The required packages for this module include:

```r
library(rattle)
```

As we work through this chapter, new R commands will be introduced. Be sure to review the command’s documentation and understand what the command does. You can ask for help using the `?` command as in:

```r
?read.csv
```

We can obtain documentation on a particular package using the `help=` option of `library()`:

```r
library(help=rattle)
```

This chapter is intended to be hands on. To learn effectively, you are encouraged to have R running (e.g., RStudio) and to run all the commands as they appear here. Check that you get the same output, and you understand the output. Try some variations. Explore.
1 Functions

```r
mult10 <- function(x)
{
  if (is.character(x))
  {
    result <- apply(sapply(x, rep, 10), 2, paste, collapse="")
    names(result) <- NULL
  }
  else
  {
    result <- x * 10
  }

  return(result)
}
```
## 2 Function Calls

\[
\begin{align*}
4 + 5 \\
## [1] 9 \\
"+"(4, 5) \\
## [1] 9 \\
1 + 2 + 3 + 4 + 5 \\
## [1] 15 \\
Reduce("+", 1:5) \\
## [1] 15 \\
\text{cmd} &\leftarrow "1 + 2 + 3 + 4 + 5" \\
\text{eval(parse(text=cmd))} \\
## [1] 15
\end{align*}
\]
3 Flow Control

```r
for (i in 0:4)
  for (j in 5:9)
    print(paste0(i, j))
```

## [1] "05"
## [1] "06"
## [1] "07"
## [1] "08"
....
4 Exercise Dataset: WeatherAUS

For our exercises we will use the `weatherAUS` dataset from `rattle` *(Williams, 2014)*. We will essentially follow the template presented in the Models module.

```r
ds <- read.csv(file="data/weatherAUS.csv")

head(ds)
## Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine
## 1 2008-12-01 Albury 13.4 22.9 0.6 NA NA
## 2 2008-12-02 Albury 7.4 25.1 0.0 NA NA
## 3 2008-12-03 Albury 12.9 25.7 0.0 NA NA
## 4 2008-12-04 Albury 9.2 28.0 0.0 NA NA
## 5 2008-12-05 Albury 17.5 32.3 1.0 NA NA
## 6 2008-12-06 Albury 14.6 29.7 0.2 NA NA

tail(ds)
## Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine
## 66667 2012-11-24 Darwin 25.5 34.1 0.0 5.0 6.2
## 66668 2012-11-25 Darwin 24.4 35.7 0.2 4.8 11.7
## 66669 2012-11-26 Darwin 25.0 35.4 0.0 7.4 11.7
## 66670 2012-11-27 Darwin 26.5 35.9 0.0 8.0 10.3
## 66671 2012-11-28 Darwin 27.4 35.0 0.0 7.8 6.5
## 66672 2012-11-29 Darwin 24.8 33.5 3.4 7.4 4.7

str(ds)
# 'data.frame': 66672 obs. of 24 variables:
# $ Date  : Factor w/ 1826 levels "2007-11-01","2007-11-02",...: 397 ...
# $ Location: Factor w/ 46 levels "Adelaide","Albany",...: 3 3 3 3 3 3 ...
# $ MinTemp  : num 13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
# $ MaxTemp  : num 22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
# $ Rainfall : num NA NA NA NA NA NA NA NA NA NA ...
# $ Evaporation: num NA NA NA NA NA NA NA NA NA NA ...

summary(ds)
# Date Location MinTemp MaxTemp
# 2009-01-01: 46 Canberra: 1826 Min. :-8.5 Min. :-3.1
# 2009-01-02: 46 Sydney: 1734 1st Qu.: 7.3 1st Qu.:17.6
# 2009-01-03: 46 Adelaide: 1583 Median: 11.7 Median :22.1
# 2009-01-04: 46 Brisbane: 1583 Mean: 11.9 Mean :22.7
# 2009-01-05: 46 Darwin: 1583 3rd Qu.:16.6 3rd Qu.:27.5
# 2009-01-06: 46 Hobart: 1583 Max. :33.9 Max. :48.1
```
5 Exercises: Prepare for Modelling

