Package ‘InformativeCensoring’

June 5, 2023

Type Package

Title Multiple Imputation for Informative Censoring

Version 0.3.6

Maintainer Jonathan Bartlett <jonathan.bartlett1@lshtm.ac.uk>

Author David Ruau [aut],
Nikolas Burkoff [aut],
Jonathan Bartlett [aut, cre],
Dan Jackson [aut],
Edmund Jones [aut],
Martin Law [aut],
Paul Metcalfe [aut]

Description Multiple Imputation for Informative Censoring.
This package implements two methods. Gamma Imputation
described in <DOI:10.1002/sim.6274> and Risk Score Imputation
described in <DOI:10.1002/sim.3480>.

License GPL (>= 2) | file LICENSE

LazyLoad yes

Depends R (>= 3.1.2), survival (>= 2.36-1)
Imports boot, dplyr (>= 0.4.3), parallel
Suggests knitr, testthat

VignetteBuilder knitr

URL https://github.com/jwb133/InformativeCensoring

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2023-06-05 14:50:02 UTC
Perform methods of multiple imputation for time to event data

Details


Author(s)

<David.Ruau@astrazeneca.com>
**col.headings**

Specify the columns of the data frame required by score imputation method

**Description**

Specify the columns of the data frame required by score imputation method

**Usage**

```r
col.headings(arm, has.event, time, Id, DCO.time, to.impute, censor.type = NULL)
```

**Arguments**

- `arm`: column name which will contain the subject’s treatment group
- `has.event`: column name which will contain whether the subject has an event (1) or not(0)
- `time`: column name of censoring/event time
- `Id`: column name of subject Id
- `DCO.time`: column name of the time at which the subject would have been censored had they not had an event before data cut off
- `to.impute`: column name of the logical column as to whether events should be imputed
- `censor.type`: column name of the column containing the reason for censoring, 0=had event, 1=non-administrative censoring 2=administrative censoring – only subjects with 1 in this column count as having an ‘event’ in the Cox model for censoring (optionally used – if not used then all subjects who are censored are used)

**Value**

A list contain the given arguments

---

**cox.zph**

Test Cox proportional hazards assumption

**Description**

See `cox.zph` function in the survival package

**Usage**

```r
cox.zph(fit, transform = "km", global = TRUE, ...)
```
Arguments

- **fit**: the result of fitting a Cox regression model, using the coxph or coxme functions.
- **transform**: a character string specifying how the survival times should be transformed before the test is performed. Possible values are "km", "rank", "identity" or a function of one argument.
- **global**: should a global chi-square test be done, in addition to the per-variable or per-term tests.
- **...**: Additional arguments to cox.zph, for example index if fit is a GammaStatList object

See Also

- cox.zph

---

**ExtractSingle**

*Extract a single risk score/gamma imputed data set/model fit*

Description

Extract a single risk score/gamma imputed data set/model fit

Usage

```r
## S3 method for class 'GammaImputedSet'
ExtractSingle(x, index)

## S3 method for class 'GammaStatList'
ExtractSingle(x, index)

## S3 method for class 'ScoreImputedSet'
ExtractSingle(x, index)

## S3 method for class 'ScoreStatList'
ExtractSingle(x, index)
```

Arguments

- **x**: The multiple imputed object
- **index**: Integer, which imputed data set/model fit should be returned

Value

The individual data set/model fit
**Description**

This function performs the Imputation described in Relaxing the independent censoring assumptions in the Cox proportional hazards model using multiple imputation. (2014) D. Jackson et al. Statist. Med. (33) 4681-4694

**Usage**

```r
gammaImpute(
  formula,
  data,
  m,
  gamma,
  gamma.factor,
  bootstrap.strata = rep(1, nrow(data)),
  DCO.time,
  ...,
  parallel = c("no", "multicore", "snow")[1],
  ncpus = 1L,
  cl = NULL
)
```

**Arguments**

- `formula`: The model formula to be used when fitting the models to calculate the cumulative hazard. A formula for coxph can include strata terms but not cluster or tt and only right-censored Surv objects can be used. Note the function does not allow multiple strata to be written as `strata(W1)+strata(W2)`, use `strata(W1,W2)` instead.

- `data`: A time to event data set for which event times are to be imputed.

- `m`: The number of imputations to be created.

