Data Science with R
Transform and Manipulate Data

Graham.Williams@togaware.com

14th February 2014

Visit http://onepager.togaware.com/ for more OnePageR's.

In this module we introduce approaches to manipulate and transform our data.

The required packages for this module include:

```
library(rattle)          # The weatherAUS datasets and normVarNames()
library(ggplot2)         # Visualise the transforms.
library(plyr)            # Transform using ddplyr()
library(dplyr)           # Transform using ddplyr()
library(reshape2)        # melt() and dcast()
```

As we work through this module, 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:

```
?read.csv
```

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

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

This present module 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.

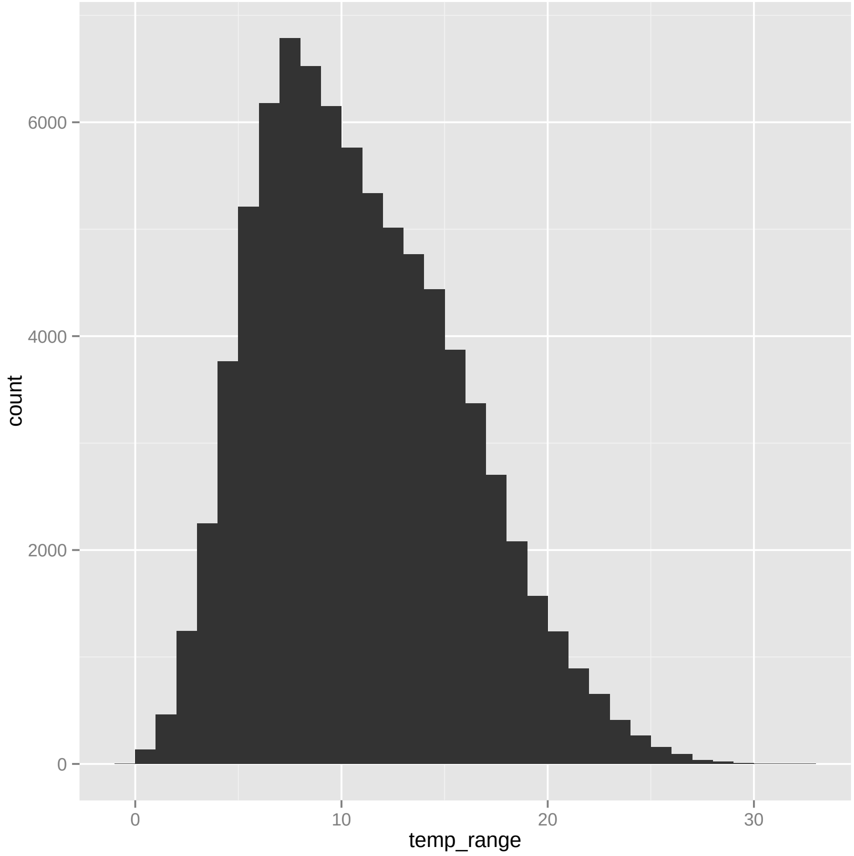
1 Data

```
library(rattle)
ds         <- weatherAUS
names(ds)  <- normVarNames(names(ds))   # Lower case variable names.
str(ds)

## 'data.frame': 82169 obs. of  24 variables:
##  $ date           : Date, format: "2008-12-01" "2008-12-02" ...
##  $ location       : Factor w/ 46 levels "Adelaide","Albany",..: 3 3 3 3 3 ...
##  $ min_temp       : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
....
```

2 Data Frame: Add a Column

Here we simply name the column as part of the data frame and it gets added to it.

