Package ‘kmcut’

June 14, 2020

Type Package

Title A package for the optimized Kaplan Meier analysis and identification and validation of prognostic gene expression based biomarkers.

Version 2.0

Author Igor B. Kuznetsov and Javed Khan

Maintainer The package maintainer <ikuznetsov@albany.edu>

Description
The main purpose of the package is to identify potential prognostic gene signatures and an optimal numeric cutoff for each signature that can be used to stratify a group of test subjects (samples) into two sub-groups with significantly different survival (better vs. worse). The main requirement of the package is that for a group of test subjects (such as patients) two types of data are available: (i) right-censored survival time data and (ii) at least one gene expression-like feature with a large proportion of unique values describing each test subject.

License GPL-3

Encoding UTF-8

LazyData true

Imports data.table,
stats,
stringr,
survival,
tools,
pracma

RoxygenNote 6.1.1

R topics documented:

`extractcolumns` *Extract a set of columns (such as a sub-set of samples) from a data table. All rows will be preserved.*

Description

Extract a set of columns (such as a sub-set of samples) from a data table. All rows will be preserved.

Usage

```
extractcolumns(fnamein, fids, fnameout, wdir)
```

Arguments

`fnamein`	character vector that specifies the name of tab-delimited text file with the input data table.
`fids`	character vector that specifies the name of text file with column ids (such as sample ids). The file must contain one column id per line, without any trailing spaces or any other additional symbols.
`fnameout`	character vector that specifies the name of output file where the new data table will be saved.
`wdir`	character vector that specifies the name of the working directory for the input/output files.

Value

no return value

Examples

```
Basic usage:
extractcolumns(fnamein="table.txt", fids="columnids.txt", fnameout="newtable.txt", wdir="c:/test")

Example with built-in data files:

library(data.table)
library(kmcut)

# Load example gene expression data table for 2 genes
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
idlist = system.file("extdata", "columnids.txt", package="kmcut")

extractcolumns(fnamein=fdat, fids=idlist, fnameout="example_samples_subset.txt", wdir="c:/test")

This will create a tab-delimited text file in directory "c:/test":
"example_samples_subset.txt"
```

extractrows *Extract a sub-set of rows (such as a group of gene ids) from a data table. All columns will be preserved.*

Description

Extract a sub-set of rows (such as a group of gene ids) from a data table. All columns will be preserved.

Usage

```
extractrows(fnamein, fids, fnameout, wdir)
```

Arguments

fnamein	character vector that specifies the name of tab-delimited text file with the input data table.
fids	character vector that specifies the name of text file with row ids (such as gene ids). The file must contain one row id per line, without any trailing spaces or any other additional symbols.
fnameout	character vector that specifies the name of output file where the new data table will be saved.
wdir	character vector that specifies the name of the working directory for the input/output files.

Value

no return value

Examples

```
Basic usage:
extractrows(fnamein="table.txt", fids="geneids.txt", fnameout="newtable.txt", wdir="c:/test")

Example with built-in data files:

library(data.table)
library(kmcut)

# Load example gene expression data table for 2 genes
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
# Load a list that contains one gene id (MYCN)
idlist = system.file("extdata", "rowids.txt", package="kmcut")
# Run the function
extractrows(fnamein=fdat, fids=idlist, fnameout="example_genes_subset.txt", wdir="c:/test")

This will create a tab-delimited text file with one row "MYCN" in directory "c:/test":
"example_genes_subset.txt"
```

`kmoptpermcut` *For each feature finds a cutoff that optimally stratifies samples into 2 groups, plots Kaplan-Meier survival curves and observed vs. expected optimization plot. Then, uses permutation test to estimate the statistical significance of the optimal cutoff.*

Description

For each feature finds a cutoff that optimally stratifies samples into 2 groups, plots Kaplan-Meier survival curves and observed vs. expected optimization plot. Then, uses permutation test to estimate the statistical significance of the optimal cutoff.

