Data Science with R

Introducing Data Mining with Rattle and R

Graham.Williams@togaware.com

Senior Director and Data Scientist, Analytics
Australian Taxation Office

Adjunct Professor, Australian National University
Fellow, Institute of Analytics Professionals of Australia

`Graham.Williams@togaware.com`
`http://datamining.togaware.com`

Overview

1. An Introduction to Data Mining

2. The Rattle Package for Data Mining

3. Moving Into R

Overview

1. An Introduction to Data Mining
2. The Rattle Package for Data Mining
3. Moving Into R

Data Mining and Big Data

- Application of
    - Machine Learning
    - Statistics
    - Software Engineering and Programming with Data
    - Intuition
- To Big Data — Volume, Velocity, Variety, Value, Veracity
- . . . to discover new knowledge
- . . . to improve business outcomes
- . . . to deliver better tailored services

The Business of Data Mining

- Australian Taxation Office
  - Lodgment ($110M)
  - Tax Havens ($150M)
  - Tax Fraud ($250M)

- Department of Immigration

- IBM Buys SPSS for $1.2B in 2009
- SAS has annual revenue approaching $3B
- Analytics is >$100B business and >$320B by 2020 (McKinsey)
- Amazon, eBay/PayPal, Google . . .

Basic Tools: Data Mining Algorithms

- Linear Discriminant Analysis (lda)
- Logistic Regression (glm)
- Decision Trees (rpart, wsrpart)
- Random Forests (randomForest, wsrf)
- Boosted Stumps (ada)
- Neural Networks (nnet)
- Support Vector Machines (kernlab)
- . . .

That's a lot of tools to learn in R!
Many with different interfaces and options.

Overview

1. An Introduction to Data Mining
2. The Rattle Package for Data Mining
3. Moving Into R

Why a GUI?

- Statistics can be complex and traps await
- **So many** tools in R to deliver insights
- Effective analyses should be scripted
- Scripting also required for repeatability
- R is a language for **programming** with data

How to remember how to do all of this in R?
How to skill up 150 data analysts with Data Mining?

Users of Rattle

Today, Rattle is used world wide in many industries

- Health analytics
- Customer segmentation and marketing
- Fraud detection
- Government

It is used by

- Consultants and Analytics Teams across business
- Universities to teach Data Mining

It is and will remain freely available.

CRAN and `http://rattle.togaware.com`

Installation

- Rattle is built using R
- Need to download and install R from cran.r-project.org
- Recommend also install RStudio from www.rstudio.org
- Then start up RStudio and install Rattle:

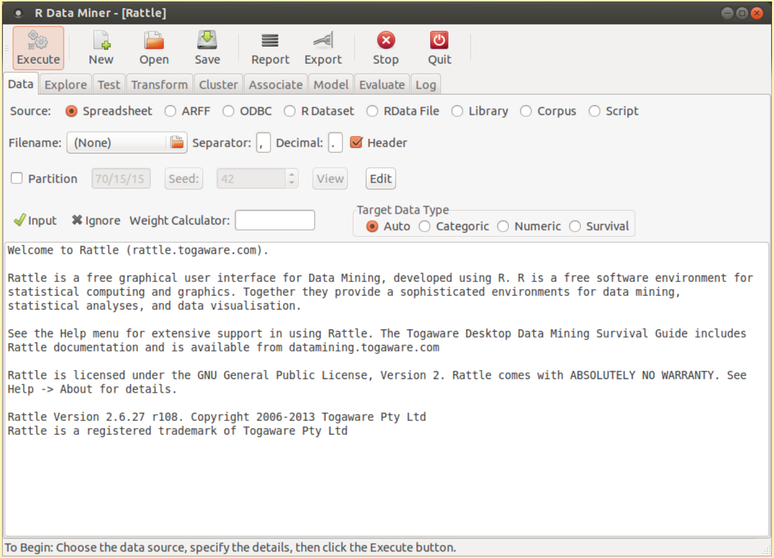
  ```
  install.packages("rattle")
  ```

- Then we can start up Rattle:

  ```
  rattle()
  ```

- Required packages are loaded as needed.

A Tour Thru Rattle: Startup



Source: ● Spreadsheet ○ ARFF ○ ODBC ○ R Dataset ○ RData File ○ Library ○ Corpus ○ Script

