Working with Election Data in R

James Dunham

July 10, 2018

Getting Started

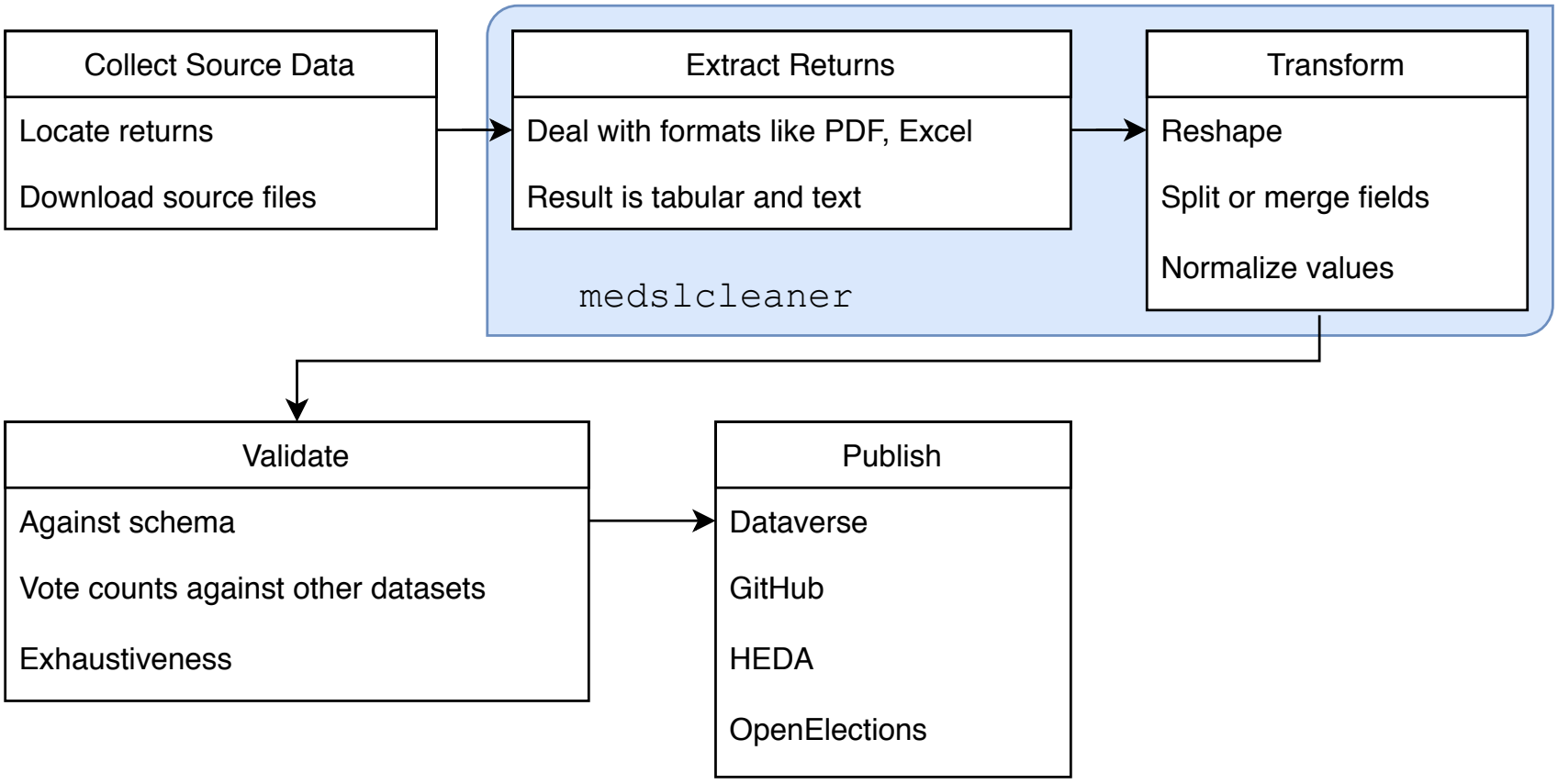
Installation

- Install R from https://cloud.r-project.org
- Also install RStudio, which is an interface for working in R: https://www.rstudio.com/products/rstudio/download

Resources

- *R for Data Science*, especially the section "Wrangle"
- DataCamp's *Introduction to R*
- *An Introduction to Statistical and Data Sciences via R*

Workflow


Collect Source Data
Locate returns
Download source files
Extract Returns
Deal with formats like PDF, Excel
Result is tabular and text
Transform
Reshape
Split or merge fields
Normalize values
medslcleaner
Validate
Against schema
Vote counts against other datasets
Exhaustiveness
Publish
Dataverse
GitHub
HEDA
OpenElections

Toolkit

```
install.packages("tidyverse")
install.packages("tidyxl")
install.packages("devtools")
devtools::install_github('MEDSL/medslcleaner')
```

Source data

	A	B	C	D	E	F	G	H
1		State of New Hampshire - General Election						
2		Merrimack County Offices						
3	November 8, 2016	Sheriff		Attorney		Treasurer		
4		Hilliard, r/d	Scatter	Murray, r/d	Scatter	Hammond, r	Rodriguez, d	Scatter
5	Allenstown	2,035	8	2,013	3	1,296	699	2
6	Andover	1,310	3	1,277		714	564	
7	Boscawen	1,661	10	1,618	13	988	602	2
8	Bow	4,585	13	4,481	12	2,672	1,840	2
9	**TOTALS**	**9,591**	34	**9,389**	28	**5,670**	3,705	6

Multiple headers:

- Row 2: `jurisdiction`
- Row 3: `office`
- Row 4: `candidate`
- Column A: `precinct`

