Package 'ModelGood'

February 19, 2015

Title Validation of risk prediction models

Version 1.0.9

Author Thomas A. Gerds

Description Bootstrap cross-validation for ROC, AUC and Brier score to assess and compare predictions of binary status responses.

Imports prodlim, parallel

Suggests party, rpart, glmnet, rms, randomForest, randomForestSRC

Maintainer Thomas A. Gerds <tag@biostat.ku.dk>

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-11-10 10:01:39

R topics documented:

`AutoSelectLRM` *Automated backward elemination for logistic regression*

Description

Wrapper for automated backward elemination for logistic regression

Usage

```
AutoSelectLRM(formula, data, ...)
```

Arguments

`formula`	passed to lrm
`data`	passed to lrm
`...`	passed to fastbw

Details

First run backward elemination via fastbw from the rms package, then fit the logistic regression model including the selected variables

Value

object of class AutoSelectLRM

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

fastbw lrm

Examples

```
library(rms)
set.seed(7)
x <- abs(rnorm(20))
d <- data.frame(y=rbinom(20,1,x/max(x)),x=x,z=rnorm(20))
fbw <- AutoSelectLRM(y~x+z,d)
predictStatusProb(fbw,newdata=d[1:3,])
```

calPlot2 *Calibration plots for binary data*

Description

Calibration plots for risk prediction models in for a binary endpoint

Usage

```
calPlot2(object, formula, data, splitMethod = "none", B = 1, M, showY,
  method = "nne", round = TRUE, bandwidth = NULL, q = 10,
  density = 55, add = FALSE, diag = !add, legend = !add, axes = !add,
  xlim, ylim, xlab = "Predicted event probability",
  ylab = "Observed proportion", col, lwd, lty, pch, cause = 1,
  percent = TRUE, giveToModel = NULL, na.action = na.fail, cores = 1,
  verbose = FALSE, ...)
```

Arguments

object	A named list of prediction models, where allowed entries are (1) R-objects for which a predictStatusProb method exists (see details), (2) a call that evaluates to such an R-object (see examples), (3) a matrix with predicted probabilities having as many rows as data in one column. For cross-validation all objects in this list must include their call.
formula	A survival or event history formula. The left hand side is used to compute the expected event status. If formula is missing, try to extract a formula from the first element in object.
data	A data frame in which to validate the prediction models and to fit the censoring model. If data is missing, try to extract a data set from the first element in object.
splitMethod	Defines the internal validation design: none/noPlan: Assess the models in the give data, usually either in the same data where they are fitted, or in independent test data. BootCv: Bootstrap cross validation. The prediction models are trained on B bootstrap samples, that are either drawn with replacement of the same size as the original data or without replacement from data of the size M. The models are assessed in the observations that are NOT in the bootstrap sample.
B	The number of cross-validation steps.
M	The size of the subsamples for cross-validation.
showY	If TRUE the observed data are shown as dots on the plot.
method	The method for estimating the calibration curve(s): "nne": The expected event status is obtained in the nearest neighborhood around the predicted event probabilities. "quantile": The expected event status is obtained in groups defined by quantiles of the predicted event probabilities.

round	If TRUE predicted probabilities are rounded to two digits before smoothing. This may have a considerable effect on computing efficiency in large data sets.
bandwidth	The bandwidth for method="nne"
q	The number of quantiles for method="quantile".
density	Gray scale for observations.
add	If TRUE the line(s) are added to an existing plot.
diag	If FALSE no diagonal line is drawn.
legend	If FALSE no legend is drawn.
axes	If FALSE no axes are drawn.
xlim	Limits of x-axis.
ylim	Limits of y-axis.
xlab	Label for y-axis.
ylab	Label for x-axis.
col	Vector with colors, one for each element of object. Passed to lines.
lwd	Vector with line widths, one for each element of object. Passed to lines.
lty	lwd Vector with line style, one for each element of object. Passed to lines.
pch	Passed to points.
cause	For competing risks models, the cause of failure or event of interest
percent	If TRUE axes labels are multiplied by 100 and thus interpretable on a percent scale.
giveToModel	List of with exactly one entry for each entry in object. Each entry names parts of the value of the fitted models that should be extracted and added to the value.
na.action	Passed to model.frame
cores	Number of cores for parallel computing. Passed as the value of the argument mc.cores when calling mclapply.
verbose	if TRUE report details of the progress, e.g. count the steps in cross-validation.
...	Used to control the subroutines: plot, axis, lines, legend. See SmartControl.

