows the Regression Tree generated on this data set to predict the total day charge. Interestingly, the tree branches on only one attribute throughout, total day minutes; continuously dividing this attribute further and further into progressively smaller ranges. This is because total day minutes is directly proportional to total day charge—the customers are charged only for the minutes they use the system. The Regression Tree is able to adapt to this fact.

Car Mileage

The *cars* dataset contains information about different models of cars from the 1970s and the early 1980s. Attributes in this data set include weight, acceleration and miles per gallon. The file *cars-rt.treeviz* shows the Regression Tree regressor induced on this data set, using miles per gallon as the continuous label.

By clicking on the top node, we see that the average mpg of cars in this dataset is around 23.5. The first split in the Regression Tree for this dataset shows that the most important factor contributing to the mileage of a car is its weight. The Regression Tree has uncovered the well-known fact that heavier cars get lower mileage. By looking at the two children of the base node, we note that the right child is bluer than the left one, that is it gets fewer miles per gallon. By highlighting the nodes, we see that cars that weigh less than 3018 lbs. get around 28.3 mpg, while cars weighing more get around 16.6 mpg.

Heading over to the heavier cars, we see that the next split is on the horsepower of the car, and that more powerful cars tend to get lower mileage. The split at the next level that is on the year the car was made, with newer cars getting better mileage. Now, let's look for an unusual car. Using the filter panel, let's try and find a node with a mean mpg < 24 but with a maximum > 30 . Doing this filter, we quickly reduce the tree to one node, cars weighing less than 3018 lbs, with more than 77 horsepower, and made before 1980. In this category, there is an unusual car; by selecting the rightmost bar on that node and drilling through via the Selections > Show Original Data menu item, we see that this car is a 1978 Dodge, weighing around 2000 lbs, with 83 hp, but getting a high 33.5 miles per gallon.

Salary Factors

The *adult* dataset contains information about working adults, extracted from the U.S. Census Bureau. It contains data about people older than 16, with a gross income of more than \$100 per year, who work at least one hour a week. We can use the Regression Tree Inducer to determine which factors influence a person's salary; as well as to give a rough prediction of what that person's salary would be, given the other information.

The file *adult-rt.treeviz* shows the Regression Tree regressor induced on the adult dataset, using gross income as the continuous label. Note that this data set is large (around 50000 records), and therefore inducing the regressor on this data set may take a few minutes on your workstation.

The bars at the top node provide a histogram of salary values in the Census Bureau's data. Note that the amount of data available decreases as the salary level increases. We have a lot of data for people earning around \$3,000 a year, but less so as that figure increases. This trend is reversed in the last part of the histogram that indicates a sizeable amount of data on people earning roughly \$100,000 per year. This discrepancy might be the result of either a genuine trend in the data, or a biased sampling.

The first division in the Regression Tree is on the age attribute. As expected, younger people generally make less money than older people. Brushing the top node and its two children nodes, we can see some summary statistics for these three groupings of people. We note that the mean salary for everyone in this study is around \$33,500, while the mean salary of people under 27 is around \$14,300; and the mean salary of those over 27 is around \$40,000.

Following the next two divisions of people under 27, we see that the tree again splits them into two categories: those 23 and under, and those over 23. Interestingly, the split past these two divisions is the same, and on the hours per week attribute, indicating that for both age ranges the more hours worked, the higher one's gross income.

Now, focusing on those over 27, we find that the tree splits immediately on the amount of education a person has had. Those with an education number 13 and over (which corresponds to a bachelor's degree), tend to make more money. By looking over the two children of the education number split, we can see that most of the people making around \$90,000 a year have at least some advanced education.

We can use the filter panel, to quickly locate those categories of people making on average over \$50,000 a year. In the filter panel, select mean > 50000. Top level nodes disappear in this filter, as making that amount of money is a rare occurrence. People with a bachelor's degree who are over 27 fall in this category. By following the left branch of the first split to the end, we find another group of people in this category: married men over 36 years old, who work over 35 hours a week and have a good education (10 years or more).

If we revisit the filter, and look for nodes with an absolute deviation of larger than \$25,000, we can find those people whose economic condition offers the widest variability. The first remaining node in this filter is those people over 27 and with a bachelor's degree. The histogram above this node shows a distribution centered around its mean, but with an unusual number of people making around \$100,000 a year.

Iris

Each record in this dataset describes five characteristics of iris flowers, petal width, petal length, sepal width, sepal length, and iris type. Our goal in this regression is to predict the petal width based on the other characteristics. The file *iris-rt.treeviz* shows the results of the Regression Tree Inducer run on this dataset in order to predict petal width.

Looking at the top node, we see a gap in the petal width values, where no flowers exist. The Regression Tree Inducer splits on this data set first using the petal length variable. If the petal length is less than 2.6, only a restricted set of petal widths seems possible. On the other hand, petal length values greater than 2.6 indicate a more even distribution with larger corresponding petal lengths. The mean petal width for those irises with petal lengths less than 2.6 is 0.24, while the corresponding mean petal width for those with lengths greater than 2.6 is 1.68. Following the large petal length irises, we see that the tree splits again on petal length, this time on the value 4.85. These two consecutive splits on

the same variable point to some kind of restricted functional relationship between these two variables.

Going back to those irises with a petal width less than 2.6, we see that the following split is on the sepal width attribute. Interestingly, the values in this part of the tree seem segregated, with those irises with sepal width less than 3.25 taking on values in three narrow but separated ranges.

Pima

This dataset is a diabetes diagnosis problem using statistics gathered from an Indian tribe in Phoenix, Arizona. The file *pima-rt.treenviz* shows the results of running the Regression Tree Inducer on this dataset, using the plasma glucose level as the predicted continuous variable.

The first split in this tree is on the diabetes indicator, showing that people with diabetes tend to have a higher plasma glucose level than those without (141 versus 110). The next split is the 2-hour serum insulin attribute, where values greater than 125 lead to higher plasma glucose.

The Regression Tree predicts by following the decisions at each of the nodes from the top node, while examining new records. For example a diabetic patient with 2-hour serum insulin of 110 would have a predicted plasma glucose level of 105.

Scatter Visualizer Sample Files

The provided sample data and configuration files demonstrate the Scatter Visualizer's features and capabilities. The following *.data* and *.scatterviz* files are in the *examples* directory. To load a data file into MineSet, open the *.schema* file.

