Package 'ClinicalUtilityRecal'

January 20, 2025

Type Package

Title Recalibration Methods for Improved Clinical Utility of Risk Scores

Version 0.1.0

Author Anu Mishra

Maintainer Anu Mishra <anmishra@uw.edu>

Description Recalibrate risk scores (predicting binary outcomes) to improve clinical utility of risk score using weighted logistic or constrained logistic recalibration methods. Additionally, produces plots to assess the potential for recalibration to improve the clinical utility of a risk model. Methods are described in detail in Mishra, A. (2019) ``Methods for Risk Markers that Incorporate Clinical Utility'' <http://hdl.handle.net/1773/44068>.

License GPL-2

Encoding UTF-8

LazyData true

Imports lattice, caret, ggplot2, stats, cowplot, nloptr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-04-15 11:10:02 UTC

Contents

calCurvPlot 2
calWt 4
constRecal 6
cvRepWtTuning 8
cvWtTuning 10
fakeData 12
nb 12
RAWgrid 13
snbRecalPlot 15
stdRecal 16
wtRecal 17

Index **20**

`calCurvPlot` *Calibration Curve Plot*

Description

Flexible function for plotting the calibration curve for a risk score and different recalibrations of a risk score. Histograms of risk score and recalibrated risk scores can be plotted using argument "hist==TRUE". Dotted guidelines for the risk threshold R are plotted and clinically releavant risk region [R_l, R_u] are plotted to help gauge calibration in clinically relevants.

Usage

```
calCurvPlot(y,p,p.std=NULL,p.recal=NULL,
            stdPlot=FALSE, recalPlot=FALSE,
            xlim=c(0,1),ylim=c(0,1),
            label="Original Risk Score",
            label2 = "Standard Recalibrated Risk Score",
            label3 = "Weighted/Constrained Recalibrated Risk Score",
            legendLab = c("Orig.", "Std.", "Wt."),
            mainTitle="Calibration of Risk Score",
            hist=TRUE,ylimHist = c(0,0.5),
            r,rl = -Inf, ru = Inf)
```

Arguments

`y`	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
`p`	Vector of risk score values
`p.std`	Vector of risk score values after standard logistic recalibration
`p.recal`	Vector of risk score values after weighted/contratined logistic recalibration
`stdPlot`	If TRUE plot calibration curve for standard logistic recalibrated risk score
`recalPlot`	If TRUE plot calibration curve for weighted/contratined recalibrated risk score
`xlim`	Limits for x-axis
`ylim`	Limits for y-axis
`label`	Label for x-axis corresponding to p vector
`label2`	Label for x-axis corresponding to p.std vector
`label3`	Label for x-axis corresponding to p.recal vector
`legendLab`	Label for legend
`mainTitle`	Main title for plot
`hist`	If true plot distribution of risk scores along with calibration curve
`ylimHist`	Limits for y-axis of histogram

r	Clinically relevant risk threshold used indicated with dotted line
rl	Lower bound of clinically relevant region indicated with dotted-dashed line
ru	Upper bound of clinically relevant region indicated with dotted-dashed line

Value

Prints calibration plot of risk model p, and optional histogram of risk score

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

Examples

```
## Not run:
data("fakeData")

### get standard recalibrated risk score
stdRecal.res <- stdRecal(y = fakeData$y,p = fakeData$p)
p.std <- stdRecal.res$p.std

### Select tuning parameter lamba using 5-fold cross-validation repeated 25 times
grid <- RAWgrid(r = 0.3,rl = -Inf,ru = Inf,p = fakeData$p,y = fakeData$y,
                cvParm = "lambda",rl.raw = 0.25,ru.raw = 0.35)
repCV <- cvRepWtTuning(y = fakeData$y,p = fakeData$p,rl = -Inf,ru = Inf,r = 0.3,
                  kFold = 5,cvRep = 25,cvParm = "lambda",tuneSeq = grid,stdErrRule = TRUE)

## Implement weighted logistic recalibration
wtRecal.res <- wtRecal(y = fakeData$y,p = fakeData$p,r = 0.3,rl = -Inf,ru = Inf,
                       lambda = repCV$cv.lambda,delta=1)
p.recal <- wtRecal.res$p.wt

### Calibration curve of only original risk score with histogram
calCurvPlot(y=fakeData$y,p=fakeData$p,p.std=NULL,p.recal=NULL,
            stdPlot=FALSE, recalPlot=FALSE,
            xlim=c(0,1),ylim=c(0,1),
            label="Original Risk Score",
            label2 = "Standard Recalibrated Risk Score",
            label3 = "Weighted/Constrained Recalibrated Risk Score",
            legendLab = c("Orig.", "Std.", "Wt."),
            mainTitle="Calibration of Risk Score",
            hist=TRUE,ylimHist = c(0,0.5),
            r=0.3,rl = -Inf, ru = Inf)

### Calibration curve of only original, standard and weighted recalibrated risk score
calCurvPlot(fakeData$y,p=fakeData$p,p.std=p.std,p.recal=p.recal,
```

```
            stdPlot=TRUE, recalPlot=TRUE,
            xlim=c(0,1),ylim=c(0,1),
            label="Original Risk Score",
            label2 = "Standard Recalibrated Risk Score",
            label3 = "Weighted/Constrained Recalibrated Risk Score",
            legendLab = c("Orig.", "Std.", "Wt."),
            mainTitle="Calibration of Risk Score",
            hist=TRUE,ylimHist = c(0,0.5),
            r=0.3,rl = -Inf, ru = Inf)

## End(Not run)
```

