Finding and handling data errors

errorlocate

Edwin de Jonge and Mark van der Loo

CBS, Department of Methodology

uRos2019 Tutorial Session, Bucharest

Error localization

Data validation and error localization answer different questions.

Data validation
Which errors are there?

Error localization
Where do I need to make changes to fix the errors?

Example

Ruleset

```
age >= 0
age <= 120
if (drivers_licence == TRUE) age >= 18
```

Data

age	drivers_licence
10	TRUE

Question:

Which field or fields would you change?

Error localization

Definition

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.

Example

Ruleset

```
if (married == TRUE ) age >= 16
if (attends == "kindergarten") age <= 6
```

Data

age	married	attends
3	TRUE	kindergarten

Question

Which field or fields would you change?

Data 2

```
age >= 0,
age < 150,
if (driver_license == TRUE) age >= 16
```

age	driver_license
10	TRUE

Question

Which field or fields would you change?

Assignment

Use `validate` to check the data and find the variable that is incorrect.

Data 2

```
age >= 0,
age < 150,
if (driver_license == TRUE) age >= 16
```

age	driver_license
10	TRUE

Question

Which field or fields would you change?

It depends on the quality of `age` and `driver_license`. We can add more weight to `age` if we think that variable has better quality.

Principle of Fellegi and Holt

Find the minimal (weighted) number of fields to adjust such that all rules, including implied rules, can be satisfied.

IP Fellegi and D Holt, JASA **71** 353 17–35 (1976).

Note

This should be used as a last resort, when no further information on the location of errors is available.

Feligi Holt (FH) formalism:

But there are exceptions...

- In balance sheets, swapping variables (2 edits) sometimes makes more sense then adjusting one value (1 edit). (see R package:`deducorrect`).
- In some data, spreading a surplus or shortage on a variable over many variables is sensible. (see R package: `rspa`).

Implied rules?

```
turnover - total.cost == profit
           profit <= 0.6 * turnover
```

This implies (substituting profit):

```
           total.cost >= 0.4 * turnover
```

We need to take into account such *essentially new* rules (edits) —unstated relations between variables that can be derived from the explicitly defined rules.

errorlocate

- R-package that implements FH.
- Is extensible (you can plug in your own detection stuff)
- provides:
  - `locate_errors`
  - `replace_errors`
  - R5 classes to add your own stuff.

errorlocate::locate_errors

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

```
## call:  x$locate(data = data, weight = weight, ...)
## located  1  error(s).
## located  0  missing value(s).
## Use 'summary', 'values', '$errors' or '$weight', to explore and retrieve
```

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
```

Assignment (small examples)

a) Find the error in this record with `locate_errors`:

age	married	attends
26	TRUE	kindergarten

b) Find the error with `locate_errors`:

age	married	attends
15	TRUE	kindergarten

c) You have more confidence in the kindergarden variable: apply a weight of 3 to age in finding the errors.

Removing errors

- Detecting errors is very useful, but then what?
- Fixing philosophy is:
  - Find erroneuous values.
  - Remove them (i.e. make them `NA`).
  - Impute them with sensible values.

Note

We could also remove erroneous records completely, but often this result in *over-deletion* and introduces a *bias*.

errorlocate::replace_errors

- Locates errors and replaces them with NA.

```
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
```

Assignment

a) Use the data set `retailers` from package `validate`.
b) Use validate to find out which records are `faulty` using the rule set

```
rules <- validator(
  to_pos = turnover >= 0
  , or_pos = other.rev >= 0
  , balance = turnover + other.rev == total.rev)
```

c) use `locate_errors` to find some errors.
d) use `replace_errors` to "fix" the data set.

```r
data(retailers, package="validate")
retailers <- retailers[c("other.rev", "total.rev", "turnover")]
rules <- validator(
  to_pos = turnover >= 0
  , or_pos = other.rev >= 0
  , balance = turnover + other.rev == total.rev)
confront(retailers, rules)
```

```
## Object of class 'validation'
## Call:
##     confront(dat = retailers, x = rules)
##
## Confrontations: 3
## With fails    : 2
## Warnings      : 0
## Errors        : 0
```

```r
errors <- locate_errors(retailers, rules)$errors
row_contains_error <- apply(errors, 1, any)
w <- which(row_contains_error)
```

```
retailers[w,c("other.rev", "total.rev", "turnover")]
```

```
##    other.rev total.rev turnover
## 3        -33      6919     6886
## 30      1831      1831     1831
## 32        NA       107      971
## 36     98350      2747     2649
## 37         4       206     1024
```

```
replace_errors(retailers[w,], rules)
```

```
##    other.rev total.rev turnover
## 3         NA      6919     6886
## 30      1831        NA     1831
## 32        NA       107       NA
## 36     98350        NA     2649
## 37         4        NA     1024
```

Internal workings:

`errorlocate`:

- translates error localization problem into a **mixed integer problem**, which is solved with `lpsolveAPI`.
- contains a small framework for implementing your own error localization algorithms.

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
```