- Windows users find these files in the directory in which MineSet was installed, under *\examples*.
- IRIX users find these files in */usr/lib/MineSet/examples/scatterviz*.

The Scatter Visualizer sample files are as follows:

- *company.data*
This file contains fictitious sales data of several insurance companies in three product categories: life insurance, auto insurance, and home insurance. The data span ten years (in increments of one year) and includes five income brackets (the customer's annual income).
- *company.scatterviz*
This file specifies that the years form one slider dimension and the income brackets form the other slider. Sales of life insurance, auto insurance, and home insurance become the three dimensions in the Scatter Visualizer landscape. The color density in the slider summary window represents the total sales of all companies across all categories of insurance.
- *company-total.scatterviz*
This file contains the same specifications as *company.scatterviz*, except that the size of each company is determined by the total sales of that company across all the categories of insurance.
- *company-life.scatterviz*
This file contains the same specifications as *company.scatterviz*, except that the color of each object indicates the life insurance sales as a fraction of total sales.
- *store-type.data* and *store-type.scatterviz*
These files show sales of various product groups by store type during a three-year period. The single independent variable for which a slider appears is time. Each entity represents a store type (such as Food Store, Drug Store, Service Station, and so forth). For each store type, the data file contains the total sales of several product groups, such as alcoholic beverages, cereal, and so forth. The data spans 36 months, in increments of one month.

The configuration file uses the month as the single slider dimension. One axis is sales of alcoholic beverages, the other is sales of tobacco products. A third axis is not used.
- *brand.data* and *brand.scatterviz*
These files show sales of several soft-drink brands in a variety of store types. In this dataset the brands form the entities, and the store types are associated with the axes. The total sales are mapped to the size of each brand. The color mapping is random. Since there are no independent variables, no slider is present.

- *cars.data* and *cars.scatterviz*
These files show the weight, horsepower, model year, and acceleration of several car models. The axes are cubic inches, mpg, and time to 60. Weight has been mapped to size.
- *people.data* and *people.scatterviz*
These files show the height, weight, density, and cholesterol level for a fictitious population sample.
- *nl.births.data* and *nl.births.scatterviz*
These files show birth patterns in the Netherlands. For each region, the population density, birth rate, and population are shown. The animation sliders are mapped to the age of the mother and the year.
- *adult94.data* and *adult94.scatterviz*
These files show a complex example with scatterviz applied to *adult.data*. The three axes in the visualization are *avg_hrswk* (that is, average hours worked per week), *avg_gross_income*, and *avg_education_num*. Unfortunately “education num” does not correspond exactly to number of years of education, but it is close. The slider on the right side animates across different age ranges. Each aggregate was created by grouping by occupation, race and sex. This means that there is an entity for every combination of values for these three attributes. The color shows different occupations, as shown in the legend. The size of each entity corresponds to record counts. The summary slider is also colored by data density. To find out how this visualization was created, you may select Start Tool Manager from the File menu. This will bring up the Tool Manager with the session used to create this example.

Initially the scene shows information for people under 20 years of age. Note that the average hours worked (about 14) and the average income (about \$4000) are low. If you animate over age using the slider, and examine the scene from the three orthogonal views (try using the lower 3 buttons to the right of the main window), you will notice various trends emerge. For example, if you orient the scene so you see only income by hours per week, you can see that people start to work longer hours as they age, until about age 25, then they seldom work more than 49 hours per week until they retire. Income, however, grows until age 50, then plateaus, then goes lower again. The actual trend depends somewhat on the career choice and other factors.

Suppose you were interested in comparing trends between the occupations craft-repair and prof-specialty. Open the Filter panel (View > Show Filter Panel) and select just “craft-repair” and “prof-specialty” from the list of occupations. Now when you animate, you can see that “prof-specialty” actually starts with lower incomes, but quickly outpaces “craft-repair” as people age. “Prof-specialty” is much higher on the education axis than “craft-repair”. You may wish to limit your filter further by showing just females, or those of a certain race. Also try selecting some of the different motion trail options while animating.

- *census.data* and *census.scatterviz*

These files also show a plot of aggregated census data. The original dataset contained about 150,000 rows. After aggregation, there is a cube (an aggregate) for every combination of education, sex, industry1, and occupations (as these were the group-by columns).

Splat Visualizer Sample Files

The provided sample data and configuration files demonstrate the Splat Visualizer’s features and capabilities. The files are in the *examples* directory.

Windows users find these files in the directory in which MineSet was installed, under *\examples*.

IRIX users find these files in */usr/lib/MineSet/examples/splatviz*.

- *mushroom*

The *mushroom.data* file contains pre-aggregated data concerning more than 5,000 mushrooms. The group by columns were: odor, gill_color, and cap_color. For every combination of these three columns in the original data, there is a count and an average edibility, where 0 is edible, and 1 is poisonous. The average edibility between 0 and 1 means some of the mushrooms in that aggregate are edible and some are poisonous, since mushrooms can not be partially poisonous.

The visualization shows that the unique values for each of these columns have been sorted along the axes according to average edibility. Odor is clearly the best determinant of edibility. Also note that most splats are either all 0 or all 1, meaning these three columns are useful in segmenting the two classes of mushrooms. In fact, the column importance feature was used to select the columns mapped to axes. Lower the opacity slider to determine which splats have the highest counts. The most opaque splat represents 288 mushrooms having common values for odor,

`gill_color`, and `cap_color`. To confirm this try filtering based on `sum_count_poison>280` and picking on the remaining splats to see their counts. Note that all mushrooms with `gill_color=buff` are poisonous.

- *adultJobs*

The *adultJobs.data* file was derived from *adult94*, a dataset provided with the distribution. It was created using an aggregation that grouped by education, occupation, `hours_worked_per_week` (binned), and age (binned). The `gross_income` column was aggregated by count and average. For a display using the Splat Visualizer, `age_bin` was mapped to a slider, while the other group-by columns were mapped to axes. The `count_gross_income` column was mapped to opacity, and `avg_gross_income` was mapped to color.

When the slider is in the left-most position, the color of the plot is almost entirely blue. This means that regardless of occupation, education, or number of hours worked, most people younger than 20 have low incomes. Move the slider to the right, and note how incomes rise faster for higher education and occupations toward the end of the axis. By the opacity variation you can see that the most common types of education are HS, some college and Bachelors degree.

Moving the Summary slider shows how the distribution of income changes with respect to the axis columns as people age.