Filename: (None) Separator: , Decimal: . ☑ Header

□ Partition 70/15/15 Seed: 42 View Edit

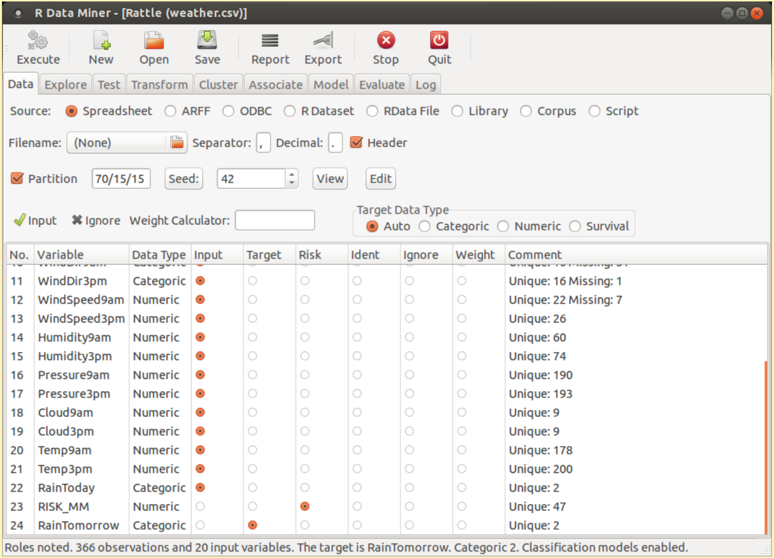
✓ Input ✖ Ignore Weight Calculator:

Target Data Type: ● Auto ○ Categoric ○ Numeric ○ Survival

```
Welcome to Rattle (rattle.togaware.com).

Rattle is a free graphical user interface for Data Mining, developed using R. R is a free software environment for
statistical computing and graphics. Together they provide a sophisticated environments for data mining,
statistical analyses, and data visualisation.

See the Help menu for extensive support in using Rattle. The Togaware Desktop Data Mining Survival Guide includes
Rattle documentation and is available from datamining.togaware.com

Rattle is licensed under the GNU General Public License, Version 2. Rattle comes with ABSOLUTELY NO WARRANTY. See
Help -> About for details.

Rattle Version 2.6.27 r108. Copyright 2006-2013 Togaware Pty Ltd
Rattle is a registered trademark of Togaware Pty Ltd
```

To Begin: Choose the data source, specify the details, then click the Execute button.

A Tour Thru Rattle: Loading Data



Source: ● Spreadsheet ○ ARFF ○ ODBC ○ R Dataset ○ RData File ○ Library ○ Corpus ○ Script

Filename: (None) Separator: , Decimal: . ☑ Header

☑ Partition 70/15/15 Seed: 42 View Edit

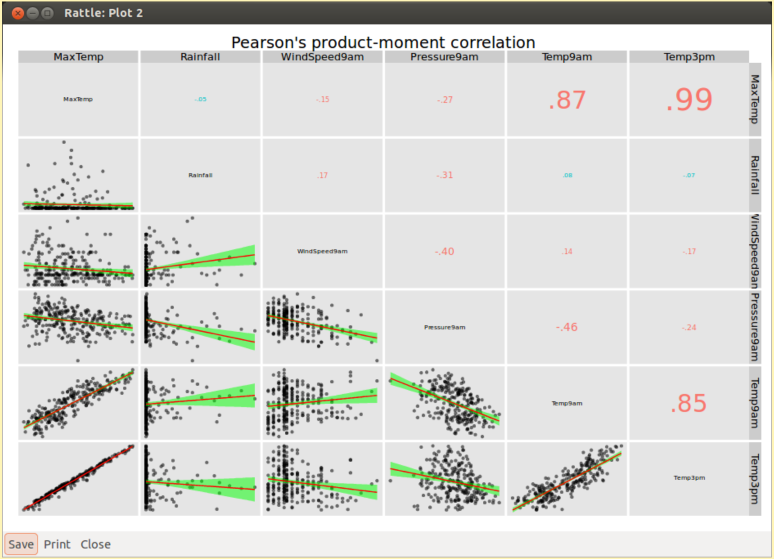
✓ Input ✖ Ignore Weight Calculator:

Target Data Type: ● Auto ○ Categoric ○ Numeric ○ Survival

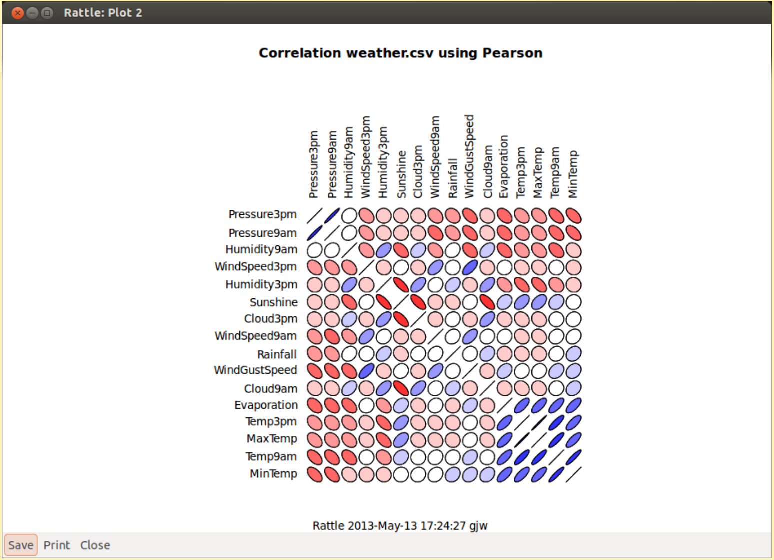
No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
11	WindDir3pm	Categoric	●	○	○	○	○	○	Unique: 16 Missing: 1
12	WindSpeed9am	Numeric	●	○	○	○	○	○	Unique: 22 Missing: 7
13	WindSpeed3pm	Numeric	●	○	○	○	○	○	Unique: 26
14	Humidity9am	Numeric	●	○	○	○	○	○	Unique: 60
15	Humidity3pm	Numeric	●	○	○	○	○	○	Unique: 74
16	Pressure9am	Numeric	●	○	○	○	○	○	Unique: 190
17	Pressure3pm	Numeric	●	○	○	○	○	○	Unique: 193
18	Cloud9am	Numeric	●	○	○	○	○	○	Unique: 9
19	Cloud3pm	Numeric	●	○	○	○	○	○	Unique: 9
20	Temp9am	Numeric	●	○	○	○	○	○	Unique: 178
21	Temp3pm	Numeric	●	○	○	○	○	○	Unique: 200
22	RainToday	Categoric	●	○	○	○	○	○	Unique: 2
23	RISK_MM	Numeric	○	○	●	○	○	○	Unique: 47
24	RainTomorrow	Categoric	○	●	○	○	○	○	Unique: 2