Usage

```
kmoptpermcut(fname, sfname, wdir, seed = NULL, min_fraction = 0.1,
  min_up_down = 1, n_iter = 100, peak_tolerance = 0.1,
  psort = FALSE, min_uval = 50, wlabels = TRUE, wpdf = TRUE,
  verbose = TRUE)
```

Arguments

`fname` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must contain sample ids that match those in 'fname'. The file must contain columns called 'stime' and 'scens', with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding "KMoptp_minf_.2f_iter_d" and corresponding extension to 'fname'.

`seed` an integer value that specifies the seed for random number generator

`min_fraction` numeric value that specifies the minimal fraction of samples in the smaller group (default is 0.1).

`min_up_down` numeric value that specifies the minimal number of up/down points on either side of the peak for pracma::findpeaks function (default is 1).

`n_iter` numeric value that specifies the number of iterations for the permutation test. The default is n_iter=100 for fast calculations. Recommended is n_iter=10000 (slow, especially for a large number of samples/features).

`peak_tolerance` numeric value that specifies the maximal difference between in heigth between top peaks. The peak within 'peak_tolerance' closest to the median value is selected.

`psort` logical value whether to sort the output table by p-values in increasing order (default is FALSE).

`min_uval` numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

`wlabels` logical value whether to write a CSV file with low/high (below/above the cutoff) group sample labels (default is TRUE).

`wpdf` logical value whether to write a PDF file with survival curves and plots (default is TRUE).

`verbose` logical value whether to print progress (default is TRUE).

Value

no return value

Examples

```
Basic usage:
kmoptpermcut(fname="table.txt",sfname="survival.txt",wdir="c:/test",n_iter=1000,seed=1234)

Example with data files included in the package:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

# Search for optimal cutoffs and run the permutation tests
kmoptpermcut(fname=fdat, sfname=sdat, wdir="c:/test", n_iter=100, seed=1234)

This will create three output files in directory "c:/test":
1) PDF file with plots: "example_genes_295_KMoptp_minf_0.10_iter_100.pdf"
2) Tab-delimited text file with the results: "example_genes_295_KMoptp_minf_0.10_iter_100.txt"
3) CSV file with low/high sample labels: "example_genes_295_KMoptp_minf_0.10_iter_100_labels.csv"
```

kmoptscut *For each feature finds a cutoff that optimally stratifies samples into 2 groups, plots Kaplan-Meier survival curves for these groups, and observed vs. expected optimization plot. Does not use the permutation test to estimate the statistical significance of the optimal cutoff.*

Description

For each feature finds a cutoff that optimally stratifies samples into 2 groups, plots Kaplan-Meier survival curves for these groups, and observed vs. expected optimization plot. Does not use the permutation test to estimate the statistical significance of the optimal cutoff.

Usage

```
kmoptscut(fname, sfname, wdir, min_fraction = 0.1, min_up_down = 1,
  peak_tolerance = 0.1, min_uval = 50, psort = FALSE,
  wlabels = TRUE, wpdf = TRUE, verbose = TRUE)
```

Arguments

`fname` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must contain sample ids that match those in 'fname'. The file must contain columns called 'stime' and 'scens', with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding "_KMopt_minf_.2f" and corresponding extension to 'fname'.

`min_fraction` numeric value that specifies the minimal fraction of samples in the smaller group (default is 0.1).

`min_up_down` numeric value that specifies the minimal number of up/down points on either side of the peak for 'pracma::findpeaks' function (default is 1).

`peak_tolerance` numeric value that specifies the maximal difference between in heigth between top peaks. The peak within 'peak_tolerance' closest to the median value is selected.

`min_uval` numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

`psort` logical value whether to sort the output table by p-values in increasing order (default is FALSE).

`wlabels` logical value whether to write a CSV file with low/high (below/above the cutoff) group sample labels (default is TRUE).

`wpdf` logical value whether to write a PDF file with survival curves and plots (default is TRUE).

