Starting with R

John Mount

The Berkeley R Language Beginner Study Group

9/17/2013

Outline

- ☐ Why use R?
- ☐ A quick example analysis.
- ☐ A bit (positive and negative) on the R programming language.
- ☐ (back to positive) Graphs in R are awesome.
- ☐ Further reading.
- ☐ Q&A

Why choose R

A number of plusses

- A number of powerful and correct statistical and machine learning techniques are available out of the box.
- R is a scripting/programming system well suited for repetition of procedures.
- R can interoperate with flat files, xlxs files and databases.
- R is open source and cross-platform.
- There are a large number of good books available on R.

You are always on your penultimate analysis. Upstream bugs get fixed and you have to re-run on revised data all of the time.

Funny only to non-starving students

- With a budget of $1,000 you can:
  - Get laughed out of the room by most statistical software or machine learning software vendors.
  - Set up a complete R work environment:
    - $700 new laptop computer (reformat and put a decent Linux on it).
    - Software: R, RStudio, RevolutionR, Postgresql, H2DB, git, vi/emacs, SQuirrelSQL, Hadoop.
    - Purchase 6 books on R and data science.

If you do stay with Windows try RevolutionR for its step debugger.

R is organized around the data frame

- Data frames are a lot like SQL tables
  - The have a schema (list of columns with names and types).
  - They have homogeneous rows (all rows must obey the schema).

Some key R packages

- reshape2: allows conversion between long and wide data formats. At least as powerful as pivot tables.
- sqldf: allows data frames to be treated directly as SQL tables. Allows very powerful aggregation.
- DBI/RJDBC: allows R to read data from any JDBC compliant database.
- xlsx: read modern Excel xlsx spreadsheets.
- ggplot2: great graphics package.
- knitr: worksheets and literate programming.

Machine learning algorithms to try early

- ☐ lm
- ☐ glm
- ☐ gam
- ☐ rpart
- ☐ svm
- ☐ randomForest

Example: logistic regression in R

```
CarData <- read.table(url('http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data'),
     sep=',',col.names=c('buying','maintenance','doors','persons','lug_boot','safety','rating')

print(CarData[1:5,])
  buying maintenance doors persons lug_boot safety rating
1  vhigh       vhigh     2       2    small    low  unacc
2  vhigh       vhigh     2       2    small    med  unacc
3  vhigh       vhigh     2       2    small   high  unacc
4  vhigh       vhigh     2       2      med    low  unacc
5  vhigh       vhigh     2       2      med    med  unacc


print(summary(CarData))
   buying    maintenance   doors     persons     lug_boot    safety      rating
 high :432   high :432   2    :432   2   :576   big  :576   high:576   acc  : 384
 low  :432   low  :432   3    :432   4   :576   med  :576   low :576   good :  69
 med  :432   med  :432   4    :432   more:576   small:576   med :576   unacc:1210
 vhigh:432   vhigh:432   5more:432

> apply(CarData,2,FUN=class)
        buying    maintenance          doors        persons       lug_boot         safety         rating
   "character"    "character"    "character"    "character"    "character"    "character"    "character"
testTrainGroup
   "character"
```

LR contd.

```
CarData$testTrainGroup <- sample(100,size=dim(CarData)[[1]],replace=T)
cTrain <- subset(CarData,testTrainGroup>20)
cTest <- subset(CarData,testTrainGroup<=20)
m <- glm(rating!='unacc' ~ buying + maintenance + doors + persons + lug_boot +safety,
     family=binomial(link = "logit"),data=cTrain)
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred

cTest$pred <- predict(m,newdata=cTest,type='response')
> print(with(cTest,table(rating,predT=pred>=0.5)))
        predT
rating   FALSE TRUE
  acc        8   66
  good       0   15
  unacc    240    3
  vgood      0    9
```

R as a programming language

R is a “scripting style” language

- Lots of “helpful” conventions (such as automatic variables, allowing truncated column names, non-signaling NULL and so on).
- On the face of it easy call by value semantics and automatic garbage collection (cuts down on dangerous value aliasing).
- A neat functional notation core, but at least two incompatible object oriented systems bolted on (S3, and S4).

