Advanced Scientific Computing with R

1. Overview

Michael Hahsler

Southern Methodist University

August 12, 2011

These slides are largely based on "An Introduction to R"
`http://CRAN.R-Project.org/`

Table of Contents

Course Overview

see Syllabus...

Table of Contents

R

- R is "GNU S". S is a language for statisticians developed at Bell Laboratories by John Chambers et al.
- R is designed by John Chambers and developed by the R Foundation.
- R is a language and environment for statistical computing and graphics
- R is the de facto standard to develop statistical software
- R implements variety of statistical and graphical techniques (linear and nonlinear modeling, statistical tests, time series analysis, classification, clustering, ...)

R II

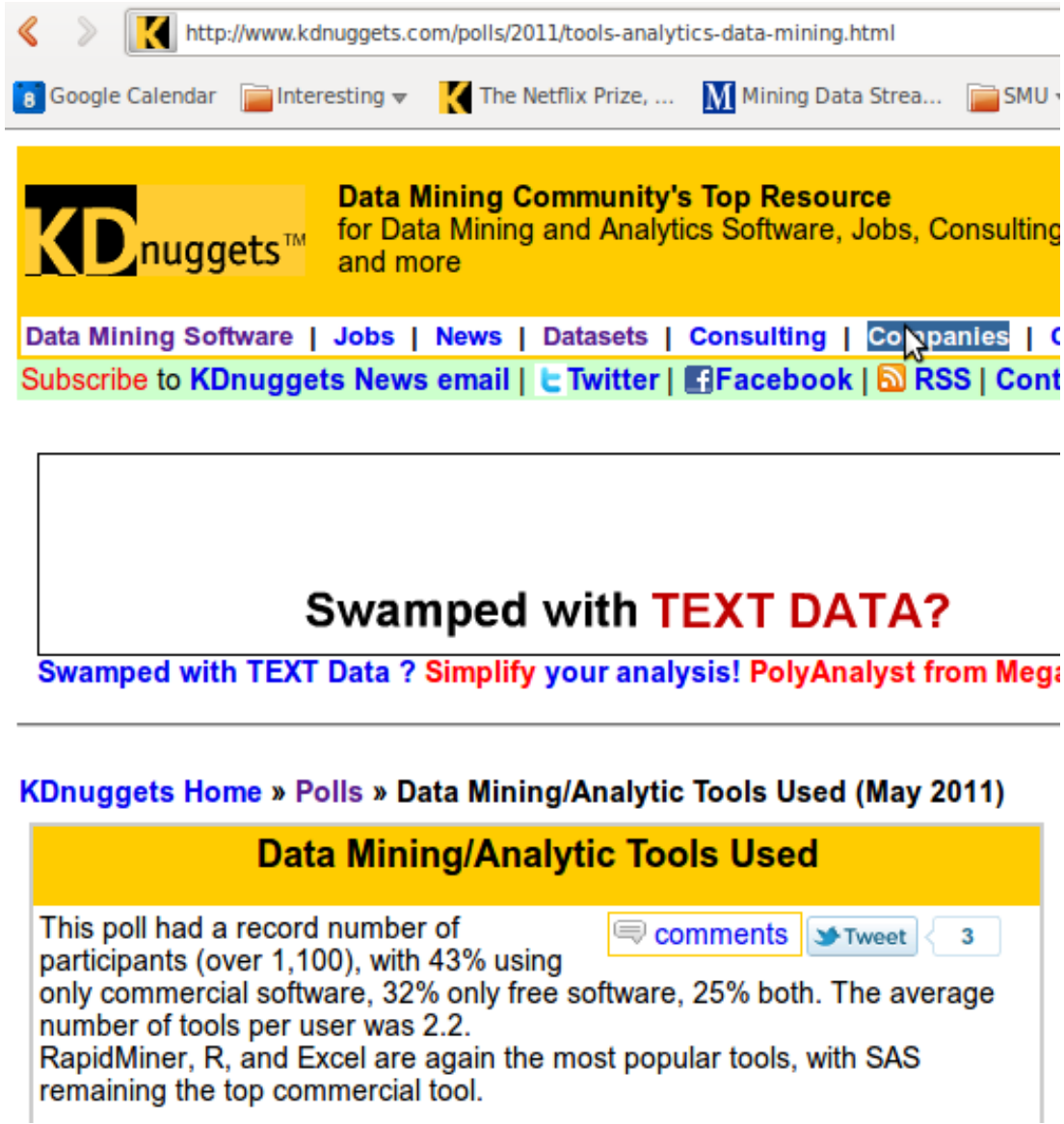
R provides

- effective data handling and storage
- operators for calculations on arrays (matrices)
- a large, coherent, integrated collection of intermediate tools for data analysis
- graphical facilities for data analysis and display
- simple and effective programming language (conditionals, loops, user defined recursive functions)
- extension mechanism with a large collection of packages

Why R?

- R is Open-Source and free to use
- R has a large and active community
- R provides state-of-the-art algorithm (> 3000 extension packages on CRAN, 2011)
- R creates beautiful visualizations (as seen in the New York Times and The Economist)
- R is used widely in industry (Revolution offers commercial solutions)
- R can be easily paralellized
- R is getting ready for big data (Revolution Analytics)

Why R?


http://www.kdnuggets.com/polls/2011/tools-analytics-data-mining.html
Google Calendar
Interesting
The Netflix Prize, ...
Mining Data Strea...
SMU
KDnuggets™
Data Mining Community's Top Resource
for Data Mining and Analytics Software, Jobs, Consulting
and more
Data Mining Software | Jobs | News | Datasets | Consulting | Companies |
Subscribe to KDnuggets News email | Twitter | Facebook | RSS | Cont
Swamped with TEXT DATA?
Swamped with TEXT Data ? Simplify your analysis! PolyAnalyst from Mega
KDnuggets Home » Polls » Data Mining/Analytic Tools Used (May 2011)
Data Mining/Analytic Tools Used
comments
Tweet
3
This poll had a record number of participants (over 1,100), with 43% using only commercial software, 32% only free software, 25% both. The average number of tools per user was 2.2.
RapidMiner, R, and Excel are again the most popular tools, with SAS remaining the top commercial tool.

Why R?

