Package 'PrimSRC'

January 22, 2015

Type Package

Title PRIM Survival Regression Classification

Version 1.0.0

Date 2015-01-20

Author
Jean-Eudes Dazard [aut, cre], Michael Choe [ctb], Michael LeBlanc [ctb], Alberto Santana [ctb]

Maintainer Jean-Eudes Dazard <jedazard@users.noreply.github.com>

Description Performs Bump Hunting by Patient Rule Induction Method
in Survival, Regression and Classification settings.

Depends R (>= 3.1.2), parallel, survival, Hmisc, MASS

URL https://github.com/jedazard/PrimSRC

Repository PrimSRC, GitHub, Inc.

License GPL (>= 3) | file LICENSE

LazyLoad yes

LazyData yes

Archs i386, x64

R topics documented:

1

PrimSRC-package   *Bump Hunting by Patient Rule Induction Method in Survival, Regression and Classification settings*

Description

Performs a Bump Hunting search by Patient Rule Induction Method. The method generates decision rules delineating a region in the predictor space, where the response is larger than its average over the entire space. The region is shaped as a hyperdimensional box that is not necessarily contiguous. Assumptions are that the multivariate input variables can be discrete or continuous and the univariate response variable can be discrete (Classification), continuous (Regression) or a time-to-event, possibly censored (Survival). It is intended to handle high-dimensional multivariate datasets, where the number of variables far exceeds that of the samples ($p \gg n$ paradigm).

Details

The current version is the initial release (version 1.0.0) that only includes the case of low dimensional situations and a survival response. Ultimately, it will include all the features described above. New features will be added soon as they are available.

Package:	PrimSRC
Type:	Package
Version:	1.0.0
Date:	2015-01-20
License:	GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

The following describes all the end-user functions (not the internal subroutines) that are needed to run a complete procedure. The other internal subroutines are not to be called by the end-user at any time. For computational efficiency, some end-user functions offer a parallelization option that is done by passing a few parameters needed to configure a cluster. This is indicated by an asterisk (* = optionally involving cluster usage). The R functions are categorized as follows:

1. NEWS `PrimSRC.news` **Function to display the NEWS file of the PrimSRC package**

2. END-USER SURVIVAL BUMP HUNTING FUNCTION
   `sbh` (*) **Function for Cross-Validated Survival Bump Hunting**
   Main and unique end-user function for fitting a cross-validated survival bump hunting model. Returns cross-validated "PRSP" object, as generated by our Patient Recursive Survival Peeling or PRSP algorithm at each iteration of the peeling sequence (inner loop of the PRSP algorithm). The object contains cross-validated estimates of all the decision-rules of covariates and other statistical quantities of interest. It also enables displaying results graphically of/for model tuning/selection, all peeling trajectories, variable traces, and survival distributions (see plotting functions below for more details). The function offers a few options such as the type of cross-validation desired (K-fold (replicated)-averaged or-combined), peeling and optimization critera for model fitting, tuning and selection and a few more parameters for the PRSP algorithm. The function takes advantage of the R package **parallel** for efficient parallel execution. It allows users to create a cluster of workstations on a local and/or remote machine(s), enabling scaling-up to the number of specified CPU cores. Discrete (or nominal) variables should be made (or re-arranged into) ordinal variables.

3. END-USER PLOTS FOR MODEL VALIDATION CHECKING AND GRAPHICAL VISU-ALIZATION OF RESULTS
   `plot_profile` **Function for Model Validation Visualization and/or Checking**
   Plot the cross-validated profiles of user's choice of statistics among the Log Hazard Ratio (LHR), Log-Rank Test (LRT), or Concordance Error Rate (CER) as a function of the model tuning parameter, that is, the optimal number of peeling steps of the peeling sequence (inner loop of our PRSP algorithm). Model validation is done by applying the optimization criterion on the user's choice of specific statistic. The goal is to find the optimal value of the K-fold cross-validated number of steps by maximization of LHR or LRT, or minimization of CER. Currently, this done internally for visualization purposes, but it will ultimately offer the option to do be interactive with the end-user as well for parameter choosing/model selection.
   `plot_scatter` **Function for 2D Visualization of Data Scatter and Box Vertices**
   Plot in a plane the cross-validated Data Scatter and Box Vertices at a given peeling step of the peeling sequence (inner loop of our PRSP algorithm). The scatterplot is drawn on a graphical device with geometrically equal scales on the X and Y axes.
   `plot_boxtraj` **Function for Visualization of Peeling Trajectory/Profiles**
   Plot the cross-validated peeling trajectories/profiles of covariates and other statistical quantities of interest at each iteration of the peeling sequence (inner loop of our PRSP algorithm). The plot includes box descriptive statistics (such as support), survival endpoint statistics (such as Maximum Event-Free Time (MEFT), Minimum Event-Free Probability (MEVP), LHR, LRT) and prediction performance (such as CER).
   `plot_boxtrace` **Function for Visualization of Covariates Traces**
   Plot the cross-validated trace curves of covariates variable importance and variable usage at each iteration of the peeling sequence (inner loop of our PRSP algorithm). The top plot shows the overlay of variable importance curves for each covariate The bottom plot shows the overlay of variable usage curves for each covariate. It is a dicretized view of variable importance. Both point to the magnitude and order in which covariates are used along the peeling sequence.
   `plot_boxkm` **Function for Visualization of Survival Distributions**
   Plot the cross-validated Kaplan-Meir estimates of survival distributions for the highest risk (inbox) versus lower-risk (outbox) groups of samples at each iteration of the peeling sequence (inner loop of our PRSP algorithm). Step #0 always corresponds to the situation where the starting box covers the entire test-set data before peeling. Cross-validated LRT, LHR of inbox samples and log-rank p-values of separation are shown at the bottom of the plot with the corresponding peeling step. P-values are lower-bounded by the precision limit given by 1/#number

of permutations (A).

4. END-USER DATASETS

Real Clinical Dataset

Publicly available dataset from the Women's Interagency HIV cohort Study (WIHS). Inclusion criteria of the study were that women at enrolment were (i) alive, (ii) HIV-1 infected, and (iii) free of clinical AIDS symptoms. Women were followed until the first of the following occurred: (i) treatment initiation (HAART), (ii) AIDS diagnosis, (iii) death, or administrative censoring. The studied outcomes were the competing risks "AIDS/Death (before HAART)" and "Treatment Initiation (HAART)". However, here, for simplification purposes, only the first of the two competing events (i.e. the time to AIDS/Death), was used in this dataset example. Likewise, the entire study enrolled 1164 women, but only the complete cases were used in this dataset example for simplification. Variables included history of Injection Drug Use ("IDU") at enrollment, African American ethnicity ("Race"), age ("Age"), and baseline CD4 count ("CD4"). The question in this dataset example is whether it is possible to achieve a prognostication of patients for AIDS and HAART.