- *adultJobs2*

The *adultJobs2* file is also based on the *adult94* dataset. Here, the axis columns are `working_class`, education, and occupation. The two columns mapped to sliders are age (binned) and `hours_worked_per_week` (binned). Again, income was aggregated by count and average for use with opacity and color, respectively. Since there are more positions on the 2D slider, there are fewer records represented by each position. This causes greater variation of color and opacity. The red region in the center of the `hrs_per_week` dimension of the Summary slider shows that nearly everyone works between 35 and 45 hours per week. Note that some occupations are aligned with specific working classes. For example, everyone in the Armed-forces has Fed-Government for their working class.

- *censusIncome*

This example is based on a dataset similar to *adult94*, but was not included with the distribution because of its size. In an attempt to understand the differences between gross income and total income, `gross_income`, `total_income`, and `hrs_per_week` have been mapped to axes. Color shows age. By studying the image we can learn that there are many records where `total_income=gross_income`, but there are also a larger portion of records with high `total_income`, but 0 `gross_income`. It is surprising that in many cases `gross_income` is greater than `total_income`.

Note where the people of different ages are concentrated. Many old people (yellow) are in the `hrs_per_wk=0` plane. They are probably retirees. Many children and young adults (blue) are in the line `gross_income=total_income=0`. Note the fairly opaque splats near the outside edges of the volume. These positions include all points that fell in the maximum bin shown for an axis. For example, the highest bin for `total_income` is 70,300+. Any point higher than 70,300 goes in this bin.

To better see the varying density, adjust the opacity slider. At low opacity scales, the diagonal lines show that for most people `gross_income=total_income`, or they have just `total_income` and no `gross_income`. As you raise the scale, you can see that almost the entire volume contains data. This dataset contains 150,000 records.

- *churn*
Churn is when a customer leaves one company for another. This example shows customer churn for a telephone company. The data used to generate this example is *churn.schema*.

Using column importance, we found that `total_day_charge`, `number_customer_service_calls`, and `international_plan` were important discriminators. These columns were mapped to axes. We then created a new numeric column, `churn`, which equals `churned==Yes`, and mapped it to color.

In the resulting visualization, red areas of the volume indicate high churn. The area corresponding to three or more customer service calls and low `total_day` charge corresponds to high churn. You might want to weight big-spending customers more heavily than others. To do this, create a new column, `total_charge`, equal to

```
`total_day_charge`+`total_eve_charge`+`total_night_charge`
```

or some power of this sum. Then map this `total_charge` column to opacity. This means every record is weighted by `total_charge`. Now the visualization shows additional areas of interest near the high end of the `total_day_charge` axis.

Tree Visualizer Sample Files

The provided sample configuration and data files demonstrate the Tree Visualizer's features and capabilities. The following files are in the *examples* directory.

- Windows users find these files in the directory in which MineSet was installed, under *examples*.
- IRIX users find these files in `/usr/lib/MineSet/examples/treeviz` and `/usr/lib/MineSet/data`.

The Tree Visualizer sample files are as follows:

- *store.data* and *store.treeviz*
When graphically displayed, these files show hypothetical sales data for a store chain. The hierarchy includes the entire chain, regions, states, cities, and individual stores. Four products are shown for each level in the hierarchy. In this configuration, heights represent sales in dollars; colors represent the percentage of the target dollar amount.
- *stateRevenue.data* and *stateRevenue.treeviz*
When graphically displayed, these files show the revenue components of every state's budgets for 1992, as obtained from the United States Census Bureau (from <http://www.census.gov/govs/state/stfin92.dat>). Heights represent the dollar amounts in taxes. The descendent nodes in the background show the contribution of various taxes to the total revenues shown in the root node.
- *beer.data* and *beer2.data*, and *beer.treeviz* and *beer2.treeviz*
When graphically displayed, these files show fictitious data based on consumer research of beer purchases. The hierarchy contains three levels:
 1. The first is category (for example, beer or ale).
 2. The second level is brand codes (randomly assigned).
 3. The third is the individual product codes; for example, twelve-pack versus six-pack (randomly assigned).

Each chart contains seven bars, representing seven age groups. Bar height represents the total dollars spent by that age group. Colors represent the percentage of dollars spent by males and females. Brands, products, and data used in these files are samples only.

Both *beer.treeviz* and *beer2.treeviz* produce the same graphical output, but they have been constructed differently. In *beer.treeviz*, each type of beer is represented by a single record, with values for male and for female consumption; these values are stored in an enumerated array.

In *beer2.treeviz*, there are seven records for each beer, with each record representing one age group. Note that in the *beer* file, the age groups are represented in the configuration file; in the *beer2* file, they are included in the data file.

The *beer* file requires less storage space than the *beer2* file; however, the configuration file is a little more complicated. In some cases, it might be easier to produce data in the form used by the *beer2* file.

Index

Symbols

-, 106
% shortest option, 181
* wildcard, 95, 188
? wildcard, 95, 188
[] wildcard, 95, 188

Numbers

2D aggregation, 11
3D landscapes, 155, 177
64-bit support, 133
 systune parameters, 133

A

accuracy
 boosting, 36
accuracy (classifiers), 79
 testing, 99
Add Column
 expression definition language, 2
Add Column button, 1
Add Column option, 62
adultJobs.data, 251
adult-salary.dtableviz, 218
adult-salary.eviviz, 231, 232, 233, 235, 236, 237
adult-salary.schema, 231

adult.schema, 206, 217, 230
adult-sex.dtableviz, 217
adult-sex-dt.treeviz, 206
adult-sex.eviviz, 230
Advanced Mode button, 53
Aggregate button, 3
Aggregate dialog box, 3, 4
Aggregate option, 62
aggregation, 177
 bar heights, 182
 color values, 182
 data points, 157, 158
 options, 4, 5
 two-dimensional, 11
algorithms
 adjusting, 73, 132
Alphabetical command, 70, 128
alphanumeric values
 filtering, 95
 searching for, 188
analyzing
 relationships, 145, 155, 177
analyzing patterns and trends, 19
And operations, 95, 188
animation, 114, 155
animation control panel
 Fast Forward button, 12
 Fast Reverse button, 12
 Forward button, 12
 Loop button, 12

