Chapter 6

Interactive Graphics

There is more to exploring data than simply generating textual and statistical summaries and graphical plots. As we have begun to see, R has some very significant capabilities for generating graphics that assist in revealing the story our data is telling us and then helps us to effectively communicate that story to others. However, R is specifically suited to generating static graphics—that is, as Wickham (2009) says, "there is no benefit displaying on a computer screen as opposed to on a piece of paper" when using R's graphics capabilities.

R graphics were implemented with the idea of presenting the data visually rather than interacting with it. We write scripts for the display. We then go back to our script to fine-tune or explore different options for the displayed data. This is great for repeatable generation of graphics but not so efficient for the "follow your nose" or "ad hoc reporting" approach to quick and efficient data exploration.

Being able to easily interact with a plot can add significantly to the efficiency of our data exploration and lead to the discovery of interesting and important patterns and relationships. Data miners will need sophisticated skills in dynamically interacting with the visualisations of data to provide themselves with significant insights. Whilst software supports this to some extent, the true insights come from the skill of the data miner. We must take time to explore our data, identify relationships, discover patterns, and understand the picture painted by the data.

Rattle provides access to two very powerful R packages for interactive data analysis, **latticist** (Andrews, 2010) and GGobi, the latter of which is accessed via **rggobi** (Lang et al., 2011). These can be initiated through the Interactive option of the Explore tab (Figure 6.1). We will introduce

G. Williams, *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery*, Use R, DOI 10.1007/978-1-4419-9890-3_6, © Springer Science+Business Media, LLC 2011

each of the tools in this chapter. Note that each application has much more functionality than can be covered here, and indeed GGobi has its own book (Cook and Swayne, 2007), which provides good details.

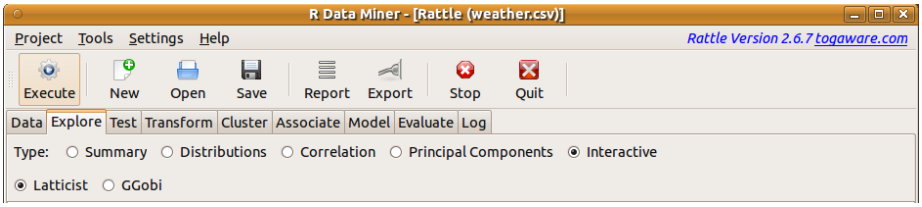

Figure 6.1: The Explore tab's Interactive option can initiate a **latticist** or GGobi session for interactive data analysis.

6.1 Latticist

Latticist (Andrews, 2010) provides a graphical and interactive interface to the advanced plotting capabilities of R's **lattice** (Sarkar, 2008). It is written in R itself and allows the underlying R commands that generate the plots to be directly edited and their effect immediately viewed. This then provides a more interactive experience with the generation of R plots. Select the Latticist radio button of the Interactive option of the Explore tab and then click the toolbar's Execute button to display **latticist**'s window, as shown in Figure 6.2.

From the R Console, we can use `latticist()` to display the same interactive window for exploring the *weather* dataset:

```
> library(latticist)
> latticist(weather)
```

With the initial Latticist window, we immediately obtain an overall view of some of the story from our data. Note that, by default, from Rattle, the plots show the data grouped by the target variable `RainTomorrow`. We see that numeric data is illustrated with a density plot, whilst categoric data is displayed using dot plots.

Many of the plots show differences in the distributions for the two groups (based on whether `RainTomorrow` is `No` or `Yes`). We might note, for example, that variables `MinTemp` and `MaxTemp` (the first two plots of the top row) have slightly higher values for the observations where it

