Package 'WPC'

July 30, 2016

Type Package

Title Weighted Predictiveness Curve

Version 1.0

Date 2016-07-29

Author
Hui Yang <huiy@amgen.com>, Rui Tang <rui_rui_tang@vrtx.com>, Jing Huang <jinghuang0@gmail.com>

Maintainer Hui Yang <huiy@amgen.com>

Description Implementing weighted predictiveness curve to visualize the marker-by-treatment relationship and measure the performance of biomarkers for guiding treatment decision.

License LGPL (>= 2)

Depends survival, msm

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2016-07-30 09:30:28

R topics documented:

WPC-package *Implement Weighted Predictiveness Curve to Visualize the Marker-by-Treatment Relationship and Measure the Performance of Biomarkers for Guiding Treatment Decision.*

Description

The package includes ten functions and one example for illustration purpose.

Based on the nature of the data, it generates predictiveness curve by utilizing either parametric or nonparametric approaches. When the unique effect of biomarker value change is indeed multiplicative with respect to hazard rate, parametric WPC (based on COX proportional hazard model) can most efficiently estimate survival rate for each biomarker value. The function `cox.wpc.est` will generate the parametric WPC using Cox model, and returns point estimates, confidence intervals and their biomarker values.

For most real world cases, little is known about the data structure, and that's when nonparametric WPC should be considered. The estimates are based on a series of overlapping windows (subpopulation). There are two ways to generate the series of overlapping windows: by fixing the number of patients within each window and by fixing the biomarker scale window width. The function `ns.windows` uses the first approach and report the detailed information of each window, while the function `ww.windows` utilizes the second approach. The function `npr.wpc.est` incorporate those two functions and their associated parameters to generate nonparametric WPC. Similar to `cox.wpc.est`, the function `npr.wpc.est` returns point estimates, confidence intervals and their each biomarker values.

The primary functions in the package are `SoloWPCCurve`, `DuoWPCCurve` and `TrioWPCCurve`. They generates the graphs of single, double and triple weighted predictiveness curves based on the point estimates and confidence intervals reported by `cox.wpc.est` and `npr.wpc.est`.

The packages can be used to compare biomarkers and identify the one with the highest impact. Equally important, by simultaneously depicting several treatment-specific WPC curves, it is easy to detect treatment heterogeneity as well as treatment-specific patterns, which in turn will help us with subgroup selection and biomarker cut-off Optimization.

Details

Index of help topics:

`DuoScattorPlot`	`Generate Scatter Plots for Time-to-Event and Biomarkers for Two Groups`
`DuoWPCCurve`	`Generate Two Weighted Predictiveness Curves in Graph`
`SoloScattorPlot`	`Generate Scatter Plots for Time-to-Event and Biomarkers for One Group`
`SoloWPCCurve`	`Generate Single Weighted Predictiveness Curve in Graph`
`TrioScattorPlot`	`Generate Scatter Plots for Time-to-Event and Biomarkers for Three Groups`

TrioWPCCurve	Generate Three Weighted Predictiveness Curves in Graph
WPC-package	Implement Weighted Predictiveness Curve to Visualize the Marker-by-Treatment Relationship and Measure the Performance of Biomarkers for Guiding Treatment Decision.
cox.wpc.est	Generate Weighted Predictiveness Curve Estimates Using Parametric Approach.
npr.wpc.est	Generate Weighted Predictiveness Curve Estimates Using Non-Parametric Approach.
ns.windows	Create a Series of Overlapping Windows by Fixing Number of Patients within each Window
surv.rate	Calculate Survival Rate at a Fix Time Point
wpcdata	A Data Example to Illustrate WPC Approach.
ww.windows	Create a Series of Overlapping Windows by Fixing Biomarker Scale Window Width

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_rui_tang@vrtx.com>, Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

Therneau T (2013). A Package for Survival Analysis in S. R package version 2.37-4, http://CRAN.R-project.org/package=survival.

Christopher H. Jackson (2011). Multi-State Models for Panel Data: The msm Package for R. *Journal of Statistical Software*, 38(**8**), 1-29. http://www.jstatsoft.org/v38/i08/.

