Package ‘bhm’

July 29, 2019

Type Package
Title Biomarker Threshold Models
Version 1.15
Date 2019-07-29
Author Bingshu E. Chen
Maintainer Bingshu E. Chen <bingshu.chen@queensu.ca>
Depends R (>= 2.10.0), coda, MASS, survival
Imports methods
Description Contains tools to fit both predictive and prognostic biomarker effects using biomarker threshold models. Evaluate the treatment effect, biomarker effect and treatment-biomarker interaction using probability index measurement. Test for treatment-biomarker interaction using residual bootstrap method.
License GPL (>= 2)
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2019-07-29 21:30:02 UTC

R topics documented:

bhm-package .................................................. 1
bhm ............................................................... 3
bhmControl ...................................................... 5
data ............................................................. 7
mpl ............................................................... 8
pIndex .......................................................... 10
pIndexControl .................................................. 12
plot ............................................................ 13
print ........................................................... 14
resboot ......................................................... 15
**Description**

This package fits biomarker threshold regression models for predictive and prognostic biomarker effects with binary data and survival data with an unknown biomarker cutoff point (Chen et al, 2014)<DOI:10.1016/j.csda.2013.05.015>. Multivariable models can also be fitted for adjusted biomarker effect (Fang et al, 2017)<DOI:10.1016/j.csda.2017.02.011>. Tools such as Probability index are included to measure treatment effect, biomarker effect or treatment-biomarker interaction(Jiang et al, 2016)<DOI:10.1002/sim.6907>.

**Details**

"bhm" is a R package for Biomarker Threshold Models. Please use the following steps to install the most recent version of 'bhm' package:
1. First, you need to install the 'devtools' package. You can skip this step if you have 'devtools' installed in your R. Invoke R and then type

```r
install.packages("devtools")
```

2. Load the devtools package.

```r
library(devtools)
```

3. Install "bhm" package from github with R commond

```r
install_github("statapps/bhm")
```

"bhm" uses different statistical methods to identify cut-point (threshold parameter) for the biomarker in either generalized linear models or Cox proportional hazards model.

A stable version of View the "bhm" package is also available from the Comprehensive R Archive Network (https://CRAN.R-project.org/package=bhm) and can be installed using R command

```r
install.packages("bhm")
```

**Author(s)**

Bingshu E. Chen

Maintainer: Bingshu E. Chen <bingshu.chen@queensu.ca>

**References**


bhm


See Also

coxph, glm, survival

Examples

# fit = bhm(y~biomarker+treatment)
# print(summary(fit))

---

bhm            Fitting Biomarker Threshold Models

Description

[bhm] is a R package for Biomarker Threshold Models. It uses either Hierarchical Bayes method or profile likehood method (Chen, et al, 2014 and Tian, et al, 2017) to identify a cut-point (thrseshold parameter) for the biomarker in either generalized linear models or Cox proportional hazards model. The model is specified by giving a symbolic description of the linear predictor and a description of the distribution family.

Usage

bhm(x, ...)

## S3 method for class 'formula'
bhm(formula, family, data, control = list(...),...)

# use
# bhm(y ~ biomarker)
# to fit a prognostic model with biomarker term only
# use
# bhm(y ~ biomarker+treatment)
# to fit a predictive model with interaciton between biomarker
# and treatment, use
# bhmFit(x, y, family, control)
# to fit a model without the formula
#
# Biomarker shall be in the first dependent variable
Arguments

formula

- an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.

family

- a description of the response distribution and link function to be used in the model. The available family function are either "binomial" for fitting a logistic regression model or "surv" for fitting a Cox proportional hazards model

data

- an optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which glm is called.

x, y

- For "bhmFit", x is a design matrix of dimension n * p and y is a vector of observations of length n for "glm" models or a "Surv" survival object for "coxph" models.

control

- a list of parameters for controlling the fitting process. See "bhmControl" for details

... additional arguments to be passed to the low level regression fitting functions (see below).

Details

'bmarker' is a Biomarker variable. This variable is required and shall be the first dependent variable in the formula.