`verbose` logical value whether to print progress (default is TRUE).

Value

no return value

Examples

```
Basic usage:
kmoptscut(fname="table.txt", sfname="survival.txt", wdir="c:/test")

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

kmoptscut(fname=fdat, sfname=sdat, wdir="c:/test")

This will create three output files in directory "c:/test":
1) PDF file with plots: "example_genes_295_KMopt_minf_0.10.pdf"
2) Tab-delimited text file with the results: "example_genes_295_KMopt_minf_0.10.txt"
3) CSV file with low/high sample labels: "example_genes_295_KMopt_minf_0.10_labels.csv"
```

kmqcut — *For each feature uses the cutoff supplied as quantile (in 0 to 100 range) to stratify samples into 2 groups, plots Kaplan-Meier survival curves, and calculates the log-rank test p-value*

Description

For each feature uses the cutoff supplied as quantile (in 0 to 100 range) to stratify samples into 2 groups, plots Kaplan-Meier survival curves, and calculates the log-rank test p-value

Usage

```
kmqcut(fname, sfname, wdir, quant = 50, min_uval = 50, psort = FALSE,
  wlabels = TRUE, wpdf = TRUE)
```

Arguments

`fname` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must be named 'sample_id' and contain sample ids that match those in 'fname'. The file must contain columns called 'stime' and 'scens', with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding "_KM_quant_d" and corresponding extension to 'fname'.

`quant` numeric value that specifies the cutoff quantile for stratification. The default is 50th quantile (the median).

`min_uval` numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

`psort` logical value whether to sort the output table by p-values in increasing order (default is FALSE).

`wlabels` logical value whether to write a CSV file with low/high (below/above the cutoff) group sample labels (default is TRUE).

`wpdf` logical value whether to write a PDF file with survival curves and plots (default is TRUE).

Value

no return value

Examples

```
Basic usage:
kmqcut(fname="table.txt", sfname="survival.txt", wdir="c:/test", quant=50)

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

kmqcut(fname=fdat, sfname=sdat, wdir="c:/test", quant=50)

This will create three output files in directory "c:/test":
1) PDF file with plots: "example_genes_295_KM_quant_50.pdf"
2) Tab-delimited text file with the results: "example_genes_295_KM_quant_50.txt"
3) CSV file with low/high sample labels: "example_genes_295_KM_quant_50_labels.csv"
```

`kmucut` *For each feature uses the user-supplied cutoff to stratify samples into 2 groups, plots Kaplan-Meier survival curves, and calculates the log-rank test p-value*

Description

For each feature uses the user-supplied cutoff to stratify samples into 2 groups, plots Kaplan-Meier survival curves, and calculates the log-rank test p-value

Usage

```
kmucut(fname, sfname, wdir, cutoff, min_uval = 50, psort = FALSE,
  wlabels = TRUE, wpdf = TRUE)
```

Arguments

`fname` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must be named 'sample_id' and contain sample ids that match those in 'fname'. The file must contain columns called 'stime' and 'scens', with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding "_KM_ucut_.2f" and corresponding extension to 'fname'.

`cutoff` numeric value that specifies the cutoff value for stratification.The same cutoff is applied to every feature in the dataset.

`min_uval` numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

`psort` logical value whether to sort the output table by p-values in increasing order (default is FALSE).

`wlabels` logical value whether to write a CSV file with low/high (below/above the cutoff) group sample labels (default is TRUE).

`wpdf` logical value whether to write a PDF file with survival curves and plots (default is TRUE).

Value

no return value

Examples

```
Basic usage:
kmucut(fname="table.txt", sfname="survival.txt", wdir="c:/test", cutoff=2.5)

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

kmucut(fname=fdat, sfname=sdat, wdir="c:/test", cutoff=5, min_uval=90)

This will create three output files in directory "c:/test":
```

```
1) PDF file with plots: "example_genes_295_KM_ucut_5.pdf"
2) Tab-delimited text file with the results: "example_genes_295_KM_ucut_5.txt"
3) CSV file with low/high sample labels: "example_genes_295_KM_ucut_5_labels.csv"
```

---

kmvalcut

Creates Kaplan-Meier survival curves for each feature from a validation data set by using a file with previously determined stratification thresholds (one threshold per feature), and calculates the log-rank test p-value.