- Path slider, 13
- Play Forward button, 12
- Play-once button, 12
- Play Reverse button, 12
- Reverse button, 12
- Single-step buttons, 12
- Speed slider, 13
- starting animation, 12
- Stop button, 12
- stopping animation, 12
- summary window, 11
- Swing button, 13
- animation control panel (Scatter Visualizer)
 - displaying, 197
 - summary window, 149
- animation control panel (Splat Visualizer), 163
 - summary window, 161
- Animation Flow buttons, 12
- any keyword
 - color values and, 182
- Apply button, 95
- Apply Classifier option, 62
- Apply Model
 - Estimated probability values mode, 15
 - Predict discrete label values mode, 15
- Apply Model button, 14
- Apply Model panel, 15
- arrays
 - geographic locations, 119
 - inducers and, 101, 103
 - sliders and, 149
- ascending sort order, 184
- association rules
 - confidence, 20, 21
 - expected, 21
 - displaying, 27
 - drill through, 27
 - expected confidence, 21
 - generating, 20
 - lift, 21
 - market basket, 20
 - multiway, 24
 - displaying, 25
 - record weighting, 24
 - sample files, 202
 - support, 21
 - minimum threshold, 21
- Association Rules Generator
 - components, 20, 22
 - displaying legends, 27
 - file requirements, 22
 - output, 20
 - overview, 19
- Association Rules Mappings panel, 26
- attaching to servers, 175
- attributes, 37, 77, 99
 - availability, 79
 - discretization algorithm, 55
 - testing, 76
- Attribute Weights in clustering, 45
- Australian maps, 118, 238
- australia.states.gfx, 238
- australia.states.hierarchy, 238
- Automatic column selection option, 86
- automatic discretization algorithm, 55
- Automatic Thresholds
 - Uniform Range, 35
 - Uniform Weight, 35
- avg keyword
 - color values and, 182
- axes
 - display options, 150
 - invisible labels, 161
 - labeling, 151, 161
- Axis Label Size option, 151, 161
- Axis Options
 - No Adjust, 150

Scale Size, 150
Axis options, 150
 Max Size, 150

B

Backfit test set option, 33, 100
backfitting classifiers, 32, 100
backfitting models, 33
Bar Label Color option, 184
bars, 177
 color options, 182
 based on keys, 183
 labels, 184
 mapping to, 182
 fixed, 183
 heights, 180
 aggregating, 182
 labeling
 colors, 184
 negative values and, 177
 scaling, 181
 searching, 187
Base Execute option, 183
Base Heights command, 192
Base Label Color option, 184
bases, 177
 color options, 182
 labels, 184
 lines, 184
 mapping to, 182
 labeling, 184
 scaling, 181
 selecting, 193
beer2.data, 253
beer2.treeviz, 253
beer.data, 253
beer.treeviz, 253

Bin Column button, 33
Bin Columns option, 62
binning, 33
bins, 33
blocks.data, 238
blocks.gfx, 238
blocks.hierarchy, 238
blocks.mapviz, 238
Boosting, 36
brand.data, 248
brand.scatterviz, 248
Breast Cancer Diagnosis dataset, 211, 224, 235, 243
breast.dtableviz, 224
breast-dt.treeviz, 211
breast.eviviz, 235
breast.schema, 224, 235
brushing, 37
budgets, 253
buttons
 Tree Visualizer
 search dialog box, 189

C

canada.provinces.gfx, 238
canada.provinces.hierarchy, 238
Canadian maps, 118, 238
cars.data, 249
cars-dt.treeviz, 205
cars.eviviz, 229
cars.scatterviz, 249
cars.schema, 205, 229
case-insensitive filters, 190
case-insensitive searches, 188
case-sensitive searches, 187

- census database sample files, 202
- censusIncome data file, 251
- Change Types option, 62
- changing colors, 50
- character strings
 - filtering, 95
 - searching, 188
- Check Expression button, 3
- child nodes
 - selecting, 193
- chi-square, 73
- choosing colors, 50, 51
- Churn dataset, 203, 205, 214, 228, 241, 252
- churn-dt.treeviz, 205
- churn.schema, 252
- classes
 - assigning records, 70, 83, 130
- classification types, 80, 81
- Classifier & Error mode, 33, 99, 100, 110
 - viewing output, 102
- Classifier only mode, 99, 102
- Classifier Options dialog box, 72
- classifiers
 - accuracy, 79
 - testing, 99
 - applying to records, 32
 - backfitting, 32, 100
 - column importance and, 55
 - confusion matrices, 100
 - defined, 37
 - generating, 38, 71, 130
 - learning curves, 101, 107
 - options, 108, 109
 - viewing output, 102
 - lift curves, 110
 - loss matrices, 101, 113
 - predicting unknown values, 112
 - record weighting, 101, 137
 - return on investment curves, 100
 - viewing output, 102
- class labels, 99
 - searching, 76
- Clear button
 - search dialog, 187, 189
- Close button
 - search dialogs, 189
- Close command, 92
- clustering algorithms
 - iterative k-means, 42
 - single-k means, 40
- .clusterviz.data files, 46
- Color Aggregation options, 182
- Color Browser, 50
 - opening, 50
- Color by Key option, 183
- Color Choose dialog box, 151, 161
- color editor, 120, 182
- color list, 120, 149, 160, 182
- Color mapping option, 149, 160, 182
- color mappings
 - Scatter Visualizer, 149
 - Splat Visualizer, 160
 - Tree Visualizer
 - null values and, 194
- colors, 182
 - bars, 182
 - based on keys, 183
 - labels, 184
 - bases, 182
 - labels, 184
 - changing, 50
 - disks, 182
 - filling by key, 183
 - grids, 151, 161
 - ground, 184
 - labels

- bars, 184
- bases, 184
- lines, 184
- sky, 184
- splats, 160
- Colors option, 182
- color swatches, 47, 50
- column importance algorithm, 55
- Column Importance tool
 - dependence, 55
 - discrete attributes and, 55
 - importance ranking, 55
 - purity measure, 55
- columns
 - aggregation options, 4, 5
 - computed, 1
 - naming, 2
 - selecting, 52, 54
 - viewing, 61
- Columns to aggregate option, 5
- Columns to remove option, 5
- command-line options, startup
 - association rules, 23
- company.data, 248
- company.scatterviz, 248
- company-total.scatterviz, 248
- comparing
 - datasets, 181
 - strings
 - filtering types and, 95
- Complementary Drill Through command, 191
- computed columns, 1
- conditional probabilities, 84, 85
- confidence, 20, 21
 - expected, 21
- configuration files, 175
 - association rules sample, 202
 - Evidence Visualizer

