Scraping text from a PDF in R

Emily Burchfield

2017-06-09

Scraping text from a PDF

Finding data can be tricky. Often, the data we want is in a format we do not. In this tutorial, I'll walk you through some techniques you can use to extract data from a PDF and import/clean it up in R.

Here's a link to a dataset that lists the names of agricultural water contract holders in California.[1] This document is a numbered list of contractors' names. Our objective is to scrape this string data off of the PDF and transform it into a nice list of the contractor names (in this case, we don't need the number column) that we can play with in R. I'll be working largely with the awesome `pdftools` package, so make sure you install that first.[2]

[1] If you want to learn more about water contracts, check this page out.

[2] This tutorial is largely based on the tutorial developed by Jeroen Ooms found here.

```
library(pdftools)
```

```
# download the file
download.file("https://www.usbr.gov/mp/cvp-water/docs/ag-contractors-website.pdf",
    ".\\data\\contract.pdf", mode = "wb")

# transform the pdf tile to a bunch of text
txt <- pdf_text(".\\data\\contract.pdf")
```

If you need to index a page in a multi-page PDF, you can simply type `cat(txt[1])`, where `1` is the page number. If you type `class(txt)`, you'll see that it's a `character` object.

```
cat(txt[1])
```

```
##                                          Ag Contractors
##   No.                                 Contractor
##        1        4-E Water District
##        2        4-M Water District
##        3        Alexander, Thomas & Karen
##        4        Anderson, Arthur, et al. (Westfall, Mary)
##        5        Anderson, R & J Property, Inc.
##        6        Anderson-Cottonwood Irrigation District
##        7        Andreotti, Beverly F., et al.
##        8        Arvin-Edison Water Storage District
##        9        Baber, Jack, et al.
##       10        Banta-Carbona Irrigation District
##       11        Bardis, Christo D., et al.
```

```
##      12         Beckley, Ralph & Ophelia
##      13         Bella Vista Water District
##      14         Broadview Water District
##      15         Butler, Les & Minnie
##      16         Butte Creek Farms, Inc.
##      17         Byrd, Ann & Osborne, Jane
##      18         Byron-Bethany Irrigation District
##      19         Cachil Dehe Band of Wintun Indians
##      20         California, State of
##      21         Carter Muncipal Water Company
##      22         Central San Joaquin Water Conservation District
##      23         Chen, Y.
##      24         Chesney, Adona
## Friday, August 04, 2006                                    Page 1 of 9
```

So `pdftools` makes it really easy to pull PDFs off the internet and bring them into R as character objects. Now, character objects are good and all, but what I really want is a `data.frame` so that I can easily extract information from my list of contract owners. Peek at the `txt` object we created. Notice all the `\r\n` stuff going on?[3] This tells essentially tells the computer to make a new line. We can use this and the spaces that distinguish between entries to split up this messy character vector into discrete pieces...

[3] If you really want to get into the depts of text formatting, check out regular expression, or regex and accept that understanding regex makes you a superhero

Splitting and sorting character vectors

When you look at the `txt` object, you'll notice list of spaces between entries. Let's start by using these spaces to split up the long objects into a list. I added `unlist` to group all pages into a single list object called `txt_split`:

```
# first split by the regex line breaks \r\n
txt_split <- unlist(strsplit(txt, "\r\n"))
txt_split <- unlist(strsplit(txt_split, "   "))
print(txt_split[0:20])
```

```
##  [1] ""                "                "
##  [3] ""                "                "
##  [5] ""                "                "
##  [7] ""                "                "
##  [9] ""                "                "
## [11] ""                "                "
## [13] ""                "                "
## [15] " Ag Contractors" "  No."
## [17] ""                "                "
```

```
## [19] ""                ""
```

Notice that there are many empty rows, shown as `""`. Let's drop those from the list.

```
txt_split <- txt_split[txt_split != ""]
print(txt_split[0:20])
```

```
##  [1] " Ag Contractors"
##  [2] "  No."
##  [3] "Contractor"
##  [4] " 1"
##  [5] "  4-E Water District"
##  [6] " 2"
##  [7] "  4-M Water District"
##  [8] " 3"
##  [9] "  Alexander, Thomas & Karen"
## [10] " 4"
## [11] "  Anderson, Arthur, et al. (Westfall, Mary)"
## [12] " 5"
## [13] "  Anderson, R & J Property, Inc."
## [14] " 6"
## [15] "  Anderson-Cottonwood Irrigation District"
## [16] " 7"
## [17] "  Andreotti, Beverly F., et al."
## [18] " 8"
## [19] "  Arvin-Edison Water Storage District"
## [20] " 9"
```

Looking better. I also don't need the columns with numbers, those containing the page number, or the heading that includes `"No."` or `"Contractor"`. The `which` function and `grep` package are good tools to find and alter string patterns. You can be pro and use regular expression or you can simply used character strings, typically by adding the agrument `fixed=T`:

```
# drop exact expressions, like the repeated
# column names using which
txt_sub <- txt_split[which(txt_split != txt_split[2])]  # drops No.

# drop rows containing 'Contractor', 'Page',
# 'August'
txt_sub <- txt_sub[-grep("Contract*", txt_sub)]
txt_sub <- txt_sub[-grep("Friday*", txt_sub)]
txt_sub <- txt_sub[-grep("Page*", txt_sub)]
```

```
# annnnddd some regex to drop numbers
txt_sub <- txt_sub[-grep("^\\s+[0-9]+$", txt_sub)]

# trim the whitespace before and after
# remaining strings
contracts_pdf <- trimws(txt_sub, which = "both")
head(contracts_pdf)
```

```
## [1] "4-E Water District"
## [2] "4-M Water District"
## [3] "Alexander, Thomas & Karen"
## [4] "Anderson, Arthur, et al. (Westfall, Mary)"
## [5] "Anderson, R & J Property, Inc."
## [6] "Anderson-Cottonwood Irrigation District"
```

$*$ is a helpful tool in regex that pretty much means anything else, i.e. `Friday`$*$ indicates strings containing the word `Friday` plus pretty much anything else.

Voila! Our contracts_pdf list is ready to go!