Description

Creates Kaplan-Meier survival curves for each feature from a validation data set by using a file with previously determined stratification thresholds (one threshold per feature), and calculates the log-rank test p-value.

Usage

```
kmvalcut(input1, input2, sfname, wdir, min_uval = 50, psort = FALSE,
  wlabels = TRUE, wpdf = TRUE)
```

Arguments

`input1` character vector that specifies the name of tab-delimited file with the table that contains one or more feature and a stratification threshold for each feature (this table is produced by kmoptscut, kmoptpermcut, kmqcut or kmucut). The file with previously determined stratification thresholds must have first two columns named as 'tracking_id' and 'CUTOFF'. The 'tracking_id' column contains feature names, the 'CUTOFF' column contains stratification threshold for each feature.

`input2` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids. Feature names must exactly match the ones in 'input1' file.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must be named 'sample_id' and contain sample ids that match those in 'fname'. The file must contain columns called 'stime' and 'scens', with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding '_KM_val' and corresponding extension to 'input2'.

`min_uval` numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

`psort` logical value whether to sort the output table by p-values in increasing order (default is FALSE).

`wlabels` logical value whether to write a CSV file with low/high (below/above the cutoff) group sample labels (default is TRUE).

`wpdf` logical value whether to write a PDF file with survival curves and plots (default is TRUE).

Value

no return value

Examples

```

Basic usage:
kmvalcut(input1="cutoffs.txt",input2="validation.txt",sfname="survival.txt",wdir="c:/test")

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

# "example_genes_295_KM_quant_50.txt" is a file with cutoffs created by 'kmqcut'
# and must exist in directory "c:/test"
kmvalcut(input1="example_genes_295_KM_quant_50.txt",input2=fdat,sfname=sdat,wdir="c:/test")

This will create three output files in directory "c:/test":
1) PDF file with plots: "example_genes_295_KM_val.pdf"
2) Tab-delimited text file with the results: "example_genes_295_KM_val.txt"
3) CSV file with low/high sample labels: "example_genes_295_KM_val_labels.csv"
```

transposetable *Transpose a data table: convert rows to columns and columns to rows. Row names will become column names, and column names will become row names.*

Description

Transpose a data table: convert rows to columns and columns to rows. Row names will become column names, and column names will become row names.

Usage

```
transposetable(fnamein, fnameout, wdir)
```

Arguments

`fnamein`	character vector that specifies the name of tab-delimited text file with the input data table.
`fnameout`	character vector that specifies the name of output file where the transposed data table will be saved.
`wdir`	character vector that specifies the name of the working directory for the input/output files.

Value

no return value

Examples

```
Basic usage:
transposetable(fnamein="table.txt", fnameout="transposedtable.txt", wdir="c:/test")

Example with built-in data files:

library(data.table)
library(kmcut)

# Load example gene expression data table for 2 genes
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")

transposetable(fnamein=fdat, fnameout="example_genes_295_transposed.txt", wdir="c:/test")

This will create a tab-delimited text file with the transposed table in directory "c:/test":
"example_genes_295_transposed.txt"
```

ucoxbatch

For each feature in the input file, this function performs a univariate Cox regression with the likelihood ratio test.

Description

For each feature in the input file, this function performs a univariate Cox regression with the likelihood ratio test.

Usage

```
ucoxbatch(fname, sfname, wdir, min_uval = 50, psort = FALSE,
  verbose = TRUE)
```

Arguments

`fname` character vector that specifies the name of the file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

`sfname` character vector that specifies the name of the file with right-censored survival time data. The file must be tab-delimited, where samples are in rows. First column must be named sample_id and contain sample ids that match those in fname. The file must contain columns called stime and scens, with survival time and censoring variable (0 or 1), respectively.