- loading, 86
- Map Visualizer, 117
 - loading, 117
 - sample, 238, 239
- Scatter Visualizer, 146
 - loading, 147
 - sample, 248
- Splat Visualizer, 159
 - loading, 159
- Tree Visualizer, 177
 - loading, 91, 92, 178
 - sample, 253
- confusion matrices, 100
- confusion matrix, 16
- connections, 175
- Constant command, 166
- consumer research sample files, 253
- consumer spending sample files, 239
- Contains search option, 95, 188
- Continuous color setting, 120, 149, 160, 182
- credit database sample files, 202
- cross-validation classification, 60, 81, 100
- Current Columns window, 61
- current views, 171

D

- database servers
 - connecting to, 175
- .data filename extensions, 93, 116, 146, 159, 177
- data files
 - Map Visualizer, 116, 119
 - sample, 238, 239
 - Option Tree inducer, 131
 - Scatter Visualizer, 146
 - sample, 248
 - Splat Visualizer, 159
 - sample, 250

- Tree Visualizer, 177
 - sample, 253
- Data Files panel, 61
- Data Files tab, 61
- DataMover
 - connecting to, 175
- data points, 13
 - aggregating, 157, 158
- datasets
 - classifying, 70, 83, 130
 - comparing, 181
 - displaying data, 115
 - 3D landscapes, 155, 177
 - drilling through
 - restrictions, 78
 - filtering, 181
 - predictions, 37
 - confusion matrices and, 58
 - sampling, 99
 - saving, 61
 - selecting multiple values, 123
- datasets, hierarchical
 - sample files, 202
- Data Transformations panel, 61
- dates
 - inducers and, 101, 103
- dates, Y2K compliant, 200
- Decision Tree Classifier
 - generating, 71
 - overview, 70
- Decision Tree Inducer
 - adjusting induction algorithm, 73
 - chi-square, 73
 - Column Importance tool and, 55
 - Gini, 73
 - overview, 70
 - pruning methods
 - confidence, 74
 - cost complexity, 74
 - viewing node information, 71
- decision trees
 - displaying, 102
 - error/loss estimates, 71, 76
 - filtering, 75
 - measure of purity, 71, 76
 - nodes
 - viewing information, 71
 - null values and, 77
 - pruning, 74
 - splitting, 73, 167
- Decrease option, 132
- defaults, resetting
 - Map Visualizer, 119
 - Scatter Visualizer, 148
 - Splat Visualizer, 161
 - Tree Visualizer, 178
- Depth slider, 191
- descending sort order, 184
- Diabetes Diagnosis dataset, 212, 226, 236
- disabling progress dialogs, 199
- discrete attributes, 55, 77
- Discrete color setting, 120, 149, 160, 182
- discrete labels, 77
- Discrete Labels menu, 77
- discretization algorithm, 55
- disks
 - color options, 182
 - mapping to, 182
 - heights, 180
- Display confusion matrix option, 100
- displaying
 - animation control panel, 197
 - classifier output, 102
 - data, 115, 155, 177
 - decision tree nodes, 71
 - decision trees, 102
 - entities, 148, 149

- labels, 151, 161
- messages, 123
 - Map Visualizer, 121
 - Scatter Visualizer, 150
 - Tree Visualizer, 183
- selected objects, 123
- splats, 160
- displaying association rules, 27
- displaying data, 11
- Display lift curves option, 110
- Display menu (Tree Visualizer), 192
- display parameters, 192
- Distance Metric in clustering, 45
- DNA Boundaries dataset, 243
- DNA dataset, 213, 227, 237
- dna.dtableviz, 227
- dna.eviviz, 237
- downloading Internet files, 199
- drilling through datasets
 - restrictions, 78
- drill through
 - association rules and, 27
- Drill Through Columns command, 154

E

- editing, 172
 - colors, 50
- Edit matrix button, 113
- Edit Prev. Op. button, 172
- endpoints, 115
 - sample files, 239
- entities
 - displaying, 148, 149
 - labeling, 149
 - null values and, 152
 - selecting, 150
 - size, 148
- Entities File field, 119
- Entity Colors option, 148
- Entity Label Color option, 149
- Entity Label Size option, 149
- Entity Legend On option, 148
- Entity Options option, 148
- Entity Shape option, 149
- Entity Size option, 148
- Equals search option, 95, 188
- Error Estimate mode, 99
 - viewing output, 103
- Error Estimate Options
 - mean absolute error, 142
 - mean square error, 142
- Error Estimation
 - Regression Tree Inducer, 142
- error/loss estimate, 71, 76
- error options (inducers), 101
- error rate
 - boosting accuracy, 36
- Estimated probability values mode, 15
- Estimate Error mode, 60, 100
 - viewing output, 102
- European maps, 118, 239
- europe.countries.gfx, 239
- europe.countries.hierarchy, 239
- Evidence Classifier, 83
- Evidence Inducer
 - Column Importance tool and, 55
- Evidence Pane
 - selecting items, 88
- Evidence Visualizer
 - overview, 83
 - predictions, 84
 - probabilities, 84, 85
 - correcting, 85

- selecting items, 88
- startup options, 87
- Execute option, 121, 151, 183
- execute statements
 - Scatter Visualizer
 - enabling warnings, 199
 - running, 199
- executing UNIX commands, 121, 151, 183
- Exit command, 92
- exiting
 - Tree Visualizer, 92
- expected confidence, 21
- expression definition language, 2

F

- far horizon, 184
- fasta.m.data, 240
- fasta.m.gfx, 240
- fasta.m.hierarchy, 240
- fasta.m.mapviz, 240
- Fast Forward button, 12
- Fast Reverse button, 12
- field names, 2
- File Menu
 - Publish on the Web, 92
- filenames
 - Scatter Visualizer, 146
 - Splat Visualizer, 159
 - Tree Visualizer, 93, 116, 177
- file requirements
 - Map Visualizer, 116
 - Option Tree inducer, 131
 - Scatter Visualizer, 146
 - Splat Visualizer, 159
 - Tree Visualizer, 177
- Filter Button, 94