Roles noted. 366 observations and 20 input variables. The target is RainTomorrow. Categoric 2. Classification models enabled.

A Tour Thru Rattle: Explore Distribution



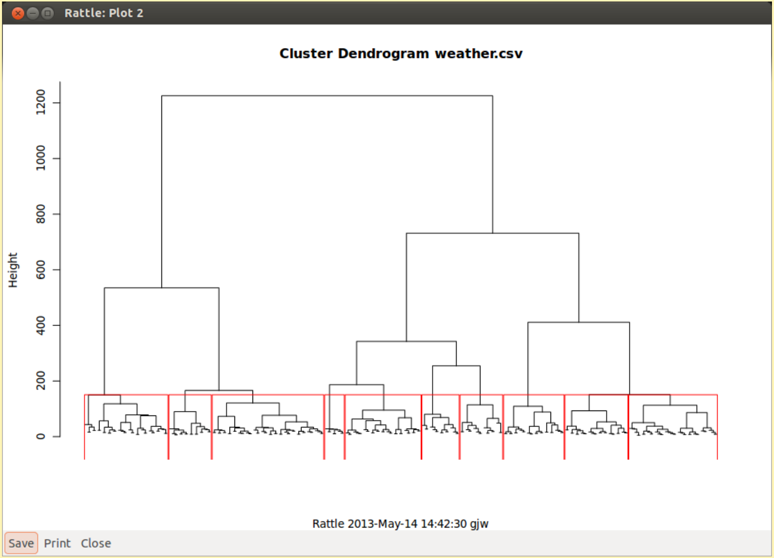
A Tour Thru Rattle: Explore Correlations


Rattle: Plot 2

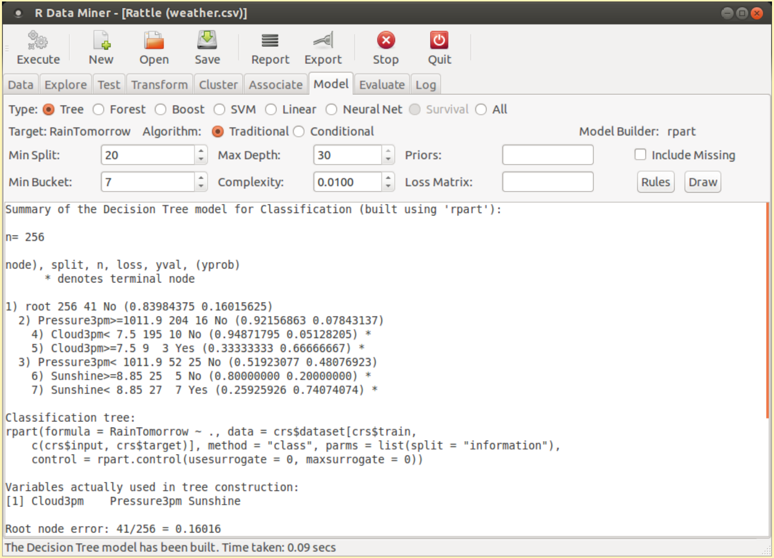
Correlation weather.csv using Pearson

Rattle 2013-May-13 17:24:27 gjw

A Tour Thru Rattle: Hierarchical Cluster


Rattle: Plot 2
Cluster Dendrogram weather.csv
Height
0
200
400
600
800
1000
1200
Rattle 2013-May-14 14:42:30 gjw
Save
Print
Close

