Package 'salbm'

May 25, 2021

Version 1.0

Type Package

Date 2021-05-20

Title Sensitivity Analysis for Binary Missing Data

Author Daniel O. Scharfstein [aut],
Aidan McDermott [aut, cre]

Maintainer Aidan McDermott <amcderm1@jhu.edu>

Description In a clinical trial with repeated measures designs, outcomes are often taken from subjects at fixed time-points. The focus of the trial may be to compare the mean outcome in two or more groups at some pre-specified time after enrollment. In the presence of missing data auxiliary assumptions are necessary to perform such comparisons. One commonly employed assumption is the missing at random assumption (MAR). The 'salbm' package allows the user to perform a (parameterized) sensitivity analysis of this assumption where the outcome of interest is binary (coded as 0, 1, or NA). In particular it can be used to examine the sensitivity of tests in the difference in outcomes to violations of the MAR assumption. See the paper Daniel O. Scharfstein, Jaron J. R. Lee, Aidan McDermott, Aimee Campbell, Edward Nunes, Abigail G. Matthews, Ilya Shpitser ``Markov-Restricted Analysis of Randomized Trials with Non-Monotone Missing Binary Outcomes: Sensitivity Analysis and Identification Results" (2021) <arXiv:2105.08868>.

Imports randomForestSRC

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2021-05-25 07:30:02 UTC

R topics documented:

`addSamples` *Sensitivity Analysis for Binary Missing Data*

Description

This is a generic method which adds bootstrap samples together with their associated results to an object of class salbm or salbmM

Usage

```
addSamples(obj, NBootstraps = 0, bBS = 1, nseeds = c(5,9), nseeds2 = c(-4,-5),
                 returnJP = TRUE, returnSamples = FALSE, ...)
```

Arguments

`obj`	an object of class salbm or salbmM
`NBootstraps`	number of bootstraps requested
`bBS`	beginning index of bootstraps
`nseeds`	a numeric vector of length 2 to be used in makeing subsamples
`nseeds2`	a numeric vector of length 2 passed to randomForestSRC
`returnJP`	a logical, if true, the item, jps is copied to the output object
`returnSamples`	a logical, if true, the bootstrap samples are added to the output object
`...`	additional parameters passed to methods

Details

If the input object, obj, is of type salbm, then obj will contain an items jps1 and jps2 which are the joint probability distributions of Y_1, ..., Y_K derived from the runs of randomForestSRC applied to treatment 1 and treatment 2 respecctively. If the input object, obj, is of type salbmM, then obj will contain an items jps1 and jps2 which are a series of probability distributions derived from randomForestSRC under the Markovian assumption paramaterized by the integer m.

In either case, these jps1 and jps2 are used to create NBootstrap samples for treatment 1 and treatment 2 respectively.. Each bootstrap sample is analysed using either salbm or salbmM.

All results in the original object, obj, are copied to the output object except that any new bootstrap results are appended to the original bootstrap results.

The parameter NBootstraps determines the number of bootstraps to create. The sBS parameter allows the new bootstrap results to be be indexed by sBS:(sBS+NBootstraps-1). This is useful when running addSamples in parallel.

nseeds is of the form c(sd1,sd2). set.seed(sd1) is run once before bootstraps from treatment group 1 are created and analyzed and set.seed(sd2) is run once before bootstraps from treatment group 2 are creaded and analyzed.

nseeds2 is also of the form c(sd1,sd2). The values of sd1 and sd2 should be negative. The index for a given bootstrap sample (a value between sBS and sBS+NBootstraps-1) is subtracted from sd1 and passed to randomForestSRC when a bootstrap from treatment group 1 is analyzed and is subtraced from sd2 and passed to randomForestSRC when a bootstrap from treatment group 2 is analyzed.

The returnJP controls whether jp1 and jp2 are copied to the output object. Since these can be large, it is often advisable to set returnJP = FALSE.

