Package ‘hdImpute’

August 7, 2023

Type Package

Title A Batch Process for High Dimensional Imputation

Version 0.2.1

BugReports https://github.com/pdwaggoner/hdImpute/issues

Maintainer Philip Waggoner <philip.waggoner@gmail.com>

Description A correlation-based batch process for fast, accurate imputation for
high dimensional missing data problems via chained random forests.
See Waggoner (2023) <doi:10.1007/s00180-023-01325-9> for more on 'hdImpute',
Stekhoven and Bühlmann (2012) <doi:10.1093/bioinformatics/btr597> for more on 'missForest',
and Mayer (2022) <https://github.com/mayer79/missRanger> for more on 'missRanger'.

License MIT + file LICENSE

Encoding UTF-8

Imports missRanger, plyr, purrr, magrittr, tibble, dplyr, tidyselect,
tidyr, cli

Suggests testthat (>= 3.0.0), knitr, rmarkdown, usethis, missForest,
tidyverse

VignetteBuilder knitr

RoxygenNote 7.2.3

Config/testthat/edition 3

URL https://github.com/pdwaggoner/hdImpute

NeedsCompilation no

Author Philip Waggoner [aut, cre]

Repository CRAN

Date/Publication 2023-08-07 21:20:02 UTC

R topics documented:

check_feature_na *Find features with (specified amount of) missingness*

Description

Find features with (specified amount of) missingness

Usage

```
check_feature_na(data, threshold)
```

Arguments

`data`	A data frame or tibble.
`threshold`	Missingness threshold in a given column/feature as a proportion bounded between 0 and 1. Default set to sensitive level at 1e-04.

Value

A vector of column/feature names that contain missingness greater than `threshold`.

Examples

```
## Not run:
check_feature_na(data = any_data_frame, threshold = 1e-04)

## End(Not run)
```

check_row_na *Find number of and which rows contain any missingness*

Description

Find number of and which rows contain any missingness

Usage

```
check_row_na(data, which)
```

Arguments

`data` A data frame or tibble.

`which` Logical. Should a list be returned with the row numbers corresponding to each row with missingness? Default set to FALSE.

Value

Either an integer value corresponding to the number of rows in `data` with any missingness (if `which = FALSE`), or a tibble containing: 1) number of rows in `data` with any missingness, and 2) a list of which rows/row numbers contain missingness (if `which = TRUE`).

Examples

```
## Not run:
check_row_na(data = any_data_frame, which = FALSE)

## End(Not run)
```

`feature_cor` *High dimensional imputation via batch processed chained random forests Build correlation matrix*

Description

High dimensional imputation via batch processed chained random forests Build correlation matrix

Usage

```
feature_cor(data, return_cor)
```

Arguments

`data` A data frame or tibble.

`return_cor` Logical. Should the correlation matrix be printed? Default set to FALSE.

Value

A cross-feature correlation matrix

References

Waggoner, P. D. (2023). A batch process for high dimensional imputation. Computational Statistics, 1-22. doi: <10.1007/s00180-023-01325-9>

van Buuren S, Groothuis-Oudshoorn K (2011). "mice: Multivariate Imputation by Chained Equations in R." Journal of Statistical Software, 45(3), 1-67. doi: <10.18637/jss.v045.i03>

Examples

```
## Not run:
feature_cor(data = data, return_cor = FALSE)

## End(Not run)
```

flatten_mat *Flatten and arrange cor matrix to be df*

Description

Flatten and arrange cor matrix to be df

Usage

```
flatten_mat(cor_mat, return_mat)
```

Arguments

`cor_mat`	A correlation matrix output from running `feature_cor()`
`return_mat`	Logical. Should the flattened matrix be printed? Default set to FALSE.

