

Package ‘nanny’

June 9, 2020

Type Package

Title High-Level Data Analysis and Manipulation in 'tidyverse' Style

Version 0.1.8

Maintainer Stefano Mangiola <mangiolastefano@gmail.com>

Description Includes wrapper functions for the main high-level data analysis and manipulations, such as cluster, dimensionality reduction, redundancy elimination, identify variable elements. It operates on tidy data frames with element, feature and value column.

License GPL-3

Depends R (>= 3.6.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports lifecycle,

magrittr,

tibble,

dplyr,

tidyselect,

purrr,

rlang,

tidyr,

methods,

lme4,

stats,

utils,

knitr,

widyr,

Rtsne,

gtools

RdMacros lifecycle

Suggests testthat,

Seurat,

limma,

GGally,

ggplot2

VignetteBuilder knitr

Biarch true

biocViews AssayDomain, Infrastructure

R topics documented:

as_matrix	2
cluster_elements	3
combine_nest	5
fill_missing	6
impute_missing	7
keep_variable	8
lower_triangular	9
nest_subset	10
nest_subset,tbl-method	11
permute_nest	11
reduce_dimensions	12
remove_redundancy	14
rotate_dimensions	16
subset	18

Index	19
--------------	-----------

as_matrix	<i>Get matrix from tibble</i>
-----------	-------------------------------

Description

Get matrix from tibble

as_matrix

as_matrix

Usage

```
as_matrix(.data, rownames = NULL, do_check = TRUE, sep_rownames = "__")

## S4 method for signature 'spec_tbl_df'
as_matrix(.data, rownames = NULL, do_check = TRUE, sep_rownames = "__")

## S4 method for signature 'tbl_df'
as_matrix(.data, rownames = NULL, do_check = TRUE, sep_rownames = "__")
```

Arguments

.data	A tibble
rownames	A character string of the rownames
do_check	A boolean
sep_rownames	A character with which multiple columns are united if rownames is a column array (e.g., rownames = c(col1, col2))

Value

A matrix

A 'tbl' with filled abundance

A 'tbl' with filled abundance

Examples

```
library(dplyr)
library(tidyr)
select(mtcars_tidy, car_model, feature, value) %>%
spread(feature, value) %>%
as_matrix(rownames = car_model)
```

cluster_elements *Get clusters of elements (e.g., elements or features)*

Description

cluster_elements() takes as input a 'tbl' formatted as |<element>|<feature>|<value>|<...>| and identify clusters in the data.

Usage

```
cluster_elements(
  .data,
  .element,
  .feature,
  .value,
  method,
  of_elements = TRUE,
  transform = NULL,
  action = "add",
  ...
)

## S4 method for signature 'spec_tbl_df'
cluster_elements(
  .data,
  .element,
  .feature,
  .value,
  method,
  of_elements = TRUE,
  transform = NULL,
  action = "add",
  ...
)

## S4 method for signature 'tbl_df'
cluster_elements(
  .data,
  .element,
  .feature,
  .value,
  method,
```

```

    of_elements = TRUE,
    transform = NULL,
    action = "add",
    ...
  )

```

Arguments

<code>.data</code>	A 'tbl' formatted as <element> <feature> <value> <...>
<code>.element</code>	The name of the element column (normally elements).
<code>.feature</code>	The name of the feature column (normally features)
<code>.value</code>	The name of the column including the numerical value the clustering is based on (normally feature value)
<code>method</code>	A character string. The cluster algorithm to use, ay the moment k-means is the only algorithm included.
<code>of_elements</code>	A boolean. In case the input is a nanny object, it indicates Whether the element column will be element or feature column
<code>transform</code>	A function to use to tranforma the data internalli (e.g., log1p)
<code>action</code>	A character string. Whether to join the new information to the input tbl (add), or just get the non-redundant tbl with the new information (get).
<code>...</code>	Further parameters passed to the function kmeans

Details

Maturing

identifies clusters in the data, normally of elements. This function returns a tibble with additional columns for the cluster annotation. At the moment only k-means clustering is supported, the plan is to introduce more clustering methods.