- `gamma`: Either column name containing the value of gamma or a vector of values giving the subject specific size of the step change in the log hazard at censoring. If a subject has NA in this column then no imputation is performed for this subject (i.e. the subject's censored time remains unchanged after imputation). If a subject has already had an event then the value of gamma is ignored. If `gamma.factor` is also used then the subject specific gamma are all multiplied by `gamma.factor`. At least one of `gamma` and `gamma.factor` must be included.

- `gamma.factor`: If used, a single numeric value. If no `gamma` then the step change in log hazard for all subjects at censoring is given by `gamma.factor`. If `gamma` is used then for each subject, the step change in log hazard is given by `gamma.factor` multiplied by the subject specific gamma. At least one of `gamma` and `gamma.factor` must be included.
bootstrap.strata
   The strata argument for stratified bootstrap sampling, see argument strata for
   the function boot::boot for further details. If argument is not used then stan-
   dard sampling with replacement will be used

DCO.time
   Either column name containing the subject’s data cutoff time or a vector of
   DCO.times for the subjects or a single number to be used as the DCO.time for
   all subjects (if imputed events are > this DCO.time then subjects are censored at
   DCO.time in imputed data sets)

... Additional parameters to be passed into the model fit function

parallel
   The type of parallel operation to be used (if any).
ncpus integer: number of processes to be used in parallel operation: typically one
   would chose this to be the number of available CPUs
cl An optional parallel or snow cluster for use if parallel="snow". If not sup-
   plied, a cluster on the local machine is created for the duration of the call.

Details
   See the Gamma Imputation vignette for further details

Value
   A GammaImputedSet.object containing the imputed data sets

See Also

   GammaImputedSet.object GammaImputedData.object

Examples

## Not run:
data(nwtco)
nwtco <- nwtco[1:500,]

#creating 2 imputed data sets (m=2) for speed, would normally create more
ans <- gammaImpute(formula=Surv(edrel,rel)~histol + instit,
   data = nwtco, m=2, gamma.factor=1, DCO.time=6209)

#subject specific gamma (multiplied by gamma.factor to give the jump)
#NA for subjects that are not to be imputed
jumps <- c(rep(NA,10),rep(1,490))
DCO.values <- rep(6209,500)

ans.2 <- gammaImpute(formula=Surv(edrel,rel)~histol + instit + strata(stage),
   data = nwtco, m=2, bootstrap.strata=strata(nwtco$stage),
   gamma=jumps, gamma.factor=1, DCO.time=DCO.values)

#can also use column names
nwtco.gamma <- jumps
nwtco$DCO.time <- DCO.values
ans.3 <- gammaImpute(formula=Surv(edrel,rel)~histol + instit + strata(stage),
                     data = nwtco, m=2, bootstrap.strata=strata(nwtco$stage),
                     gamma="gamma", DCO.time="DCO.time")

## End(Not run)

---

### GammaImputedData.object

**GammaImputedData object**

**Description**

An object which contains

**Slots**

- data: A data frame containing the time to event data with 3 new columns impute.time and impute.event, the imputed event/censoring times and event indicators (for subjects whose data is not imputed these columns contain the unchanged event/censoring time and event indicator) and internal_gamma_val which is the value of gamma used for each subject in this data set
- default.formula: The default model formula which will be used when fitting the imputed data

---

### GammaImputedSet.object

**GammaImputedSet object**

**Description**

An object which contains the set of gamma imputed data frames. Use the ExtractSingle function to extract a single GammaImputedData objects. Use the ImputeStat function to fit models to the entire set of imputed data frames

**Details**

It contains the following:

**Slots**

- data: A data frame containing the unimputed time to event data (along with a column internal_gamma_val which is the value of gamma used for each subject in this data set)
- m: The number of imputed data sets
- gamma.factor: The value of gamma.factor used with the imputation
- impute.time: A matrix (1 column per imputed data set) containing the imputed times
- impute.event: A matrix (1 column per imputed data set) containing the imputed event indicators
- default.formula: The default model formula which will be used when fitting the imputed data
See Also

GammaImputedData.object

---

**Description**

An S3 object which contains the point estimate and test statistic after fitting a model to a GammaImputedData object.

**Details**

The function `print.GammaStat` has been implemented

The object contains the following:

**Slots**

- `model` The model used to create the fit
- `method` The model used for the fit
- `estimate` A point estimate of the test statistic
- `var` The estimate for the variance of the test statistic

---

**Description**

The object containing the results of fitting models to a GammaImputedSet object.

**Details**

A `summary.GammaStatList` has been implemented which performs Rubin’s multiple imputation rules.