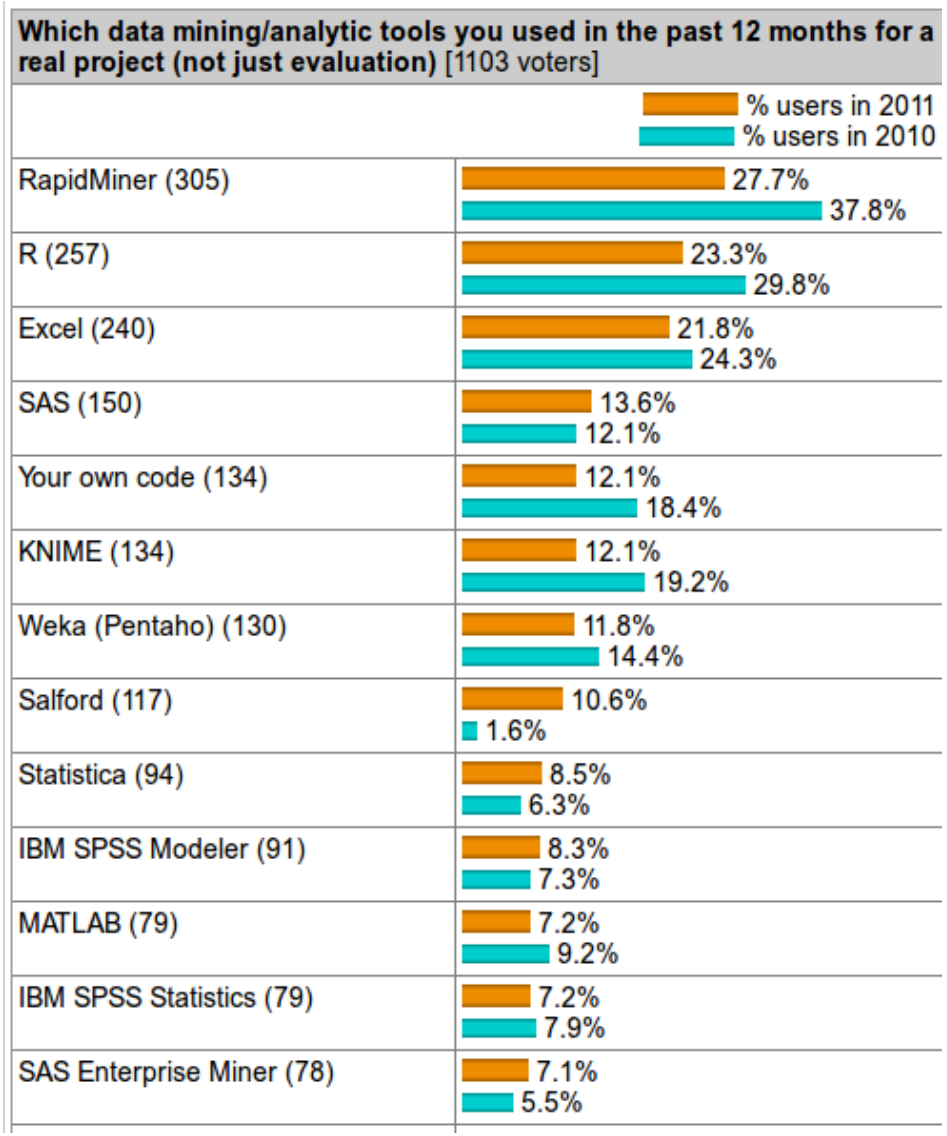

Which data mining/analytic tools you used in the past 12 months for a real project (not just evaluation) [1103 voters]
% users in 2011
% users in 2010
RapidMiner (305)
27.7%
37.8%
R (257)
23.3%
29.8%
Excel (240)
21.8%
24.3%
SAS (150)
13.6%
12.1%
Your own code (134)
12.1%
18.4%
KNIME (134)
12.1%
19.2%
Weka (Pentaho) (130)
11.8%
14.4%
Salford (117)
10.6%
1.6%
Statistica (94)
8.5%
6.3%
IBM SPSS Modeler (91)
8.3%
7.3%
MATLAB (79)
7.2%
9.2%
IBM SPSS Statistics (79)
7.2%
7.9%
SAS Enterprise Miner (78)
7.1%
5.5%

Table of Contents

Installing R

R is available for Linux/Unix, Windows, OS X and as source code.

`http://cran.r-project.org/`

Table of Contents

A first session

Create a working directory and start R.

```
R> x <- 1:10          # create a vector x
R> x
 [1]  1  2  3  4  5  6  7  8  9 10
R> y <- x + 1         # add one and assign to y
R> y
 [1]  2  3  4  5  6  7  8  9 10 11
R> # show objects in environment
R> ls()
[1] "txt" "x"   "y"


R> # leave R
R> q()
```

How to get help

R comes with online help

```
R> ? ls                    # get help
R> help("ls")
R> # same as above
R> help.start()
R> # start help browser
R> ?? solve                # keyword search
```