Synthetic.1, Synthetic.2, Synthetic.3, Synthetic.4 Four Simulated Survival Models Datasets

Modeling survival models #1-4 with censoring as a regression function of some informative predictors, depending on the model used. In some models, this represents un-informative noisy variables, which are not part of the design matrix (models #2 and #3). In one example, the signal is limited to a box-shaped region R of the predictor space (model #4). Survival time was generated from an exponential model with with rate parameter λ (and mean $\frac{1}{\lambda}$) according to a Cox-PH model with hazard exp(eta), where eta(.) is the regression function. Censoring indicator were generated from a uniform distribution on [0, 3]. In these synthetic examples, all covariates are continuous, i.i.d. from a multivariate uniform distribution on [0, 1].

Known Bugs/Problems : None at this time.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

- "Michael Choe, M.D." <mjc206@case.edu>

- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Acknowledgments: This project was partially funded by the National Institutes of Health NIH - National Cancer Institute (R01-CA160593) to J-E. Dazard and J.S. Rao.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

See Also

- makeCluster (R package **parallel**)
- plot.survfit (R package **survival**)
- wihs (R package **randomForestSRC**)

---

cbindlist	*Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: [cv.pval](#) and [cv.comb.box](#). It is used to bind a list of matrices by columns (even of different number of rows) into a single matrix.

Usage

```
cbindlist(list, trunc)
```

Arguments

list	List of numeric matrices.
trunc	Integer scalar for truncation to the same number of rows.

---

cv.ave.box	*Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: [cv.box.rep](#) and [cv.null](#)

Usage

```
cv.ave.box(x, times, status, K, arg, probval, timeval, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability.
seed	Positive `integer` scalar for the seed.

| cv.ave.fold | *Internal Subroutine (never to be called by end-user)* |

Description

Internal Subroutine of: `cv.ave.box`

Usage

```
cv.ave.fold(x, times, status, K, arg, probval, timeval, seed)
```

Arguments

x	Numeric `matrix` of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
K	Positive `integer` scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- `alpha` = fraction to peel off at each step.
- `beta` = minimum support size resulting from the peeling sequence.
- `minn` = minimum number of observation in a box.
- `L` = fixed peeling length.
- `peelcriterion` in {"hr" (LHR), "lr" (LRT)}.

probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability.
seed	Positive `integer` scalar for the seed.

---

cv.ave.peel	*Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.ave.fold`

Usage

```
cv.ave.peel(x, times, status,
            traindata, trainstatus, traintime,
            testdata, teststatus, testtime,
            K, arg, probval, timeval, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
traindata	Numeric matrix of training covariates.
trainstatus	Logical vector of training event indicators.
traintime	Numeric vector of training survival times.
testdata	Numeric matrix of test covariates.
teststatus	Logical vector of test event indicators.
testtime	Numeric vector of test survival times.
K	Positive `integer` scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability.
seed	Positive `integer` scalar for the seed.

---

cv.box.rep   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: sbh

Usage

```
cv.box.rep(x, times, status,
           B, K, arg, cvtype, probval, timeval,
           parallel, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
B	Positive integer scalar of the number of replications of the cross-validation procedure.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

cvtype	Character vector describing the cross-validation technique in {"none", "averaged", "combined"}.
probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability.
parallel	Logical. Is parallel computing to be performed? Optional.
seed	Positive integer scalar of the user seed to reproduce the results.

---

cv.comb.box *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.box.rep` and `cv.null`

Usage

```
cv.comb.box(x, times, status, K, arg, probval, timeval, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability.
seed	Positive integer scalar for the seed.

---

cv.comb.fold *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.comb.box`

Usage

```
cv.comb.fold(x, times, status, K, arg, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use: • alpha = fraction to peel off at each step. • beta = minimum support size resulting from the peeling sequence. • minn = minimum number of observation in a box. • L = fixed peeling length. • peelcriterion in {"hr" (LHR), "lr" (LRT)}.
seed	Positive integer scalar for the seed.

---

cv.comb.peel   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: cv.comb.fold

Usage

```
cv.comb.peel(x, times, status,
             traindata, trainstatus, traintime,
             testdata, teststatus, testtime,
             K, arg, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
traindata	Numeric matrix of training covariates.
trainstatus	Logical vector of training event indicators.
traintime	Numeric vector of training survival times.
testdata	Numeric matrix of test covariates.
teststatus	Logical vector of test event indicators.
testtime	Numeric vector of test survival times.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use: • alpha = fraction to peel off at each step. • beta = minimum support size resulting from the peeling sequence. • minn = minimum number of observation in a box. • L = fixed peeling length. • peelcriterion in {"hr" (LHR), "lr" (LRT)}.
seed	Positive integer scalar for the seed.

---

cv.folds   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.ave.fold` and `cv.comb.fold`

Usage

```
cv.folds(n, K, seed = NULL)
```

Arguments

n	Integer scalar giving the number of observations to be split into groups.
K	Integer giving the number of groups into which the observations should be randomly split. Setting K also specifies the type of folds to be generated. Possible types are 'random cross-validation', 'leave-one-out cross-validation', or no cross-validation:

- K in $\{2,...,n-1\}$ yields random cross-validation.
- K = 1 yields no cross-validation.
- K = n yields leave-one-out cross-validation.

seed	Integer scalar seed for RNG of random splitting of the data.

---

cv.null   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.pval`

Usage

```
cv.null(x, times, status, cvtype, A, K, arg)
```

Arguments

x	Numeric `matrix` of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
cvtype	Character vector describing the cross-validation technique in {"none", "averaged", "combined"}.
A	Positive `integer` scalar of the number of permutations for the computation of cross-validated p-values. Defaults to 1000.
K	Positive `integer` scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

---

cv.pval   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: sbh

Usage

```
cv.pval(x, times, status, cvtype, A, K, arg, obs.chisq, parallel, conf)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
cvtype	Character vector describing the cross-validation technique in {"none", "averaged", "combined"}.
A	Positive integer scalar of the number of permutations for the computation of cross-validated p-values. Defaults to 1000.
K	Positive integer scalar of the number of folds for the cross-validation procedure.
arg	Character vector describing the parameters to use:
	• alpha = fraction to peel off at each step.
	• beta = minimum support size resulting from the peeling sequence.
	• minn = minimum number of observation in a box.
	• L = fixed peeling length.
	• peelcriterion in {"hr" (LHR), "lr" (LRT)}.
obs.chisq	Numeric vector of observed LRT values at each peeling step.
parallel	Logical. Is parallel computing to be performed? Optional. Defaults to FALSE.
conf	List of parameters for cluster configuration. Inputs for R package **parallel** function makeCluster (R package **parallel**) for cluster setup. Optional, defaults to NULL. See details for usage.

---

endpoints   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `cv.ave.peel` and `cv.comb.peel`. Returns the maximum time value and the corresponding minimum survival probability for every box of the trajectory (box peeling sequence). Also returns the time value and/or the corresponding survival probability depending on what is specified by the user.

Usage

```
endpoints(ind, timemat, probmat, timeval, probval)
```

Arguments

ind	Numeric vector taking on values in {0,1}.
timemat	Numeric matrix of survival times for each peeling step (by rows) and test samples (by columns).
probmat	Numeric matrix of survival probabilities for each peeling step (by rows) and test samples (by columns).
probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time. Defaults to NULL.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability. Defaults to NULL.

---

is.empty   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `endpoints`, `list2array`, `list2mat` and `cbindlist`. Checks if object is empty. Represents the empty array, matrix, or vector of zero dimension or length. Often returned by expressions and functions whose value is undefined. It returns a ogical: TRUE if its argument is empty and FALSE otherwise.

Usage