`wdir` character vector that specifies the name of the working directory for the input/output files. Output file names are automatically created by adding "_ucoxbatch.txt" to 'fname'.

`min_uval`	numeric value that specifies the minimal percentage of unique values per feature (default is 50) Features that have less than 'min_uval' percent unique values are excluded from the analysis.
`psort`	logical value whether to sort the output table by p-values in increasing order (default is FALSE).
`verbose`	logical value whether to print progress (default is TRUE).

Value

no return value

Examples

```
Basic usage:
ucoxbatch(fname="table.txt", sfname="survival.txt", wdir="c:/test")

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

ucoxbatch(fname=fdat, sfname=sdat, wdir="c:/test")

This will create a tab-delimited text file with the results in directory "c:/test":
"example_genes_295_ucoxbatch.txt"
```

ucoxpred *For each feature in the input file, this function fits a univariate Cox regression model on a training dataset and then uses the model to predict risk scores for a test dataset.*

Description

For each feature in the input file, this function fits a univariate Cox regression model on a training dataset and then uses the model to predict risk scores for a test dataset.

Usage

```
ucoxpred(fname1, sfname1, fname2, sfname2, wdir, min_uval = 50,
  psort = FALSE, verbose = TRUE)
```

Arguments

fname1 character vector that specifies the name of the training file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

sfname1 character vector that specifies the name of the file with right-censored survival time data for training dataset. The file must be tab-delimited, where samples are in rows. First column must be named sample_id and contain sample ids that match those in fname1. The file must contain columns called stime and scens, with survival time and censoring variable (0 or 1), respectively.

fname2 character vector that specifies the name of the test file with feature(s) for each sample. The file must be tab-delimited, where features are in rows and samples are in columns. First column must contain feature names. Column names must contain sample ids.

sfname2 character vector that specifies the name of the file with right-censored survival time data for test dataset. The file must be tab-delimited, where samples are in rows. First column must be named sample_id and contain sample ids that match those in fname2. The file must contain columns called stime and scens, with survival time and censoring variable (0 or 1), respectively.

wdir character vector that specifies the name of the working directory for the input/output files. Three output files are automatically created by adding:
'_cox_train_sum.txt' to 'fname1', '_train_score.txt' to 'fname1';
'_test_score.txt' to 'fname2';

min_uval numeric value that specifies the minimal percentage of unique values per feature (default is 50). Features that have less than 'min_uval' percent unique values are excluded from the analysis.

psort logical value whether to sort the output table by p-values in increasing order (default is FALSE).

verbose logical value whether to print progress (default is TRUE).

Value

no return value

Examples

```
Basic usage:
ucoxpred(fname1="train.txt", sfname1="survivaltrain.txt", fname2="test.txt", sfname2="survivaltest.txt", wdi

Example with built-in data files:

library(stats)
library(survival)
library(stringr)
library(data.table)
library(tools)
library(pracma)
library(kmcut)

# Load example gene expression data and survival data for 2 genes and 295 samples
fdat = system.file("extdata", "example_genes_295.txt", package="kmcut")
sdat = system.file("extdata", "survival_data_295.txt", package="kmcut")

# Use the example dataset as a training dataset and as a test dataset (perform resubstitution).
# NOTE: In a typical real world application training and test datasets will be different from one another.
ucoxpred(fname1=fdat, sfname1=sdat, fname2=fdat, sfname2=sdat, wdir="c:/test")

This will create three output files in directory "c:/test":
1) Tab-delimited text file with Cox summary for the training data: "example_genes_295_cox_train_sum.txt"
2) Tab-delimited text file with risk scores for training data: "example_genes_295_train_score.txt"
3) Tab-delimited text file with risk scores for test data: "example_genes_295_test_score.txt"
```

Index