MineSet™
User's Guide 2.6 Addendum

Document Number 007-3915-001

CONTRIBUTORS

Written by Sandra Motroni and Helen Vanderberg
Illustrated by Dany Galgani
Production by Kirsten Pekarek
Engineering contributions by Barry Becker, Dave Bouvier, Jeff Brainerd, Cliff Brunk, Eric Eros, Ariel Faigon, Eben Haber, Georges Harik, Ara Jerahian, Andy Kar, Ed Karrels, Ronny Kohavi, Alex Kozlov, Clay Kunz, Alan Norton, Peter Rathmann, Dan Sommerfield, Peter Welch, and Brett Zane-Ulman.

St. Peter's Basilica image courtesy of ENEL SpA and InfoByte SpA. Disk Thrower image courtesy of Xavier Berenguer, Animatica.

MineSet™ User's Guide 2.6 Addendum
Document Number 007-3915-001

Contents

List of Figures

List of Tables

About This Guide

This chapter contains the following sections:

- "How to Use this Book" on page xi
- "Where to Find More Information About Mineset" on page xii
- "Illustrations in This Guide" on page xiii
- "Typographical Conventions" on page xiii

How to Use this Book

This *Addendum* is designed to be used with the *MineSet User's Guide*. The *User's Guide* has not been modified for the 2.6 release, so this *Addendum* describes what is new and what has changed from release 2.5.

Note: If you want to use the Tech Pubs Library or IRIS InSight search engine to find information, search both the *Addendum* and the *User's Guide* to ensure that you find everything that is pertinent.

Table i lists which tools are described in which books.

Table i Where to Find Information

For This Tool	Look Here
Decision Table	*MineSet User's Guide* for basic information and Chapter 1, "What's New in MineSet 2.6," in this Addendum for additions and changes in 2.6
Decision Tree	*MineSet User's Guide* for basic information and Chapter 1, "What's New in MineSet 2.6," in this Addendum for additions and changes in 2.6

Table i (continued) Where to Find Information

For This Tool	Look Here
Evidence Visualizer	*MineSet User's Guide* for basic information and Chapter 1, "What's New in MineSet 2.6," in this Addendum for additions and changes in 2.6
Histogram Visualizer	Chapter 1, "What's New in MineSet 2.6," in this Addendum
Record Viewer	Chapter 3, "Using the Record Viewer," in this Addendum
Rules Visualizer	Chapter 2, "Using the Association Rules Tool," and Appendix A, "Using the Association Rules Generator With Transaction-Style Data," in this Addendum
Scatter Visualizer	*MineSet User's Guide* for basic information and Chapter 1, "What's New in MineSet 2.6," in this Addendum for additions and changes in 2.6
Web publishing option	Chapter 1, "What's New in MineSet 2.6," in this Addendum
Various new features	Chapter 1, "What's New in MineSet 2.6," in this Addendum
All other tools and subjects	*MineSet User's Guide*

Where to Find More Information About Mineset

For more information about MineSet:

- Visit http://mineset.sgi.com
- Join the MineSet mailing list at http://mineset.sgi.com/maillist.html
- Visit http://support.sgi.com
- Call 1-800-800-4SGI (U.S. and Canada) or contact your local Silicon Graphics sales office outside the U.S. and Canada
- Send e-mail to mineset@sgi.com for MineSet-specific problems

Illustrations in This Guide

The hard copy of this documentation provides all screen shots and illustrations in black and white. The online version, however, provides these visuals in full, original color. Thus, if you are reading the hard copy version and find a particular graphic or screen shot difficult to see, go to the respective page of the online version for greater clarity.

Typographical Conventions

The following type conventions and symbols are used in this guide:

Italics Executable names, filenames, utilities, and variables to be supplied by the user.

Bold Keywords and functions

`Fixed-width type`
On-screen command-line text and prompts.

`Bold fixed-width type`
User input, including keyboard keys (printing and non-printing); literals supplied by the user in examples, code, and syntax statements.

[ ] Syntax statement arguments surrounded by square brackets denote that these arguments are optional.

Chapter 1

What's New in MineSet 2.6

This chapter describes the new features in MineSet 2.6. Table 1-1 provides an overview of these features, with references to where the subjects are described in greater detail, if applicable.

Table 1-1 New Feature Overview

New Feature	Short Description
New Licensing Plans	MineSet has several new licensing plans that are easily tailored to your needs. See "New Licensing Plans" on page 3.
Internationalization	MineSet 2.6 provides support for international datasets. Text labels in the graphical interface still appear in English, but you may now view multibyte column names and data values in the language corresponding to the data encoding. See "Internationalization" on page 4.
64-Bit Support	Large memory (64-bit) is supported on IRIX 6.4 and later releases. See "64-Bit Support" on page 8.
Year 2000 Compliance	MineSet now supports Y2K-compliant dates. See "Year 2000 Compliance" on page 8.
New Mining Tool Plugin API	MineSet 2.6 provides an API that third-party vendors can use to extend the functionality of MineSet through the use of plugins. See "New Mining Tool Plugin API" on page 9.
New Data-Importing Utility	The new MineSet data-importing utility, dataschema, automatically creates MineSet data and schema files from flat file formats. See "New Data-importing Utility (dataschema)" on page 9.
New Bin Names	Bin names have a new format that better defines the range of values within each bin. See "New Bin Names" on page 11.
New Histogram Visualizer	The new Histogram Visualizer automatically bins all of the continuous-type columns in the data and sends the result to the Statistics Visualizer for display. See "New Histogram Visualizer" on page 12.

Table 1-1 (continued) New Feature Overview

New Feature	Short Description
New Record Viewer	The new Java-based Record Viewer provides extra functionality such as record numbering, sorting by various criteria, filtering, searching, and writing to HTML tables. The functionality is described in Chapter 3, "Using the Record Viewer."
New Features in Tool Manager	The Tool Manager has two new features: • The new Column Sort operation sorts the list of columns by name. • The Add Column and Filter panels now support the **if** (*A*) **then** (*B*) **else** (*C*) expression. This expression means that if A is true, use the value of B, otherwise use the value of C.
New Format for Rules Visualizer	Previous versions of MineSet used a separate tool for association rules visualization. Now, association rules are shown using the Scatter Visualizer, and there are some changes in the generation of the rules. A full description of the new format is covered in Chapter 2, "Using the Association Rules Tool." Configuration file information for the new format is found in Appendix A, "Using the Association Rules Generator With Transaction-Style Data."
Evidence Visualizer Filtering	The Evidence Visualizer now allows filtering on the set of displayed attributes.
Scatter Visualizer Enhancements	The Scatter Visualizer now allows you to show a trail of motion to demonstrate the changing animation path of an entity. See "Scatter Visualizer Enhancements" on page 13.
Scatter Visualizer Configuration File Enhancements	Several statements have been added to the Scatter Visualizer configuration file to support the new association rules visualization capability of the Scatter Visualizer. See "Scatter Visualizer Configuration File Enhancements" on page 14.
Decision Table Visualization Enhancements	The Decision Table Visualizer has two new features. It now allows filtering on attribute values in the same way that the Scatter Visualizer does, and it has an Evidence mode that is the same as the Evidence mode of the Evidence Visualizer. See "Decision Table Visualization Enhancements" on page 16.
New Decision Tree Inducer Options	The Decision Tree Inducer now has an extended set of splitting criteria and pruning methods. See "New Decision Tree Inducer Options" on page 19.

Table 1-1 (continued) New Feature Overview

New Feature	Short Description
New Web Publishing Option	All visualization tools now provide the option of creating a file based on a visualization that may be published on the Web. See "New Web Publishing Option" on page 21.
Changes in File Exchange Procedures Between MineSet and SAS	There have been a few changes to the file exchange procedures between MineSet and SAS. See "Changes in File Exchange Procedures Between MineSet and SAS" on page 21

New Licensing Plans

MineSet implements a client-server architecture. The client and the server need separate licenses. Typically, a client is used by a single user on a desktop system, while the server runs on a larger system that may be shared by multiple clients simultaneously. The client side runs the Tool Manager and the visualization tools, and the server side runs the DataMover and analytics. You may run the client and the server on the same system, but you need both a client license and a server license.

Client licenses are simple. There is only one type of client license, and it is tied to a system ID. Once you have a MineSet client license on your system, you may run an unlimited number of visualizers and tool-manager processes on that system as long as they read local files and do not connect to a server.

Server licenses are more complex. In essence, each server license allows one simultaneous connection to the MineSet server. For example, if you have five users working simultaneously, each using one client, you need five server licenses on the server system. Unlike a client license which is unlimited on one system, a server license means support for only one active session.

There are two types of MineSet server licenses:

- Varsity, which is available to research and education institutions only
- Normal, which is available to all other customers

Normal server licenses are priced in such a way that they become cheaper the more licenses you buy. As of MineSet 2.6, there are three tiers of Normal server licenses:

- Basic, which provides the first session on the server.
- 2 to 4, which provides a less expensive license for each additional session up to four sessions. You must have at least one Basic license in order to obtain a "2 to 4" license.
- 5 and up, which provides an even less expensive license for each session above four. You must have at least one Basic license and three "2 to 4" licenses to obtain a "5 and up" license.

Lastly, there is a special kind of inexpensive server evaluation license called "MineSet Light." If your dataset has fewer than 5,000 records, MineSet uses all of them. If your dataset has more than 5,000 records, MineSet auto-samples to 5,000 records and uses only those.

Mixing Varsity with any other license type is not allowed. Mixing Normal with Light licenses is not recommended. In MineSet 2.6, exceeding the number of licenses on the server generates a warning. The warning alerts you to the fact that you are exceeding your server licensing terms. Stricter control of license usage may be introduced in the future.

Internationalization

Beginning with version 2.6, MineSet supports international datasets. Text labels in the graphical interface still appear in English, but you can now 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 WorldView product is installed. For other languages and encodings see "Extending to Other Languages and Encodings" on page 5.

Setting the Locale

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 UNIX 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 IRIX documentation.

Extending to Other Languages and Encodings

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 page 5 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*
   - *Eviviz*
   - *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 1-2 for an example).
5. Set the locale to *locale_name* and invoke MineSet.

Example 1-1 Resource File Changes for Korean

The changes needed for Korean are given in Table 1-2. The fonts listed came from the lists in the following files:

- */usr/lib/X11/fonts/ps2xlfd_map.korean*
- */usr/lib/X11/fonts/ps2xlfd_map.korean.outline*

Table 1-2 Resource File Changes for Korean (ko_KR.euc)

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-*-*-*-*-*-*-*	zoom2*fontList: -*-*-medium-r-*-*-6-*-*-*-*-*-*-*;-ksg-*-medium-*--12-*:
	zoom3*fontList: -*-*-medium-r-*-*-8-*-*-*-*-*-*-*	zoom3*fontList: -*-*-medium-r-*-*-8-*-*-*-*-*-*-*;-ksg-*-medium-*--12-*:
	zoom4*fontList: -*-*-medium-r-*-*-10-*-*-*-*-*-*-*	zoom4*fontList: -*-*-medium-r-*-*-10-*-*-*-*-*-*-*;-ksg-*-medium-*--14-*:
	zoom5*fontList: -*-*-medium-r-*-*-12-*-*-*-*-*-*-*	zoom5*fontList: -*-*-medium-r-*-*-12-*-*-*-*-*-*-*;-ksg-*-medium-*--14-*:
	zoom6*fontList: -*-*-medium-r-*-*-14-*-*-*-*-*-*-*	zoom6*fontList: -*-*-medium-r-*-*-14-*-*-*-*-*-*-*;-ksg-*-medium-*--18-*:
	zoom7*fontList: -*-*-medium-r-*-*-16-*-*-*-*-*-*-*	zoom7*fontList: -*-*-medium-r-*-*-16-*-*-*-*-*-*-*;-ksg-*-medium-*--24-*:
	zoom8*fontList: -*-*-medium-r-*-*-24-*-*-*-*-*-*-*	zoom8*fontList: -*-*-medium-r-*-*-24-*-*-*-*-*-*-*;-ksg-*-medium-*--24-*:

64-Bit Support

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 pthreads. 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 1-3 lists the systune parameter values that Silicon Graphics recommends (for more details see the systune(1M) man page):

Table 1-3 systune Parameters

Parameter	Definition	Recommended Value
rlimit_pthread_cur	Current limit on the number of threads	1024
rlimit_rss_cur	Current limit on memory usage	The amount of physical memory on your machine
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 file	1024 or the limit on the number of threads

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

Year 2000 Compliance

MineSet now 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: numbers greater than 68 are assumed to be the years 1969 to 1999, and numbers 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.

