Data errors, how to find them?

Edwin de Jonge, Statistics Netherlands

@edwindjonge | github.com/edwindj

Use of R in Official Statistics 2018
6th international conference

Who am I?

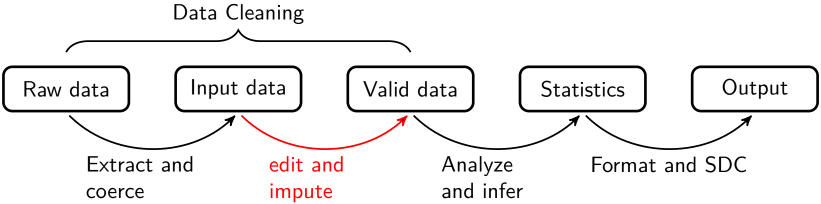
- Data scientist / Methodologist at Statistics Netherlands (aka CBS).
- Author of several R-packages, including `whisker`, `validate`, `errorlocate`, `docopt`, `daff`, `tableplot`, `ffbase`, `chunked`, ...
- Co-author of *Statistical Data Cleaning with applications in R (2018)* (together with @markvdloo)

Data cleaning...

A large part of your job is spent in data-cleaning:

- getting your data in the right shape (e.g. `tidyverse`, `dplyr`)
- assessing missing data (e.g. `VIM`, `datamaid`)
- checking validity (e.g. `validate`)
- locating and removing errors: **`errorlocate`**!
- impute values for missing or erroneous data (e.g. `simputation`, `VIM`, `recipes`)

Statistical Value Chain



KEEP
CALM
AND
VALIDATE

Validation rules?

Package `validate` allows to:

- formulate explicit data rule that data must conform to:

```r
library(validate)
check_that( data.frame(age=160, driver_license=TRUE),
  age >= 0,
  age < 150,
  if (driver_license == TRUE) age >= 16
)
```

Explicit validation rules:

- Give a clear overview what the data must conform to.
- Can be used to reason about.
- Can be used to fix/correct data!
- Find error, and when found correct it.

Note:

- Manual fix is error prone, not reproducible and not feasible for large data sets.
- Large rule set have (very) complex behavior, e.g. entangled rules: adjusting one value may invalidate other rules.

Error localization

Error localization is a procedure that points out fields in a data set that can be altered or imputed in such a way that all validation rules can be satisfied.

Find the error:

```r
library(validate)
check_that( data.frame(age=160, driver_license=TRUE),
  age >= 0,
  age < 150,
  if (driver_license == TRUE) age >= 16
)
```

It is clear that age has an erroneous value, but for more complex rule sets it is less clear.

Multivariate example:

```
check_that( data.frame( age     = 3
                      , married = TRUE
                      , attends = "kindergarten"
                      )
          , if (married == TRUE) age >= 16
          , if (attends == "kindergarten") age <= 6
          )
```

Ok, clear that this is a faulty record, but what is the error?

Feligi Holt formalism:

> *Find the minimal (weighted) number of variables that cause the invalidation of the data rules.*

Makes sense! (But there are exceptions. . .)

Implemented in `errorlocate` (second generation of `editrules`).

Formal description (1)

Rule $r_i(x)$

A rule a disjunction of atomic clauses:

$$r_i(\boldsymbol{x}) = \bigvee_j C_i^j(\boldsymbol{x})$$

with:

$$C_i^j(\boldsymbol{x}) = \begin{cases} \boldsymbol{a}^T\boldsymbol{x} \leq b \\ \boldsymbol{a}^T\boldsymbol{x} = b \\ x_j \in F_{ij}\text{with } F_{ij} \subseteq D_j \\ x_j \not\in F_{ij}\text{with } F_{ij} \subseteq D_j \end{cases}$$

Rule system:

The rules form a system $R(\boldsymbol{x})$:

$$R_H(\boldsymbol{x}) = \bigwedge_i r_i$$

If $R_H(\boldsymbol{x})$ is true for record $\boldsymbol{x}$, then the record is valid, otherwise one (or more) of the rules is violated.

Mixed Integer Programming to FH

Each rule set $R(\boldsymbol{x})$ can be translated into a mip problem and solved.

$$\begin{aligned} \text{Minimize } f(\mathbf{x}) = 0; \\ \text{s.t. } \mathbf{Rx} \leq \mathbf{d} \end{aligned}$$

- $f(\boldsymbol{x})$ is the (weighted) number of changed variable: $\delta_i \in 0, 1$

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \delta_i$$

- $\boldsymbol{R}$ contains rules: $\boldsymbol{R}_H(\boldsymbol{x}) \leq \boldsymbol{d}_H$ and soft constraints: $\boldsymbol{R}_0(\boldsymbol{x}, \boldsymbol{\delta}) \leq \boldsymbol{d}_0$ that try fix the values of $\boldsymbol{x}$ to the measured values.

errorlocate

- translates your rules automatically into a mip form.
- Uses `lpSolveAPI` to solve the problem.
- contains a small framework for implementing your own error localization algorithms.

errorlocate::locate_errors

```
locate_errors( data.frame( age     = 3
                  , married = TRUE
                  , attends = "kindergarten"
                  )
     , validator( if (married == TRUE) age >= 16
                , if (attends == "kindergarten") age <= 6
                )
     )$errors
```

```
##        age married attends
## [1,] FALSE    TRUE   FALSE
```

errorlocate::replace_errors

```
replace_errors(
    data.frame( age     = 3
              , married = TRUE
              , attends = "kindergarten"
              )
  , validator( if (married == TRUE) age >= 16
             , if (attends == "kindergarten") age <= 6
             )
)
```

```
##   age married      attends
## 1   3      NA kindergarten
```

Pipe %>% friendly

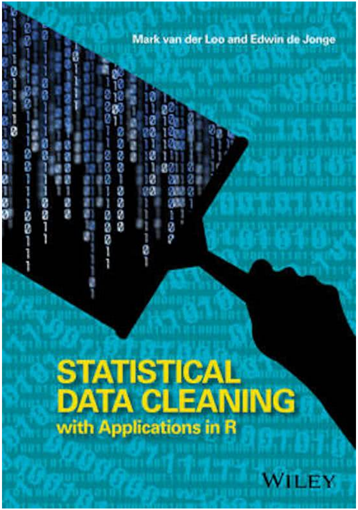
The `replace_errors` function is pipe friendly:

```
rules <- validator(age < 150)

data_noerrors <-
  data.frame(age=160, driver_license = TRUE) %>%
  replace_errors(rules)

errors_removed(data_noerrors) # contains errors removed
```

Interested?


Mark van der Loo and Edwin de Jonge
STATISTICAL DATA CLEANING
with Applications in R
WILEY

SDCR

M. van der Loo and E. de Jonge (2018) *Statistical Data Cleaning with applications in R* Wiley, Inc.

errorlocate

- Available on CRAN

More theory?

← See book

Thank you for your attention (and enjoy The Hague)!

Centraal Bureau voor de Statistiek