Value

salbm returns a list which contains the following:

`Main1R`	results for treatment group 1 in wide format
`Main1RL`	results for treatment group 1 in long format
`Main1wts`	means and standard deviations for trt1
`jps1`	joint distribution returned from randomForestRSC, trt 1
`Samp1R`	results for bootstrap samples trt1 in wide format
`Samp1RL`	results for bootstrap samples trt1 in long format
`Samp1wts`	means and standard deviations of bootstrap samples trt1.
`Main2R`	results for treatment group 2 in wide format
`Main2RL`	results for treatment group 2 in long format
`Main2wts`	means and standard deviations for trt2
`jps2`	joint distribution returned from randomForestRSC trt 2
`Samp2R`	results for bootstrap samples trt2 in wide format
`Samp2RL`	results for bootstrap samples trt2 in long format
`Samp2wts`	means and standard deviations of bootstrap samples trt2.
`data`	the salbm data object supplied in the call to salbm
`K`	the value of K supplied in the call to salbm
`ntree`	the value of ntree supplied in the call to salbm
`alphas`	the value of alphas supplied in the call to salbm
`seeds`	the value of seeds supplied in the call to salbm
`seeds2`	the value of seeds2 supplied in the call to salbm
`bBS`	the value of bBS supplied in the call to salbm
`eBS`	the value of eBS supplied in the call to salbm
`NBootstraps`	the value of NBootstraps supplied in the call to salbm

See Also

The salbn_userDoc.pdf file in the Examples subdirectory.

Examples

```
# Clinical trial data with two arms placebo/dose.

data <- list( trt1 = trt1, trt2 = trt2 )
# original run - no bootstraps
R <- salbm( data = data , K = 6, ntree = 1000,
            seeds = c(22,18), seeds2 = c(-2,-3),
            alphas = -5:5, NBootstraps=0 )

# add 100 bootstraps
Rupd <- addSamples(R, NBootstraps=100, sBS=1,
          nseeds = c(81,80), nseeds2 = c(-6,-1),
          returnJP=FALSE)


# Markov assumption m = 2 again no bootstraps
RM <- salbmM( data = data , K = 6, m = 2,
              ntree = 1000, seeds2 = c(-2,-3),
              alphas = -5:5, NBootstraps=0 )

# add 100 bootstraps
RMupd <- addSamples(RM, NBootstraps=100, sBS=1,
         nseeds = c(81,80), nseeds2 = c(-6,-1),
         returnJP=FALSE)

# Markov assumption m = 3, empirical estimates
RME <- salbmMEst( data = data , K = 11, m = 3,
              MEst = TRUE, NMest = 25000,
              ntree = 1000, seeds2 = c(-2,-3),
              alphas = -5:5, NBootstraps=100 )

# and add 100 bootstraps
RMEupd <- addSamples(RME, NBootstraps=100, sBS=101,
         nseeds = c(81,80), nseeds2 = c(-6,-1),
         returnJP=FALSE)
```

`getCI` *Sensitivity Analysis for Binary Missing Data*

Description

Given an object returned by salbmCombine the getCI function extracts estimates and confidence intervals.

Usage

```
getCI( Res, ci=c("lb4","ub4") )
```

Arguments

Res	an object returned by salbmM, salbm with confidence intervals
ci	indicates which confidence interval to return. See the help for salbmCombine for the list of confidence interval types.

Details

Looks for ECI and SCI in the input object. ECI contains confidence bands for estimates of E[Yt | alpha] and their differences. SCI contains confidence bands for estimates of the cumulative sum E[Y1+Y2+...+YK | alpha] and their differences.

Value

returns a list with 1. K the value of the last timepoing 2. alphas the vector of sensitivity parameters 3. ER1 an n by 4 matrix where n is the length of the alphas vector. ER1 contains expected values of Y at time K for the first treatment group trt1. The columns are alpha, E[Y_K| alpha], lb, ub where lb and ub are the lower bound and upper bound of the confidence interval chosen. 4. ER2 as ER1 but for the second treatment arm trt2. 5. ERD the difference in estimates trt2 - trt1. 6. SR1 cumulative sum estimates for trt 1 7. SR2 cumulative sum estimates for trt 2 8. SRD differences in cumulative sum estimates trt 2 - trt 1.

Examples

```
# select confidence interval type
Res  <-  getCI( R, ci=c("lb2","ub2"))
ER1  <-  Res$ER1
plot( x = ER1[,1], y = ER1[, 2] )
```

---

getEYnomo *Sensitivity Analysis for Binary Missing Data*

Description

returns the difference in the expected value of the outcome, Y, when Y is not observed and the expected value Y when Y is observed as a percentage of the expected value of Y when observed.