It is best to tell people R has “call by value” calling semantics

```
> a <- c(1,2)
> f <- function(a) { a[[1]] <-7; a }
> f(a)
[1] 7 2
> print(a)
[1] 1 2
```

This is a lot like Matlab. Except in Matlab most everything is a matrix, in R everything is a vector or list.

R actually has fairly arbitrary calling semantics

```
> g <- function(z) {1}
> z+1
Error: object 'z' not found
> g(z+1)
[1] 1

> f <- function(expr) { expr + 1}
> f(5)
[1] 6
> squealer <- function(x) { print(paste('see',x)); x}
> f(squealer(5))
[1] "see 5"
[1] 6
> f(squealer(x+1))
Error in paste("see", x) : object 'x' not found
> f(x+1)
Error in f(x + 1) : object 'x' not found
> f <- function(expr) { print(deparse(substitute(expr))) }
> f(squealer(5))
[1] "squealer(5)"
> f(x+1)
[1] "x + 1"
```

R actually uses lazy evaluation of expressions. And R allows non referentially transparent access to the calling information. The dirty secret of programming is you want all other code to be referentially transparent so when you try something incredibly clever (and not referentially transparent) things work the way you hoped.

R language warts / common errors

c() is *not* Lisp cons

- c() is a vector operator. It does not like structures (but also does not defend itself from structures).

```
> class(m)
[1] "glm" "lm"
> class(c(m,m)[[1]])
[1] "numeric"
> class(c(m,m)[1])
[1] "list"
```

[] is not really an element selection operator

- [] is actually a subset operator that is willing to cast down from a data frame to a single row.
- Really you should always use [,drop=F] if you want a data frame and [,drop=T] if you want a row and [[]] when you want an element.
- Data frames work as lists of columns not arrays of rows (so you can't actually use [[]] on them for much).

```
> class(CarData[1,,drop=F])
[1] "data.frame"
> class(CarData[1,,drop=T])
[1] "list"
> dim(CarData)
[1] 1728    8
> length(CarData[[1]])
[1] 1728
```

Matrices are different than data frames

```
> mat <- matrix(c(1,2,3,4),nrow=2,ncol=2)
> print(mat)
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> class(mat[1,])
[1] "numeric"
> class(mat[1,,drop=F])
[1] "numeric"
> mat[1,]
[1] 1 3
```

Data frames (in general) don't like list entries, but don't defend against them

```
> d1 <- data.frame(
   Date=as.POSIXlt(strptime('6/1 Sat 2013','%m/%d %a %Y')),X0=c(99),X1=c(99))
> class(d1$Date)
[1] "POSIXct" "POSIXt"
> d2 <- data.frame(Date=c('6/1 Sat 2013'),X0=c(99),X1=c(99))
> d2$Date <- c(as.POSIXlt(strptime(d2$Date,'%m/%d %a %Y')))
> class(d2$Date)
[1] "POSIXlt" "POSIXt"
```

Lazy evaluation and implicit print are not always your friends

```
> ggplot(data=CarData)
Error: No layers in plot
> f <- function() { ggplot(data=CarData); 7 }
> f()
[1] 7
> f <- function() { print(ggplot(data=CarData)); 7 }
> f()
Error: No layers in plot
```