New Mining Tool Plugin API

Now, with the use of an API, third-party vendors can extend the functionality of MineSet 2.6.

The MineSet Mining Tool Plugin API provides a means for third parties to plug in a GUI to the Mining Panel in the Tool Manager, save options to a *.mineset* file, and send these options to the DataMover. The DataMover then runs the third-party mining tool program and manages its output of models and model visualization files. The model visualization files are then sent back to the Tool Manager, which runs the requested visualization. Models created by third-party plugin mining tools can later be applied to a dataset using the Apply Model transformation in the Tool Manager.

At startup, MineSet looks for third-party dynamic shared object (DSO) libraries in */usr/lib/mineset/plugins*.

In MineSet 2.6, a clustering algorithm named AutoClassPro (ACPro) from Ultimode Systems is available as an add-on. If ACPro is installed, documentation for it can be found in */usr/acpro/doc*.

New Data-importing Utility (dataschema)

dataschema is a MineSet data-importing utility that automatically creates MineSet data and schema files from flat file formats.

dataschema Features

The *dataschema* features are:

- It handles arbitrary text (flat) input files as long as:
  - There is only one record per line
  - Fields within each record are separated by some special character (which can be any character)
- It imposes no limits on input data sizes such as the number of columns or rows.
- It automatically identifies the field separator character.
- It automatically identifies column types.

- It supports column names on the first line of input (if they are given).
- It supports either UNIX or DOS style CR/LF ends of lines.
- It supports leading and trailing space stripping from constant-length fields.
- It supports MineSet dates and missing (null) values.

dataschema requires either perl4 or perl5 to run.

Running dataschema

You can call *dataschema* from any command shell with the input files you want it to process. For example, if you type:

```
dataschema /tmp/mydata.csv
```

dataschema reads */tmp/mydata.csv,* analyzes it, and creates two output files in the current directory, *mydata.schema* and *mydata.data*, that can be read by MineSet. Editing *mydata.schema* for further customization is encouraged (especially for changing column names). If your input contains column names on the first line (separated by the same separator as the actual data fields), these column names are used in the schema file.

dataschema is flexible and supports several options. Invoking *dataschema* without any arguments prints out a usage message including all the supported options.

For more information on *dataschema*, visit the following Web page:

http://mineset.sgi.com/utils/dataschema.html

New Bin Names

In MineSet 2.6, bin names have a new format. MineSet 2.5 used the convention - 10, 10-20, 20-30, 30+ for bin names, which led to some confusion. The - 10 was often thought to be negative ten, and it was not clear which bin contained the boundary points. MineSet 2.6 uses a modified interval notation for bin names:

(*lower-bound* . . . *upper-bound*]

The "(" indicates that the lower bound is not included in the range. The "]" indicates that the upper bound 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, more formally, { X : 10.5 < X <= 12.6 }. If the lower bound 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, or more formally, { X : X <= 10.5 }.

If the upper bound 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, or more formally, { X : X > 12.6 }.

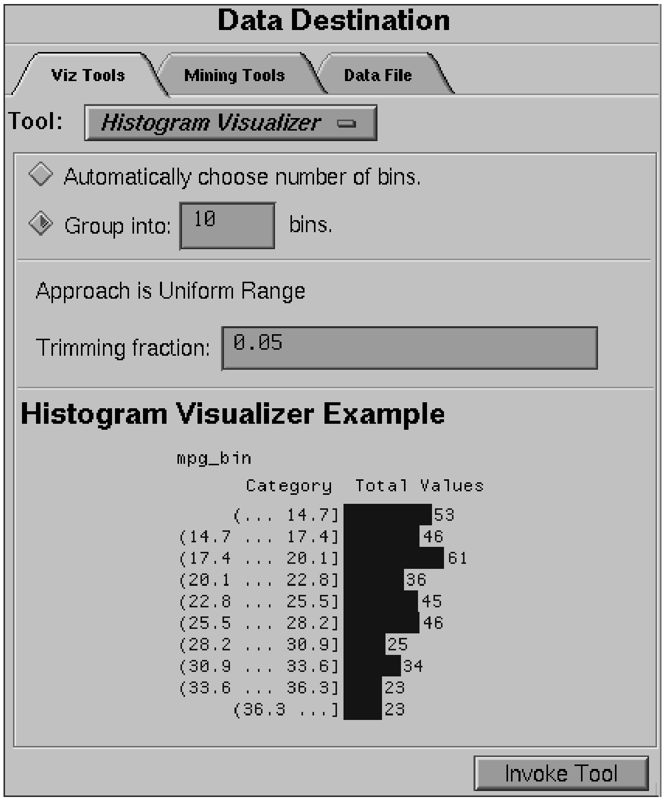
The example of the MineSet 2.5 bin names - 10, 10-20, 20-30, 30+ can be expressed in MineSet 2.6 as (... 10], (10 ... 20], (20 ... 30], (30 ...]. Other examples make the naming scheme clear:

(... -1]	Values under and including -1
(-1 ... 10]	Values over -1 up to and including 10
(10 ... 20]	Values over 10 up to and including 20
(20 ...]	Values over 20

New Histogram Visualizer

The Histogram Visualizer automatically bins all of the continuous-type columns of in the data and sends the result to the Statistics Visualizer. Figure 1-1 shows 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 factor. The trimming factor 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.



Data Destination

Tool: Histogram Visualizer

◇ Automatically choose number of bins.

◆ Group into: 10 bins.

Approach is Uniform Range

Trimming fraction: 0.05

Histogram Visualizer Example

```
mpg_bin
          Category   Total Values
       (... 14.7]  53
 (14.7 ... 17.4]  46
 (17.4 ... 20.1]  61
 (20.1 ... 22.8]  36
 (22.8 ... 25.5]  45
 (25.5 ... 28.2]  46
 (28.2 ... 30.9]  25
 (30.9 ... 33.6]  34
 (33.6 ... 36.3]  23
       (36.3 ...]  23
```

Figure 1-1 Histogram Visualizer Options

Scatter Visualizer Enhancements

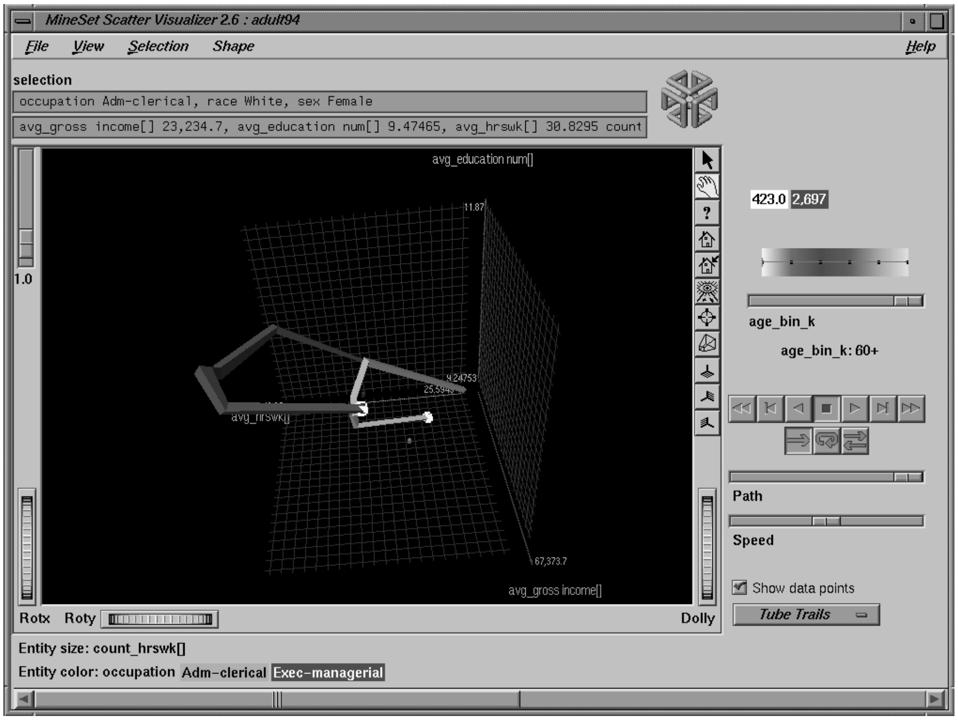
The Scatter Visualizer now allows you to show a trail of motion to demonstrate the changing animation path of an entity. When you create an animation, the trail shows behind each selected entity, in the form you have selected. The motion option menu, located at the bottom right of the ScatterViz control panel, allows you to select from:

- No trails—the default
- Line trails—a thin colored line
- Fade-out trails—a similar colored line, most opaque at its most recent position
- Tube trails—trails in 3D tubular form, showing changes of size as the entity moves through the animation path. Too many tube trails may slow animation noticeably.

All trails are color-coded according to the originating entity. If an entity changes from red to blue as the summary slider changes from one position to another, the corresponding trail will also be shown changing color gradually between the two positions. Trails are made between points whose unmapped attributes stay the same over the course of the path.

Aggregated data grouped by a small number of columns tends to be an excellent candidate for the display of motion trails. Initially, motion trails are displayed for all points in the scatterplot affected by the path. Selecting any entity by clicking on it with the mouse causes only the selected point to display a trail. This can be used to reduce visual clutter. Entities with null positions appear as breaks in the trails.

Figure 1-2 shows an example of the Scatter Visualizer with tube motion trails.



selection

occupation Adm-clerical, race White, sex Female

avg_gross income[] 23,234.7, avg_education num[] 9.47465, avg_hrswk[] 30.8295 count

age_bin_k: 60+

Entity size: count_hrswk[]

Entity color: occupation Adm-clerical Exec-managerial

Figure 1-2 Example of ScatterViz Tube Motion Trails

Scatter Visualizer Configuration File Enhancements

To support the new association rules visualization capability of the Scatter Visualizer, the following statements have been added to the Scatter Visualizer configuration file. Refer first to Appendix D, “Creating Data and Configuration Files for the Scatter Visualizer,” in the *MineSet User’s Guide* to understand the basic format of the Scatter Visualizer’s configuration file.

Disk Height Statement

The optional **disk height** statement describes how a field is to be mapped to a disk height. The available clauses are the same as for the **size** statement. This statement must be present for disks to appear. The syntax of the **disk height** statement is:

```
disk height clause1, clause2,...
```

For a full description of the **size** statement, refer to the “View Section” of Appendix D, “Creating Data and Configuration Files for the Scatter Visualizer,” in the *MineSet User’s Guide*.

Disk Color Statement

The optional **disk color** statement describes how a field is to be mapped to the disk color. The available clauses are the same as for the **color** statement. If a **disk height** statement exists, but no **disk color** statement, the disks are the same colors as the entities. The syntax of **disk color** is:

```
disk color clause1, clause2,...
```

For a full description of the **color** statement, refer to the “View Section” of Appendix D, “Creating Data and Configuration Files for the Scatter Visualizer,” in the *MineSet User’s Guide*.

Drillthrough Statement

The **drillthrough** statement specifies a string-valued attribute that provides the filter expression used when drilling through on selected entities. The syntax has the form:

```
drillthrough var
```

This mapping option is useful when the dataset loaded into the Scatter Visualizer does not match the original dataset from which the data was derived. This most commonly occurs if some intermediate mining algorithm transformed the original dataset into a new dataset with different columns. In MineSet, this happens when the Association Rules Generator produces rules and outputs them as ScatterViz configuration and data files to be visualized. If a drill through column is specified using this statement, then the Scatter Visualizer bypasses normal drill through based on column preferences, and uses this column to build the filtering expression by “anding” together the expressions in the drill through column corresponding to the entities that were selected.