Details

For method "nne" the optimal bandwidth with respect to is obtained with the function dpik from the package KernSmooth for a box kernel function.

Value

list with elements: time, Frame and bandwidth (NULL for method quantile).

Author(s)

Thomas Alexander Gerds

References

TA Gerds, PA Andersen, and Kattan MW. Calibration plots for risk prediction models in the presence of competing risks. Statistics in Medicine, page to appear, 2014.

Examples

```
set.seed(40)
N=40
Y=rbinom(N,1,.5)
X1=rnorm(N)
X1[Y==1]=rnorm(sum(Y==1),mean=rbinom(sum(Y==1),1,.5))
X2=rnorm(N)
X2[Y==0]=rnorm(sum(Y==0),mean=rbinom(sum(Y==0),3,.5))
dat <- data.frame(Y=Y,X1=X1,X2=X2)
lm1 <- glm(Y~X1,data=dat,family="binomial")
lm2 <- glm(Y~X2,data=dat,family="binomial")
calPlot2(list(lm1,lm2),data=dat)
```

---

click.Roc *Click on ROC curve*

Description

Show marker value sensitivity and specificity at mouse point

Usage

```
click.Roc(object, pch = 19, label = TRUE, adj, col = "orange", cex = 3,
  ...)
```

Arguments

object	An object obtained with function Roc
pch	the symbol of the tag
label	If TRUE label the tag.
adj	passed to text to adjust of the legend relative to clickpoint.
col	the color of the tag
cex	the size of the tag
...	passed to identify

Details

A tag is set on the ROC curve at the mouse click and corresponding marker value, sensitivity and specificity shown below the click-point.

Value

the values at the tag

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

identify Roc

Examples

```
## Not run:
x <- abs(rnorm(20))
d <- data.frame(y=rbinom(1:20,1,p=x/max(x)))
r <- Roc(y~x,data=d)
plot(r)
click.Roc(r)

## End(Not run)
```

`Ctree` *S3 wrapper function for party's ctree method*

Description

S3 wrapper function for party's ctree method

Usage

```
Ctree(...)
```

Arguments

`...`	passed to `ctree`

Details

The ModelGood crossvalidation functionality works only for S3 classes.

Value

object of class Ctree which contains a ctree object

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

ctree

Examples

```
library(party)
set.seed(7)
x <- abs(rnorm(20))
d <- data.frame(y=rbinom(20,1,x/max(x)),x=x,z=rnorm(20))
ct <- Ctree(y~x+z,d)
plot(ct$ctree)
predictStatusProb(ct,newdata=d[1:3,])
```

ElasticNet *Wrapper function for glmnet*

Description

Wrapper function for glmnet

Usage

```
ElasticNet(formula, data, nfolds = 10, ...)
```

Arguments

formula	Formula where the right hand side specifies the response and the left hand side the predictor matrix
data	A data frame in which formula is evaluated
nfolds	nfolds: number of cross-validation folds in cv.glmnet (default in function is 10)
...	passed on to glmnet

Details

This function first calls `cv.glmnet` and then evaluates glmnet at the hyper parameter which optimizes the cross-validation criterion.

Value

Object with class ElasticNet

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

predictStatusProb

Examples

```
# Generate some data with binary response Y
 # depending on X1 and X2 and X1*X2
set.seed(40)
N <- 40
X1 <- rnorm(N)
X2 <- rbinom(N,1,.4)
X3 <- rnorm(N)
expit <- function(x) exp(x)/(1+exp(x))
lp <- expit(1 + X1 + X2 + X3)
Y <- factor(rbinom(N,1,lp))
dat <- data.frame(Y=Y,X1=X1,X2=X2,X3=X3)

efit <- ElasticNet(Y~X1+X2+X3,data=dat,family="binomial",alpha=0.1)
Brier(efit,verbose=FALSE)
```

`plot.Roc` *ROC curves for risk prediction models*

Description

ROC curves for risk prediction models

Usage

```
## S3 method for class 'Roc'
plot(x, ylab = "Sensitivity", xlab = "1-Specificity", models,
  type = "l", shadow = FALSE, simu = FALSE, control, grid = FALSE,
  diag = TRUE, box = FALSE, lwd = 2, lty, col, add = FALSE,
  axes = TRUE, legend, auc, percent = TRUE, ...)
```