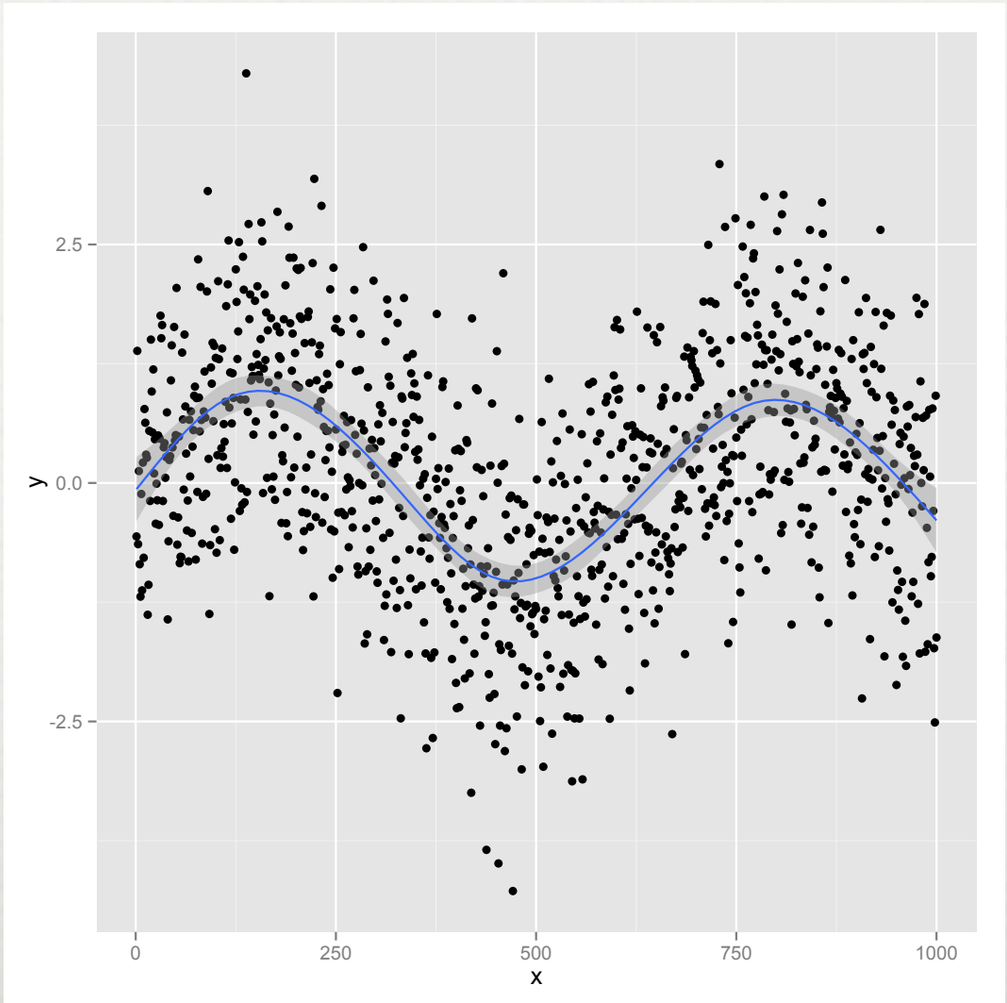
3: Learn to Examine Structures

- unclass()
- str()
- dput()
- names()
- dimnames()
- slotNames()
- getSlots()

R is object oriented, so in principle just about any call can be hijacked to be a method of the class of the object you are trying to examine (like print() and summary()). But deep down most complicated objects have an underlying list structure.

(back to positive) Graphs in R are awesome

```
> library('ggplot2')
> d <- data.frame(x=1:1000)
> d$y <- sin(0.01*d$x) + rnorm(dim(d)[[1]])
> ggplot(data=d,aes(x=x,y=y)) + geom_point() + geom_smooth()
```