A Tour Thru Rattle: Decision Tree



Type: ● Tree ○ Forest ○ Boost ○ SVM ○ Linear ○ Neural Net ○ Survival ○ All

Target: RainTomorrow Algorithm: ● Traditional ○ Conditional Model Builder: rpart

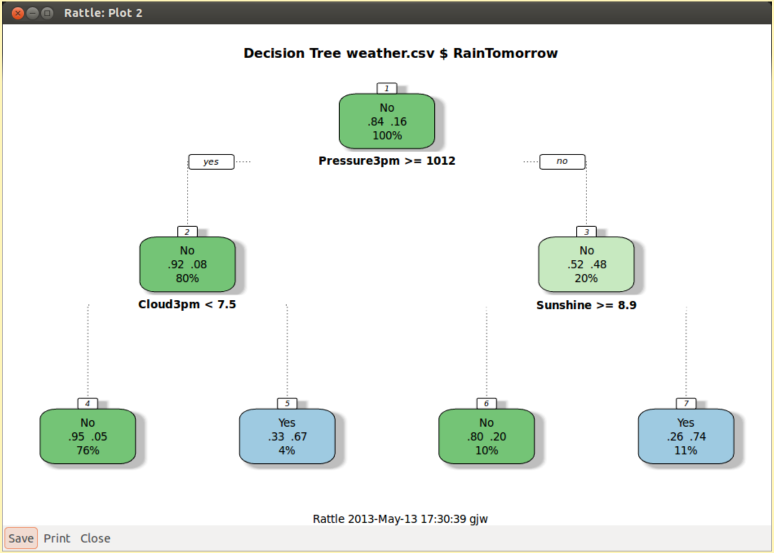
Min Split: 20 Max Depth: 30 Priors: □ Include Missing

Min Bucket: 7 Complexity: 0.0100 Loss Matrix:

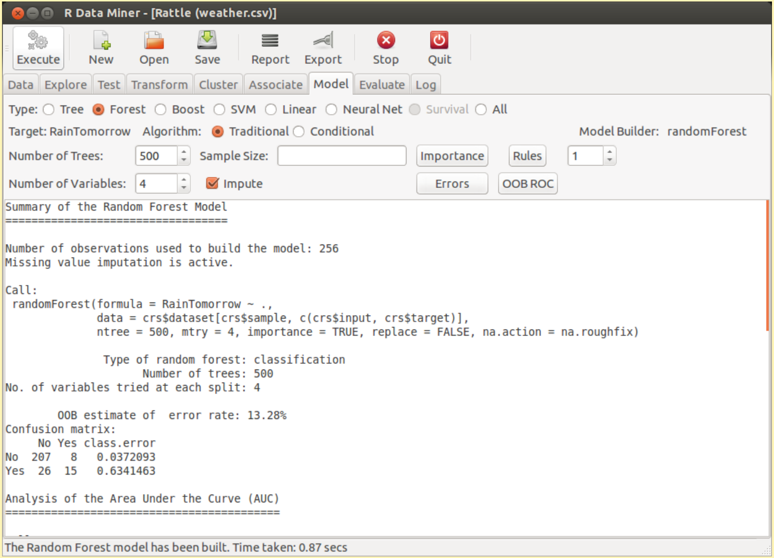
```
Summary of the Decision Tree model for Classification (built using 'rpart'):

n= 256

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 256 41 No (0.83984375 0.16015625)
  2) Pressure3pm>=1011.9 204 16 No (0.92156863 0.07843137)
    4) Cloud3pm< 7.5 195 10 No (0.94871795 0.05128205) *
    5) Cloud3pm>=7.5 9  3 Yes (0.33333333 0.66666667) *
  3) Pressure3pm< 1011.9 52 25 No (0.51923077 0.48076923)
    6) Sunshine>=8.85 25  5 No (0.80000000 0.20000000) *
    7) Sunshine< 8.85 27  7 Yes (0.25925926 0.74074074) *

Classification tree:
rpart(formula = RainTomorrow ~ ., data = crs$dataset[crs$train,
    c(crs$input, crs$target)], method = "class", parms = list(split = "information"),
    control = rpart.control(usesurrogate = 0, maxsurrogate = 0))

Variables actually used in tree construction:
[1] Cloud3pm    Pressure3pm Sunshine

Root node error: 41/256 = 0.16016
```

The Decision Tree model has been built. Time taken: 0.09 secs

A Tour Thru Rattle: Decision Tree Plot



A Tour Thru Rattle: Random Forest



Type: ○ Tree ● Forest ○ Boost ○ SVM ○ Linear ○ Neural Net ○ Survival ○ All

Target: RainTomorrow Algorithm: ● Traditional ○ Conditional Model Builder: randomForest