See Also

Package Survival

`cox.wpc.est` *Generate Weighted Predictiveness Curve Estimates Using Parametric Approach.*

Description

This function generates weighted predictiveness curve estimates and/or confident bands using parametric approach.

Usage

```
cox.wpc.est(event, censor, marker, cutoff, quantile)
```

Arguments

event	This is the survival time. It is a positive numerical vector with no missing values.
censor	This specifies censor information. It is a vector, with 1 indicating an event and 0 indicating right censored. No missing values are allowed.
marker	This is the biomarker information (or other interesting variables). It is numerical with no missing values.
cutoff	This is to define the time cutoff.
quantile	This specifies the quantile of the confident band. Default is 0.95, 95% Confident band will be generated.

Details

The Cox proportional hazard model with a single biomarker will be used to derive and draw the predictiveness curve for parametric WPC. The relationship could be written in the form of the survival function as follows: $S(t) = [S_0(t)]^{expx\beta}$, where S(t) is survival function, $S_0(t)$ is baseline survivor function, and x is the biomarker of interest. The effect of the biomarker is expressed by the $exp(x\beta)$ term and quantified as a shift from the baseline survival $S_0(t)$. Because $S_0(t)$ is always between 0 and 1, a positive coefficient β will decrease the survival function with increasing biomarker values; a negative coefficient β will increase the survival function with decreasing biomarker values. For any given time t, the baseline survival function $S_0(t)$ could be estimated. Therefore, with a fixed coefficient estimate and fixed time, we could do such prediction for a range of x values by fitting x values into the formula earlier and then connect the predictions derived from the smallest x value to the largest x value. That will form the predictiveness curve for that particular time point.

Value

A list with components:

x	a vector of biomarker values.
s	A vector of survival rate estimates for each biomarker value.
lb	A vector of confident lower bands
ub	A vector of confident upper band

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

npr.wpc.est

Examples

```
## install packages "survival" and "msm"

library("survival")
library("msm")

cox.object = cox.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,quantile=0.95)

print(cox.object)
```

`DuoScattorPlot` *Generate Scatter Plots for Time-to-Event and Biomarkers for Two Groups*

Description

This function will generate the scatter plot of time-to-event and biomarker for two dataset. It helps to visualize the relationship between survival endpoints and biomarkers. It can also help to compare the two datasets

Usage

```
DuoScattorPlot(data1, data2, cutoff, xlab, ylab, main, ylim, xlim, col1, col2, col3, lwd,
pch1, pch2, legendloc, legendtxt, ncol)
```

Arguments

`data1`	Data object 1 with three variables included: *event*: the survival time, a positive numerical vector with no missing values; *censor*: the censor information, a vector with 1 indicating an event and 0 indicating right censored; *marker*: the biomarker information, or other interesting variables.
`data2`	Data object 2 with the same structure as data object 1.
`cutoff`	This is to define the interesting data cutoff time point to see the relationship between time-to-events and markers.
`xlab`	It is the title for x axis; default is "Marker".
`ylab`	It is the title for y axis; default is "Time to Event".
`main`	It is the title for the plot; default is "Scattor Plot".
`ylim`	It creates the continuous scale of y axis of the plot; default is "c(0,3600)".
`xlim`	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
`col1`	It defines the color of the dot in the dataset 1; default is "red".
`col2`	It defines the color of the dot in the dataset 2; default is "black".
`col3`	It defines the color of the cutoff line; default is "tomato".

lwd	It defines the width of the cutoff line; default is "2".
pch1	It defines the type of the dot for event; default is "20".
pch2	It defines the type of the dot for censor; default is "21".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; default is "c("death-group1","censor-group1","death-group2","censor-group2")".
ncol	It specifies the number of columns displayed in legend; default=1

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

SoloScattorPlot, TrioScattorPlot

Examples

```
## Create two data objects for the function:


tmppb = wpcdata[wpcdata$ATRT=="Placebo",]
tmptrt = wpcdata[wpcdata$ATRT=="Treatment",]
o.data1 =data.frame(event=tmppb$OSday, censor=tmppb$OScensor, marker=tmppb$Biomarker1)
o.data2 =data.frame(event=tmptrt$OSday, censor=tmptrt$OScensor, marker=tmptrt$Biomarker1)

        ## Draw the scattor plot for the three data objects:

DuoScattorPlot(o.data1,o.data2,180,xlab=c("Marker"),ylab=c("Survival Rate"),
main=c("Weighted Predictiveness Curve"),ylim=c(0,600),xlim=c(0,100),
col1="red",col2="black",lwd=2,pch1=20,pch2=21,legendloc="bottomright",ncol=1)
```

DuoWPCCurve *Generate Two Weighted Predictiveness Curves in Graph*

Description

This function will generate two weighted predictiveness curves using the estimates provided by "npr.wpc.est" or "cox.wpc.est" functions. It can be used to compare the relationships between survival rate and biomarker from two different curves.

