

# Package ‘imputeMissings’

October 13, 2022

**Type** Package

**Title** Impute Missing Values in a Predictive Context

**Version** 0.0.3

**Date** 2016-09-5

**Imports** randomForest,stats

**Author** Matthijs Meire, Michel Ballings, Dirk Van den Poel

**Maintainer** Matthijs Meire <Matthijs.Meire@ugent.be>

**Description** Compute missing values on a training data set and impute them on a new data set. Current available options are median/mode and random forest.

**License** GPL (>= 2)

**LazyData** TRUE

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2016-09-06 10:04:45

## R topics documented:

compute . . . . .	1
impute . . . . .	3
<b>Index</b>	<b>5</b>

---

compute	<i>Compute the missing values to later impute them in another dataset</i>
---------	---

---

**Description**

When the median/mode method is used: character vectors and factors are imputed with the mode. Numeric and integer vectors are imputed with the median. When the random forest method is used predictors are first imputed with the median/mode and each variable is then predicted and imputed with that value. For predictive contexts there is a `compute` and an `impute` function. The former is used on a training set to learn the values (or random forest models) to impute (used to predict). The latter is used on both the training and new data to impute the values (or deploy the models) learned by the `compute` function.

**Usage**

```
compute(data, method = "median/mode", ...)
```

**Arguments**

<code>data</code>	A data frame with dummies or numeric variables. When <code>method=="median/mode"</code> columns can be of type "character". When <code>method="randomForest"</code> columns cannot be of type "character".
<code>method</code>	Either "median/mode" or "randomForest"
<code>...</code>	additional arguments for randomForest

**Value**

Values or models used for imputation

**Author(s)**

Matthijs Meire, Michel Ballings, Dirk Van den Poel, Maintainer: <Matthijs.Meire@UGent.be>

**See Also**

[impute](#)

**Examples**

```
#Compute the values on a training dataset and impute them on new data.
#This is very convenient in predictive contexts. For example:

#define training data
(train <- data.frame(v_int=as.integer(c(3,3,2,5,1,2,4,6)),
  v_num=as.numeric(c(4.1,NA,12.2,11,3.4,1.6,3.3,5.5)),
  v_fact=as.factor(c('one', 'two',NA,'two', 'two', 'one', 'two', 'two')),
  stringsAsFactors = FALSE))

#Compute values on train data
#randomForest method
values <- compute(train, method="randomForest")
#median/mode method
values2 <- compute(train)
```

```

#define new data
(newdata <- data.frame(v_int=as.integer(c(1,1,2,NA)),
                      v_num=as.numeric(c(1.1,NA,2.2,NA)),
                      v_fact=as.factor(c('one', 'one', 'one',NA)),
                      stringsAsFactors = FALSE))

#locate the NA's
is.na(newdata)
#how many missings per variable?
colSums(is.na(newdata))

#Impute on newdata
impute(newdata,object=values) #using randomForest values
impute(newdata,object=values2) #using median/mode values

#One can also impute directly in newdata without the compute step
impute(newdata)

#Flag parameter
impute(newdata,flag=TRUE)

```

---

impute

---

*Impute missing values with the median/mode or randomForest*


---

## Description

When the median/mode method is used: character vectors and factors are imputed with the mode. Numeric and integer vectors are imputed with the median. When the random forest method is used predictors are first imputed with the median/mode and each variable is then predicted and imputed with that value. For predictive contexts there is a `compute` and an `impute` function. The former is used on a training set to learn the values (or random forest models) to impute (used to predict). The latter is used on both the training and new data to impute the values (or deploy the models) learned by the `compute` function.

## Usage

```
impute(data, object = NULL, method = "median/mode", flag = FALSE)
```

## Arguments

<code>data</code>	A data frame with dummies or numeric variables. When <code>method=="median/mode"</code> columns can be of type "character". When <code>method="randomForest"</code> columns cannot be of type "character".
<code>object</code>	If <code>NULL</code> <code>impute</code> will call <code>compute</code> on the current dataset. Otherwise it will accept the output of a call to <code>compute</code>
<code>method</code>	Either "median/mode" or "randomForest". Only works if <code>object = NULL</code>
<code>flag</code>	Add dummy variables to indicate which rows have been imputed for each variable

**Value**

An imputed data frame.

**Author(s)**

Matthijs Meire, Michel Ballings, Dirk Van den Poel, Maintainer: <Matthijs.Meire@UGent.be>

**See Also**

[compute](#)

**Examples**

```
#Compute the values on a training dataset and impute them on new data.
#This is very convenient in predictive contexts. For example:

#define training data
(train <- data.frame(v_int=as.integer(c(3,3,2,5,1,2,4,6)),
                    v_num=as.numeric(c(4.1,NA,12.2,11,3.4,1.6,3.3,5.5)),
                    v_fact=as.factor(c('one', 'two', NA, 'two', 'two', 'one', 'two', 'two')),
                    stringsAsFactors = FALSE))

#Compute values on train data
#randomForest method
values <- compute(train, method="randomForest")
#median/mode method
values2 <- compute(train)

#define new data
(newdata <- data.frame(v_int=as.integer(c(1,1,2,NA)),
                     v_num=as.numeric(c(1.1,NA,2.2,NA)),
                     v_fact=as.factor(c('one', 'one', 'one', NA)),
                     stringsAsFactors = FALSE))

#locate the NA's
is.na(newdata)
#how many missings per variable?
colSums(is.na(newdata))

#Impute on newdata
impute(newdata,object=values) #using randomForest values
impute(newdata,object=values2) #using median/mode values

#One can also impute directly in newdata without the compute step
impute(newdata)

#Flag parameter
impute(newdata,flag=TRUE)
```

# Index

compute, 1, 4

impute, 2, 3