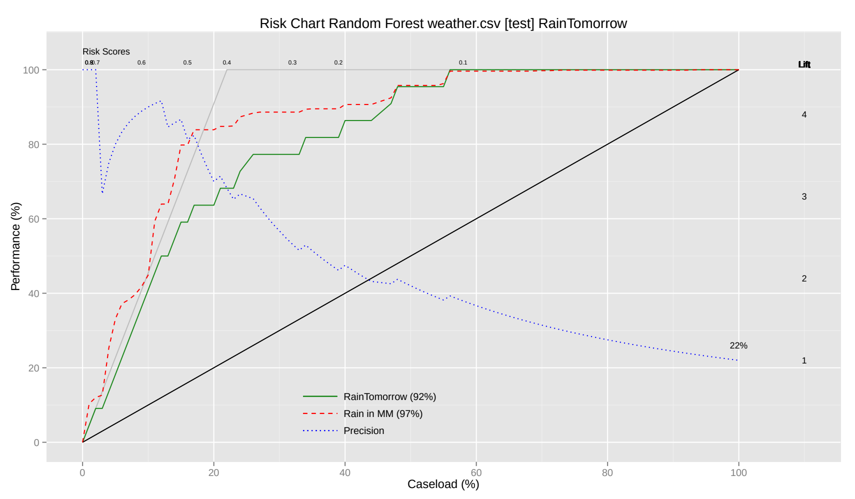
Number of Trees: 500 Sample Size: 1

Number of Variables: 4 ☑ Impute

```
Summary of the Random Forest Model
==================================

Number of observations used to build the model: 256
Missing value imputation is active.

Call:
 randomForest(formula = RainTomorrow ~ .,
              data = crs$dataset[crs$sample, c(crs$input, crs$target)],
              ntree = 500, mtry = 4, importance = TRUE, replace = FALSE, na.action = na.roughfix)

               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 13.28%
Confusion matrix:
     No Yes class.error
No  207   8   0.0372093
Yes  26  15   0.6341463

Analysis of the Area Under the Curve (AUC)
==========================================
```

The Random Forest model has been built. Time taken: 0.87 secs

A Tour Thru Rattle: Risk Chart



Overview

1 An Introduction to Data Mining

2 The Rattle Package for Data Mining

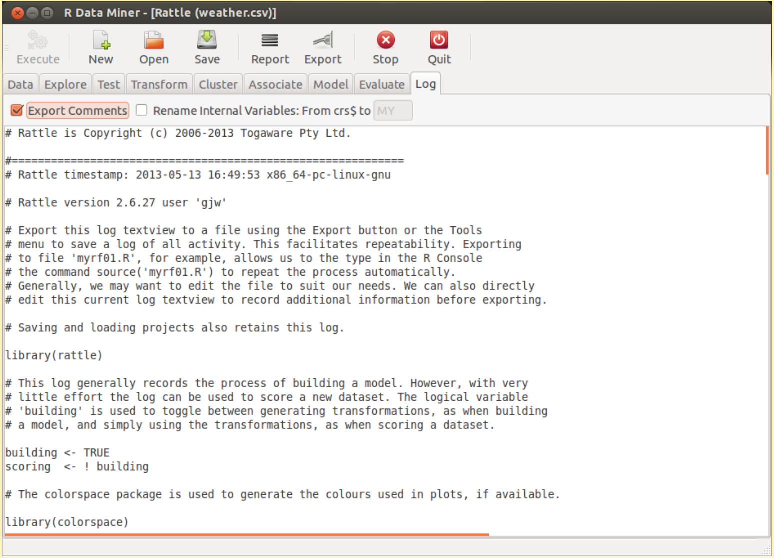
3 Moving Into R

Data Miners are Programmers of Data

- Data miners are programmers of data
- A GUI can only do so much
- R is a powerful statistical language