Arguments

`x`	object obtained with `Roc`
`ylab`	Label y-axis
`xlab`	Label x-axis
`models`	Selection of models to plot. Should be a subset of `names(x$models)`. Makes sense when `x` contains multiple ROC curves.
`type`	The line type
`shadow`	Experimental. Show results of cross-validation.
`simu`	Experimental. Show noinformation results.
`control`	Control which estimates of the ROC curves to draw.
`grid`	If `TRUE` add a grid in the background of the graph.
`diag`	If `TRUE` add a diagonal line.
`box`	If `TRUE` add a box around the graph.

lwd	Vector of line widths for the ROC curves.
lty	Vector of line types for the ROC curves.
col	Vector of colours for the ROC curves.
add	If TRUE add ROC curves to existing plot.
axes	If TRUE draw axes.
legend	If TRUE draw a legend.
auc	If TRUE add the area under the curve to the legend.
percent	If TRUE show percent axes.
...	Use for smart control of some plot elements.

Details

Multiple ROC curves are shown in one graph.

Value

ROC curves

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

Roc

Examples

```
# generate som data
set.seed(40)
N=40
Y=rbinom(N,1,.5)
X1=rnorm(N)
X1[Y==1]=rnorm(sum(Y==1),mean=rbinom(sum(Y==1),1,.5))
X2=rnorm(N)
X2[Y==0]=rnorm(sum(Y==0),mean=rbinom(sum(Y==0),1,.5))
dat <- data.frame(Y=Y,X1=X1,X2=X2)

# fit two logistic regression models
lm1 <- glm(Y~X1,data=dat,family="binomial")
lm2 <- glm(Y~X2+X1,data=dat,family="binomial")
plot(Roc(list(lm1,lm2),data=dat))

# add the area under the curves

plot(Roc(list(lm1,lm2),data=dat),auc=TRUE)

# alternatively, one can directly work with formula objects:
plot(Roc(list(LR.X1=Y~X1,LR.X1.X2=Y~X2+X1),data=dat),auc=TRUE)
```

```
# beyond the logistic regression model.
# the following example is optimized for speed
# illustrating the syntax,
# and not for optimized for performance of the
# randomForest or elastic net
library(randomForest)
library(glmnet)
dat$Y=factor(dat$Y)
rf <- randomForest(Y~X1+X2,data=dat,ntree=10)
en <- ElasticNet(Y~X1+X2,data=dat,nfolds=10,alpha=0.1)
set.seed(6)
rocCV=Roc(list(RandomForest=rf,ElasticNet=en,LogisticRegression=lm2),
  data=dat,
  verbose=FALSE,
  splitMethod="bootcv",
  B=4,
  cbRatio=1)
plot(rocCV,yaxis.las=2,legend.title="4 bootstrap-crossvalidation steps")
```

`predictStatusProb` *Probability Predictions*

Description

Function to extract probabilistic event status predictions from various diagnostic and prognostic models with binary status response. The function has a speficic method depending on the 'class' of the object.

Usage

```
## S3 method for class 'glm'
predictStatusProb(object,newdata,...)
```

Arguments

`object`	A model from which predicted probabilities can be extracted for the indiviuals in newdata.
`newdata`	A data frame containing data for which the `object` can provide predict probabilities. In medical applications `newdata` will typically consist of the data of patients whose data were not used for building the model.
`...`	Additional arguments that are passed on to the current method.

Details

The function delivers predicted probabilities tailored for the model performance measures of the package. These probabilities are extracted from a fitted model of class `CLASS` with the function `predictStatusProb.CLASS`. See `help(Roc)` for details.

Value

A vector with the predicted status probability for each row in `NROW(newdata)`.

Note

It is rather easy to write a new predictStatusProb method, see `help(Roc)`. However, if you do not succeed, please send me an email.

The performance, in particular when doing cross-validation where the model is evaluated many times, can be improved by supressing in the call to the model all the computations that are not needed for probability prediction, for example standard error calculations.