Further help can be found at `http://cran.r-project.org/`

- Manuals section (read: "An Introduction to R")
- Task Views section to find packages
- Search section to find answers (mailing lists, etc.)

The R language

- R is case sensitive
- expressions are evaluated, printed and the result is lost unless assigned with <-
- Commands are separated either by a semi-colon (';'), or by a newline
- expressions are grouped by braces ('' and '')
- Comments start with a hashmark ('#')

Data permanency

During an R session, objects are created and stored by name:

```
R> ls()
[1] "txt" "x"   "y"
```

Objects are kept over several sessions in a file (`.RData`). Objects can be removed.

```
R> rm(x)
R> # remove x
R> ls()
[1] "txt" "y"
```

Table of Contents

Vectors and assignment

Vectors are the basic data structure in R. Scalars do not exist! Almost all numbers are seen as "numeric" (double).

```
R> 1
[1] 1
```

```
R> x <- c(10.4, 5.6, 3.1, 6.4, 21.7)  # c combines values
R> x
[1] 10.4  5.6  3.1  6.4 21.7
R> 1/x                                # element-wise division
[1] 0.0962 0.1786 0.3226 0.1562 0.0461
R> y <- c(x, 0, x)                    # more combination
R> y
 [1] 10.4  5.6  3.1  6.4 21.7  0.0 10.4  5.6  3.1  6.4 21.7
```