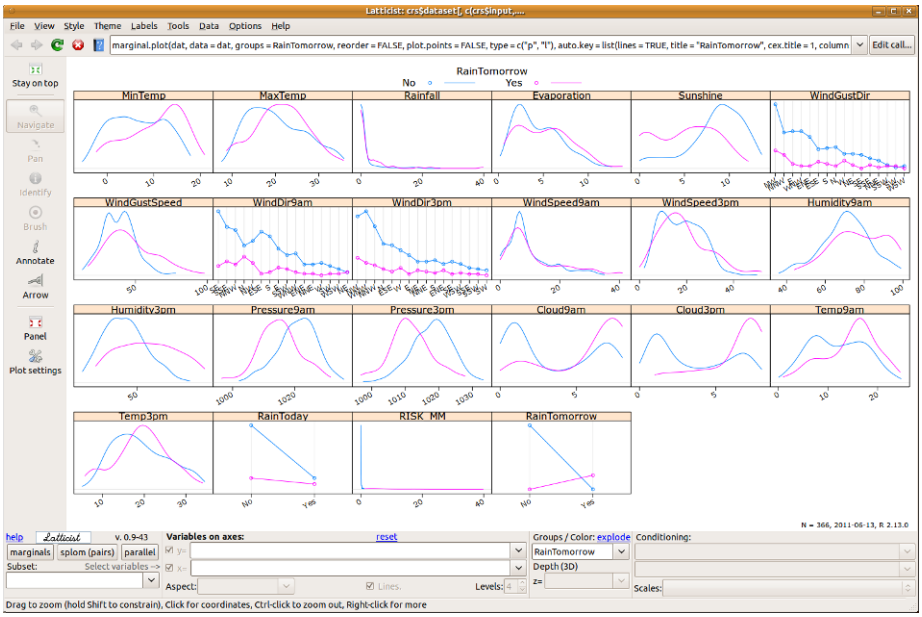

Figure 6.2: The Explore tab's Interactive option can initiate a **latticist** session for interactive data analysis..

rains tomorrow. The third plot suggests that the amount of `Rainfall` today seems to be almost identically distributed for observations where it does not rain tomorrow and those where it does. The fifth plot then indicates that there seems to be less `Sunshine` on days prior to days on which it rains.

There is an extensive set of features available for interacting with the visualisations. The actual command used to generate the current plot is shown at the top of the window. We can modify the command and immediately see the result, either by editing the command in place or clicking the Edit call... button. The latter results in the display of a small text window in which the command can be edited. There are buttons in the main window's toolbar to open the help page for the current plot, to reload the plot, and to navigate to previous plots.

The default plot is a plot of the marginal distribution of the variables. The buttons near the bottom left of the window allow us to select between marginal, splom (pairs), and parallel coordinates plots. A splom is a scatter plot matrix similar to that in Section 5.2.8. A parallel coordinates plot draws a line for each observation from one variable to the next, as in

Figure 6.3. Parallel coordinates plots can be quite useful in identifying groups of observations with similar values across multiple variables.

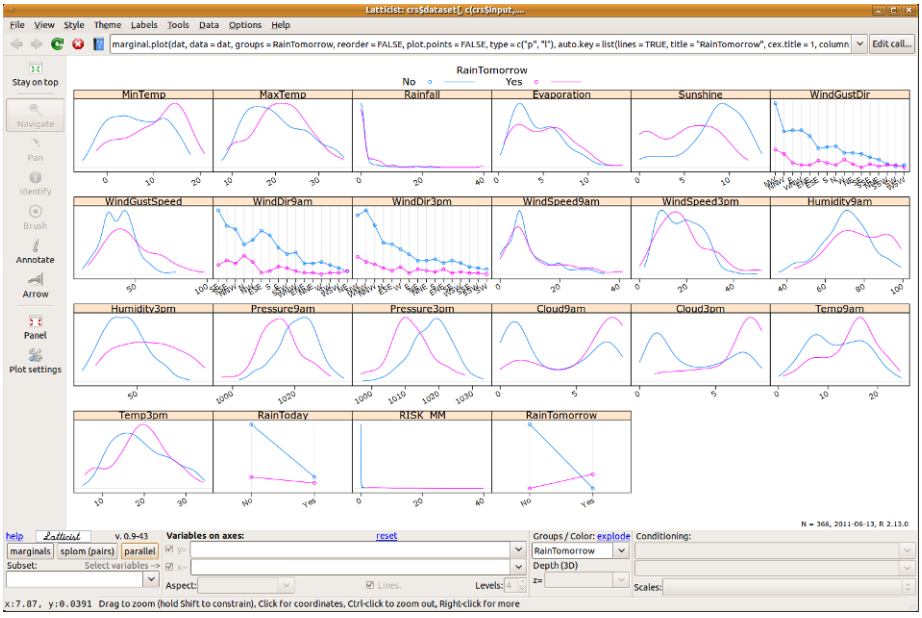

Figure 6.3: The parallel coordinates plot from **latticist**.