Following the template presented in the Models module, we continue with setting up some of the modelling parameters.

```r
target <- "RainTomorrow"
risk <- "RISK_MM"
dsname <- "weather"

d$s["target"] <- as.factor(ds[[target]])
summary(ds$s["target"])
## RainTomorrow
## No :50187
## Yes :15259
## NA's: 1226
....

vars <- colnames(ds)
ignore <- vars[c(1, 2, if (exists("risk")) which(risk==vars))] 
vars <- setdiff(vars, ignore)
(inputs <- setdiff(vars, target))
## [1] "MinTemp" "MaxTemp" "Rainfall" "Evaporation"
## [5] "Sunshine" "WindGustDir" "WindGustSpeed" "WindDir9am"
## [9] "WindDir3pm" "WindSpeed9am" "WindSpeed3pm" "Humidity9am"
## [13] "Humidity3pm" "Pressure9am" "Pressure3pm" "Cloud9am"
....
nobs <- nrow(ds)
dim(ds$vars))
## [1] 66672 21

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

set.seed(142)

length(train <- sample(nobs, 0.7*nobs))
## [1] 46670

length(test <- setdiff(seq_len(nobs), train))
## [1] 20002
```
6 Exercise: varWeights()

The first exercise is to write a function to take a dataset and return probabilities associated with each input variable in the dataset, that relate to the correlation between the input variable and the target variable.

\[
\text{varw} \leftarrow \text{varWeights}(\text{form}, \text{ds})
\]

We will use the R correlation functions to calculate the correlation between each column (variable) of the data frame and the values of the target vector. Below are some hints.

\[
\begin{align*}
\text{n1} & \leftarrow \text{ds["Temp3pm"]} \\
\text{c1} & \leftarrow \text{ds["WindGustDir"]} \\
\text{t1} & \leftarrow \text{ds[[target]]}
\end{align*}
\]

\[
\begin{align*}
\text{cor(as.numeric(n1), as.numeric(t1), use="pairwise.complete.obs")} & \quad \# [1] -0.1857 \\
\text{cor(as.numeric(c1), as.numeric(t1), use="pairwise.complete.obs")} & \quad \# [1] 0.04414
\end{align*}
\]

The template for the function is:

\[
\text{varWeights} \leftarrow \text{function(formula, data)} \\
\quad \{ \\
\quad \quad \text{...} \\
\quad \}
\]

The actual solution will produce the following output:

\[
\begin{align*}
\text{varWeights(form, ds)} \\
\text{##} & \quad \text{Date} \quad \text{Location} \quad \text{MinTemp} \quad \text{MaxTemp} \quad \text{Rainfall} \\
\text{##} & \quad 0.0028545 \quad 0.0004961 \quad 0.0210742 \quad 0.0336550 \quad 0.0544825 \\
\text{##} & \quad \text{Evaporation} \quad \text{Sunshine} \quad \text{WindGustDir} \quad \text{WindGustSpeed} \quad \text{WindDir9am} \\
\text{##} & \quad 0.0249688 \quad 0.1006235 \quad 0.0096812 \quad 0.0518932 \quad 0.0067119 \\
\text{....}
\end{align*}
\]

It is time now to write that function.
7 Exercise: selectVars()

The next exercise is to write a function that will return \( n \) variables chosen at random from all of the variables in a dataset, but chosen with a probability proportional to the correlation of the target variable.

```r
vars <- selectVars(form, ds, 3)
```

The `sample()` function might come in use for this function. Note the `prob=` argument of `sample`. We will, of course, also make use of the `varWeights()` function we defined previously.