We can utilize this function to compare the performance between non-parametric predictiveness curve and parametric(cox) predictiveness curve, or compare the performance from non-parametric

predictiveness curves using two different sets of parameters, or compare the predictiveness curves by using data from two different treatment groups and therefore compare treatment-by-biomarker relationships.

Usage

```
DuoWPCCurve(wpc1, wpc2, xlab, ylab, main, ylim, xlim, type, col1, col2, lwd,
legendloc, legendtxt, confi, ptsest)
```

Arguments

wpc1	It is the object1 generated by function cox.wpc.est or npr.wpc.est.
wpc2	It is the object2 generated by function cox.wpc.est or npr.wpc.est.
xlab	It is the title for x axis; default is "Marker".
ylab	It is the title for y axis; default is "Survival Rate".
main	It is the title for the plot; default is "Weighted Predictiveness Curve".
ylim	It creates the continuous scale of y axis of the plot; default is "c(0,1)".
xlim	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
type	It defines the type of the curves; default is "l".
col1	It defines the color of the curve 1 from object 1; default is "red".
col2	It defines the color of the curve 2 from object 2; default is "blue".
lwd	It defines the width of the curve; default is "2".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; default is "c("Method1")".
confi	It provides the option of drawing the confidence bands; default is "N", which means no confidence band is needed; "Y" will report the confidence band.
ptsest	It provides the option of drawing the point estimates; default is "N", which means no point estimates is needed; "Y" will report the point estimates.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

SoloWPCCurve, TrioWPCCurve

Examples

```
# Get the estiamte of predictiveness curve from npr.wpc.est functions

npr.object = npr.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,method="number.subjt",weights="normal",
nsub=10,sspeed=1,df=2,confi="NO")

# Get the estiamte of predictiveness curve from cox.wpc.est functions

cox.object = cox.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,quantile=0.95)

# Print Predictiveness Curve

DuoWPCCurve(npr.object,cox.object,xlab="Marker",ylab="Survival Rate",
main="Weighted Predictiveness Curve",ylim=c(0,1),xlim=c(0,100),type="l",
col1="red",col2="blue",lwd=2,legendloc="bottomright",
legendtxt=c("treatment","placebo"),confi="N", ptsest="N")
```

`npr.wpc.est` *Generate Weighted Predictiveness Curve Estimates Using Non-Parametric Approach.*

Description

This function generates weighted predictiveness curve estimates and/or confident bands using non-parametric approach.

Usage

```
npr.wpc.est(event, censor, marker, cutoff, method, weights, wdth, nsub, sspeed, df,
confi, nbtsp, quantile)
```

Arguments

`event`	This is the survival time. It is a positive numerical vector with no missing values.
`censor`	This specifies censor information. It is a vector, with 1 indicating an event and 0 indicating right censored. No missing values are allowed.
`marker`	This is the biomarker information (or other interesting variables). It is numerical with no missing values.
`cutoff`	This is to define the interesting data cutoff time point. The weighted predictiveness curve will be plotted based on this time point.
`method`	This is to specify the method used to define the series of overlapping windows. Two options are provided: method=window.width when the approach of fixing the biomarker scale window width is used. method=number.subjt when the approach of fixing the number of subjects within each window is used.

`weights` This is to specify the weight function, which will be applied to the Kaplan Meier approach for the survival rate estimates within each window. There are four options provided for this weight function: "uniform", "normal", "trunnormal", and "huber".

`wdth` "This is to specify window width, which is defined based on the biomarker scale. The smaller the window width is, the more the overlapping windows are specified. This parameter needs to be specified when we are using the fixed window width approach.