- Filtering
 - expression definition language, 2
- filtering
 - data, 181
 - decision trees, 75
- Filter option, 62
- Filter Out % Shortest option, 181
- Filter Panel command, 75, 190
- Find File button, 119
- First Child command, 193
- Fit Data to Model, 17
- Fit Data to Model mode, 17
- flat maps, 239
- flat planes, 115
- floating-point numbers, 54
- formats
 - messages, 183
- Forward button, 12
- Further Classifier Options command, 72, 87, 131

G

- Gain Ratio option, 73, 167
- Gaussian command, 166
- Gender attribution dataset, 206, 217, 230
- generating association rules, 20
- geographical objects, 116, 119
- geographic regions, 118
 - legends, 120
 - messages, 121
 - scaling, 120
- Geography File option, 119
- gfx files, 116
 - samples, 238, 239
- Gini, 73
- Go Back command, 193

Go Forward command, 193
 Go menu (Tree Visualizer), 192
 Grid (X, Y, Z) Size option, 151, 161
 Grid Color option, 151, 161
 grids
 color options, 151, 161
 line spacing, 151, 161
 ground colors, 184
 Group-By columns option, 5

H

Height Aggregation option, 182
 Height filter slider, 190
 Help menu
 Tree Visualizer, 193
 Hidden option, 190
 Hide Label Distance option, 151, 161
 hiding
 labels, 151, 161
 hierarchical data
 sample files, 202
 hierarchies, 177
 moving through, 193
 Hierarchy field, 186
 hierarchy files, 116
 samples, 238, 239
 specifying, 119
 Hierarchy option, 77
 highlighting objects
 Scatter Visualizer, 153
 Histogram Visualizer, 97
 trimming factor, 97
 history window, 172
 holdout classification, 80, 100
 holdout ratio, 100

Home command, 192
 home locations, 192
 setting, 192
 Hypothyroid Diagnosis dataset, 211, 225, 236, 243
 hypothyroid.dtableviz, 225, 226, 227, 228
 hypothyroid.eviviz, 236
 hypothyroid.schema, 211, 225, 236

I

Ignore Case In Filters option, 190
 Ignore Case In Searches option, 187, 188
 importance (defined), 84
 importance ranking, 55
 Inducer Options dialog box, 87
 inducers
 class labels, 99
 error options, 101
 running, 99
 setting options, 100
 tracking progress, 102
 induction algorithms
 adjusting, 73, 132
 insurance sample files, 248
 internationalization, 106
 extending to other languages and encodings, 104
 LANG, 104
 locale, 104, 105
 resource files, 104
 resource files example, 105
 resource files, 104
 example, 105
 setting the locale, 104, 105
 Internet files, 199
 invisible labels, 151, 161
 invoking
 Decision Table Inducer, 63

- Scatter Visualizer, 146
- Splat Visualizer
 - resetting defaults and, 161
- Iris classification dataset, 209, 221, 233, 241
- iris.dtableviz, 222, 224
- iris-dt.treeviz, 209
- iris.eviviz, 233
- iris.schema, 209, 233
- Is Null operator, 95, 188
- iterative k-means clustering, 41

K

- keys
 - coloring bars and, 183
- Kind of mapping color option, 182

L

- Label Probability command, 70, 128
- labels
 - axes, 151, 161
 - bars
 - colors, 184
 - bases, 184
 - color options
 - bars, 184
 - bases, 184
 - distance between, 151, 161
 - entities, 149
 - inducers and, 77, 99
 - main windows, 121
 - resizing, 149
 - splats, 161
- landscapes, 155, 177
- LANG, in internationalization, 104
- Laplace correction option, 85

- large memory support, 133
 - system parameters, 133
- Last Child command, 193
- Learning Curve mode, 108
- learning curves, 101, 107, 109
 - options, 108, 109
 - viewing output, 102
- Legend On option, 120
- legends
 - association rules, 27
 - entities, 148
 - geographic regions, 120
 - summary, 149
- Level option, 76
- lift, 21
- lift curve, 16
- lift curves, 110
- Limit tree height to option, 73
- Linear command, 166
- Line Color option, 184
- line colors, 184
- line spacing (grids), 151, 161
- loading files
 - Scatter Visualizer, 146
 - Tree Visualizer, 91, 92, 177
- locale
 - resource files, 104
 - example, 105
- locale, in internationalization, 104, 105
- loading files
 - Cluster Visualizer, 46
- Loop button, 12
- loss matrix, 101, 113

M

- main window

- Statistics Visualizer, 170
- main windows
 - Map Visualizer
 - labeling, 121
- Make Fixed option, 183
- Mapping option, 120
- mappings
 - geographic locations, 119
 - null values and, 152, 194
 - strings, 156
- Map Visualizer
 - data files, 116, 119
 - displaying data, 115
 - file requirements, 116
 - geographic locations, 118
 - main window
 - labeling, 121
 - null values and, 129
 - options
 - resetting, 121
 - saving, 121
 - startup, 117
 - resetting defaults, 119
 - saving defaults, 122
 - startup options, 117
- .mapviz extensions, 117
- market basket analysis, 20
- Mark Flags command, 192
- Matches search option, 95, 188
- mathematical expressions, 2
- Max # root options, 132
- max keyword
 - color values and, 182
- Max/Scale Heights option, 181
- Mean error/loss standard deviation option, 76
- mean squared error, 142
- measure of purity, 71, 76
- Message option, 121, 183
- messages, 123
 - Map Visualizer, 121
 - Scatter Visualizer, 150
 - Tree Visualizer, 183
- MINESET_WARN_EXECUTE variable, 199
- MineSet mtr extension, 200
- Min fitness ratio, 132
- minimum support threshold, 21
- min keyword
 - color values and, 182
- models
 - backfitting, 33
 - loading pre-existing, 14
 - record weighting, 32
 - selecting, 14
- modifying colors, 50
- Move Left command, 193
- Move Right command, 193
- Move Up command, 193
- moving through views, 171
- mtr files, 199
- multiple values, 123
- multiprocessor version, 72, 131, 133, 134
- multiway rules, 24
 - displaying, 25
 - Tool Manager and, 25
- Mushroom classification dataset, 210, 222, 233, 242, 250
 - confusion matrix for, 111, 112, 113
- mushroom.data, 250
- mushroom.dtableviz, 222
- mushroom-dt.treeviz, 210
- mushroom.eviviz, 233, 235, 236, 237
- mushroom.schema, 210, 222, 233
- Mutual Info option, 73, 167

