The statistical value chain and data validation

Mark van der Loo and Edwin de Jonge

CBS, Department of Methodology

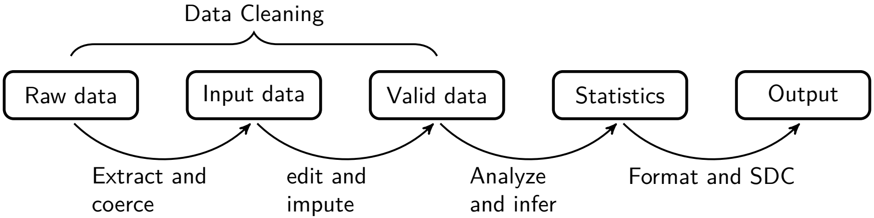
uRos2019 Tutorial Session, The Hague

This tutorial

https://github.com/data-cleaning/uRos2019_tutorial

The Statistical Value Chain

Statistical Value Chain



Notes

- This part only pertains to the data processing stage. Collection, design, dissemination is not included.
- The fixed points are well-defined statistical products.

The SVC: Remarks

- Actual data processing is not necessarily linear accross the chain
- In production architectures a more flexible model is often used where the definition of interfaces between processing steps play a crucial role. The chain shown here is a general example covering most steps in some way.

Data validation

Definition (ESS handbook on validation)

Data validation is an activity in which it is verified whether or not a combination of values is a member of a set of acceptable value combinations.

Validation rules

The set of acceptable values combinations are defined by *validation rules*, e.g. `IF age <= 14 THEN has_job == "no"`.

Observe

validation rules define, to large extend, the products in the SVC

`validate`: *data validation infrastructure for R*

A domain-specific language for rule definition

Define *any* check on your data, using the *full power* of the R language.

Rules as first-class citizens

- CRUD operations (create, read, update, delete)
- Summarize, plot, investigate rules
- Rich metadata

Validate data

- Confront data with rules
- CRUD on results, summarize, plot
- Export to ESS standard reporting format (upcoming)

Assignment 1

Try the following code.

```r
library(validate)
library(magrittr)
data(retailers)
head(retailers)
retailers %>%
  check_that(turnover + other.rev == total.rev
             , turnover > 0, other.rev > 0 ) %>%
  summary()
```

Assignment 1

```
library(validate)
library(magrittr)
data(retailers)
retailers %>%
  check_that(turnover + other.rev == total.rev
             , turnover > 0, other.rev > 0 ) %>%
  summary()
```

```
##   name items passes fails nNA error warning
## 1   V1    60     19     4  37 FALSE   FALSE
## 2   V2    60     56     0   4 FALSE   FALSE
## 3   V3    60     23     1  36 FALSE   FALSE
##                                       expression
## 1 abs(turnover + other.rev - total.rev) < 1e-08
## 2                                  turnover > 0
## 3                                 other.rev > 0
```

Data validation with `validate`

```r
library(validate)
data(retailers)
head(retailers,3)[3:7]
```

```
##   staff turnover other.rev total.rev staff.costs
## 1    75       NA        NA      1130          NA
## 2     9     1607        NA      1607         131
## 3    NA     6886       -33      6919         324
```

Data validation with `validate`

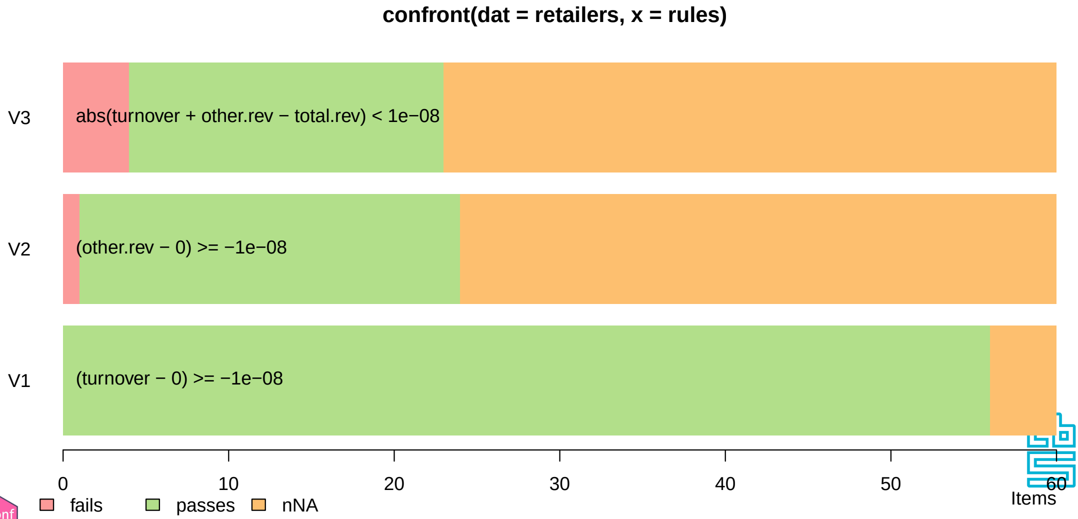
```r
rules <- validator(
   turnover >= 0
  , other.rev >= 0
  , turnover + other.rev == total.rev
)

out <- confront(retailers, rules)
summary(out)
```