Usage

```
getEYnomo( Y, BC, ratio=FALSE)
```

Arguments

Y	a vector of observed data including NAs.
BC	a vector of salbm estimates
ratio	if true return the ratio 100 * E[Y|non-obs] / E[Y|obs]; otherwise return 100 * (E[Y|non-obs] - E[Y|obs])/ E[y|obs]

Details

Typically, BC is a vector of salbm at different sensitivity parameters alpha. The function computes: 100 (E[Y | Y not observed] - E[Y | Y is observed]) / E[Y | Y is observed] when ratio is FALSE and computes 100 E[Y | Y not observed] / E[Y | Y is observed] when ratio is TRUE

Value

A vector containing 100 (E[Y | Y not observed] - E[Y | Y is observed]) / E[Y | Y is observed] or 100 E[Y | Y not observed] / E[Y | Y is observed]

Examples

```
# trt1 is one of two arms of a clinical trial with
# trt1[,K] being the measurments at the last timepoint
# BC has two columns the first containing sensitivity
# paramenters and the second their corrisponding salbm
# estimates

pdiff  <- getEYnomo( trt1[,K], BC[,2] )
plot( x = BC[,1], y = pdiff )
```

---

getXCI *Sensitivity Analysis for Binary Missing Data*

Description

Given an object returned by salbmCombine the getXCI function extracts difference estimates and confidence intervals by pairs of sensitivity paramaters.

Usage

```
getXCI( Res, ci=c("lb4","ub4") )
```

Arguments

Res	an object returned by salbmCombine with confidence intervals
ci	indicates which confidence interval to return. See the help for salbmCombine for the list of confidence interval types.

Details

Looks for ECI and SCI in the input object. For each pair of sensitivity parameters ECI contains confidence bands for estimates of E[Yt | alpha] and their differences. SCI contains confidence bands for estimates of the cumulative sum E[Y1+Y2+...+YK | alpha] and their differences.

Value

returns a list with 1. K the value of the last time-point 2. alphas the vector of sensitivity parameters 3. ER1 an n^2 by 5 matrix where n is the length of the alphas vector. ER1 contains expected values of Y at time K for the first treatment group trt1. The columns are alpha1, alpha2, E[Y_K| alpha], lb, ub where lb and ub are the lower bound and upper bound of the confidence interval chosen, alpha1 is the sensitivity parameter associated with treatment group 1 and alpha2 the sensitivity parameter associated with treatment group 2. 4. ER2 as ER1 but for the second treatment arm trt2. 5. ERD the difference in estimates trt2 - trt1. 6. SR1 cumulative sum estimates for trt 1 7. SR2 cumulative sum estimates for trt 2 8. SRD differences in cumulative sum estimates trt 2 - trt 1.

Examples

```
fplots <- getXCI( Res=Results, lb=c("lb2","ub2") )
alphas <- fplots$alphas
ERD    <- fplots$ERD

## significant points
Epts   <-  ERD[ sign(ERD[,4]) == sign(ERD[,5]), ]

filled.contour( x = alphas, y = alphas,
   z              = matrix(ERD[,3],length(alphas),length(alphas), byrow=TRUE),
   xlab           = expression(paste(alpha, " (Treatment 1)")),
   ylab           = expression(paste(alpha, " (Treatment 2)")),
   nlevels        = 8,
 color.palette  = colorRampPalette(c( "#993404","#D95F0E","#FE9929","#FFD9BE","#FFFFD4"),
                                   space="rgb"),
   plot.axis      = ( points( Epts[,c(1,2)], pch=15, cex=0.8, col = "#777799FF" ) )
)
```

`PrepDescribeTrt` *Sensitivity Analysis for Binary Missing Data*

Description

Given a dataframe Y in which columns represent timepoints and rows represent subjects creates summaries for display.