Axis Statement orderby Clause

The **orderby** clause of the **axis** statement allows you to specify that the labels along an axis are alphabetically ordered (for string values mapped to an axis). The only **orderby** option available is “alpha,” for alphabetical ordering. The statement:

```
axis LHS, orderby alpha;
```

forces string values to appear alphabetically on the LHS axis. If no **orderby** clause is present, string values are ordered by the attribute mapped to color.

For a full description of the **axis** statement, refer to “The View Section” of Appendix D in the *MineSet User’s Guide*.

Decision Table Visualization Enhancements

The Decision Table Visualizer has two new features: filtering and Evidence mode.

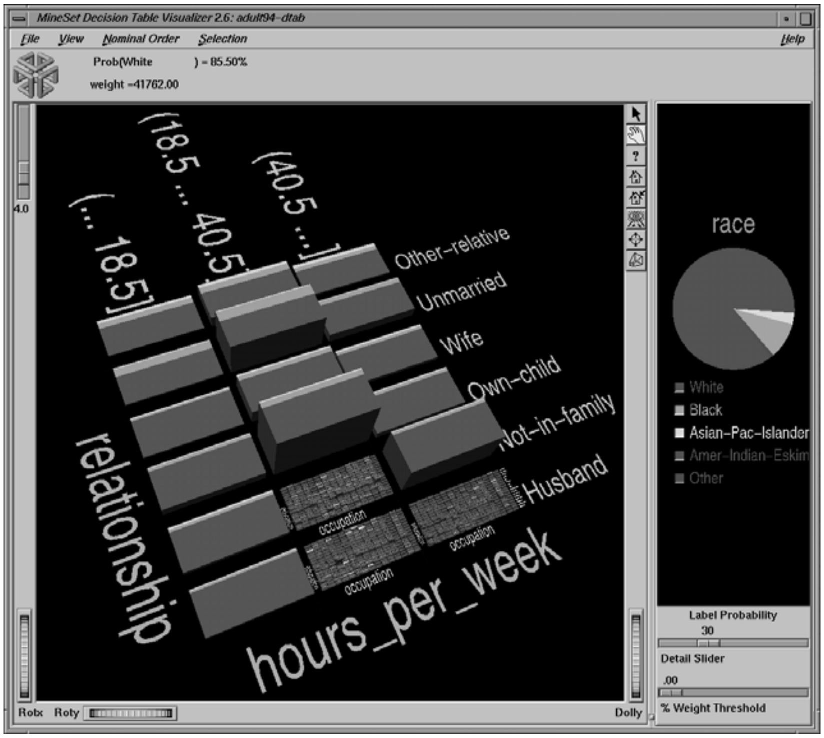
Filtering in the Decision Table Visualizer

The Decision Table Visualizer now allows filtering on attribute values in the same way that the Scatter Visualizer does. See the “View Menu” section of Chapter 7,” Using the Scatter Visualizer,” in the *MineSet User’s Guide* for a discussion of the use of filtering.

Evidence Mode in the Decision Table Visualizer

The Decision Table Visualizer now has an Evidence mode that is the same as the Evidence mode of the Evidence Visualizer. Go to the View menu of the Decision Table, and choose Evidence Mode to activate it.

The distribution in each cake now shows conditional probabilities, rather than distributions based on the record weights, which are shown initially. This is useful if one of the classes is small.



MineSet Decision Table Visualizer 2.6: adult94-dtab

Prob(White) = 85.50%

weight =41762.00

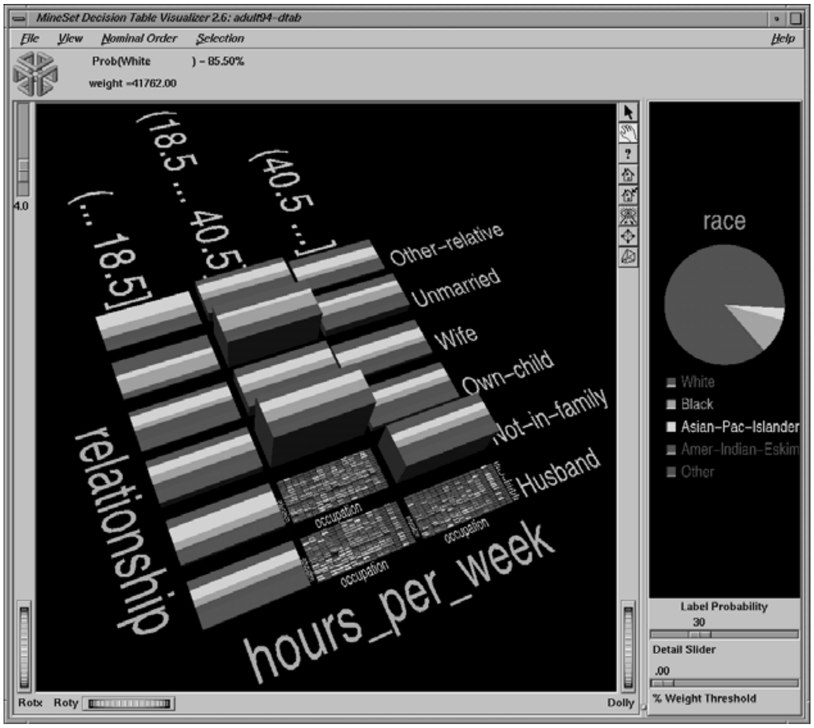
Label Probability 30

Detail Slider .00

% Weight Threshold

Figure 1-3 Decision Table Breakdown Showing Distribution Based on Race

In Figure 1-3, the label is “race” and the distribution of the data is based on weight. Because most records in this dataset are labeled `race=white`, it is difficult to discern what values give evidence for other races. Switching to Evidence mode (Figure 1-4) makes it clear which regions give evidence for races less prevalent in the data. See “Selecting Items in the Main Window,” in Chapter 13 of the *MineSet User’s Guide* for the technical description of how evidence is computed.



MineSet Decision Table Visualizer 2.6: adult94-dtab

Prob(White) = 85.50%
weight =41762.00

Label Probability 30

Detail Slider .00

% Weight Threshold

Figure 1-4 Decision Table Showing Conditional Probabilities

Normalized conditional probabilities (evidence) are shown at each cake when using the Evidence mode. From the visualizer window pull down the View menu and choose Show as Evidence.

In Figure 1-4, if the racial breakdown at a cake matches the prior probability (shown in the Label Probability window at the right of the view area), then the slices are of equal size. Bigger or smaller slices indicate correspondingly more or less evidence for a given race. If the slices for a cake are all of equal size, then the racial breakdown for that combination of values is the same as the prior distribution (the distributions shown originally in the pie in the right window.)

New Decision Tree Inducer Options

The Decision Tree Inducer now has an extended set of splitting criteria and pruning methods.

Splitting Criteria

The set of splitting criteria in MineSet 2.6 has been extended to include chi-square and Gini.

Chi-square 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.

Pruning Methods

MineSet 2.6 has three pruning options for decision trees: Confidence, Cost Complexity, and None.

Confidence

Confidence is the default pruning method used in MineSet, and is based on the heuristic pruning techniques developed in C4.5. It compares the resubstitution error of a subtree with the error if that subtree were replaced with a single node. If the error rate of the node is within the confidence interval of the subtree error, then the subtree is replaced by the single node.

The confidence pruning parameter allows you to change the amount of pruning that MineSet performs. Higher values indicate more pruning; lower values indicate less pruning. The parameter is used to scale the size of a confidence interval in which pruning occurs. The lowest possible value is 0. 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. There is no upper limit on the confidence pruning parameter. The default factor, 0.7, has been determined empirically to be a reasonable setting in many domains.

Cost Complexity

Cost complexity is the pruning technique developed in CART (Classification And Regression Trees). 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 the minimum cost. 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.

None

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.

New Web Publishing Option

All visualization tools now provide the option of creating a file based on a visualization that may be published on the Web. From the File menu, select the “Publish on the Web” option. This runs a script that produces a *.mtr* file which is placed in a selectable directory. The default directory to appear in the file selection dialog box is one of the following:

1. A directory defined by the `MINESET_WEB_DIR` environment variable, if present
2. Or the directory `$HOME/public_html` if present
3. Otherwise, the current working directory is used.

Changes in File Exchange Procedures Between MineSet and SAS

There have been a few changes to the file exchange procedures between MineSet and SAS. The following sections describe the changes or replace equivalent sections in Chapter 9, “File Exchange Between MineSet and SAS,” in the *MineSet User's Guide*:

- “SAS Installation Location” describes a new procedure for when SAS is not installed in the default location.
- “Converting MineSet Data Files to SAS Data Sets” replaces the first few paragraphs of the equivalent section in the *MineSet User's Guide*.
- “Converting SAS Data Sets to MineSet Data Files” replaces the first few paragraphs of the equivalent section in the *MineSet User's Guide*.

SAS Installation Location

If SAS is not installed in the default location (*/usr/sbin/sas* on IRIX, *c:\sas\sas.exe* on Windows NT), the environment variable SAS_CMD must be set to the installed location of the SAS executable. For example:

```
setenv SAS_CMD /usr/people/joe/sas/bin/sas  (IRIX)
set “SAS_CMD=c:\Program Files\sas\sas.exe”  (Windows NT)
```

Converting MineSet Data Files to SAS Data Sets

Use *mineset2sas* to convert MineSet data files into SAS data sets. The syntax for this is:

```
mineset2sas <MineSet file> <SAS datafile> [options]
```

The options are:

- **-svsc** to save the script sent to SAS. The script normally is deleted after use.
- **-names** <*namefile*> to save trimmed column names in <*namefile*>. The script normally is deleted after use.

For example:

```
mineset2sas cars cars.ssd01 -svsc -names cars.names
```

Converting SAS Data Sets to MineSet Data Files

Use *sas2mineset* to convert SAS data sets into MineSet data files. The syntax for this is:

```
sas2mineset <SAS datafile> <MineSet file> [options]
```

The options are:

- `-nodata` creates only a *.schema* file, no *.data* file.
- `-svsc` saves the scripts sent to SAS.
- `-nolabel` indicates that you do not want labels used for column names.
- `-names` <*namefile*> restores long column names from <*namefile*>, created by *mineset2sas*.

For example:

```
sas2mineset houses.ssd01 -svsc -names houses.names
```

Chapter 2

Using the Association Rules Tool

This chapter discusses the components and capabilities of the Association Rules generation and visualization tool. After a brief overview, the first sections cover the kind of rules that are generated during the rule generation step and define the vocabulary. The next sections explain how to construct a visualization of the rules using the Scatter Visualizer. The final sections list and describe the sample files provided for these tools, and show how you can convert an old style *.ruleviz* file to a *.scatterviz* format. This chapter replaces Chapter 9, "Using the Rules Visualizer," in the *MineSet User's Guide*.

This chapter contains the following sections:

Overview of Association Rules Generation and Visualization

The Association Rules tool lets 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 the frequency of items occurring together in a database.

Discovering and graphically displaying association rules can be relevant to many enterprises. Some examples of where the Association Rules tool 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.

The execution sequence of association rules generation and visualization is shown schematically in Figure 2-1.