Source data

	A	B	C	D	E	F	G	H
1		State of New Hampshire - General Election						
2		Merrimack County Offices						
3	November 8, 2016	Sheriff		Attorney		Treasurer		
4		Hilliard, r/d	Scatter	Murray, r/d	Scatter	Hammond, r	Rodriguez, d	Scatter
5	Allenstown	2,035	8	2,013	3	1,296	699	2
6	Andover	1,310	3	1,277		714	564	
7	Boscawen	1,661	10	1,618	13	988	602	2
8	Bow	4,585	13	4,481	12	2,672	1,840	2
9	**TOTALS**	**9,591**	34	**9,389**	28	**5,670**	3,705	6

Multiple-column or “merged” cells:

- `Sheriff`
- `Attorney`
- `Treasurer`

Typical approach

```
# required packages
library(tidyverse)
library(medslcleaner)
library(readxl)
library(tidyxl)

# Get the path to the packaged example
merrimack_path = spreadsheet_example('merrimack')

# Use `read_excel` from the `readxl` package
sheet = read_excel(merrimack_path, col_names = FALSE)
```

Result

```
sheet[-c(1:2), 1:5]
```

```
## # A tibble: 7 x 5
##   ..1          ..2              ..3     ..4          ..5
##   <chr>        <chr>            <chr>   <chr>        <chr>
## 1 42682        Sheriff          <NA>    Attorney     <NA>
## 2 <NA>         Hilliard, r/d Scatter Murray, r/d Scatter
## 3 Allenstown 2035               8       2013         3
## 4 Andover    1310               3       1277         <NA>
## 5 Boscawen   1661               10      1618         13
## 6 Bow        4585               13      4481         12
## 7 TOTALS     9591               34      9389         28
```

Alternative approach

Using `medslcleaner` and `tidyxl`,

- Identify which cells are *data* and which are *headers*
- Define the relationships between data cells and header cells

```
# read with tidyxl
cells = xlsx_cells(merrimack_path, sheet = 1)

# take a subset for inspection
peek = cells %>%
  select(address, row, col, character, data_type,
    numeric) %>%
  filter(row %in% 3:5 & col %in% 1:5)
```

Result

```
head(peek, 12)
```

```
## # A tibble: 12 x 6
##    address   row   col character    data_type numeric
##    <chr>   <int> <int> <chr>        <chr>       <dbl>
##  1 A3          3     1 <NA>         date           NA
##  2 B3          3     2 Sheriff      character      NA
##  3 C3          3     3 <NA>         blank          NA
##  4 D3          3     4 Attorney     character      NA
##  5 E3          3     5 <NA>         blank          NA
##  6 A4          4     1 <NA>         blank          NA
##  7 B4          4     2 Hilliard, r/d character      NA
##  8 C4          4     3 Scatter      character      NA
##  9 D4          4     4 Murray, r/d  character      NA
## 10 E4          4     5 Scatter      character      NA
## 11 A5          5     1 Allenstown   character      NA
## 12 B5          5     2 <NA>         numeric      2035
```

Associate headers with cells

```
# cells = cells %>%
#   filter(row > 2) %>%
#
# cells %>%
#   filter(row > 4) %>%
#   arrange(row, col) %>%
#   select(address, row, col, character, numeric, precinct,
```

Associate headers with cells

```
# cells = cells %>%
#   behead('NNW', 'office') %>%
#   behead('N', 'candidate')
#
# cells %>%
#   arrange(row, col) %>%
#   select(address, row, col, character, numeric, precinct
```

Finalize

```
# cells = cells %>%
#   select(address, row, col, precinct, office, candidate,
#
# head(cells)
```

Schema

Field schema define our expectations about data:

```
- name: votes
  title: Vote Count
  description: Number of votes received.
  source: Precinct returns for `jurisdiction`.
  type: integer
  constraints:
    required: true
```

Representation in R

```
data(fields, package = 'medslcleaner')
fields[['votes']]
```

```
## $name
## [1] "votes"
##
## $title
## [1] "Vote Count"
##
## $description
## [1] "Number of votes received."
##
## $source
## [1] "Precinct returns for `jurisdiction`."
##
## $type
## [1] "integer"
##
## $constraints
```

Validation

```
data(wyoming, package = 'medslcleaner')
wyoming %>%
  mutate(precinct = substr(precinct, 1, 10)) %>%
  select(state_postal, jurisdiction, precinct, office, cand
  head()
```

```
##  state_postal jurisdiction  precinct  office  candidate w
## 1           WY     Albany Shields St US House [Write-in]   TRU
## 2           WY     Albany Albany Cou US House [Write-in]   TRU
## 3           WY     Albany Harmony Sc US House [Write-in]   TRU
## 4           WY     Albany Centennial US House [Write-in]   TRU
## 5           WY     Albany Rock River US House [Write-in]   TRU
## 6           WY     Albany Shields St US House [Write-in]   TRU
```

Validation

```
validate(wyoming)
## Validating:
##    year
##    state_postal
##    jurisdiction
##    precinct
##    office
##    district
##    stage
##    special
##    candidate
##    writein
##    party
##    mode
##    votes
##    dataverse
## Success!
```

Validation

```
returns = data.frame(votes = c(2, NA))
returns
```

```
##   votes
## 1     2
## 2    NA
```

```
# validate_field(returns, 'votes')
#> Error: votes has missing values.

select_missing(returns, 'votes')
```

```
## 1/2 rows have missing "votes" values
```

```
##    votes
## 1:    NA
```

Validation

```
# validate(returns)
#> Error: .data does not have name year
```

Further Resources

- medslcleaner github: https://github.com/MEDSL/medslcleaner
- tidyxl documentation: https://nacnudus.github.io/tidyxl/
- *Spreadsheet Munging Strategies*: https://nacnudus.github.io/spreadsheet-munging-strategies