Usage

```
PrepDescribeTrt(Y)
```

Arguments

Y	Is an N0 by NT dataframe with each column representing timepoints and rows representing subjects. Individual values Y_ij = 0, when the event of interest has not occured for subject i at time j, Y_ij = 1, when the event of interest has occured for subject i at time j, Y_ij = 2 or NA when the measurement of the event of interest for subject i at time j is missing.

Details

PrepDescriveTrt is used in conjunction with describeTrt to produce a simple summary of a salmb dataframe.

Value

PrepDescribeTrt returns a table with 10 columns: 1. t 2. Number On Study 3. Number Observed 4. last seen 5. Proportion last seen (of on-study) 6. Proportion last seen (of observed) 7. inter miss 8. proportion inter missing (of onstudy) 9. mean observed 10. std observed

See Also

salbm, salbmM

Examples

```
data(trt1)

# switch the 2s to NAs for tables.
trt1[ trt1 == 2 ] <- NA
prep1 <- PrepDescribeTrt(trt1)
```

PrepInferenceAssumption

Sensitivity Analysis for Binary Missing Data

Description

Given an object returned from salbmCombine this function prepares results for presentation in a table.

Usage

```
PrepInferenceAssumption(M, CItp=2, qt = c(0.025, 0.975), SampCIType=1, carm=2)
```

Arguments

M	is an object returned by the salbm function salbmCombine. It should contain confidence bands ECI and SCI and means and standard deviations from bootstraps (Main1wts, Main2wts, Samp1wts and Samp2wts).
CItp	a number ranging from 1 to 4, indicating which confidence interval to use.
qt	two values passed to quantile in producing confidence intervals.
SampCIType	type of CI for samples
carm	comparison arm

Details

PrepInferenceAssumption is used in conjunction with influenceAssumption to produce a summary of an salmb results object,

It produces summeries for each arm of the study and the difference in estimates between arms under various assumptions:

MCAR when the mean value is substituted for missing at each timepoint

missing=0 when 0 is substituted for missing at each timepoint

missing=1 when 1 is substituted for missing at each timepoint

benchmark results returned from salbm or salbmM

Value

a matrix which inferenceAssumption can display.

See Also

salbm, salbmM, influenceAssumption

Examples

```
# M is an object returned from salbm's salbmCombine function

prep <- PrepInferenceAssumption( M )
inferenceAssumption(prep)
```

PrepMissingPattern *Sensitivity Analysis for Binary Missing Data*

Description

Given a dataframe Y in which columns represent timepoints and rows represent subjects creates missing data summaries for display using missingPattern.

Usage

```
PrepMissingPattern(Y)
```

Arguments

Y	Is an N0 by NT dataframe with each column representing timepoints and rows representing subjects. Individual values Y_ij = 0, when the event of interest has not occured for subject i at time j, Y_ij = 1, when the event of interest has occured for subject i at time j, Y_ij = 2 or NA when the measurement of the event of interest for subject i at time j is missing.

Details

PrepMissingPattern is used in conjunction with missingPattern to produce a simple summary of missing data in a salmb dataframe.

Value

Returns a list with items

Mat	a matrix of frequencies and percentages
tots	a matrix of subtotals
K	the number of timepoints (columns) in the original data

Examples

```
data(trt1)
prep1 <- PrepMissingPattern(trt1)
```

salbm *Sensitivity Analysis for Binary Missing Data*

Description

For a list of dataframes, where each frame is of the form (Y_1,Y_2, ..., Y_K) and Y_t takes the values 0, 1, or 2 (missing), salbm estimates E[Y_t | alpha] where alpha is one of a number of sensitivity paramaters.

Usage

```
salbm( data, Narm = length(data), K, ntree,
       seeds = 1:length(data), seeds2 = -1 - 1:length(data),
       alphas, NBootstraps = 0, bBS = 1,
       returnJP = TRUE, returnSamples = FALSE)
```

Arguments

data	a list of dataframes
Narm	the number of dataframes to process
K	the number of time-points
ntree	the number of trees in the random forest passed to randomForestSRC
seeds	vector of positive numbers used as seeds in producing bootstrap samples. There should be at least one seed for each treatment arm.
seeds2	vector of negative numbers passed to randomForestSRC. There should be at least one seed for each treatment arm.
alphas	vector of sensitivity parameters
NBootstraps	number of bootstrap samples to be created and analyzed
bBS	Start bootstrap number. Bootstrap ids are given as bBS:eBS where eBS = bBS + NBootstraps - 1. Setting bBS to a value other than 1 is useful when running salbm in parallel.
returnJP	Logical indicating if the joint probability distribution for each treatment group should be returned. This can be used by addSamples to create Bootstrap samples.
returnSamples	Logical indicating if generated bootstrap samples should be returned

