Package 'FCBF'

April 15, 2020

Type Package

Title Fast Correlation Based Filter for Feature Selection

Version 1.4.0

Description This package provides a simple R implementation for the Fast Correlation Based Filter described in Yu, L. and Liu, H.; Feature Selection for High-Dimensional Data: A Fast Correlation Based Filter Solution,Proc. 20th Intl. Conf. Mach. Learn. (ICML-2003), Washington DC, 2003
The current package is an intent to make easier for bioinformaticians to use FCBF for feature selection, especially regarding transcriptomic data.This implies discretizing expression (function discretize_exprs) before calculating the features that explain the class, but are not predictable by other features.
The functions are implemented based on the algorithm of Yu and Liu, 2003 and Rajarshi Guha's implementation from 13/05/2005 available (as of 26/08/2018) at http://www.rguha.net/code/R/fcbf.R .

License MIT + file LICENSE

Encoding UTF-8

LazyData False

RoxygenNote 6.1.1

Imports ggplot2, gridExtra, pbapply, parallel, SummarizedExperiment, stats, mclust

Suggests caret, mlbench, SingleCellExperiment, knitr, rmarkdown, testthat, BiocManager

biocViews GeneTarget, FeatureExtraction, Classification, GeneExpression, SingleCell, ImmunoOncology

VignetteBuilder knitr

Depends R (>= 3.6)

git_url https://git.bioconductor.org/packages/FCBF

git_branch RELEASE_3_10

git_last_commit ad6355e

git_last_commit_date 2019-10-29

Date/Publication 2020-04-14

Author Tiago Lubiana [aut, cre], Helder Nakaya [aut, ths]

Maintainer Tiago Lubiana <tiago.lubiana.alves@usp.br>

R topics documented:

`discretize_exprs` *discretize_exprs Simple discretizing of gene expression*

Description

This function takes the range of values for each gene in a previously normalized expression table (genes/variables in rows, samples/ observations in columns) and uses it for a width-based discretization. Each feature is divide into "n" bins of equal width. The first bin is attributed the class 'low' and the next bins are assigned to "high". It transposes the original expression table.

Usage

```
discretize_exprs(expression_table, number_of_bins = 3,
  method = "varying_width", alpha = 1, centers = 3,
  min_max_cutoff = 0.25, progress_bar = TRUE)
```

Arguments

`expression_table`
: A previously normalized expression table Note: this might drastically change the number of selected features.

`number_of_bins`
: Number of equal-width bins for discretization. Note: it is a binary discretization, with the first bin becoming one class ('low') and the other bins, another class ('high'). Defaults to 3.

`method`
: Method applied to all genes for discretization. Methods available: "varying_width" (Varying width binarization, default, described in function description. Modulated by the number_of_bins param), "mean" (Split in ON/OFF by each gene mean expression), "median" (Split in ON/OFF by each gene median expression), "mean_sd"(Split in low/medium/high by each assigning "medium" to the interval between mean +- standard_deviation. Modulated by the alpha param, which enlarges (>1) or shrinks (<1) the "medium" interval.),), "kmeans"(Split in different groups by the kmeans algorithm. As many groups as specified by the centers param) and "min_max_percent" (Similat to the "varying width", a binarization threshold in a percent of the min-max range is set. (minmaxpercent param)), "GMM" (A Gaussian Mixture Model as implemented by the package mclust, trying to fit 2:5 Gaussians)

alpha	Modulator for the "mean_sd" method.Enlarges (>1) or shrinks (<1) the "medium" interval. Defaults to 1.
centers	Modulator for the "kmeans" method. Defaults to 3.
min_max_cutoff	<- Modulator for the "min_max_percent" method. Defaults to 0.25.
progress_bar	Enables a progress bar for the discretization. Defaults to TRUE.

Value

A data frame with the discretized features in the same order as previously

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
head(discrete_expression[, 1:4])
```

---

discretize_exprs_supervised *supervised_disc_df*

Description

Uses several discretizations and selects the one that is best for a given variable (gene) in comparison to a target class by equivocation

Usage

```
discretize_exprs_supervised(expression_table, target, parallel = FALSE)
```

Arguments

expression_table	A previously normalized expression table
target	A series of labels matching each of the values in the gene vector (genes in rows, cells/samples in columns)
parallel	Set calculations in parallel. May be worth it if the number of rows and columns is really large. Do watchout for memory overload.

Value

A data frame with the discretized features in the same order as previously

Examples

