

Package ‘w4mclassfilter’

December 19, 2019

Version 0.98.16

Date 2019-12-17

Title W4M Class Filter

Description Filter Workflow4Metabolomics dataMatrix, sampleMetadata, and variableMetadata files by sample-class or variable-attribute range, imputing zero for NA values and eliminating zero-variance rows and columns from the data-matrix.

Author Arthur C Eschenlauer <esch0041@umn.edu>

Maintainer Arthur C Eschenlauer <eschen@alumni.princeton.edu>

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.0.2

Roxygen list(markdown = TRUE)

URL <https://github.com/HegemanLab/w4mclassfilter>

Imports utils

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

w4m_filter_by_sample_class	2
w4m_filter_imputation	5
w4m_filter_median_imputation	6
w4m_filter_no_imputation	7
w4m_filter_zero_imputation	8
w4m__nonzero_var	9
w4m__var_by_rank_or_file	10

Index	12
--------------	-----------

w4m_filter_by_sample_class

Filter W4M data matrix by sample-class

Description

Filter a set of retention-corrected W4M files (dataMatrix, sampleMetadata, variableMetadata) by sample-class or feature-attributes

Usage

```
w4m_filter_by_sample_class(
  dataMatrix_in,
  sampleMetadata_in,
  variableMetadata_in,
  dataMatrix_out,
  sampleMetadata_out,
  variableMetadata_out,
  classes = c(),
  include = FALSE,
  class_column = "class",
  samplename_column = "sampleMetadata",
  name_varmetadata_col1 = TRUE,
  name_smplmetadata_col1 = TRUE,
  variable_range_filter = c(),
  data_imputation = w4m_filter_zero_imputation,
  order_vrbl = "variableMetadata",
  order_smpl = "sampleMetadata",
  centering = c("none", "centroid", "median", "medoid")[1],
  failure_action = function(...) { cat(paste(..., SEP = "\n")) }
)
```

Arguments

dataMatrix_in input data matrix (rows are feature names, columns are sample names)

sampleMetadata_in input sample metadata (rows are sample names, one column's name matches class_column)

variableMetadata_in input variable metadata (rows are variable names)

dataMatrix_out output data matrix (rows are feature names, columns are sample names)

sampleMetadata_out output sample metadata (rows are sample names, one column's name matches class_column)

variableMetadata_out output variable metadata (rows are variable names)

<code>classes</code>	character array: names of sample classes to include or exclude; default is an empty array
<code>include</code>	logical: TRUE, include named sample classes; FALSE (the default), exclude named sample classes
<code>class_column</code>	character: name of "class" column, defaults to "class"
<code>samplename_column</code>	character: name of column with sample name, defaults to "sampleMetadata"
<code>name_varmetadata_col1</code>	logical: TRUE, name column 1 of variable metadata as "variableMetadata"; FALSE, no change; default is TRUE
<code>name_smplmetadata_col1</code>	logical: TRUE, name column 1 of sample metadata as "sampleMetadata"; FALSE, no change; default is TRUE
<code>variable_range_filter</code>	character array: array of filters specified as 'variableMetadataColumnName:min:max'; default is empty array
<code>data_imputation</code>	function(m): default imputation method for 'intb' data, where intensities have background subtracted - impute zero for NA
<code>order_vrbl</code>	character: name of column of variableMetadata on which to sort, defaults to "variableMetadata" (i.e., the first column)
<code>order_smpl</code>	character: name of column of sampleMetadata on which to sort, defaults to "sampleMetadata" (i.e., the first column)
<code>centering</code>	character: center samples by class column (which names treatment). Possible choices: "none", "centroid", "medoid", or "median"
<code>failure_action</code>	function(x, ...): action to take upon failure - defaults to 'print(x,...)'

Details

The W4M files `dataMatrix`, `sampleMetadata`, and `variableMetadata` must be a consistent set, i.e., there must be metadata in the latter two files for all (and only for) the samples and variables named in the columns and rows of `dataMatrix`.

For multivariate statistics functions, samples and variables with zero variance must be eliminated, and missing values are problematic.

Furthermore, frequently, it is desirable to analyze a subset of samples (or features) in the `dataMatrix`.

This function manipulates produces a set of files with imputed missing values, omitting features and samples that are not consistently present within the set or have zero variance. Secondly, it provides a selection-capability for samples based on whether their sample names match a regular expression pattern; this capability can be used either to select for samples with matching sample names or to exclude them. Thirdly, it provides a selection-capability for features based on whether their metadata lie within the ranges specified by 'variable_range_filter'.