N

- Naive-Bayes, 123
- naming
 - columns, 2
- negative values, 177
- network connections, 175
- New column name text field, 15
- Next button, 189
- Next field, 171
- nl.births.data, 249
- nl.births.scatterviz, 249
- nodes, 177
 - base heights, 181
 - decision trees
 - viewing information, 71
 - disk heights, 180
 - filtering, 190
 - finding specific, 186
 - selecting child, 193
- Nominal Order menu, 70, 91, 128
- Normalized Mutual Info option, 73, 167
- Normalize Heights option, 180
- Normalize On option, 120
- Normalize Subtree command, 191
- Nulls command, 192
- null values, 152, 193
 - decision trees and, 77
 - mapping, 152, 194
 - objects and, 192
 - predicting, 112
 - splats and, 162
- numbers
 - filtering, 95
 - searching for, 188

O

- objects
 - displaying messages
 - Map Visualizer, 121
 - Scatter Visualizer, 150
 - Tree Visualizer, 183
 - geographical, 116, 119
 - null heights and, 192
 - selecting
 - null values and, 130, 195
 - Tree Visualizer, 190
 - viewing selected, 123
 - zero heights and, 192
- Open command, 91, 92
- opening files
 - Scatter Visualizer, 146
 - Tree Visualizer, 91, 92, 177
- operators
 - relational, 188
 - filtering data, 95
- Option Nodes, 132
 - defined, 130
- Option Tree classifier
 - generating, 130
- Option Tree inducer
 - adjusting induction algorithm, 132
 - required files, 131
- Origin of cars dataset, 205, 216, 229, 241
- Or operations, 95, 188
- Other Options option, 150, 161
- Outline option, 190
- outlines, 192
- Outlines File field, 119

P

- parallel computing, 72, 131, 134

- parallelization, 133
 - parameters
 - display options, 192
 - Parent button, 193
 - Party affiliation dataset, 210, 223, 234, 242
 - Path slider, 13
 - people.data, 249
 - people.scatterviz, 249
 - Percent option, 76
 - perhouse.perage.data, 239
 - perhouse.perage.mapviz, 239
 - pima.dtableviz, 226
 - pima-dt.treeviz, 212
 - pima.schema, 212, 226, 237
 - Play Forward button, 12
 - Play-once button, 12
 - Play Reverse button, 12
 - population.australia.data, 238
 - population.australia.mapviz, 238
 - population.canada.data, 238
 - population.canada.mapviz, 238
 - population.europe.data, 239
 - population.europe.mapviz, 239
 - population sample files, 238, 239
 - population sampling, 137
 - population.usa.cities.data, 239
 - population.usa.cities.mapviz, 239
 - population.usa.data, 239
 - population.usa.mapviz, 239
 - Predict discrete label values mode, 15
 - predicting unknown values, 112
 - predictions, 37, 84
 - confusion matrices and, 58
 - Prev field, 171
 - Previous button, 189
 - Print Image command, 91, 92
 - prior probability, 84, 85
 - probabilities, 84
 - correcting, 85
 - generating, 85
 - probability estimates, 32, 38
 - product categories sample files, 202
 - product group sample files, 202
 - progress dialogs, disabling, 87, 117, 199
 - Pruning factor option, 74
 - pruning methods
 - Decision Tree Inducer
 - confidence, 74
 - cost complexity, 74
 - Publish on the Web command, 92
 - purity, 52, 71, 76
 - testing, 53
 - purity measure, 55
 - Purity option, 76
- Q**
- quantities, 177
 - quiet option, 87, 117, 199
 - quitting
 - Tree Visualizer, 92
- R**
- random samples, 99
 - random seeds, 100
 - records
 - assigning to classes, 70, 83, 130
 - availability, 79
 - classifiers and, 32
 - unlabeled, 80

- Record Viewer
 - overview, 135
 - renumbering rows, 136
 - save as, 137
 - saving data, 137
 - searching, 136
 - starting, 136
 - record weighting, 32, 101, 137
 - association rules and, 24
 - Regression Tree Inducer
 - error estimation, 142
 - options, 139
 - Cost Complexity Pruning, 141
 - Limit tree height by, 140
 - Split lower bound, 141
 - Splitting criterion, 140
 - overview, 138
 - regressor, 138
 - relational operators, 188
 - filtering data, 95
 - relationships, analyzing, 145, 155, 177
 - Remove Columns option, 62
 - renumbering rows
 - Record Viewer, 136
 - Reopen command, 91, 92
 - required files
 - Map Visualizer, 116
 - Option Tree inducer, 131
 - Scatter Visualizer, 146
 - Splat Visualizer, 159
 - Tree Visualizer, 177
 - Reset Options button, 121, 151, 161, 184
 - resetting defaults
 - Map Visualizer, 119
 - Scatter Visualizer, 148
 - Splat Visualizer, 161
 - Tree Visualizer, 178
 - resetting tool options
 - Map Visualizer, 121
 - Splat Visualizer, 161
 - Tree Visualizer, 184
 - resizing labels, 149
 - Return-on-Investment curve, 100, 143
 - revenue sample files, 253
 - Reverse button, 12
 - ROI curve, 100, 143
 - ROI Curve option, 100
 - rules files, 22
 - sample, 202
 - running clustering, 42
 - running execute statements
 - Scatter Visualizer, 199
 - running UNIX commands, 121, 151, 183
- ## S
- Salary Factors dataset, 208, 218, 231
 - sales sample files, 248, 253
 - sample files
 - census database, 202
 - Cluster Visualizer, 202
 - credit database, 202
 - product categories, 202
 - product group, 202
 - Sample option, 62
 - Sample Results of a Search in the Tree Visualizer, 189
 - save as
 - Record Viewer, 137
 - Save As command, 92
 - Save Current History As command, 185
 - saving data, 61
 - Record Viewer, 137
 - saving tool options
 - Map Visualizer, 121
 - Scatter Visualizer, 151
 - Tree Visualizer, 185