"interaction" is an option of fitting model with itneractin term. When interaction = TRUE, a predictive biomarker model will be fitted. When interaction = FALSE, a prognostic biomarker model will be fitted. Both Biomarker and Treatment variables are required if 'interaction' = TRUE and 'treatment' shall be the second variable in the formula.

"bhmFit" and "bhmGibbs" are the workhorse functions: they are not normally called directly but can be more efficient where the response vector, design matrix and family have already been calculated.

"x.cdf" is a function that maps biomarker values to interval (0, 1) using its empirical cumulative distribution function. After the threshold parameters are identified, the biomarker variable will be transformed back to its original scale.

Value

bhm returns an object of class inheriting from "bhm" which inherits from the class glm or 'coxph'. See later in this section.

The function "summary" (i.e., "summary.bhm") can be used to obtain or print a summary of the results, for example, the 95

An object of class "bhm" is a list containing at least the following components:

c.max

- a vector of the mean estimates for the threshold parameter(s)

coefficients

- a named vector of coefficients from 'bhm'

c.fit

- fitted conditional regression model given c = c.max

cg

- Gibbs sample for threshold parameter c

bg

- Gibbs sample for the coefficients beta
Note

The logistic regression part are based on codes wrote by Tian Fang.

Author(s)

Bingshu E. Chen (bingshu.chen@queensu.ca)

References


See Also

glm, coxph, bhmControl

Examples

```r
## Generate a random data set
n = 300
b = c(0.5, 1, 1.5)
data = surv.gendat(n, c0 = 0.40, beta = b)
age = runif(n, 0, 1)*100
tm = data[, 1]
status = data[, 2]
trt = data[, 3]
ki67 = data[, 4]
## fit a biomarker threshold survival model with one single cut point
fit = bhm(Surv(tm, status)~ki67+trt+age, interaction = TRUE, B=5, R=10)

## here B=5 and R=10 is used for test run. In general, B > 500 and R > 2000 is
## recommend for the analysis of biomarker variable. To fit a model with
## two cut points, use:
##
## fit = bhm(Surv(tm, status)~bmk+trt+age, B = 500, R = 2000, c.n = 2)
##
## To print the output, use
##
## print(fit)
```
**bhmControl**  
*Auxiliary function for bhm fitting*

**Description**

Auxiliary function for bhm fitting. Typically only used internally by 'bhmFit', but may be used to construct a control argument to either function.

**Usage**

```r
bhmControl(method = 'Bayes', interaction, biomarker.main, alpha, B, R, thin, epsilon, c.n, beta0, sigma0)
```

**Arguments**

- `method` : choose either 'Bayes' for Bayes method with MCMC or 'profile' for profile likelihood method with Bootstrap. The default value is 'Bayes'
- `interaction` : an option of fitting model with interaction term. When interaction = TRUE, a predictive biomarker model will be fitted. When interaction = FALSE, a prognostic biomarker model will be fitted. The default value is interaction = TRUE.
- `biomarker.main` : include biomarker main effect, default is TRUE
- `B` : number of burn in
- `R` : number of replications for Bayes method or number of Bootstrap for profile likelihood method
- `thin` : thinning parameter for Gibbs samples, default is 2
- `epsilon` : biomarker (transformed) step length for profile likelihood method, default is 0.01
- `alpha` : significance level (e.g. alpha=0.05)
- `c.n` : number of threshold (i.e. the cut point), default is 1
- `beta0` : initial value for mean of the prior distribution of beta, default is 0
- `sigma0` : initial value for variance of the prior distribution of beta, default is 10000

**Details**

Control is used in model fitting of "bhm".

**Value**

This function checks the internal consisteny and returns a list of value as inputed to control model fit of bhm.

**Note**

Based on code from Tian Fang.
To fit a prognostic model for biomarker with two cut-points, 500 burn-in samples and 10000 Gibbs samples,

```r
ctl = bhmControl(interaction = FALSE, B = 500, R = 10000, c.n = 2)
```

then fit the following model

```r
# fit = bhmFit(x, y, family = 'surv', control = ctl)
```

---

**Description**

dataset for biomarker threshold model (bhm)

**Usage**

- to generate survival data, use:
  ```r
  surv.gendat(n, c0, beta)
  ```
- to generate glm data, use:
  ```r
  glm.gendat(n, c0, beta)
  ```

**Arguments**

- `n` sample size
- `c0` cut off point, for example `c0 = 0.4`
- `beta` regression coefficient, for example, `beta = c(0.3, log(0.5), log(0.25))`

**Format**

The format of the data set for analysis shall be a data frame with a response variable (either a Surv object for Cox model or a glm response variable object) and at least one dependent variable as the biomarker variable.
**mpl**

*Joint models for clustered data with robust variance*

**Details**

data set of prostate cancer in the 'survival' package is used as an example in paper by Chen, et al. (2014).