`nsub` This is to specify the fixed number of patients within each window. The smaller the number of patients within each window, the more the overlapping windows are specified. This parameter need to be specified when we are using the fixed number of subject within each window arrpoach.

`sspeed` This is to specify the window sliding step. The window is gradually moving from small values on the left to the large values on the right. This variable specifies the window sliding step being removed from the left and added on the right, in order to keep the same window width for each window.

`df` It defines the degree of polynomials used for loess function when the local regression method is implemented. Normally, we take the value of 1 or 2. Here df=2 as default.

`confi` This provides the option of reporting the confident band. If confi="NO", the confident band will not be generated. If confi="YES", the confident band will be generated. Since we are using the bootstrap resampling method, it can be time-consuming to generate the confident band. Default is "NO".

`nbtsp` This specifies the number of resampling for generating confident band. This number needs to be specified if the confi=YES. Default is 1000.

`quantile` This specifies the quantile of the confident band. Default is 0.95, 95% Confident band will be generated.

Details

Given the series of overlapping sliding windows, for a fixed survival time *cutoff*, the survival probability within each window is estimated using Kaplan-Meier method and assigned to the median biomarker value within that window. For a given biomarker value, the window works to borrow information from its neighborhood to enhance the estimation of survival rate. Three weight options are incorporated: normal, Huber, and uniform (i.e., no weight) to give the user the maximum flexibility.

Repeating the process and assigning the survival rate estimate for each biomarker value, we can obtain the pair of data, in term of biomarker value and survival rate estimates, for each window. From those series of paired data, we can draw the survival rate estimation curve of the biomarker value. To avoid over-fitting, we implement a local regression (loess) method to smooth across all window-specific median estimates to generate a relatively smooth predictiveness curve.

To have a measure of the precision of the predictiveness curve, we also provide the option of drawing the confident intervals in addition to the point estimates. Since it is very challenging to derive a close-form formula in this non-parametric setting, we use a non-parametric bootstrap technique to construct the confident bands.

Value

A list with components:

x	a vector of biomarker values for each overlapping window.
s	A vector of survival rate estimates for each overlapping window.
lb	A vector of lower band of survival rate estimates for each overlapping window.
ub	A vector of upper band of survival rate estimates for each overlapping window.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

cox.wpc.est

Examples

```
## install packages "survival" and "msm"

library("survival")
library("msm")



## Fixed 10 subjects within each window and window sliding step is 1,
## normal weight function is used:

npr.object1 = npr.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,method="number.subjt",weights="normal",
nsub=10,sspeed=1,df=2,confi="NO")
print(npr.object1)

## Fixed biomarker scale window width 10 and window sliding step is 1,
## huber weight function is used:

## Not run: npr.object2 = npr.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,method="window.width",weights="huber",
wdth=10,sspeed=1, df=2, confi="YES", nbtsp=100)
print(npr.object2)
## End(Not run)
```

`ns.windows` *Create a Series of Overlapping Windows by Fixing Number of Patients within each Window*

Description

This function creates a series of overlapping windows by fixing the number of patients within each window.

Usage

```
ns.windows(event, censor, marker, nsub, sspeed)
```

Arguments

`event`	This is the survival time. It is a positive numerical vector with no missing values.
`censor`	This specifies censor information. It is a vector, with 1 indicating an event and 0 indicating right censored. No missing values are allowed.
`marker`	This is the biomarker information (or other interesting variables). It is numerical with no missing values.
`nsub`	This is to specify the fixed number of patients within each window. The smaller the number of patients within each window, the more the overlapping windows are specified.
`sspeed`	This is to specify the window sliding step. Since the window is gradually moving from small values on the left to the large values on the right. This variable specifies the window sliding step being removed form the left and added on the right, in order to keep the same number of subjects in each window.

Details

It begins by ordering all the subjects based on their biomarker values from low to high. Let $x_1, x_2, \ldots, x_n$ be the ordered unique values of X observed in the data. Then a series of overlapping windows can be defined using two parameters: γ - the number of patients within each window and ν - the number of patients being rotated out for each moving step. The window is gradually moving from small values on the left to large values on the right, in order to keep the same number of patient in each window. The first window starts from the first subject to the $(\gamma + 1)$th subject. The second window will move forward by νth subjects and including from $(\gamma + \nu + 1)$th subject till $(\gamma + 2 \times \nu)$th subject. This process continues until all subjects have been included in at least one window. Subjects can be included in several windows.