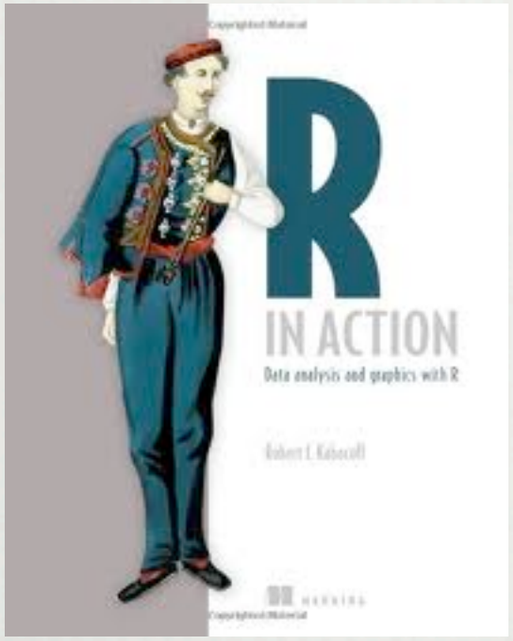
Further reading

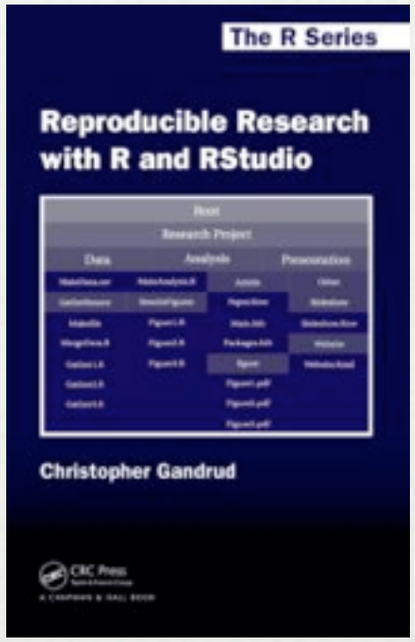
Web resources

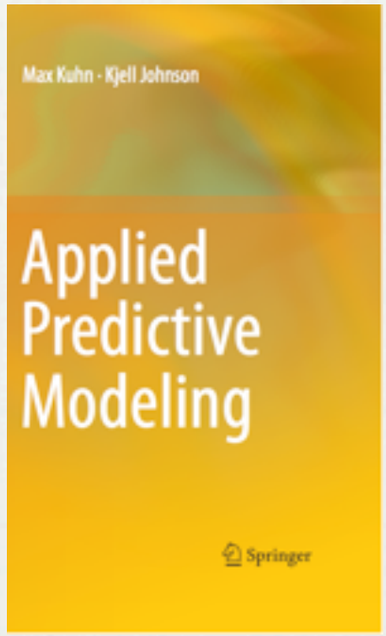
- http://www.win-vector.com/blog/2009/09/survive-r/
- http://learnr.wordpress.com/
- http://groups.google.com/group/help-R
- r-project.org
- stackoverflow.com
- http://stackoverflow.com/questions/102056/how-to-search-for-r-materials
- www.rseek.org
- http://search.r-project.org/
- http://h4dev.com/entries?search=r+search+engine

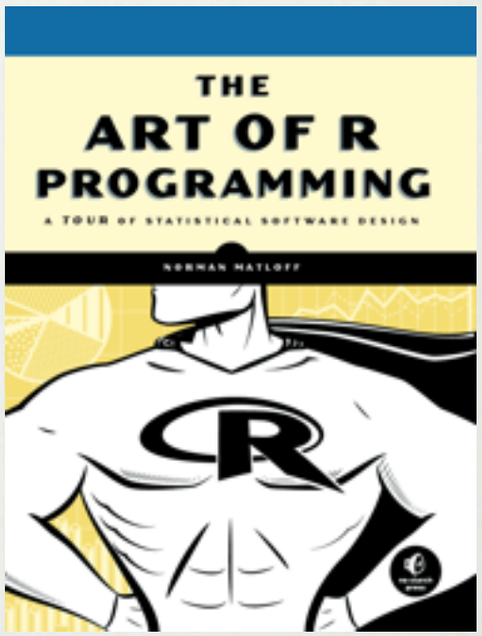
It is very unfortunate R is a letter of the alphabet. Also, I am not a huge fan of any R-based search systems I have used. However, help() is very very useful for finding arguments of methods you already know about. Quick apology-in my earlier draft Univ. of Minnesota database I mention on slide 6 is not publicly available. Gene Tani offered this correction and his database of resources (which seems to be the most comprehensive, listing 13 sources): http://h4dev.com/entries?search=r+search+engine .