```
data(scDengue)
exprs <- as.data.frame(SummarizedExperiment::assay(scDengue, 'logcounts'))
exprs <- exprs [1:200, 1:120]
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
discrete_expression <- as.data.frame(discretize_exprs_supervised(exprs,target))
fcbf(discrete_expression,target, thresh = 0.05, verbose = TRUE)
```

discretize_gene_supervised
discretize_gene_supervised

Description

Uses several discretizations and selects the one that is best for a given variable (gene) in comparison to a target class by equivocation Note that set.seed() should be used for reproducing the results. The inner kmeans #' function would, otherwise, provide different results each time.

Usage

```
discretize_gene_supervised(gene, target, output = "discretized_vector",
  discs = c(".split_vector_in_two_by_median",
  ".split_vector_in_two_by_mean", ".split_vector_by_kmeans",
  ".split_vector_in_three_by_mean_sd",
  ".split_vector_in_two_by_min_max_thresh"), vw_params = c(0.25, 0.5,
  0.75), kmeans_centers = c(2, 3, 4), sd_alpha = c(0.75, 1, 1.25))
```

Arguments

gene	A previously normalized gene expression vector
target	A series of labels matching each of the values in the gene vector
output	If it is equal to 'discretized_vector', the output is the vector. I it is 'su', returns a dataframe. Defaults to 'discretized_vector'
discs	Defaults to c(".split_vector_in_two_by_median", split_vector_in_two_by_mean", ".split_vector_by_kmeans", ".split_vector_in_three_by_mean_sd", ".split_vector_in_two_by_vw")
vw_params	cuttof parameters for the varying width function. Defaults to 0.25, 0.5 and 0.75
kmeans_centers	Numeric vector with the number of centers to use for kmeans. Defaults to 2, 3 and 4
sd_alpha	Parameter for adusting the 'medium' level of the mean +- sd discretization. Defaults to sd_alpha = c(0.75, 1, 1.25))

Details

Note that a seed for random values has to bew set for reproducibility. Otherwise, the "kmeans" value might vary from iteration to iteration.

Value

A data frame with the discretized features in the same order as previously

Examples

```
data(scDengue)
exprs <- as.data.frame(SummarizedExperiment::assay(scDengue, 'logcounts'))
gene <- exprs['ENSG00000166825',]
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
set.seed(3)
```

```
discrete_expression <- as.data.frame(discretize_gene_supervised(gene, target))
table(discrete_expression)
```

`fcbf` *Fast Correlation Based Filter function.*

Description

This functions allows selection of variables from a feature table of discrete/categorial variables and a target class. The function is based on the algorithm described in Yu, L. and Liu, H.; Feature Selection for High-Dimensional Data A Fast Correlation Based Filter Solution, Proc. 20th Intl. Conf. Mach. Learn. (ICML-2003), Washington DC, 2003

Usage

```
fcbf(x, y, thresh = 0.25, n_genes = NULL, verbose = FALSE,
  samples_in_rows = FALSE, balance_classes = FALSE)
```

Arguments

`x`	A table of features (samples in rows, variables in columns, and each observation in each cell)
`y`	A target vector, factor containing classes of the observations. Note: the observations must be in the same order as the parameter x
`thresh`	A threshold for the minimum correlation (as determined by symettrical uncertainty) between each variable and the class. Defaults to 0.25. Note: this might drastically change the number of selected features.
`n_genes`	Sets the number of genes to be selected in the first part of the algorithm. If left unchanged, it defaults to NULL and the thresh parameter is used. Caution: it overrides the thresh parameter altogether.
`verbose`	Adds verbosity. Defaults to FALSE.
`samples_in_rows`	A flag for the case in which samples are in rows and variables/genes in columns. Defaults to FALSE.
`balance_classes`	Balances number of instances in the target vector y by sampling the number of instances in the minor class from all others. The number of samplings is controlled by resampling_number. Defaults to FALSE.

Details

Obs: For gene expression, you will need to run discretize_exprs first

Value

Returns a data frame with the selected features index (first row) and their symmetrical uncertainty values regarding the class (second row). Variable names are present in rownames

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
head(discrete_expression[,1:4])
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
fcbf(discrete_expression,target, thresh = 0.05, verbose = TRUE)
fcbf(discrete_expression,target, n_genes = 100)
```

---

get_ig *Get information gain*

Description

This functions runs information gain for a feature table and a class, returning the scores of information gain for all features

Usage

```
get_ig(x, y)
```

Arguments

x	A table of features (observations in rows, variables in columns)
y	A target vector, factor containing classes of the observations. Note: the observations must be in the same order as the parameter x.

