DATA ANALYSIS FOR THE MASSES

Jan Vitek

76% of data miners prefer R

http://www.rexeranalytics.com/files/Rexer_Data_Science_Survey_Highlights_Apr-2016.pdf

The state of DA today?

The face of DA for the next decade?

The process of inspecting, cleansing, transforming, and modeling data in order to discover useful information...

76% of data miners prefer R

The average analytics professional reports using 5 software tools

R is the tool used by the most people (76%)

A large number of tools have substantial market penetration

R

Introduction to R for Quantitative Finance

Functional and Phylogenetic Ecology in R

Using R for Digital Soil Mapping

Introduction to R for Political Analysis

Learning R for Geospatial Analysis

The R Companion to Political Analysis

DEEP LEARNING Made Easy with R

Hands-on Matrix Algebra using R

Introduction to R for Business Intelligence

Social Media Mining with R

Bioinformatics with R Cookbook
THE R PROJECT FOR STATISTICAL COMPUTING

R is a free software environment for statistical computing and graphics.

- 2,000,000+ users in academe and industry
- 11,172 open source packages on CRAN
- 195,609 questions on stack overflow
- 900+ attendees useR! conference
- 55+ user groups worldwide

FORTY YEARS OF S

- SCS Library (AT&T) convenient statistical routines written in FORTRAN for portability
- 1976: First release of S; influenced by Tukey’s work, allows interactive use of SCS routines for EDA
- Key idea was a uniform interface to native routines, same calling sequence and a standard data representation
- 1978: S integrated to the newly developed UNIX operating system, takes advantage of its portability and tooling
- 1985: John Chambers & Richard Becker publish S book

TWENTY FIVE YEARS OF R

Need a free teaching language for statistical computing

- Started as a 1000 line Scheme interpreter
- 1993 R announced on an S mailing list
- 1995 R open-sourced (GNU)
- 1997 R core group
- 2000 R 1.0
- 2003 R Foundation
- 2004 UseR! conference in Vienna
- 2015 R Consortium

Latest version has over 800,000 lines of C and R code
TWENTY YEARS OF CRAN

Provide a decentralized and modularized way to develop software
Inspired by Perl (CPAN) and LaTeX (CTAN).
• standardized format for packages, vignettes and tests
• automated and manual checking of incoming software
• testing on 3 releases of R, for 12 OSes and 7 architectures
• manual communication with maintainers

Growth
• 6 accepted packages/day
• 230 updates/day

NUTSHELL

What R is…
• vectorized
• dynamic
• lazy
• functional

What R isn’t…
• fast
• low-footprint
• concurrent
• distributed

LAZY

Lazy evaluation delays evaluation of arguments until their value is needed while avoiding repeated evaluations.

```r
assert(x==2, print("Error"))
assert<-function(C,P) if(C){P}
with(fd, carb*den)
```
Dynamic languages support run-time extension of programs, by adding new code, by extending definitions or by modifying types.

```r
with <- function(df, e,...)
  eval( substitute(e),
       df, parent.frame))
```

Modification of a value creates a copy, unless it is unshared.

This allows the following pleasing property to hold:

```r
assert(x[1]==5)
f(x)        f <- function(a) { a[1]<-0}
assert(x[1]==5)
```

Modification of a value creates a copy, if the value is shared:

```
x <- 3
y <- x
```

```
x[1] <- 3
```

```
x <- 1
y <- x
```

```
x[1] <- 1
```

```
y[1] <- 1
```
Tidy Data

Import data is often ‘messy’, obtaining tidy data entails changing representation, validating encodings, and reshaping so that

- Each variable forms a column.
- Each observation forms a row.
- Each type of observational unit is a data frame

“80% of data analysis is … cleaning & preparing” — Dasu, Johnson 2003

Hadley Wickham
(RStudio)

Tidy Data
J. of Statistical Software, 2014

grades <- tbl_df(read.table("tbl.csv"))

<table>
<thead>
<tr>
<th>ID</th>
<th>Test</th>
<th>Fall</th>
<th>Spring</th>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final</td>
<td>90</td>
<td>94</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>Midtm</td>
<td>16</td>
<td>??</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>Final</td>
<td>93</td>
<td>90</td>
<td>96</td>
</tr>
</tbody>
</table>

grades[grades == "??"] <- NA

<table>
<thead>
<tr>
<th>ID</th>
<th>Test</th>
<th>Fall</th>
<th>Spring</th>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final</td>
<td>90</td>
<td>94</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>Midtm</td>
<td>16</td>
<td>NA</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>Final</td>
<td>93</td>
<td>90</td>
<td>96</td>
</tr>
</tbody>
</table>
TIDY

grades %>%
  mutate_at('Spring', as.character) %>%
  mutate_at('Spring', as.integer) %>%
  complete(ID, Test) %>%
  gather(Semester, Score, Fall:Winter) %>%
  spread(Test, Score) -> grades2

grades %>%
  mutate_at('Spring', as.character) %>%
  mutate_at('Spring', as.integer) %>%
  complete(ID, Test) %>%
  gather(Semester, Score, Fall:Winter) %>%
  spread(Test, Score) -> grades2

