Package ‘SIMMS’

January 7, 2021

Version 1.3.1
Type Package
Title Subnetwork Integration for Multi-Modal Signatures
Date 2021-01-07
Author Syed Haider [aut, cre],
             Paul C. Boutros [aut],
             Michal Grzadkowski [ctb]
Maintainer Syed Haider <Syed.Haider@icr.ac.uk>
Imports randomForestSRC (>= 2.9.1)
Depends R (>= 3.2.0), survival (>= 2.36-2), MASS (>= 7.3-12), glmnet
         (>= 1.9-8), doParallel (>= 1.0.10), foreach (>= 1.4.3)
Description Algorithms to create prognostic biomarkers using biological genesets or networks.
License GPL-2
LazyLoad yes
Encoding UTF-8
Suggests knitr (>= 1.4), rmarkdown (>= 0.9.5), xtable (>= 1.7-4)
VignetteBuilder knitr
RoxygenNote 7.1.0
NeedsCompilation no
Repository CRAN
Date/Publication 2021-01-07 15:30:02 UTC

R topics documented:

SIMMS-package .................................................. 2
calculate.meta.survival ...................................... 4
calculate.network.coefficients ............................... 5
calculate.sensitivity.stats .................................. 7
centre.scale.dataset .......................................... 8
create.classifier.multivariate .............................. 8
SIMMS-package

Description

Algorithms to create prognostic biomarkers using biological networks

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>SIMMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Syed Haider, Michal Grzadkowski & Paul C. Boutros

Examples

```r
options("warn" = -1);

# get data directory
```
data.directory <- get.program.defaults(networks.database = "test")[["test.data.dir";]

# initialise params
output.directory <- tempdir();
data.types <- c("mRNA");
feature.selection.datasets <- c("Breastdata1");
training.datasets <- c("Breastdata1");
validation.datasets <- c("Breastdata2");
feature.selection.p.thresholds <- c(0.5);
feature.selection.p.threshold <- 0.5;
learning.algorithms <- c("backward", "forward", "glm");
top.n.features <- 5;

# compute network HRs for all the subnet features
derive.network.features(
  data.directory = data.directory,
  output.directory = output.directory,
  data.types = data.types,
  feature.selection.datasets = feature.selection.datasets,
  feature.selection.p.thresholds = feature.selection.p.thresholds,
  networks.database = "test"
);

# preparing training and validation datasets.
# Normalisation & patientwise subnet feature scores
prepare.training.validation.datasets(
  data.directory = data.directory,
  output.directory = output.directory,
  data.types = data.types,
  p.threshold = feature.selection.p.threshold,
  feature.selection.datasets = feature.selection.datasets,
  datasets = unique(c(training.datasets, validation.datasets)),
  networks.database = "test"
);

# create classifier assessing univariate prognostic power of subnetwork modules (Train and Validate)
create.classifier.univariate(
  data.directory = data.directory,
  output.directory = output.directory,
  feature.selection.datasets = feature.selection.datasets,
  feature.selection.p.threshold = feature.selection.p.threshold,
  training.datasets = training.datasets,
  validation.datasets = validation.datasets,
  top.n.features = top.n.features
);

# create a multivariate classifier (Train and Validate)
create.classifier.multivariate(
  data.directory = data.directory,
  output.directory = output.directory,
  feature.selection.datasets = feature.selection.datasets,
  feature.selection.p.threshold = feature.selection.p.threshold,
  training.datasets = training.datasets,
calculate.meta.survival

Fit a meta-analytic Cox proportional hazards model to a single feature

Description

Takes a meta-analysis data object and fits a Cox proportional hazards model (possibly with adjustment for some specific covariates) by median-dichotomizing patients within each individual dataset.

Usage

calculate.meta.survival(
  feature.name,
  expression.data,
  survival.data,
  rounding = 3,
  other.data = NULL,
  data.type.ordinal = FALSE,
  centre.data = "median"
)

Arguments

feature.name Character indicate what feature (gene/probe/etc.) should be extracted for analysis
expression.data A list where each component is an expression matrix (patients = columns, genes = rows) for a different dataset
calculate.network.coefficients

**survival.data**  A list where each component is an object of class Surv

**rounding**  How many digits after the decimal place to include

**other.data**  A list of other covariates to be passed to the Cox model (all elements in this list are used)

**data.type.ordinal**  Logical indicating whether to treat this datatype as ordinal. Defaults to FALSE

**centre.data**  A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'

**Value**

Returns a vector containing the HR, p-value, n, and 95% confidence limits of the HR (see fit.coxmodel() for details)