- Scale to Filter command, 94
- scaling
 - bars, 181
 - bases, 181
 - entities, 148
 - geographic regions, 120
- Scatter Visualizer
 - animation control panel
 - displaying, 197
 - summary window, 149
 - color mappings, 149
 - data files, 146
 - file requirements, 146
 - loading files, 146
 - null values and, 152
 - options
 - saving, 151
 - resetting defaults, 148
 - selecting columns, 54
 - selecting objects, 153
 - starting, 146
- .scatterviz filename extensions, 146
- Search button, 189
- Search command, 186
- searches
 - specifying search criteria, 188
 - wildcards, 95, 188
- searching
 - Record Viewer, 136
- Search Panel command, 75
- search spotlights, 190
 - turning off, 189
- Select button, 189
- selecting bases, 193
- selecting colors, 50, 51
- selecting entities, 150
- selecting models, 14
- selecting multiple values, 123
- selecting objects
 - null values and, 130, 195
 - Scatter Visualizer, 153
 - Tree Visualizer, 190
- Selection menu
 - Evidence Visualizer, 91
 - Tree Visualizer, 191
- select mode
 - Evidence Visualizer, 88
- Send to Tool Manager command, 226
- servers
 - connecting to, 175
- Set All button
 - search dialog, 187
- Set Home command, 192
- Set Minimum Weight per Bin option, 85
- setting the locale, 104, 105
- Show Animation Panel command, 197
- Show menu (Tree Visualizer), 185
- Show Original Data command, 226
- Show Values command, 191
- Show Window Decoration command, 197
- Simple Bayes, 123
- Simple mode, 53
- single k-means clustering, 40
- Single-Step buttons, 12
- sky colors, 184
- slider
 - creation, 148, 154
 - creation in Splat Visualizer, 162
 - mapping options, 150
- Slider options, 150
- small values, filtering out, 190
- Solid option, 190
- Sort By Importance command, 90
- Sort by Key option, 184
- sorting, 33, 184

- sort order
 - specifying, 184
- Speed slider, 13
- Sphere command, 166
- Splat Colors option, 160
- splats
 - color options, 160
 - defined, 155
 - displaying, 160
 - drawing options, 160, 165
 - labeling, 161
- Splat Shape option, 160
- Splats option, 160
- Splat Type menu, 165
- Splat Visualizer
 - aggregating data points, 157, 158
 - animation control panel, 163
 - summary window, 161
 - color mappings, 160
 - data files, 159
 - displaying data, 155, 165
 - file requirements, 159
 - null values and, 162
 - options
 - resetting, 161
 - resetting defaults, 161
 - starting
 - resetting defaults and, 161
 - startup options, 160
- .splatviz.data files, 161
- .splatviz.schema files, 161
- Split Lower Bound option, 74
- Splitting criterion option, 73, 167
- spotlights, 190
 - turning off, 189
- standard deviation, 76
- starting
 - Decision Table Inducer, 63
 - Scatter Visualizer, 146
 - Splat Visualizer
 - resetting defaults and, 161
 - starting Record Viewer, 136
 - Start Tool Manager command
 - Tree Visualizer, 92
 - stateRevenue.data, 253
 - stateRevenue.treeviz, 253
 - Stop button, 12
 - store.data, 253
 - store.treeviz, 253
 - store-type.data, 248
 - store-type.scatterviz, 248
 - strings
 - comparing, 95
 - filtering, 95
 - mapping, 156
 - searching, 188
 - Subtract Minimum Evidence command, 90
 - Subtree weight option, 76
 - summary legends, 149
 - Summary options, 149, 161
 - summary values, 177
 - summary window (Scatter Visualizer), 149
 - summary window (Splat Visualizer), 161
 - support, 21
 - minimum threshold, 21
 - Swing button, 13
 - system parameters
 - 64-bit support and, 133
 - rlimit__nofile_cur, 134
 - rlimit__rss_cur, 133
 - rlimit__vmem_cur, 134
 - rlimit__pthread_cur, 133

T

Table History buttons, 171

tables

- classifiers and, 99
- processing options, 61
- saving data, 61

telecom.data, 239

telecom.mapviz, 239

Test attribute option, 76

testing classifier accuracy, 99

Test Model

- confusion matrix, 16
- lift curve, 16

Test Model panel, 16

Test set error/loss option, 76

Test value option, 76

text field

- Thresholds for selected column are, 36

Texture command, 166

The, 28, 163

three-dimensional landscapes, 155, 177

Tool Manager

- multiway rules
- figure, 25

tools

- multiple selection and, 123

Treat Nulls as Zeros option, 77

Tree Visualizer

- classifiers and, 102
- data files, 177
- exiting, 92
- file requirements, 177
- filtering data, 181
- getting information, 190
- loading files, 91, 92, 177
- moving through, 192
- null values and, 193

options

- resetting, 184
- saving, 185

printing, 91, 92

resetting defaults, 178

saving defaults, 185

selecting child nodes, 193

selecting columns, 54

selecting objects, 190

- null values and, 130, 195
- spotlighting information, 189, 190

Tree Visualizer's Search Dialog Box (IRIX), 187

Tree Visualizer Options, 178

Tree Visualizer Options dialog box, 179

Tree Visualizer Selections Menu, 191

.treviz extensions, 177

trimming factor, 97

two-dimensional aggregation, 11

U

UNIX commands, 121, 151, 183

UNIX startup commands

- association rules, 23

unlabeled records, 80

usa.cities.gfx, 239

usa.cities.hierarchy, 239

usa.cities.lines.gfx, 239

usa.cities.lines.hierarchy, 239

usa.states.gfx, 239

usa.states.hierarchy, 239

Use approach menu

- Auto, 31
- Min weight per bin, 34

Use loss matrix option, 101, 113

Use Weight in clustering, 45

Use Weight menu, 36

Use Weight option, 101

US maps, 118, 239

V

values, selecting multiple, 123

variables

as filters, 94

View All command, 192

View History button, 172

View Menu, 185

View menu

Decison Table Visualizer, 69

Evidence Visualizer, 90

Scatter Visualizer, 197

views

current, 171

Map Visualizer, 114, 115

moving through, 171

Tree Visualizer

moving through, 192

spotlighting information, 189, 190

vote.dtableviz, 224

vote-dt.treenviz, 211

vote.eviviz, 235

vote.schema, 235

voting records example, 210, 223, 234, 242

W

-warnexecute option, 199

warnings, 199

Web files, 199

Weight command, 70, 128

Weight is Attribute in clustering, 45

Weight is Attribute option, 101

“what if” questions, 83

wildcards

Map Visualizer, 95

Tree Visualizer, 188

X

.Xdefaults files, 199

X sliders, 149

Y

Y2K compliance, 200

year 2000 compliance, 200

Y sliders, 149

Z

Zeros command, 192

zero values

objects and, 192