Package 'cytominer'

September 17, 2017

Type Package

Title Methods for Image-Based Cell Profiling

Version 0.1.0

Description Typical morphological profiling datasets have millions of cells and hundreds of features per cell. When working with this data, you must clean the data, normalize the features to make them comparable across experiments, transform the features, select features based on their quality, and aggregate the single-cell data, if needed. 'cytominer' makes these steps fast and easy. Methods used in practice in the field are discussed in Caicedo (2017) <doi:10.1038/nmeth.4397>. An overview of the field is presented in Caicedo (2016) <doi:10.1016/j.copbio.2016.04.003>.

Depends R (>= 3.3.3)

License BSD_3_clause + file LICENSE

LazyData TRUE

Imports caret (>= 6.0.76), doParallel (>= 1.0.10), dplyr (>= 0.7.2), foreach (>= 1.4.3), futile.logger (>= 1.4.3), magrittr (>= 1.5), purrr (>= 0.2.3), rlang (>= 0.1.2), tibble (>= 1.3.4), tidyr (>= 0.7.1)

Suggests DBI (>= 0.7), dbplyr (>= 1.1.0), knitr (>= 1.17), lazyeval (>= 0.2.0), readr (>= 1.1.1), rmarkdown (>= 1.6), RSQLite (>= 2.0), stringr (>= 1.2.0), testthat (>= 1.0.2)

VignetteBuilder knitr

URL https://github.com/cytomining/cytominer

BugReports https://github.com/cytomining/cytominer/issues

RoxygenNote 6.0.1

NeedsCompilation no

Author Tim Becker [aut],
Allen Goodman [aut],
Claire McQuin [aut],
Mohammad Rohban [aut],
Shantanu Singh [aut, cre]

Maintainer Shantanu Singh <shsingh@broadinstitute.org>

Repository CRAN

Date/Publication 2017-09-17 18:25:04 UTC

R topics documented:

aggregate 2
correlation_threshold 3
count_na_rows 4
drop_na_columns 5
drop_na_rows 6
generalized_log 6
normalize 7
replicate_correlation 8
transform 9
variable_importance 10
variable_select 11
variance_threshold 13

Index **14**

`aggregate` *Aggregate data based on given grouping.*

Description

`aggregate` aggregates data based on the specified aggregation method.

Usage

```
aggregate(population, variables, strata, operation = "mean", ...)
```

Arguments

`population`	tbl with grouping (metadata) and observation variables.
`variables`	character vector specifying observation variables.
`strata`	character vector specifying grouping variables for aggregation.
`operation`	optional character string specifying method for aggregation. This must be one of the strings `"mean"`, `"median"`, `"mean+sd"`.
`...`	optional arguments passed to aggregation operation

Value

aggregated data of the same class as `population`.

Examples

```
population <- tibble::data_frame(
   Metadata_group = c("control", "control", "control", "control",
                      "experiment", "experiment", "experiment", "experiment"),
   Metadata_batch = c("a", "a", "b", "b", "a", "a", "b", "b"),
   AreaShape_Area = c(10, 12, 15, 16, 8, 8, 7, 7)
 )
variables <- c("AreaShape_Area")
strata <- c("Metadata_group", "Metadata_batch")
aggregate(population, variables, strata, operation = "mean")
```

`correlation_threshold` *Remove redundant variables.*

Description

`correlation_threshold` returns list of variables such that no two variables have a correlation greater than a specified threshold.

Usage

```
correlation_threshold(variables, sample, cutoff = 0.9, method = "pearson")
```

Arguments

`variables`	character vector specifying observation variables.
`sample`	tbl containing sample used to estimate parameters.
`cutoff`	threshold between [0,1] that defines the minimum correlation of a selected feature.
`method`	optional character string specifying method for calculating correlation. This must be one of the strings `"pearson"` (default), `"kendall"`, `"spearman"`.

Details

`correlation_threshold` is a wrapper for `caret::findCorrelation`.

