Package ‘GenForImp’

February 27, 2015

Type Package

Title The Forward Imputation: A Sequential Distance-Based Approach for Imputing Missing Data

Version 1.0

Date 2015-02-27

Author Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

Maintainer Alessandro Barbiero <alessandro.barbiero@unimi.it>

Description Two methods based on the Forward Imputation approach are implemented for the imputation of quantitative missing data. One method alternates Nearest Neighbour Imputation and Principal Component Analysis (function 'ForImp.PCA'), the other uses Nearest Neighbour Imputation with the Mahalanobis distance (function 'ForImp.Mahala').

License GPL-3

Depends mvtnorm, sn

NeedsCompilation no

Repository CRAN

Date/Publication 2015-02-27 19:31:42

R topics documented:

`GenForImp-package` *The Forward Imputation: A Sequential Distance-Based Approach for Imputing Missing Data*

Description

Two methods based on the Forward Imputation (*ForImp*) approach are implemented for the imputation of quantitative missing data. One method alternates the Nearest Neighbour Imputation (NNI) method and Principal Component Analysis (function `ForImp.PCA`), the other uses NNI with the Mahalanobis distance (function `ForImp.Mahala`). *ForImp* is a sequential distance-based approach that performs imputation of missing data in a forward, step-by-step process involving subsets of units according to their "completeness rate". During the iterative process, the complete part of data is updated thus becoming larger and larger. No initialization of missing entries is required. *ForImp* is inherent in the nonparametric and exploratory-descriptive framework since it does not require a priori distribution assumptions on data. Two supplementary functions (`missing.gen` and `missing.gen0`) are also provided to generate Missing Completely At Random (MCAR) values on a data matrix.

Details

Package:	GenForImp
Type:	Package
Version:	1.0
Date:	2015-02-27
License:	GPL-3

Author(s)

Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

Maintainer: Alessandro Barbiero <alessandro.barbiero@unimi.it>

References

Solaro, N., Barbiero, A., Manzi. G., Ferrari, P.A. (2014). Algorithmic-type imputation techniques with different data structures: Alternative approaches in comparison. In: Vicari, D., Okada, A., Ragozini, G., Weihs, C. (eds), *Analysis and modeling of complex data in behavioural and social sciences*, Studies in Classification, Data Analysis, and Knowledge Organization. Springer International Publishing, Cham (CH): 253-261 `http://link.springer.com/chapter/10.1007/978-3-319-06692-9_27`

Solaro, N., Barbiero, A., Manzi, G., Ferrari, P.A. (2015) A sequential distance-based approach for imputing missing data: The Forward Imputation. Under review

`ForImp.Mahala`	*Imputation of missing data by using Nearest Neighbour Imputation with the Mahalanobis distance*

Description

This function imputes quantitative missing data by using Nearest Neighbour Imputation (NNI) with the Mahalanobis distance in a forward and sequential step-by-step process that starts from the complete part of data.

Usage

```
ForImp.Mahala(mat, probs=seq(0, 1, 0.1), q="10%", add.unit=TRUE, squared=FALSE,
tol=1e-6)
```

Arguments

`mat`	a quantitative data matrix with missing entries.
`probs`	vector of probabilities with values in $[0, 1]$ for computing quantiles of Mahalanobis distances in selection of donors. Default option: `probs=seq(0,1,0.1)` calculates the deciles of distances. Quantiles are computed with the generic function `quantile`.
`q`	string of the form `"X%"`, with `X`=integer. It gives the quantile of Mahalanobis distances corresponding to the first `"X%"` distances as computed (and named) by the function `quantile` with probabilities specified in the argument `probs`.
`add.unit`	a logical value. If `add.unit=TRUE` (default), the covariance matrix in the Mahalanobis distance is computed at every step of the procedure by including also the incomplete unit whose donors are to be selected. Otherwise, `add.unit=FALSE` indicates that computation involves the complete units only.
`squared`	a logical value indicating if the Mahalanobis distance has to be used (`squared= FALSE`, default) or the squared Mahalanobis distance (`squared=TRUE`).
`tol`	tolerance factor introduced to prevent numerical problems occuring when distances of complete units are equal to the choosen quantile `q`. Default is `tol=1e-6`.

Details

`ForImp.Mahala` is a forward imputation method alternative to the `ForImp.PCA` procedure for imputing quantitative missing data (see `ForImp.PCA`). It does not embrace Stage 1 since it works directly on the original variables. Regarding Stage 2, the basic metric for the NNI method is the Mahalanobis distance. Steps 2 to 3 are therefore iteratively repeated until the starting data matrix is completely imputed.

Unlike `ForImp.PCA`, the `ForImp.Mahala` procedure requires that the number n of units is equal or greater than the number p of variables at every step of the procedure, otherwise the covariance matrix involved in the Mahalanobis distance is not invertible. For further details, see the references below.

Value

The imputed data matrix.