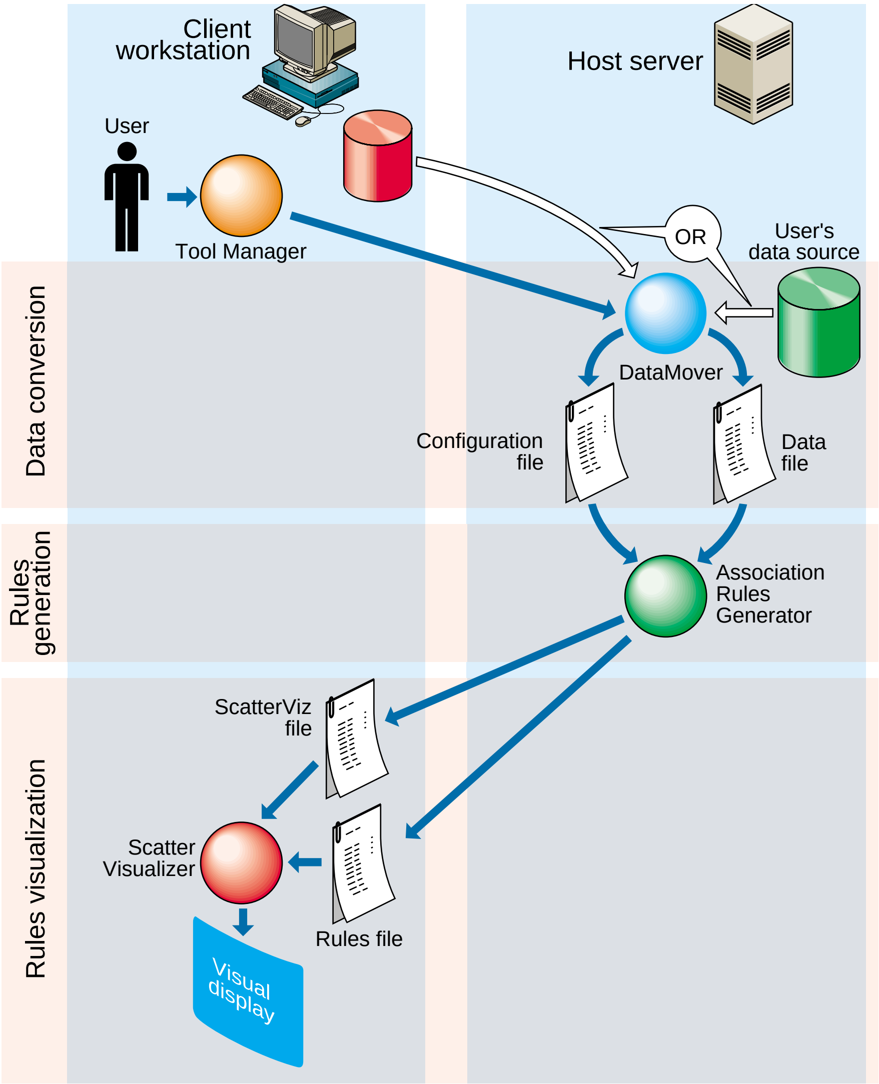


Figure 2-1 Execution Sequence of Association Rules Generation

Association Rules Generation

The Association Rules Generator can generate both simple (one-to-one) and multiway association rules. This section describes simple association rules. For a description of multiway rules, see “Multiway Association Rules” on page 38.

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 items in a record, then a rule such as:

X=>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 first number, the *confidence* of the rule, quantifies how often *X* and *Y* occur together as a fraction of the number of records in which *X* occurs. For example, if the confidence is 50%, *X* and *Y* occur together 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%.

- The second number, the *support* of the rule, quantifies how often *X* and *Y* occur together in the file 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. You can also specify a minimum confidence threshold for the generated rules. The minimum confidence threshold default is 50%.

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

  - A rule might have business value only if a reasonably significant fraction of records support the rule. For example, if everyone who buys caviar also buys vodka, the rule Caviar =>Vodka has 100% confidence. However, if only a handful of people buy caviar, the rule might be of limited value to the 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 third number, *expected confidence*, is the frequency of occurrence of the RHS item 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. Expected confidence gives an indication of what the confidence would be if there were no relationship between the items.
- The fourth number is *lift*. The lift is the ratio of confidence to expected confidence. The greater the number, the more unexpected the rule.

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=>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=>Y, nothing is known about X. Rules specify implications only from the LHS to the RHS.

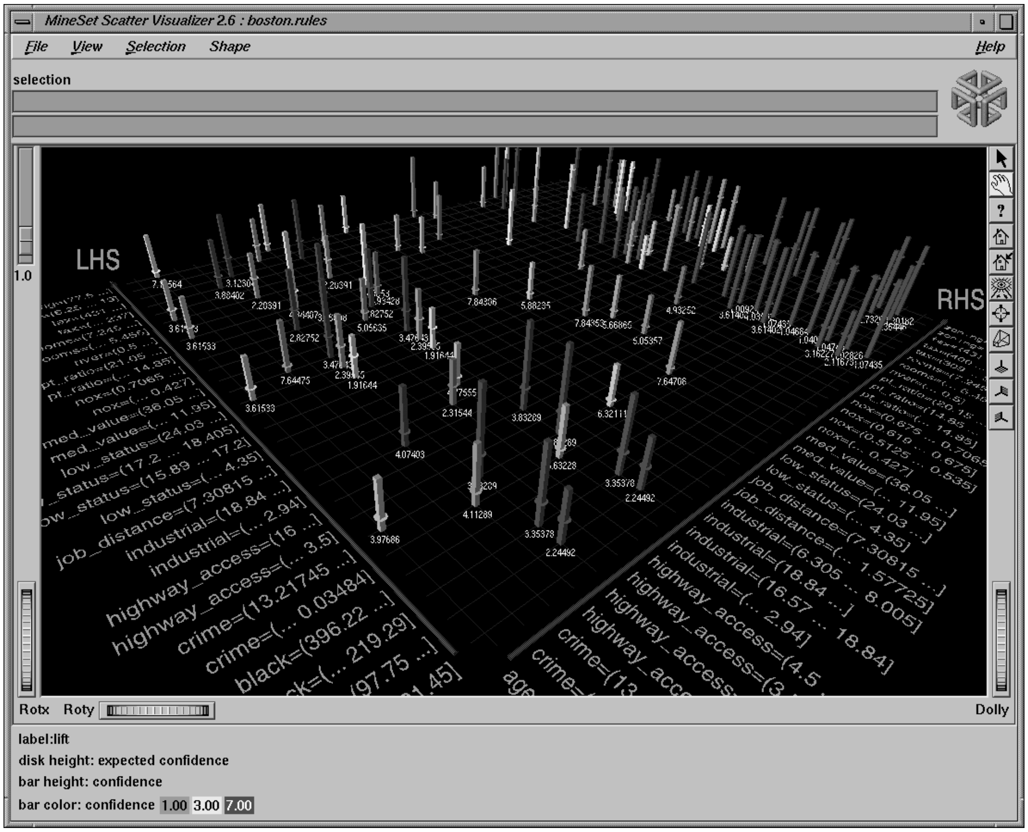
Table 2-1 summarizes the four numbers that quantify the strength of each association rule.

Table 2-1 Association Rules Components

Measure	Description	Statistical Description
Support	Frequency of LHS and RHS occurring together.	P(LHS ∩ RHS)
Confidence	Of all occurrences of LHS, the fraction where RHS is also seen, or the support divided by the frequency of occurrence of LHS items.	P(RHS ∣ LHS)
Expected confidence	Frequency of occurrence of RHS items.	P(RHS)
Lift	Ratio of confidence to expected confidence.	

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 in the Scatter Visualizer. The left-hand side (LHS) items are on one axis, and right-hand side (RHS) items are on the other. As shown in Figure 2-2, attributes of a rule are displayed at the junction of its LHS and RHS item. The display can include bars, disks, and labels.



label:lift
disk height: expected confidence
bar height: confidence
bar color: confidence 1.00 3.00 7.00

Figure 2-2 Detail View of the Association Rules Visualizer's Main Window

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 until the item labels appear (see the Dolly description in "Thumbwheels" in Chapter 6 of the *MineSet User's Guide*). 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 (see Figure 2-6). 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) can be displayed at the bottom of the main window.

File Requirements

The Tool Manager creates the two files that are required to generate the rules visualization:

- A rules file that results from running the Association Rules Generator, named the *.rules.data* file
- A *.rules.scatterviz* file

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

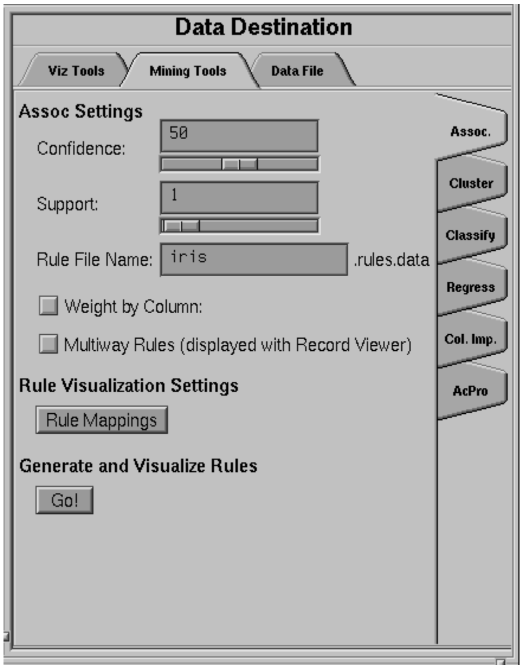
Configuring the Association Rules Tool Using the Tool Manager

This section describes how the components of the Association Rules tool can be configured using the Tool Manager. The Tool Manager greatly simplifies the task of configuring the Association Rules tool. However, if you prefer, you can construct a configuration file for this tool using an editor (see Appendix A, “Using the Association Rules Generator With Transaction-Style Data,” in this *Addendum*) or by invoking MIndUtil directly to produce the rules (see Appendix I, “Command-Line Interface to MIndUtil: Analytical Data Mining Algorithms,” in the *MineSet User's Guide*).”

The steps required to connect to a data source are described in Chapter 3, “The Tool Manager,” of the *MineSet User's Guide*.

Setting Up Associations

To show how to set up simple associations, the following example uses the cars database table. Let's say that you want to find out if there is an association between miles per gallon, horsepower, and the year the car was built. For example, did mileage improve over time? Did engines become less powerful? The following steps (and Figure 2-3) show you how to set up the associations and map table columns to the data you want to study.



Assoc Settings

Confidence: 50

Support: 1

Rule File Name: iris .rules.data

□ Weight by Column:

□ Multiway Rules (displayed with Record Viewer)

Rule Visualization Settings

Generate and Visualize Rules

Figure 2-3 Initial Tool Manager Window for Association Generation

1. Connect to a MineSet server. Refer to Chapter 2, “Setting Up MineSet,” in the *MineSet User’s Guide* if you need help.
2. Open a data source.
3. (Optional step) Number-valued columns are binned automatically, using uniform weight. If you prefer different bins, from the Data Transformations pane, choose specific numeric columns to bin before using the associations engine. Alternatively, to have each value considered individually, use the “Change Types” transformation to convert the column to string type. This prevents automatic binning altogether. (The binning operation and the options available for it are described in detail in Chapter 3, “The Tool Manager,” in the *MineSet User’s Guide*.) Use conversion of type to string carefully, as it may lead to less “meaningful” rules from the association engine. For example, instead of using discrete values for the weightlbs attribute in the cars table such as 3504, 3693, 3436, 3433, and so on, it may be more meaningful to give weightlbs_bin value ranges such as 1600-2500, 2501-3500, and so on.
4. Choose the Mining Tools tab from the Data Destination tab.

5. Choose the Assoc. tab (abbreviation for Associations) from the Mining Tools tab.

   After selecting a data source, you can run the Association Rules Generator immediately. Or you can choose settings from the following selections:

   Confidence—lets you specify the minimum confidence threshold for rules. Rules with a confidence below this value are not generated. The default is 50%. The possible values are 0–100.

   Support—lets you specify the minimum support threshold as a percentage of the total number of records. Rules with a support below this value are not generated. The default is 1%. The possible values are 0–100.

6. (Optional) Once you have made your association rule options selections, click the *RuleViz Mappings* button to map columns to visual elements.

Record Weighting

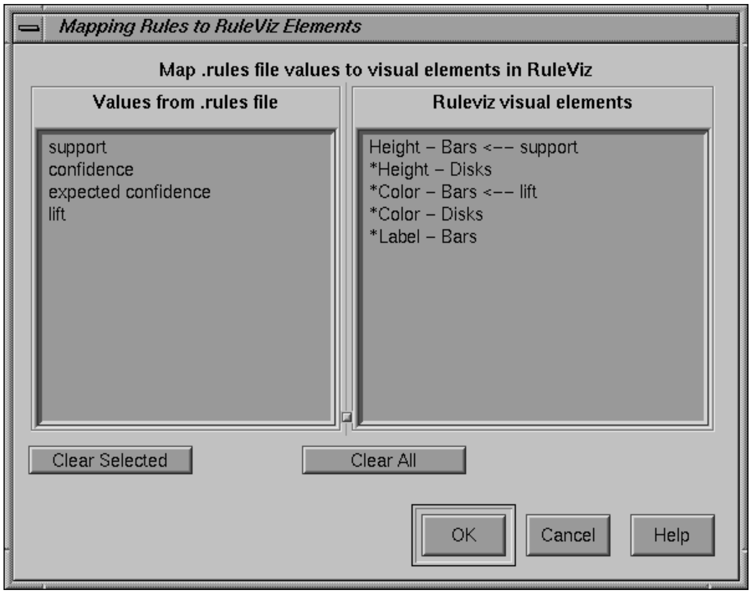
The Association Rules tool allows for record weighting for those cases in which you want to specify that certain records are more important than others or when you want to compensate for uneven sampling. If *Weight by Column* is not checked, then each record has a weight of one.