Value

A list with components:

`xwin`	A series of marker values which will be assigned to the estimated survival rates within each window.

ntotal	The total number of overlapping windows defined.
wdata	A list of overlapping windows and each list representing a window with data frame of event, censor and marker
nsam	A vector, with each number representing the number of patients within each overlapping window.
winsize	A vector, with each number representing the half width for each overlapping window. Each window width is two times of it.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

ww.windows

Examples

```
## Window width is specified as 10 and window sliding step is 1:

object = ns.windows(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1, nsub=10, sspeed=1)
print(object)
```

SoloScattorPlot *Generate Scatter Plots for Time-to-Event and Biomarkers for One Group*

Description

This function will generate the scatter plot of time-to-event and biomarker for one dataset. It helps to visualize the relationship between survival endpoints and biomarkers.

Usage

```
SoloScattorPlot(data, cutoff, xlab, ylab, main, ylim, xlim, col1, col2, lwd, pch1, pch2,
legendloc, legendtxt, ncol)
```

Arguments

data	It is a data object with three variables included: *event*: the survival time, a positive numerical vector with no missing values; *censor*: the censor information, a vector with 1 indicating an event and 0 indicating right censored; *marker*: the biomarker information, or other interesting variables.
cutoff	This is to define the interesting data cutoff time point to see the relationship between time-to-events and markers.
xlab	It is the title for x axis; default is "Marker".
ylab	It is the title for y axis; default is "Time to Event".
main	It is the title for the plot; default is "Scattor Plot".
ylim	It creates the continuous scale of y axis of the plot; default is "c(0,3600)".
xlim	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
col1	It defines the color of the dot; default is "red".
col2	It defines the color of the cutoff line; default is "red".
lwd	It defines the width of the cutoff line; default is "2".
pch1	It defines the type of the dot for event; default is "20".
pch2	It defines the type of the dot for censor; default is "21".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; default is "c("death","censor")".
ncol	It specifies the number of columns displayed in legend; default=1

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

DuoScattorPlot, TrioScattorPlot

Examples

```
## Create the data object for the function

o.data = data.frame(event=wpcdata$OSday, censor=wpcdata$OScensor, marker=wpcdata$Biomarker1)

## Print out the figure:

SoloScattorPlot(o.data,180,xlab=c("Marker"),ylab=c("Survival Rate"),
main=c("Weighted Predictiveness Curve"),ylim=c(0,600),xlim=c(0,100),
col1="red",col2="red",lwd=2,pch1=20,pch2=21,legendloc="bottomright",ncol=1)
```

SoloWPCCurve *Generate Single Weighted Predictiveness Curve in Graph*

Description

This function will generate one single weighted predictiveness curve in graph using the estimates provided by "npr.wpc.est" function. It helps to visualize the relationship between survival rate and biomarker.

Usage

```
SoloWPCCurve(wpc, xlab, ylab, main, ylim , xlim, type, col, lwd, legendloc,
legendtxt, confi, ptsest)
```

Arguments

wpc	It is the object generated by function cox.wpc.est or npr.wpc.est.
xlab	It is the title for x axis; default is "Marker".
ylab	It is the title for y axis; default is "Survival Rate".
main	It is the title for the plot; default is "Weighted Predictiveness Curve".
ylim	It creates the continuous scale of y axis of the plot; default is "c(0,1)".
xlim	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
type	It defines the type of the curve; default is "l".
col	It defines the color of the curve; default is "red".
lwd	It defines the width of the curve; default is "2".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; defalut is "c("Method1")".
confi	It provides the option of drawing the confidence bands; default is "N", which means no confidence band is needed; "Y" will report the confidence band.
ptsest	It provides the option of drawing the point estimates; default is "N", which means no point estimates is needed; "Y" will report the point estimates.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