Value

A tbl object with additional columns with cluster labels

A tbl object with additional columns with cluster labels

A tbl object with additional columns with cluster labels

Examples

```
cluster_elements(mtcars_tidy, car_model, feature, value, method="kmeans",centers = 2)
```

`combine_nest`*Combine columns and nest data for each permutation*

Description

`combine_nest()` takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | and returns a 'tbl' with data nested for each combination. The package used in the backend is `gtools` (Gregory R. Warnes, Ben Bolker, and Thomas Lumley, 2020).

Usage

```
combine_nest(.data, .names_from, .values_from)

## S4 method for signature 'spec_tbl_df'
combine_nest(.data, .names_from, .values_from)

## S4 method for signature 'tbl_df'
combine_nest(.data, .names_from, .values_from)
```

Arguments

`.data` A 'tbl' formatted as | <element> | <feature> | <value> | <...> |
`.names_from` The columns to build the permutations on (e.g., `c(col1, col2)`)
`.values_from` The columns to be nested for each permutation (e.g., `c(col3, col4, col5)`)

Details

Maturing

...

Value

A nested 'tbl'
A 'tbl' with filled abundance
A 'tbl' with filled abundance

Examples

```
combine_nest(mtcars_tidy, car_model, c(feature, value))
```

fill_missing

Fill feature value if missing from element-feature pairs

Description

fill_missing() takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | and returns a 'tbl' with an additional adjusted value column. This method uses scaled counts if present.

Usage

```
fill_missing(.data, .element, .feature, .value, fill_with)

## S4 method for signature 'spec_tbl_df'
fill_missing(.data, .element, .feature, .value, fill_with)

## S4 method for signature 'tbl_df'
fill_missing(.data, .element, .feature, .value, fill_with)
```

Arguments

.data	A 'tbl' formatted as <element> <feature> <value> <...>
.element	The name of the element column
.feature	The name of the feature/gene column
.value	The name of the feature/gene value column
fill_with	A numerical value with which fill the missing data points

Details

Maturing

This function fills the value of missing element-feature pair using the median of the element group defined by the formula

Value

A 'tbl' non-sparse value
A 'tbl' with filled abundance
A 'tbl' with filled abundance

Examples

```
fill_missing(mtcars_tidy, car_model, feature, value, fill_with = 0)
```

impute_missing *Impute feature value if missing from element-feature pairs*

Description

impute_missing() takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | and returns a 'tbl' with an additional adjusted value column. This method uses scaled counts if present.

Usage

```
impute_missing(.data, .element, .feature, .value, .formula)

## S4 method for signature 'spec_tbl_df'
impute_missing(.data, .element, .feature, .value, .formula)

## S4 method for signature 'tbl_df'
impute_missing(.data, .element, .feature, .value, .formula)
```

Arguments

.data	A 'tbl' formatted as <element> <feature> <value> <...>
.element	The name of the element column
.feature	The name of the feature/gene column
.value	The name of the feature/gene value column
.formula	A formula with no response variable, representing the desired linear model where the first covariate is the factor of interest and the second covariate is the unwanted variation (of the kind ~ factor_of_intrest + batch)

Details

Maturing

This function imputes the value of missing element-feature pair using the median of the element group defined by the formula

Value

A 'tbl' non-sparse value
 A 'tbl' with imputed abundance
 A 'tbl' with imputed abundance

Examples

```
impute_missing(mtcars_tidy, car_model, feature, value, ~1)
```

keep_variable	<i>Keep top variable features across elements</i>
---------------	---

Description

keep_variable() takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | and returns a 'tbl' with the filtered most variable features. The formula used is from limma::plotMDS (Robinson et al., 2010, <doi:10.1093/bioinformatics/btp616>)

Usage

```
keep_variable(.data, .element, .feature, .value, top = Inf, transform = NULL)

## S4 method for signature 'spec_tbl_df'
keep_variable(.data, .element, .feature, .value, top = Inf, transform = NULL)

## S4 method for signature 'tbl_df'
keep_variable(.data, .element, .feature, .value, top = Inf, transform = NULL)
```

Arguments

.data	A 'tbl'
.element	A character name of the element column
.feature	A character name of the transcript/gene column
.value	A character name of the read count column
top	An integer. How many top genes to select
transform	A function to use to transform the data internally (e.g., log1p)

Details

Maturing

...