To enable record weighting, click the *Weight by Column* checkbox. When the box is checked, a popup menu appears that 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: Not All Records Were Sampled Equally" in Chapter 10 of the *MineSet User's Guide* for a further explanation of record weighting.

Mapping Columns to Visual Elements

The Association Rules tool lets you map attributes of the rules to visual elements of the display. Clicking on the *RuleViz Mappings* button brings up the Association Rules Mappings panel shown in Figure 2-4.


Mapping Rules to RuleViz Elements
Map .rules file values to visual elements in RuleViz
Values from .rules file
Ruleviz visual elements
Clear Selected
Clear All
OK
Cancel
Help

support
confidence
expected confidence
lift

Height – Bars <–– support
*Height – Disks
*Color – Bars <–– lift
*Color – Disks
*Label – Bars

Figure 2-4 Association Rules Mappings Panel

The visual elements that can be mapped are listed below; the items preceded by "*" are optional:

- *Height - Bars*—lets you specify what the bar heights represent.
- **Height - Disks*—lets you specify what the disk heights represent.
- **Color - Bars*—lets you specify what the bar colors represent.
- **Color - Disks*—lets you specify what the disk colors represent.
- **Label - Bars*—lets you specify what the bar labels represent.

The default mappings are as follows:

- Support to bar height
- Lift to bar color

Starting the Visualizer

There are five ways to start the rules visualizer:

- Use the Tool Manager to configure and start the Association Rules tool (see “Configuring the Association Rules Tool Using the Tool Manager” on page 30). The Association Rules tool automatically launches the Scatter Visualizer tool.
- Double-click the Scatter Visualizer icon, which is in the MineSet page of the icon catalog. The icon is labeled *.rules.scatterviz*. Because no configuration file is specified, the start-up screen requires you to use File > Open to select one.
- If you know which configuration file you want to use, double-click the icon for that 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 via the Tool Manager).
- Drag the configuration file icon onto the Scatter Visualizer icon. This starts the Scatter Visualizer and automatically loads the configuration file you specified. This works even if the configuration filename does not end in *.scatterviz*.
- Enter this command at the UNIX shell command-line prompt:

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

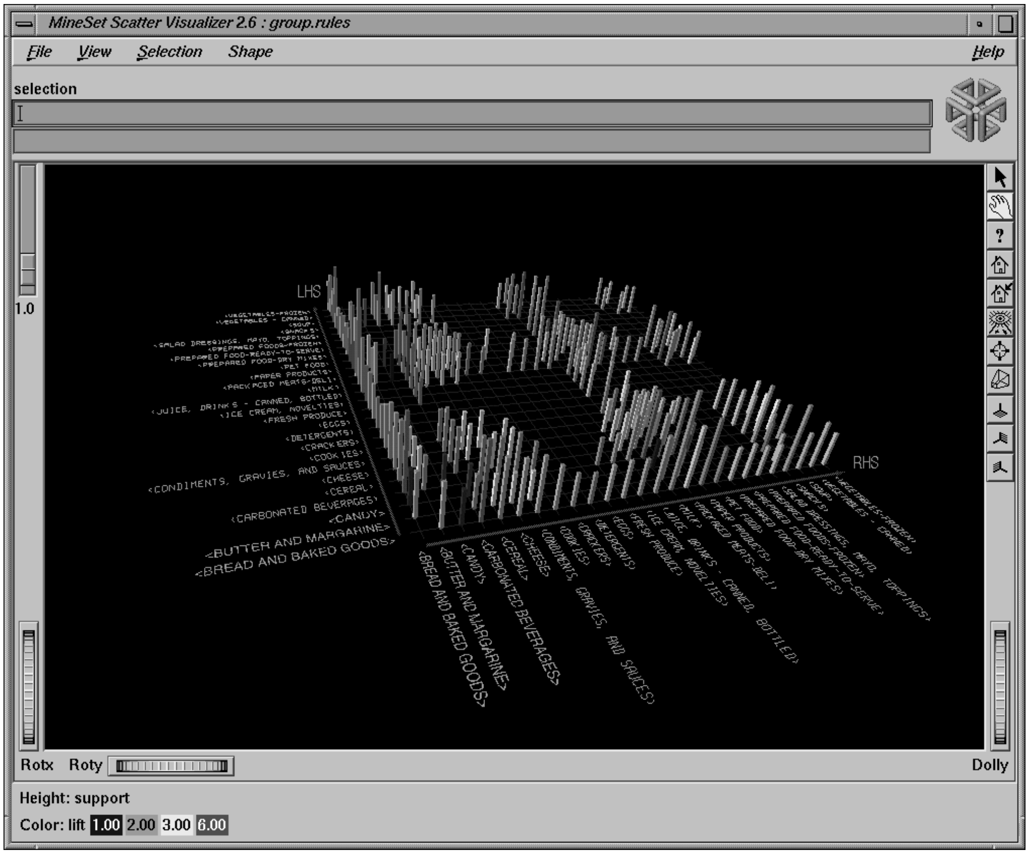
When starting the Scatter Visualizer, you must specify the configuration file, not the data file.

Note: If you wish to eliminate the dialogs that pop up to indicate progress, use the **`-quiet`** option. You can enable this option permanently by adding the line following line to your *.Xdefaults* file:

```
*minesetQuiet:TRUE
```

Interpreting Association Rules in the Scatter Visualizer

The Association Rules tool displays the data from a rules file in the Scatter Visualizer using the specifications of a valid configuration file. For example, specifying *group.rules.scatterviz* results in the image shown in Figure 2-5.



Height: support

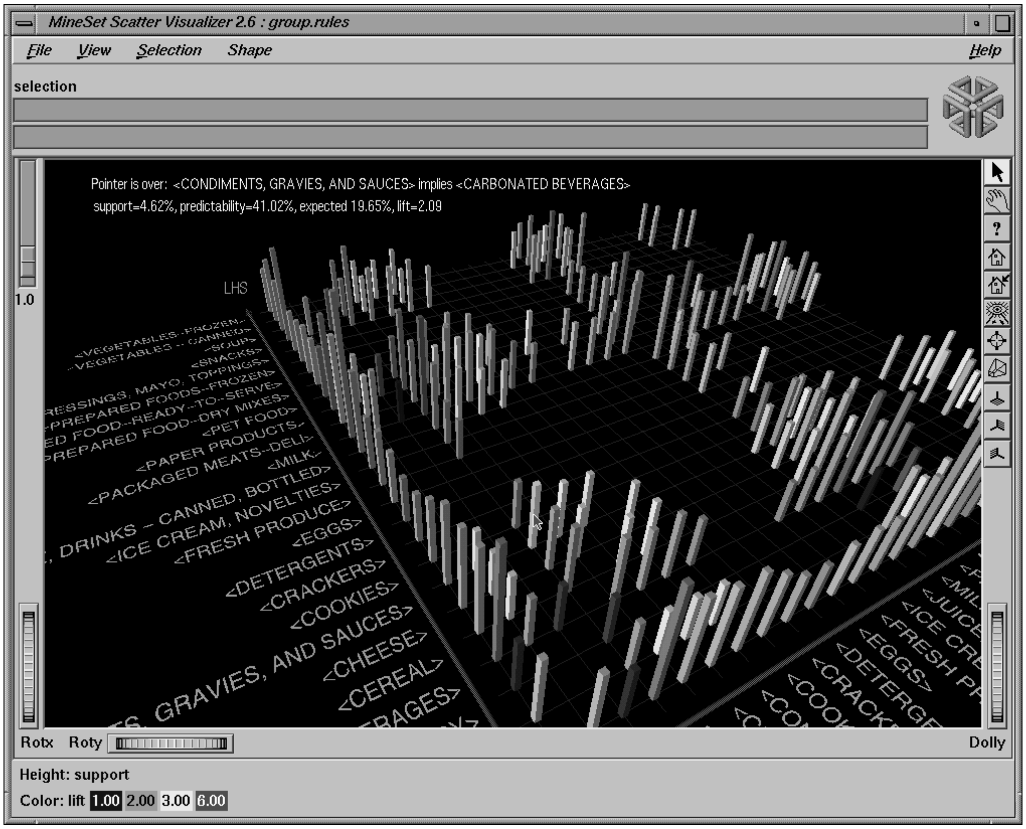
Color: lift 1.00 2.00 3.00 6.00

Figure 2-5 Initial Association Rules View When Specifying group.rules.scatterviz

The rules are presented on a grid, initially displayed with left-hand side (LHS) items displayed on the left side of the window and right-hand side (RHS) items on the right. A rule is displayed at the junction of its LHS and RHS items. The display can include bars, disks, and labels. For example, in Figure 2-5, 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. (To view the grid and rules at closer range, use the Dolly thumbwheel, described in "Thumbwheels" in Chapter 6 of the *MineSet User's Guide*.)

You can change the labels as well as what the heights and colors of the bars and disks represent by modifying the configuration file via the Tool Manager (see Chapter 3, "The Tool Manager," in the *MineSet User's Guide*) 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.



Pointer is over: <CONDIMENTS, GRAVIES, AND SAUCES> implies <CARBONATED BEVERAGES>
support=4.62%, predictability=41.02%, expected 19.65%, lift=2.09

Height: support
Color: lift 1.00 2.00 3.00 6.00

Figure 2-6 Cursor Over a Bar Which Represents a Rule

Placing the mouse cursor over an Association Rules object as shown in Figure 2-6 causes that object's information to be displayed. The information is displayed as long as the cursor remains over the object. If you position the cursor over an object and click the left mouse button, that same information appears in the Selection Window, which is above the main window, under the selection label. In addition, the bar gets selected and appears in a separate window containing all selected rules. Multiple rules may be selected by holding down the Shift key while clicking.

This information remains visible until another object is selected, or until no object is selected (if you click the black background). Using the mouse, you can cut and paste text from the selection window into other applications, such as reports or databases.

Drill Through

The drill through expression is determined by "anding" together 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, MineSet shows all the records that satisfy the rule.

See Chapter 18, "Selection and Drill Through," in the *MineSet User's Guide* for more information about drill through.

External Controls

Several external controls surround the main window, including buttons and thumbwheels. (These are the same as those in other MineSet visualization tools and are described in "Buttons" and "Thumbwheels" in Chapter 6 of the *MineSet User's Guide.*)

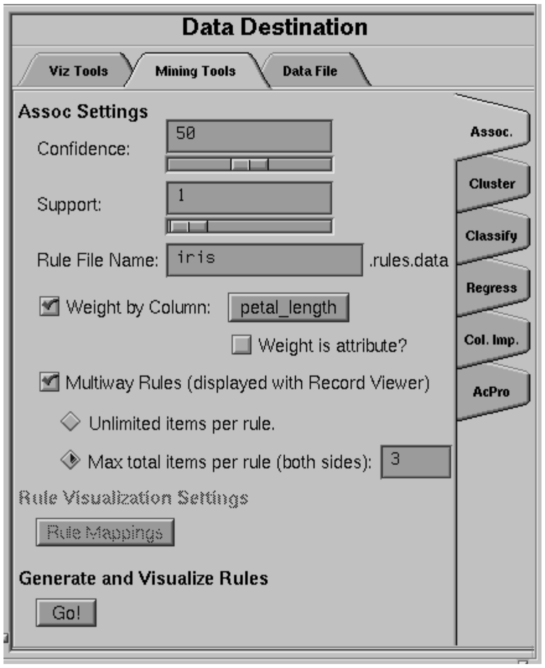
Pulldown Menus

Since association rules are displayed using the Scatter Visualizer, the pulldown menus are documented in "Pulldown Menus," in Chapter 7 of the *MineSet User's Guide.*

Multiway Association Rules

In some cases, it is useful to have more complex rules that have multiple items on the LHS and/or the RHS. These are multiway association rules. Figure 2-7 illustrates the Tool Manager Association panel configured for multiway rules generation.

