Package 'ClustImpute'

December 12, 2020

Type Package

Title K-means clustering with build-in missing data imputation

Version 0.1.6

Author Oliver Pfaffel

Maintainer Oliver Pfaffel <opfaffel@gmail.com>

Description This clustering algorithm deals with missing data via weights that are imposed on missings and successively increased. See the vignette for details.

License GPL-3

Encoding UTF-8

LazyData true

Imports ClusterR, copula, dplyr, magrittr, rlang

Suggests psych, ggplot2, knitr, rmarkdown, testthat (>= 2.1.0), tidyr, Hmisc, tictoc, spelling, corrplot, covr

VignetteBuilder knitr

RoxygenNote 7.1.0

Language en-US

NeedsCompilation no

Repository CRAN

Date/Publication 2020-12-12 15:30:42 UTC

R topics documented:

check_replace_dups *Check and replace duplicate (centroid) rows*

Description

Internal function of ClustImpute: check new centroids for duplicate rows and replace with random draws in this case.

Usage

```
check_replace_dups(centroids, X, seed)
```

Arguments

centroids	Matrix of centroids
X	Underlying data matrix (without missings)
seed	Seed used for random sampling

ClustImpute *K-means clustering with build-in missing data imputation*

Description

Clustering algorithm that produces a missing value imputation using on the go. The (local) imputation distribution is defined by the currently assigned cluster. The first draw is by random imputation.

Usage

```
ClustImpute(
  X,
  nr_cluster,
  nr_iter = 10,
  c_steps = 1,
  wf = default_wf,
  n_end = 10,
  seed_nr = 150519,
  assign_with_wf = TRUE
)
```

Arguments

X	Data frame with only numeric values or NAs
nr_cluster	Number of clusters
nr_iter	Iterations of procedure
c_steps	Number of clustering steps per iteration
wf	Weight function. linear up to n_end by default
n_end	Steps until convergence of weight function to 1
seed_nr	Number for set.seed()
assign_with_wf	Default is True. If set to False, then the weight function is only applied in the centroid computation, but ignored in the cluster assignment.

Value

complete_data Completed data without NAs

clusters For each row of complete_data, the associated cluster

centroids For each cluster, the coordinates of the centroids

imp_values_mean Mean of the imputed variables per draw

imp_values_sd Standard deviation of the imputed variables per draw

Examples

```
# Random Dataset
set.seed(739)
n <- 750 # numer of points
nr_other_vars <- 2
mat <- matrix(rnorm(nr_other_vars*n),n,nr_other_vars)
me<-4 # mean
x <- c(rnorm(n/3,me/2,1),rnorm(2*n/3,-me/2,1))
y <- c(rnorm(n/3,0,1),rnorm(n/3,me,1),rnorm(n/3,-me,1))
dat <- cbind(mat,x,y)
dat<- as.data.frame(scale(dat)) # scaling

# Create NAs
dat_with_miss <- miss_sim(dat,p=.1,seed_nr=120)

# Run ClustImpute
res <- ClustImpute(dat_with_miss,nr_cluster=3)

# Plot complete data set and cluster assignment
ggplot2::ggplot(res$complete_data,ggplot2::aes(x,y,color=factor(res$clusters))) +
ggplot2::geom_point()

# View centroids
res$centroids
```

default_wf *K-means clustering with build-in missing data imputation*

Description

Default weight function. One minus the return value is multiplied with missing(=imputed) values. It starts with 1 and goes to 0 at n_end.

Usage

```
default_wf(n, n_end = 10)
```

Arguments

n	current step
n_end	steps until convergence of weight function to 0

Value

value between 0 and 1

Examples

```
x <- 0:20
plot(x,1-default_wf(x))
```

miss_sim *Simulation of missings*

Description

Simulates missing at random using a normal copula to create correlations between the missing (type="MAR"). Missings appear in each column of the provided data frame with the same ratio.

Usage

```
miss_sim(dat, p = 0.2, type = "MAR", seed_nr = 123)
```

Arguments

dat	Data frame with only numeric values
p	Fraction of missings (for entire data frame)
type	Type of missingness. Either MCAR (=missing completely at random) or MAR (=missing at random)
seed_nr	Number for set.seed()

Value

data frame with only numeric values and NAs

Examples

```
data(cars)
cars_with_missings <- miss_sim(cars,p = .2,seed_nr = 4)
summary(cars_with_missings)
```

`predict.kmeans_ClustImpute` *Prediction method*

Description

Prediction method

Usage

```
## S3 method for class 'kmeans_ClustImpute'
predict(object, newdata, ...)
```

Arguments

`object`	Object of class kmeans_ClustImpute
`newdata`	Data frame
`...`	additional arguments affecting the predictions produced - not currently used

Value

integer value (cluster assignment)

Examples

```
# Random Dataset
set.seed(739)
n <- 750 # numer of points
nr_other_vars <- 2
mat <- matrix(rnorm(nr_other_vars*n),n,nr_other_vars)
me<-4 # mean
x <- c(rnorm(n/3,me/2,1),rnorm(2*n/3,-me/2,1))
y <- c(rnorm(n/3,0,1),rnorm(n/3,me,1),rnorm(n/3,-me,1))
dat <- cbind(mat,x,y)
dat<- as.data.frame(scale(dat)) # scaling

# Create NAs
dat_with_miss <- miss_sim(dat,p=.1,seed_nr=120)
```

```
res <- ClustImpute(dat_with_miss,nr_cluster=3)
predict(res,newdata=dat[1,])
```

`var_reduction` *Reduction of variance*

Description

Computes one minus the ratio of the sum of all within cluster variances by the overall variance

Usage

```
var_reduction(clusterObj)
```

Arguments

`clusterObj`	Object of class kmeans_ClustImpute

Value

integer value typically between 0 and 1

Examples

```
# Random Dataset
set.seed(739)
n <- 750 # numer of points
nr_other_vars <- 2
mat <- matrix(rnorm(nr_other_vars*n),n,nr_other_vars)
me<-4 # mean
x <- c(rnorm(n/3,me/2,1),rnorm(2*n/3,-me/2,1))
y <- c(rnorm(n/3,0,1),rnorm(n/3,me,1),rnorm(n/3,-me,1))
dat <- cbind(mat,x,y)
dat<- as.data.frame(scale(dat)) # scaling

# Create NAs
dat_with_miss <- miss_sim(dat,p=.1,seed_nr=120)

res <- ClustImpute(dat_with_miss,nr_cluster=3)
var_reduction(res)
```

Index