**Source**

prostate dataset can be loaded with 'library(survival)'.

**References**


**Examples**

```r
#data(data)
## maybe str(data) ; plot(data) ...
c0 = 0.4
b = c(-0.5, 1.5, 1.3)
data = surv.gendat(n=300, c0 = c0, beta = b)
```

**Description**

`mpl` is a function to fit Joint model for clustered binary and survival data using maximum penalized likelihood (MPL) with robust Jackknife variance.

**Usage**

```r
mpl(formula, ...)
```

## S3 method for class 'formula'

```r
mpl(formula, formula.glm, formula.cluster, data, weights=NULL, subset = NULL, max.iter=100, tol = 0.005, jackknife=TRUE, ...)
```

##use:

```r
# fit = mpl(Surv(time, status) ~ w + z, y~x, ~cluster, data = data)
```
Arguments

- **formula** an object of class "formula" (or one that can be coerced to that class): a symbolic description of the Cox proportional hazards model to be fitted for survival data.

- **formula.glm** an object of class "formula" (or one that can be coerced to that class): a symbolic description of the generalized linear model to be fitted for binary data.

- **formula.cluster** an object of class "formula" (or one that can be coerced to that class): a symbolic description of the cluster variable.

- **data** an optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which mpl is called.

- **weights** To be added in the future

- **subset** To be added in the future

- **max.iter** Maximum number of iterations, default is max.iter = 100

- **tol** Tolerance for convergence, default is tol = 0.005

- **jackknife** Jackknife method for variance, default is jackknife = TRUE

- **...** additional arguments to be passed to the low level regression fitting functions (see below).

Details

mpl(y~x, Surv(time, event)~w+z, ~cluster) will fit penalized likelihood for binary and survival data. Function print(x) can be used to print a summary of mpl results.

Value

mpl returns an object of class inheriting from "mpl". When jackknife = TRUE, an object of class "mpl" is a list containing the following components:

- **theta** the maximum estimate of the regression coefficients and variance component

- **OR_HR** Odds ratios (OR) and hazard ratios (HR) for binary and survival outcomes, respectively

- **ase** Asymptotic standard error for theta, which is usually underestimated

- **jse** Jackknife standard error of theta based on resampling, this is considered to be more robust

Note

Based on code from J. Wang.

Author(s)

Bingshu E. Chen (bingshu.chen@queensu.ca)
References


See Also

glmcoxph

Examples

## No run
#
# fit = mpl(y~x, Surv(tm, status) ~ trt + ki67, ~center.id)
#

---

pIndex

Probability Index for Survival Time Difference

Description

[pIndex] is a function to estimate and test difference of survival time among groups. It is defined as $p = \Pr\{T_1 < T_2\}$, where $T_1$ is survival time for subjects in group 1 and $T_2$ is survival time in group 2.

Usage

pIndex(x, ...)

## S3 method for class 'formula'
pIndex(formula, data, control = list(...), ...)

### To estimate probability index for treatment and control groups (define by trt):
#
# fit = pIndex(Surv(time, status) ~ trt)
#
### To estimate probability index difference for treatment and control groups (define by trt) between biomarker positive and biomarker negative subjects (i.e. Treatment-biomarker interaction):
#
# fit = pIndex(Surv(time, status) ~ trt+biomarker)
#

Arguments

formula an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
pIndex

data

an optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which pIndex is called.

x

Here covariate x is a design matrix of dimension n * 1 (for two sample test) or dimension n * 2 (for treatment * biomarker interaction).

control

a list of parameters for controlling the fitting process. See 'pIndexControl' for details

... additional arguments to be passed to the low level regression fitting functions (see below).

