Package ‘uni.survival.tree’

March 22, 2021

Type Package

Title A Survival Tree Based on Stabilized Score Tests for
High-dimensional Covariates

Version 1.5

Author Takeshi Emura and Wei-Chern Hsu

Maintainer Takeshi Emura <takeshiemura@gmail.com>

Description
A classification (decision) tree is constructed from survival data with high-dimensional covariates.
The method is a robust version of the logrank tree, where the variance is stabilized.
The main function ``uni.tree'' returns a classification tree for a given survival dataset.
The inner nodes (splitting criterion) are selected by minimizing the P-value of the two-sample the score tests.
The decision of declaring terminal nodes (stopping criterion) is the P-value threshold given by an argument (specified by user).
This tree construction algorithm is proposed by Emura et al. (2021, in review).

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Depends survival,compound.Cox

NeedsCompilation no

Repository CRAN

Date/Publication 2021-03-22 06:40:02 UTC

R topics documented:

feature.selected *The names of features that are selected in a tree*

Description

The function returns the names of features (covariates) that are selected as the internal nodes of a tree. Only the names of the covariates are shown by excluding the cutt-off values.

Usage

```
feature.selected(tree)
```

Arguments

`tree`	:an object made from the "uni.tree" function

Details

The outputs show important features for predicting survival outcomes.

Value

An array of characters that are the names from those covariates selected in the tree

Examples

```
data(Lung,package=”compound.Cox”)
train_Lung=Lung[which(Lung[,”train”]==TRUE),] #select training data
t.vec=train_Lung[,1]
d.vec=train_Lung[,2]
x.mat=train_Lung[,-c(1,2,3)]
res=uni.tree(t.vec,d.vec,x.mat,P.value=0.01,d0=0.01,S.plot=FALSE,score=TRUE)
feature.selected(res)
```

KM.split *Kaplan-Meier estimator of binary splitting*

Description

Given a cut-off-point and selected covariate, return the survival curve for binary classification and the P-value of two sample log-rank test.

Usage

```
KM.split(t.vec, d.vec, X.mat, x.name, cutoff)
```

Arguments

`t.vec` :Vector of survival times (time to either death or censoring)

`d.vec` :Vector of censoring indicators (1=death, 0=censoring)

`X.mat` :n by p matrix of covariates, where n is the sample size and p is the number of covariates

`x.name` :the name of covariate

`cutoff` :cut-off-point

Value

P-value of two sample logrank test and a plot of two KM estimates

Examples

```
data(Lung,package="compound.Cox”)
train_Lung=Lung[which(Lung[,"train”]==TRUE),] #select training data
t.vec=train_Lung[,1]
d.vec=train_Lung[,2]
x.mat=train_Lung[,-c(1,2,3)]
KM.split(t.vec,d.vec,x.mat,x.name="ANXA5”,cutoff=1)
```

`risk.classification` *The risk ranks of the samples predicted by a tree*

Description

The function returns the ranks (1=the lowest risk, 2=the 2nd lowest risk, ..., k=the highest risk) predicted for the samples.

Usage

```
risk.classification(tree, X.mat)
```

Arguments

`tree` :an object made from the "uni.tree" function

`X.mat` :n by p matrix of covariates from the samples, where n is the sample size and p is the number of covariates

Details

If the tree has k terminal nodes, then the response 1 respresents the lowest risk and k represents the highest risk.

Value

A vector of integers, 1, 2, ..., k, that represent the ranks predicted for the samples.

Examples

```
data(Lung,package="compound.Cox")
train_Lung=Lung[which(Lung[,"train"]==TRUE),] #select training data
t.vec=train_Lung[,1]
d.vec=train_Lung[,2]
x.mat=train_Lung[,-c(1,2,3)]
res=uni.tree(t.vec,d.vec,x.mat,P.value=0.01,d0=0.01,S.plot=FALSE,score=TRUE)
risk.classification(res,x.mat)
```

`uni.logrank` *Univariate binary splits by the logrank test*

Description

The output is the summary of significance tests for binary splits, where the cut-off values are optimized for each covariate.

Usage

```
uni.logrank(t.vec, d.vec, X.mat)
```

Arguments

`t.vec` :Vector of survival times (time to either death or censoring)

`d.vec` :Vector of censoring indicators (1=death, 0=censoring)

`X.mat` :n by p matrix of covariates, where n is the sample size and p is the number of covariates

Details

The output can be used to construct a logrank tree.

Value

A dataframe containing:

Pvalue: the P-value of the two-sample logrank test, where the cut-off value is optimized

cut_off_point: the optimal cutt-off values of the binary splits given a feature

left.sample.size: the sample size of a left child node

right.sample.size: the sample size of a right child node

Examples

```
data(Lung,package="compound.Cox")
train_Lung=Lung[which(Lung[,"train"]==TRUE),] #select training data
t.vec=train_Lung[,1]
d.vec=train_Lung[,2]
x.mat=train_Lung[,-c(1,2,3)]
uni.logrank(t.vec,d.vec,x.mat)
```

uni.tree *A survival tree based on stabilized score tests*

Description

This function returns a classification (decision) tree for a given survival dataset. The decision of making inner nodes (splitting criterion) is based on the univariate score tests. The decision of declaring terminal nodes (stopping criterion) is the P-value threshold given by an argument. This tree construction algorithm is proposed by Emura et al. (2021).