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

predict,Roc

Examples

```
library(rms)
set.seed(7)
x <- abs(rnorm(20))
d <- data.frame(y=rbinom(20,1,x/max(x)),x=x,z=rnorm(20))
nd <- data.frame(y=rbinom(8,1,x/max(x)),x=abs(rnorm(8)),z=rnorm(8))
fit <- lrm(y~x+z,d)
predictStatusProb(fit,newdata=nd)
```

---

reclass *Risk reclassification table*

Description

Tabulate grouped risks predicted by two different methods, models, algorithms

Usage

```
reclass(list, newdata, cuts = seq(0, 100, 25), digits = 1)
```

Arguments

list	A list with two elements. Each element should either be a vector with probabilities, or an object for which predictStatusProb can extract predicted risk based on newdata.
newdata	Passed on to predictStatusProb
cuts	Risk quantiles to group risk
digits	Number of digits to show for the predicted risks

Details

All risks are multiplied by 100 before

Value

reclassification table

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

predictStatusProb

Examples

```
set.seed(40)
N <- 40
X1 <- rnorm(N)
X2 <- rbinom(N,1,.4)
X3 <- rnorm(N)
expit <- function(x) exp(x)/(1+exp(x))
lp <- expit(X1 + X2 + X3)
Y <- factor(rbinom(N,1,lp))
dat <- data.frame(Y=Y,X1=X1,X2=X2,X3=X3)
lm1 <- glm(Y~X1,data=dat,family="binomial")
lm2 <- glm(Y~X1+X2,data=dat,family="binomial")

rc <- reclass(list("lrm.X1"=lm1,"lrm.X1.X2"=lm2),newdata=dat)
print(rc)
plot(rc)

rc2 <- reclass(list("lrm.X1"=lm1,"lrm.X1.X2"=lm2),newdata=dat,cuts=c(0,5,10,50,100))
print(rc2)
plot(rc2)
```

Roc *Comparing prediction models with Receiver operating characteristics and Brier scores*

Description

Evaluation of the performance of risk prediction models with binary status response variable (case/control or similar). Roc curves are either based on a single continuous marker, or on the probability prediction of an event. Probability predictions are extracted from a given (statistical) model, such as logistic regression, or algorithm, such as random forest. The area under the curve and the Brier score is used to summarize and compare the performance.

Usage

```
## S3 method for class 'list'
 Roc(object, formula, data, splitMethod='noSplitMethod',
noinf.method=c('simulate'), simulate='reeval', B, M, breaks, cbRatio=1,
RocAverageMethod='vertical',
RocAverageGrid=switch(RocAverageMethod, 'vertical'=seq(0,1,.01),
'horizontal'=seq(1,0,-.01)), model.args=NULL, model.parms=NULL,
keepModels=FALSE, keepSampleIndex=FALSE, keepCrossValRes=FALSE,
keepNoInfSimu, slaveseed, cores=1, na.accept=0, verbose=FALSE, ...)
```

Arguments

`object`	A named list of R objects that represent predictive markers, prediction models, or prediction algorithms. The function predictStatusProb is called on the R objects to extract the predicted risk (see details). For cross-validation (e.g. when `splitMethod` is 'bootcv') all the R objects in this list must include a `call` which can be evaluated in a learning subset of the data.
`formula`	A formula whose left hand side is used to identify the binary outcome variable in `data`. If missing, use the formula of the (first) model in object.
`data`	A data set in which to validate the prediction models. If missing, the function tries to extract the data from the call of the (first) model in object. The data set needs to have the same structure, variable names, factor levels, etc., as the data in which the models were trained. If the subjects in data were not used to train the models given in `object`, this leads to an external validation situation. However, note that if one of the elements in `object` is a formula then it is evaluated in this data set.
`splitMethod`	Method for estimating the generalization error. `none`:Assess the models in the data given by `data`. If this data set coincides with the train data where the models were fitted this yields an apparent (or re-substitution) estimate of performance. Otherwise, this leads to an external validation situation. `bootCV`: Internal bootstrap cross validation. The prediction models are trained on `B` bootstrap samples of the `data`. Bootstrap samples are either drawn with replacement from `data` (same size), or without replacement of the size `M` where `M` is a number smaller than `NROW(data)`. The model performance parameters (Roc, Brier, AUC) are estimated with the observations that are NOT in the current bootstrap sample. `boot632`: Linear combination of the apparent performance and the BootCV performance using the constant weight .632 (see Efron & Tibshirani, 1997). `boot632plus`: Linear combination of apparent performance and Bootcv using weights dependent on how the models perform in permuted data (see Efron & Tibshirani, 1997). `noinf`: Assess the models trained in permutations of `data`.
`noinf.method`	Experimental: For .632+ method the way to obtain no-information performance. This can either be 'simulate' or 'none'.