Value

A 'tbl' with filtered features
 A 'tbl' with filled abundance
 A 'tbl' with filled abundance

Examples

```
keep_variable(mtcars_tidy, car_model, feature, value, top=10)
```

lower_triangular *Keep rows corresponding of a lower triangular matrix built from two columns*

Description

lower_triangular() takes as input a 'tbl' formatted as |<element>|<feature>|<value>|<...>| and returns a filtered 'tbl'

Usage

```
lower_triangular(.data, .col1, .col2, .value)

## S4 method for signature 'spec_tbl_df'
lower_triangular(.data, .col1, .col2, .value)

## S4 method for signature 'tbl_df'
lower_triangular(.data, .col1, .col2, .value)
```

Arguments

.data	A 'tbl'
.col1	A column name
.col2	A column name
.value	A column names of the value column

Details

Maturing

...

Value

A 'tbl' with filtered rows
 A 'tbl' with filled abundance
 A 'tbl' with filled abundance

Examples

```
library(dplyr)
library(purrr)
library(tidyr)

mtcars_tidy_permuted =
  mtcars_tidy %>%
  filter(feature == "mpg") %>%
  head(5) %>%
  permute_nest(car_model, c(feature, value))

mtcars_tidy_permuted %>%
```

```
# Summarise mpg
mutate(data = map(data, ~ .x %>% summarise(mean(value)))) %>%
unnest(data) %>%

# Lower triangular
lower_triangular(car_model_1, car_model_2, `mean(value)`)
```

nest_subset

Nest according to selected-column-wise information

Description

nest_subset() takes as input a ‘tbl’ and returns a nested ‘tbl’ according to only selected-column-related columns

Usage

```
nest_subset(.data, ..., .names_sep = NULL)

## S4 method for signature 'spec_tbl_df'
nest_subset(.data, ..., .names_sep = NULL)

## S4 method for signature 'tbl_df'
nest_subset(.data, ..., .names_sep = NULL)
```

Arguments

.data	A ‘tbl’
...	The name of the columns of interest
.names_sep	Deprecated by tidyr

Details

Maturing

This function extracts only selected-column-related information for downstream analysis (e.g., visualisation). It is disruptive in the sense that it cannot be passed anymore to nanny function.

Value

A ‘tbl’ object
A ‘tbl’ object
A ‘tbl’ object

Examples

```
nest_subset(mtcars_tidy, data = -car_model)
```

```
nest_subset, tbl-method
      nest_subset
```

Description

nest_subset

Usage

```
## S4 method for signature 'tbl'
nest_subset(.data, ..., .names_sep = NULL)
```

Arguments

.data	A 'tbl'
...	The name of the columns of interest
.names_sep	Deprecated by tidyr

Value

A 'tbl' object

```
permute_nest      Permute columns and nest data for each permutation
```

Description

permute_nest() takes as input a 'tbl' formatted as |<element>|<feature>|<value>|<...>| and returns a 'tbl' with data nested for each permutation. The package used in the backend is gtools (Gregory R. Warnes, Ben Bolker, and Thomas Lumley, 2020)

Usage

```
permute_nest(.data, .names_from, .values_from)

## S4 method for signature 'spec_tbl_df'
permute_nest(.data, .names_from, .values_from)

## S4 method for signature 'tbl_df'
permute_nest(.data, .names_from, .values_from)
```

Arguments

.data	A 'tbl' formatted as <element> <feature> <value> <...>
.names_from	The columns to build the permutations on (e.g., c(col1, col2))
.values_from	The columns to be nested for each permutation (e.g., c(col3, col4, col5))

Details**Maturing**

...