```
is.empty(x)
```

Arguments

x	array, matrix or vector of any type.

---

lapply.array   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: [sbh](#) and [cv.comb.box](#). Computes FUN element-wise on entries of a list of matrices (even of different row or column numbers).

Usage

```
lapply.array(X, trunc = NULL, sub = NULL,
             fill = NA, MARGIN = 1:2, FUN, ...)
```

Arguments

X	List of numeric matrices.
trunc	Integer scalar for truncation to the same number of rows.
sub	Integer scalar for reaching out to one sub-level of the list. Defaults to NULL.
fill	Numeric scalar to fill in the missing values. Defaults to NA.
MARGIN	Integer vector giving the subscripts which the function will be applied over. E.g., for a matrix 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns. Where X has named dimnames, it can be a character vector selecting dimension names. Defaults to 1:2.
FUN	The function to be applied. In the case of functions like +, %*%, etc., the function name must be backquoted or quoted.
...	Optional arguments to FUN.

---

lapply.mat   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: [sbh](#), [cv.comb.box](#) and [cv.ave.box](#). Compute FUN element-wise on entries of a list of vectors (even of different lengths).

Usage

```
lapply.mat(X, trunc = NULL, sub = NULL,
           fill = NA, MARGIN = 2, FUN, ...)
```

Arguments

X	List of numeric vectors.
trunc	Integer scalar for truncation to the same length.
sub	Integer scalar for reaching out to one sub-level of the list. Defaults to NULL.
fill	Numeric scalar to fill in the missing values. Defaults to NA.
MARGIN	Integer vector giving the subscripts which the function will be applied over. E.g., for a matrix 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns. Where X has named dimnames, it can be a character vector selecting dimension names. Defaults to 2.
FUN	The function to be applied. In the case of functions like +, %*%, etc., the function name must be backquoted or quoted.
...	Optional arguments to FUN.

---

list2array   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: sbh, cv.null and lapply.mat. Used to bind a list of matrices (even of different row or column numbers) by third dimension of a single 3D array.

Usage

```
list2array(list, trunc = NULL, sub = NULL, fill = NA)
```

Arguments

list	List of numeric matrices.
trunc	Integer scalar for truncation to the same number of rows.
sub	Integer scalar for reaching out to one sub-level of the list. Defaults to NULL.
fill	Numeric scalar to fill in the missing values. Defaults to NA.

---

list2mat   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: sbh, cv.null and lapply.mat. Used to bind a list of vectors (even of different length) by rows into a single matrix.

Usage

```
list2mat(list, trunc = NULL, sub = NULL, fill = NA)
```

Arguments

list	List of numeric vectors.
trunc	Integer scalar for truncation to the same length.
sub	Integer scalar for reaching out to one sub-level of the list. Defaults to NULL.
fill	Numeric scalar to fill in the missing values. Defaults to NA.

---

myround   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: sbh. Go to the nearest digit rounding. Note that for rounding off a 0.5, the "go to the even digit" standard is NOT used here.

Usage

```
myround(x, digits = 0)
```

Arguments

x	Numeric vector
digits	Integer scalar indicating the number of decimal points (precision) to be used. Defaults to 0.

---

peel.box   *Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: cv.ave.peel and cv.comb.peel.

Usage

```
peel.box(x, times, status,
         traindata, traintime, trainstatus,
         wt, arg, seed)
```

Arguments

x	Numeric matrix of original covariates.
times	Numeric vector of original survival times.
status	Logical vector of original event indicators.
traindata	Numeric matrix of training covariates.
traintime	Numeric vector of training survival times.
trainstatus	Logical vector of training event indicators.
wt	Numeric vector of weights.

arg   Character vector describing the parameters to use:

- alpha = fraction to peel off at each step.
- beta = minimum support size resulting from the peeling sequence.
- minn = minimum number of observation in a box.
- L = fixed peeling length.
- peelcriterion in {"hr" (LHR), "lr" (LRT)}.

seed   Positive integer scalar of the user seed to reproduce the results.

Details

The maximal number of peeling steps (ncut) is determined either by alpha and beta metaparameters or the smallest possible fraction of the training data, i.e. $\frac{1}{n}$:

- $ceiling(\frac{\log(\mathtt{beta})}{\log(1-\mathtt{alpha})})$ if alpha and beta are fixed by user

- $ceiling(\frac{\log(1/n)}{\log(1-\mathtt{alpha})})$ if alpha is fixed by user and beta is fixed by data

- $ceiling(\frac{\log(\mathtt{beta})}{\log(1-(1/n))})$ if alpha is fixed by data and beta is fixed by user

- $ceiling(\frac{\log(1/n)}{\log(1-(1/n))})$ if alpha and beta are fixed by data

If L is not used to specify a fixed number of peeling steps (i.e. NULL), then beta is used as the stopping rule instead.

---

plot_boxkm   *Function for Visualization of Survival Distributions*

Description

Plot the cross-validated Kaplan-Meir estimates of survival distributions for the highest risk (inbox) versus lower-risk (outbox) groups of samples at each iteration of the peeling sequence (inner loop of our PRSP algorithm). Some plotting parameters are further defined in the function plot.survfit (R package **survival**).

Usage

```
plot_boxkm(peelobj,
           main = NULL, xlab = "Time", ylab = "Probability",
           precision, mark = 3, col = 1, lty = 1, lwd = 1, cex = 1,
           only.last = FALSE, nr = NULL, nc = NULL, ...)
```

Arguments

peelobj   Object of class "PRSP" as generated by the main function [sbh](#).

main   Character vector. Main Title. Defaults to NULL.

xlab   Character vector. X axis label. Defaults to "Time".

ylab   Character vector. Y axis label. Defaults to "Probability".

precision   Precision of cross-validated log-rank p-values of separation between two survival curves. Lower bounded by the value $\frac{1}{A}$ of the number of replication A.

mark	Integer vector of mark parameters, which will be used to label the curves. Defaults to 3.
col	Integer vector of specifying colors for each curve. Defaults to 1.
lty	Integer vector of specifying line types for each curve. Defaults to 1.
lwd	Numeric vector of values for line widths. Defaults to 1.
cex	Numeric scalar specifying the size of the marks. Defults to 1.
only.last	Logical scalar defining whether only the last step of the peeling sequence should be plotted. Defaults to FALSE.
nr	Integer scalar of the number of rows in the plot. If NULL, defaults to 3.
nc	Integer scalar of the number of columns in the plot. If NULL, defaults to 4.
...	Generic arguments passed to other plotting functions, including plot.survfit (R package **survival**).

Details

Step #0 always corresponds to the situation where the starting box covers the entire test-set data before peeling. Cross-validated LRT, LHR of inbox samples and log-rank p-values of separation are shown at the bottom of the plot with the corresponding peeling step. P-values are lower-bounded by the precision limit given by 1/#number of permutations (A).

Value

Invisible. None. Displays the plot(s) on the specified device.

Note

End-user plotting function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>
- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Acknowledgments: This project was partially funded by the National Institutes of Health NIH - National Cancer Institute (R01-CA160593) to J-E. Dazard and J.S. Rao.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).
- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.
- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

See Also

- plot.survfit (R package **survival**)

Examples