- Professional data mining
  - Scripting
  - Transparency
  - Repeatability

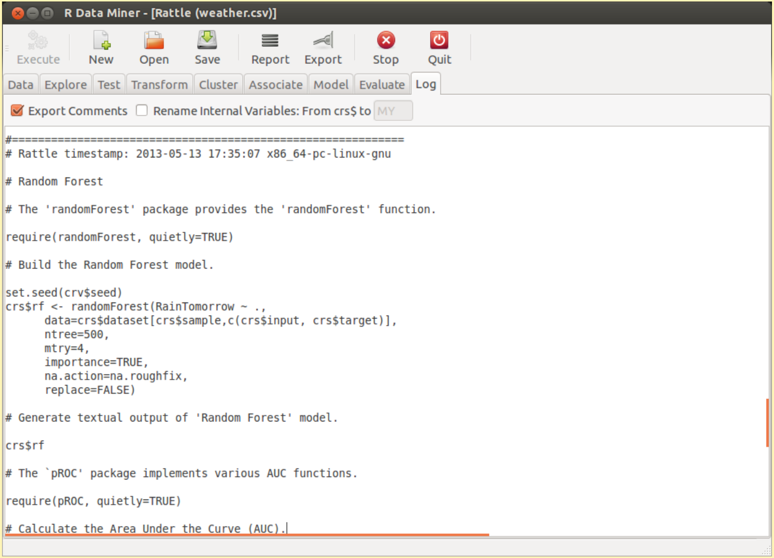
From GUI to CLI — Rattle's Log Tab



☑ Export Comments □ Rename Internal Variables: From crs$ to MY

```
# Rattle is Copyright (c) 2006-2013 Togaware Pty Ltd.

#============================================================
# Rattle timestamp: 2013-05-13 16:49:53 x86_64-pc-linux-gnu

# Rattle version 2.6.27 user 'gjw'

# Export this log textview to a file using the Export button or the Tools
# menu to save a log of all activity. This facilitates repeatability. Exporting
# to file 'myrf01.R', for example, allows us to the type in the R Console
# the command source('myrf01.R') to repeat the process automatically.
# Generally, we may want to edit the file to suit our needs. We can also directly
# edit this current log textview to record additional information before exporting.

# Saving and loading projects also retains this log.

library(rattle)

# This log generally records the process of building a model. However, with very
# little effort the log can be used to score a new dataset. The logical variable
# 'building' is used to toggle between generating transformations, as when building
# a model, and simply using the transformations, as when scoring a dataset.

building <- TRUE
scoring  <- ! building

# The colorspace package is used to generate the colours used in plots, if available.

library(colorspace)
```

From GUI to CLI — Rattle's Log Tab



☑ Export Comments □ Rename Internal Variables: From crs$ to

```
#============================================================
# Rattle timestamp: 2013-05-13 17:35:07 x86_64-pc-linux-gnu

# Random Forest

# The 'randomForest' package provides the 'randomForest' function.

require(randomForest, quietly=TRUE)

# Build the Random Forest model.

set.seed(crv$seed)
crs$rf <- randomForest(RainTomorrow ~ .,
      data=crs$dataset[crs$sample,c(crs$input, crs$target)],
      ntree=500,
      mtry=4,
      importance=TRUE,
      na.action=na.roughfix,
      replace=FALSE)

# Generate textual output of 'Random Forest' model.

crs$rf

# The `pROC' package implements various AUC functions.

require(pROC, quietly=TRUE)

# Calculate the Area Under the Curve (AUC).
```

Step 1: Load the Dataset

```
dsname <- "weather"
ds     <- get(dsname)
dim(ds)

## [1] 366  24

names(ds)

##  [1] "Date"          "Location"      "MinTemp"       "...
##  [5] "Rainfall"      "Evaporation"   "Sunshine"      "...
##  [9] "WindGustSpeed" "WindDir9am"    "WindDir3pm"    "...
## [13] "WindSpeed3pm"  "Humidity9am"   "Humidity3pm"   "...
....
```

Step 2: Observe the Data — Observations

```
head(ds)

##          Date Location MinTemp MaxTemp Rainfall Evapora...
## 1 2007-11-01 Canberra     8.0    24.3      0.0        ...
## 2 2007-11-02 Canberra    14.0    26.9      3.6        ...
## 3 2007-11-03 Canberra    13.7    23.4      3.6        ...
....
```

```
tail(ds)

##            Date Location MinTemp MaxTemp Rainfall Evapo...
## 361 2008-10-26 Canberra     7.9    26.1        0      ...
## 362 2008-10-27 Canberra     9.0    30.7        0      ...
## 363 2008-10-28 Canberra     7.1    28.4        0      ...
....
```

Step 2: Observe the Data — Structure

```
str(ds)

## 'data.frame': 366 obs. of  24 variables:
##  $ Date         : Date, format: "2007-11-01" "2007-11-...
##  $ Location     : Factor w/ 46 levels "Adelaide","Alba...
##  $ MinTemp      : num  8 14 13.7 13.3 7.6 6.2 6.1 8.3 ...
##  $ MaxTemp      : num  24.3 26.9 23.4 15.5 16.1 16.9 1...
##  $ Rainfall     : num  0 3.6 3.6 39.8 2.8 0 0.2 0 0 16...
##  $ Evaporation  : num  3.4 4.4 5.8 7.2 5.6 5.8 4.2 5.6...
##  $ Sunshine     : num  6.3 9.7 3.3 9.1 10.6 8.2 8.4 4....
##  $ WindGustDir  : Ord.factor w/ 16 levels "N"<"NNE"<"N...
##  $ WindGustSpeed: num  30 39 85 54 50 44 43 41 48 31 ...
##  $ WindDir9am   : Ord.factor w/ 16 levels "N"<"NNE"<"N...
##  $ WindDir3pm   : Ord.factor w/ 16 levels "N"<"NNE"<"N...
....
```

Step 2: Observe the Data — Summary

```
summary(ds)

##       Date                     Location      MinTemp  ...
##  Min.   :2007-11-01   Canberra     :366   Min.   :-5.3...
##  1st Qu.:2008-01-31   Adelaide     :  0   1st Qu.: 2.3...
##  Median :2008-05-01   Albany       :  0   Median : 7.4...
##  Mean   :2008-05-01   Albury       :  0   Mean   : 7.2...
##  3rd Qu.:2008-07-31   AliceSprings :  0   3rd Qu.:12.5...
##  Max.   :2008-10-31   BadgerysCreek:  0   Max.   :20.9...
##                       (Other)      :  0              ...
##     Rainfall      Evaporation       Sunshine      Wind...
##  Min.   : 0.00   Min.   : 0.20   Min.   : 0.00   NW   ...
##  1st Qu.: 0.00   1st Qu.: 2.20   1st Qu.: 5.95   NNW  ...
##  Median : 0.00   Median : 4.20   Median : 8.60   E    ...
....
```