Value

A nested 'tbl'

A 'tbl' with filled abundance

A 'tbl' with filled abundance

Examples

```
permute_nest(mtcars_tidy, car_model, c(feature, value))
```

reduce_dimensions *Dimension reduction of the feature value data*

Description

reduce_dimensions() takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | and calculates the reduced dimensional space of the feature value. The functions available are PCA, MDS (Robinson et al., 2010, <doi:10.1093/bioinformatics/btp616>), tSNE (Laurens van der Maaten, 2009)

Usage

```
reduce_dimensions(
  .data,
  .element,
  .feature,
  .value,
  method,
  .dims = 2,
  top = Inf,
  of_elements = TRUE,
  transform = NULL,
  scale = TRUE,
  action = "add",
  ...
)

## S4 method for signature 'spec_tbl_df'
reduce_dimensions(
  .data,
  .element,
  .feature,
  .value,
  method,
```

```

    .dims = 2,
    top = Inf,
    of_elements = TRUE,
    transform = NULL,
    scale = TRUE,
    action = "add",
    ...
)

## S4 method for signature 'tbl_df'
reduce_dimensions(
  .data,
  .element,
  .feature,
  .value,
  method,
  .dims = 2,
  top = Inf,
  of_elements = TRUE,
  transform = NULL,
  scale = TRUE,
  action = "add",
  ...
)

```

Arguments

<code>.data</code>	A 'tbl' formatted as <element> <feature> <value> <...>
<code>.element</code>	The name of the element column (normally elements).
<code>.feature</code>	The name of the feature column (normally features)
<code>.value</code>	The name of the column including the numerical value the clustering is based on (normally feature value)
<code>method</code>	A character string. The dimension reduction algorithm to use (PCA, MDS, tSNE).
<code>.dims</code>	A list of integer vectors corresponding to principal components of interest (e.g., list(1:2, 3:4, 5:6))
<code>top</code>	An integer. How many top genes to select for dimensionality reduction
<code>of_elements</code>	A boolean. In case the input is a nanny object, it indicates Whether the element column will be element or feature column
<code>transform</code>	A function to use to transform the data internally (e.g., log1p)
<code>scale</code>	A boolean for method="PCA", this will be passed to the 'prcomp' function. It is not included in the ... argument because although the default for 'prcomp' is FALSE, it is advisable to set it as TRUE.
<code>action</code>	A character string. Whether to join the new information to the input tbl (add), or just get the non-redundant tbl with the new information (get).
<code>...</code>	Further parameters passed to the function prcomp if you choose method="PCA" or Rtsne if you choose method="tSNE"

Details**Maturing**

This function reduces the dimensions of the feature values. It can use multi-dimensional scaling (MDS) of principal component analysis (PCA).

Value

A tbl object with additional columns for the reduced dimensions

A tbl object with additional columns for the reduced dimensions

A tbl object with additional columns for the reduced dimensions

Examples

```
reduce_dimensions(mtcars_tidy, car_model, feature, value, method="PCA")
reduce_dimensions(mtcars_tidy, car_model, feature, value, method="MDS")
reduce_dimensions(mtcars_tidy, car_model, feature, value, method="tSNE")
```

`remove_redundancy` *Drop redundant elements (e.g., elements) for which feature (e.g., feature/gene) abundances are correlated*

Description

`remove_redundancy()` takes as input a 'tbl' formatted as | <element> | <feature> | <value> | <...> | for correlation method, and returns a 'tbl' with dropped elements (e.g., elements). The backend function used is `widyr::pairwise_cor` (David Robinson, 2020)

Usage

```
remove_redundancy(
  .data,
  .element,
  .feature,
  .value,
  of_elements = TRUE,
  correlation_threshold = 0.9,
  top = Inf,
  transform = NULL
)

## S4 method for signature 'spec_tbl_df'
remove_redundancy(
  .data,
  .element,
```

```

    .feature,
    .value,
    of_elements = TRUE,
    correlation_threshold = 0.9,
    top = Inf,
    transform = NULL
  )

## S4 method for signature 'tbl_df'
remove_redundancy(
  .data,
  .element,
  .feature,
  .value,
  of_elements = TRUE,
  correlation_threshold = 0.9,
  top = Inf,
  transform = NULL
)