Finally, this function provides as an advanced option to compute one of three types of centers for each treatment:

- "centroid" - Return only treatment-centers computed for each treatment as the mean intensity for each feature.
- "median" - Return only treatment-centers computed for each treatment as the median intensity for each feature.
- "medoid" - Return only treatment-centers computed for each treatment as the sample most similar to the other samples (the medoid).
 - By definition, the medoid is the sample having the smallest sum of its distances from other samples in the treatment.
 - Distances computed in principal-components space.
 - * Principal components are uncorrelated, so they are used here to minimize the distortion of computed distances by correlated features.
- "none" - Return all samples; do not computing centers

Inputs (dataMatrix_in, sampleMetadata_in, variableMetadata_in) may be:

- character: path to input tab-separated-values-file (TSV)
- data.frame
- matrix: allowed for dataMatrix_in only
- list: must have a member named "dataMatrix", "sampleMetadata", or "variableMetadata" for dataMatrix_in, sampleMetadata_in, or variableMetadata_in, respectively.
- environment: must have a member named "dataMatrix", "sampleMetadata", or "variableMetadata" for dataMatrix_in, sampleMetadata_in, or variableMetadata_in, respectively.

Outputs (dataMatrix_out, sampleMetadata_out, variableMetadata_out) may be:

- character: path to write a tab-separated-values-file (TSV)
- list: will add a member named "dataMatrix", "sampleMetadata", or "variableMetadata" for dataMatrix_out, sampleMetadata_out, or variableMetadata_out, respectively.
- environment: will add a member named "dataMatrix", "sampleMetadata", or "variableMetadata" for dataMatrix_out, sampleMetadata_out, or variableMetadata_out, respectively.

Please see the package vignette for further details.

Value

logical: TRUE only if filtration succeeded

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://workflow4metabolomics.org/>

Examples

```
## Not run:
# set the paths to your input files
dataMatrix_in <- "tests/testthat/input_dataMatrix.tsv"
sampleMetadata_in <- "tests/testthat/input_sampleMetadata.tsv"
variableMetadata_in <- "tests/testthat/input_variableMetadata.tsv"

# set the paths to your (nonexistent) output files
#   in a directory that DOES need to exist
dataMatrix_out <- "tests/testthat/output_dataMatrix.tsv"
sampleMetadata_out <- "tests/testthat/output_sampleMetadata.tsv"
variableMetadata_out <- "tests/testthat/output_variableMetadata.tsv"

# Example: running the filter to exclude only unwanted samples
#   include = FALSE means exclude samples with class blankpos
w4m_filter_by_sample_class(
  dataMatrix_in = dataMatrix_in
, dataMatrix_out = dataMatrix_out
, variableMetadata_in = variableMetadata_in
, variableMetadata_out = variableMetadata_out
, sampleMetadata_out = sampleMetadata_out
, sampleMetadata_in = sampleMetadata_in
, classes = c("M")
, include = TRUE
, class_column = "gender"
, samplename_column = "sampleMetadata"
, name_varmetadata_col1 = TRUE
, name_smplmetadata_col1 = TRUE
, variable_range_filter = c()
, data_imputation = w4m_filter_zero_imputation
, order_vrbl = "variableMetadata"
, order_smpl = "sampleMetadata"
, centering = "none"
, failure_action = function(...) { cat(paste(..., SEP = "\n")) }
)

## End(Not run)
```

w4m_filter_imputation	<i>Impute missing intensities to zero for W4M data matrix (deprecated)</i>
-----------------------	--

Description

Substitute zero for missing or negative intensity values in W4M data matrix (synonym for w4m_filter_zero_imputation, deprecated)

Usage

```
w4m_filter_imputation(m)
```

Arguments

m matrix: W4M data matrix potentially containing NA or negative values

Value

matrix: input data matrix with zeros substituted for negative or NA values

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://workflow4metabolomics.org/>

Examples

```
# input contains negative and missing values
my_input <- matrix(c(NA,1,-1,2), ncol = 2, nrow = 2)

# expected output converts negative and missing values to zero
my_expected <- matrix(c(0,1,0,2), ncol = 2, nrow = 2)

# run the imputation method to generate actual output
my_output <- w4m_filter_imputation(my_input)

# validate actual output against expected output
all.equal(my_output, my_expected, check.attributes = FALSE)
```

w4m_filter_median_imputation

Impute missing intensities to median for W4M data matrix

Description

Substitute median feature intensity (across all samples) for missing values and zero for negative values in W4M data matrix

Usage

```
w4m_filter_median_imputation(m)
```

Arguments

m matrix: W4M data matrix potentially containing NA or negative values

Value

matrix: input data matrix with zeros substituted for negative values and median substituted for missing values

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://workflow4metabolomics.org/>

Examples

```
# input contains negative and missing values
my_input <- matrix(c(NA,-1,3,2), ncol = 2, nrow = 2)

# expected output converts negative and missing values to zero
my_expected <- matrix(c(3,0,3,2), ncol = 2, nrow = 2)

# run the imputation method to generate actual output
my_output <- w4m_filter_median_imputation(my_input)

# validate actual output against expected output
all.equal(my_output, my_expected, check.attributes = FALSE)
```

w4m_filter_no_imputation

*Do not impute missing intensities to zero for W4M data matrix,
but convert negative intensities to zero*