**Author(s)**

Paul C. Boutros

**Examples**

```r
data.directory <- get.program.defaults()[["test.data.dir"])]
data.types <- c("mRNA");
x1 <- load.cancer.datasets(
   datasets.to.load = c('Breastdata1'),
   data.types = data.types,
   data.directory = data.directory
);
x2 <- calculate.meta.survival(
   feature.name = "1000_at",
   expression.data = x1$all.data[[data.types[1]]],
   survival.data = x1$all.survobj
);
```

---

**calculate.network.coefficients**

*Calculate Cox statistics for input dataset*

**Description**

Function to compute hazard ratios for the genes in pathway-derived networks, by aggregating input datasets into one training cohort. The hazard ratios are computed for each pair by calculating the HR of each gene independently and as an interaction (i.e. $y = HR(A) + HR(B) + HR(A:B)$)
Usage

```r
calculate.network.coefficients(
  data.directory = ".", 
  output.directory = ".", 
  training.datasets = NULL, 
  data.types = c("mRNA"), 
  data.types.ordinal = c("cna"), 
  centre.data = "median", 
  subnets.file.flattened = NULL, 
  truncate.survival = 100, 
  subset = NULL 
)
```

Arguments

data.directory  Path to the directory containing datasets as specified by training.datasets
output.directory  Path to the output folder where intermediate and results files will be saved
training.datasets  A vector containing names of training datasets
data.types  A vector of molecular datatypes to load. Defaults to c('mRNA')
data.types.ordinal  A vector of molecular datatypes to be treated as ordinal. Defaults to c('cna')
centre.data  A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'
subnets.file.flattened  File containing all the binary interactions derived from pathway-derived networks
truncate.survival  A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation
subset  A list with a Field and Entry component specifying a subset of patients to be selected whose annotation Field matches Entry

Value

Returns a list of matrices for each of the data types. Matrices contain nodes HR/P, edges HR and edges P.

Author(s)

Syed Haider & Paul C. Boutros
**calculate.sensitivity.stats**

*Computes sensitivity measures*

---

**Description**

Computes sensitivity measures: TP, FP, TN, FN, Sensitivity, Specificity, Accuracy

**Usage**

`calculate.sensitivity.stats(all.data = NULL)`

**Arguments**

- `all.data` A data matrix containing predicted and real risk groups

**Value**

A vector containing TP, FP, TN, FN, Sensitivity, Specificity, Accuracy

**Author(s)**

Syed Haider
centre.scale.dataset  Centre and scale a data matrix

Description

Centre and scale a data matrix. Scaling is done on each column separately

Usage

centre.scale.dataset(x = NULL, centre.data = "median")

Arguments

  x  A sample by feature data matrix

  centre.data  A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'

Value

A centred and scaled data matrix

Author(s)

Syed Haider

Examples

  tmp <- matrix(data = rnorm(100, 10, 2), nrow = 20);
  tmp.scaled.median <- centre.scale.dataset(x = tmp);
  tmp.scaled.mean <- centre.scale.dataset(x = tmp, centre.data = "mean");
  tmp.scaled.custom <- centre.scale.dataset(x = tmp, centre.data = 0.3);

create.classifier.multivariate

Trains and tests a multivariate survival model

Description

Trains a model on training datasets. Predicts the risk score for all the training & datasets, independently. This function also predicts the risk score for combined training datasets cohort and validation datasets cohort. The risk score estimation is done by multivariate models fit by fit.survivalmodel. The function also predicts risk scores for each of the top.n.features independently.
create.classifier.multivariate

Usage

create.classifier.multivariate(
  data.directory = "./",  # Path to the directory containing datasets as specified by feature.selection.datasets, training.datasets, validation.datasets
  output.directory = "./",  # Path to the output folder where intermediate and results files will be saved
  feature.selection.datasets = NULL,  # A vector containing names of datasets used for feature selection in function derive.network.features()
  feature.selection.p.threshold = 0.05,  # One of the P values that were used for feature selection in function derive.network.features(). This function does not support vector of P values as used in derive.network.features() for performance reasons
  training.datasets = NULL,  # A vector containing names of training datasets
  validation.datasets = NULL,  # A vector containing names of validation datasets
  top.n.features = 25,  # A numeric value specifying how many top ranked features will be used for univariate survival modelling
  models = c("1", "2", "3"),  # A character vector specifying which of the models ('1' = N+E, '2' = N, '3' = E) to run
  learning.algorithms = c("backward", "forward"),  # A character vector specifying which learning algorithm to be used for model fitting and feature selection. Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  alpha(glm = c(1),  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  k.fold(glm = 10,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  seed.value = 51214,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  cores(glm = 1,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  rf.ntree = 1000,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  rf.mtry = NULL,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  rf.nodesize = 15,  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  rf.samptype = "swor",  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  rf.sampsize = function(x) { x * 0.66 },  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
  ...  # Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
)