Details

pIndex(y~x) will estimate probability index of two groups (eg. treatment vs control) defined by x. pIndex(y~x1 + x2) will estimate the difference of probability index of x1 (eg. treatment vs control) between biomarker positive and biomarker negative groups (x2). Function print(x) can be used to print a summary of pIndex results.

Value

pIndex returns an object of class inheriting from "pIndex". When B > 0, an object of class "pIndex" is a list containing at least the following components:

theta

the estimated probability index

theta.b

Bootstrap or Jackknife sample of the probability index

sd

standard deviation of theta based on resampling

ci

(1-alpha) percent confidence interval based on resampling

Note

This function is part of the bhm package.

Author(s)

Bingshu E. Chen (bingshu.chen@queensu.ca)

References


See Also

bhm,pIndexControl,
Examples

```r
## Generate a random data set
n = 300
b = c(0.5, 1, 1.5)
data = surv.gendat(n, c0 = 0.40, beta = b)
age = runif(n, 0, 1)*100
tm = data[, 1]
status = data[, 2]
trt = data[, 3]
ki67 = data[, 4]

# No run

# fit = pIndex(Surv(tm, status) ~ trt + ki67)
```

---

**pIndexControl**  
*Auxiliary function for pIndex fitting*

**Description**

Auxiliary function for pIndex fitting. Typically only used internally by `pIndexFit`, but may be used to construct a control argument to either function.

**Usage**

```r
pIndexControl(method = c("Efron", "Elc", "Elw", "Pic"),
               model = c("default", "local", "threshold"),
               ci = c("Bootstrap", "Jackknife"), weights = NULL,
               kernel = NULL, h = 0.1, w = seq(0.05, 0.95, 0.05),
               alpha = 0.05, B = 0, pct = 0.5, tau=NULL)
```

**Arguments**

- **method**: choose either ‘Efron’ for Efron method, ‘Elc’ for conditional empirical likelihood, ‘Elw’ for weighted empirical likelihood method, and ‘Pic’ for piecewise exponential distribution. The default value is ‘Efron’
- **model**: ‘default’ for default pIndex model, ‘local’ for kernel method, ‘threshold’ for threshold method
- **ci**: Method to construct confidence interval, ‘Bootstrap’ for Bootstrap method and ‘Jackknife’ for Jackknife method
- **weights**: case weight
- **kernel**: kernel function types, including "gaussian", "epanechnikov", "rectangular", "triangular", "biweigt", "cosine", "optcosine". The default value is ‘gaussian’
- **h**: bandwidth, default is 0.1
Control is used in model fitting of 'pIndex'.

This function checks the internal consistency and returns a list of value as inputed to control model fit of pIndex.

Based on code from Bingshu E. Chen.

Bingshu E. Chen

bhm, pIndex

## To calculate the probability index for a biomarker with conditional empirical likelihood
## and the corresponding 90 percent CI using Bootstrap method with 10000 bootstrap sample

ctl = pIndexControl(method = 'Elc', ci = 'Bootstrap', B = 10000, alpha = 0.1)

## then fit the following model
##
# fit = pIndex(y~x1 + x2, family = 'surv', control = ctl)
##
plot

Plot a fitted biomarker threshold model

Description

Several different type of plots can be produced for biomarker threshold models. Plot method is used to provide a summary of outputs from "bhm", "pIndex", "resboot".

Use "methods(plot)" and the documentation for these for other plot methods.

Usage

## S3 method for class 'bhm'
plot(x, type = c("profile", "density"), ...)
## S3 method for class 'pIndex'
plot(x, ...)
## S3 method for class 'resboot'
plot(x, ...)

Arguments

x
a class returned from "bhm", "pIndex" or "resboot" fit.

type
type of plot in bhm object, "profile" to plot profile likelihood, "density" to plot trace and density of the threshold distribution.

...other options used in plot().

Details

plot.bhm is called to plot either the profile likelihood function or the threshold density function.

plot.pIndex is called to plot local probability index (pIndex) of a continuous biomarker.

plot.resboot is called to plot the bootstrap distribution of the likelihood ratio test statistics for biomarker threshold models (resboot).

