Package ‘rolr’

March 21, 2017

Type Package

Title Finding Optimal Three-Group Splits Based on a Survival Outcome

Version 1.0.0

Author Pingping Qu and John Crowley

Maintainer Pingping Qu <quping14@gmail.com>

Description Provides fast procedures for exploring all pairs of cutpoints of a single covariate with respect to survival and determining optimal cutpoints using a hierarchical method and various ordered logrank tests.

License GPL-3

Encoding UTF-8

LazyData true

Imports stats, survival

RoxygenNote 5.0.1

Suggests knitr, rmarkdown, testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2017-03-21 06:51:14 UTC

R topics documented:

`rhier` *Finding Optimal Cutpoints Using a Hierarchical Method*

Description

Using a hierarchical method, `rhier` is used to find two optimal cutpoints to divide the entire dataset into three groups based on a continuous covariate and a survival outcome. Making use of the running logrank test (`rlr`), the method first identifies an optimal cutpoint that gives the largest logrank statistic to split into two groups, and then repeats the process in each of the resulting groups to find additional two cutpoints. It then takes the cutpoint that gives the larger test statistic between the two as the second optimal cutpoint.

Usage

```
rhier(times, status, x, ns = 15, alt = "decrease", method = "approximate")
```

Arguments

`times` Survival outcome.

`status` Censoring indicator which takes 1 when an event occurs at end of study and 0 otherwise.

`x` A continuous covariate.

`ns` Minimum number of subjects in each group after dichotomizing the covariate.

`alt` A character that takes either `"decrease"` or `"increase"` to represent a positive or negative association between the covariate and survival.

`method` A character that takes either `"approximate"` or `"exact"` where an approximate or exact method will be used.

Value

Returns a list with one element being the two optimal cutpoints obtained.

References

See main package help page.

See Also

`rsolr12`, `rmolr`

Examples

```
library(rolr)

#simulate data with true underlying cutpoints and hazard goes up as covariate goes up
d=simdata(nn = 150, hr = c(1, 2, 3), hazard.func = "step",
          props=c(1/3, 1/3, 1/3), censoring.rate = 0)

#finding optimal cutpoints using alt = 'decrease' option
res=rhier(times=d$times, status=d$status, x=d$x, ns=15, alt='decrease')

#do it again using alt = 'increase', the results are the same as earlier
#because it doesn't matter what you choose for the alt option
res2=rhier(times=d$times, status=d$status, x=d$x, ns=15, alt='increase')
```

rlr *Calculating Running Logrank Test*

Description

rlr is used to calculate a logrank test for every two groups obtained from dichotomizing a continuous covariate x at a particular point. It will examine all values in x except the first and last ns points.

Usage

```
rlr(times, status, x, ns = 15, trend = "decrease", method = "approximate")
```

Arguments

times
: Survival outcome.

status
: Censoring indicator which takes 1 when an event occurs at the end of a study and 0 otherwise.

x
: A continuous covariate.

ns
: Minimum number of subjects in each group, whether it is the group with x < cutpoint, or the group with x >= cutpoint.

trend
: A character that takes either "decrease" or "increase" to represent a positive or negative relationship between the covariate and survival.

method
: A character that takes either "approximate" or "exact" where an approximate or exact method will be used to calculate the running logrank test.

Details

When the association is positive, that is, larger covariate values leading to worse survival, and you enter trend = "decrease", the test statistics will be positive, but if you enter trend = "increase" the test statistics will be negative. Opposite is true when the association is negative. You want to make sure to enter the option so that the resulting test statistics are positive.

Value

A matrix of four columns as the following -

xcutoff - All cutpoints that have been used to dichotomize the sample (that is, all values of the covariate x except the first and last ns points)

L - Numerators of the logrank z tests for all cutpoints considered.

V - Denominators of the logrank z tests for all cutpoints considered.

logrank.stat - The logrank z tests for all cutpoints considered.

References

See main package help page.

Examples

```
library(rolr)

##### -------- Example 1

#simulate survival where hazard increases as covariate increases
d=simdata(nn = 150, hr.linear = 2, hazard.func = "linear", censoring.rate = 0)

#using trend = 'decrease', the test statistics are positive, which is good
res=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='decrease')
head(res)

#do it again with trend = 'increase', now the test statistics are negative.
#So you want to switch to trend='decrease'.
res2=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='increase')
head(res2)

#Note that the test statistics are the same except the signs
res[,'logrank.stat']+res2[,'logrank.stat']

#do it with exact method, how close is it to the approximate method?
res3=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='decrease',
         method="exact")
cor(res[,'logrank.stat'], res3[,'logrank.stat'])

##### -------- Example 2

#Simulate survival where hazard decreases as covariate increases
d=simdata(nn = 150, hr.linear = 1/3, hazard.func = "linear", censoring.rate = 0)

#using trend = 'decrease', and the test statistics are negative, which
#is not good
res=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='decrease')
head(res)

#do it again with trend = 'increase', now the test statistics are positive,
#which is good
```

```
res2=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='increase')
head(res2)

#Note that the test statistics are the same except the signs
res[,'logrank.stat']+res2[,'logrank.stat']

#do it with exact method, how close is it to the approximate method?
res3=rlr(times=d$times, status=d$status, x=d$x, ns=15, trend='increase',
         method="exact")
cor(res[,'logrank.stat'], res3[,'logrank.stat'])
```

`rmolr` *Finding Optimal Cutpoints Using Modified Ordered Logrank (MOL) Tests*

Description

Using the modified ordered logrank test (MOL), the `rmolr` function finds two optimal cutpoints to divide the entire dataset into three groups based on a continuous covariate and a survival outcome. It is a fast procedure that makes use of the running logrank test (`rlr`) to improve on computing speed.