DuoWPCCurve, TrioWPCCurve

Examples

```
# Get the estimate of predictiveness curve from npr.wpc.est functions
# and print the corresponding predictiveness curve

npr.object = npr.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,method="number.subjt",weights="normal",
nsub=10,sspeed=1,df=2,confi="NO")

SoloWPCCurve(npr.object,xlab="Marker",ylab="Survival Rate",
main="Weighted Predictiveness Curve",ylim=c(0,1),xlim=c(0,100),
type="l",col="red",lwd=2,confi="N",ptsest="Y")

# Get the estimate of predictiveness curve from cox.wpc.est functions
# and print the corresponding predictiveness curve

cox.object = cox.wpc.est(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1,cutoff=180,quantile=0.95)

SoloWPCCurve(cox.object,xlab="Marker",ylab="Survival Rate",
main="Weighted Predictiveness Curve",ylim=c(0,1),xlim=c(0,100),
type="l",col="red",lwd=2,confi="N",ptsest="Y")
```

---

surv.rate *Calculate Survival Rate at a Fix Time Point*

Description

This function is implemented in the npr.wpc.est function.

Usage

```
surv.rate(data, cutoff, wts, xwin)
```

Arguments

data	Data with event - the survival time, a positive numerical vector with no missing values; censor- censor information, a vector with 1 indicating an event and 0 indicating right censored
cutoff	This is to define the interesting data cutoff time point to see the relationship between time-to-events and markers.
wts	This is to specify the weight function, which will be applied to the Kaplan Meier approach for the survival rate estimates within each window. There are four options provided for this weight function: "uniform", "normal", "trunnormal", and "huber".
xwin	A series of marker values which will be assigned to the estimated survival rates within each window.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

npr.wpc.est

`TrioScattorPlot` *Generate Scatter Plots for Time-to-Event and Biomarkers for Three Groups*

Description

This function will generate the scatter plot of time-to-event and biomarker for three dataset. It helps to visualize the relationship between survival endpoints and biomarkers. It can also help to compare the three datasets

Usage

```
TrioScattorPlot(data1, data2, data3, cutoff, xlab, ylab, main, ylim, xlim, col1,
col2, col3, col4, lwd, pch1, pch2, legendloc, legendtxt, ncol)
```

Arguments

`data1`	Data object 1 with three variables included: *event*: the survival time, a positive numerical vector with no missing values; *censor*: the censor information, a vector with 1 indicating an event and 0 indicating right censored; *marker*: the biomarker information, or other interesting variables.
`data2`	Data object 2 with the same structure as data object 1.
`data3`	Data object 3 with the same structure as data object 1.
`cutoff`	This is to define the interesting data cutoff time point to see the relationship between time-to-events and markers.
`xlab`	It is the title for x axis; default is "Marker".
`ylab`	It is the title for y axis; default is "Time to Event".
`main`	It is the title for the plot; default is "Scattor Plot".
`ylim`	It creates the continuous scale of y axis of the plot; default is "c(0,3600)".
`xlim`	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
`col1`	It defines the color of the dot in the dataset 1; default is "red".
`col2`	It defines the color of the dot in the dataset 2; default is "blue".

col3	It defines the color of the dot in the dataset 3; default is "black".
col4	It defines the color of the cutoff line; default is "tomato".
lwd	It defines the width of the cutoff line; default is "2".
pch1	It defines the type of the dot for event; default is "20".
pch2	It defines the type of the dot for censor; default is "21".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; default is "c("death-group1","censor-group1","death-group2","censor-group2",,"death-group3","censor-group3")".
ncol	It specifies the number of columns displayed in legend; default=1

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

SoloScattorPlot, DuoScattorPlot

Examples

```
## Create three data objects for the function:

tmppb = wpcdata[wpcdata$TRTA=="Placebo",]
tmptrt1 = wpcdata[wpcdata$TRTA=="LowDose",]
tmptrt2 = wpcdata[wpcdata$TRTA=="HighDose",]
o.data1 =data.frame(event=tmppb$OSday, censor=tmppb$OScensor, marker=tmppb$Biomarker1)
o.data2 =data.frame(event=tmptrt1$OSday, censor=tmptrt1$OScensor, marker=tmptrt1$Biomarker1)
o.data3 =data.frame(event=tmptrt2$OSday, censor=tmptrt2$OScensor, marker=tmptrt2$Biomarker1)

## Draw the scattor plot for the three data objects:

TrioScattorPlot(o.data1,o.data2,o.data3,180,xlab=c("Marker"),ylab=c("Surovival Rate"),
main=c("Weighted Predictiveness Curve"),ylim=c(0,600),xlim=c(0,100),col1="red",
col2="black",col3="blue",lwd=2,pch1=20,pch2=21,legendloc="bottomright",ncol=1)
```