Author(s)

Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

References

Solaro, N., Barbiero, A., Manzi. G., Ferrari, P.A. (2014). Algorithmic-type imputation techniques with different data structures: Alternative approaches in comparison. In: Vicari, D., Okada, A., Ragozini, G., Weihs, C. (eds), *Analysis and modeling of complex data in behavioural and social sciences*, Studies in Classification, Data Analysis, and Knowledge Organization. Springer International Publishing, Cham (CH): 253-261

Solaro, N., Barbiero, A., Manzi, G., Ferrari, P.A. (2015) A sequential distance-based approach for imputing missing data: The Forward Imputation. Under review

See Also

`ForImp.PCA`

Examples

```
# EXAMPLE with multivariate normal data (MVN)
# require('mvtnorm')
# number of variables
p <- 5
# correlation matrix
rho <- 0.8
Rho <- matrix(rho, p, p)
diag(Rho) <- 1
Rho
# mean vector
vmean <- rep(0,p)
vmean
# number of units
n <- 1000
# percentage of missing values
percmiss <- 0.2
nummiss <- n*p*percmiss
nummiss
# generation of a complete matrix
set.seed(1)
x0 <- rmvnorm(n, mean=vmean, sigma=Rho)
x0
# generating a matrix with missing data
x <- missing.gen(x0, nummiss)
# imputing missing values
xForImpMahala <- ForImp.Mahala(x)
xForImpMahala
# computing the Relative Mean Square Error
```

```
error <- sum(apply((x0-xForImpMahala)^2/diag(var(x0)),2,sum)) / n
error


# EXAMPLE with real data
data(airquality)
m0 <- airquality
m0
# selecting the first 4 columns, with quantitative data
m <- m0[, 1:4]
m
# imputation
mi <- ForImp.Mahala(m)
mi
# plot of imputed values for variable "Ozone"
ozone.miss.ind <- which(is.na(m)[,1])
plot(mi[ozone.miss.ind,1], axes=FALSE, pch=19, ylab="imputed values of Ozone",
  xlab="observation index")
axis(2)
axis(1, at=1:length(ozone.miss.ind), labels=ozone.miss.ind, las=2)
box()
abline(v=1:length(ozone.miss.ind), lty=3, col="grey")
```

ForImp.PCA — *Imputation of missing data by alternating Nearest Neighbour Imputation and Principal Component Analysis*

Description

This function imputes quantitative missing data by alternating Nearest Neighbour Imputation (NNI) method and Principal Component Analysis (PCA) in a forward and sequential step-by-step process that starts from the complete part of data.

Usage

```
ForImp.PCA(mat, stand=FALSE, cor=FALSE, r=2, probs=seq(0, 1, 0.1), q="10%",
tol=1e-6)
```

Arguments

`mat`	a quantitative data matrix with missing entries.
`stand`	a logical value indicating if variables should be standardized (`stand=TRUE`), or not (`stand=FALSE`, default) before the imputation process starts.
`cor`	a logical value. If `cor=TRUE`, PCA is run on the correlation matrix; if `cor=FALSE` the covariance matrix is used (default).
`r`	a positive value (equal or greater than 1) indicating the order of the weighted Minkowski distance to be computed for selecting donors. In particular, `r=1` is Manhattan (or city-block) distance, and `r=2` is Euclidean distance (default). `r=Inf` denotes Lagrange (or Chebyshev or sup) distance.

`probs`	vector of probabilities with values in $[0, 1]$ for computing quantiles of Minkowski distances in selection of donors. Default option: `probs=seq(0,1,0.1)` calculates the deciles of distances. Quantiles are computed with the generic function `quantile`.
`q`	string of the form `"X%"`, with `X`=integer. It gives the quantile of Minkowski distances corresponding to the first `"X%"` distances as computed (and named) by the function `quantile` with probabilities specified in the argument `probs`.
`tol`	tolerance factor introduced to prevent numerical problems occuring when distances of complete units are equal to the choosen quantile `q`. Default is `tol=1e-6`.

Details

The `ForImp.PCA` procedure exploits the PCA method for setting-up so-called "Pseudo-Principal Components" (PPCs). These are artificial, missing-data free variables computed for all the units (i.e. both complete and incomplete) by using common observed variables without missing values. PPCs synthesize the main relevant information in the complete part of data and transfer them to incomplete units in a way functional to the subsequent application of the NNI method. NNI is then applied to PPCs (with a weighted Minkowski distance of order $r, r \geq 1$) in order to select donors for incomplete units. All is performed in four stages: Stage 0: data preparation; Stage 1: running PCA and computing PPC scores; Stage 2: application of the NNI method; Stage 3: imputation. Steps 1 to 3 are iteratively repeated until the starting data matrix is completely imputed.

`ForImp.PCA` does not require a number n of units greater than the number p of variables at every step of the procedure. It can work even in the presence of a starting data matrix with $n < p$. This further variant is indicated as *ForImp with PCO*, i.e. Forward Imputation with the Principal Coordinates Analysis. For further details, see the references below.

Value

The imputed data matrix.