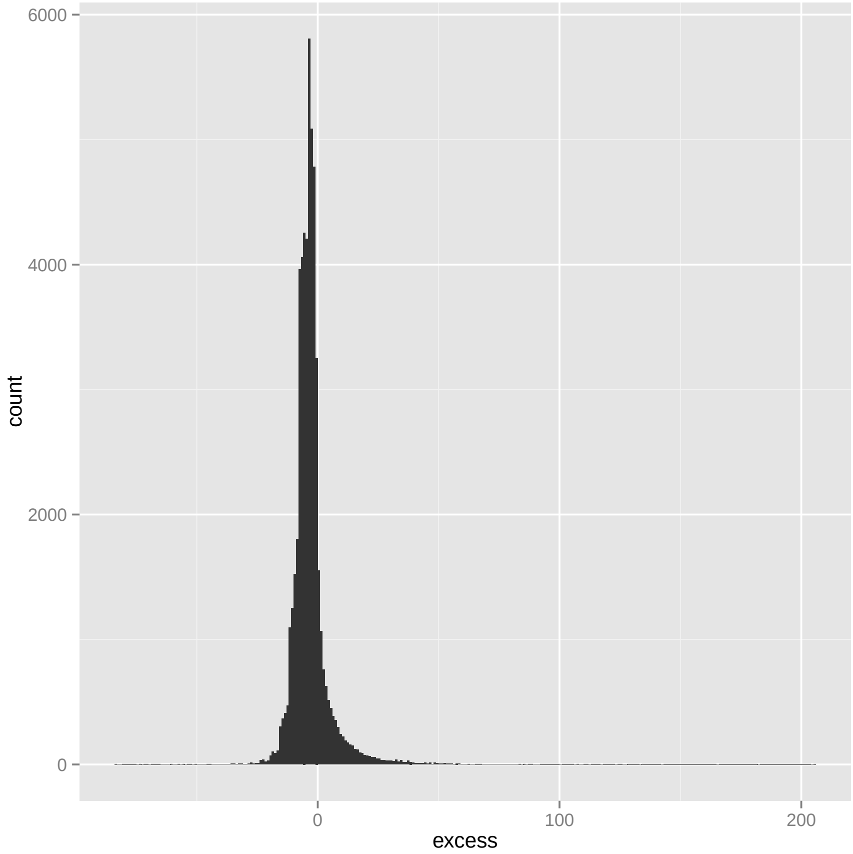
```
ds$temp_range <- ds$max_temp - ds$min_temp
str(ds)

## 'data.frame': 82169 obs. of  25 variables:
##  $ date          : Date, format: "2008-12-01" "2008-12-02" ...
##  $ location      : Factor w/ 46 levels "Adelaide","Albany",..: 3 3 3 3 3 ...
##  $ min_temp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
....

p <- ggplot(ds, aes(x=temp_range))
p <- p + geom_bar(binwidth=1)
p
```



3 Transform: Add a Column

An alternative is to use `transform()` which can be neater when adding several columns, avoiding the use of the `$` nomenclature.

```
ds <- transform(ds,
                temp_range=max_temp-min_temp,
                excess=rainfall-evaporation)
sum(ds$excess, na.rm=TRUE)

## [1] -161947

str(ds)

## 'data.frame': 82169 obs. of  26 variables:
##  $ date          : Date, format: "2008-12-01" "2008-12-02" ...
##  $ location      : Factor w/ 46 levels "Adelaide","Albany",..: 3 3 3 3 3 ...
##  $ min_temp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
....

ggplot(ds, aes(x=excess)) + geom_bar(binwidth=1)
```



4 Subset Data

Exercise: Research the subset() function and illustrate its usage.

5 Transform Using DPlyR

The plyr (Wickham, 2012a) package provides a collection of the most useful functions for manipulating data. It's concepts, once understood, are very powerful and allow us to express numerous tasks simply and efficiently.

Like `apply()`, the `plyr` functions operate on data frames, matrices, lists, vectors or arrays. An operation is applied to some collection of items (e.g., each group of observations or group of list elements) in the input data structure, and the results are packaged into a new data structure.

Generally, the pattern is like `ddply(data, variables, function, ...)` where in this case (as indicated by the first d) the input data is a data frame and the result (the second d) is also a data frame. The rows of the data frame will be grouped by the variables identified, and for each group the function is applied to obtain the resulting data. The remaining arguments are treated as arguments to the function.

Exercise: Explore and provide examples.

