# Package ‘rpst’

June 6, 2017

**Type**  Package  
**Title**  Recursive Partitioning Survival Trees  
**Version**  1.0.0  
**Date**  2017-6-6  
**Author**  Yewei Li  
**Maintainer**  Yewei Li &lt;lywk12@163.com&gt;  
**Description**  An implementation of Recursive Partitioning Survival Trees via a node-splitting rule that builds decision tree models that reflected within-node and within-treatment responses. The algorithm aims to find the maximal difference in survival time among different treatments.  
**License**  GPL-3  
**Depends**  R (>= 2.0.0)  
**Imports**  plotrix, survival  
**NeedsCompilation**  no  
**Repository**  CRAN  
**Date/Publication**  2017-06-06 15:41:08 UTC

## R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>gen.data</td>
<td>2</td>
</tr>
<tr>
<td>plot.rpst</td>
<td>3</td>
</tr>
<tr>
<td>predict.rpst</td>
<td>4</td>
</tr>
<tr>
<td>print.rpst</td>
<td>5</td>
</tr>
<tr>
<td>rpst</td>
<td>5</td>
</tr>
<tr>
<td>rpst.km</td>
<td>7</td>
</tr>
</tbody>
</table>

Index 8
**Description**

Generate survival data for simulations.

**Usage**

```r
gen.data(n=100L, p=10L, c = 4, beta = NULL)
```

**Arguments**

- `n`: The number of observations.
- `p`: The number of predictors of interest.
- `c`: The censoring parameter to control the censoring rate. Default is 4.
- `beta`: The coefficient values in the underlying regression model.

**Value**

A list with the following components: `data`, `beta`.

- `data`: The simulated dataset.
- `beta`: The coefficients.

**Author(s)**

Yewei Li

**References**


**See Also**

- `rpst`, `predict.rpst`, `plot.rpst`.

**Examples**

```r
#require(survival)
data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)
```
plot.rpst

**plot.rpst**

*Produces a tree plot of a "rpst" object*

---

**Description**

Produces a tree plot of a "rpst" object.

**Usage**

```r
## S3 method for class 'rpst'
plot(x, xlength = 1, ylength = 3, xshift = 0.5, ysegment = 6,...)
```

**Arguments**

- `x`: a "rpst" project.
- `xlength`: xlength of the plot.
- `ylength`: ylength of the plot.
- `xshift`: xshift of the plot.
- `ysegment`: ysegment of the plot.
- `...`: additional summary arguments.

**Author(s)**

Yewei Li

**References**


**See Also**

`rpst`, `print.rpst`, `predict.rpst`.

**Examples**

```r
#require(survival)
data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)
plot.rpst(tree)
```
predict.rpst  
*Make predictions from a "rpst" object*

**Description**

Similar to other predict methods, which returns predictions from a fitted "rpst" object.

**Usage**

```r
## S3 method for class 'rpst'
predict(object, newdata,...)
```

**Arguments**

- `object`: Output from the `rpst` function.
- `newdata`: New data used for prediction.
- `...`: additional summary arguments.

**Value**

`predict.rpst` returns a dataframe that contain a new column "class", which contains the class labels of each sample.

**Author(s)**

Yewei Li

**References**


**See Also**

`print.rpst`, `predict.rpst`, `plot.rpst`.

**Examples**

```r
#require(survival)
##not run##
data <- gen.data(n = 100, p = 10, c = 2)
tree <- rpst(data$data, minsize = 2)
class = predict.rpst(tree, data$data)$class

data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
class = predict.rpst(tree, data$data)$class
```
**print.rpst**

*Print out a "rpst" object*

---

**Description**

Print node and node information details of "rpst".

**Usage**

```r
## S3 method for class 'rpst'
print(x, ...)
```

**Arguments**

- `x`: a "rpst" project.
- `...`: additional summary arguments.

**Author(s)**

Yewei Li

**See Also**

`rpst`, `predict.rpst`, `plot.rpst`.

**Examples**

```r
# require(survival)
data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)
```

---

**rpst**

*Recursive Partitioning Survival Trees*

---

**Description**

Build recursive partitioning survival trees via a node-splitting rule that builds decision tree models that reflected within-node and within-treatment responses.

**Usage**

```r
rpst(data, datapath, maxlay = 12, minsize = 5)
```
Arguments

- **data**: Input dataframe, it should be set in the particular way showed in detail.
- **datapath**: The file path of the input dataframe.
- **maxlay**: The maximum layer of the survival tree.
- **minsize**: The minimum sample size in the each leaf node of the survival tree.

Details

The input dataframe should be set in the following way: (i) The variable names are specified in the first line in the data file. (ii) The variable types are specified in the second line in the data file. The variable types can include treatment (1), survival time and censor status (2), ordered (3), binary (4), nominal (5), and unwanted (6) variables.

For potential predictors, they are categorized in three types, including ordered, binary, and nominal ones.

Specifically, ordered covariates include continuous and ordinal data types, binary covariates must have only two factors (they are typically coded as 0 and 1), and nominal covariates must have more than two factors. The number in the parenthesis represents each variable type.

Author(s)

Yewei Li

References


See Also

- `print.rpst`
- `predict.rpst`
- `plot.rpst`

Examples

```r
#require(survival)
data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)
```
rpst.km

## Description

Produces KM-curve plot of a "rpst" object.

## Usage

```r
rpst.km(object, newdata, ...)
```

## Arguments

- `object`: a "rpst" project.
- `newdata`: New data used for prediction and plot.
- `...`: additional summary arguments.

## Author(s)

Yewei Li

## References


## See Also

- `rpst`
- `print.rpst`
- `predict.rpst`

## Examples

```r
#require(survival)
data <- gen.data(n = 20, p = 5, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)

rpst.km(tree, data$data)

#not run#
data <- gen.data(n = 100, p = 10, c = 2)
tree <- rpst(data$data, minsize = 2)
print.rpst(tree)

#rpst.km(tree, data$data)
```
Index

gen.data, 2

plot.rpst, 2, 3, 4–6
predict.rpst, 2–4, 4, 5–7
print.rpst, 3, 4, 5, 6, 7

rpst, 2, 3, 5, 5, 7
rpst.km, 7