```
#===================================================
# Loading the library and its dependencies
#===================================================
library("PrimSRC")

#===================================================
# Loading of the Synthetic and Real datasets
# (see description of datasets)
#===================================================
data("Synthetic.1", "Real", package="PrimSRC")

## Not run:
    #===============================================================================
    # Replicated Combined Cross-Validation (RCCV)
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    #===============================================================================
    postscript(file=paste(getwd(), "/RCCV probability curves for model #1.ps", sep=""),
               horizontal = TRUE, width = 11.5, height = 8.5)
    plot_boxkm(peelobj = CVCOMBREP.synt1,
               main = paste("RCCV probability curves for model #1", sep=""),
               xlab = "Time", ylab = "Probability",
               cex = 1, yscale = 1, precision = 1/CVCOMBREP.synt1$A)
    dev.off()

## End(Not run)
```

---

plot_boxtrace   *Function for Visualization of Covariates Traces*

Description

Plot the cross-validated trace curves of covariates variable importance and variable usage at each iteration of the peeling sequence (inner loop of our PRSP algorithm). Plot up to three "PRSP"" objects.

Usage

```
plot_boxtrace(peelobj, peelobj2 = NULL, peelobj3 = NULL,
              main = NULL, xlab = "Box Mass", ylab = "Variable Range",
              center = FALSE, scale = FALSE, hline = NULL,
              col = 1, col2 = col, col3 = col,
              lty = 1, lty2 = lty, lty3 = lty,
              lwd = 1, lwd2 = lwd, lwd3 = lwd,
              cex = 1, cex2 = cex, cex3 = cex,
              add = FALSE, add.legend = FALSE, text.legend = NULL, ...)
```

Arguments

`peelobj`	Object of class "PRSP" as generated by the main function [sbh]().
`peelobj2`	Object of class "PRSP" as generated by the main function [sbh](). Defaults to NULL.
`peelobj3`	Object of class "PRSP" as generated by the main function [sbh](). Defaults to NULL.
`main`	`Character vector`. Main Title. Defaults to NULL.
`xlab`	`Character vector`. X axis label. Defaults to "Box Mass".
`ylab`	`Character vector`. Y axis label. Defaults to "Variable Range".
`center`	`Logical scalar`. Shall the data be centered before?
`scale`	`Logical scalar`. Shall the data be scaled before?
`hline`	`Numeric scalar` of where the data is centred. If specified (i.e. not NULL), an horizontal line about this value is added to the plot.
`col`	Line color of object peelobj. Defaults to 1.
`col2`	Line color of object peelobj2. Defaults to col.
`col3`	Line color of object peelobj3. Defaults to col.
`lty`	Line type of object peelobj. Defaults to 1.
`lty2`	Line type of object peelobj2. Defaults to lty.
`lty3`	Line type of object peelobj3. Defaults to lty.
`lwd`	Line width of object peelobj. Defaults to 1.
`lwd2`	Line width of object peelobj2. Defaults to lwd.
`lwd3`	Line width of object peelobj3. Defaults to lwd.
`cex`	Symbol expansion. Defaults to 1.
`cex2`	Symbol expansion2. Defaults to cex.
`cex3`	Symbol expansion3. Defaults to cex.
`add`	`Logical scalar`. Should all the PRSP trace plots be overlaid?. Defaults to `FALSE`.
`add.legend`	`Logical scalar`. Should the legend be added to the current open graphics device?. Defaults to `FALSE`.
`text.legend`	`Character vector` of legend content. Defaults to NULL.
`...`	Generic arguments passed to other plotting functions.

Details

The top plot shows the overlay of variable importance curves for each covariate The bottom plot shows the overlay of variable usage curves for each covariate. It is a dicretized view of variable importance. Both point to the magnitude and order in which covariates are used along the peeling sequence.

Value

Invisible. None. Displays the plot(s) on the specified `device`.

Note

End-user plotting function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>
- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).
- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.
- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Examples

```
#==================================================
# Loading the library and its dependencies
#==================================================
library("PrimSRC")


#==================================================
# Loading of the Synthetic and Real datasets
# (see description of datasets)
#==================================================
data("Synthetic.1", "Real", package="PrimSRC")


## Not run:
    #===============================================================================
    # Replicated Combined Cross-Validation (RCCV)
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    #===============================================================================
    if (CVCOMBREP.synt1$plot) {
        postscript(file=paste(getwd(), "/RCCV trace plots for model #1.ps", sep=""),
                   horizontal = FALSE, width = 8.5, height = 5)
        plot_boxtrace(peelobj = CVCOMBREP.synt1,
                      main = paste("RCCV trace plots for model #1", sep=""),
                      center = TRUE, scale = FALSE,
                      ylab = "(centered) Variable Range", hline = c(0,0,0),
                      col = 2:4, lty = c(1,1,1))
        dev.off()
    }


## End(Not run)
```

---

plot_boxtraj   *Function for Visualization of Peeling Trajectory/Profiles*

Description

Plot the cross-validated peeling trajectories/profiles of covariates and other statistical quantities of interest at each iteration of the peeling sequence (inner loop of our PRSP algorithm). Plot up to three "PRSP"" objects.

Usage

```
plot_boxtraj(peelobj, peelobj2 = NULL, peelobj3 = NULL,
             main = NULL, xlab = "Box Mass", ylab = "Variable Range",
             col = 1, col2 = col, col3 = col,
             lty = 1, lty2 = lty, lty3 = lty,
             lwd = 1, lwd2 = lwd, lwd3 = lwd,
             cex = 1, cex2 = cex, cex3 = cex,
             add = FALSE, add.legend = FALSE, text.legend = NULL,
             nr = NULL, nc = NULL, ...)
```

Arguments

peelobj	Object of class "PRSP" as generated by the main function sbh.
peelobj2	Object of class "PRSP" as generated by the main function sbh. Defaults to NULL.
peelobj3	Object of class "PRSP" as generated by the main function sbh. Defaults to NULL.
main	Character vector. Main Title. Defaults to NULL.
xlab	Character vector. X axis label. Defaults to "Box Mass".
ylab	Character vector. Y axis label. Defaults to "Variable Range".
	Plotting parameters for the three "PRSP"" objects:
col	Line color of object peelobj. Defaults to 1.
col2	Line color of object peelobj2. Defaults to col.
col3	Line color of object peelobj3. Defaults to col.
lty	Line type of object peelobj. Defaults to 1.
lty2	Line type of object peelobj2. Defaults to lty.
lty3	Line type of object peelobj3. Defaults to lty.
lwd	Line width of object peelobj. Defaults to 1.
lwd2	Line width of object peelobj2. Defaults to lwd.
lwd3	Line width of object peelobj3. Defaults to lwd.
cex	Symbol expansion. Defaults to 1.
cex2	Symbol expansion2. Defaults to cex.
cex3	Symbol expansion3. Defaults to cex.
add	Logical scalar. Should this plot be added to an open graphics device?. Defaults to FALSE.
add.legend	Logical scalar. Should the legend be added to the current open graphics device?. Defaults to FALSE.

text.legend	Character vector of legend content. Defaults to NULL.
nr	Integer scalar of the number of rows in the plot. If NULL, defaults to 3.
nc	Integer scalar of the number of columns in the plot. If NULL, defaults to 3.
...	Generic arguments passed to other plotting functions.

Details

The plot includes box descriptive statistics (such as support), survival endpoint statistics (such as Maximum Event-Free Time (MEFT), Minimum Event-Free Probability (MEVP), LHR, LRT) and prediction performance (such as CER).

Value

Invisible. None. Displays the plot(s) on the specified device.

Note

End-user plotting function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>
- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Acknowledgments: This project was partially funded by the National Institutes of Health NIH - National Cancer Institute (R01-CA160593) to J-E. Dazard and J.S. Rao.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).
- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.
- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Examples