Usage

```
rmolr(times, status, x, ns = 15, alt = "decrease", method = "approximate")
```

Arguments

`times` Survival outcome.

`status` Censoring indicator which takes 1 when an event occurs at end of study and 0 otherwise.

`x` A continuous covariate.

`ns` Minimum number of subjects in each group after dichotomizing the covariate.

`alt` A character that takes either `"decrease"` or `"increase"` to represent a positive or negative association between the covariate and survival.

`method` A character that takes either `"approximate"` or `"exact"` where an approximate or exact method will be used.

Details

When the true association is positive, that is, larger covariate values lead to worse survival, and you enter `alt = "decrease"`, the test statistics will be positive, but if you enter `trend = "increase"` the test statistics will be negative. Opposite is true when the true association is negative. You want to make sure to enter the option so that the resulting test statistics are positive.

Value

Returns a list with two elements, with the first being the test statistics for all cutpoints considered and the second being the best splits from the MOL tests.

References

See main package help page.

See Also

rsolr12, rhier

Examples

```
library(rolr)

##### -------- Example 1

#simulate data with true cutpoints and hazard goes up as covariate goes up
d=simdata(nn = 150, hr = c(1, 2, 3), hazard.func = "step",
          props=c(1/3, 1/3, 1/3), censoring.rate = 0)

#using alt = 'decrease', the test statistics are positive, so the results
#are correct.
res=rmolr(times=d$times, status=d$status, x=d$x, ns=15, alt='decrease')
names(res)

#do it again using alt = 'increase', now the test statistics are negative
#so the results are not right. So you have to switch back to alt='decrease'
#to get positive statistics and the correct optimal cutpoints here.
res2=rmolr(times=d$times, status=d$status, x=d$x, ns=15, alt='increase')
names(res2)

##### -------- Example 2

#Simulate data with true cutpoints and hazard goes down as covariate goes up
d=simdata(nn = 150, hr = c(3, 2, 1), hazard.func = "step",
          props=c(1/3, 1/3, 1/3), censoring.rate = 0)

#using alt = 'decrease', the test statistics are negative and so the results
#are not right.
res=rmolr(times=d$times, status=d$status, x=d$x, ns=15, alt='decrease')
res[['best.splits.molr']]

#do it again using alt = 'increase', now the test statistics are positive
#and thus the results are correct.
res2=rmolr(times=d$times, status=d$status, x=d$x, ns=15, alt='increase')
res2[['best.splits.molr']]
```

rolr — *rolr: A package for computing optimal three-group splits based on a continuous covariate and a survival outcome using ordered logrank tests (OLR)*

Description

The rolr package provides three main functions, `rsolr12`, `rmolr`, and `rhier`, for fast computation of three-group optimal cutpoints using two simple OLR tests, a modified OLR test, and a hierarchical method, respectively.

Author(s)

Pingping Qu and John Crowley

References

Crowley J., Mitchell A., Qu P., Morgan G. and Barlogie B. Optimal three group splits based on a survival outcome. In: Frontiers of Biostatistical Methods and Applications in Clinical Oncology, 2017.

Liu P. Y. and Tsai W. Y. A modified logrank test for censored survival data under order restrictions. Statistics and probablility Letters 41:57-63, 1999.

Liu P. Y., Tsai W. Y. and Wolf, M. Design and analysis for survival data under order restrictions with a modified logrank test. Statistics in medicine 17:1469-1479, 1998.

Crowley J., LeBlanc M., Gentleman R. and Salmon, S. Exploratory methods in survival analysis. In: IMS Laecture Notes 27:55-77, 1995.

LeBlanc M. and Crowley J. Survival trees by goodness of split. Journal of the American Statistical Association 88:457-467, 1993.

Liu P. Y., Greeen S., Wolf M. and Crowley J. Testing against ordered alternatives for censored survival data. Journal of the American Statistical Association 88:163-160, 1993.

Crowley J. Some extensions of the logrank test. In: Medical Informatics. D. A. B. Lindberg and P. L. Reicherts, Eds. Lecture Notes 4:213-223, 1979.

rsolr12 — *Finding Optimal Cutpoints Using Simple Ordered Logrank (SOL) Tests*

Description

Using two simple ordered logrank tests (SOL-1 and SOL-2), the `rsolr12` function finds two optimal cutpoints to divide the entire dataset into three groups based on a continuous covariate and a survival outcome. It is a fast procedure that makes use of the running logrank test (`rlr`) to improve on computing speed.