Value

A vector of correlation-based ranked features

Examples

```
## Not run:
flatten_mat(cor_mat = cor_mat, return_mat = FALSE)

## End(Not run)
```

hdImpute *Complete hdImpute process: correlation matrix, flatten, rank, create batches, impute, join*

Description

Complete hdImpute process: correlation matrix, flatten, rank, create batches, impute, join

Usage

```
hdImpute(data, batch, pmm_k, n_trees, seed, save)
```

Arguments

`data` Original data frame or tibble (with missing values)

`batch` Numeric. Batch size.

`pmm_k` Integer. Number of neighbors considered in imputation. Default set at 5.

`n_trees` Integer. Number of trees used in imputation. Default set at 15.

`seed` Integer. Seed to be set for reproducibility.

`save` Should the list of individual imputed batches be saved as .rds file to working directory? Default set to FALSE.

Details

Step 1. group data by dividing the `row_number()` by batch size (`batch`, number of batches set by user) using integer division. Step 2. pass through `group_split()` to return a list. Step 3. impute each batch individually and time. Step 4. generate completed (unlisted/joined) imputed data frame

Value

A completed, imputed data set

References

Waggoner, P. D. (2023). A batch process for high dimensional imputation. Computational Statistics, 1-22. doi: <10.1007/s00180-023-01325-9>

Stekhoven, D. J., & Bühlmann, P. (2012). MissForest—non-parametric missing value imputation for mixed-type data. Bioinformatics, 28(1), 112-118. doi: <10.1093/bioinformatics/btr597>

Examples

```
## Not run:
impute_batches(data = data,
batch = 2,  pmm_k = 5, n_trees = 15,
seed = 123, save = FALSE)

## End(Not run)
```

`impute_batches` *Impute batches and return completed data frame*

Description

Impute batches and return completed data frame

Usage

```
impute_batches(data, features, batch, pmm_k, n_trees, seed, save)
```

Arguments

`data` Original data frame or tibble (with missing values)

`features` Correlation-based vector of ranked features output from running `flatten_mat()`

`batch` Numeric. Batch size.

`pmm_k` Integer. Number of neighbors considered in imputation. Default at 5.

`n_trees` Integer. Number of trees used in imputation. Default at 15.

`seed` Integer. Seed to be set for reproducibility.

`save` Should the list of individual imputed batches be saved as .rds file to working directory? Default set to FALSE.

Details

Step 1. group data by dividing the `row_number()` by batch size (`batch`, number of batches set by user) using integer division. Step 2. pass through `group_split()` to return a list. Step 3. impute each batch individually and time. Step 4. generate completed (unlisted/joined) imputed data frame

Value

A completed, imputed data set

References

Waggoner, P. D. (2023). A batch process for high dimensional imputation. Computational Statistics, 1-22. doi: <10.1007/s00180-023-01325-9>

Stekhoven, D. J., & Bühlmann, P. (2012). MissForest—non-parametric missing value imputation for mixed-type data. Bioinformatics, 28(1), 112-118. doi: <10.1093/bioinformatics/btr597>

Examples

```
## Not run:
impute_batches(data = data, features = flat_mat,
batch = 2,  pmm_k = 5, n_trees = 15, seed = 123,
save = FALSE)

## End(Not run)
```

`mad` *Compute variable-wise mean absolute differences (MAD) between original and imputed dataframes.*

Description

Compute variable-wise mean absolute differences (MAD) between original and imputed dataframes.

Usage

```
mad(original, imputed, round)
```

Arguments

`original` A data frame or tibble with original values.

`imputed` A data frame or tibble that has been imputed/completed.

`round` Integer. Number of places to round MAD scores. Default set to 3.

Value

‘mad_scores‘ as ‘p‘ x 2 tibble. One row for each variable in `original`, from 1 to ‘p‘. Two columns: first is variable names (‘var‘) and second is associated MAD score (‘mad‘) as percentages for each variable.

Examples

```
## Not run:
mad(original = original_data, imputed = imputed_data, round = 3)

## End(Not run)
```

Index