`calWt` *Calibration Weights*

Description

Calculates observation weights used for weighted calibration method using LOESS smoother. Observations with predicted risks outside clinically relevant interval [Rl,Ru] are downweighted.

Usage

```
calWt(rl,ru,p,y,r,lambda,delta,returnSmoothedEvent=FALSE)
```

Arguments

`rl`	Lower bound of clinically relevant region
`ru`	Upper bound of clinically relevant region
`p`	Vector of risk score values
`y`	Vector of binary outcomes, with 1 indicating event (cases) and 0 indicating no event (controls)
`r`	Clinically relevant risk threshold
`lambda`	Tuning parameter for weights inside relevant region
`delta`	Weight assigned to observations outside relevant region
`returnSmoothedEvent`	If TRUE returns smoothed observed event rate used for calculationg weights

Details

Computes observation weights for weighted recalibration method. Observations with risk scores near a pre-defined clinically relevant risk threshold r are given weights near 1, while observations far from the risk threshold are down-weighted.

For observations with risk scores within a pre-defined clinically relevant region [R_l, R_u], the weighting function follows an exponential decay form with observations recieving smaller weights as they move farther from the clincally relevant risk threhsold. Distance is measured by the squared

difference between the risk threshold and $o(p_i)$, where $o(p_i)$ isa smoothed observed event rate,obtained via LOESS regression of y on the risk scores p_i.

The amount of decay is set by the tuning parameter λ, with larger lambda indicating little down-weighting applied. For observations outside the clinically relevant risk interval, a weight of $delta$ is assigned. An indicator type weight can be choosen by select large λ (e.g $\lambda = 10$). Clinically relevant region can be [R_l, R_u] = [$-\infty$,∞], and in these cases it is not necessary to specify $delta$. See Mishra et al (2020) for functional form of weights and more details.

Value

`wt`	Vector of weights, between 0 and 1, to be used for weighted recalibration
`o`	Vector of smoothed observed event rates

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

See Also

`wtRecal`

Examples

```
#load data
data(fakeData)

#exponential decay type weight
expWt <- calWt(rl=-Inf,ru=Inf,p=fakeData$p,y=fakeData$y,r=0.2,
               lambda=0.05,delta=0.2,returnSmoothedEvent = TRUE)
plot(expWt$o,expWt$wt,xlab="Smoothed Observed Event Rate",
     ylab="Calibration Weight",ylim=c(0,1))


#indicator type weight
indWt <- calWt(rl=0.1,ru=0.3,p=fakeData$p,y=fakeData$y,r=0.2,
               lambda=0.05,delta=0.2,returnSmoothedEvent = TRUE)
plot(indWt$o,indWt$wt,xlab="Smoothed Observed Event Rate",
    ylab="Calibration Weight",ylim=c(0,1))
```

`constRecal` *Constrained Logistic Recalibration*

Description

When recalibrating a risk model, where the intended purpose of the risk model is to prescribe an intervention to those deemed at high risk, it is desirable to have good calibration at the clinically relevant threshold used to define high risk (i.e. threshold used to identify who recieves treatment). This can be achieved by maximizing the clinical utilty of a risk model, which should in turn produce good calibration near the clinical relevant risk threshold. This function realibrates risk scores (predicting binary outcome) by estimating the recalibration intercept and slope by maximizing the logistic likelihood over a restricted parameter space (Mishra et al. [2020]).