The object contains the following:

**Slots**

- `fits` A list of GammaStat objects containing the model fits for the imputed data sets
- `statistics` A list with two elements: estimates and vars which contain the coefficient estimates and their variances one column per covariate one row per imputed data set
- `m` The number of model fits
ImputeStat

S3 generic to fit model(s) to risk score/gamma Imputed objects

Description

S3 generic to fit model(s) to risk score/gamma Imputed objects

Usage

## S3 method for class 'GammaImputedData'
ImputeStat(
  object,
  method = c("Cox", "weibull", "exponential")[1],
  formula = NULL,
  ...,
)

## S3 method for class 'GammaImputedSet'
ImputeStat(
  object,
  method = c("Cox", "weibull", "exponential")[1],
  formula = NULL,
  ...
  parallel = c("no", "multicore", "snow")[1],
  ncpus = 1L,
  cl = NULL
)

ImputeStat(
  object,
  method = c("logrank", "Wilcoxon", "Cox", "weibull", "exponential")[1],
  formula = NULL,
  ...
)

## S3 method for class 'ScoreImputedSet'
ImputeStat(
  object,
  method = c("logrank", "Wilcoxon", "Cox")[1],
  formula = NULL,
  ...
  parallel = c("no", "multicore", "snow")[1],
  ncpus = 1L,
  cl = NULL
)
Arguments

object: A ScoreImputedData, ScoreImputedSet, GammaImputedData or GammaImputedSet object to fit the model to.

method: The type of statistical model to fit. There are three methods which can be performed when using Risk Score imputation:
   "logrank": a logrank test using survival::survdiff
   "Wilcoxon": Peto & Peto modification of the Gehan-Wilcoxon test using survival::survdiff with rho=1
   "Cox": Fit a cox model using survival::coxph

For gamma imputation the model can be "Cox" (using survival::coxph), "weibull" or "exponential" both using survival::coxph.

formula: The model formula to fit. If no formula argument is used, then object$default.formula will be used. For risk score imputation this is ~ treatment.group and for gamma imputation this is the formula used when fitting the Cox model. For method="Cox", additional covariates can be included by explicitly giving a formula argument. For logrank/Wilcoxon only additional strata terms can be included.

In all cases only the right hand side of the formula is required. The survival object on the left hand side is created automatically. E.g. for a Cox model could use formula=~arm + covar1. The cluster and tt options cannot be used. See the vignettes for further details.

... Additional arguments which are passed into the model fit function.

parallel: The type of parallel operation to be used (if any), can be used for GammaImputedSet and ScoreImputedSet.

ncpus: integer: number of processes to be used in parallel operation: typically one would chose this to be the number of available CPUs, can be used for GammaImputedSet and ScoreImputedSet.

c1: An optional parallel or snow cluster for use if parallel="snow". If not supplied, a cluster on the local machine is created for the duration of the call, can be used for GammaImputedSet and ScoreImputedSet.

See Also

ScoreStat.object ScoreImputedData.object

MakeTimeDepScore Create a valid ScoreTD object

Description

Create a valid ScoreTD object
NN.options

Usage

MakeTimeDepScore(data, Id, time.start, time.end)

Arguments

data A data frame of time dependent covariates
Id The column name of the subject Id
time.start The covariates are valid for the time [time.start, time.end] where time.start is the
column name of time.start
time.end The covariates are valid for the time [time.start, time.end] where time.end is the
column name of time.end

Value

A ScoreTD object

NN.options

Create a list of options which control the nearest neighbour algorithm
for risk score imputation

Description

Create a list of options which control the nearest neighbour algorithm for risk score imputation

Usage

NN.options(NN = 5, w.censoring = 0.2, min.subjects = 20)

Arguments

NN The (maximum) number of subjects to be included in the risk set
w.censoring The weighting on the censoring risk score when calculating distances for the
nearest neighbour calculation A weighting of (1-w.censoring) is used for the
event risk score
min.subjects If using time dependent score imputation include at least this number of subjects
when fitting the Cox model (i.e. include some subjects who were censored/had
event earlier than the censored observation if neccessary)

Value

A list of options used within the ScoreImputedData function
ScoreImpute

Perform risk score multiple imputation method

Description

Perform risk score multiple imputation method

Usage

ScoreImpute(
  data,
  event.model,
  censor.model = event.model,
  col.control,
  NN.control = NN.options(),
  time.dep = NULL,
  m,
  bootstrap.strata = rep(1, nrow(data)),
  ...,  
  parallel = c("no", "multicore", "snow")[1],
  ncpus = 1L,
  cl = NULL
)