```

Arguments

<code>.data</code>	A 'tbl' formatted as <element> <feature> <value> <...>
<code>.element</code>	The name of the element column (normally elements).
<code>.feature</code>	The name of the feature column (normally features)
<code>.value</code>	The name of the column including the numerical value the clustering is based on (normally feature value)
<code>of_elements</code>	A boolean. In case the input is a nanny object, it indicates Whether the element column will be element or feature column
<code>correlation_threshold</code>	A real number between 0 and 1. For correlation based calculation.
<code>top</code>	An integer. How many top genes to select for correlation based method
<code>transform</code>	A function to use to tranforma the data internalli (e.g., log1p)

Details

Maturing

This function removes redundant elements from the original data set (e.g., elements or features). For example, if we want to define cell-type specific signatures with low element redundancy. This function returns a tibble with dropped recundant elements (e.g., elements). Two redundancy estimation approaches are supported: (i) removal of highly correlated clusters of elements (keeping a representative) with `method="correlation"`; (ii) removal of most proximal element pairs in a reduced dimensional space.

Value

A tbl object with with dropped recundant elements (e.g., elements).
 A tbl object with with dropped recundant elements (e.g., elements).
 A tbl object with with dropped recundant elements (e.g., elements).

Examples

```
remove_redundancy(mtcars_tidy, car_model, feature, value)
```

```
rotate_dimensions Rotate two dimensions (e.g., principal components) of an arbitrary angle
```

Description

rotate_dimensions() takes as input a 'tbl' formatted as | <DIMENSION 1> | <DIMENSION 2> | <...> | and calculates the rotated dimensional space of the feature value.

Usage

```
rotate_dimensions(
  .data,
  dimension_1_column,
  dimension_2_column,
  rotation_degrees,
  .element,
  of_elements = TRUE,
  dimension_1_column_rotated = NULL,
  dimension_2_column_rotated = NULL,
  action = "add"
)

## S4 method for signature 'spec_tbl_df'
rotate_dimensions(
  .data,
  dimension_1_column,
  dimension_2_column,
  rotation_degrees,
  .element,
  of_elements = TRUE,
  dimension_1_column_rotated = NULL,
  dimension_2_column_rotated = NULL,
  action = "add"
)

## S4 method for signature 'tbl_df'
rotate_dimensions(
  .data,
  dimension_1_column,
  dimension_2_column,
  rotation_degrees,
  .element,
```



```

    of_elements = TRUE,
    dimension_1_column_rotated = NULL,
    dimension_2_column_rotated = NULL,
    action = "add"
  )

```

Arguments

`.data` A 'tbl' formatted as | <element> | <feature> | <value> | <...> |

`dimension_1_column` A character string. The column of the dimension 1

`dimension_2_column` A character string. The column of the dimension 2

`rotation_degrees` A real number between 0 and 360

`.element` The name of the element column (normally elements).

`of_elements` A boolean. In case the input is a nanny object, it indicates Whether the element column will be element or feature column

`dimension_1_column_rotated` A character string. The column of the rotated dimension 1 (optional)

`dimension_2_column_rotated` A character string. The column of the rotated dimension 2 (optional)

`action` A character string. Whether to join the new information to the input tbl (add), or just get the non-redundant tbl with the new information (get).

Details

Maturing

This function to rotate two dimensions such as the reduced dimensions.

Value

A tbl object with additional columns for the reduced dimensions. additional columns for the rotated dimensions. The rotated dimensions will be added to the original data set as '<NAME OF DIMENSION> rotated <ANGLE>' by default, or as specified in the input arguments.

A tbl object with additional columns for the reduced dimensions. additional columns for the rotated dimensions. The rotated dimensions will be added to the original data set as '<NAME OF DIMENSION> rotated <ANGLE>' by default, or as specified in the input arguments.