Step 2: Observe the Data — Variables

```
id     <- c("Date", "Location")
target <- "RainTomorrow"
risk   <- "RISK_MM"
(ignore <- union(id, risk))

## [1] "Date"     "Location" "RISK_MM"

(vars   <- setdiff(names(ds), ignore))

##  [1] "MinTemp"       "MaxTemp"       "Rainfall"      "...
##  [5] "Sunshine"      "WindGustDir"   "WindGustSpeed" "...
##  [9] "WindDir3pm"    "WindSpeed9am"  "WindSpeed3pm"  "...
## [13] "Humidity3pm"   "Pressure9am"   "Pressure3pm"   "...
....
```

Step 3: Clean the Data — Remove Missing

```
dim(ds)

## [1] 366  24

sum(is.na(ds[vars]))

## [1] 47

ds <- ds[-attr(na.omit(ds[vars]), "na.action"),]
```

Step 3: Clean the Data — Remove Missing

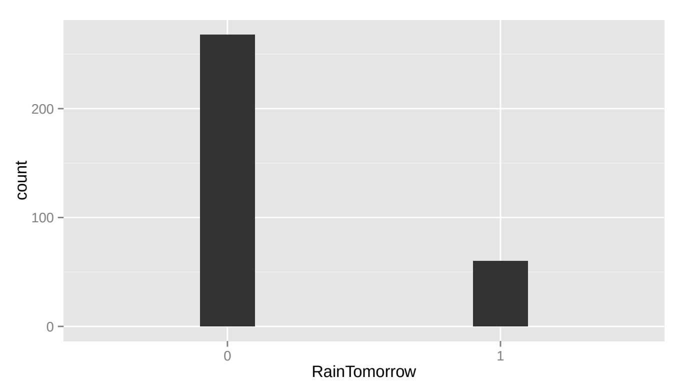
```
dim(ds)

## [1] 328  24

sum(is.na(ds[vars]))

## [1] 0
```

Step 3: Clean the Data—Target as Categoric

```
summary(ds[target])

##   RainTomorrow
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.183
##  3rd Qu.:0.000
##  Max.   :1.000
....

ds[target] <- as.factor(ds[[target]])
levels(ds[target]) <- c("No", "Yes")
```

Step 3: Clean the Data—Target as Categoric

```
summary(ds[target])

##  RainTomorrow
##  0:268
##  1: 60
```



Step 4: Prepare for Modelling

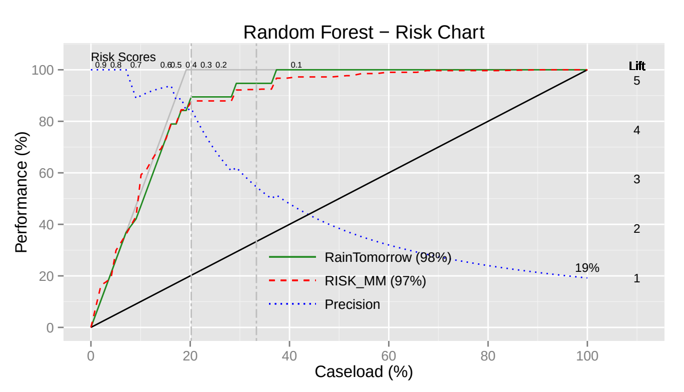
```
(form <- formula(paste(target, "~ .")))

## RainTomorrow ~ .

(nobs <- nrow(ds))

## [1] 328

train <- sample(nobs, 0.70*nobs)
length(train)

## [1] 229

test  <- setdiff(1:nobs, train)
length(test)

## [1] 99
```

Step 5: Build the Model—Random Forest

```
library(randomForest)
model <- randomForest(form, ds[train, vars], na.action=na.omit)
model

##
## Call:
##  randomForest(formula=form, data=ds[train, vars], ...
##               Type of random forest: classification
##                     Number of trees: 500
## No. of variables tried at each split: 4
....
```

Step 6: Evaluate the Model—Risk Chart

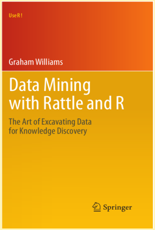
```
pr <- predict(model, ds[test,], type="prob")[,2]
riskchart(pr, ds[test, target], ds[test, risk],
          title="Random Forest - Risk Chart",
          risk=risk, recall=target, thresholds=c(0.35, 0.15))
```

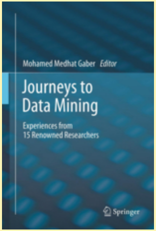


Resources and References

- **OnePageR**: `http://onepager.togaware.com` – Tutorial Notes
- Rattle: `http://rattle.togaware.com`
- Guides: `http://datamining.togaware.com`
- Practise: `http://analystfirst.com`