My current recommended R library





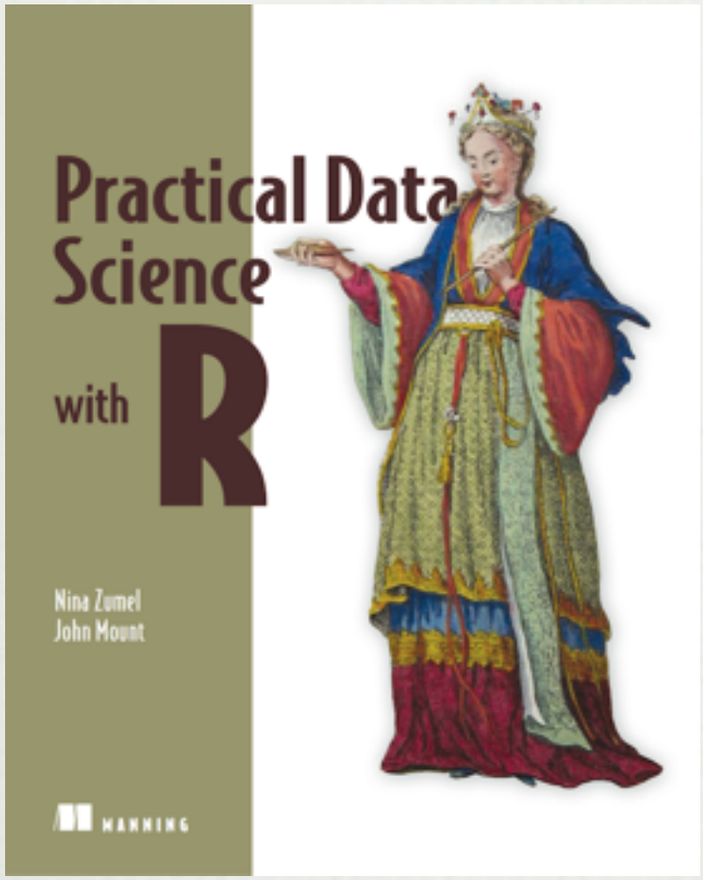




Win-Vector LLC
www.win-vector.com

"R in Action" is currently my favorite book for solid statistics and modeling in R (there is a new edition on the way). "Reproducible Research with R and RStudio" gives a good overview of the tool and work environment you should put around R (RStudio, git, knitr and so on). "Applied Predictive Modeling" is a excellent new book on machine learning and statistics using R. "The Art of R Programming" remains where I go for programming and debugging questions.

My future recommended R library

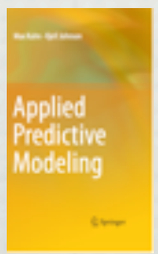


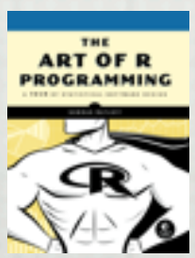
Practical Data Science with R
PART 1: INTRODUCTION TO DATA SCIENCE
- **Chapter 1: The Data Science Process**
- **Chapter 2: Starting With R And Data**
- **Chapter 3: Exploring Data**
- **Chapter 4: Managing Data**

PART 2:MODELING METHODS
- **Chapter 5: Choosing and Evaluating Models**
- **Chapter 6: Using Memorization Methods**
- **Chapter 7: Using Linear and Logistic Regression**
- **Chapter 8: Using Unsupervised Methods**
- **Chapter 9: Exploring Advanced Methods**

PART 3: USING RESULTS
- **Chapter 10: Managing Models in Production**
- **Chapter 11: Building successful presentations**
- **Chapter 12: Presenting to different audiences**
- **Chapter 13: Deployment Documentation**
- **Chapter 14: Conclusions, what to take away**

Appendices
- **Appendix A: Working With R And Other Tools**
- **Appendix B: Important Statistical Concepts**
- **Appendix D: Further Reading**





Win-Vector LLC
www.win-vector.com

Thank you

Slides up soon at: http://www.win-vector.com/blog/