Arguments

data The data set for which imputation is required

event.model The right hand side of the formula to be used for fitting the Cox model for calculating the time to event score e.g. ~Z1+Z2+Z3.

censor.model The right hand side of the formula to be used for fitting the Cox model for calculating the time to censoring score if not included then event.model will be used

col.control A list of the columns names of data which are used by the imputation algorithm See example below and for further details of these columns and their purpose see col.headings

NN.control Parameters which control the nearest neighbour algorithm. See NN.options

time.dep A ScoreTD object, to be included if the time dependent score imputation method is to be used, otherwise it should be NULL

m The number of data sets to impute

bootstrap.strata When performing the bootstrap procedure for fitting the models, how should the data be stratified (see strata argument to boot::boot). if argument is not used then standard sampling with replacement is used to generate the bootstrap data

... Additional arguments passed into the Cox model Note the subset and na.action arguments should not be used (na.fail will be used when fitting the Cox model)
parallel  The type of parallel operation to be used (if any).
ncpus integer: number of processes to be used in parallel operation: typically one would chose this to be the number of available CPUs
cl An optional parallel or snow cluster for use if parallel="snow". If not supplied, a cluster on the local machine is created for the duration of the call.

Details
Note that coxph may fail to converge and the following output Warning in fitter(X, Y, strats, offset, init, control, weights = weights, : Ran out of iterations and did not converge
It is possible to use ridge regression by including a ridge term in the model formula (e.g. ~Z1+ridge(Z2, theta=1)). See ridge for further details

Value
A ScoreImputedSet object

See Also
ScoreImputedSet.object

Examples

data(ScoreInd)

col.control <- col.headings(has.event="event", time="time",
Id="Id",arm="arm",
DCO.time="DCO.time",
to.impute="to.impute")

## Not run:
an <- ScoreImpute(data=ScoreInd,event.model=~Z1+Z2+Z3+Z4+Z5,
col.control=col.control, m=5,
bootstrap.strata=ScoreInd$arm,
NN.control=NN.options(NN=5,w.censoring = 0.2))

## End(Not run)

---

ScoreImputedData.object

ScoreImputedData object

Description
An object which contains
Slots

data A data frame containing the time to event data with 2 new columns impute.time and impute.event, the imputed event/censoring times and event indicators (for subjects whose data is not imputed these columns contain the unchanged event/censoring time and event indicator)

col.control The list of column names the risk score imputation method requires see col.headings for further details. If censor.type was not used then col.control$censor.type="using_has.event_col"

default.formula The default model formula which will be used when fitting the imputed data using a Cox model

ScoreImputedSet.object

ScoreImputedSet object

Description

An object which contains the set of score imputed data frames. Use the ExtractSingle function to extract a single ScoreImputedData object. Use the ScoreStat function to fit models to the entire set of imputed data frames

Details

It contains the following:

Slots

data A data frame containing the unimputed time to event data

col.control The list of column names the score imputation method requires see col.headings for further details

m The number of imputed data sets

impute.time A matrix (1 column per imputed data set) containing the imputed times

impute.event A matrix (1 column per imputed data set) containing the imputed event indicators

default.formula The default model formula which will be used when fitting the imputed data using a Cox model

See Also

ScoreImputedData.object
ScoreInd

Simulated time to event data with 5 time independent covariates

Description

This dataset is inspired by the simulation described in Hsu and Taylor, Statistics in Medicine (2009) 28:462-475 with an additional DCO.time column.

Format

A data.frame containing a row per subject with eleven columns:

Fields

- **Id**: subject identifier
- **arm**: factor for treatment group control=0, active=1
- **Z1**: binary time independent covariate
- **Z2**: continuous time independent covariate
- **Z3**: binary time independent covariate
- **Z4**: continuous time independent covariate
- **Z5**: binary time independent covariate
- **event**: event indicator (1 yes, 0 no)
- **time**: subject censoring/event time (in years)
- **to.impute**: logical, should an event time be imputed for this subject? (this is ignored if subject has event time)
- **DCO.time**: The time the subject would have been censored if they had not had an event or been censored before the data cut off date

ScoreStat.object

ScoreStat object

Description

An S3 object which contains the point estimate and test statistic after fitting a model to a ScoreImputedData object.

Details

The functions print.ScoreStat and as.vector.ScoreStat have been included.