Description

Substitute zero for negative intensity values in W4M data matrix, but not for missing intensity values

Usage

```
w4m_filter_no_imputation(m)
```

Arguments

m matrix: W4M data matrix potentially containing negative values

Value

matrix: input data matrix with zeros substituted for negative values

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://workflow4metabolomics.org/>

Examples

```
# input contains negative and missing values
my_input <- matrix(c(NA,1,-1,2), ncol = 2, nrow = 2)

# expected output converts negative and missing values to zero
my_expected <- matrix(c(NA,1,0,2), ncol = 2, nrow = 2)

# run the imputation method to generate actual output
my_output <- w4m_filter_no_imputation(my_input)

# validate actual output against expected output
all.equal(my_output, my_expected, check.attributes = FALSE)
```

w4m_filter_zero_imputation

Impute missing values to zero for W4M data matrix

Description

Substitute zero for missing or negative intensity values in W4M data matrix

Usage

```
w4m_filter_zero_imputation(m)
```

Arguments

m matrix: W4M data matrix potentially containing NA or negative values

Value

matrix: input data matrix with zeros substituted for negative or NA values

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://workflow4metabolomics.org/>

Examples

```
# input contains negative and missing values
my_input <- matrix(c(NA,1,-1,2), ncol = 2, nrow = 2)

# expected output converts negative and missing values to zero
my_expected <- matrix(c(0,1,0,2), ncol = 2, nrow = 2)

# run the imputation method to generate actual output
my_output <- w4m_filter_zero_imputation(my_input)

# validate actual output against expected output
all.equal(my_output, my_expected, check.attributes = FALSE)
```

w4m__nonzero_var	<i>Support function to eliminate rows or columns that have zero variance</i>
------------------	--

Description

(w4mclassfilter support function) Produce matrix from matrix xpre where all rows and columns having zero variance have been removed

Usage

```
w4m__nonzero_var(m)
```

Arguments

m	matrix: W4M data matrix potentially having rows or columns having zero variance
---	---

Value

matrix: input data matrix after removal of rows or columns having zero variance

Examples

```
m <- matrix(
  c(
    1, 2, 3, 4,
    3, 3, 3, 4,
    5, 7, 11, 4,
```

```

      13, 17, 19, 4
    )
    , nrow = 4
    , ncol = 4
    , byrow = TRUE
  )
  rownames(m) <- c("A", "B", "C", "D")
  colnames(m) <- c("W", "X", "Y", "Z")
  expected <- matrix(
    c(
      1, 2, 3,
      5, 7, 11,
      13, 17, 19
    )
    , nrow = 3
    , ncol = 3
    , byrow = TRUE
  )
  rownames(expected) <- c("A", "C", "D")
  colnames(expected) <- c("W", "X", "Y")
  all.equal(w4m__nonzero_var(m), expected)

```

w4m__var_by_rank_or_file

Support function to compute variances of matrix rows or columns

Description

(w4mclassfilter support function) Compute variances of rows or columns of a W4M data matrix

Usage

```
w4m__var_by_rank_or_file(m, dim = 1)
```

Arguments

m	matrix: W4M data matrix for which variance must be computed for rows or columns
dim	integer: For variances of rows, dim == 1, for variances of columns, dim == 2

Value

vector of numeric: variances for rows or columns

Author(s)

Art Eschenlauer, <esch0041@umn.edu>

See Also

<https://github.com/HegemanLab/w4mclassfilter>

<http://stackoverflow.com/a/25100036>

Examples

```
m <- base::matrix(
  c(
    1, 2, 3,
    5, 7, 11,
    13, 17, 19
  )
, nrow = 3
, ncol = 3
, byrow = TRUE
)
rowvars <- w4m__var_by_rank_or_file(m = m, dim = 1)
expecteds <- c(stats::var(c(1,2,3)),stats::var(c(5,7,11)),stats::var(c(13,17,19)))
base::all.equal(rowvars, expecteds)
colvars <- w4m__var_by_rank_or_file(m = m, dim = 2)
expecteds <- c(stats::var(c(1,5,13)),stats::var(c(2,7,17)),stats::var(c(3,11,19)))
base::all.equal(colvars, expecteds)
```

Index

w4m__nonzero_var, [9](#)
w4m__var_by_rank_or_file, [10](#)
w4m_filter_by_sample_class, [2](#)
w4m_filter_imputation, [5](#)
w4m_filter_median_imputation, [6](#)
w4m_filter_no_imputation, [7](#)
w4m_filter_zero_imputation, [8](#)