TrioWPCCurve *Generate Three Weighted Predictiveness Curves in Graph*

Description

This function will generate three weighted predictiveness curves in graph using the estimates provided by "npr.wpc.est" or "cox.wpc.est" functions. It can be used to compare the relationships between survival rate and biomarker from three different curves.

similarly, We can utilize this function to compare the performances from non-parametric predictiveness curves using three different sets of parameters, or compare the predictiveness curves by using data from three different treatment groups and therefore compare treatment-by-biomarker relationships.

Usage

```
TrioWPCCurve(wpc1, wpc2, wpc3, xlab, ylab, main, ylim, xlim, type, col1, col2, col3,
lwd, legendloc, legendtxt, confi, ptsest)
```

Arguments

wpc1	It is the object1 generated by function cox.wpc.est or npr.wpc.est.
wpc2	It is the object2 generated by function cox.wpc.est or npr.wpc.est.
wpc3	It is the object3 generated by function cox.wpc.est or npr.wpc.est.
xlab	It is the title for x axis; default is "Marker".
ylab	It is the title for y axis; default is "Survival Rate".
main	It is the title for the plot; default is "Weighted Predictiveness Curve".
ylim	It creates the continuous scale of y axis of the plot; default is "c(0,1)".
xlim	It creates the continuous scale of y axis of the plot; default is "c(0,100)".
type	It defines the type of the curves; default is "l".
col1	It defines the color of the curve 1 from object 1; default is "red".
col2	It defines the color of the curve 2 from object 2; default is "blue".
col3	It defines the color of the curve 3 from object 2; default is "black".
lwd	It defines the width of the curve; default is "2".
legendloc	It specifies the location of the legend; default is "bottomright".
legendtxt	It provides the text of the legend; default is "c("Method1")".
confi	It provides the option of drawing the confidence bands; default is "N", which means no confidence band is needed; "Y" will report the confidence band.
ptsest	It provides the option of drawing the point estimates; default is "N", which means no point estimates is needed; "Y" will report the point estimates.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

SoloWPCCurve, DuoWPCCurve

Examples

```
tmppb = wpcdata[wpcdata$TRTA=="Placebo",]
tmptrt1 = wpcdata[wpcdata$TRTA=="LowDose",]
tmptrt2 = wpcdata[wpcdata$TRTA=="HighDose",]
o.data1 =data.frame(event=tmppb$OSday, censor=tmppb$OScensor, marker=tmppb$Biomarker1)
o.data2 =data.frame(event=tmptrt1$OSday, censor=tmptrt1$OScensor, marker=tmptrt1$Biomarker1)
o.data3 =data.frame(event=tmptrt2$OSday, censor=tmptrt2$OScensor, marker=tmptrt2$Biomarker1)

## Not run: npr.object1 = npr.wpc.est(event=o.data1$event, censor=o.data1$censor,
marker=o.data1$marker,cutoff=180,method="window.width",weights="huber",
wdth=10,sspeed=1, df=2, confi="YES", nbtsp=1000)
npr.object2 = npr.wpc.est(event=o.data2$event, censor=o.data2$censor,
marker=o.data2$marker,cutoff=180,method="window.width",weights="huber",
wdth=10,sspeed=1, df=2, confi="YES", nbtsp=1000)
npr.object3 = npr.wpc.est(event=o.data3$event, censor=o.data3$censor,
marker=o.data3$marker,cutoff=180,method="window.width",weights="huber",
wdth=10,sspeed=1, df=2, confi="YES", nbtsp=1000)

TrioWPCCurve(npr.object1,npr.object2,npr.object3,xlab="Marker",ylab="Survival Rate",
main="Weighted Predictiveness Curve",ylim=c(0,1),xlim=c(0,100),type="l",col1="red",
col2="blue",col3="black",lwd=2,legendloc="bottomright",legendtxt=c("Method1",
"Method2","Method3"),confi="Y")
## End(Not run)
```

---

wpcdata *A Data Example to Illustrate WPC Approach.*

Description

This survival data example is to illustrate WPC approach. 90 patients are randomized into three different arms, 1 placebo and 2 treatments (high dose and low dose). For each patient, four biomarkers are measured at baseline. Overall survival and progression free survival information are collected.