The parallel coordinates plot in Figure 6.3 exposes some structure in the *weather* dataset. The top variable in this case is the target variable, `RainTomorrow`. The other variables are `Sunshine`, `Rainfall`, `MaxTemp`, and `MinTemp`. Noting that each line represents a single observation (the weather details for each day), we might observe that for days when there is less sunshine it is more likely to rain tomorrow, and similarly when there is more sunshine it is less likely to rain tomorrow. We can observe a strong band of observations with no rain tomorrow, higher amounts of sunshine today, and little or no rainfall today. From there (to the remaining two variables) we observe less structure in the data.

There is a lot more functionality available in **latticist**. Exploring many of the different options through the interface is fruitful. We can add arrows and text to plots and then export the plots for inclusion in other documents. The data can be subset and grouped in a variety of ways using the variables available. This can lead to many insights, following our nose, so to speak, in navigating our way through the data. All the time we are on the lookout for structure and must remember to capture it to support the story that we find the data telling us.

6.2 GGobi

GGobi is also a powerful open source tool for visualising data, supporting two of the most useful interactive visualisation concepts, known as brushing and tours. GGobi is not R software as such[1] but is integrated with R through **rggobi** (Lang et al., 2011) and `ggobi()`. Key uses in a data mining context include the exploration of the distribution of observations for multiple variables, visualisations of missing values, exploration for the development of classification models, and cluster analysis. Cook and Swayne (2007) provide extensive coverage of the use of GGobi, particularly relevant in a data mining context.

To use GGobi from the Interactive option of the Explore tab, the GGobi application will need to be installed. GGobi runs under GNU/Linux, Mac OS/X, and Microsoft Windows and is available for download from `http://www.ggobi.org/`.

GGobi is very powerful indeed, and here we only cover some basic functionality. With GGobi we are able to explore high-dimensional data through highly dynamic and interactive graphics that include tours, scatter plots, bar plots, and parallel coordinates plots. The plots are interactive and linked with brushing and identification. Panning and zooming are supported. Data can be rotated in 3D, and we can tour high-dimensional data through 1D, 2D, and 2x1D projections, with manual and automatic control of projection pursuits.

We are also able to interact with GGobi by issuing commands through the R Console, and thus we can script some standard visualisations from R using GGobi. For example, patterns found in data using R or Rattle can be automatically passed to GGobi for interactive exploration. Whilst interacting with GGobi plots we can also highlight points and have them communicated back to R for further analysis.

Scatter plot

We can start GGobi from Rattle by clicking the Execute button whilst having selected GGobi under the Interactive option of the Explore tab, as in Figure 6.1.

We can also initiate GGobi with `rggobi()`, providing it with a data frame to load. In this example, we remove the first two variables (`Date` and `Location`) and pass on to `rggobi()` the remaining variables:

[1] A project is under way to implement the concepts of GGobi directly in R.

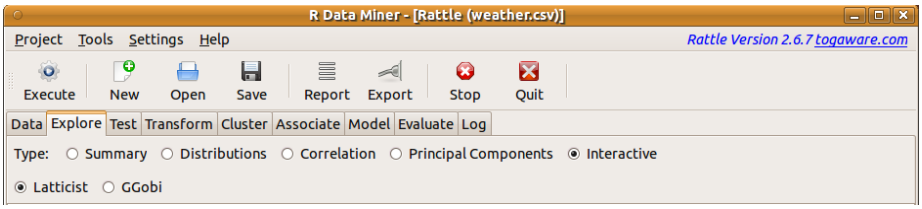

Figure 6.4: The Explore tab's Interactive option can initiate a GGobi session for interactive data analysis. Select GGobi and then click Execute.

```
> library(rggobi)
> gg <- rggobi(weather[-c(1,2)])
```

On starting, GGobi will display the two windows shown in Figure 6.5. The first provides controls for the visualisations and the other displays the default visualisation (a two-variable scatter plot of the first two variables of the data frame supplied, noting that we have removed `Date` and `Location`).