Usage

```
rsolr12(times, status, x, ns = 15, alt = "decrease",
  method = "approximate")
```

Arguments

`times` Survival outcome.

`status` Censoring indicator which takes 1 when an event occurs at end of study and 0 otherwise.

`x` A continuous covariate.

`ns` Minimum number of subjects in each group after dichotomizing the covariate.

`alt` A character that takes either `"decrease"` or `"increase"` to represent a positive or negative association between the covariate and survival.

`method` A character that takes either `"approximate"` or `"exact"` where an approximate or exact method will be used.

Details

When the association is positive, that is, larger covariate values leading to worse survival, and you enter `alt = "decrease"`, the test statistics will be positive, but if you enter `trend = "increase"` the test statistics will be negative. Opposite is true when the association is negative. You want to make sure to enter the option so that the resulting test statistics are positive.

Value

Returns a list with three elements, the first one being the test statistics for all cutpoints considered (except the first and last `ns` points), and the second and third elements being the best splits obtained from using the SOL-1 and SOL-2 tests.

References

See main package help page.

See Also

`rmolr`, `rhier`

Examples

```
library(rolr)

##### -------- Example 1

#simulate data with 2 underlying true cutpoints and hazard goes up as x goes up
d=simdata(nn = 150, hr = c(1, 2, 3), hazard.func = "step",
         props=c(1/3, 1/3, 1/3), censoring.rate = 0)

#using alt = 'decrease', the test statistics are positive, so it is good
```

```
res=rsolr12(times=d$times, status=d$status, x=d$x, ns=15, alt='decrease')
names(res)
res[['best.splits.solr1']]
res[['best.splits.solr2']]

#do it again using alt = 'increase', now the test statistics are negative and
#so the results are not right. So you have to switch back to alt='decrease' to
#get positive statistics and the correct optimal cutpoints here.
res2=rsolr12(times=d$times, status=d$status, x=d$x, ns=15, alt='increase')
res2[['best.splits.solr1']]
res2[['best.splits.solr2']]

##### -------- Example 2

#simulate data with true cutpoints and hazard goes down as covariate goes up
d=simdata(nn = 150, hr = c(3, 2, 1), hazard.func = "step",
          props=c(1/3, 1/3, 1/3), censoring.rate = 0)

#using alt = 'decrease', the test statistics are negative (so the results
#are not right).
res=rsolr12(times=d$times, status=d$status, x=d$x, ns=15, alt='decrease')
res[['best.splits.solr1']]
res[['best.splits.solr2']]

#do it again using alt = 'increase', now it is right
res2=rsolr12(times=d$times, status=d$status, x=d$x, ns=15, alt='increase')
res2[['best.splits.solr1']]
res2[['best.splits.solr2']]
```

simdata *Simulating Survival Times as Functions of a Single Covariate*

Description

simdata is used to simulate survival data from an exponential distribution. When the hazard function is a step function, we assume 3 underlying groups obtained by applying two cutpoints x1 and x2 to the covariate so that group 1 is x < x1, group 2 is x >= x1 and x < x2, and group 3 is x >= x2. The hazard is a function of the covariate x simulated from a uniform distribution from [0, 2]; it can be either a linear function, a step function (with three groups), or a constant (in which case no association exists between the covariate and survival).

Usage

```
simdata(nn = 300, const = 365, hr = c(1, 2, 3), hr.linear = 3,
  props = c(1/3, 1/3, 1/3), hazard.func = "step", censoring.rate = 0,
  seed = 1)
```

Arguments

`nn` Sample size.

`const` A constant that all of the hazard functions will be divided by. The bigger it is, the longer the survival times will be. Default is 365.

`hr` A three-element vector representing the hazards for each of the groups 1 to 3 when the `hazard.func = "step"`.

`hr.linear` A scalar representing the hazard ratio when the covariate increases by one unit. This is used with `hazard.func = "linear"`.

`props` A three-element vector representing the proportions of groups 1 to 3 when `hazard.func = "step"`.

`hazard.func` A character that can take either `'step'`, `'linear'`, or `'none'` to represent a step, linear or no association between the covariate and survival, respectively. When it is `"step"`, the entire set is divided into 3 groups based on the covariate with group proportions specified in the `props` argument.

`censoring.rate` The amount of censoring desired. Default = 0.

`seed` The random seed used to generate the data. Default = 1.

Value

A data frame with survival times (times), censoring indicator (status), covariate (x), three groups obtained by cutting the covariate if `hazard.func = "step"` (x3), and censoring rate (censoring.rate).

Examples

```
library(rolr)

#simulate survival with a step hazard function
d1=simdata(nn = 150, hr = c(1, 2, 3), props = c(1/3, 1/3, 1/3),
           hazard.func = "step", censoring.rate = 0)
head(d1)

#simulate survival with a linear hazard function
d2=simdata(nn = 150, hr.linear = 2, hazard.func = "linear", censoring.rate = 0)
head(d2)

#simulate survival with no association with the covariate
d3=simdata(nn = 150, hazard.func = "none", censoring.rate = 0)
head(d3)
```

Index