Details

The list data contains Narm dataframes each of which has the form (Y_1,Y_2, ..., Y_K) where each Y_t takes the value 0 or 1 when Y_t is observed and has value 0 or 1 respectively and Y_t takes the value NA (or 2) when Y_t is not observed.

For each dataframe separately, randomForestSRC is used to create the joint probability distribution f(Y_1,Y_2, ..., Y_K) where Y_t can take three possible values, 0, 1, or missing (represented by the value 2).

This joint distributins is "tilted" once for each sensitivity parameter alpha to produce the fully observed joint distribution, t(Y_1,Y_2, ... Y_K | alpha).

Statistics such as E[Y_K | Y_K is fully observed, alpha] can then be computed.

Precision estimates are calculated using the bootstrap with bootstraps generated from f(Y_1,Y_2, ..., Y_K).

Even for quite modest values of K the number of possible combinations of {Y_i} can be large and the representation in memory of the joint distribution function, f, can be problematic. For larger sized values of K it might be worth considering placing a Markovian assumption on the distributions of f(Y_j) and this can be analysed using salbmM.

Value

salbm returns a list which contains the following:

`Main1R`	results for treatment group 1 in wide format
`Main1RL`	results for treatment group 1 in long format
`Main1wts`	means and standard deviations for trt1
`jps1`	joint distribution returned from randomForestRSC, trt 1
`Samp1R`	results for bootstrap samples trt1 in wide format
`Samp1RL`	results for bootstrap samples trt1 in long format
`Samp1wts`	means and standard deviations of bootstrap samples trt1.
`Main2R`	results for treatment group 2 in wide format
`Main2RL`	results for treatment group 2 in long format
`Main2wts`	means and standard deviations for trt2
`jps2`	joint distribution returned from randomForestRSC trt 2
`Samp2R`	results for bootstrap samples trt2 in wide format
`Samp2RL`	results for bootstrap samples trt2 in long format
`Samp2wts`	means and standard deviations of bootstrap samples trt2.
`data`	the salbm data object supplied in the call to salbm
`K`	the value of K supplied in the call to salbm
`ntree`	the value of ntree supplied in the call to salbm
`alphas`	the value of alphas supplied in the call to salbm
`seeds`	the value of seeds supplied in the call to salbm
`seeds2`	the value of seeds2 supplied in the call to salbm
`bBS`	the value of bBS supplied in the call to salbm
`eBS`	the value of eBS supplied in the call to salbm
`NBootstraps`	the value of NBootstraps supplied in the call to salbm

See Also

salbmM, addSamples.

Examples

```
data(trt1)
data(trt2)
data <- list( trt1 = trt1[,1:3], trt2 = trt2[,1:3] )

R <- salbm( data = data, K = 3, ntree = 500,
            seeds = c(22,18), seeds2 = c(-2,-3),
            alphas = -1:1, NBootstraps=0 )
```

`salbmCombine` *Sensitivity Analysis for Binary Missing Data*

Description

Combines main and bootstrap results.

Usage

```
salbmCombine(x0, Samps=NULL, div=c(NA,NA))
```

Arguments

`x0`	main results from original data.
`Samps`	list of sample results
`div`	low and high alpha points

Details

Combines data and produces bootstrap summaries including confidence intervals.

Four types of confidence intervals are calculated after combining the results. Fixing the sensitivity parameter, let E be an estimate for a treatment group and for $b = 1, 2, \ldots, B$ let E_b be the corresponding estimate for bootstrap b. Let SE be a standard error estimate for E and SE_b be a standard error estimate for E_b. Let $q(\{x_i\}, v)$ be the vth quantile for the collection $\{x_i\}$.