If you check the *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."



Assoc Settings

Confidence: 50

Support: 1

Rule File Name: iris .rules.data

☑ Weight by Column: petal_length

□ Weight is attribute?

☑ Multiway Rules (displayed with Record Viewer)

◇ Unlimited items per rule.

◆ Max total items per rule (both sides): 3

Rule Visualization Settings

Generate and Visualize Rules

Figure 2-7 Initial Tool Manager Window for 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.

Sample Files

The provided sample data and configuration files demonstrate the features and capabilities of the Association Rules tool.

Sample Rules Visualization Files

The following sample rules and configuration files are provided for visualization. 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 */usr/lib/MineSet/scatterviz/examples* directory:

- *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.

- *people94.rules.data* and *people94.rules.scatterviz*

  These files provide the generated rules and configuration specifications for a census database, 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 database from Germany, showing associations between credit history, employment, savings, and other variables.

Converting from .ruleviz to .scatterviz

If you have existing *.ruleviz* files that you wish to convert to *.scatterviz* format, there are a few simple modifications you need to make. This can be done by editing the existing *.ruleviz* file and saving it as a *.scatterviz* file. Example 2-1 and Example 2-2 show the differences between the *.ruleviz* and *.scatterviz* formats. Example 2-2 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 2-1 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 2-2 group.rules.scatterviz

```
MineSet 2.6
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;
}
```

Chapter 3

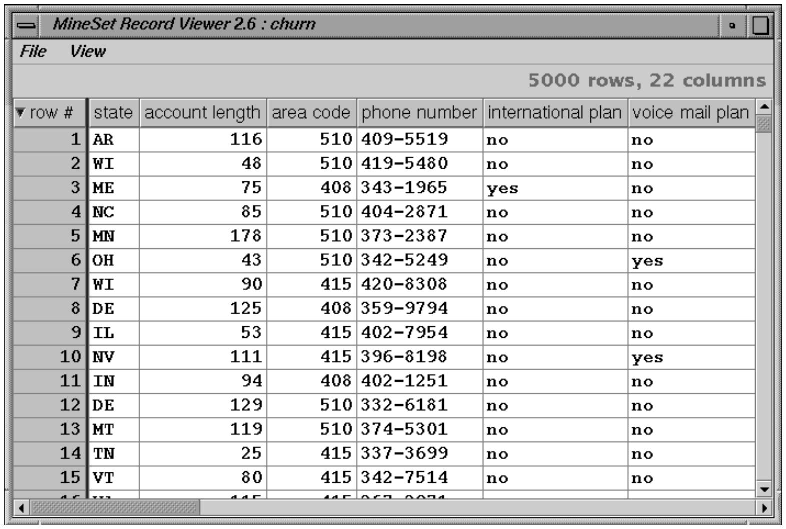
Using the Record Viewer

This chapter discusses the capabilities of the Record Viewer, and contains the following sections:

Overview of the Record Viewer

The Record Viewer allows you to view your data directly, as Figure 3-1 shows. 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
- Renumber the rows after sorting or filtering
- Search for a given value
- Save your manipulated file in a number of formats



5000 rows, 22 columns

▾ row #	state	account length	area code	phone number	international plan	voice mail plan
1	AR	116	510	409-5519	no	no
2	WI	48	510	419-5480	no	no
3	ME	75	408	343-1965	yes	no
4	NC	85	510	404-2871	no	no
5	MN	178	510	373-2387	no	no
6	OH	43	510	342-5249	no	yes
7	WI	90	415	420-8308	no	no
8	DE	125	408	359-9794	no	no
9	IL	53	415	402-7954	no	no
10	NV	111	415	396-8198	no	yes
11	IN	94	408	402-1251	no	no
12	DE	129	510	332-6181	no	no
13	MT	119	510	374-5301	no	no
14	TN	25	415	337-3699	no	no
15	VT	80	415	342-7514	no	no

Figure 3-1 The Record Viewer

Starting the Record Viewer

There are three ways to start the Record Viewer:

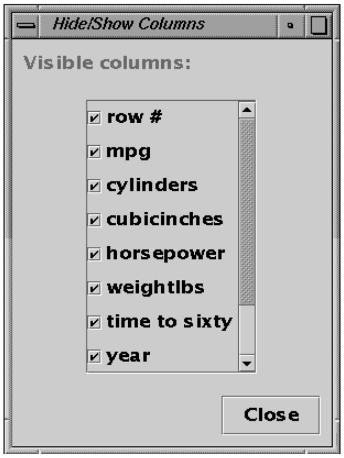
- From the Tool Manager:
  - Click the Viz Tools tab in the Data Destination panel.
  - Choose Record Viewer from the Tool popup menu.
- Double-click the Record Viewer icon, which is in the MineSet page of the icon catalog. The icon is labeled *recordview*. Because no file is specified, the start-up screen requires you to use File > Open to select one.
- Enter this command at the UNIX shell command-line prompt:

  `recordview` `[` *filename* `]`

Manipulating the Columns

There are three ways to manipulate the columns in the Record Viewer:

- You can resize columns by clicking anywhere on the right column divider and dragging the divider to the position you want.
- You can rearrange the columns by clicking and dragging the title cell of the column to its new location.
- You can hide columns by choosing “Hide/Show columns” from the View menu and deselecting the columns you wish to hide. You can show the columns again by reselecting them. Figure 3-2 shows the Hide/Show Columns panel.



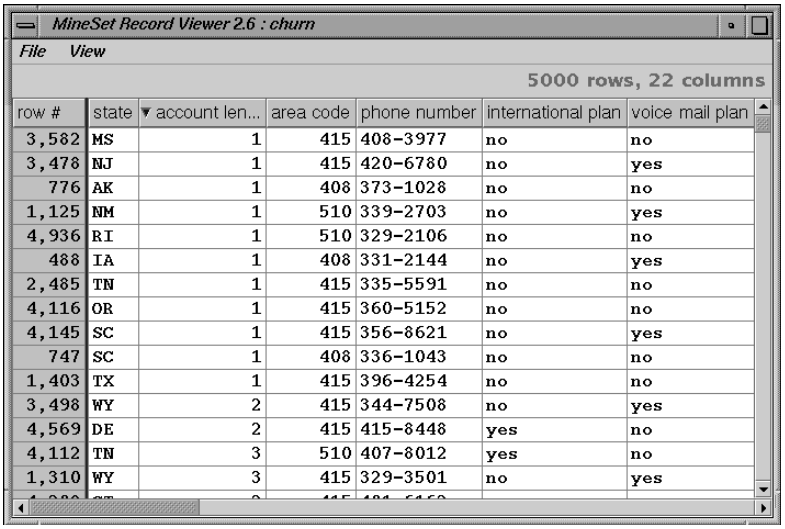
Visible columns:

☑ row #
☑ mpg
☑ cylinders
☑ cubicinches
☑ horsepower
☑ weightlbs
☑ time to sixty
☑ year

Figure 3-2 Record Viewer Hide/Show Column Panel

Sorting Records

The Record Viewer allows you to sort your records by the values in any given column. To sort a column, click on the title cell. To reverse the sort, click the title cell again. To return the records to their original order, click the row # column title. Figure 3-3 shows the same data as in Figure 3-1, sorted by the account length column.



5000 rows, 22 columns

row #	state	▾ account len...	area code	phone number	international plan	voice mail plan
3,582	MS	1	415	408-3977	no	no
3,478	NJ	1	415	420-6780	no	yes
776	AK	1	408	373-1028	no	no
1,125	NM	1	510	339-2703	no	yes
4,936	RI	1	510	329-2106	no	no
488	IA	1	408	331-2144	no	yes
2,485	TN	1	415	335-5591	no	no
4,116	OR	1	415	360-5152	no	no
4,145	SC	1	415	356-8621	no	yes
747	SC	1	408	336-1043	no	no
1,403	TX	1	415	396-4254	no	no
3,498	WY	2	415	344-7508	no	yes
4,569	DE	2	415	415-8448	yes	no
4,112	TN	3	510	407-8012	yes	no
1,310	WY	3	415	329-3501	no	yes

Figure 3-3 Churn Data Sorted by Account Length

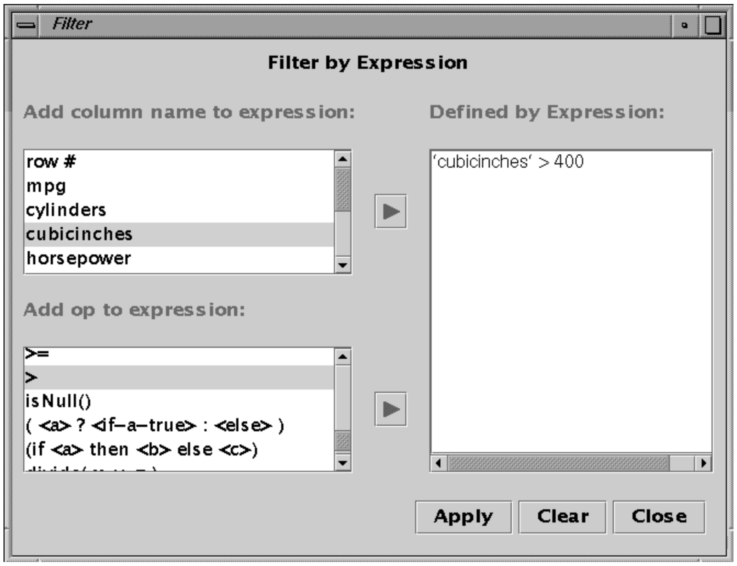
Filtering Data

The Record Viewer allows you to filter your data so that you need only see the range of values that interests you. To filter your data, choose Filter panel (Figure 3-4) from the View menu. Figure 3-5 shows the cars data file filtered to show only those cars with engines larger than 400 cubic inches.

To remove the filtering, clear the expression in the expression window and click *Apply* or choose Remove filter from the Record Viewer View menu.

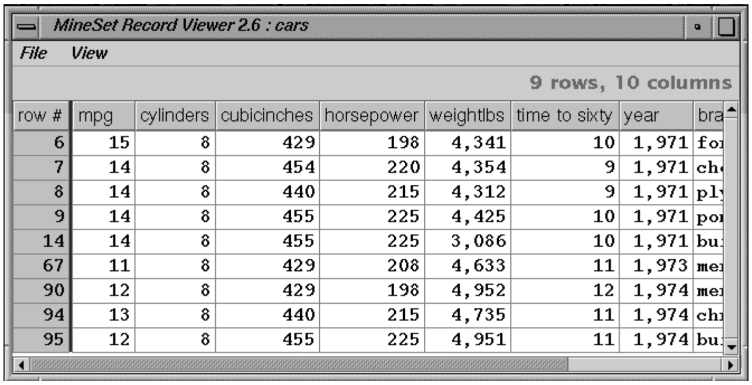
You may have as many filter panels open as you wish. To apply more than one filter at a time, first apply one, renumber the rows (see “Renumbering Rows” on page 48), then apply the next.

Note: Renumbering the rows cannot be undone. To return to your original data, you must reopen the file.



Defined by Expression: 'cubicinches' > 400

Figure 3-4 Filter Panel



9 rows, 10 columns

row #	mpg	cylinders	cubicinches	horsepower	weightlbs	time to sixty	year	bra
6	15	8	429	198	4,341	10	1,971	for
7	14	8	454	220	4,354	9	1,971	ch
8	14	8	440	215	4,312	9	1,971	ply
9	14	8	455	225	4,425	10	1,971	po
14	14	8	455	225	3,086	10	1,971	bu
67	11	8	429	208	4,633	11	1,973	me
90	12	8	429	198	4,952	12	1,974	me
94	13	8	440	215	4,735	11	1,974	ch
95	12	8	455	225	4,951	11	1,974	bu

Figure 3-5 Cars Data Filtered by Cubic Inches

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 Renumber rows from the View menu.