Arguments

data.directory  Path to the directory containing datasets as specified by feature.selection.datasets, training.datasets, validation.datasets
output.directory  Path to the output folder where intermediate and results files will be saved
feature.selection.datasets  A vector containing names of datasets used for feature selection in function derive.network.features()
feature.selection.p.threshold  One of the P values that were used for feature selection in function derive.network.features(). This function does not support vector of P values as used in derive.network.features() for performance reasons
training.datasets  A vector containing names of training datasets
validation.datasets  A vector containing names of validation datasets
top.n.features  A numeric value specifying how many top ranked features will be used for univariate survival modelling
models  A character vector specifying which of the models ('1' = N+E, '2' = N, '3' = E) to run
learning.algorithms  A character vector specifying which learning algorithm to be used for model fitting and feature selection. Defaults to c('backward', 'forward'). Available options are: c('backward', 'forward', 'glm', 'randomforest')
alpha.glm  A numeric vector specifying elastic-net mixing parameter alpha, with range alpha ranging from [0,1]. 1 for LASSO (default) and 0 for ridge. For multiple values of alpha, most optimal value is selected through cross validation on training set.

k.fold.glm  A numeric value specifying k-fold cross validation if glm was chosen in learning.algorithms.

seed.value  A numeric value specifying seed for glm k-fold cross or random forest validation if glm was chosen in learning.algorithms.

cores.glm  An integer value specifying number of cores to be used for glm if it was chosen in learning.algorithms.

rf.ntree  An integer value specifying the number of trees in random forest. Defaults to 1000. This should be tuned after starting with a large forest such as 1000 in the initial run and assessing the results in output/OOB_error__TRAINING_* to see where the OOB error rate stabilizes, and then rerunning with the stabilized rf.ntree parameter.

rf.mtry  An integer value specifying the number of variables randomly selected for splitting a node. Defaults to sqrt(features), which is the same as in the underlying R package random survival forest randomForestSRC::rfsrc.

rf.nodesize  An integer value specifying number of unique cases in a terminal node. Defaults to 15, which is the same as in the underlying R package random survival forest randomForestSRC::rfsrc.

rf.samptype  An character string specifying name of sampling. Defaults to sampling without replacement 'swor'. Available options are: c('swor', 'swr')

rf.sampsizex A function specifying sampling size when rf.samptype is set to sampling without replacement ('swor'). Defaults to 66%: function(x){x * .66}

... other params to be passed on to the random forest call to the underlying R package random survival forest randomForestSRC::rfsrc.

Value

The output files are stored under output.directory/output/

Author(s)

Syed Haider & Vincent Stimper

Examples

# see package's main documentation
create.classifier.univariate

Trains and tests a univariate (per subnetwork module) survival model

Description

Trains a model on training datasets. Predicts the risk score for all the training & datasets, independently. This function also predicts the risk score for combined training datasets cohort and validation datasets cohort. The risk score estimation is done by multivariate models fit by `fit.survivalmodel`. The function also predicts risk scores for each of the top.n.features independently.

Usage

```r
create.classifier.univariate(
  data.directory = ".", 
  output.directory = ".", 
  feature.selection.datasets = NULL, 
  feature.selection.p.threshold = 0.05, 
  training.datasets = NULL, 
  validation.datasets = NULL, 
  top.n.features = 25, 
  models = c("1", "2", "3")
)
```

Arguments

- **data.directory**: Path to the directory containing datasets as specified by `feature.selection.datasets`, `training.datasets`, `validation.datasets`
- **output.directory**: Path to the output folder where intermediate and results files will be saved
- **feature.selection.datasets**: A vector containing names of datasets used for feature selection in function `derive.network.features()`
- **feature.selection.p.threshold**: One of the P values that were used for feature selection in function `derive.network.features()`. This function does not support vector of P values as used in `derive.network.features()` for performance reasons
- **training.datasets**: A vector containing names of training datasets
- **validation.datasets**: A vector containing names of validation datasets
- **top.n.features**: A numeric value specifying how many top ranked features will be used for univariate survival modelling
- **models**: A character vector specifying which of the models ('1' = N+E, '2' = N, '3' = E) to run
create.KM.plot

Value

The output files are stored under `output.directory/output/`

Author(s)

Syed Haider

Examples

```r
# see package's main documentation
```

create.KM.plot  

Plots Kaplan-meier survival curve for a given risk grouping & survival params

Description

A generic method to plot KM curves

Usage

```r
create.KM.plot(
  riskgroup = NULL,
  survtime = NULL,
  survstat = NULL,
  file.name = NULL,
  main.title = "",
  resolution = 100
)
```