simulate	Experimental: For .632+ method. If 'reeval' then the models are re-build in the current permuted data for computing the no-information Roc curve.
B	Number of repetitions for internal crossvalidation. The meaning depends on the argument splitMethod: When splitMethod in c('Bootcv','Boot632','Boot632plus') it is the number of bootstrap samples, default is 100. Otherwise it is ignored.
M	The size of the bootstrap samples for cross-validation without replacement.
breaks	Break points for computing the Roc curve. Defaults to seq(0,1,.01) when crossvalidation is applied, i.e., when splitMethod in c('Bootcv','Boot632','Boot632plus'). Otherwise use all unique values of the predictive marker.
cbRatio	Experimental. Cost/benefit ratio. Default value is 1, meaning that misclassified cases are as bad as misclassified controls.
RocAverageMethod	Method for averaging ROC curves across data splits. If 'horizontal' average crossvalidated specificities for fixed sensitivity values, specified in RocAverageGrid, otherwise, if 'vertical', average crossvalidated specificities for fixed sensitivity values. See Fawcett, T. (2006) for details.
RocAverageGrid	Grid points for the averaging of Roc curves. A sequence of values at which to compute averages across the ROC curves obtained for different data splits during crossvalidation.
model.args	List of extra arguments that can be passed to the predictStatusProb methods. The list must have an entry for each entry in object.
model.parms	List with exactly one entry for each entry in object. Each entry names parts of the value of the fitted models that should be extracted and added to the output (see value).
keepModels	If FALSE keep only the names of the elements of object. If 'Call' then keep the call of the elements of object. Else, add the object as it is to the output.
keepSampleIndex	Logical. If FALSE remove the cross-validation index (which tells who was in the learn and who in the validation set) from the output list which otherwise is included in the method part of the output list.
keepCrossValRes	Logical. If TRUE add all B crossvalidation results to the output (see value). Defaults to TRUE.
keepNoInfSimu	Logical. If TRUE add the B results in permuted data (for no-information performance) to the output (see value). Defaults to FALSE.
slaveseed	Vector of seeds, as long as B, to be given to the slaves in parallel computing to control the models build in crossvalidation loop.
cores	Number of cores for parallel computing. Passed as the value of the argument mc.cores when calling mclapply.
na.accept	For 'Bootcv' estimate of performance. The maximal number of bootstrap samples in which the training the models may fail This should usually be a small number relative to B.
verbose	if TRUE the procedure is reporting details of the progress, e.g. it prints the current step in cross-validation procedures.
...	Used to pass arguments to submodules.

Details

All functions work on a list of models to ease comparison.

Bootstrap-crossvalidation techniques are implemented to estimate the generalization performance of the model(s), i.e., the performance which can be expected in new subjects.

By default, when crossvalidation is involved, the ROC curve is approximated on a grid of either sensitivities or specificities and not computed at all unique changepoints of the crossvalidated ROC curves, see Fawcett, T. (2006). The (density of the) grid can be controlled with the argument: RocAverageGrid

Missing data in the response or in the marker/predicted risk cause a failure.

For each R object which potentially can predict a probability for an event, there should be a corresponding `predictStatusProb` method:

For example, to assess a prediction model which evaluates to a `myclass` object one defines a function called `predictStatusProb.myclass` with arguments `object,newdata,...`. For example, the function predictStatusProb.lrm looks like this:

predictStatusProb.lrm <- function(object,newdata,...) p <- as.numeric(predict(object,newdata=newdata,type='fitted')) class(p) <- 'predictStatusProb' p

Currently implemented are `predictStatusProb` methods for the following R-functions:

`numeric` (marker values are passed on)

`formula` (single predictor: extracted from newdata and passed on, multiple predictors: projected to score by logistic regression)

`glm` (from `library(stats)`

`lrm` (from `library(Design)`

`rpart` (from `library(rpart)`)

`BinaryTree` (from `library(party)`)

`ElasticNet` (a wrapper for glmnet from `library(glmnet)`)

`randomForest` from `library(randomForest)`

`rfsrc` from `library(randomForestSRC)`

Value

Object of class `Roc` or class `Brier`.