Author(s)

Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

References

Gower, J.C. (2005). Principal coordinates analysis. In: Armitage, P., Colton, T. (eds), Encyclopedia of biostatistics. John Wiley & Sons, Ltd., New York

Solaro, N., Barbiero, A., Manzi. G., Ferrari, P.A. (2014). Algorithmic-type imputation techniques with different data structures: Alternative approaches in comparison. In: Vicari, D., Okada, A., Ragozini, G., Weihs, C. (eds), *Analysis and modeling of complex data in behavioural and social sciences*, Studies in Classification, Data Analysis, and Knowledge Organization. Springer International Publishing, Cham (CH): 253-261

Solaro, N., Barbiero, A., Manzi, G., Ferrari, P.A. (2015). A sequential distance-based approach for imputing missing data: The Forward Imputation. Under review

See Also

ForImp.Mahala

Examples

```
# EXAMPLE with multivariate skew-normal data (MSN)
# require('sn')
# number of variables
p <- 5
# association matrix
omega <- 0.8
Omega <- matrix(omega, p, p)
diag(Omega) <- 1
Omega
# skewness parameter
alpha <- 4
alpha <- rep(alpha,p)
alpha
# number of units
n <- 500
# percentage of missing values
percmiss <- 0.2
nummiss <- n*p*percmiss
nummiss
## computation of output parameters
## covariance matrix and univariate means and skewnesses
param <- list(xi=rep(0,p), Omega=Omega, alpha=alpha, nu=Inf)
cp <- dp2cp(param, "SN")
cp
# correlation matrix
rho <- cov2cor(cp$var.cov)
rho
# generation of a complete matrix
set.seed(1)
x0 <- rmsn(n, Omega=Omega, alpha=alpha)
x0
# generating a matrix with missing data
x <- missing.gen(x0, nummiss)
# imputing missing values
xForImpPCA <- ForImp.PCA(x)
xForImpPCA
# computing the Relative Mean Square Error
error <- sum(apply((x0-xForImpPCA)^2/diag(var(x0)),2,sum)) / n
error

# EXAMPLE with real data
data(airquality)
m0 <- airquality
m0
# selecting the first 4 columns, with quantitative data
m <- m0[, 1:4]
m
```

```
# imputation
mi <- ForImp.PCA(m)
mi
# plot of imputed values for variable "Ozone"
ozone.miss.ind <- which(is.na(m)[,1])
plot(mi[ozone.miss.ind,1], axes=FALSE, pch=19, ylab="imputed values of Ozone",
  xlab="observation index")
axis(2)
axis(1, at=1:length(ozone.miss.ind), labels=ozone.miss.ind, las=2)
box()
abline(v=1:length(ozone.miss.ind), lty=3, col="grey")

# EXAMPLE with n < p: ForImp with PCO
# require('mvtnorm')
p <- 20
n <- 10
sigma <- matrix(0.4, p, p)
diag(sigma) <- 1
# complete matrix
set.seed(2)
xtrue <- rmvnorm(n=n, mean=rep(0, p), sigma=sigma)
rownames(xtrue) <- 1:n
colnames(xtrue) <- paste("V", 1:p, sep="")
xtrue
# matrix with 10 missing values
xmiss <- missing.gen(xtrue, 10)
xmiss
# number of missing values per unit
apply(is.na(xmiss),1,sum)
# imputed matrix
ximp <- ForImp.PCA(xmiss)
ximp
# computing the Relative Mean Square Error
error <- sum(apply((xtrue-ximp)^2/diag(var(xtrue)),2,sum)) / n
error
```

`missing.gen` *Generating random missing values on a data matrix*

Description

The function generates a number of missing values (NA) completely at random on a data matrix. Totally missing rows (i.e., rows with all NA) are avoided.

Usage

```
missing.gen(mat, nummiss)
```

Arguments

`mat`	a matrix of numerical data.
`nummiss`	number of missing values.

Details

The function generates a number of missing values (NA) completely at random on a data matrix. Totally missing rows (i.e., rows with all NA) are avoided.

Value

The data matrix with missing values (NA).

Author(s)

Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

See Also

`missing.gen0`

Examples

```
sigma <- matrix(0.4, 4, 4)
diag(sigma) <- 1
x0 <- rmvnorm(n=100, mean=rep(0, 4), sigma=sigma)
x0 # complete matrix
x <- missing.gen(x0, 50)
x # matrix with 50 missing values
```

`missing.gen0` *Generating random missing values on a data matrix*

Description

The function generates a number of missing values (NA) completely at random on a data matrix.

Usage

```
missing.gen0(mat, nummiss)
```

Arguments

`mat`	a matrix of numerical data.
`nummiss`	number of missing values.

Details

The function generates a number of missing values (NA) completely at random on a data matrix.

Value

The data matrix with missing values (NA).

Author(s)

Nadia Solaro, Alessandro Barbiero, Giancarlo Manzi, Pier Alda Ferrari

See Also

`missing.gen`

Examples

```
sigma <- matrix(0.4, 4, 4)
diag(sigma) <- 1
x0 <- rmvnorm(n=100, mean=rep(0, 4), sigma=sigma)
x0 # complete matrix
x <- missing.gen0(x0, 50)
x # matrix with 50 missing values
```

Index