Arguments

- `riskgroup` A vector containing dichotomized risk groups
- `survtime` A vector containing survival time of the samples
- `survstat` A vector containing survival status of the samples
- `file.name` A string containing full qualified path of the output tiff file
- `main.title` A string specifying main title of the image
- `resolution` A numeric value specifying resolution of the tiff image of KM survival curves. Defaults to 100

Value

The KM survival curves are stored under `output.dir/graphs/`
create.sensitivity.plot

Plots sensitivity analysis for class label dichotomization at supplied survtime cutoffs

Description

A method to compute sensitivity, specificity and accuracy at all the survtime cutoff steps provided

Usage

create.sensitivity.plot(
  riskscore = NULL,
  riskgroup = NULL,
  survtime = NULL,
  survstat = NULL,
  survtime.cutoffs = c(seq(5, 10, 1)),
  output.directory = ".",
  file.stem = NULL,
  main.title = "",
  resolution = 100
)

Arguments

riskscore A vector containing predicted risk scores
riskgroup A vector containing dichotomized risk groups
survtime A vector containing survival time of the samples
survstat A vector containing survival status of the samples
survtime.cutoffs A vector containing cutoff time points used to dichotomize patients into low- and high-risk groups
output.directory Path to the output folder where intermediate and results files will be saved
file.stem A string containing base name for image and text files produced by this method
main.title A string specifying main title of the image
resolution A numeric value specifying resolution of the tiff image of KM survival curves. Defaults to 100

Value

The sensitivity analysis plots are stored under output.directory/graphs/. The sensitivity analysis results are stored under output.directory/output/
create.survivalplots  
Plots Kaplan-meier survival curves

Description

Plots Kaplan-meier survival curves for all the training & datasets, independently as well as combined training datasets cohort and validation datasets cohort. The function also plots KM survival curves for each of the top.n.features independently.

Usage

```r
create.survivalplots(  
data.directory = ".",  
output.directory = ".",  
training.datasets = NULL,  
validation.datasets = NULL,  
top.n.features = 25,  
learning.algorithms = c("backward", "forward"),  
truncate.survival = 100,  
survtime.cutoffs = c(seq(5, 10, 1)),  
main.title = FALSE,  
KM.plotting.fun = "create.KM.plot",  
plot.univariate.data = FALSE,  
plot.multivariate.data = TRUE,  
resolution = 100  
)
```

Arguments

data.directory  Path to the directory containing datasets as specified by training.datasets, validation.datasets  
output.directory  Path to the output folder where intermediate and results files were saved  
training.datasets  A vector containing names of training datasets  
validation.datasets  A vector containing names of validation datasets  
top.n.features  A numeric value specifying how many top ranked features will be used for univariate survival modelling  
learning.algorithms  A character vector specifying which learning algorithm to be used for model fitting and feature selection. Defaults to c("backward", "forward"). Available options are: c("backward", "forward", "glm", "randomforest")
create.survobj

    V
     
     

   truncate.survival
     A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation

survtime.cutoffs
     A vector containing survival cutoff time points to be used for dichotomization of patients into risk groups for sensitivity analysis

main.title
     A logical to specify plot's main title. Defaults to FALSE

KM.plotting.fun
     A string containing the name of the method to use for plotting KM curves. Defaults to create.KM.plot

plot.univariate.data
     Logical to indicate whether to plot univariate results for all subnetworks. Default to FALSE

plot.multivariate.data
     Logical to indicate whether to plot multivariate results for all subnetworks. Defaults to TRUE

resolution
     A numeric value specifying resolution of the png images of KM survival curves. Defaults to 100

Value

The KM survival curves are stored under output.directory/graphs/

Author(s)

Syed Haider

Examples

    # see package's main documentation

create.survobj Utility function for loading meta-analysis lists

Description

Create Surv objects from an annotation-matrix with handling for different time units.

Usage

    create.survobj((annotation = NULL, truncate.survival = 100)
Arguments

annotation  A patient annotation matrix (patients = rows) with (at least) columns for survival, survstat, and survtime.unit

truncate.survival A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation

Value

Returns an object of class Surv

Author(s)

Paul C. Boutros

Examples

annotation.file <- paste(
  get.program.defaults()[["test.data.dir"]],
  "/Breastdata2/patient_annotation.txt", sep = ""
);
annotation <- read.table(
  annotation.file,
  header = TRUE,
  row.names = 1,
  sep = "\t"
);

# select the appropriate survtime and survstat variable for this dataset
annotation$survstat <- annotation[,"Var.dfs"];
annotation$survtime <- annotation[,"Var.tdfs"];
annotation$survtime.unit <- annotation[,"Var.tdfs.unit"];

# only keep samples with survival data
annotation <- annotation[!is.na(annotation$survstat) & !is.na(annotation$survstat),];
surv.obj <- create.survobj((annotation = annotation);