The template for the function is:

```r
selectVars <- function(formula, data, n)
{
  ...
}
```

The actual solution will produce the following output:

```r
selectVars(form, ds, 3)
## [1] "Pressure3pm" "MaxTemp" "RISK_MM"
selectVars(form, ds, 3)
## [1] "Cloud9am" "RISK_MM" "Sunshine"
selectVars(form, ds, 3)
## [1] "WindDir9am" "Humidity3pm" "Cloud9am"
selectVars(form, ds, 3)
## [1] "Cloud3pm" "Evaporation" "Humidity3pm"
```
8 Exercise: wsrpart()

This exercise is to write a function to build a subspace decision tree. The function wsrpart() (for weighted subspace rpart) will take a dataset (data) and return a decision tree (built using rpart()) that uses only a subspace of the variables available. The number of variables to use is, by default, $\text{trunc}(\log_2(n + 1))$ (overridden by nvars) and the variables are chosen according to the weighted selection implemented through selectVars().

The idea is similar, but not identical to, the concept of random forests developed by Leo Breiman. Note that Breiman’s original random forest paper (on which this idea is based) specifies $\text{trunc}(\log_2 n + 1)$, which is ambiguous in terms of being either $\text{trunc}(\log_2(n + 1))$ or $\text{trunc}(\log_2(n) + 1)$, although he probably meant the latter.

```
dt <- wsrpart(form, data)
```

The template for the function is:

```r
wsrpart <- function(formula, data, nvars, ...)
{
...
}
```

Notice the use of “...” in the argument list. This allows us to pass other arguments on through to rpart().

The actual solution will produce the following output:

```
## system.time(model <- wsrpart(form, ds[train, vars]))
## user system elapsed
## 0.159  0.011  2.497

model
## A multiple rpart model with 1 tree.
##
## Variables used (11): Pressure9am, Cloud3pm, Sunshine, Rainfall, WindGustDir,
##                      Humidity3pm, WindDir9am, RainToday,
##                      ....
```
9 Exercise: Multiple Decision Trees

Now extend the function `wsrpart()` to build multiple decision trees. The function will take a formula, a dataset (data) and a number of trees to build (ntrees), and returns a list of ntrees decision trees. Each element of the list will itself be a list, with at least one element named `model`. This is the actual rpart model. The result should be of class `mrpart` for “multiple rpart.”

The actual solution will produce the following output:

```r
system.time(model <- wsrpart(form, ds[train, vars], 4))
## user    system   elapsed
## 94.024   2.833   34.958
class(model)
## [1] "mrpart"
length(model)
## [1] 50
class(model[[1]]$model)
## [1] "rpart"
model[[1]]$model
## n=45731 (939 observations deleted due to missingness)
## node), split, n, loss, yval, (yprob)
##   * denotes terminal node
## 1) root 45731 10680 No (0.7665 0.2335)
## 2) RainToday=No 35102 5572 No (0.8413 0.1587) *
## 3) RainToday=Yes 10629 5106 No (0.5196 0.4804)
## 6) Cloud3pm< 6.5 4728 1531 No (0.6762 0.3238) *
## 7) Cloud3pm>6.5 5901 2326 Yes (0.3942 0.6058) *
```
10 Exercise: `predict.mrpart()`

Score a Dataset Using the Forest

Define a function `predict.mrpart()`. Returns the proportion of trees voting for the positive case, assuming binary classification models.

The actual solution will produce the following output:

```r
predict(model, ds[test,vars])
## 8 9 12 16 19 27 36 45 46 51 55 59
## No No Yes No No No No No No No No
## 61 62 66 68 78 83 86 87 88 89 93 95
## No No No No No No No No No No No
....
```
11 Further Reading

The Rattle Book, published by Springer, provides a comprehensive introduction to data mining and analytics using Rattle and R. It is available from Amazon. Other documentation on a broader selection of R topics of relevance to the data scientist is freely available from http://datamining.togaware.com, including the Datamining Desktop Survival Guide.

This module is one of many OnePageR modules available from http://onepager.togaware.com. In particular follow the links on the website with a * which indicates the generally more developed OnePageR modules.

Other resources include:


12 References