```
#===================================================
# Loading the library and its dependencies
#===================================================
library("PrimSRC")

#===================================================
# Loading of the Synthetic and Real datasets
# (see description of datasets)
#===================================================
```

```
data("Synthetic.1", "Real", package="PrimSRC")

## Not run:
    #===================================================================================
    # Replicated Combined Cross-Validation (RCCV)
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    #===================================================================================
    if (CVCOMBREP.synt1$plot) {
        postscript(file=paste(getwd(), "/RCCV peeling traj. for model #1.ps", sep=""),
                   horizontal = FALSE, width = 8.5, height = 8.5)
        plot_boxtraj(peelobj = CVCOMBREP.synt1,
                     main = paste("RCCV peeling trajectories for model #1", sep=""),
                     col = 2:4, lty = c(1,1,1))
        dev.off()
    }

## End(Not run)
```

---

plot_profile   *Function for Model Validation Visualization and/or Checking*

Description

Plot the cross-validated profiles of user's choice of statistics among the Log Hazard Ratio (LHR), Log-Rank Test (LRT), or Concordance Error Rate (CER) as a function of the model tuning parameter, that is, the optimal number of peeling steps of the peeling sequence (inner loop of our PRSP algorithm).

Usage

```
plot_profile(peelobj,
             main = NULL, xlab = "Peeling Steps", ylab = "Mean Profiles",
             add.sd = TRUE, add.legend = TRUE, add.profiles = TRUE,
             pch = 20, col = 1, lty = 1, lwd = 2, cex = 2, ...)
```

Arguments

peelobj	Object of class "PRSP" as generated by the main function sbh.
main	Character vector. Main Title. Defaults to NULL.
xlab	Character vector. X axis label. Defaults to "Peeling Steps".
ylab	Character vector. Y axis label. Defaults to "Mean Profiles".
add.sd	Logical scalar. Shall the standard error bars be plotted? Defaults to TRUE.
add.legend	Logical scalar. Shall the legend be plotted? Defaults to TRUE.
add.profiles	Logical scalar. Shall the individual profiles (for all replicates) be plotted? Defaults to TRUE.
pch	Symbol number for all the profiles. Defaults to 20.
col	Line color of the mean profile. Defaults to 1. If more than eight profiles are plotted, line colors will be recycled.

`lty`	Line type of the mean profile. Defaults to 1.
`lwd`	Line width of the mean profile. Defaults to 2.
`cex`	Symbol expansion for all the profiles. Defaults to 2.
`...`	Generic arguments passed to other plotting functions.

Details

Model validation is done by applying the optimization criterion on the user's choice of specific statistic. The goal is to find the optimal value of the K-fold cross-validated number of steps by maximization of LHR or LRT, or minimization of CER. Currently, this done internally for visualization purposes, but it will ultimately offer the option to do be interactive with the end-user as well for parameter choosing/model selection.

Value

Invisible. None. Displays the plot(s) on the specified `device`.

Note

End-user plotiing function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>
- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Acknowledgments: This project was partially funded by the National Institutes of Health NIH - National Cancer Institute (R01-CA160593) to J-E. Dazard and J.S. Rao.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).
- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.
- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Examples

```
#==================================================
# Loading the library and its dependencies
#==================================================
library("PrimSRC")

#==================================================
# Loading of the Synthetic and Real datasets
```

```
# (see description of datasets)
#===================================================
data("Synthetic", "Real", package="PrimSRC")

## Not run:
    #===============================================================================
    # Replicated Combined Cross-Validation (RCCV)
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    #===============================================================================
    postscript(file=paste(getwd(), "/RCCV tuning profiles for model #1.ps", sep=""),
               horizontal = FALSE, width = 8.5, height = 5)
    plot_profile(peelobj = CVCOMBREP.synt1,
                 main = "RCCV tuning profiles for model #1",
                 xlab = "Peeling Steps", ylab = "Mean Profiles",
                 pch=20, col="black", lty=1, lwd=2, cex=2,
                 add.sd = TRUE, add.legend = TRUE, add.profiles = TRUE)
    dev.off()

## End(Not run)
```

---

plot_scatter   *Function for 2D Visualization of Data Scatter and Box Vertices*

Description

Plot in a plane the cross-validated Data Scatter and Box Vertices at a given peeling step of the
peeling sequence (inner loop of our PRSP algorithm).

Usage

```
plot_scatter(peelobj,
             main = NULL, xlab = "", ylab = "",
             proj = c(1,2), splom = TRUE, boxes = TRUE,
             steps = peelobj$cvfit$cv.nsteps,
             add.legend = TRUE, pch = 16, cex = 0.7, col = 1,
             box.col = 2, box.lty = 2, box.lwd = 1, ...)
```

Arguments

peelobj	Object of class "PRSP" as generated by the main function sbh.
main	Character vector. Main Title. Defaults to NULL.
xlab	Character vector. X axis label. Defaults to "".
ylab	Character vector. Y axis label. Defaults to "".
proj	Integer vector of length two, specifying the dimensions of the projection plane. Defaults to c(1,2).
splom	Logical scalar. Shall the scatter plot of points inside the box(es) be plotted? Default to TRUE.

boxes	`Logical` scalar. Shall the box vertices be plotted or just the scatter of points? Default to `TRUE`.
steps	`Integer vector`. Vector of peeling steps at which one wants to plot the box vertices. Defaults to the last peeling step only.
add.legend	`Logical` scalar. Shall the legend of steps numbers be plotted? Defaults to `TRUE`.
pch	Symbol number for the scatter plot. Defaults to 16.
cex	Symbol expansion. Defaults to 0.7.
col	Symbol color. Defaults to 1.
box.col	Line color of the box vertices. Defaults to 2.
box.lty	Line type of box vertices. Defaults to 2.
box.lwd	Line width of box vertices. Defaults to 1.
...	Generic arguments passed to other plotting functions.

Details

The scatterplot is drawn on a graphical device with geometrically equal scales on the X and Y axes.

Value

Invisible. None. Displays the plot(s) on the specified `device`.

Note

End-user plotting function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

- "Michael Choe, M.D." <mjc206@case.edu>

- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Acknowledgments: This project was partially funded by the National Institutes of Health NIH - National Cancer Institute (R01-CA160593) to J-E. Dazard and J.S. Rao.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Examples