1. Let $q_l = q(\{E_b\}, 0.025)$ and $q_u = q(\{E_b\}, 0.975)$. Then a 95% confdence interval for E is given by (q_l, q_u). These are refered to as c("lb1","ub1") or type 1 confidence intervals.

2. Let $q_s = q(\{\mid E_b - E \mid\}, 0.95)$. Then a 95% confdence interval for E is given by $(E - qs, E + qs)$. These are refered to as c("lb2","ub2") or type 2 confidence intervals.

3. Let $t_b = (E_b - E)/SE_b$, eqnq_l = q({t_b}, 0.025) and $q_u = q(\{t_b\}, 0.975)$. Let M be the mean of E_b. Then a 95% confdence interval for eqnE is given by $(M - q_u SE, M - q_l SE)$ These are refered to as c("lb3","ub3") or type 3 confidence intervals.

4. Let $t_b = (E_b - E)/SE_b$, and $q_s = q(\{\mid t_b \mid\}, 0.95)$. Let M be the mean of E_b. Then a 95% confdence interval for E is given by $(M - q_s SE, M + q_s SE)$ These are refered to as c("lb4","ub4") or type 4 confidence intervals.

Computing SE

Let D be a dataset with n rows and T a fixed timepoint. Three standard deviations are computed

SD_0 the standard deviation of the data in D at timepoint T when 0 is substituted for missing values in D,

SD_1 the standard deviation of the data in D at timepoint T when 1 is substituted for missing values in D, and

SD_m the standard deviation of the data in D at timepoint T when mean value at time T is substituted for missing values at time T.

A low, mid and high values of alpha are choosen and denoted by a_l, a_0, and a_u respectively. In salbm $a_0 = 0$. Then a standard error is computed as:

$$SE = \begin{cases} \frac{(\alpha - a_l)SD_0 + (a_0 - \alpha)SD_m}{(a_0 - a_l)\sqrt{n}} & \text{when } \alpha < 0 \\ \frac{(a_u - \alpha)SD_1 + (\alpha - a_0)SD_m}{(a_u - a_0)\sqrt{n}} & \text{when } \alpha \geq 0 \end{cases}$$

Value

a list with combined results

See Also

salbm

Examples

```
data(trt1)
data(trt2)
data <- list( trt1 = trt1, trt2 = trt2 )

## main run
x0 <- salbm( data = data , K = 6, ntree = 250,
             seeds = c(22,18), seeds2 = c(-2,-3),
             alphas = -5:5, NBootstraps=0 )

## add Bootstraps
samp1 <- addSamples(obj=x0, seeds=c(99,12),
             seeds2 = c(-45,-80), bBS=1,
             NBootstraps=250)
## add more Bootstraps
samp2 <- addSamples(obj=x0, seeds=c(9,2),
             seeds2 = c(-54,-8), bBS=251,
             NBootstraps=250)

## Together
R <- salbmCombine(x0=x0, Samps=list(samp1,samp2))
```

salbmM *Sensitivity Analysis for Binary Missing Data*

Description

For a list of dataframes, where each frame is of the form (Y_1,Y_2, ..., Y_K) and Y_t takes the values 0, 1, or 2 (missing), salbmM estimates E[Y_t | alpha] where alpha is one of a number of sensitivity paramaters under a Markovian assumption of order m.

Usage

```
salbmM( data, Narm = length(data), m, K, ntree,
        EmpEst=FALSE, NEst=0,
        seeds = 1:length(data), seeds2 = -1 - 1:length(data),
        alphas, NBootstraps = 0, bBS = 1,
        returnJP = TRUE, returnSamples = FALSE )
```

Arguments

data	a list of dataframes
Narm	the number of dataframes to process
m	order of the Markov assumption, note 2m+2 < K
K	The number of time-points
ntree	The number of trees in the random forest passed to randomForestSRC
EmpEst	logical, indicating if empirical estimation should be used when calculating the mean value of Yt.
NEst	The number of values of Yt to use in calculating the mean of Yt.
seeds	vector of positive numbers used as seeds in producing bootstrap samples. There should be at least one seed for each treatment arm.
seeds2	vector of negative numbers passed to randomForestSRC. There should be at least one seed for each treatment arm.
alphas	vector of sensitivity parameters
NBootstraps	number of bootstrap samples to be created and analyzed
bBS	Start Bootstrap number. Bootstrap IDs are given as bBS:eBS where eBS = bBs + NBootstraps - 1. Setting bBS and eBS is useful when running salbmM in parallel.
returnJP	Logical indicating if the list of joint probability distributions returned by random forest for each treatment group should be returned. This is used by addSamples to create Bootstrap samples.
returnSamples	Logical indicating if generated bootstrap samples should be returned