Usage

```
uni.tree(
  t.vec,
  d.vec,
  X.mat,
  P.value = 0.01,
  d0 = 0.01,
  S.plot = FALSE,
  score = TRUE
)
```

Arguments

`t.vec`	:Vector of survival times (time to either death or censoring)
`d.vec`	:Vector of censoring indicators (1=death, 0=censoring)
`X.mat`	:n by p matrix of covariates (features), where n is the sample size and p is the number of covariates
`P.value`	:the threshold of P-value for stop splitting (stopping criterion)
`d0`	:A positive constant to stabilize the variance of score statistics (Witten & Tibshirani 2010)
`S.plot`	:call for plot the KM estimator for each split
`score`	:TRUE = score test (Emura et al. 2019); FALSE = log-rank test

Details

In order to stabilize the univariate score tests, a small value "d0" is added to the variance of the score statistics (Witten and Tibshirani 2010). d0=0 corresponds to the logrank test. To perform a large number of the score tests, the "compound.Cox" packages (Emura et al.2019) is applied with d0 as a option.

Value

A nested list describing a classification tree, consisting of inner nodes and terminal node.

References

Emura T, Hsu WC, Chou WC (2021). A survival tree based on stabilized score tests for high-dimensional covariates, in review

Emura T, Matsui S, Chen HY (2019). compound.Cox: Univariate Feature Selection and Compound Covariate for Predicting Survival, Computer Methods and Programs in Biomedicine 168: 21-37.

Witten DM, Tibshirani R (2010) Survival analysis with high-dimensional covariates. Stat Method Med Res 19:29-51

Examples

```
data(Lung,package="compound.Cox")
train_Lung=Lung[which(Lung[,"train"]==TRUE),] #select training data
t.vec=train_Lung[,1]
d.vec=train_Lung[,2]
x.mat=train_Lung[,-c(1,2,3)]
uni.tree(t.vec,d.vec,x.mat,P.value=0.01,d0=0.01,S.plot=FALSE,score=TRUE)
```

X.pathway_discrete.balanced

Generate a matrix of gene expressions (discrete version of X.pathway() against to Emura (2012)) in the presence of gene pathways

---

Description

Generate a matrix of gene expressions in the presence of gene pathways, we first produce the matrix by X.pathway(Emura et al. 2012), then we change each value to 1 ~ 4 depend on the quantile.

Usage

```
X.pathway_discrete.balanced(n, p, q1, q2, rho1 = 0.5, rho2 = 0.5)
```

Arguments

n	:the number of individuals (sample size)
p	:the number of genes
q1	:the number of genes in the first pathway
q2	:the number of genes in the second pathway
rho1	:the correlation coefficient for the first pathway
rho2	:the correlation coefficient for the second pathway

Value

X n by p matrix of gene expressions

References

Emura T, Chen YH, Chen HY (2012). Survival Prediction Based on Compound Covariate under Cox Proportional Hazard Models. PLoS ONE 7(10): e47627. doi:10.1371/journal.pone.0047627

Examples

```
## generate 6 gene expressions from 10 individuals
X.pathway_discrete.balanced(n=10,p=6,q1=2,q2=2,rho1=0.5,rho2=0.5)
```

X.pathway_discrete.imbalanced

Generate a matrix of unbalance gene expressions (discrete version of X.pathway() against to Emura (2012)) in the presence of gene pathways

---

Description

Generate a matrix of gene expressions in the presence of gene pathways, we first produce the matrix by X.pathway(Emura et al. 2012), then we change each value to 1 ~ 3 depend on the quantile and randomly replace a element to 4 in the last p-(q1+q2) column for each row.

Usage

```
X.pathway_discrete.imbalanced(n, p, q1, q2, rho1 = 0.5, rho2 = 0.5)
```

Arguments

n	:the number of individuals (sample size)
p	:the number of genes
q1	:the number of genes in the first pathway
q2	:the number of genes in the second pathway
rho1	:the correlation coefficient for the first pathway
rho2	:the correlation coefficient for the second pathway

Value

X n by p matrix of gene expressions

References

Emura T, Chen YH, Chen HY (2012). Survival Prediction Based on Compound Covariate under Cox Proportional Hazard Models. PLoS ONE 7(10): e47627. doi:10.1371/journal.pone.0047627

Examples

```
## generate 6 gene expressions from 10 individuals
X.pathway_discrete.imbalanced(n=10,p=6,q1=2,q2=2,rho1=0.5,rho2=0.5)
```

Index