Vector arithmetic

```
R> x
[1] 10.4  5.6  3.1  6.4 21.7
R> y
 [1] 10.4  5.6  3.1  6.4 21.7  0.0 10.4  5.6  3.1  6.4 21.7

R> x+y  # elements of the shorter array are recycled!
 [1] 20.8 11.2  6.2 12.8 43.4 10.4 16.0  8.7  9.5 28.1 32.1

R> sum(x)
[1] 47.2
R> length(x)
[1] 5
```

Sequences and Integers

```
R> s1 <- 1:5              # sequence of integers
R> s1
[1] 1 2 3 4 5
R> class(s1)
[1] "integer"
R> s2 <- seq(-1, 1, by=.2)    # using seq()
R> s2
 [1] -1.0 -0.8 -0.6 -0.4 -0.2  0.0  0.2  0.4  0.6  0.8  1.0

R> rep(s1, times=2)
 [1] 1 2 3 4 5 1 2 3 4 5
R> rep(s1, each=2)
 [1] 1 1 2 2 3 3 4 4 5 5
```

Try `? seq` and `? rep`

Logical vectors

```
R> x
[1] 10.4  5.6  3.1  6.4 21.7
R> l <- x>13    # compare each value in x
R> l
[1] FALSE FALSE FALSE FALSE  TRUE
R> class(l)
[1] "logical"
R> as.numeric(l)
[1] 0 0 0 0 1
```

The usual relational operators are available (e.g., `<`, `<=`, `>`, `>=`, `==`, `!=`, `&`, `|`). See `?"<"` and `?"&"` (quotation marks are necessary!)

Missing Values/Infinity

```
R> z <- c(1:3,NA)
R> z
[1]  1  2  3 NA
R> ind <- is.na(z)      # find missing values
R> ind
[1] FALSE FALSE FALSE  TRUE
R> 0/0                  # creates a NaN (not a number)
[1] NaN
R> 2^5000               # infinity
[1] Inf
```

See ?NA and ?Inf

Character vectors

```
R> string <- c("Hello", "Ola")
R> string
[1] "Hello" "Ola"
R> # pasting strings together
R> paste(string, "World!")
[1] "Hello World!" "Ola World!"
R> labs <- paste(c("X","Y"), 1:10, sep="")
R> labs
 [1] "X1"  "Y2"  "X3"  "Y4"  "X5"  "Y6"  "X7"  "Y8"  "X9"
[10] "Y10"
```

See ?paste

Selecting and modifying subsets

```
R> x
[1] 10.4  5.6  3.1  6.4 21.7
R> # select the first element (index starts with 1!)
R> x[1]
[1] 10.4
R> # remove the first element
R> x[-1]
[1]  5.6  3.1  6.4 21.7
R> # select elements (integer vector)
R> x[2:4]
[1] 5.6 3.1 6.4
R> # select elements (logical vector)
R> x[x>7]
[1] 10.4 21.7
R> # replace elements
R> x[x>7] <- NA
R> x
[1]  NA 5.6 3.1 6.4  NA
```

Selecting and modifying subsets II

```
R> # using names
R> fruit <- c(5, 10, 1, 20)
R> names(fruit) <- c("orange", "banana", "apple", "peach")
R> fruit
orange banana  apple  peach
     5     10      1     20
R> lunch <- fruit[c("apple","orange")]
R> lunch
 apple orange
     1      5
```

See ?"["

Table of Contents

Exercises

1. Create a vector with 10 numbers (3, 12, 6, -5, 0, 8, 15, 1, -10, 7) by you and assign it to x.
2. What is the "data type" of x? How can you find out?
3. Subtract 5 from the 2nd, 4th, 6th, etc. element in x.
4. Compute the sum and the average for x (there are functions for that).
5. Reverse the order of the elements in x.
6. Find out which numbers in x are negative.
7. Remove all entries with negative numbers from x.
8. How long is x now (use a function).
9. Remove x from the environment/workspace (session).
10. Create the a vector of strings containing "CSE 8001", "CSE 8002", ..., "CSE 8100" using paste.