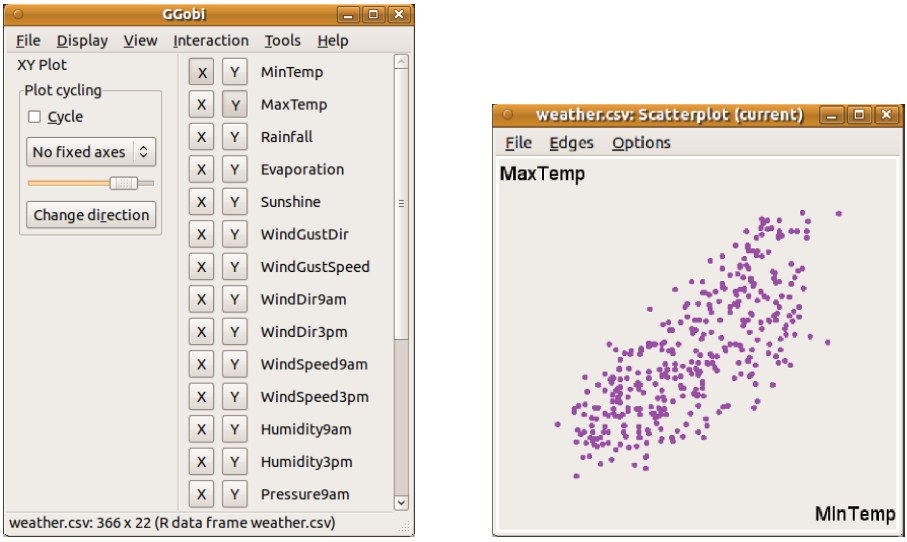

Figure 6.5: The GGobi application control and scatter plot windows.

The control window provides menus to access all of the functionality of GGobi. Below the menu bar, we can currently see the `XY Plot` (i.e., scatter plot) options. Two variables are selected from the variable list on the right side of the control window. The variables selected for display

in the scatter plot are for the x-axis (X) and the y-axis (Y). By default, the first (`MinTemp`) and second (`MaxTemp`) are the chosen variables in our dataset. We can choose any of our variables to be the X or the Y by clicking the appropriate button. This will immediately change what is displayed in the plot.

Multiple Plots

Any number of plots can be displayed simultaneously. From the Display menu, we can choose a New Scatterplot Display to have two (or more) plots displayed at one time, each in its own window. Figure 6.6 shows two scatter plots, with the new one chosen to display `Evaporation` against `Sunshine`. Changes that we make in the controlling window affect the *current*, plot which can be chosen by clicking the plot. We can also do this from the R Console using `display()`:

```
> display(gg[1], vars=list(X="Evaporation", Y="Sunshine"))
```

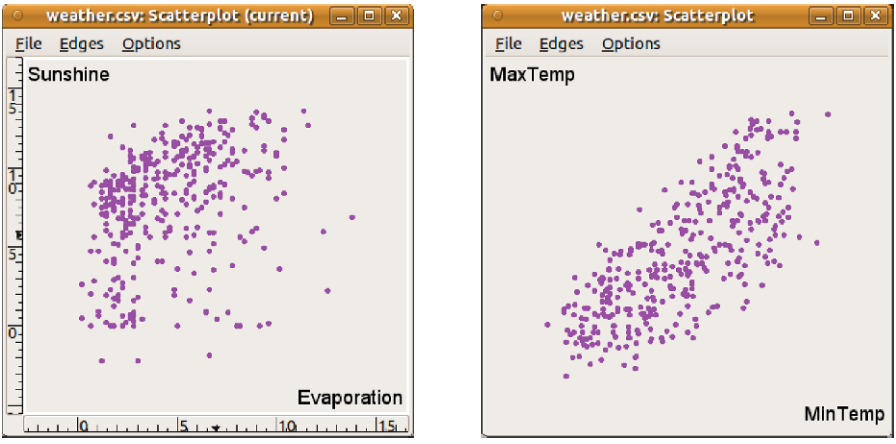

Figure 6.6: Multiple scatter plots from GGobi with and without axes.

Brushing

Brushing allows us to select observations in any plot and see them highlighted in all plots. This lets us visualise across many more dimensions than possible with a single two-dimensional plot.

From a data mining perspective, we are usually most interested in the relationship between the input variables and the target variable (using the variable `RainTomorrow` in our examples). We can highlight its two different values for the different observations using colour. From the Tools menu, choose Automatic Brushing to display the window shown in Figure 6.7.