Value

A dataframe containing the SU values for each feature

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
ig_values <- get_ig(discrete_expression[,],target[])
ig_values[1:10,]
```

get_su *Symmetrical Uncertainty diagnostic*

Description

This functions runs symmetrical uncertainty for a feature table and a class, returning the scores of symmetrical uncertainty for all features

Usage

```
get_su(x, y, samples_in_rows = FALSE, bar_of_progress = FALSE)
```

Arguments

x	A table of features (observations in rows, variables in columns)
y	A target vector, factor containing classes of the observations. Note: the observations must be in the same order as the parameter x.
samples_in_rows	A flag for the case in which samples are in rows and variables/genes in columns. Defaults to FALSE.
bar_of_progress	A flag to show progress. Defaults to FALSE.

Value

A dataframe containing the SU values for each feature

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
su_values <- get_su(discrete_expression[,],target[])
su_values[1:10,]
```

IG *Information Gain This functions runs Information Gain for two features, returning the score*

Description

Information Gain This functions runs Information Gain for two features, returning the score

Usage

```
IG(x, y, base = exp(1))
```

Arguments

x	A vector containing a categorical feature
y	A vector containing other categorical feature

Value

A numerical value for the Information Gain score

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
discrete_expression_gene_1 <- discrete_expression$V1
discrete_expression_gene_2 <- discrete_expression$V2
IG(discrete_expression_gene_1,discrete_expression_gene_2)
```

---

plot_one_module *plot_one_module*

Description

Takes as input an adjacency matrix from the find_modules function (output = "matrix") an a list of hubs (FCBF - selected) (Networks are inspired in the CEMiTool package networks) And returns a pdf plotting the modules with their connections

Usage

```
plot_one_module(adjacency_matrix, name, color = "#B30000FF", n = 10)
```

Arguments

adjacency_matrix	An adjacency matrix from the find modules function (dataframe with a 'genes' column)
name	The name of the plot.
color	The color of the plot. Defaults to a shade of red ("#B30000FF").
n	The max number of gene labels to show. Defaults to 10.
genes	A character vector with the FCBF selected genes

Value

A gg object

scDengue *Dengue infected macrophages; gene expression data from GEO study GSE110496*

Description

Expression data from single cells, from adengue virus infection study by Zanini et al, #' 2018. The expression was filtered to get cells 12 hours after infection with #' a multiplicity of infection (moi) of 1 (dengue) or uninfected(ctrl). Gene counts were normalized via Bioconductor package "SCNorm".

Usage

```
data(scDengue)
```

Format

An object of class `SingleCellExperiment`

Details

Gene expression has to be discretized for use in FCBF.

Source

GEO

References

Zanini, F., Pu, S. Y., Bekerman, E., Einav, S., & Quake, S. R. (2018). Single-cell transcriptional dynamics of flavivirus infection. Elife, 7, e32942. PubMed

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
# Discretize gene expression
discrete_expression <- as.data.frame(discretize_exprs(exprs))
fcbf_features <- fcbf(discrete_expression,
                              target,
                              thresh = 0.05,
                              verbose = TRUE)
```

SU — *Symmetrical Uncertainty diagnostic This functions runs symmetrical uncertainty for two features, returning the score*

Description

Symmetrical Uncertainty diagnostic This functions runs symmetrical uncertainty for two features, returning the score

Usage

```
SU(x, y, base = exp(1))
```

Arguments

x	A vector containing a categorical feature
y	A vector containing other categorical feature

Value

A numerical value for the Symetrical Uncertainty score

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
discrete_expression_gene_1 <- discrete_expression$V1
discrete_expression_gene_2 <- discrete_expression$V2
SU(discrete_expression_gene_1,discrete_expression_gene_2)
```

su_plot — *Symmetrical Uncertainty diagnostic*

Description

This functions runs symmetrical uncertainty for a feature table and a class, returning an histogram of the scores

Usage

```
su_plot(x, y)
```

Arguments

x	A table of features (observations in rows, variables in columns)
y	A target vector, factor containing classes of the observations. Note: the observations must be in the same order as the parameter x.

Value

Plots an histogram of symmetrical uncertainty values regarding the class.

Examples

```
data(scDengue)
exprs <- SummarizedExperiment::assay(scDengue, 'logcounts')
discrete_expression <- as.data.frame(discretize_exprs(exprs))
infection <- SummarizedExperiment::colData(scDengue)
target <- infection$infection
su_plot(discrete_expression,target)
```

Index