Depending on the `splitMethod` the object includes the following components:

`Roc, Brier, Auc`
: A list of Roc curve(s), Brier scores (BS), and areas under the curves (Auc), one for each element of argument `object`, estimated according to `splitMethod`.

`weight`
: The weight used to linear combine the `AppRoc` and the `BootcvRoc` Only available if `splitMethod` is one of 'Boot632', or 'Boot632plus'.

`overfit`
: Estimated `overfit` of the model(s). Only if `splitMethod` is one of 'Boot632', or 'Boot632plus'.

`call`
: The call that produced the object

`models`
: See keepModels

`method`
: Summary of the splitMethod used.

Author(s)

Thomas Gerds <tag@biostat.ku.dk>

References

Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27, 861-874.

Gerds, Cai & Schumacher (2008). The Performance of Risk Prediction Models. Biometrical Journal, Vol 50, 4, 457-479.

Efron, Tibshirani (1997) Journal of the American Statistical Association 92, 548–560 Improvement On Cross-Validation: The .632+ Bootstrap Method.

Wehberg, S and Schumacher, M (2004) A comparison of nonparametric error rate estimation methods in classification problems. Biometrical Journal, Vol 46, 35–47

Examples

```
## Generate some data with binary response Y
## depending on X1 and X2 and X1*X2
set.seed(40)
N <- 40
X1 <- rnorm(N)
X2 <- abs(rnorm(N,4))
X3 <- rbinom(N,1,.4)
expit <- function(x) exp(x)/(1+exp(x))
lp <- expit(-2 + X1 + X2 + X3 - X3*X2)
Y <- factor(rbinom(N,1,lp))
dat <- data.frame(Y=Y,X1=X1,X2=X2)

# single markers, one by one
r1 <- Roc(Y~X1,data=dat)
plot(r1,col=1)
r2 <- Roc(Y~X2,data=dat)
lines(r2,col=2)

# or, directly multiple in one
r12 <- Roc(list(Y~X1,Y~X2),data=dat)
plot(r12)

## compare logistic regression
lm1 <- glm(Y~X1,data=dat,family="binomial")
lm2 <- glm(Y~X1+X2,data=dat,family="binomial")
r1=Roc(list(LR.X1=lm1,LR.X1.X2=lm2))
summary(r1)
Brier(list(lm1,lm2))

# machine learning
library(randomForest)
dat$Y=factor(dat$Y)
rf <- randomForest(Y~X2,data=dat)
rocCV=Roc(list(RandomForest=rf,LogisticRegression=lm2),
    data=dat,
```

```
    splitMethod="bootcv",
    B=3,
    cbRatio=1)
plot(rocCV)

# compute .632+ estimate of Brier score
bs <- Brier(list(LR.X1=lm1,LR.X2=lm2),
    data=dat,
    splitMethod="boot632+",
    B=3)
bs
#'
```

`Sensitivity` *Compute sensitivity, specificity and predictive values*

Description

Compute sensitivity, specificity and predictive values

Usage

```
Sensitivity(x,event,cutoff,comparison=">=",...)
       Specificity(x,event,cutoff,comparison=">=",...)
       NPV(x,event,cutoff,comparison=">=",...)
       PPV(x,event,cutoff,comparison=">=",...)
```

Arguments

`x`	Either a binary 0,1 variable, or a numeric marker which is cut into binary.
`event`	Binary response variable. Either a 0,1 variable where 1 means 'event', or a factor where the second level means 'event'.
`cutoff`	When x is a numeric marker, it is compared to this cutoff to obtain a binary test.
`comparison`	How x is to be compared to the cutoff value
`...`	passed on to `binom.test`

Details

Confidence intervals are obtained with `binom.test`

Value

list with Sensitivity, Specificity, NPV, PPV and confidence interval

Author(s)

Thomas A. Gerds <tag@biostat.ku.dk>

See Also

binom.test

Examples

```
set.seed(17)
x <- rnorm(10)
y <- rbinom(10,1,0.4)
Sensitivity(x,y,0.3)
Specificity(x,y,0.3)
PPV(x,y,0.3)
NPV(x,y,0.3)

Diagnose(x,y,0.3)
```

Index