Figure 6.7: GGobi's automatic brushing. The frequencies along the bottom will be different depending on whether or not the data is partitioned within Rattle.

From the list of variables that we see at the top of the resulting window, we can choose `RainTomorrow` (after scrolling through the list of variables to find `RainTomorrow` at the bottom of the list). Notice that the number ranges that are displayed in the lower colour map change to reflect the range of values associated with the chosen variable. For `RainTomorrow`, which has only the values 0 and 1, any observations having `RainTomorrow` values of 0 will be coloured purple, whilst those with a value of 1 will be coloured yellow.

We click on the Apply button for the automatic brushing to take effect. Any plots that GGobi is currently displaying (and any new plots we cause to be displayed from now on) will colour the observations appropriately,

as in Figure 6.8. This colouring of points across multiple plots is referred to as *brushing*.

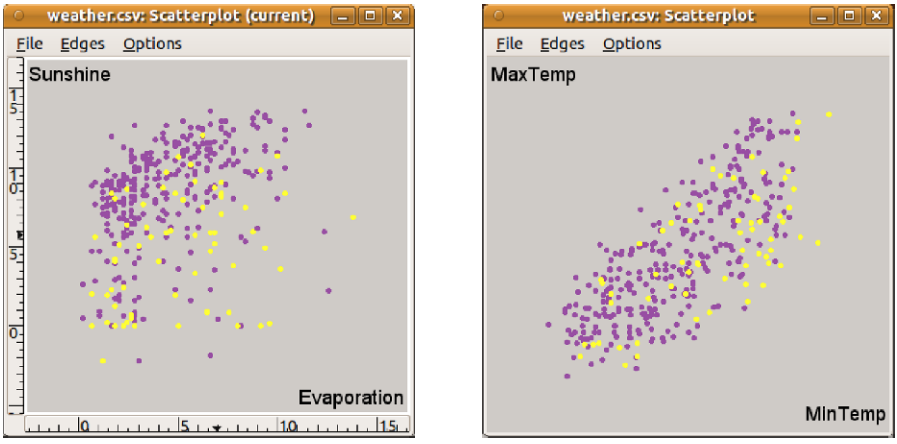

Figure 6.8: Automatic brushing of multiple scatterplots using GGobi.

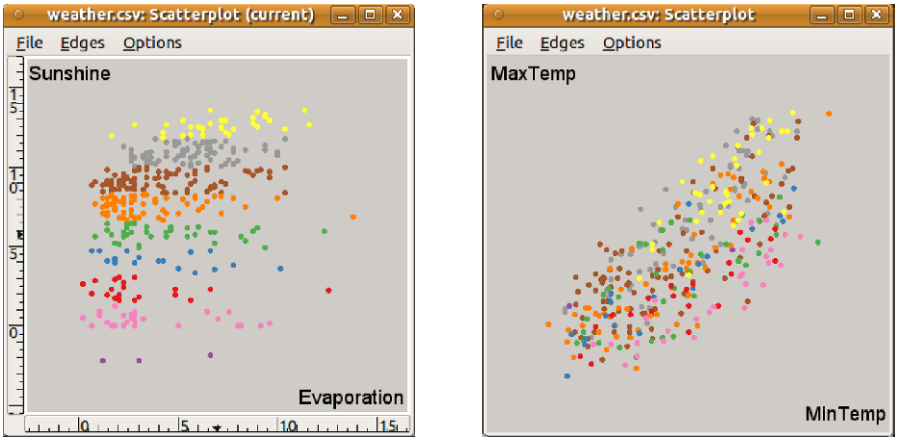

Figure 6.9: Colourful brushing of multiple scatterplots.

Our plots can be made somewhat more colourful by choosing a numeric variable, like `Sunshine`, as the choice for automatic brushing. We can see the effect in Figure 6.9. GGobi provides an extensive collection of colour schemes to choose from for these gradients, for example. Under the Tools menu, select the Color Schemes option. A nice choice could be `YlOrRd9`.

Other Plots

The Display menu provides a number of other options for plots. The Scatterplot Matrix, for example, can be used to display a matrix of scatter plots across many variables at one time. We've seen this already in both Rattle itself and **latticist**. However, GGobi offers brushing and linked views across all of the currently displayed GGobi plots.