Assignment 2

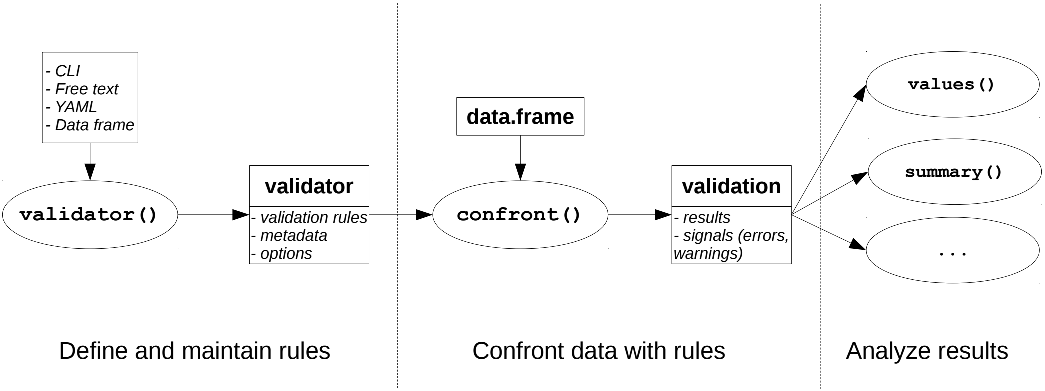
1. Adapt the previous exercise so you use `validator`.
2. Use `confront` for validation and store the results in a variable called `out`.
3. Try `plot(out)`.
4. Try `as.data.frame(out)` (use `View` to inspect the result)

Plotting output

```r
plot(out)
```



The validate package



Reading rules from file

```
### myrulez.txt

# some basic checks
staff >= 0
turnover >= 0
other.rev >= 0
# account balance checks
turnover + other.rev == total.rev
# other commom sense stuff
if (staff >= 1) staff.costs >= 1
```

```
rulez <- validator(.file="myrulez.txt")
```

Assignment 3

1. Create a new textfile
2. Define 10 rules for the `retailers` dataset
3. Read the rules (`validator(.file="your file")`)
4. `confront` rules with data
5. Summarize and plot the results.
6. Use `as.data.frame` and `View` to convert and display the results.
7. Make a `plot` of the `validator` object.

uRosConf 2019

A few extra's (if we have time)

Domain Specific Language

Validation *Domain Specific Language* (DSL)

Any R statement resulting in a `logical`.

Examples

```
# Range checks
has_job %in% c('yes','no')
turnover >= 0
# Multivariate checks
abs(profit) <= 0.6 * turnover
# Multi-row checks
mean(profit) > 10
# Logical implications
if (staff > 0) staff.costs > 0
```

Validation DSL

Comparisons

`>, >=,==, <=, <, %in%`

Boolean operations

`!, all(), any(), &, &&, |, ||, if () else`

Text search

`grepl`

Functional dependencies (Armstrong)

`city + zipcode ~ streetname`

Refer to the dataset with `.`

`nrow(.) == 40, "turnover" %in% names(.)`

uRosConf 2019

Transient assignments (macros) using :=

Example 1

$$\max\left(\frac{x}{x^*}, \frac{x^*}{x}\right) \leq 10$$

```
med := median(turnover,na.rm=TRUE)
hb := pmax(turnover/med, med/turnover, na.rm=TRUE)
hb <= 10
```

Example 2

```
beta_2 := coefficients(lm(turnover ~ profit))[2]
beta_2 >= 0
```

Variable groups

Many variables, same rule

```
G := var_group(staff, turnover, other.rev, total.costs)
G >= 0
```

Error handling

```r
out <- check_that(women, hite > 0, weight>0)
out
```

```
## Object of class 'validation'
## Call:
##     check_that(women, hite > 0, weight > 0)
##
## Confrontations: 2
## With fails    : 0
## Warnings      : 0
## Errors        : 1
```

```r
errors(out)
```

```
## $V1
## [1] "object 'hite' not found"
```