The restricted space only includes recalibration parameters $alpha_0$ and $alpha_1$ that produce a recalibrated risk model with high sNB. The restricted parameter space is defined all $alpha$ parameters that result in a recalibrated risk score within one-standard-error of the maximum possible sNB. See Mishra et al (2020), for full details

Usage

```
constRecal(y,p,r,int=NULL,alphaLB=c(-10,0),
           alphaUB=c(10,10),ftol=1e-8,xtol=1e-4,maxeval=1e6)
```

Arguments

`y`	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
`p`	Vector of risk score values
`r`	Clinically relevant risk threshold
`int`	Two-dimensional vector of initial recalibration parameter ($alpha$) values. If not specified the recalibration parameters estimated under standard logistic recalibration will be used
`alphaLB`	Lower bound of box-contrained search space
`alphaUB`	Upper bound of box-contrained search space
`ftol`	Controls tolerance of optimization procedure with respect to changes in sNB
`xtol`	Controls tolerance of optimization procedure with respect to changes in alpha
`maxeval`	Maximum number of interations performed during optimization procedure

Details

To solve this optimization problem the DIRECT optimization method is implemented via the NLOPTR package. See Jones et al (1993) and Ypma et al (2014) for full description of optimization method and implementation details. Note this is not a convex optimization problem, so a global optimizer is used.

Value

alpha	Recalibration parameters obtained from Constrained Logistic Recalibration
y	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
p.const	Vector of constrained logistic reclaibrated risk scores
snbLB	Lower bound used to define constraint region. The solution is constrained to the region where recalibration parameters result in risk model with sNB greater or equal to this bound. This is one-standard error lower than the maximum sNB
optimRes	Full output from the NLOPTR optimization routine

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

Ypma, J., Borchers, H. W., Eddelbuettel, D., & Ypma, M. J. (2020). Package 'nloptr'.

Ypma, J. (2014). Introduction to nloptr: an R interface to NLopt. Tech. rep.

D. R. Jones, C. D. Perttunen, and B. E. Stuckmann, "Lipschitzian optimization without the lipschitz constant," J. Optimization Theory and Applications, vol. 79, p. 157 (1993).

Examples

```
## Not run:
### Load data ##
data(fakeData)

## Implementing standard logistic recalibration
stdRecal.res <- stdRecal(y = fakeData$y,p = fakeData$p)
stdRecal.res$alpha #standard recalibration parameters
p.std <- stdRecal.res$p.std

## Look at potential sNB under recalibration plot
snbRecalPlot(p = fakeData$p,p.std = p.std,y = fakeData$y,r = 0.3)

## Implementing constrained logistic recalibration
constRecal.res <- constRecal(y = fakeData$y,p = fakeData$p,r = 0.3)
constRecal.res$alpha #constrained logistic recalibration parameters
p.recal <- constRecal.res$p.const

## comparing standardized net benefit of the two
nb(y = fakeData$y,p = fakeData$p,r = 0.3)$snb #original
nb(y = stdRecal.res$y,p = stdRecal.res$p.std,r = 0.3)$snb #std recal
nb(y = constRecal.res$y,p = constRecal.res$p.const,r = 0.3)$snb #weighted

## Generate calibration plots
```

```
### Calibration curve of only original, standard and weighted recalibrated risk score
calCurvPlot(y=fakeData$y,p=fakeData$p,p.std=p.std,p.recal=p.recal,
            stdPlot=TRUE, recalPlot=TRUE,
            xlim=c(0,1),ylim=c(0,1),
            label="Original Risk Score",
            label2 = "Standard Recalibrated Risk Score",
            label3 = "Weighted/Constrained Recalibrated Risk Score",
            legendLab = c("Orig.", "Std.", "Wt."),
            mainTitle="Calibration of Risk Score",
            hist=TRUE,ylimHist = c(0,0.5),
            r=0.3,rl = -Inf, ru = Inf)


## End(Not run)
```

`cvRepWtTuning` *Repeated Cross Validation for Weight Tuning Parameter Selection*

Description

Calibration weights require specification of tuning parameter $delta$ or $lambda$. Since a single round of cross-validation can be noisy, cross-validation can be repeated multiple times with independent random partitions and the results be averaged. This function implements a repeated K-fold cross-validation where tuning parameter $labmda$ or $delta$ is selected by maximizing standardized net benefit (sNB) (i.e. repeated `cvWtTuning` procedure).

A a "one-standard error" rule can be used for selecting tuning parameters. Under the "one-standard error" rule the calibration weight tuning parameter ($lambda$ or $delta$) is selected such that corresponding cross-validated sNB is within one-standard deviation of the maximum cross-validated sNB. This provides protection against overfitting the data and selecting a tuning parameter that is too extreme. If the "one-standard error" rule is not implemented, then the tuning parameter with the larged average cross-validted sNB (across folds and repetition) will be selected.