```
#==================================================
# Loading the library and its dependencies
#==================================================
library("PrimSRC")

#==================================================
# Loading of the Synthetic and Real datasets
# (see description of datasets)
#==================================================
data("Synthetic.1", "Real", package="PrimSRC")

## Not run:
    #==============================================================================
    # Replicated Combined Cross-Validation (RCCV)
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    #==============================================================================
    postscript(file=paste(getwd(), "/Scatterplot for model #1.ps", sep=""),
               horizontal = FALSE, width = 5, height = 5)
    plot_scatter(peelobj=CVCOMBREP.synt1,
                 main = paste("Scatter plot for model #1", sep=""),
                 proj = c(1,2), splom = TRUE, boxes = TRUE,
                 steps = CVCOMBREP.synt1$cvfit$cv.nsteps,
                 add.legend = TRUE, cex = 0.5,
                 box.col = "red", box.lty = 3, box.lwd = 1)
    dev.off()

## End(Not run)
```

PrimSRC.news   *Function to Display the NEWS File*

Description

Function to display the log file of updates of the **PrimSRC** package.

Usage

```
PrimSRC.news(...)
```

Arguments

...   Further arguments passed to or from other methods.

Value

None.

Note

End-user function.

Author(s)

- "Jean-Eudes Dazard, Ph.D." `<jedazard@users.noreply.github.com>`

- "Michael Choe, M.D." `<mjc206@case.edu>`

- "Michael LeBlanc, Ph.D." `<mleblanc@fhcrc.org>`

- "Alberto Santana, MBA." `<ahs4@case.edu>`

Maintainer: "Jean-Eudes Dazard, Ph.D." `<jedazard@users.noreply.github.com>`

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Real	Real *Clinical Dataset*

Description

Publicly available dataset sfrom the Women's Interagency HIV cohort Study (WIHS). Inclusion criteria of the study were that women at enrolment were (i) alive, (ii) HIV-1 infected, and (iii) free of clinical AIDS symptoms. Women were followed until the first of the following occurred: (i) treatment initiation (HAART), (ii) AIDS diagnosis, (iii) death, or administrative censoring. The studied outcomes were the competing risks "AIDS/Death (before HAART)" and "Treatment Initiation (HAART)". However, here, for simplification purposes, only the first of the two competing events (i.e. the time to AIDS/Death), was used in this dataset example. Likewise, the entire study enrolled 1164 women, but only the complete cases were used in this dataset example for simplification. Variables included history of Injection Drug Use ("IDU") at enrollment, African American ethnicity ("Race"), age ("Age"), and baseline CD4 count ("CD4"). The question in this dataset example is whether it is possible to achieve a prognostication of patients for AIDS and HAART. See Bacon et al. (2005) for more details.

Usage

```
Real
```

Format

Dataset consists of a `numeric` matrix containing $n = 485$ *complete* observations (samples) by rows and $p = 6$ variables by columns ($p < n$ case), including the censoring indicator and (censored) time-to-event variables. Compressed Rda data file.

Author(s)

- "Jean-Eudes Dazard, Ph.D." `<jedazard@users.noreply.github.com>`

- "Michael Choe, M.D." `<mjc206@case.edu>`

- "Michael LeBlanc, Ph.D." `<mleblanc@fhcrc.org>`

- "Alberto Santana, MBA." `<ahs4@case.edu>`

Maintainer: "Jean-Eudes Dazard, Ph.D." `<jedazard@users.noreply.github.com>`

Source

See real clinical data application in Dazard et al., 2015.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

- Bacon M.C, von Wyl V., Alden C. et al. 2005. "*Semi-Supervised Methods to Predict Patient Survival from Gene Expression Data.* Clin Diagn Lab Immunol 12(9): 1013-1019.

See Also

- `wihs` (R package **randomForestSRC**)

---

sbh   *Function for Cross-Validated Survival Bump Hunting*

Description

Main and unique end-user function for fitting a cross-validated survival bump hunting model. Returns cross-validated "PRSP" object, as generated by our Patient Recursive Survival Peeling or PRSP algorithm at each iteration of the peeling sequence (inner loop of the PRSP algorithm).

Usage