The default method, plot.default has its own help page. Use methods("plot") to get all the methods for the plot generic.

Author(s)

Bingshu E. Chen

See Also

The default method for plot

plot.default, glm.bhm, pIndex, resboot
Examples

# plot(fit)
#
# plot for bhm object
#
# plot(fit, type = 'density')
#
print

print a fitted object or a summary of fitted object

Description

print and summary are used to provide a short summary of outputs from "bhm", "pIndex", "resboot".

Usage

## S3 method for class 'bhm'
print(x, ...)  
## S3 method for class 'mpl'
print(x, digits = 3, ...)  
## S3 method for class 'pIndex'
print(x, ...)  
## S3 method for class 'resboot'
print(x, ...)  
## S3 method for class 'summary.bhm'
print(x, ...)  
## S3 method for class 'bhm'
summary(object, ...)

Arguments

x a class returned from bhm, pIndex or resboot fit

digits number of digits to be printed

... other options used in print()

object object returned from model fit

Details

print.bhm is called to print object or summary of object from the biomarker threshold models bhm. print.pIndex is called to print object or summary of object from the probability index model pIndex. print.resboot is called to print object or summary of object from the residual bootstrap method for biomarker threshold models resboot. summary(fit) provides detail summary of ‘bhm’ model fit, including parameter estimates, standard errors, and 95 percent CIs.

The default method, print.default has its own help page. Use methods("print") to get all the methods for the print generic.
Author(s)
Bingshu E. Chen

See Also
The default method for print print.default. Other methods include glm, bhm, pIndex, resboot.

Examples
#
# print(fit)
#

------

resboot
  Rresidual Bootstrap Test (RBT) for treatment-biomarker interaction

Description
resboot is a function to test the existence of treatment-biomarker interaction in biomarker threshold model
g(Y) = b0+b1*I(w>c) + b2*z + b3*I(w>c)*z.

Usage
resboot(x, ...)

## S3 method for class 'formula'
resboot(formula, family, data=list(...), B = 100, epsilon = 0.01, ...)
#
###To test the null hypothesis of interaction between treatment variable
###(define by z) and biomarker variables (define by w) for survival dataa,
###use:
###
# fit = resboot(Surv(time, status) ~ w + z + w:z)
#

Arguments

formula an object of class "formula"(or one that can be coerced to that class): a symbolic
description of the model to be fitted. The details of model specification are given
under 'Details'.
family default is family = 'Surv' for survival data.
data an optional data frame, list or environment (or object coercible by 'as.data.frame'
to a data frame) containing the variables in the model. If not found in data, the
variables are taken from environment(formula), typically the environment from
which resboot is called.
Here covariate $x$ is a design matrix of dimension $n \times 1$ (for two sample test) or dimension $n \times 2$ (for treatment * biomarker interaction).

Number of bootstraps, default is $B = 100$

Biomarker (transformed) step length for profile likelihood method, default is $\epsilon = 0.01$

additional arguments to be passed to the low level regression fitting functions (see below).

Details

resboot(y~w + z + w:z) will give residual bootstrap p-value for interaction between biomarker variable (w) and treatment variable (z). The null hypothesis is given by $H_0: b_3 = 0$, where $b_3$ is the regression coefficient for the interaction term $l(w>c)*z$. Function print(x) can be used to print a summary of resboot results.

Value

resboot returns an object of class inheriting from "resboot". When $B > 0$, an object of class "resboot" is a list containing at least the following components:

- theta: the estimated maximum of likelihood ratio statistics
- theta.b: Bootstrap sample of theta
- sd: standard deviation of theta based on resampling
- ci: (1-alpha) percent confidence interval for theta based on resampling

Note

Based on code from Parisa Gavanji.

Author(s)

Bingshu E. Chen (bingshu.chen@queensu.ca)

References


See Also

bhm coxph

Examples

```r
##
## Generate a random data set
n = 300
b = c(0.5, 1, 1.5)
data = surv.gendat(n, c0 = 0.40, beta = b)
```
tm = data[, 1]
status = data[, 2]
trt = data[, 3]
ki67 = data[, 4]
#
### No run
#
# fit = resboot(Surv(tm, status) ~ trt + ki67)
#