6 Summarise Data Using dplyr()

dplyr (Wickham and Francois, 2014) introduces a grammar of data manipulation and processes data much more efficiently than plyr (Wickham, 2012a) (anywhere from 20 times to 1000 times faster) and other R packages through parallel processing using Rcpp (Eddelbuettel and Francois, 2013).

```
weatherAUS %.%
  group_by(Location) %.%
  summarise(total = sum(Rainfall)) %.%
  arrange(desc(total)) %.%
  head(5)
```

```
## Source: local data frame [5 x 2]
##
##         Location total
## 1         Darwin 10448
## 2 SydneyAirport  5005
## 3    PearceRAAF    NA
## 4          Perth  3547
## 5        Bendigo  3069
```

7 Removing Columns

```
tail(ds$excess)

## [1] 32.0 -6.4 -2.2  1.6   NA 37.8

names(ds)

##  [1] "date"             "location"         "min_temp"
##  [4] "max_temp"         "rainfall"         "evaporation"
##  [7] "sunshine"         "wind_gust_dir"    "wind_gust_speed"
## [10] "wind_dir_9am"     "wind_dir_3pm"     "wind_speed_9am"
## [13] "wind_speed_3pm"   "humidity_9am"     "humidity_3pm"
## [16] "pressure_9am"     "pressure_3pm"     "cloud_9am"
## [19] "cloud_3pm"        "temp_9am"         "temp_3pm"
## [22] "rain_today"       "risk_mm"          "rain_tomorrow"
## [25] "temp_range"       "excess"

ds$excess <- NULL
tail(ds$excess)

## NULL

names(ds)

##  [1] "date"             "location"         "min_temp"
##  [4] "max_temp"         "rainfall"         "evaporation"
##  [7] "sunshine"         "wind_gust_dir"    "wind_gust_speed"
## [10] "wind_dir_9am"     "wind_dir_3pm"     "wind_speed_9am"
## [13] "wind_speed_3pm"   "humidity_9am"     "humidity_3pm"
## [16] "pressure_9am"     "pressure_3pm"     "cloud_9am"
## [19] "cloud_3pm"        "temp_9am"         "temp_3pm"
## [22] "rain_today"       "risk_mm"          "rain_tomorrow"
## [25] "temp_range"
```

8 Subset Data

Exercise: Discuss the subset function.

9 Wide to Long Data

Let's take a sample dataset to illustrate the concepts of wide and long data.

```
dss <- subset(ds, date==max(date))
dim(dss)

## [1] 46 25

head(dss)

##              date      location min_temp max_temp rainfall evaporation
## 1767  2014-01-30        Albury     20.6     39.7        0          NA
## 3503  2014-01-30 BadgerysCreek     16.7     34.0        0          NA
## 5239  2014-01-30         Cobar     23.1     38.3        0        14.0
....
```

This data is in wide format. We can convert it to long format, which is sometimes useful when using, for example, ggplot2 (Wickham and Chang, 2013). We use reshape2 (Wickham, 2012b) to do this. In long format we essentially maintain a single measurement per observation. The measurement for our data are all those columns recording some measure of the weather—that is, all variables except for `date` and `location`.

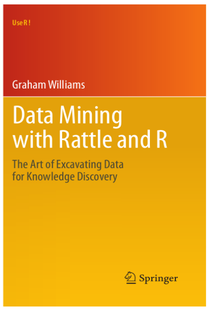
```
library(reshape2)
dssm <- melt(dss, c("date", "location"))
dim(dssm)

## [1] 1058    4

head(dssm)

##         date      location variable value
## 1 2014-01-30        Albury min_temp  20.6
## 2 2014-01-30 BadgerysCreek min_temp  16.7
## 3 2014-01-30         Cobar min_temp  23.1
....

tail(dssm)

##            date     location   variable value
## 1053 2014-01-30   SalmonGums temp_range  19.6
## 1054 2014-01-30      Walpole temp_range  13.5
## 1055 2014-01-30       Hobart temp_range    11
....

dssm[sample(nrow(dssm), 6),]

##           date     location     variable value
## 52  2014-01-30    Newcastle     max_temp  31.2
## 781 2014-01-30 AliceSprings    cloud_3pm     1
## 918 2014-01-30   Launceston   rain_today    No
....
```

This is now clearly long data.