By default, the Scatterplot Matrix will display the first four variables in our dataset, as shown in Figure 6.10. We can add and remove variables by selecting the appropriate buttons in the control window, which we notice has changed to include just the Scatterplot Matrix options rather than the previous Scatterplot options. Any manual or automatic brushing in effect will also be reflected in the scatter plots, as we can see in Figure 6.10.

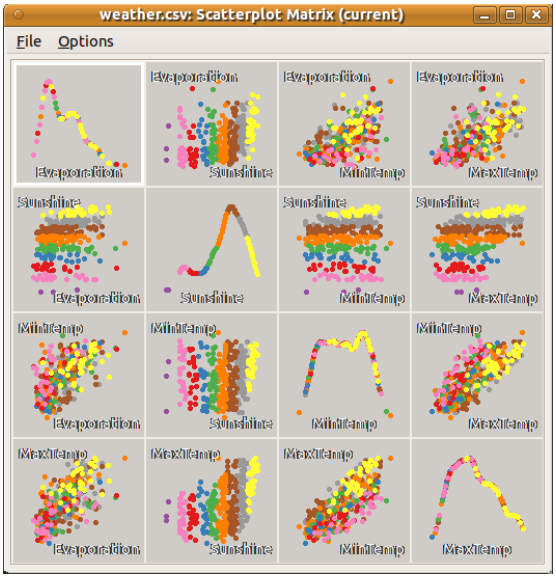

Figure 6.10: GGobi's scatter plot matrix.

A parallel coordinates plot is also easily generated from GGobi's Display menu. An example can be seen in Figure 6.11, showing five variables, beginning with `Sunshine`. The automatic brushing based on `Sunshine` is still in effect, and we can see that the coloured lines emanate from the left end of the plot within colour groups. The yellow lines represent observations with a higher value of `Sunshine`, and we can see that these generally correspond to higher values of the other variables here, except

for the final variable (`Rainfall`).

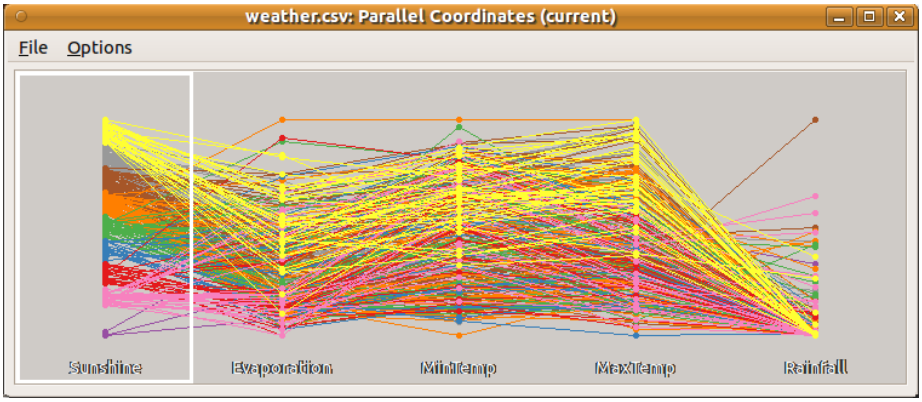

Figure 6.11: GGobi's parallel coordinates plot.

As with many of the approaches to data visualisation, when there are many observations the plots can become rather crowded and lose some of their usefulness. For example, a scatter plot over very many points will sometimes become a solid block of points showing little useful information.

Quality Plots Using R

We can save the plots generated by GGobi into an R script file and then have R generate the plots for us. This allows the plots to be regenerated as publication-quality graphics using R's capabilities. **DescribeDisplay** (Wickham et al., 2010) is required for this:

```
> install.packages("DescribeDisplay")
> library(DescribeDisplay)
```

Then, within GGobi, we choose from the Tools menu to Save Display Description. This will prompt us for a filename into which GGobi will write an R script to recreate the current graphic. We can load this script into R with `dd_load()` and then generate a plot in the usual way:

```
> pd <- dd_load("ggobi-saved-display-description.R")
> pdf("ggobi-rplot-deductions-outliers")
> plot(pd)
> dev.off()
> ggplot(pd)
```