```
sbh(dataset, discr, B = 100, K = 5, A = 1000, cpv = FALSE,
    cvtype = "combined", cvcriterion = "lrt",
    arg = "beta=0.05,alpha=0.1,minn=10,L=NULL,peelcriterion=\"lr\"",
    probval = NULL, timeval = NULL,
    parallel = FALSE, conf = NULL, seed = NULL)
```

Arguments

dataset	Input dataset containing the observed survival and status indicator variables in the first two columns, respectively.
discr	Logical vector describing what covariates are discrete. Defaults to logical(ncol(dataset)-2).
B	Positive integer scalar of the number of replications of the cross-validation procedure. Defaults to 100.
K	Positive integer scalar of the number of folds for the cross-validation procedure. Defaults to 5.
A	Positive integer scalar of the number of permutations for the computation of cross-validated p-values. Defaults to 1000.
cpv	logical scalar. Flag for computation of cross-validated p-values. Defaults to FALSE. If computation of cross-validated p-value is deired, then running with the parallelization option is strongly advised, as it may take a while otherwise.
cvtype	Character vector describing the cross-validation technique in {"none", "averaged", "combined"}. Defaults to "combined".
cvcriterion	character vector describing the cross-validation optimization criterion in {"lhr", "lrt", "cer"}. Defaults to "lrt". Automatically set to NULL if cvtype="none".
arg	Character vector describing the parameters to use: • alpha = fraction to peel off at each step. Defaults to 0.1. • beta = minimum support size resulting from the peeling sequence. Defaults to 0.05. • minn = minimum number of observation in a box. Defaults to 10. • L = fixed peeling length. Defaults to NULL. • peelcriterion in {"hr" (LHR), "lr" (LRT)}. Defaults to "lr".
probval	Numeric scalar of the survival probability at which we want to get the endpoint box survival time. Defaults to NULL.
timeval	Numeric scalar of the survival time at which we want to get the endpoint box survival probability. Defaults to NULL.
parallel	Logical. Is parallel computing to be performed? Optional. Defaults to FALSE.
conf	List of parameters for cluster configuration. Inputs for R package **parallel** function makeCluster (R package **parallel**) for cluster setup. Optional, defaults to NULL. See details for usage.
seed	Positive integer scalar of the user seed to reproduce the results.

Details

The PRSP object contains cross-validated estimates of all the decision-rules of covariates and other statistical quantities of interest. It also enables displaying results graphically of/for model tuning/selection, all peeling trajectories, variable traces, and survival distributions (see plotting functions below for more details).

The function offers a number of options for the type of cross-validation desired: K-fold (replicated)-averaged or-combined, as well as peeling and optimization critera for model fitting, tuning and selectio and a few more parameters for the PRSP algorithm.

The function takes advantage of the R package **parallel**, which allows users to create a cluster of workstations on a local and/or remote machine(s), enabling scaling-up with the number of CPU

cores specified and efficient parallel execution. Discrete (or nominal) variables should be made (or re-arranged into) ordinal variables.

The use of a seed allows to reproduce the results within the same type of session: the same seed will reproduce the same results within a non-parallel session or within a parallel session, but it will not necessarily give the exact same results (up to sampling variability) between a non-parallelized and parallelized session due to the different management of the seed between the two (see parallel RNG and value of retuned seed below).

To run a parallel session (and parallel RNG) of the PrimSRC procedures (`parallel=TRUE`), argument `conf` is to be specified (i.e. non NULL). It must list the specifications of the following parameters for cluster configuration: "names", "cpus", "type", "homo", "verbose", "outfile". These match the arguments described in function `makeCluster` of the R package **parallel**. All fields are required to properly configure the cluster, except for "names" and "cpus", which are the values used alternatively in the case of a cluster of type "SOCK" (socket), or in the case of a cluster of type other than "SOCK" (socket), respectively. See examples below.

- "names": `names` : `character` vector specifying the host names on which to run the job. Could default to a unique local machine, in which case, one may use the unique host name "localhost". Each host name can potentially be repeated to the number of CPU cores available on the corresponding machine.
- "cpus": `spec` : `integer` scalar specifying the total number of CPU cores to be used across the network of available nodes, counting the workernodes and masternode.
- "type": `type` : `character` vector specifying the cluster type ("SOCK", "PVM", "MPI").
- "homo": `homogeneous` : `logical` scalar to be set to `FALSE` for inhomogeneous clusters.
- "verbose": `verbose` : `logical` scalar to be set to `FALSE` for quiet mode.
- "outfile": `outfile` : `character` vector of the output log file name for the workernodes.

Note that the actual creation of the cluster, its initialization, and closing are all done internally. In addition, when random number generation is needed, the creation of separate streams of parallel RNG per node is done internally by distributing the stream states to the nodes (For more details see function makeCluster (R package **parallel**) and/or http://www.stat.uiowa.edu/~luke/R/cluster/cluster.html.

Value

Object of class "PRSP" (Patient Recursive Survival Peeling) `List` containing the following 16 fields:

x	numeric `matrix` of original covariates.
times	numeric `vector` of observed failure / survival times.
status	numeric `vector` of observed event indicator in {1,0}.
B	positive `integer` of the number of replications used in the cross-validation procedure.
K	positive `integer` of the number of folds used in the cross-validation procedure.
A	positive `integer` of the number of permutations used for the computation of cross-validated p-values.
cpv	logical scalar returned flag of computation of cross-validated p-values.
cvtype	`character` vector of the cross-validation technique used.
cvcriterion	`character` vector of cross-validation optimization criterion used.
arg	`character` vector of the parameters used:

probval	Numeric scalar of survival probability used.
timeval	Numeric scalar of survival time used.
cvfit	List with 9 fields of cross-validated estimates:

- cv.maxsteps: `numeric scalar` of maximal ceiled-mean of number of peeling steps over the replicates
- cv.nsteps: `numeric scalar` of optimal number of peeling steps according to the optimization criterion
- cv.sign: `binary character vector` in {-1,+1} of direction of peeling for each variable. Invariable for directed peeling (as is the case here).
- cv.trace: `numeric vector` of variable usage traces at each step
- cv.boxind: `logical matrix` in TRUE, FALSE of sample box membership indicator (columns) by peeling steps (rows)
- cv.vertices: `numeric matrix` of box vertices. LB=Lower Bound and UB=Upper Bound by rows. Dimensions by columns
- cv.rules: `data.frame` of decision rules on the variable (columns) by peeling steps (rows)
- cv.stats: `numeric matrix` of box quantities of interest (columns) by peeling steps (rows)
- cv.pval: `numeric vector` of cross-validated log-rank p-values of sepraration of survival distributions

cvprofiles	List (B) of `numeric vectors`, one for each replicate, of the cross-validated statistic used in the optimization criterion (set by user) as function of the number of peeling steps.
plot	`logical` scalar of the returned flag for plotting results (TRUE if CV successful).
seed	User seed(s) used: `integer` of a single value, if parallelization is used `integer vector` of values, one for each replication, if parallelization is not used.

Note

Unique end-user function for fitting the Survival Bump Hunting model.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>
- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).
- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

See Also

- makeCluster (R package **parallel**)

Examples

```
#=================================================
# Loading the library and its dependencies
#=================================================
library("PrimSRC")

#=================================================
# PrimSRC package news
#=================================================
PrimSRC.news()

#===============================================
# PrimSRC package citation
#===============================================
citation("PrimSRC")

#=================================================
# Loading of the Synthetic and Real datasets
# (see description of datasets)
#=================================================
data("Synthetic.1", "Real", package="PrimSRC")

#=================================================
# Simulated dataset example from model #1 (n=250, p=3)
# Replicated Combined Cross-Validation (RCCV)
# Peeling criterion = LRT
# Optimization criterion = LRT
# No parallelization
# No computation of cross-validated p-values
#=================================================
CVCOMBREP.synt1 <- sbh(dataset = Synthetic.1,
                       cvtype = "combined", cvcriterion = "lrt",
                       B = 10, K = 5, cpv = FALSE, probval = 0.5,
                       arg = "beta=0.05,
                              alpha=0.1,
                              minn=10,
                              L=NULL,
                              peelcriterion=\"lr\"",
                       parallel = FALSE, conf = NULL, seed = 777)

# some output results:
CVCOMBREP.synt1$cvfit$cv.maxsteps
CVCOMBREP.synt1$cvfit$cv.nsteps
CVCOMBREP.synt1$cvfit$cv.rules$frame
CVCOMBREP.synt1$cvfit$cv.trace
CVCOMBREP.synt1$cvfit$cv.stats

#=================================================
# Real dataset example (n=485, p=4)
```

```
# Replicated Combined Cross-Validation (RCCV)
# Peeling criterion = LRT
# Optimization criterion = LRT
# No parallelization
# No computation of cross-validated p-values
#===================================================
CVCOMBREP.real <- sbh(dataset = Real,
                      discr = c(0,1,1,0),
                      cvtype = "combined", cvcriterion = "lrt",
                      B = 10, K = 5, cpv = FALSE, probval = 0.5,
                      arg = "beta=0.05,
                             alpha=0.1,
                             minn=10,
                             L=NULL,
                             peelcriterion=\"lr\"",
                      parallel = FALSE, conf = NULL, seed = 777)

# some output results:
CVCOMBREP.real$cvfit$cv.maxsteps
CVCOMBREP.real$cvfit$cv.nsteps
CVCOMBREP.real$cvfit$cv.rules$frame
CVCOMBREP.real$cvfit$cv.trace
CVCOMBREP.real$cvfit$cv.stats

