Package 'lodi'

July 22, 2025

Type Package

Title Limit of Detection Imputation for Single-Pollutant Models

Version 0.9.2

URL https://github.com/umich-cphds/lodi

BugReports https://github.com/umich-cphds/lodi/issues

Description Impute observed values below the limit of detection (LOD) via censored likelihood multiple imputation (CLMI) in single-pollutant models, developed by Boss et al (2019) <doi:10.1097/EDE.0000000000001052>. CLMI handles exposure detection limits that may change throughout the course of exposure assessment. 'lodi' provides functions for imputing and pooling for this method.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.1.0)

Imports stats, rlang (>= 0.3.0)

RoxygenNote 6.1.1

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Jonathan Boss [aut],
Alexander Rix [aut, cre]

Maintainer Alexander Rix <alexrix@umich.edu>

Repository CRAN

Date/Publication 2020-02-07 15:30:02 UTC

Contents

clmi 2
lod_cca 3
lod_root2 4
pool.clmi 5
toy_data 6

Index **7**

`clmi` *Censored Likelihood Multiple Imputation*

Description

This function performs censored likelihood multiple imputation for single-pollutant models where the pollutant of interest is subject to varying detection limits across batches (this function will also work if there is only one distinct detection limit). The function outputs a list containing the imputed datasets and details regarding the imputation procedure (i.e., number of imputed dataset, covariates used to impute the non-detects, etc).

Usage

```
clmi(formula, df, lod, seed, n.imps = 5, verbose = FALSE)
```

Arguments

`formula`	A formula in the form of `exposure ~ outcome + covariates`. That is, the first variable on the right hand side of `formula` should be the outcome of interest.
`df`	A data.frame with `exposure`, `outcome` and `covariates`.
`lod`	Name of limit of detection variable in `df`.
`seed`	For reproducability.
`n.imps`	Number of datasets to impute. Default is 5.
`verbose`	If `TRUE`, `clmi` prints out useful debugging information while running. Default is `FALSE`.

Details

`clmi` is somewhat picky regarding the `formula` parameter. It tries to infer what transformation you'd like to apply to the exposure you are imputing, what the exposure is, and what the outcome is. It attempts to check to make sure that everything is working correctly, but it can fail. Roughly, the rules are:

- The left hand side of formula should be the exposure you are trying to impute.
- The exposure may be optionally wrapped in a univariate transformation function. If the transformation function is not univariate, you ought to get an error about a "complicated" transformation.
- The first variable on the right hand side of `formula` should be your outcome of interest.

Note

- `clmi` only supports categorical variables that are numeric, (i.e., not factors or characters). You can use the `model.matrix` function to convert a data frame with factors to a numeric design matrix and subsequently convert that matrix back into a data frame using `as.data.frame`.
- If you get the error message "L-BFGS-B needs finite values of 'fn'", try normalising your data.

References

Boss J, Mukherjee B, Ferguson KK, et al. Estimating outcome-exposure associations when exposure biomarker detection limits vary across batches. Epidemiology. 2019;30(5):746-755. 10.1097/EDE.0000000000001052

Examples

```
library(lodi)

# Note that the outcome of interest is the first variable on the right hand
# side of the formula.
clmi.out <- clmi(poll ~ case_cntrl + smoking + gender, toy_data, lod, 1)

# you can specify a transformation to the exposure in the formula
clmi.out <- clmi(log(poll) ~ case_cntrl + smoking + gender, toy_data, lod, 1)
```

---

lod_cca *Single pollutant complete case analysis.*

Description

lod_cca is a helper function that does complete case analysis for single pollutant models. The function can be used to compare with clmi.

Usage

```
lod_cca(formula, df, type)
```

Arguments

`formula`	A R formula in the form outcome ~ exposure + covariates.
`df`	A data.frame that contains the variables `formula` references.
`type`	The type of regression to perform. Acceptable options are linear and logistic.

Examples

```
library(lodi)
# load lodi's toy data
data("toy_data")
x <- lod_cca(case_cntrl ~ poll + smoking + gender, toy_data, logistic)
# see the fit model
x$model
```

`lod_root2` *Single pollutant* `sqrt(2)` *imputation.*

Description

lod_root2 is a helper function that performs single imputation with `lod / sqrt(2)`, a common ad hoc approach used in single-pollutant modeling. The function can be used to compare with clmi.

Usage

```
lod_root2(formula, df, lod, type)
```

Arguments

`formula`	A R formula in the form `outcome ~ exposure + covariates`.
`df`	A data.frame that contains the variables `formula` references.
`lod`	Name of the limit of detection variable.
`type`	The type of regression to perform. Acceptable options are linear and logistic.

Note

Depending on the transformation used, a "Complicated transformation" error may occur. For example, the transformation `a * exposure` will cause an error. In this case, define a transformation function as `f <- function(exposure) a * exposure` and use `f` in your formula. This technical limitation is unavoidable at the moment.

Examples

```
# load lodi's toy data
library(lodi)
data("toy_data")
lodi.out <- lod_root2(case_cntrl ~ poll + smoking + gender, toy_data, lod,
                      logistic)
# see the fit model
lodi.out$model

# we can log transform poll to make it normally distributed
lodi.out <- lod_root2(case_cntrl ~ log(poll) + smoking + gender, toy_data,
                      lod, logistic)
lodi.out$model
```

```
# transforming the exposure results in a new column being added to data,
# representing the transformed lod.
head(lodi.out$data)

# You can even define your own transformation functions and use them
f <- function(x) exp(sqrt(x))
lodi.out <- lod_root2(case_cntrl ~ f(poll) + smoking + gender, toy_data, lod,
                        logistic)
head(lodi.out$data)
```

`pool.clmi` *Calculate pooled estimates from* `clmi.out` *objects using Rubin's rules*

Description

Calculate pooled estimates from `clmi.out` objects using Rubin's rules

Usage

```
pool.clmi(formula, clmi.out, type)
```

Arguments

`formula`	Formula to fit. Exposure variable should end in `_transform_imputed`.
`clmi.out`	An object generated by clmi.
`type`	Type of regression to pool. Valid types are logistic and linear.

Examples

```
# continue example from clmi
# fit model on imputed data and pool results
library(lodi)
data("toy_data")
clmi.out <- clmi(log(poll) ~ case_cntrl + smoking + gender, toy_data, lod, 1)
results <- pool.clmi(case_cntrl ~ poll_transform_imputed + smoking, clmi.out,
                       logistic)

results$output
```

`toy_data` *Synthetic toy data for clmi*

Description

Synthetic toy data for clmi

Usage

```
toy_data
```

Format

A data.frame with 100 observations on 6 variables:

id Patient ID number.

case_cntrl Patient's case-control status. Either 1 or 0.

poll Concentration of pollutant in patient's blood sample.

smoking Smoking status. Either 1 or 0.

gender Gender. 1 for male, 0 for female.

batch1 Batch status. Integer

lod batch's limit of detection for patient.

Index

∗ **datasets**
toy_data, 6

clmi, 2

lod_cca, 3
lod_root2, 4

pool.clmi, 5

toy_data, 6