- Book: Data Mining using Rattle/R
- Chapter: Rattle and Other Tales
- Paper: A Data Mining GUI for R — R Journal, Volume 1(2)


Use R!
Graham Williams
Data Mining with Rattle and R
The Art of Excavating Data for Knowledge Discovery
Springer


Mohamed Medhat Gaber Editor
Journeys to Data Mining
Experiences from 15 Renowned Researchers
Springer

Contributed Research Articles 45

Rattle: A Data Mining GUI for R

by Graham J Williams

Abstract Data mining delivers insights, patterns, and descriptive and predictive models from the large amounts of data available today in many organisations. The data miner draws heavily on methodologies, techniques and algorithms from statistics, machine learning, and computer science. R increasingly provides a powerful platform for data mining. However, scripting and programming is sometimes a challenge for data analysts moving into data mining. The Rattle package provides a graphical user interface specifically for data mining using R. It also provides a stepping stone toward using R as a programming language for data analysis.

Introduction

Data mining combines concepts, tools, and algorithms from machine learning and statistics for the analysis of very large datasets, so as to gain insights, understanding, and actionable knowledge.

Closed source data mining products have facilitated the uptake of data mining in many organisations. These products offer off-the-shelf ease-of-use that makes them attractive to the many new data miners in a market place desperately seeking high levels of analytical skills.

R is ideally suited to the many challenging tasks associated with data mining. R offers a breadth and depth in statistical computing beyond what is available in commercial closed source products. Yet R remains, primarily, a programming language for the highly skilled statistician, and out of the reach of many.

Rattle (the R Analytical Tool To Learn Easily) is a graphical data mining application written in and providing a pathway into R (Williams, 2009b). It has been developed specifically to ease the transition from basic data mining, as necessarily offered by GUIs, to sophisticated data analyses using a powerful statistical language.

Rattle brings together a multitude of R packages that are essential for the data miner but often not easy for the novice to use. An understanding of R is not required in order to get started with Rattle—this will gradually grow as we add sophistication to our data mining. Rattle's user interface provides an entree into the power of R as a data mining tool.

Rattle is used for teaching data mining at numerous universities and is in daily use by consultants and data mining teams world wide. It is also available as a product within Information Builders' WebFocus business intelligence suite as RStat.

Rattle is one of several open source data mining tools (Chen et al., 2007). Many of these tools are also directly available within R (and hence Rattle) through packages like **RWeka** (Hornik et al., 2009) and **arules** (Hahsler et al., 2005).

Implementation

Rattle uses the Gnome graphical user interface as provided through the **RGtk2** package (Lawrence and Lang, 2006). It runs under various operating systems, including GNU/Linux, Macintosh OS/X, and MS/Windows.

The GUI itself has been developed using the Glade interactive interface builder. This produces a programming-language-independent XML description of the layout of the widgets that make up the user interface. The XML file is then simply loaded by an application and the GUI is rendered!

Glade allows the developer to freely choose to implement the functionality (i.e., the widget callbacks) in a programming language of choice, and for Rattle that is R. It is interesting to note that the first implementation of Rattle actually used Python for implementing the callbacks and R for the statistics, using **rpy**. The release of **RGtk2** allowed the interface elements of Rattle to be written directly in R so that Rattle is a fully R-based application.

Underneath, Rattle relies upon an extensive collection of R packages. This is a testament to the power of R—it delivers a breadth and depth of statistical analysis that is hard to find anywhere else. Some of the packages underlying Rattle include **ada**, **arules**, **doBy**, **ellipse**, **fBasics**, **fpc**, **gplots**, **Hmisc**, **kernlab**, **mice**, **network**, **party**, **playwith**, **pmml**, **randomForest**, **reshape**, **rggobi**, **RGtk2**, **ROCR**, **RODBC**, and **rpart**. These packages are all available from the Comprehensive R Archive Network (CRAN). If a package is not installed but we ask through Rattle for some functionality provided by that package, Rattle will popup a message indicating that the package needs to be installed.

Rattle is not only an interface though. Additional functionality that is desired by a data miner has been written for use in Rattle, and is available from the **rattle** package without using the Rattle GUI. The **pmml** package (Guazzelli et al., 2009) is an offshoot of the development of Rattle and supports the export of models from Rattle using the open standard XML based PMML, or Predictive Model Markup Language (Data Mining Group, 2008). Models exported from R in this way can be imported into tools like the ADAPA decision engine running on cloud computers, Teradata's Warehouse Miner for deployment as SQL over a very large database, and Information Builder's WebFocus which handles

The R Journal Vol. 1/2, December 2009 ISSN 2073-4859

Thank You

Question Time

This document, sourced from StartL.Rnw revision 282, was processed by KnitR version 1.5 of 2013-09-28 and took 4 seconds to process. It was generated by gjw on nyx running Ubuntu 13.10 with Intel(R) Xeon(R) CPU W3520 @ 2.67GHz having 4 cores and 12.3GB of RAM. It completed the processing 2014-02-14 06:20:04.