---

derive.network.features

Derive univariate features from pathway-derived networks

Description

This function fits Cox model to features as well as interaction between features. The coefficients of features are subsequently used to compute impact score of each of the pathway-derived networks.
Usage

derive.network.features(
    data.directory = ".",
    output.directory = ".",
    data.types = c("mRNA"),
    data.types.ordinal = c("cna"),
    centre.data = "median",
    feature.selection.fun = "calculate.network.coefficients",
    feature.selection.datasets = NULL,
    feature.selection.p.thresholds = c(0.05),
    truncate.survival = 100,
    networks.database = "default",
    subset = NULL,
    ...
)

Arguments

data.directory Path to the directory containing datasets as specified by feature.selection.datasets
output.directory Path to the output folder where intermediate and results files will be saved
data.types A vector of molecular datatypes to load. Defaults to c(’mRNA’)
data.types.ordinal A vector of molecular datatypes to be treated as ordinal. Defaults to c(’cna’)
centre.data A character string specifying the centre value to be used for scaling data. Valid values are: ‘median’, ‘mean’, or a user defined numeric threshold e.g. ‘0.3’ when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to ‘median’
feature.selection.fun Name of the function to be used to estimate network coefficients. Defaults to ‘calculate.network.coefficients’
feature.selection.datasets A vector containing names of training datasets to be used to compute cox statistics
feature.selection.p.thresholds A vector containing P values to be used as threshold for including features into overall impact score of a network
truncate.survival A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation
networks.database Name of the pathway networks database. Default to NCI PID/Reactome/Biocarta i-e "default"
subset A list with a Field and Entry component specifying a subset of patients to be selected from each dataset whose annotation Field matches Entry
dichotomize.dataset

Dichotomize a single dataset

Description

Split a dataset into two groups by median-dichotomization

Usage

dichotomize.dataset(x, split.at = "median")
**Arguments**

- `x` A vector of values to be dichotomized
- `split.at` An character string or a numeric value that is be used to dichotomize. Valid values are: 'median', 'mean', or a user defined numeric threshold. Defaults to 'median'

**Value**

A vector of the data dichotomized onto a 0/1 (low/high) scale.

**Author(s)**

Syed Haider & Paul C. Boutros

**Examples**

```r
tmp <- rnorm(100);
tmp.groups.median <- dichotomize.dataset(tmp);
tmp.groups.mean <- dichotomize.dataset(tmp, split.at = "mean");
tmp.groups.custom <- dichotomize.dataset(tmp, split.at = 0.3);
```

**Description**

Takes a meta-analysis list (and possibly extra data) and dichotomizes based on a specific gene, then returns the unlisted data to the caller.

**Usage**

```r
dichotomize.meta.dataset(
  feature.name,
  expression.data,
  survival.data,
  other.data = NULL,
  data.type.ordinal = FALSE,
  centre.data = "median"
)
```
dichotomize.meta.dataset

Arguments

feature.name  Character indicate what feature (gene/probe/etc.) should be extracted for analysis
expression.data  A list where each component is an expression matrix (patients = columns, genes = rows) for a different dataset
survival.data  A list where each component is an object of class Surv
other.data  A list of other covariates to be unlisted in the final output (all elements in this list are used)
data.type.ordinal  Logical indicating whether to treat this datatype as ordinal. Defaults to FALSE
centre.data  A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'

Details

NB: other.data handling of missing components (i.e. those present in only some datasets) has not been debugged (but may work regardless).

Value

Returns a list containing components groups (after dichotomization), survtime (in the units of the input data), and survstat. Additional vectors are unlisted from other.data if that parameter is not NULL.

Author(s)

Syed Haider & Paul C. Boutros

Examples

data.directory <- get.program.defaults()[["test.data.dir"]];
data.types <- c("mRNA");
x1 <- load.cancer.datasets(
  datasets.to.load = c('Breastdata1'),
  data.types = data.types,
  data.directory = data.directory
);
x2 <- dichotomize.meta.dataset(
  feature.name = "1000_at",
  expression.data = x1$all.data[[data.types[1]]],
  survival.data = x1$all.survobj
);
fit.coxmodel  

Fit a Cox proportional hazards model

Description

Fit a Cox model (possibly with some linear adjustments) and return key statistics about the fit.

Usage

```r
fit.coxmodel(
  groups,  
survobj,  
stages = NA,  
  rounding = 3,  
  other.data = NULL,  
  data.type.ordinal = FALSE
)
```

Arguments

- `groups`  
  Grouping of patients (passed directly to coxph, so factors & continuous variables are okay)

- `survobj`  
  An object of class Surv (from the survival package) – patient ordering needs to be identical as for groups

- `stages`  
  DEPRECATED! Use other.data instead.

