Package 'subscreen'

July 19, 2017

Type Package

Title Systematic Screening of Study Data for Subgroup Effects

Version 0.2.2

Author Hermann Kulmann, Bodo Kirsch, Susanne Lippert, Thomas Schmelter

Maintainer Bodo Kirsch <bodo.kirsch@bayer.com>

Description Systematically screens study data for subgroup effects and visualizes these.

License GPL

LazyData TRUE

Imports utils, plyr, data.table, grDevices, graphics

Suggests parallel, shiny, survival, DT

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-07-19 14:04:46 UTC

R topics documented:

`subscreen-package` *This package allows analyzing a large number of subgroups simultaneously. The workflow for this subgroup screening is split into two parts:*

Description

(i) calculation of the results for the different subgroups (subscreencalc) and (ii) visualization (subscreenshow).

Details

For the calculation of the subgroup results, a patient level dataset needs to be provided, which (for each patient) contains the endpoint data, treatment assignment and all categorical baseline characteristics, by which the subgroups are defined. The results for all subgroups that can be defined by the combination of up to a fixed number of factors, which is only limited by the computational time, will be calculated and stored for further analysis, e.g., visualization. In the second step these results will visualized in a Shiny based interactive graphical user interface, called Subgroup Explorer.

---

subscreencalc *Systematic screening of study data for subgroups*

Description

This function systematically calculates the defined outcome for every combination of subgroups up to the given level (max_comb), i.e. the number of maximum combinations of subgroup defining factors. If, e.g., in a study sex, age group (<=60, >60), BMI group (<=25, >25) are of interest, subgroups of level 2 would be, e.g, male subjects with BMI>25 or young females, while subgroups of level 3 would be any combination of all three variables.

Usage

```
subscreencalc(data, eval_function, endpoints, treat = "trtp",
  subjectid = "subjid", factors = NULL, min_comb = 1, max_comb = 3,
  nkernel = 1, par_functions = "", verbose = T)
```

Arguments

data	dataframe with study data
eval_function	name of the function for data analysis
endpoints	vector containing the names of the endpoint variables in data
treat	name of variable in data that contains the treatment identfier, defaults to trtp
subjectid	name of variable in data that contains the subject identifier, defaults to subjid
factors	vector containg the names of variables that define the subgroups, defaults to NULL. If set to NULL, all variables in data are used that are not included in subjectid, treat, and endpoints
min_comb	minimum number of factor combination levels to define subgroups, defaults to 1
max_comb	maximum number of factor combination levels to define subgruops, defaults to 3

nkernel	number of kernels for parallelization (defaults to 1)
par_functions	vector of names of functions used in eval_function to be exported to cluster (needed only if nkernel > 1)
verbose	switch on/off output of computational information

Details

The evaluation function (eval_function) has to defined by the user. The result needs to be a vector of numerical values, e.g., outcome variable(s) and number of observations/subjects. The input of eval_function is a data frame with the same structure as the input data frame (data) used in the subsreencalc call. See example below. Potential errors occurring due to small subgroups should be caught and handled within eval_function.

Value

an object of type SubScreenResult of the form list(sge=H, max_comb=max_comb, min_comb=min_comb, subjectid=subjectid, endpoints=endpoints, treat=treat, factors=factors, results_total=eval_function(cbind(F,T)))

Examples

```
# get the pbc data from the survival package
require(survival)
data(pbc, package="survival")

# generate categorical versions of some of the baseline covariates
pbc$ageg <- pbc$age <= median(pbc$age)
pbc$albuming <- pbc$albumin <= median(pbc$albumin)
pbc$phosg <- pbc$alk.phos <= median(pbc$alk.phos)
pbc$astg <- pbc$ast <= median(pbc$ast)
pbc$bilig <- pbc$bili <= median(pbc$bili)
pbc$cholg <- pbc$chol <= median(pbc$chol)
pbc$copperg <- pbc$copper <= median(pbc$copper)

# redefine censoring variable, consider transplant/death as event
pbc$event <- pbc$status
pbc$event[pbc$event==2] <- 1

pbcdat <- pbc[!is.na(pbc$trt), ]

# define function the eval_function()
hazardratio <- function(x) {
   hr <- tryCatch(exp(coxph(Surv(time, event) ~ trt, data=x)$coefficients[[1]]),
                  warning=function(w) {NA})
   N1 <- sum(x$trt==1)
   N2 <- sum(x$trt==2)

   data.frame(N1=N1, N2=N2, hr=hr)
 }

 # run subscreen
```

```
results <- subscreencalc(data=pbcdat,
                     eval_function=hazardratio,
                     endpoints = c("time", "event"),
                     treat="trt",
                     subjectid = "id",
                     factors=c("ageg", "sex", "bilig", "cholg", "copperg", "astg",
                               "albuming", "phosg"))

# visualize the results of the subgroup screening with a Shina app
## Not run: subscreenshow(results, PreSelectTarge="hr", PreSelectXAxis="N1")
```

---

subscreenshow *Systematic screening of study data for subgroups*

Description

Start the Shiny based intactive visualization tool to show the subgroup results generated by subscreencalc.

Usage

```
subscreenshow(scresults, host = NULL, port = NULL, ColorPoint = "white",
  ColorClicked = "red", ColorSelected = "green", ColorParent = "orange",
  ColorReference = "darkorange", ColorBG = "#424242",
  ColorBGplot = "#424242", ColorText = "#6b6b6b", PreSelectScale = "lin",
  PreSelectTarget = "", PreSelectXAxis = "", pickradius = 5)
```

Arguments

scresults	SubScreenResult object with results from a subscreencalc call
host	host name or IP address for Shiny display
port	port number for Shiny display
ColorPoint	Color for the points displaying the subgroup results (defaults to white)
ColorClicked	Color of picked points (defaults to red)
ColorSelected	Color selected (filtered) subgroups (defaults to green)
ColorParent	Color of parent subgroup (defaults to orange)
ColorReference	Color of reference line (defaults to darkorange)
ColorBG	Background color of the Shiny app
ColorBGplot	Background color of the plot
ColorText	Color of the legend text
PreSelectScale	Scale of y axis "lin" or "log"
PreSelectTarget	Preselected target variable (y-axis)
PreSelectXAxis	Variable for x-axis
pickradius	radius (in pixel) for interactive selection of subgroups

Index