Usage

```
cvRepWtTuning(y,p,r,rl,ru,kFold=5,cvRep=25,cvParm,tuneSeq,stdErrRule=TRUE,int.seed=11111)
```

Arguments

`y`	Vector of binary outcomes, with 1 indicating event (cases) and 0 indicating no event (controls)
`p`	Vector of risk score values
`r`	Clinically relevant risk threshold
`rl`	Lower bound of clinically relevant region
`ru`	Upper bound of clinically relevant region
`kFold`	Number of folds for cross-validation

cvRep	Number of cross-validation repititions
cvParm	Parameter to be selected via cross-validation. Can be either $delta$ the weight assigned to observations outside the clinically relevant region [R_l,R_u], or the $lambda$ tuning parameter controlling exponential decay within the clinically relevant region [R_l,R_u]
tuneSeq	Sequence of values of tuning parameters to perform cross-validation over
stdErrRule	Use "one-standard" error rule selecting tuning parameter
int.seed	Intial seed set for random splitting of data into K folds

Details

To estimate the standard deviation of the cross-validated sNV, the dependence between the different partitions of cross-validation needs to be accounted for. Gelman (1992) give a variance estimator of convergence diagnostic statistic used when Markov Chain Monte Carlo with multiple chains are performed. The variance estimator accounts for both the variability of the statistic “within" a single chain, and the variance of the statistic across, or “between", chains. Analogously, we can use this framework to estimate the “within" repetition variance (i.e. variation in sNB from a single round of K-fold cross-validation) and the “between" repetition variance. We denote the ‘within" repetition variance as W and the “between" repetition variance as B . We augment this formula slightly from that given in Gelman (1992) to account for the fact that as the number of cross-validation repetitions increases, the between-repetition variability should decrease. See Mishra et al (2020) for full expressions of B and W.

Value

cv.sNB	Standardized net benefit (sNB) of tuning parameter selected via cross-validatoin
cv.RAW	Corresponding RAW value given cross-valiated selected tuning parameter
cv.lambda	$lambda$ value selected via cross-validation if $cvParm = lambda$, otherwise user specified $lambda$ value
cv.delta	$delta$ value selected via cross-validation if $cvParm = delta$, otherwise user specified $lambda$ value
avgCV.res	Averaged (across-replications) cross-validated sNB for sequence of tuning parameters
W	Estimate of "with-in" repetition variance. Will only return if stdErrRule==TRUE
B	Estimate of "between" repetition variance. Will only return if stdErrRule==TRUE
fullList	List of cross-valiation results for all fold and repititions

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. Statistical science, 7(4), 457-472.

See Also

calWt, RAWgrid, nb, cvWtTuning

Examples

```
### Load data ##
## Not run:
data(fakeData)

### Get grid of tuning parameters  ###
grid <- RAWgrid(r = 0.3,rl = -Inf,ru = Inf,p = fakeData$p,y = fakeData$y,
                cvParm = "lambda",rl.raw = 0.25,ru.raw = 0.35)

### Implement repeated k-fold cross validation
repCV <- cvRepWtTuning(y = fakeData$y,p = fakeData$p,rl = -Inf,ru = Inf,r = 0.3,
                  kFold = 5,cvRep = 25,cvParm = "lambda",tuneSeq = grid,stdErrRule = TRUE)

## cross-validation results
repCV$avgCV.res

## cross-validation selected lambda, RAW, and sNV
cv.lambda <- repCV$cv.lambda
cv.RAW <- repCV$cv.RAW
cv.RAW <- repCV$cv.sNB

## End(Not run)
```

---

cvWtTuning *Cross-validation for Selecting Weight Tuning Parameter*

Description

Calibration weights require specification of tuning parameter $delta$ or $lambda$. This function uses K-fold cross-validation to select tuning parameter used for calibration weights, with standardized net benfeit (sNB) as objective function. Either one of $delta$ or $lambda$ must be specificed. The sequence of tuning parameters can be obtained from the RAWgrid function.