A tbl object with additional columns for the reduced dimensions. additional columns for the rotated dimensions. The rotated dimensions will be added to the original data set as '<NAME OF DIMENSION> rotated <ANGLE>' by default, or as specified in the input arguments.

Examples

```

mtcars_tidy_MDS = reduce_dimensions(mtcars_tidy, car_model, feature, value, method="MDS"
rotate_dimensions(mtcars_tidy_MDS, `Dim1`, `Dim2`, .element = car_model, rotation_degree

```

`subset`*Extract selected-column-wise information*

Description

`subset()` takes as input a 'tbl' and returns a 'tbl' with only selected-column-related columns

Usage

```
subset(.data, .column)

## S4 method for signature 'spec_tbl_df'
subset(.data, .column)

## S4 method for signature 'tbl_df'
subset(.data, .column)

## S4 method for signature 'tbl'
subset(.data, .column)
```

Arguments

<code>.data</code>	A 'tbl'
<code>.column</code>	The name of the column of interest

Details

Maturing

This function extracts only selected-column-related information for downstream analysis (e.g., visualisation). It is disruptive in the sense that it cannot be passed anymore to nanny function.

Value

A 'tbl' object
A 'tbl' object
A 'tbl' object
A 'tbl' object

Examples

```
subset(mtcars_tidy, car_model)
```

Index

as_matrix, [2](#)
as_matrix, spec_tbl_df-method
 (*as_matrix*), [2](#)
as_matrix, tbl_df-method
 (*as_matrix*), [2](#)

cluster_elements, [3](#)
cluster_elements, spec_tbl_df-method
 (*cluster_elements*), [3](#)
cluster_elements, tbl_df-method
 (*cluster_elements*), [3](#)

combine_nest, [5](#)
combine_nest, spec_tbl_df-method
 (*combine_nest*), [5](#)
combine_nest, tbl_df-method
 (*combine_nest*), [5](#)

fill_missing, [6](#)
fill_missing, spec_tbl_df-method
 (*fill_missing*), [6](#)
fill_missing, tbl_df-method
 (*fill_missing*), [6](#)

impute_missing, [7](#)
impute_missing, spec_tbl_df-method
 (*impute_missing*), [7](#)
impute_missing, tbl_df-method
 (*impute_missing*), [7](#)

keep_variable, [8](#)
keep_variable, spec_tbl_df-method
 (*keep_variable*), [8](#)
keep_variable, tbl_df-method
 (*keep_variable*), [8](#)

lower_triangular, [9](#)
lower_triangular, spec_tbl_df-method
 (*lower_triangular*), [9](#)
lower_triangular, tbl_df-method
 (*lower_triangular*), [9](#)

nest_subset, [10](#)
nest_subset, spec_tbl_df-method
 (*nest_subset*), [10](#)
nest_subset, tbl-method, [11](#)

nest_subset, tbl_df-method
 (*nest_subset*), [10](#)

permute_nest, [11](#)
permute_nest, spec_tbl_df-method
 (*permute_nest*), [11](#)
permute_nest, tbl_df-method
 (*permute_nest*), [11](#)

reduce_dimensions, [12](#)
reduce_dimensions, spec_tbl_df-method
 (*reduce_dimensions*), [12](#)
reduce_dimensions, tbl_df-method
 (*reduce_dimensions*), [12](#)

remove_redundancy, [14](#)
remove_redundancy, spec_tbl_df-method
 (*remove_redundancy*), [14](#)
remove_redundancy, tbl_df-method
 (*remove_redundancy*), [14](#)

rotate_dimensions, [16](#)
rotate_dimensions, spec_tbl_df-method
 (*rotate_dimensions*), [16](#)
rotate_dimensions, tbl_df-method
 (*rotate_dimensions*), [16](#)

subset, [18](#)
subset, spec_tbl_df-method
 (*subset*), [18](#)
subset, tbl-method (*subset*), [18](#)
subset, tbl_df-method (*subset*), [18](#)