Value

character vector specifying observation variables to be excluded.

Examples

```
suppressMessages(suppressWarnings(library(magrittr)))
sample <- tibble::data_frame(
   x = rnorm(30),
   y = rnorm(30)/1000
 )

sample %<>% dplyr::mutate(z = x + rnorm(30) / 10)
variables <- c("x", "y", "z")

head(sample)
cor(sample)

# `x` and `z` are highly correlated; one of them will be removed

correlation_threshold(variables, sample)
```

`count_na_rows` *Count the number of* NA*s per variable.*

Description

`count_na_rows` counts the number of NAs per variable.

Usage

```
count_na_rows(population, variables)
```

Arguments

`population`	tbl with grouping (metadata) and observation variables.
`variables`	character vector specifying observation variables.

Value

data frame with frequency of NAs per variable.

Examples

```
population <- tibble::data_frame(
   Metadata_group = c("control", "control", "control", "control",
                      "experiment", "experiment", "experiment", "experiment"),
   Metadata_batch = c("a", "a", "b", "b", "a", "a", "b", "b"),
   AreaShape_Area = c(10, 12, 15, 16, 8, 8,7, 7),
   AreaShape_length = c(2, 3, NA, NA, 4, 5, 1, 5)
)
```

```
variables <- c('AreaShape_Area','AreaShape_length')
count_na_rows(population, variables)
```

`drop_na_columns` *Remove variables with NA values.*

Description

`drop_na_columns` returns list of variables which have greater than a specified threshold number of `NA`s.

Usage

```
drop_na_columns(population, variables, cutoff = 0.05)
```

Arguments

`population`	tbl with grouping (metadata) and observation variables.
`variables`	character vector specifying observation variables.
`cutoff`	threshold between [0,1]. Variables with an `NA` frequency > `cutoff` are returned.

Value

character vector specifying observation variables to be excluded.

Examples

```
 population <- tibble::data_frame(
   Metadata_group = c("control", "control", "control", "control",
                      "experiment", "experiment", "experiment", "experiment"),
   Metadata_batch = c("a", "a", "b", "b", "a", "a", "b", "b"),
   AreaShape_Area = c(10, 12, 15, 16, 8, 8 ,7, 7),
   AreaShape_Length = c(2, 3, NA, NA, 4, 5, 1, 5)
 )
variables <- c('AreaShape_Area','AreaShape_Length')
drop_na_columns(population, variables)
```

drop_na_rows *Drop rows that are* NA *in all variables.*

Description

drop_na_rows drops rows that are NA in all variables.

Usage

```
drop_na_rows(population, variables)
```

Arguments

population	tbl with grouping (metadata) and observation variables.
variables	character vector specifying observation variables.

Value

population without rows that have NA in all variables.

Examples

```
 population <- tibble::data_frame(
   Metadata_group = c("control", "control", "control", "control",
                      "experiment", "experiment", "experiment", "experiment"),
   Metadata_batch = c("a", "a", "b", "b", "a", "a", "b", "b"),
   AreaShape_Area = c(10, 12, NA, 16, 8, 8, 7, 7),
   AreaShape_Length = c(2, 3, NA, NA, 4, 5, 1, 5)
 )
variables <- c('AreaShape_Area','AreaShape_Length')
drop_na_rows(population, variables)
```

generalized_log *Generalized log transform data.*

Description

generalized_log transforms specified observation variables using $x = log((x+sqrt(x^2+offset^2))/2)$.

Usage

```
generalized_log(population, variables, offset = 1)
```

Arguments

population	tbl with grouping (metadata) and observation variables.
variables	character vector specifying observation variables.
offset	optional offset parameter for the transformation.

Value

transformed data of the same class as population.

Examples

```
population <- tibble::data_frame(
   Metadata_Well = c("A01", "A02", "B01", "B02"),
   Intensity_DNA = c(8, 20, 12, 32)
 )
variables <- c("Intensity_DNA")
generalized_log(population, variables)
```

---

normalize *Normalize observation variables.*

Description

normalize normalizes observation variables based on the specified normalization method.

