

Web Data Collection with R

Regular Expressions / RegEx - A Case Study

Peter Meißner / 2016-02-29 – 2016-03-04 / ECPR WSMT

Getting to know the page

Scraping Strategy

Scraping

Transforming / Reading Data

First Result

Extraction

Extension (1)

Getting to know the page

first glance at:

<http://ajps.org/list-of-reviewers/>

```
url <- "http://ajps.org/list-of-reviewers/"  
browseURL(url)
```

getting to know the page

- ▶ look at the source code (Cntr-U)
- ▶ inspecting elements (Cntr-Shift-I)

surprise

- ▶ reviewer lists are not part of the web page but available as PDF downloads

Scraping Strategy

getting PDFs

1. download page / load into R
 - ▶ `read_html()` [*rvest*]
2. extract anchor nodes `<a ...>`
 - ▶ `html_nodes(..., xpath=...)` [*rvest*]
3. extract href attribute from nodes
 - ▶ `html_attr(..., "href")` [*rvest*]
4. filter links (keep those entailing: 'review'; four digits; 'pdf')
 - ▶ `str_detect(..., "review.*\\d{4}.*pdf")` [*stringr*]
5. download PDFs to disk
 - ▶ `download.file(..., ..., mode="wb")` [*utils*]

extracting information from PDF

6. converting PDF to something we can work with

- ▶ e.g. Adobe Acrobat Pro
 - ▶ HTML, XML, TXT, ...
- ▶ e.g. Xpdf (<http://www.foolabs.com/xpdf/download.html>)
 - ▶ HTML, TXT, ...
 - ▶ WINDOWS:
<http://www.foolabs.com/xpdf/download.html> - add install path to path variable / see:
<http://www.computerhope.com/issues/ch000549.htm>
 - ▶ Linux e.g.: `sudo apt-get install poppler-utils`

7. load into R and use Regular Expressions extract information

Scraping

getting PDFs

```
# packages needed  
require(rvest)  
require(stringr)
```

getting PDFs

```
# url with list of reviews
url <- "http://ajps.org/list-of-reviewers/"

# get page
content <- read_html(url)

# get anchor (<a href=...>) nodes via xpath
ankers  <- html_nodes(content, xpath = "//a")

# get value of ankers' href attribute
hrefs   <- html_attr(ankers, "href",
                      default="NO HREF IN HERE")
```

getting PDFs

```
# filter links: should entail ...  
# 'review', four-digit number, 'pdf'  
pdf <- hrefs[ str_detect(hrefs, "review.*\\d{4}.*pdf") ]  
pdf  
  
## [1] "http://ajpsblogging.files.wordpress.com/2015/04/ajp  
## [2] "http://ajpsblogging.files.wordpress.com/2014/01/ajp  
## [3] "http://ajpsblogging.files.wordpress.com/2013/08/rev  
## [4] "http://ajpsblogging.files.wordpress.com/2013/08/rev  
## [5] "http://ajpsblogging.files.wordpress.com/2013/08/rev
```

getting PDFs

```
# names for PDFs on disk
```

```
basename(pdf)
```

```
## [1] "ajps-reviewers-2014.pdf" "ajps_reviewers_2013.pdf"
```

```
## [3] "reviewers_2012.pdf"      "reviewers_2011.pdf"
```

```
## [5] "reviewers_2010.pdf"
```

```
str_extract(pdf, "\\d{4}.pdf")
```

```
## [1] "2014.pdf" "2013.pdf" "2012.pdf" "2011.pdf" "2010.pdf"
```

```
pdf_names <- str_extract(pdf, "\\d{4}.pdf")
```

```
# download pdfs
```

```
for(i in seq_along(pdf)) {
```

```
  download.file(pdf[i], pdf_names[i], mode="wb")
```

```
}
```

Transforming / Reading Data

transforming PDFs - function

```
# WINDOWS: xpdf: http://www.foolabs.com/xpdf/download.html
#   add install path to path variable / see: http://www.com
# Linux: sudo apt-get install poppler-utils

pdftotext <- function(fname){
  fname_txt <- str_replace(fname, ".pdf", ".txt")
  system2(command = "pdftotext", args = fname)
  return(fname_txt)
}
```


transforming PDFs - execution

```
# transform PDFs to text  
pdftotext(pdf_names[1])  
pdftotext(pdf_names[2])  
pdftotext(pdf_names[3])  
pdftotext(pdf_names[4])
```

loading text

```
# load text of PDF  
text1 <- readLines("2013.txt", warn=FALSE)
```

first glance at text

```
substring(text1, 1, 60)[6:14]
```

```
## [1] "parentheses!at!the!end!of!each!reviewer's!name!ind  
## [2] "completed!in!2013.!!"  
## [3] "!"  
## [4] "Max!!Abrahms,!Johns!Hopkins!(!2!)!"  
## [5] "Alan!I.!Abramowitz,!Emory!University!(2!)!"  
## [6] "James!Adams,!UC!Davis!(4)!"  
## [7] "Claire!L.!Adida,!UCSD!(!2!)!"  
## [8] "Marina!Agranov!,!Caltech!(!1!)!"  
## [9] "John!S!Ahlquist!,!University!of!Wisconsin,!Madison"
```