- `rounding`  
  How many digits of precision should be returned?

- `other.data`  
  A data-frame (or matrix?) of variables to be controlled in the Cox model. If null, no adjustment is done. No interactions are fit.

- `data.type.ordinal`  
  Logical indicating whether to treat this datatype as ordinal. Defaults to FALSE

Value

A list containing two elements. `cox.stats` containing a vector or matrix: HR, lower 95% CI of HR, upper 95% CI of HR, P-value (for groups), number of samples (total with group assignments, although some may not be included in fit for other reasons so this is an upper-limit). `cox.obj` containing coxph model object

Author(s)

Syed Haider & Paul C. Boutros
Examples

```
survtime <- sample(seq(0.1,10,0.1), 100, replace = TRUE);
survstat <- sample(c(0,1), 100, replace = TRUE);
survobj <- Surv(survtime, survstat);
groups <- sample(c('A','B'), 100, replace = TRUE);
fit.coxmodel(
  groups = as.factor(groups),
  survobj = survobj
);
```

fit.interaction.model  Cox model two features separately and together

Description

Using a meta-analysis dataset take two features and Cox model them separately and together and extract HRs and p-values.

Usage

```
fit.interaction.model(
  feature1,
  feature2,
  expression.data,
  survival.data,
  data.type.ordinal = FALSE,
  centre.data = "median"
)
```

Arguments

- **feature1**: String indicating what feature (gene/probe/etc.) should be extracted for analysis
- **feature2**: String indicating what feature (gene/probe/etc.) should be extracted for analysis
- **expression.data**: A list where each component is an expression matrix (patients = columns, features = rows) for a different dataset
- **survival.data**: A list where each component is an object of class Surv
- **data.type.ordinal**: Logical indicating whether to treat this datatype as ordinal. Defaults to FALSE
- **centre.data**: A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user-defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'
Details

The interaction model compares cases where feature1 and feature2 concord (both high or both low) to those where they do not. That is, the model is \( y = x_1 + x_2 + (x_1 == x_2) \) and not the typical \( y = x_1 + x_2 + x_1:x_2 \).

Value

Returns a vector of six elements containing (HR,P) pairs for feature1, feature2, and the interaction.

Author(s)

Syed Haider & Paul C. Boutros

Examples

data.dir <- get.program.defaults()["test.data.dir"];
data.types <- c("mRNA");
x1 <- load.cancer.datasets(
    datasets.to.load = c('Breastdata1'),
    data.types = data.types,
    data.directory = data.dir
);
x2 <- fit.interaction.model(
    feature1 = "1000_at",
    feature2 = "2549_at",
    expression.data = x1$all.data[[data.types[1]]],
    survival.data = x1$all.survobj
);
top.n.features = 25,
models = c("1", "2", "3")
)

Arguments

data.directory Path to the directory containing datasets as specified by feature.selection.datasets, training.datasets

output.directory Path to the output folder where intermediate and results files will be saved

feature.selection.datasets A vector containing names of datasets used for feature selection in function derive.network.features()

feature.selection.p.threshold One of the P values that were used for feature selection in function derive.network.features(). This function does not support vector of P values as used in derive.network.features() for performance reasons

training.datasets A vector containing names of training datasets to be used to train multivariate survival model

top.n.features A numeric value specifying how many top ranked features will be used to train the multivariate survival model

models A character vector specifying which models ('1' = N+E, '2' = N, '3' = E) to run

Value

The output files are stored under output.directory/output/

Author(s)

Syed Haider

See Also

create.classifier.multivariate

Examples

# see package's main documentation
get.adjacency.matrix  
A utility function to convert tab delimited networks file into adjacency matrices

Description
A utility function to convert tab-delimited networks file into adjacency matrices

Usage
get.adjacency.matrix(subnets.file = NULL)

Arguments
- subnets.file  
  A tab-delimited file containing networks. New networks start with a new line with '#'. at the beginning of network name and subsequent lines contain a binary interaction per line

Value
A list of adjacency matrices

Author(s)
Syed Haider

Examples
subnets.file <- get.program.defaults()$"subnets.file";
all.adjacency.matrices <- get.adjacency.matrix(subnets.file);

get.chisq.stats  
Applies survdiff function

Description
Applies survdiff on different prognoses groups and computes Logrank P using chisquare statistics.

Usage
get.chisq.stats(groups, survobj)
Arguments

- groups: Grouping of patients (passed directly to survdiff, so factors & continuous variables are okay)
- survobj: An object of class Surv (from the survival package) – patient ordering needs to be identical as for groups

Value

A vector containing: Chisq, degrees of freedom (DOF) and Logrank P-value.