Details

For each dataframe separately, randomForestSRC is used to create a set of joint distributions f(Yn-m, Yn-m+1, ..., Yn-1, Yn, Yn+1, ... Yn+m+1) where Yi can take three possible values, 0, 1, or missing (represented by the value 2). The Markovian assumption of order m can be summarized as f(Y_n | Y_i, i = 1, 2, ..., n-1, n+1, ..., K) = f(Y_n | Y_i, i = max(1,n-m), ..., n-1, n+1, ..., min(n+m+1,K)) for n > 1.

RandomForestSRC is used to estimate the joint distributions, f_i(Y_n | Y_n-m, ..., Y_n-1, Y_n+1, ..., Y_n+m+1). For each sensitivity parameter, alpha, these distributions are used to compute the E[Y_K | alpha] Bootstraping is carried out using the f_i.

Because of the Markov assumption the full distribution f can be replaced by a set of distributions of order no more than 2m+2. This allows estimation in situations where K is large and estimation of the full joint distribution is unfeasable.

Value

salbmM returns a list which contains the following:

`Main1R`	results for treatment group 1 in wide format
`Main1RL`	results for treatment group 1 in long format
`Main1wts`	means and standard deviations for trt1
`jps1`	joint distribution returned from randomForestRSC, trt 1
`Samp1R`	results for bootstrap samples trt1 in wide format
`Samp1RL`	results for bootstrap samples trt1 in long format
`Samp1wts`	means and standard deviations of bootstrap samples trt1.
`Main2R`	results for treatment group 2 in wide format
`Main2RL`	results for treatment group 2 in long format
`Main2wts`	means and standard deviations for trt2
`jps2`	joint distribution returned from randomForestRSC trt 2
`Samp2R`	results for bootstrap samples trt2 in wide format
`Samp2RL`	results for bootstrap samples trt2 in long format
`Samp2wts`	means and standard deviations of bootstrap samples trt2.
`data`	the salbm data object supplied in the call to salbmM
`m`	the Markov paramater supplied in the call to salbmM
`K`	the value of K supplied in the call to salbmM
`ntree`	the value of ntree supplied in the call to salbmM
`NEst`	the value of NEst supplied in the call to salbmM
`alphas`	the value of alphas supplied in the call to salbmM
`seeds`	the value of seeds supplied in the call to salbmM
`seeds2`	the value of seeds2 supplied in the call to salbmM
`bBS`	the value of bBS supplied in the call to salbmM
`eBS`	the value of eBS supplied in the call to salbmM
`NBootstraps`	the value of NBootstraps supplied in the call to salbmM

Examples

```
# Clinical trial data with two arms.
data(trt1)
data(trt2)
data <- list( trt1 = trt1, trt2 = trt2 )

R     <-  salbmM( data = data , m = 2, K = 6, ntree = 1000,
             seeds = c(22,18), seeds2 = c(-2,-3),
             alphas = -8:8, NBootstraps=0 )
```

`The Trt data.` *Sensitivity Analysis for Binary Missing Data*

Description

These dataframes store data from a repeated measures clinical trial. The data are simulated. Trt1 contains data from the control arm and Trt2 contains data from the active arm.

Usage

```
data(trt1)
data(trt2)
```

Format

trt1 and trt2 are 300 by 6 dataframes

Details

In each of trt1 and trt2 a row represtents the outcome measure for one subject at each of 3 timepoints. The data are coded as 0 or 1 for the binary outcome and as 2 for missing data. The datasets are delibertly small and are used in the help examples.

- y1 measure at baseline time-point 1
- y2 measure at follow-up time-point 2
- y3 measure at follow-up time-point 3
- y4 measure at follow-up time-point 4
- y6 measure at follow-up time-point 5
- y6 measure at follow-up time-point 6

Index