Usage

```
normalize(population, variables, strata, sample, operation = "standardize",
  ...)
```

Arguments

population	tbl with grouping (metadata) and observation variables.
variables	character vector specifying observation variables.
strata	character vector specifying grouping variables for grouping prior to normalization.
sample	tbl containing sample that is used by normalization methods to estimate parameters. sample has same structure as population. Typically, sample corresponds to controls in the experiment.
operation	optional character string specifying method for normalization. This must be one of the strings "standardize" (default), "robustize".
...	arguments passed to normalization operation

Value

normalized data of the same class as `population`.

Examples

```
suppressMessages(suppressWarnings(library(magrittr)))
population <- tibble::data_frame(
   Metadata_group = c("control", "control", "control", "control",
                      "experiment", "experiment", "experiment", "experiment"),
   Metadata_batch = c("a", "a", "b", "b", "a", "a", "b", "b"),
   AreaShape_Area = c(10, 12, 15, 16, 8, 8, 7, 7)
 )
variables <- c('AreaShape_Area')
strata <- c('Metadata_batch')
sample <- population %>% dplyr::filter(Metadata_group == 'control')
cytominer::normalize(population, variables, strata, sample, operation = "standardize")
```

`replicate_correlation` *Measure replicate correlation of variables.*

Description

'replicate_correlation' measures replicate correlation of variables.

Usage

```
replicate_correlation(sample, variables, strata, replicates,
  replicate_by = NULL, split_by = NULL, cores = NULL)
```

Arguments

`sample`	tbl containing sample used to estimate parameters.
`variables`	character vector specifying observation variables.
`strata`	character vector specifying grouping variables for grouping prior to normalization.
`replicates`	number of replicates.
`replicate_by`	optional character string specifying column containing the replicate id.
`split_by`	optional character string specifying column by which to split the sample into batches; replicate correlations will be calculate per batch.
`cores`	optional integer specifying number of CPU cores used for parallel computing using `doParallel`.

Value

data frame of variable quality measurements

Examples

```
set.seed(123)
x1 <- rnorm(10)
x2 <- x1 + rnorm(10) / 100
y1 <- rnorm(10)
y2 <- y1 + rnorm(10) / 10
z1 <- rnorm(10)
z2 <- z1 + rnorm(10) / 1

batch <- rep(rep(1:2, each=5), 2)

treatment <- rep(1:10, 2)

replicate_id <- rep(1:2, each=10)

sample <-
  tibble::data_frame(x = c(x1, x2), y = c(y1, y2), z = c(z1, z2),
                     Metadata_treatment = treatment,
                     Metadata_replicate_id = replicate_id,
                     Metadata_batch = batch)

head(sample)

# `replicate_correlation`` returns the median, min, and max
# replicate correlation (across batches) per variable
replicate_correlation(sample = sample,
                      variables = c("x", "y", "z"),
                      strata = c("Metadata_treatment"),
                      replicates = 2,
                      split_by = "Metadata_batch",
                      replicate_by = "Metadata_replicate_id",
                      cores = 1)
```

`transform` *Transform observation variables.*

Description

`transform` transforms observation variables based on the specified transformation method.

Usage

```
transform(population, variables, operation = "generalized_log", ...)
```

Arguments

`population`	tbl with grouping (metadata) and observation variables.
`variables`	character vector specifying observation variables.

operation	optional character string specifying method for transform. Currently, only "generalized_log" (default) is implemented.
...	arguments passed to transformation operation.

Value

transformed data of the same class as population.

Examples

```
population <- tibble::data_frame(
   Metadata_Well = c("A01", "A02", "B01", "B02"),
   Intensity_DNA = c(8, 20, 12, 32)
 )
variables <- c("Intensity_DNA")
transform(population, variables, operation = "generalized_log")
```

---

variable_importance *Measure variable importance.*

Description

variable_importance measures importance of variables based on specified methods.