Usage

```
data("wpcdata")
```

Format

A data frame with 90 observations on the following 13 variables.

`SUBJID` a numeric vector, indicating subject id information.

`TRTA` a factor, indicating three different arms, with levels `HighDose` `LowDose` `Placebo`

`ATRT` a factor, indicating whether patients receive placebo or treatment, with levels `Placebo` `Treatment`

`Biomarker1` a numeric vector, first biomarker with the value between 0 and 100

`Biomarker2` a numeric vector, second biomarker with the value between 0 and 200

`Biomarker3` a numeric vector, third biomarker with the value between 0 and 100

`Biomarker4` a numeric vector, fourth biomarker with the value between 0 and 200

`OSday` a numeric vector, overall survival in days

`OSmonth` a numeric vector, overall survival in months

`OScensor` a numeric vector, censor information for overall survival, 0 = alive and 1 = dead

`PFSday` a numeric vector, progression free survival in days

`PFSmonth` a numeric vector, progression free survival in months

`PFScensor` a numeric vector, censor information for progression free survival, 0 = censor and 1 = event

See Also

`SoloScattorPlot`, `DuoScattorPlot`, `TrioScattorPlot`

`ww.windows` *Create a Series of Overlapping Windows by Fixing Biomarker Scale Window Width*

Description

This function creates a series of overlapping windows by fixing the biomarker scale window width

Usage

```
ww.windows(event, censor, marker, wdth, sspeed)
```

Arguments

`event`	This is the survival time. It is a positive numerical vector with no missing values.
`censor`	This specifies censor information. It is a vector, with 1 indicating an event and 0 indicating right censored. No missing values are allowed.
`marker`	This is the biomarker information (or other interesting variables). It is numerical with no missing values.

`wdth` This is to specify window width of each overlapping window. The window width is defined based on the biomarker scale. The smaller the window width is, the more the overlapping windows are specified.

`sspeed` This is to specify the window sliding step. Since the window is gradually moving from small values on the left to the large values on the right. This variable specifies the window sliding step being removed from the left and added on the right, in order to keep the same window width for each window.

Details

It begins by ordering all the subjects based on their biomarker values from low to high. Let $x_1, x_2, \ldots, x_n$ be the ordered unique values of X observed in the data. Then a series of overlapping windows can be defined using two parameters: γ - the biomarker-scale window width and ν - the window sliding step. Because the window is gradually moving from small values on the left to large values on the right, in order to keep the same window width for each window. The first window starts from the first subject with the smallest biomarker value x_1, including subjects whose biomarker values are in the biomarker-scale window of $[x_1, x_1 + \gamma]$. The second window will move forward by ν biomarker-scale width, and include subjects whose biomarker values dropped in the second window $[x_1 + \nu, x_1 + \nu + \gamma]$. This process continues until all subjects have been included in at least one window. Subjects can be included in several windows.

Value

A list with components:

`xwin` A series of marker values which will be assigned to the estimated survival rates within each window.

`ntotal` The total number of overlapping windows defined.

`wdata` A list of overlapping windows and each list representing a window with data frame of event, censor and marker

`nsam` A vector, with each number representing the number of patients within each overlapping window.

`winsize` A vector, with each number representing the half width for each overlapping window. Each window width is two times of it.

Note

R packages **survival** and **msm** need to be installed before running the function.

Author(s)

Hui Yang <huiy@amgen.com>, Rui Tang <rui_tang@vrtx.com> and Jing Huang <jinghuang0@gmail.com>

References

Yang H., Tang R., Hale M. and Huang J. (2016) A visualization method measuring the performance of biomarkers for guiding treatment decisions *Pharmaceutical Statistics*, 15(**2**), 1539-1612

See Also

ns.windows

Examples

```
## Window width is specified as 10 and window sliding step is 1:


object = ww.windows(event=wpcdata$OSday, censor=wpcdata$OScensor,
marker=wpcdata$Biomarker1, wdth=10, sspeed=1)

print(object)
```

Index