Package: missForest (via r-universe)

September 17, 2024

Type Package

Title Nonparametric Missing Value Imputation using Random Forest

Version 1.5

Date 2022-04-14

Author Daniel J. Stekhoven <stekhoven@stat.math.ethz.ch>

Maintainer Daniel J. Stekhoven <stekhoven@stat.math.ethz.ch>

Imports randomForest,foreach,itertools,iterators,doRNG

Suggests doParallel

Description The function 'missForest' in this package is used to impute missing values particularly in the case of mixed-type data. It uses a random forest trained on the observed values of a data matrix to predict the missing values. It can be used to impute continuous and/or categorical data including complex interactions and non-linear relations. It yields an out-of-bag (OOB) imputation error estimate without the need of a test set or elaborate cross-validation. It can be run in parallel to save computation time.

License GPL (>= 2)

URL https://www.r-project.org, https://github.com/stekhoven/missForest

Repository https://stekhoven.r-universe.dev

RemoteUrl https://github.com/stekhoven/missforest

RemoteRef HEAD

RemoteSha 5e5e26035eb43b37720af5b962c9590d6c7389cc

Contents

missForest-package 2
missForest 2
mixError 6
nrmse 7
prodNA 8
varClass 8

Index **10**

`missForest-package` *Nonparametric Missing Value Imputation using Random Forest*

Description

'missForest' is used to impute missing values particularly in the case of mixed-type data. It can be used to impute continuous and/or categorical data including complex interactions and nonlinear relations. It yields an out-of-bag (OOB) imputation error estimate. Moreover, it can be run parallel to save computation time.

Details

Package:	missForest
Type:	Package
Version:	1.4
Date:	2013-12-31
License:	GPL (>= 2)
LazyLoad:	yes

The main function of the package is `missForest` implementing the nonparametric missing value imputation. See `missForest` for more details.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

References

Stekhoven, D.J. and Buehlmann, P. (2012), 'MissForest - nonparametric missing value imputation for mixed-type data', Bioinformatics, 28(1) 2012, 112-118, doi: 10.1093/bioinformatics/btr597

`missForest` *Nonparametric Missing Value Imputation using Random Forest*

Description

'missForest' is used to impute missing values particularly in the case of mixed-type data. It can be used to impute continuous and/or categorical data including complex interactions and nonlinear relations. It yields an out-of-bag (OOB) imputation error estimate. Moreover, it can be run parallel to save computation time.

Usage

```
missForest(xmis, maxiter = 10, ntree = 100, variablewise = FALSE,
                       decreasing = FALSE, verbose = FALSE,
                       mtry = floor(sqrt(ncol(xmis))), replace = TRUE,
                       classwt = NULL, cutoff = NULL, strata = NULL,
                       sampsize = NULL, nodesize = NULL, maxnodes = NULL,
                      xtrue = NA, parallelize = c('no', 'variables', 'forests'))
```

Arguments

xmis	a data matrix with missing values. The columns correspond to the variables and the rows to the observations.
maxiter	maximum number of iterations to be performed given the stopping criterion is not met beforehand.
ntree	number of trees to grow in each forest.
variablewise	logical. If 'TRUE' the OOB error is returned for each variable separately. This can be useful as a reliability check for the imputed variables w.r.t. to a subsequent data analysis.
decreasing	logical. If 'FALSE' then the variables are sorted w.r.t. increasing amount of missing entries during computation.
verbose	logical. If 'TRUE' the user is supplied with additional output between iterations, i.e., estimated imputation error, runtime and if complete data matrix is supplied the true imputation error. See 'xtrue'.
mtry	number of variables randomly sampled at each split. This argument is directly supplied to the 'randomForest' function. Note that the default value is sqrt(p) for both categorical and continuous variables where p is the number of variables in 'xmis'.
replace	logical. If 'TRUE' bootstrap sampling (with replacements) is performed else subsampling (without replacements).
classwt	list of priors of the classes in the categorical variables. This is equivalent to the randomForest argument, however, the user has to set the priors for all categorical variables in the data set (for continuous variables set it 'NULL').
cutoff	list of class cutoffs for each categorical variable. Same as with 'classwt' (for continuous variables set it '1').
strata	list of (factor) variables used for stratified sampling. Same as with 'classwt' (for continuous variables set it 'NULL').
sampsize	list of size(s) of sample to draw. This is equivalent to the randomForest argument, however, the user has to set the sizes for all variables.
nodesize	minimum size of terminal nodes. Has to be a vector of length 2, with the first entry being the number for continuous variables and the second entry the number for categorical variables. Default is 1 for continuous and 5 for categorical variables.
maxnodes	maximum number of terminal nodes for trees in the forest.