10 Long to Wide Data

```
dssmc <- dcast(dssm, date + location ~ variable)
dim(dss)

## [1] 46 25

dim(dssmc)

## [1] 46 25

head(dss)

##              date      location min_temp max_temp rainfall evaporation
## 1767  2014-01-30        Albury     20.6     39.7        0          NA
## 3503  2014-01-30 BadgerysCreek     16.7     34.0        0          NA
## 5239  2014-01-30         Cobar     23.1     38.3        0        14.0
## 6975  2014-01-30  CoffsHarbour     16.8     27.7        0        19.8
## 8711  2014-01-30         Moree     16.5     34.6        0         9.6
## 10478 2014-01-30     Newcastle     18.0     31.2        0          NA
##       sunshine wind_gust_dir wind_gust_speed wind_dir_9am wind_dir_3pm
## 1767        NA             N              35           SE           NE
## 3503        NA             E              35           NE          ENE
....

head(dssmc)

##         date      location min_temp max_temp rainfall evaporation sunshine
## 1 2014-01-30      Adelaide     21.7     36.7        0           8     10.7
## 2 2014-01-30        Albany     16.2     24.1        0         7.4       12
## 3 2014-01-30        Albury     20.6     39.7        0        <NA>     <NA>
## 4 2014-01-30  AliceSprings     23.2     41.1        0        14.2     12.9
## 5 2014-01-30 BadgerysCreek     16.7       34        0        <NA>     <NA>
## 6 2014-01-30      Ballarat     11.2     34.9        0        <NA>     <NA>
##   wind_gust_dir wind_gust_speed wind_dir_9am wind_dir_3pm wind_speed_9am
## 1            SW              35          ESE          WSW              2
## 2          <NA>            <NA>           SE          ESE             17
....
```

11 Further Reading



Use R!

Graham Williams

Data Mining with Rattle and R

The Art of Excavating Data for Knowledge Discovery

Springer

The Rattle Book, published by Springer, provides a comprehensive introduction 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.

12 References

Eddelbuettel D, Francois R (2013). *Rcpp: Seamless R and C++ Integration.* R package version 0.10.6, URL http://www.rcpp.org,http://dirk.eddelbuettel.com/code/rcpp.html,http://blog.r-enthusiasts.com/tag/rcpp/.

R Core Team (2013). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Wickham H (2012a). *plyr: Tools for splitting, applying and combining data.* R package version 1.8, URL http://CRAN.R-project.org/package=plyr.

Wickham H (2012b). *reshape2: Flexibly reshape data: a reboot of the reshape package.* R package version 1.2.2, URL http://CRAN.R-project.org/package=reshape2.

Wickham H, Chang W (2013). *ggplot2: An implementation of the Grammar of Graphics.* R package version 0.9.3.1, URL http://CRAN.R-project.org/package=ggplot2.

Wickham H, Francois R (2014). *dplyr: dplyr: a grammar of data manipulation.* R package version 0.1, URL http://CRAN.R-project.org/package=dplyr.

Williams GJ (2009). "Rattle: A Data Mining GUI for R." *The R Journal*, **1**(2), 45–55. URL http://journal.r-project.org/archive/2009-2/RJournal_2009-2_Williams.pdf.

Williams GJ (2011). *Data Mining with Rattle and R: The art of excavating data for knowledge discovery.* Use R! Springer, New York. URL http://www.amazon.com/gp/product/1441998896/ref=as_li_qf_sp_asin_tl?ie=UTF8&tag=togaware-20&linkCode=as2&camp=217145&creative=399373&creativeASIN=1441998896.

Williams GJ (2014). *rattle: Graphical user interface for data mining in R.* R package version 3.0.2, URL http://rattle.togaware.com/.

This document, sourced from TransformO.Rnw revision 282, was processed by KnitR version 1.5 of 2013-09-28 and took 4.4 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 06:09:23.