Searching in the Record Viewer

The Record Viewer allows you to search for a value in your data. To open the Search panel (Figure 3-6), choose Search panel from the View menu. To search, type in the value, highlight the columns you want to search in, and click *Find Next* or *Find Previous*.

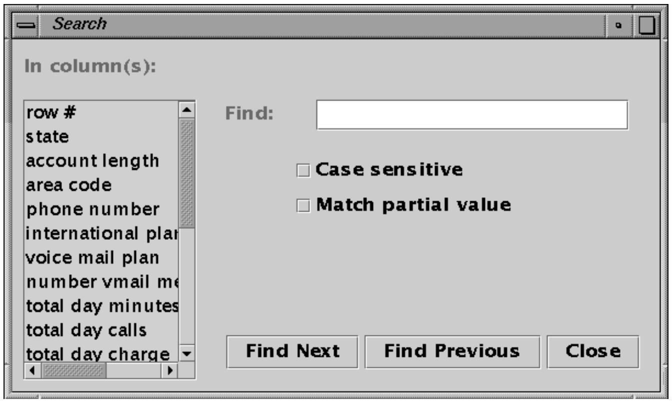
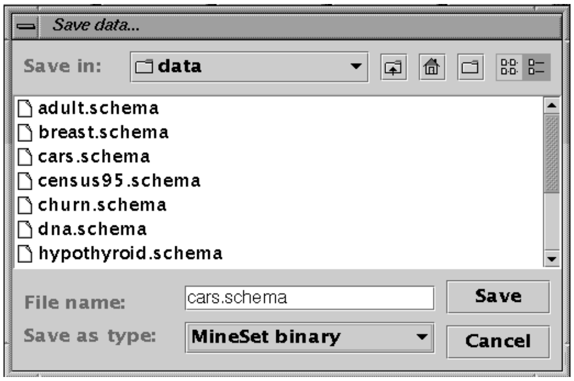


Figure 3-6 Search Panel

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 Save or Save As from the File menu.

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 (Figure 3-7), where you can enter the desired filename and the type of format you wish.



Save in: data

File name: cars.schema

Save as type: MineSet binary

Figure 3-7 Save Data Screen

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. Figure 3-8 shows the “Save as type” popup menu.

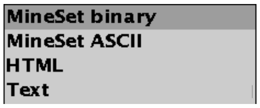


Figure 3-8 “Save as type” Popup Menu

Chapter 4

MineSet User's Guide Errata

This chapter corrects some of the errors in the *MineSet User's Guide.* All of the chapter and section references refer to the *User's Guide*.

This chapter contains the following sections:

MineSet Help

Context-sensitive help is available throughout Mineset. Type `shift-F1` to turn the mouse cursor into a question mark, then click on the area for which you would like help.

Chapter 3, "The Tool Manager"

In Chapter 3, "The Tool Manager," two sections, "The Add Column Button" and "The Filter Button" point the reader to the Tree Visualizer Appendix (Appendix B) for a further explanation of the available operators and functions. Actually, the Tree Visualizer and the Tool Manager have slightly different options available.

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:

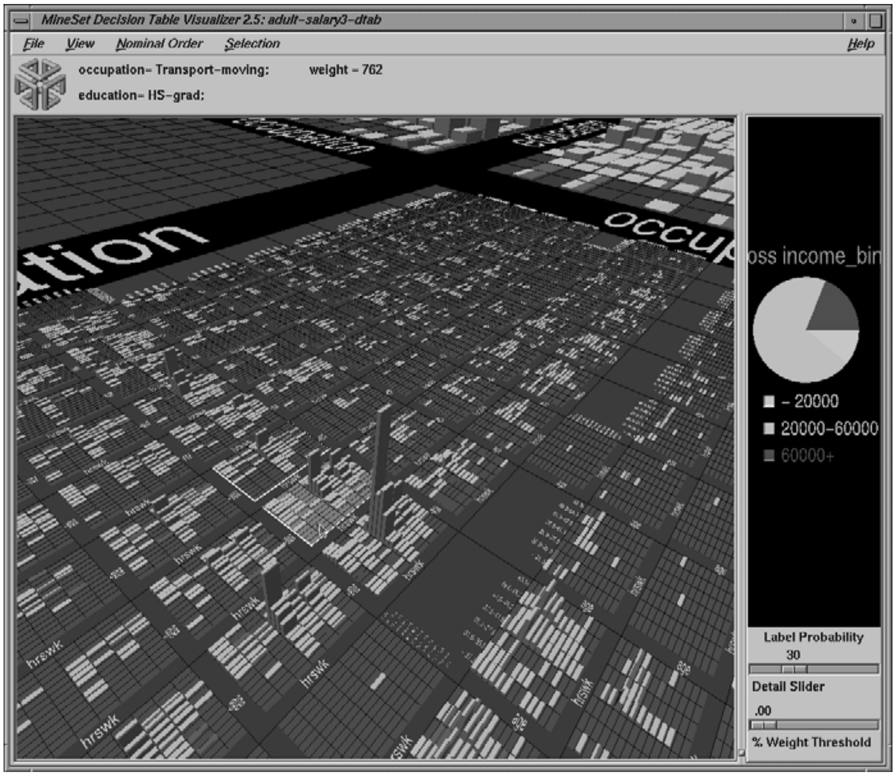
+	addition
-	subtraction
*	multiplication
/	division
( )	parentheses for grouping expressions
%	modulo (remainder after division)
!	logical NOT
~	logical NOT
&&	logical AND
| |	logical OR
^	logical exclusive OR
==	equal to
!=	not equal to
<=	less than or equal to
<	less than
>=	greater than or equal 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 ( )	standard if/then/else
( ) ? ( ) : ( )	C syntax if/then/else
divide(x, y, z)	divide x by y, and give value z if y is 0

Chapter 14, “Inducing and Visualizing the Decision Table”

In Chapter 14, “Inducing and Visualizing the Decision Table,” Figure 14-13 is incorrect. Figure 4-1 shows the correct illustration.



MineSet Decision Table Visualizer 2.5: adult-salary3-dtab

occupation= Transport-moving: weight = 762

education= HS-grad:

Label Probability 30

Detail Slider .00

% Weight Threshold

Figure 4-1 Correction for Figure 14-13, Closer Inspection of the Adult Dataset

Chapter 15, "Inducing and Visualizing the Regression Tree"

In Chapter 15, "Inducing and Visualizing the Regression Tree," the "Decision Nodes" subsection of the "Visualizing the Regression Tree" section should read as follows:

Decision nodes specify the attribute that is tested at the node. Values (or ranges of values) against which the attributes are tested are shown at the lines. Each possible value for the attribute matches exactly one line. For example, the root of the Regression Tree in Figure 15-1 tests the attribute age; the two lines emanating from the node partition values for that attribute (<= 27.5, > 27.5) so that every possible value matches either the right branch or the left branch. If the value is unknown and there is no line labeled with a question mark, the mean or median label value at the current node is predicted.

Also in Chapter 15, the second and third paragraphs of the "Node Information," subsection of the "Visualizing the Regression Tree" section should be deleted.

Appendix A, "Flat File Support for MineSet"

In "The .schema File" section of Appendix A, the "Data Statements" subsection lists the data types allowed in data statements. Enumerations, fixed arrays, and enumerated arrays were inadvertently left out of the list (they are described in later subsections of Appendix A, however).

The following is the corrected wording:

Data Statements

The data statements declare the columns in the data file. The columns must be declared in the order they appear in the data file. The format of most data statements is:

`type` *name*`;`

where type is **int**, **float**, **double string**, **dataString**, **date**, and **fixedString**(*n*), where *n* is an integer representing the width of the string; *name* is the variable name. Unlike in C, only one variable can be declared per statement.

Other supported types include enumerations, fixed arrays, and enumerated arrays. These data types must be declared inside the 'input' section, before the declaration of the specific column.

Appendix D, “Creating Data and Configuration Files for the Scatter Visualizer”

In Appendix D, “Creating Data and Configuration Files for the Scatter Visualizer,” the “The Max Clause” subsection of the “Size Statement” section is incorrect. The correct wording is as follows:

The Max Clause

Normally, the size variable is mapped to the size of the entities, so that the biggest entity has a size of 5. This size can be changed by specifying a different value. If there is no size variable, the default maximum size is 5. The **max** clause has the form:

```
max float
```

Appendix E, “Creating Data and Configuration Files for the Splat Visualizer”

In Appendix E, “Creating Data and Configuration Files for the Splat Visualizer,” the “Opacity Statement” section is incorrect. The correct version is as follows:

Opacity Statement

In the Splat Visualizer, the opacity is based on counts, or more generally, record weights.

If a column is mapped to this requirement, it is used to weight each record (rather than using 1) when computing a value for the opacity. Thus, if you had a column with values for population, density, or the result of a count aggregation, you might want to map this column to the opacity (weight) requirement. If you had no such column, the requirement can be left unmapped, and a column of 1’s is used by default.

The **opacity** statement describes how a field of data is mapped to the opacity of the splats. The **opacity** statement consists of a series of clauses, separated by commas:

```
opacity clause1, clause2,...
```

Alternatively, the clauses can be given in separate opacity statements.

The Opacity Variable

The first clause normally contains the name of a field to be mapped to opacity. The field must be of a number type (**int**, **float**, or **double**), of which **float** is the most efficient.

The Max Clause

The **max** clause allows you to alter the initial opacity setting for the scene. The most opaque splat in the scene will match the value specified in this **max** clause. The default is 1. The **max** clause has the form:

```
max float
```

Appendix A

Using the Association Rules Generator With Transaction-Style Data

This appendix replaces Appendix F, "Creating Data and Configuration Files for the Rules Visualizer," in the *MineSet User's Guide*, and describes how to use two MineSet commands, *assoccvt* and *assocgen*, to find the association rules in a file presented in transaction-style format. To generate association rules from data stored in a table, use the Tool Manager interface, described in Chapter 2, "Using the Association Rules Tool."

This appendix contains the following sections:

- "About Transaction-Style Data" on page 58
- "Association Rules Generator" on page 62
- "Rules Visualization" on page 68

Note: The programs used in this appendix are installed on the MineSet server. Only rules with lift greater than or equal to 1 are produced by the command-line Association Rules Generator.

The examples used in this appendix can be found in the */usr/lib/MineSet/assoccvt/examples/* and */usr/lib/MineSet/assocgen/examples/* directories. Descriptions and instructions for use can be found in the README file in these directories.

Note: Read Chapter 2, "Using the Association Rules Tool," before using this appendix.

About Transaction-Style Data

MineSet provides tools for analyzing and visualizing data that is stored in a tabular format, either in a database, or in a flat file. The MineSet Tool Manager allows you to generate and visualize the association rules in a table. If your data is in transaction format, you can use the utilities described in this appendix to find the association rules in your data. Transaction-style data is data in a flat file in which transactions are split across several rows. Each row has two columns; one corresponds to a transaction identifier (transaction ID), and the other corresponds to an item in the transaction.

For example, in a point-of-sale file representing supermarket transactions, a file might look like this:

```
10012 wheat bread
10012 beer
10012 tortilla chips
10012 eggs
10013 cereal
10013 chicken
10014 all-purpose flour
```

Rows are grouped by transaction ID. With this style of data, MineSet finds associations between items present in the same transaction. For example, the data may contain the association “eggs implies beer.”

There are three steps to finding the association rules present in transaction-style data:

- Converting the data to MineSet’s format (the *assoccvt* step)
- Running the Association Rules Generator (the *assocgen* step)
- Loading the results into the MineSet Tool Manager for further manipulation (for example, visualization using the Scatter Visualizer or the Record Viewer)

Association Data Converter Requirements

The association data converter requires:

- A raw data file (consisting of your own data for running associations)
- A format file, which describes the raw data file’s format

Raw Data File

The raw data file must be in the format shown in “About Transaction-Style Data” on page 58, in which there is one transaction ID and one item per row. Neither the transaction ID nor the item need be contiguous within the row of data, however. In addition, the records in the file must be of exactly the same length, and they must be grouped by transaction ID (though they need not be sorted).