## Not run:
    #================================================================================
    # Replicated Combined Cross-Validation
    # Simulated Synthetic.1 from model #1 (n=250, p=3)
    # Peeling criterion = LRT
    # Optimization criterion = LRT
    # With parallelization
    # With computation of cross-validated p-values
    # For cluster configuration of a:
    #   1- WINDOWS multicore PC => use SOCKET communication
    #   2- LINUX cluster        => use MPI     communication
    #      Here, a file named ".nodes" (e.g. in the home directory)
    #      must contain the list of nodes of the cluster
    #================================================================================
    if (.Platform$OS.type == "windows") {
        cpus <- detectCores()
        conf <- list("names" = rep("localhost", cpus),
                     "cpus" = cpus,
                     "type" = "SOCK",
                     "homo" = TRUE,
                     "verbose" = TRUE,
                     "outfile" = "")
    } else if (.Platform$OS.type == "unix") {
        hosts <- scan(file=paste(Sys.getenv("HOME"), "/.nodes", sep=""),
                      what="",
                      sep="\n")
        hostnames <- unique(hosts)
        nodes <- length(hostnames)
        cpus <-  length(hosts)/length(hostnames)
        conf <- list("cpus" = nodes * cpus,
                     "type" = "MPI",
                     "homo" = TRUE,
                     "verbose" = TRUE,
```

```
                         "outfile" = "")
    } else {
        warning("OS not recognized \n")
    }

    CVCOMBREP.synt1 <- sbh(dataset = Synthetic.1,
                           cvtype = "combined", cvcriterion = "lrt",
                           B = 10, K = 5, A = 1024, cpv = TRUE, probval = 0.5,
                           arg = "beta=0.05,
                                  alpha=0.1,
                                  minn=10,
                                  L=NULL,
                                  peelcriterion=\"lr\"",
                           parallel = TRUE, conf = conf, seed = 777)

    # some output results:
    CVCOMBREP.synt1$cvfit$cv.maxsteps
    CVCOMBREP.synt1$cvfit$cv.nsteps
    CVCOMBREP.synt1$cvfit$cv.rules$frame
    CVCOMBREP.synt1$cvfit$cv.trace
    CVCOMBREP.synt1$cvfit$cv.pval
    CVCOMBREP.synt1$cvfit$cv.stats

## End(Not run)
```

Synthetic.1   Synthetic.1 *Dataset*

Description

Modeling survival model #1 as described in Dazard et al. (2015) with censoring. Here, the regression function uses all of the predictors, which are also part of the design matrix. Survival time was generated from an exponential model with rate parameter λ (and mean $\frac{1}{\lambda}$) according to a Cox-PH model with hazard exp(eta), where eta(.) is the regression function. Censoring indicator were generated from a uniform distribution on [0, 3]. In this synthetic example, all covariates are continuous, i.i.d. from a multivariate uniform distribution on [0, 1].

Usage

```
Synthetic.1
```

Format

Each dataset consists of a `numeric` `matrix` containing $n = 250$ observations (samples) by rows and $p = 5$ variables by columns ($p < n$ case), including the censoring indicator and (censored) time-to-event variables. Compressed Rda data file.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Source

See simulated survival model #1 in Dazard et al., 2015.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Synthetic.2	Synthetic.2 *Dataset*

Description

Modeling survival model #2 as described in Dazard et al. (2015) with censoring. Here, the regression function uses some informative predictors. The rest represent un-informative noisy variables, which are not part of the design matrix. Survival time was generated from an exponential model with rate parameter λ (and mean $\frac{1}{\lambda}$) according to a Cox-PH model with hazard exp(eta), where eta(.) is the regression function. Censoring indicator were generated from a uniform distribution on [0, 3]. In this synthetic example, all covariates are continuous, i.i.d. from a multivariate uniform distribution on [0, 1].

Usage

```
Synthetic.2
```

Format

Each dataset consists of a `numeric matrix` containing $n = 250$ observations (samples) by rows and $p = 5$ variables by columns ($p < n$ case), including the censoring indicator and (censored) time-to-event variables. Compressed Rda data file.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Source

See simulated survival model #2 in Dazard et al., 2015.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Synthetic.3	Synthetic.3 *Dataset*

Description

Modeling survival model #3 as described in Dazard et al. (2015) with censoring. Here, the regression function does not include any of the predictors. This means that none of the variables is informative (noisy), and are not part of the design matrix. Survival time was generated from an exponential model with rate parameter λ (and mean $\frac{1}{\lambda}$) according to a Cox-PH model with hazard exp(eta), where eta(.) is the regression function. Censoring indicator were generated from a uniform distribution on [0, 3]. In this synthetic example, all covariates are continuous, i.i.d. from a multivariate uniform distribution on [0, 1].

Usage

```
Synthetic.3
```

Format

Each dataset consists of a numeric matrix containing $n = 250$ observations (samples) by rows and $p = 5$ variables by columns ($p < n$ case), including the censoring indicator and (censored) time-to-event variables. Compressed Rda data file.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

- "Alberto Santana, MBA." <ahs4@case.edu>

Maintainer: "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>

Source

See simulated survival model #3 in Dazard et al., 2015.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

Synthetic.4   Synthetic.4 *Dataset*

Description

Modeling survival model #4 as described in Dazard et al. (2015) with censoring. Here, the regression function uses all of the predictors, which are also part of the design matrix. In thiss example, the signal is limited to a box-shaped region R of the predictor space. Survival time was generated from an exponential model with rate parameter λ (and mean $\frac{1}{\lambda}$) according to a Cox-PH model with hazard exp(eta), where eta(.) is the regression function. Censoring indicator were generated from a uniform distribution on [0, 3]. In this synthetic example, all covariates are continuous, i.i.d. from a multivariate uniform distribution on [0, 1].

Usage

```
Synthetic.4
```

Format

Each dataset consists of a numeric matrix containing $n = 250$ observations (samples) by rows and $p = 5$ variables by columns ($p < n$ case), including the censoring indicator and (censored) time-to-event variables. Compressed Rda data file.

Author(s)

- "Jean-Eudes Dazard, Ph.D." <jedazard@users.noreply.github.com>
- "Michael Choe, M.D." <mjc206@case.edu>
- "Michael LeBlanc, Ph.D." <mleblanc@fhcrc.org>

Source

See simulated survival model #4 in Dazard et al., 2015.

References

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2015). "*Cross-validation and Peeling Strategies for Survival Bump Hunting using Recursive Peeling Methods.*" (Submitted).

- Dazard J-E., Choe M., LeBlanc M. and Rao J.S. (2014). "*Cross-Validation of Survival Bump Hunting by Recursive Peeling Methods.*" In JSM Proceedings, Survival Methods for Risk Estimation/Prediction Section. Boston, MA, USA. American Statistical Association IMS - JSM, p. 3366-3380.

- Dazard J-E. and J. S. Rao (2010). "*Local Sparse Bump Hunting.*" J. Comp Graph. Statistics, 19(4):900-92.

---

updatecut	*Internal Subroutine (never to be called by end-user)*

Description

Internal Subroutine of: `peel.box` Update the splitpoints in the calling function.

Usage

```
updatecut(x, fract)
```

Arguments

x	numeric `matrix` of training covariates.
fract	numeric scalar quantile.

Index