TIDY

<table>
<thead>
<tr>
<th>ID</th>
<th>Test</th>
<th>Fall</th>
<th>Spring</th>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final</td>
<td>90</td>
<td>94</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>Midtm</td>
<td>16</td>
<td>NA</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>Final</td>
<td>93</td>
<td>90</td>
<td>96</td>
</tr>
</tbody>
</table>

grades %>%
  mutate_at('Spring', as.character) %>%
  mutate_at('Spring', as.integer) %>%
  complete(ID, Test) %>%
  gather(Semester, Score, Fall:Winter) %>%
  spread(Test, Score) -> grades2

TIDY

<table>
<thead>
<tr>
<th>ID</th>
<th>Test</th>
<th>Fall</th>
<th>Spring</th>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final</td>
<td>90</td>
<td>94</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>Midtm</td>
<td>16</td>
<td>NA</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>Final</td>
<td>93</td>
<td>90</td>
<td>96</td>
</tr>
</tbody>
</table>

grades %>%
  mutate_at('Spring', as.character) %>%
  mutate_at('Spring', as.integer) %>%
  complete(ID, Test) %>%
  gather(Semester, Score, Fall:Winter) %>%
  spread(Test, Score) -> grades2
**TRANSFORM**

A grammar of data manipulation

**One data frame verbs**
- `filter()` to select observations based on their values.
- `arrange()` to reorder the observations
- `select()`, `rename()` to select variables based on their names
- `mutate()` to add new variables as functions of existing ones
- `summarise()` to condense multiple values to a single value
- `sample_n()` to take random samples

**Grouping verb**
- `group_by()` perform any operation “by group”

**Multiple table verbs**
- `inner_join(x,y), left_join, right_join, full_join` only includes observations in both x and y, in x, in y, in x or y
- `semi_join(x,y), anti_join` keep, drop, all in x that have a match in y
- `intersect(x,y), union, setdiff` set operations

---

**VISUALIZE**

The grammar of graphics

Statistical graphic specifications are expressed in:

1) **DATA**: a set of operations that create variables from datasets
2) **TRANS**: variable transformations (e.g., rank)
3) **SCALE**: scale transformations (log)
4) **COORD**: a coordinate system (polar)
5) **ELEMENT**: graphs (points) and aesthetic attributes (color)
6) **GUIDE**: one or more guides (axes, legends, etc.)
ggplot(mpg, aes(x=model, y=hwy)) + geom_bar(stat="identity")

ggplot(mpg, aes(x=displ, y=hwy)) + geom_point()

ggplot(mpg, aes(x=displ, y=hwy)) + geom_point(color=class)
ggplot(mpg, aes(x=displ, y=hwy)) + geom_point(color=class, size=cyl) + theme_economist() + scale_colour_economist()

REPRODUCE

2001 Sweave — integration of R code into LaTeX to create dynamic reports, updated automatically if data or analysis change

2011 Knitr — inspired by Sweave. Expands it to support R Markdown, caching, TikZ graphics, other languages, outputs PDF, HTML, LaTeX…
BIg Data

matter: an R package for direct interactions with larger-than-memory datasets, stored in an arbitrary number of files of any size.
matter enables a wide variety of data exploration and manipulation steps and is extensible to many bioinformatics applications

```r
sims <- mapply(ctx, x, sum, function(...) {sum(...)})
@: commutative @: pure
@: visit(-1:0, 0:-1, 0:0, 0:1.1:0)
```

Evolving Language Ecosystems (ELE)

- Modern languages are characterized by rich ecosystems that include compilers, interpreters, IDEs, libraries, help pages, manuals, stackoverflow...
- To remain relevant, languages need to evolve, they must add new features, their libraries must adapt to new requirements, implementations must change to meet new performance goals.
- The ELE project explores the fundamental techniques and algorithms for evolving entire language ecosystems. We will reduce the cost of wide-ranging changes to programming languages and obviate the need for devising new languages.

Types

R is untyped, statically, and strongly typed, at runtime

What if we could add types, gradually, partially?

```r
function(x, narm=FALSE, dims=1L) {
  if (is.matrix(x))
    x <- as.matrix(x)
  if (dims < 1L ||
    dims > length(dim(x))-1L)
    stop("invalid 'dims'")
}
```

```r
function(x :~ Matrix(N,...),
  narm :: Logical = FALSE,
  dims :: Range(1,dim(x)-1L=1L) {
```
PARALLELISM

R’s implementation is not thread safe, how do we parallelize?

foreach allows iteration over elements in a collection without explicit loop counter in parallel via doMC (parallel/multicore), doMPI (Rmpi) …

partools Miscellaneous utilities for parallelizing large computations. Alternative to MapReduce. File splitting and distributed operations such as sort and aggregate.

The ELE project investigates safe thread parallelism

Following our collaboration with Intel

Synthesize OpenMP code for a subset of R

PERFORMANCE

R performance is hampered by dynamic features of the language

Tierney and Kalibera introduced a byte-code compiler

Oracle research is building a Java compiler

The ELE project builds LLVM JIT

Figure 2: Total time spent

Figure 3: Just-in-time compiler.

Figure 4: Slowdown of Python and R, normalized to C for the Shootout benchmarks.

Tomas Kalibera

CTU

Relative to C

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

0 1 2 3 4 5 6 7 8 9 10

CTU

INTERACTIVITY: the ability to quickly test out ideas is key to exploratory data analysis

DYNAMISM: the dynamic features of the language are key for learning and development

ECOSYSTEM: R is not a language it is a rich ecosystem

HUMANS IN THE LOOP: without a strong open source community R wouldn’t be here

TAKE-AWAYS
Data analysis for the masses