`xtrue` optional. Complete data matrix. This can be supplied to test the performance. Upon providing the complete data matrix 'verbose' will show the true imputation error after each iteration and the output will also contain the final true imputation error.

`parallelize` should 'missForest' be run parallel. Default is 'no'. If 'variables' the data is split into pieces of the size equal to the number of cores registered in the parallel backend. If 'forests' the total number of trees in each random forests is split in the same way. Whether 'variables' or 'forests' is more suitable, depends on the data. See Details.

Details

After each iteration the difference between the previous and the new imputed data matrix is assessed for the continuous and categorical parts. The stopping criterion is defined such that the imputation process is stopped as soon as both differences have become larger once. In case of only one type of variable the computation stops as soon as the corresponding difference goes up for the first time. However, the imputation last performed where both differences went up is generally less accurate than the previous one. Therefore, whenever the computation stops due to the stopping criterion (and not due to 'maxiter') the before last imputation matrix is returned.

The normalized root mean squared error (NRMSE) is defined as:

$$\sqrt{\frac{mean((X_{true} - X_{imp})^2)}{var(X_{true})}}$$

where X_{true} the complete data matrix, X_{imp} the imputed data matrix and 'mean'/'var' being used as short notation for the empirical mean and variance computed over the continuous missing values only.

The proportion of falsely classified (PFC) is also computed over the categorical missing values only.

For feasibility reasons 'ntree', 'mtry', 'nodesize' and 'maxnodes' can be chosen smaller. The number of trees can be chosen fairly small since growing many forests (e.g. p forests in each iteration) all observations get predicted a few times. The runtime behaves linear with 'ntree'. In case of high-dimensional data we recommend using a small 'mtry' (e.g. 100 should work) to obtain an appropriate imputation result within a feasible amount of time.

Using an appropriate backend 'missForest' can be run parallel. There are two possible ways to do this. One way is to create the random forest object in parallel (parallelize = "forests"). This is most useful if a single forest object takes long to compute and there are not many variables in the data. The second way is to compute multiple random forest classifiers parallel on different variables (parallelize = "variables"). This is most useful if the data contains many variables and computing the random forests is not taking too long. For details on how to register a parallel backend see for instance the documentation of 'doParallel').

See the vignette for further examples on how to use missForest.

I thank Steve Weston for his input regarding parallel computation of 'missForest'.

Value

`ximp` imputed data matrix of same type as 'xmis'.

OOBerror	estimated OOB imputation error. For the set of continuous variables in 'xmis' the NRMSE and for the set of categorical variables the proportion of falsely classified entries is returned. See Details for the exact definition of these error measures. If 'variablewise' is set to 'TRUE' then this will be a vector of length 'p' where 'p' is the number of variables and the entries will be the OOB error for each variable separately.
error	true imputation error. This is only available if 'xtrue' was supplied. The error measures are the same as for 'OOBerror'.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

References

Stekhoven, D.J. and Buehlmann, P. (2012), 'MissForest - nonparametric missing value imputation for mixed-type data', Bioinformatics, 28(1) 2012, 112-118, doi: 10.1093/bioinformatics/btr597

See Also

mixError, prodNA, randomForest

Examples

```
## Nonparametric missing value imputation on mixed-type data:
data(iris)
summary(iris)

## The data contains four continuous and one categorical variable.

## Artificially produce missing values using the 'prodNA' function:
set.seed(81)
iris.mis <- prodNA(iris, noNA = 0.2)
summary(iris.mis)

## Impute missing values providing the complete matrix for
## illustration. Use 'verbose' to see what happens between iterations:
iris.imp <- missForest(iris.mis, xtrue = iris, verbose = TRUE)

## The imputation is finished after five iterations having a final
## true NRMSE of 0.143 and a PFC of 0.036. The estimated final NRMSE
## is 0.157 and the PFC is 0.025 (see Details for the reason taking
## iteration 4 instead of iteration 5 as final value).

## The final results can be accessed directly. The estimated error:
iris.imp$OOBerror

## The true imputation error (if available):
iris.imp$error

## And of course the imputed data matrix (do not run this):
## iris.imp$ximp
```

mixError *Compute Imputation Error for Mixed-type Data*

Description

'mixError' is used to calculate the imputation error particularly in the case of mixed-type data. Given the complete data matrix and the data matrix containing the missing values the normalized root mean squared error for the continuous and the proportion of falsely classified entries for the categorical variables are computed.

