Package 'Iscores'

January 20, 2025

Type Package

Title Proper Scoring Rules for Missing Value Imputation

Version 1.1.0

Author Loris Michel, Meta-Lina Spohn, Jeffrey Naef

Maintainer Loris Michel <michel@stat.math.ethz.ch>

Description
Implementation of a KL-based scoring rule to assess the quality of different missing value imputations in the broad sense as introduced in Michel et al. (2021) <arXiv:2106.03742>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.1

Depends parallel, stats, ranger, kernlab

NeedsCompilation no

Repository CRAN

Date/Publication 2022-01-05 00:00:08 UTC

Contents

class.balancing *Balancing of Classes*

Description

Balancing of Classes

Usage

```
class.balancing(X.proj.complete, Y.proj, drawA, Xhat, ids.with.missing, vars)
```

Arguments

X.proj.complete	matrix with complete projected observations.
Y.proj	matrix with projected imputed observations.
drawA	vector of indices corresponding to current missingness pattern.
Xhat	matrix of full imputed observations.
ids.with.missing	vector of indices of observations with missing values.
vars	vectors of variables in projection.

Value

a list of new X.proj.complete and Y.proj.

combine2Forests *Combining projection forests*

Description

Combining projection forests

Usage

```
combine2Forests(mod1, mod2)
```

Arguments

mod1	first forest
mod2	second forest

Value

a new forest combining the first and the second forest

combineForests *Combining a list of forest*

Description

Combining a list of forest

Usage

```
combineForests(list.rf)
```

Arguments

list.rf	a list of forests

Value

a forest combination of the forests stored in list.rf

compute_drScore *compute the density ratio score*

Description

compute the density ratio score

Usage

```
compute_drScore(object, Z = Z, num.trees.per.proj, num.proj)
```

Arguments

object	a crf object.
Z	a matrix of candidate points.
num.trees.per.proj	an integer, the number of trees per projection.
num.proj	an integer specifying the number of projections.

Value

a numeric value, the DR I-Score.

`densityRatioScore` *Computation of the density ratio score*

Description

Computation of the density ratio score

Usage

```
densityRatioScore(
  X,
  Xhat,
  x = NULL,
  num.proj = 10,
  num.trees.per.proj = 1,
  projection.function = NULL,
  min.node.size = 1,
  normal.proj = T
)
```

Arguments

`X`	a matrix of the observed data containing missing values.
`Xhat`	a matrix of imputations having same size as X.
`x`	pattern of missing values.
`num.proj`	an integer specifying the number of projections.
`num.trees.per.proj`	an integer, the number of trees per projection.
`projection.function`	a function providing the user-specific projections.
`min.node.size`	the minimum number of observations in a leaf of a tree.
`normal.proj`	a boolean, if TRUE, sample from the NA of the pattern and additionally from the non NA. If FALSE, sample only from the NA of the pattern.

Value

a fitted random forest based on random projections

doevaluation *doevaluation: compute the imputation KL-based scoring rules*

Description

doevaluation: compute the imputation KL-based scoring rules

Usage

```
doevaluation(
  imputations,
  methods,
  X.NA,
  m,
  num.proj,
  num.trees.per.proj,
  min.node.size,
  n.cores = 1,
  projection.function = NULL
)
```

Arguments

imputations	a list of list of imputations matrices containing no missing values of the same size as X.NA
methods	a vector of characters indicating which methods are considered for imputations. It should have the same length as the list imputations.
X.NA	a matrix containing missing values, the data to impute.
m	the number of multiple imputation to consider, defaulting to the number of provided multiple imputations.
num.proj	an integer specifying the number of projections to consider for the score.
num.trees.per.proj	an integer, the number of trees per projection.
min.node.size	the minimum number of nodes in a tree.
n.cores	an integer, the number of cores to use.
projection.function	a function providing the user-specific projections.

Value

a vector made of the scores for each imputation method.

Iscores *Iscores: compute the imputation KL-based scoring rules*

Description

Iscores: compute the imputation KL-based scoring rules

Usage

```
Iscores(
  imputations,
  methods,
  X.NA,
  m = length(imputations[[1]]),
  num.proj = 100,
  num.trees.per.proj = 5,
  min.node.size = 10,
  n.cores = 1,
  projection.function = NULL,
  rescale = TRUE
)
```

Arguments

imputations	a list of list of imputations matrices containing no missing values of the same size as X.NA
methods	a vector of characters indicating which methods are considered for imputations. It should have the same length as the list imputations.
X.NA	a matrix containing missing values, the data to impute.
m	the number of multiple imputation to consider, defaulting to the number of provided multiple imputations.
num.proj	an integer specifying the number of projections to consider for the score.
num.trees.per.proj	an integer, the number of trees per projection.
min.node.size	the minimum number of nodes in a tree.
n.cores	an integer, the number of cores to use.
projection.function	a function providing the user-specific projections.
rescale	a boolean, TRUE if the scores should be rescaled such that the max score is 0.

Value

a vector made of the scores for each imputation method.

Examples

```
n <- 100
X <- cbind(rnorm(n),rnorm(n))
X.NA <- X
X.NA[,1] <- ifelse(stats::runif(n)<=0.2, NA, X[,1])

imputations <- list()

imputations[[1]] <- lapply(1:5, function(i) {
 X.loc <- X.NA
 X.loc[is.na(X.NA[,1]),1] <- mean(X.NA[,1],na.rm=TRUE)
 return(X.loc)
})

imputations[[2]] <- lapply(1:5, function(i) {
 X.loc <- X.NA
 X.loc[is.na(X.NA[,1]),1] <- sample(X.NA[!is.na(X.NA[,1]),1],
 size = sum(is.na(X.NA[,1])), replace = TRUE)
 return(X.loc)
})

methods <- c("mean","sample")

Iscores(imputations,
methods,
X.NA,
num.proj=5
)
```

`sample.vars.proj` *Sampling of Projections*

Description

Sampling of Projections

Usage

```
sample.vars.proj(ids.x.na, X, projection.function = NULL, normal.proj = T)
```

Arguments

`ids.x.na`	a vector of indices corresponding to NA in the given missingness pattern.
`X`	a matrix of the observed data containing missing values.
`projection.function`	a function providing the user-specific projections.
`normal.proj`	a boolean, if TRUE, sample from the NA of the pattern and additionally from the non NA. If FALSE, sample only from the NA of the pattern.

Value

a vector of variables corresponding to the projection.

`truncProb` *Truncation of probability*

Description

Truncation of probability

Usage

```
truncProb(p)
```

Arguments

p	a numeric value between 0 and 1 to be truncated

Value

a numeric value, the truncated probability.

Index