Format File

The format file specifies the format of the raw data file to the association data converter. This format file must contain the following items in the order shown:

- The letter “S” to indicate the file type
- The number of bytes in each row, excluding the end-of-line character. Each row in the data file must have this many bytes.
- The number of fields that make up the transaction ID
- The total number of bytes in the transaction ID
- The offset and number of bytes for each field that makes up the transaction ID
- The number of fields that make up the item
- The total number of bytes in the item
- The offset and length in bytes for each field that makes up the item
- A flag indicating whether or not there is a field describing the item. If so, this field will be output in the generated rules along with the name. This should be either a 0 (meaning No) or 1 (meaning Yes)
- If the description flag is 1, the following are also required:
  - Number of fields that make up the description
  - Total number of bytes in the description
  - Offset and length in bytes for each field that makes up the description

Note: Each column number is zero-based.

Most data files use only one field each for the item and the transaction identifier. For the example data listed above, assuming that each line is 80 characters wide (plus one for the end-of-line character), the format file would be:

```
S
80
1 5
0 5
1 74
6 74
0
```

This format file allows for a great deal of flexibility. For instance, there need be no separator between the transaction identifier and the item. The two may even be overlapping, as in:

```
bread  10012wheat  25
apple  10012fuji   25
banana 10012bunch  25
```

In this case, both the transaction ID and the item contain two fields of different lengths. If the total line width is 21 bytes, then the format file would be:

```
S
21     (21 bytes per line, not including end-of-line character)
2 7    (transaction ID is two fields, 7 bytes total)
7 5    (first transaction ID field is 5 bytes starting at column 7)
19 2   (second transaction ID field is 2 bytes starting at column 19)
2 13   (item is two fields, 13 bytes total)
0 6    (first item field is 6 bytes starting at column 0)
12 7   (second item field is 7 bytes starting at column 12)
0      (no description fields)
```

Association Data Converter Command-Line Operation

To find the association rules in your data, you must first convert the data into an intermediate binary format, which the Association Rules Generator uses to find the rules. This conversion step uses the raw data file and format file, and produces two new files:

- The *output data file*, containing the converted data
- The *output names* file, containing auxiliary descriptor information used by the Association Rules Generator

The *assoccvt* program converts the data. Its usage is:

```
assoccvt [-ifile raw] [-ofile binary] format names
```

where *raw* is the name of the raw data file, *binary* is the name of the produced binary data file, *format* is the name of the format file, and *names* is the name of the output names file. If the **-ifile** parameter is omitted, standard input is used instead. Similarly, if the **-ofile** parameter is omitted, the standard output is used instead.

Association Data Converter Examples

The following command illustrates the use of the association data converter on the example file in */usr/lib/MineSet/assoccvt/examples*. The file *sing.data* is an example of data in transaction format and has some simple grocery store transactions. Each line has a transaction number and the name of an item bought in that transaction. The format of this file is described by *sing.format*.

```
assoccvt -ifile sing.data -ofile sing.bin sing.format sing.names
```

To test whether the files for data conversion are correctly installed, run the preceding command from the shell command line. Then, using the UNIX *diff* command, compare the files created to those with the same name in */usr/lib/MineSet/assoccvt/examples*. Compare *sing.bin* with */usr/lib/MineSet/assoccvt/examples/sing.bin*, and compare *sing.names* with */usr/lib/MineSet/assoccvt/examples/sing.names*.

Association Rules Generator

The Association Rules Generator takes items in a set of data and generates association rules from them. The required input files are described in the following subsections. The output of the Association Rules Generator is a specially formatted rules file, which can be loaded into MineSet as a flat file for examination.

Association Rules Generator File Requirements

The Association Rules Generator program, *assocgen*, requires:

- A *data file* in the internally required format
- A *names file* in the internally required format

Association Rules Generator Command-Line Operation

Rules are generated by invoking the *assocgen* command, along with one or more parameters. Options fall into one of the following categories:

- *Rule Generation Options*—control the process of rule generation.
- *Rule Restriction Options*—place restrictions on the set of generated rules.

The **-ropts** string separates the two sets of options. This string is required if any options from the second set are used.

An example rule generation command line might be:

```
assocgen -prev 20 -tran sing.bin -ropts -names sing.names \
        -rout sing.rules
```

See “Rule Generation Options” and “Rule Restriction Options” for explanations of the parameters.

Note: In the *assocgen* program, support is called prevalence, and confidence is called predictability. Therefore, the parameters for the support and confidence thresholds are **-prev** and **-pred**.

Rule Generation Options

Table A-1 lists the set of options for controlling the rule-generation process. A description of each option follows the table. In the following description, **%s** represents a string-valued parameter, **%d** an integer-valued parameter, and **%f** a floating point-valued parameter.

Table A-1 Options for Controlling Rule Generation

Option Format	Default Value	Comments
-tran %s	(stdin)	Data file path
-prev %f	(1.0)	Support threshold (as a percentage)
-uniq %d		Number of items in dataset
-dir %s	(/usr/tmp)	Directory for temporary files
-tprefix %s	(A_)	Prefix for temporary files
-msg %s	(*assocgen.msg*)	Message file

-tran %s Specifies the path for the file. By default, the file is read from *stdin*.

-prev %f Specifies the minimum support threshold as a percentage of the total number of records. The default is 1.0%. If the support threshold results in a minimum count less than 3, an error message is displayed, and no rules are generated.

-uniq %d Specifies the number of unique or distinct items across all records (if known). Specifying this (or an upper bound) speeds processing.

-dir %s Specifies the directory in which to store temporary files, including the message file (see **-msg,** below). The default is the current directory.

-tprefix %s Specifies the prefix to be used for temporary files, except the message file (see **-msg,** below). The default prefix is *A_*.

-msg %s Specifies the message file, which contains diagnostic output. The default is *assocgen.msg*.

Rule Restriction Options

Table A-2 lists the set of options for restricting generated rules. Options in this set are used after those listed in Table A-1 and separated on the command line from the former options by **-ropts**. A description of each option follows the table.

Table A-2 Options for Restricting Generated Rules

Option Format	Default Value	Comments
-pred %f	(50.0)	Minimum confidence (as a percentage)
-names %s		Name of file containing item descriptions
-rout %s	(stdout)	Name of file in which to output rules

-pred %f Specifies the minimum confidence threshold for rules. Rules with a confidence below this value are not generated. The default is 50%.

-names %s Specifies the name of the file that contains the descriptions of the items. This is typically the names file created during the *assoccvt* step.

-rout %s Specifies the name of the file to which rules are to be written. If this is not specified, rules are written to *stdout*.

Association Rule Example

The data listed in Table A-3 is an example of market basket data. This data can be found in the file */usr/lib/MineSet/assoccvt/examples/sing.data*.

Table A-3 Data Example

Transaction ID	Item
10	Jam
10	Eggs
10	Chips
10	Bread
10	Butter
10	Milk

Table A-3 (continued) Data Example

Transaction ID	Item
20	Soda
20	Eggs
20	Butter
20	Bread
30	Soda
30	Eggs
30	Milk
30	Bread
30	Butter
40	Eggs
40	Chips
40	Juice
40	Bread
50	Milk
50	Chips
50	Bread
50	Beer
60	Soda
60	Juice
60	Beer
70	Beer
70	Chips
70	Wine
80	Juice

Table A-3 (continued) Data Example

Transaction ID	Item
80	Cookies
80	Chips
90	Chips
90	Cookies
90	Milk
95	Bread
95	Cookies
95	Milk

You can generate rules from the data in Table A-3 by using first *assoccvt* (see "Association Rules Generator Command-Line Operation" on page 62), and then running the *assocgen* command:

```
assocgen -prev 20 -tran sing.bin -ropts -names sing.names -rout sing.rules
```

The rules file that is output has the following format:

```
1    1    30.0000 100.00 40.00 Butter Eggs
```

The fields in each line correspond to:

- The number of items on the LHS of the rule (always 1)
- The number of items on the RHS of the rule (always 1)
- The support
- The confidence
- The expected confidence
- The name (or code) of the item on the LHS
- The name (or code) of item on the RHS

The expected confidence is the frequency of occurrence of the RHS items. The difference between expected confidence and observed confidence is a measure of the increase in predictive power due to the presence of the LHS. Expected confidence gives an indication of what the confidence would be if there were no relationship between the items.

For a further description of the relationships between support, confidence, expected confidence, and lift, see Chapter 2, "Using the Association Rules Tool."

If the minimum support threshold is 20% (8 records out of 38 in the example below), and the default minimum confidence threshold is 50%, the *assocgen* program generates the set of rules shown in Table A-4.

Table A-4 Rule Generation Example 1

1	1	30.0000	100.00	40.00	Butter	Eggs
1	1	40.0000	66.67	40.00	Bread	Eggs
1	1	20.0000	66.67	40.00	Soda	Eggs
1	1	20.0000	66.67	60.00	Juice	Chips
1	1	20.0000	66.67	60.00	Beer	Chips
1	1	20.0000	66.67	60.00	Cookies	Chips
1	1	30.0000	60.00	60.00	Milk	Chips
1	1	40.0000	100.00	60.00	Eggs	Bread
1	1	30.0000	100.00	60.00	Butter	Bread
1	1	40.0000	80.00	60.00	Milk	Bread
1	1	20.0000	66.67	60.00	Soda	Bread
1	1	30.0000	75.00	30.00	Eggs	Butter
1	1	20.0000	66.67	30.00	Soda	Butter
1	1	30.0000	50.00	30.00	Bread	Butter
1	1	40.0000	66.67	50.00	Bread	Milk
1	1	20.0000	66.67	50.00	Butter	Milk
1	1	20.0000	66.67	50.00	Cookies	Milk

Table A-4 (continued) Rule Generation Example 1

1	1	30.0000	50.00	50.00	Chips	Milk
1	1	20.0000	50.00	50.00	Eggs	Milk
1	1	20.0000	66.67	30.00	Butter	Soda
1	1	20.0000	50.00	30.00	Eggs	Soda

Rules Visualization

The rules visualizer graphically displays, using the Scatter Visualizer, the rules resulting from the Association Rules Generator. The rules visualization requires:

- A configuration file, which specifies various display parameters (the *.schema* file)
- A rules file in the internally required format (the *.rules* file)

Rules Visualization File Requirements

The rules generated by *assocgen* can be loaded into MineSet as a flat file for further analysis and visualization. You can use the Scatter Visualizer, for example, to visualize the rules in the same way that the Tool Manager is used to visualize rules generated from tabular data.

The Schema File

To visualize rules, you need a MineSet *.schema* file corresponding to the *.rules* file you just created. An example *.schema* file can be found in */usr/lib/MineSet/assocgen/examples/rules.schema*. Edit this file to specify the name of the file containing the rules. Once you've done this, you can load the *.schema* file into MineSet as a flat file. The *.schema* file describes the columns in the rules file:

```
MineSet 2.6
# Example schema for loading rules into MineSet

input {
        options backslash on;
        file “assoc.rules.data”;
        int `lhs size`;
        int `rhs size`;
        float support;
        float confidence;
        float `expected confidence`;
        string LHS;
        string RHS;
}
```

Change the name of the file in the `file` clause to your rules file. Once you load the data into MineSet, it may be useful to add a column `` `lift` ``, of type `double`, with the expression `` `confidence`/`expected confidence` ``.

The Rules File

The rules file is generated by the Association Rules Generator (see “Association Rules Generator” on page 62).

Index

Tell Us About This Manual

As a user of Silicon Graphics products, you can help us to better understand your needs and to improve the quality of our documentation.

Any information that you provide will be useful. Here is a list of suggested topics:

- General impression of the document
- Omission of material that you expected to find
- Technical errors
- Relevance of the material to the job you had to do
- Quality of the printing and binding

Please send the title and part number of the document with your comments. The part number for this document is 007-3915-001.

Thank you!

Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
  - On the Internet: techpubs@sgi.com
  - For UUCP mail (through any backbone site): *[your_site]*!sgi!techpubs
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 650-932-0801
- To send your comments by **traditional mail**, use this address:

  Technical Publications
  Silicon Graphics, Inc.
  2011 North Shoreline Boulevard, M/S 535
  Mountain View, California 94043-1389