Author(s)

Syed Haider

Examples

```r
survtime <- sample(seq(0.1,10,.01), 100, replace = TRUE);
survstat <- sample(c(0,1), 100, replace = TRUE);
survobj <- Surv(survtime, survstat);
groups <- sample(c('A','B'), 100, replace = TRUE);
get.chisq.stats(
    groups = as.factor(groups),
    survobj = survobj
);
```

get.program.defaults

A utility function to return the inst/ directory of the installed package and other default settings

Description

A utility function to return the inst/ directory of the installed package to get the test datasets and other program related data contents

Usage

```r
get.program.defaults(networks.database = "default")
```

Arguments

- networks.database: Name of the pathway networks database. Default to NCI PID/Reactome/Biocarta i-e "default"

Value

Returns a list of paths to the input directories/files where the contents of this package are installed
load.cancer.datasets

Author(s)

  Syed Haider

Examples

  program.data <- get.program.defaults();

load.cancer.datasets  Load all cancer meta-analysis datasets

Description

  Returns a list of lists containing all cancer meta-analysis datasets

Usage

  load.cancer.datasets(
    tumour.only = TRUE,
    with.survival.only = TRUE,
    truncate.survival = 100,
    datasets.to.load = "all",
    data.types = c("mRNA"),
    datasets.file = "datasets.txt",
    data.directory = ".",
    verbose = FALSE,
    subset = NULL
  )

Arguments

  tumour.only  Logical indicating if we should only load tumour samples (TRUE, the default)
  with.survival.only  Logical indicating if we should only load samples with survival data (TRUE, the default)
  truncate.survival  A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation
  datasets.to.load  A vector of datasets to be loaded. If ‘all’, then all available datasets are loaded
  data.types  A vector of molecular datatypes to load. Defaults to c(‘mRNA’)
  datasets.file  A file in data.directory containing a listing of all usable datasets
  data.directory  A directory containing all data-files to be loaded
  verbose  Logical indicating whether or not status messages should be given
  subset  A list with a Field and Entry component specifying a subset of patients to be selected whose annotation Field matches Entry
**Value**

Returns a meta-analysis list of lists

**Author(s)**

Syed Haider & Paul C. Boutros

**Examples**

```r
data.dir <- get.program.defaults()["test.data.dir"]; x1 <- load.cancer.datasets(
  datasets.to.load = c("Breastdata1"),
  data.types = c("mRNA"),
  data.directory = data.dir
);
```

---

**make.matrix**

Utility function used by `get.adjacency.matrix()`

**Description**

Utility function used by `get.adjacency.matrix()`

**Usage**

```r
make.matrix(vertices, interactions)
```

**Arguments**

- **vertices**: Comma separated list of nodes
- **interactions**: Comma separated list of edges

**Value**

Returns adjacency matrix

**Author(s)**

Syed Haider

**Examples**

```r
x1 <- make.matrix("a,b,c", "a:b,b:c");
```
pred.survivalmodel

Apply a multivariate survival model to validation datasets

Description

Predicts the risk score for all the training & validation datasets, independently. This function also predicts the risk score for combined training datasets cohort and validation datasets cohort. The risk score estimation is done by multivariate models fit by fit.survivalmodel. The function also predicts risk scores for each of the top.n.features independently. TO BE DEPRECATED AND HAS BEEN REPLACED BY create.classifier.multivariate

Usage

pred.survivalmodel(
  data.directory = ".",
  output.directory = ".",
  feature.selection.datasets = NULL,
  feature.selection.p.threshold = 0.05,
  training.datasets = NULL,
  validation.datasets = NULL,
  top.n.features = 25,
  models = c("1", "2", "3"),
  write.risk.data = TRUE
)

Arguments

data.directory  Path to the directory containing datasets as specified by feature.selection.datasets, training.datasets, validation.datasets
output.directory  Path to the output folder where intermediate and results files will be saved
feature.selection.datasets  A vector containing names of datasets used for feature selection in function derive.network.features()
feature.selection.p.threshold  One of the P values that were used for feature selection in function derive.network.features(). This function does not support vector of P values as used in derive.network.features() for performance reasons
training.datasets  A vector containing names of training datasets
validation.datasets  A vector containing names of validation datasets
top.n.features  A numeric value specifying how many top ranked features will be used for univariate survival modelling
models  A character vector specifying which of the models ('1' = N+E, '2' = N, '3' = E) to run
write.risk.data

A toggle to control whether risk scores and patient risk groups should be written to file

Value

The output files are stored under output.directory/output/

Author(s)