The object contains the following:

The test statistic should be normally distributed and hence for the logrank test \( Z = \frac{(O_2 - E_2)}{\sqrt{V_2}} \), i.e. the square root of the standard Chi squared statistic (with the appropriate sign)
Slots

- `model`  The model used to create the fit
- `method`  The method used for the fit
- `estimate`  A point estimate of the test statistic
- `var`  The estimate for the variance of the test statistic
- `statistic`  The test statistic given by `estimate/sqrt(var)`

Description

The object containing the results of fitting models to a `ScoreImputedSet` object.

Details

A summary `ScoreStatList` has been implemented.

The object contains the following

Slots

- `fits`  A list of `ScoreStat` objects containing the model fits for the imputed data sets
- `statistics`  A `ScoreStatSet` object containing the statistics
- `m`  The number of model fits

See Also

- `ScoreStatList.object`  `ScoreStatList`
- `ScoreStatSet.object`  `ScoreStat.object`

Description

S3 generic to create a `ScoreStatSet` object

Usage

`ScoreStatSet(x)`

Arguments

- `x`  The object to convert into a `ScoreStatSet` object
**Value**

A ScoreStatSet object

**See Also**

ScoreStatSet.object

---

**ScoreStatSet.object**

An object which contains the test statistic and estimators for a set of model fits to imputed data using risk score imputation

---

**Description**

The object is a Mx3 matrix, one row per imputed data set and columns: estimate (the point estimates), var (their variances) and Z (the test statistic). M must be > 4

**Details**

Note the Z should be ~ standard normal (so we do not use the chi_squared test statistic see ScoreStat.object for further details)

The summary.ScoreStatSet function will apply the MI averaging procedures and estimates of the test statistic and p-value

**See Also**

summary.ScoreStatSet

---

**ScoreTD.object**

A ScoreTD object

---

**Description**

This data frame holds time dependent covariates for use with risk score imputation

**Details**

The data frame contains the following columns: 'Id' for subject ID 'time.start' and 'time.end' the range of time for which the covariate values are valid - i.e. [time.start,time.end] Additional columns are the time dependent covariates

All data for a single subject should be stored in consecutive rows, sorted by time and the starting time of a row should match the ending time of the previous row

**See Also**

MakeTimeDepScore
ScoreTimeDep | Simulated time dependent variables for time to event data

**Description**

This data set contains time dependent covariates for the ScoreInd time to event data.

**Format**

A data.frame containing 1 row per subject-visit

**Fields**

- **Id** The Subject Id
- **start** The covariate given in each row are for a given subject from time 'start'...
- **end** ... until time end
- **W1** The value of a (binary) time dependent variable for the subject with the given 'Id' between times 'start' and 'end'
- **W2** The value of a (continuous) time dependent variable for the subject with the given 'Id' between times 'start' and 'end'

**Summary.ScoreStatSet** | Summary object of ScoreStatSet object

**Description**

This object contains the multiple imputed averages/p-values of a set of estimates from risk score imputed data sets.

**Details**

A print.Summary.ScoreStatSet function has been implemented

This object contains three lists meth1 and meth2 and methRubin meth1 averages the point estimates to produce an F test statistic, meth2 averages the test statistics and produces a t test statistic and methRubin follows Rubin's standard rules and is used for calculating confidence intervals

See the vignette for further details.

meth1, meth2 and methRubin are lists with the following elements: estimate: average estimator for meth1, NOTE: for meth2 this is the average test statistic, var: estimate of variance of "estimate" field

test.stat: test statistic
distribution: distribution of statistical test (i.e. F or t)
p.value: p-value of test
Index

col.headings, 3, 12, 14
cox.zph, 3, 4

ExtractSingle, 4

gammaImpute, 5
GammaImputedData.object, 6, 7, 8
GammaImputedSet.object, 6, 7
GammaStat.object, 8
GammaStatList.object, 8

ImputeStat, 9
InformativeCensoring
   (InformativeCensoring-package), 2
   InformativeCensoring-package, 2

MakeTimeDepScore, 10, 17

NN.options, 11, 12

ridge, 13

ScoreImpute, 12
ScoreImputedData.object, 10, 13, 14
ScoreImputedSet.object, 13, 14
ScoreInd, 15, 18
ScoreStat.object, 10, 15, 16, 17
ScoreStatList.object, 16
ScoreStatSet, 16
ScoreStatSet.object, 16, 17, 17
ScoreTD.object, 17
ScoreTimeDep, 18
summary.ScoreStatSet, 17, 18