Usage

```
variable_importance(sample, variables, operation = "replicate_correlation",
  ...)
```

Arguments

sample	tbl containing sample used to estimate parameters.
variables	character vector specifying observation variables.
operation	optional character string specifying method for computing variable importance. Currently, only "replicate_correlation" (default) is implemented.
...	arguments passed to variable importance operation.

Value

data frame containing variable importance measures.

Examples

```
set.seed(123)
x1 <- rnorm(10)
x2 <- x1 + rnorm(10) / 100
y1 <- rnorm(10)
y2 <- y1 + rnorm(10) / 10
z1 <- rnorm(10)
z2 <- z1 + rnorm(10) / 1

batch <- rep(rep(1:2, each=5), 2)

treatment <- rep(1:10, 2)

replicate_id <- rep(1:2, each=10)

sample <-
  tibble::data_frame(x = c(x1, x2), y = c(y1, y2), z = c(z1, z2),
                     Metadata_treatment = treatment,
                     Metadata_replicate_id = replicate_id,
                     Metadata_batch = batch)

head(sample)

# `replicate_correlation`` returns the median, min, and max
# replicate correlation (across batches) per variable
variable_importance(sample = sample,
                    variables = c("x", "y", "z"),
                    operation = "replicate_correlation",
                    strata = c("Metadata_treatment"),
                    replicates = 2,
                    split_by = "Metadata_batch",
                    replicate_by = "Metadata_replicate_id",
                    cores = 1)
```

`variable_select` *Select observation variables.*

Description

`variable_select` selects observation variables based on the specified variable selection method.

Usage

```
variable_select(population, variables, sample = NULL,
  operation = "variance_threshold", ...)
```

Arguments

population	tbl with grouping (metadata) and observation variables.
variables	character vector specifying observation variables.
sample	tbl containing sample that is used by some variable selection methods. sample has same structure as population.
operation	optional character string specifying method for variable selection. This must be one of the strings "variance_threshold", "correlation_threshold", "drop_na_columns".
...	arguments passed to selection operation.

Value

variable-selected data of the same class as population.

Examples

```
# In this example, we use `correlation_threshold` as the operation for
# variable selection.

suppressMessages(suppressWarnings(library(magrittr)))
population <- tibble::data_frame(
   x = rnorm(100),
   y = rnorm(100)/1000
 )

population %<>% dplyr::mutate(z = x + rnorm(100) / 10)

sample <- population %>% dplyr::slice(1:30)

variables <- c("x", "y", "z")

operation <- "correlation_threshold"

cor(sample)

# `x` and `z` are highly correlated; one of them will be removed

head(population)

futile.logger::flog.threshold(futile.logger::ERROR)

variable_select(population, variables, sample, operation) %>% head()
```

variance_threshold *Remove variables with near-zero variance.*

Description

variance_threshold returns list of variables that have near-zero variance.

Usage

```
variance_threshold(variables, sample)
```

Arguments

variables	character vector specifying observation variables.
sample	tbl containing sample used to estimate parameters.

Details

variance_threshold is a reimplementation of caret::nearZeroVar, using the default values for freqCut and uniqueCut.

Value

character vector specifying observation variables to be excluded.

Examples

```
sample <- tibble::data_frame(
   AreaShape_Area = c(10, 12, 15, 16, 8, 8, 7, 7, 13, 18),
   AreaShape_Euler = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
 )
variables <- c("AreaShape_Area", "AreaShape_Euler")
variance_threshold(variables, sample)
```

Index

aggregate, 2

correlation_threshold, 3
count_na_rows, 4

drop_na_columns, 5
drop_na_rows, 6

generalized_log, 6

normalize, 7

replicate_correlation, 8

transform, 9

variable_importance, 10
variable_select, 11
variance_threshold, 13