Usage

```
mixError(ximp, xmis, xtrue)
```

Arguments

ximp	imputed data matrix with variables in the columns and observations in the rows. Note there should not be any missing values.
xmis	data matrix with missing values.
xtrue	complete data matrix. Note there should not be any missing values.

Value

imputation error. In case of continuous variables only this is the normalized root mean squared error (NRMSE, see 'help(missForest)' for further details). In case of categorical variables onlty this is the proportion of falsely classified entries (PFC). In case of mixed-type variables both error measures are supplied.

Note

This function is internally used by missForest whenever a complete data matrix is supplied.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

See Also

missForest

Examples

```
## Compute imputation error for mixed-type data:
data(iris)

## Artificially produce missing values using the 'prodNA' function:
set.seed(81)
iris.mis <- prodNA(iris, noNA = 0.2)
```

```
## Impute missing values using 'missForest':
iris.imp <- missForest(iris.mis)

## Compute the true imputation error manually:
err.imp <- mixError(iris.imp$ximp, iris.mis, iris)
err.imp
```

---

nrmse *Normalized Root Mean Squared Error*

Description

'nrmse' computes the normalized root mean squared error for a given complete data matrix, imputed data matrix and the data matrix containing missing values.

Usage

```
nrmse(ximp, xmis, xtrue)
```

Arguments

ximp	imputed data matrix with variables in the columns and observations in the rows. Note there should not be any missing values.
xmis	data matrix with missing values.
xtrue	complete data matrix. Note there should not be any missing values.

Value

see Title.

Note

The NRMSE can only be computed for continuous data. For categorical or mixed-type data see mixError.

This function is internally used by mixError.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

References

Oba et al. (2003), 'A Bayesian missing value estimation method for gene expression profile data', Bioinformatics, 19(16), 2088-2096

See Also

mixError

prodNA *Introduce Missing Values Completely at Random*

Description

'prodNA' artificially introduces missing values. Entries in the given dataframe are deleted completely at random up to the specified amount.

Usage

```
prodNA(x, noNA = 0.1)
```

Arguments

x	dataframe subjected to missing value introduction.
noNA	proportion of missing values w.r.t. the number of entries of 'x'.

Value

dataframe with missing values.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

See Also

missForest

Examples

```
data(iris)
## Introduce 5% of missing values to the iris data set
iris.mis <- prodNA(iris, 0.05)
summary(iris.mis)
```

varClass *Extract Variable Types from a Dataframe*

Description

'varClass' returns the variable types of a dataframe. It is used internally in several functions of the 'missForest'-package.

Usage

```
varClass(x)
```

Arguments

x	data frame with variables in the columns.

Value

a vector of length p where p denotes the number of columns in 'x'. The entries are "numeric" for continuous variables and "factor" for categorical variables.

Note

This function is internally used by `missForest` and `mixError`.

Author(s)

Daniel J. Stekhoven, <stekhoven@stat.math.ethz.ch>

See Also

`missForest`, `mixError`, `nrmse`

Examples

```
data(iris)
varClass(iris)

## We have four continuous and one categorical variable.
```

Index

∗ **NA**
    missForest, 2
    missForest-package, 2
    mixError, 6
    prodNA, 8
∗ **classes**
    missForest, 2
    missForest-package, 2
    mixError, 6
    prodNA, 8
    varClass, 8
∗ **error**
    nrmse, 7
∗ **nonparametric**
    missForest, 2
    missForest-package, 2
∗ **package**
    missForest-package, 2

missForest, *2*, 2, *6*, *8*, *9*
missForest-package, 2
mixError, *5*, 6, *7*, *9*

nrmse, 7, *9*

prodNA, *5*, 8

randomForest, *5*

varClass, 8