- ▶ some useless/wrong characters → cleansing
- ▶ get rid of spaces
- ▶ get rid of parantheses
- ▶ information scheme is:
 FirstName Lastname, Institution (NumberOfReviews)
- ▶ followed by actual extraction

preparation

```
text1_tmp <-  
  text1 %>%  
    str_replace_all("[!\\f]", " ") %>%      # drop form feed  
    str_replace_all("\\]", " ") %>%          # drop ]  
    str_replace_all("\\(|\\)", "") %>%        # drop ( )  
    str_replace(" ,", ",") %>%              # correct space  
    str_replace_all("  ", " ") %>%          # correct space  
    str_trim()                               # correct space  
  
text1_tmp <-  
  text1_tmp[text1_tmp != ""] # drop empty lines  
  
text1_tmp <-  
  text1_tmp[-c(1:5)]           # drop non data
```

cleaned up

```
text1_tmp[1:10]
```

```
## [1] "Max Abrahms, Johns Hopkins 2"  
## [2] "Alan I. Abramowitz, Emory University 2"  
## [3] "James Adams, UC Davis 4"  
## [4] "Claire L. Adida, UCSD 2"  
## [5] "Marina Agranov, Caltech 1"  
## [6] "John S Ahlquist, University of Wisconsin, Madison  
## [7] "Faisal Ahmed, Oxford University 2"  
## [8] "T.K. Ahn, Seoul National University 1"  
## [9] "Ariel Ahram, Virginia Tech 1"  
## [10] "Deniz Aksoy, Princeton University 1"
```

First Result

Reviewers

```
length(grep("Konstanz", text1_tmp))
```

```
## [1] 6
```

```
length(grep("Harvard", text1_tmp))
```

```
## [1] 24
```

```
length(grep("Berlin", text1_tmp))
```

```
## [1] 3
```

```
length(grep("Bamberg", text1_tmp))
```

```
## [1] 0
```

```
length(grep("UCLA", text1_tmp))
```

```
## [1] 2
```

Extraction

names

```
names <-  
  text1_tmp %>%  
  str_extract("^.*?," ) %>%  
  str_replace_all(" |"," ") %>%  
  str_trim( )  
sample(names, 10)
```

```
## [1] "David Karol" "Charles Daniel Myers"  
## [3] "Anna Harvey" "Kåre Vernby"  
## [5] "Sean Cain" "Kent Tedin"  
## [7] "Matthew Lee Blackwell" "Amanda Driscoll"  
## [9] "Jay Gatrell" "Andrew Therriault"
```

institutions

```
institution <-  
  text1_tmp %>%  
  str_extract(",.*\\d") %>%  
  str_replace_all("^ ,|^,|^\\d$", "") %>%  
  str_trim()  
sample(institution, 7)
```

```
## [1] "University of Wisconsin"      "Hebrew University  
## [3] "Bucknell University"          "Erasmus University  
## [5] "University of Mississippi"    "The World Bank"  
## [7] "University of Houston"
```

reviews

```
reviews <-  
  text1_tmp %>%  
    str_extract("\\d+") %>%  
    as.numeric  
table(reviews)
```

```
## reviews  
##      1      2      3      4      5  
## 947 181  34    7    1
```

reviews

```
data.frame(n=reviews, names, institution)[reviews > 3, ]
```

##	n	names	institution
## 3	4	James Adams	UC Davis
## 27	4	Scott Ashworth	University of Chicago
## 541	4	Cindy D. Kam	Vanderbilt University
## 592	5	Gregory Koger	University of Miami
## 684	4	Neil Malhotra	Stanford University
## 950	4	Leslie Schwindt Bayer	Rice University
## 1095	4	Erik Voeten	Georgetown University
## 1152	4	Jonathan Woon	University of Pittsburgh

save data gathered so far

```
revdat <- data.frame(  
  reviews,  
  names,  
  institution,  
  stringsAsFactors = FALSE  
)  
save(revdat, file = "revdat.Rdata")
```

Extension (1)

geocoding institutions

```
require(ggmap)
# geocoding takes a while -> save results
# 2500 requests allowed per day
if ( !file.exists("scenario1_inst_geocoded_pos.rdata")){
  pos <- geocode(institution)
  geocodeQueryCheck()
  save(pos, file="scenario1_inst_geocoded_pos.rdata")
} else {
  load("scenario1_inst_geocoded_pos.rdata")
}
```

plot coordinates

```
mapWorld <- borders("world")
```

```
##
```

```
## # maps v3.1: updated 'world': all lakes moved to separate
```

```
## # 'lakes' database. Type '?world' or 'news(package="map")'
```

```
map <-
```

```
  ggplot() +
```

```
  mapWorld +
```

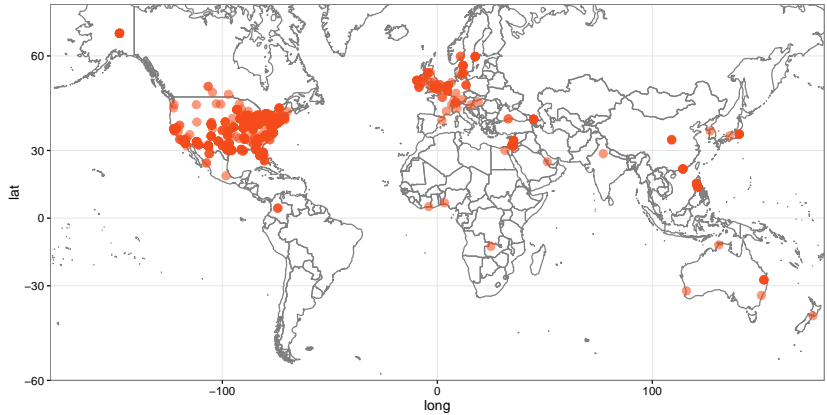
```
  geom_point(aes(x=pos$lon, y=pos$lat) ,  
             color="#F54B1A90", size=3 ,  
             na.rm=T) +
```

```
  theme_bw() +
```

```
  coord_map(xlim=c(-180, 180), ylim=c(-60,70))
```


plot coordinates

map # *ajps* 2013 reviewers worldwide



Extension (2)

- ▶ grab article authors for some years
- ▶ and compare to reviewers