Syed Haider

See Also

create.classifier.multivariate

Examples

# see package's main documentation

```r

```
Arguments

- **data.directory**  Path to the directory containing datasets as specified by `datasets`
- **output.directory**  Path to the output folder where intermediate and results files will be saved
- **data.types**  A vector of molecular datatypes to load. Defaults to c('mRNA')
- **data.types.ordinal**  A vector of molecular datatypes to be treated as ordinal. Defaults to c('cna')
- **min.ordinal.threshold**  A named vector specifying minimum percent threshold for each ordinal data type to be used prior to estimating coefficients. Coefficient for features not satisfying minimum threshold will not be estimated, and set to 0. Defaults to cna threshold as 3 percent
- **centre.data**  A character string specifying the centre value to be used for scaling data. Valid values are: 'median', 'mean', or a user defined numeric threshold e.g. '0.3' when modelling methylation beta values. This value is used for both scaling as well as for dichotomising data for estimating univariate betas from Cox model. Defaults to 'median'
- **p.threshold**  Cox P value threshold to be applied for selecting features (e.g. genes) which will contribute to patient risk score estimation. Defaults to 0.5
- **feature.selection.datasets**  A vector containing names of datasets used for feature selection in function `derive.network.features()`
- **datasets**  A vector containing names of all the datasets to be later used for training and validation purposes
- **truncate.survival**  A numeric value specifying survival truncation in years. Defaults to 100 years which effectively means no truncation
- **networks.database**  Name of the pathway networks database. Default to NCI PID/Reactome/Biocarta i.e. "default"
- **write.normed.datasets**  A toggle to control whether processed mRNA and survival data should be written to file
- **subset**  A list with a Field and Entry component specifying a subset of patients to be selected whose annotation Field matches Entry

Value

The output files are stored under `output.directory/output/`

**Author(s)**

Syed Haider
prepare.training.validation.datasets

Examples

# get data directory
data.directory <- get.program.defaults()["test.data.dir"];

# initialise params
output.directory <- tempdir();
data.types <- c("mRNA");
feature.selection.datasets <- c("Breastdata1");
training.datasets <- c("Breastdata1");
validation.datasets <- c("Breastdata1", "Breastdata2");

# preparing training and validation datasets.
# Normalisation & patientwise subnet feature scores
prepare.training.validation.datasets(
data.directory = data.directory,
output.directory = output.directory,
data.types = data.types,
feature.selection.datasets = feature.selection.datasets,
datasets = unique(c(training.datasets, validation.datasets)),
networks.database = "test"
);
Index

* FeatureSelection
  derive.network.features, 16
* IO
  get.program.defaults, 26
  load.cancer.datasets, 27
  prepare.training.validation.datasets, 30
* Kaplan-meier
  create.KM.plot, 12
  create.survivalplots, 14
* Networks
  get.adjacency.matrix, 25
  make.matrix, 28
* Sensitivity
  calculate.sensitivity.stats, 7
* Specificity
  calculate.sensitivity.stats, 7
* accuracy
  create.sensitivity.plot, 13
* center
  centre.scale.dataset, 8
* centre
  centre.scale.dataset, 8
* package
  SIMMS-package, 2
* scale
  centre.scale.dataset, 8
* sensitivity
  create.sensitivity.plot, 13
* specificity
  create.sensitivity.plot, 13
* survival
  calculate.meta.survival, 4
  calculate.network.coefficients, 5
  create.classifier.multivariate, 8
  create.classifier.univariate, 11
  create.KM.plot, 12
  create.sensitivity.plot, 13
  create.survivalplots, 14
  create.survobj, 15
  dichotomize.dataset, 18
  dichotomize.meta.dataset, 19
  fit.coxmodel, 21
  fit.interaction.model, 22
  fit.survivalmodel, 23
  get.chisq.stats, 25
  pred.survivalmodel, 29
  calculate.meta.survival, 4
  calculate.network.coefficients, 5
  calculate.sensitivity.stats, 7
  centre.scale.dataset, 8
  create.classifier.multivariate, 8
  create.classifier.univariate, 11
  create.KM.plot, 12
  create.sensitivity.plot, 13
  create.survivalplots, 14
  create.survobj, 15
  derive.network.features, 16
  dichotomize.dataset, 18
  dichotomize.meta.dataset, 19
  fit.coxmodel, 21
  fit.interaction.model, 22
  fit.survivalmodel, 23
  get.chisq.stats, 25
  pred.survivalmodel, 29
  fit.coxmodel, 21
  fit.interaction.model, 22
  fit.survivalmodel, 23
  get.adjacency.matrix, 25
  get.chisq.stats, 25
  get.program.defaults, 26
  load.cancer.datasets, 27
  make.matrix, 28
  pred.survivalmodel, 29
  prepare.training.validation.datasets, 30
  SIMMS (SIMMS-package), 2
  SIMMS-package, 2

33