Usage

```
cvWtTuning(p,y,r,rl,ru,kFold=5,cvParm,tuneSeq,cv.seed=1111)
```

Arguments

y	Vector of binary outcomes, with 1 indicating event (cases) and 0 indicating no event (controls)
p	Vector of risk score values
r	Clinically relevant risk threshold
rl	Lower bound of clinically relevant region
ru	Upper bound of clinically relevant region
kFold	Number of folds for cross-validation
cvParm	Parameter to be selected via cross-validation. Can be either $delta$ the weight assigned to observations outside the clinically relevant region [R_l,R_u], or the $lambda$ tuning parameter controlling exponential decay within the clinically relevant region [R_l,R_u]
tuneSeq	Sequence of values of tuning parameters to perform cross-validation over
cv.seed	Intial seed set for random splitting of data into K folds

Value

cv.res	Matrix containing sequence of tuning parameters and corresponding cross-validation sNB
cv.param	Value of tuning parameter selected via cross validation
cv.full	Matrix of cross-validation results for all folds

Note

Note this function does not split data into training and validaion set, but performs the K-fold cross-validation procedure on all data included. We advise that a separate, validation subset should be split from the data used in this function.

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

See Also

calWt, RAWgrid, nb, cvRepWtTuning

fakeData *Dataset for Recalibration Purposes*

Description

Simulated dataset that can be used for recalibration purposes.

Usage

```
data("fakeData")
```

Format

A data frame with 1000 observations on the following 2 variables.

`p` a numeric vector or risk scores bounded between 0 and 1

`y` a numeric vector of indicator of events (cases) and non-events (control)

Details

See Mishra et al (2020) for description of simuation settings to generate data (simulation example 3).

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

Examples

```
data(fakeData)
```

nb *Net Benefit*

Description

Estimates the net benefit and standardized net benefit of a risk score given a risk threshold R. Additionally estimates the standardized net benefit of a "treat-all" (everyone recieves intervention) treatment rule.

Usage

```
nb(y, p, r)
```

Arguments

y	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
p	Vector of risk score values
r	Clinically relevant risk threshold

Value

nb	Net benefit of risk score p
snb	Standardized net benefit of risk score p. Net benefit of p dived by the prevalence
snb.all	Standardized net benefit of a decision rule where everyone recieves treatment

Author(s)

Anu Mishra

References

Pauker, S. G., & Kassirer, J. P. (1980). The threshold approach to clinical decision making. New England Journal of Medicine, 302(20), 1109-1117.

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

RAWgrid *RAW grid for Cross Validation*

Description

Tuning parameters for calibration weights may not be intutitive. Instead, relative average weight (RAW) can be used to find sensible tuning parameters. The RAW is the average weight of observations within the RAW region, $[R_{l,RAW}, R_{u,RAW}]$, divided by the average weight of observations outside RAW region.

This function elicits a grid of tuning parameter $lambda$ or $delta$ given a sequence of relative average weights (RAW) values, to be used for cross-validation. One tuning parameter $delta$ or $lambda$ must be specified.

Two clinically relevant lower and upper bound values must be specified. The first, $[R_l, R_u$, define the clinically relevant region used for the weighting function. The second, $[R_{l,RAW}, R_{u,RAW}]$, define the complemenatry regions used for defining relative average weights.

Usage

```
RAWgrid(r, rl, ru, p, y, rawSeq=seq(0.1,0.9,0.1), cvParm,
delta = NULL, lambda = NULL, rl.raw, ru.raw)
```

Arguments

r	Clinically relevant risk threshold
rl	Lower bound of clinically relevant region
ru	Upper bound of clinically relevant region
p	Vector of risk score values
y	Vector of binary outcomes, with 1 indicating event (cases) and 0 indicating no event (controls)
rawSeq	Sequence of relative average weights
cvParm	Tuning parameter that will be selected via cross-validation. Can either be $lambda$ or $delta$
delta	Calibration weight for observations outside clinically relevant region $[R_l, R_u]$. Needs to be specified if cvParm="lambda", otherwise should be left blank, and function will generate a sequence of delta for cross-validaton.
lambda	Tuning parameter for controlling exponential decay of calibration weights. Needs to be specified if cvParm="delta", otherwise should be left blank, and function will generate a sequence of lambda for cross-validaton.
rl.raw	Lower bound for defining the relative average weight region of interest
ru.raw	Upper bound for defining the relative average weight region of interest

Value

rrWt.seq	Matrix containing the input RAW sequence, and corresponding weight tuning parameter lambda and weight value delta (with one fixed depending on cvParm selection). If 'NA' is returned there may be too few events within the weight clinically relevant RAW region, meaning that RAW value is not possible

Warning

Event rate outside RAW interval, widen RAW intervalIf no or too cases are inside RAW interval [R_l,R_u], cross-validation procedure may not be stable, so this warning indicates RAW interval should widen to include more cases.

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

See Also

calWt, cvWtTuning, cvRepWtTuning

Examples

```
### Load data ##
data(fakeData)

### Get Grid of Tuning Parameters ###
grid <- RAWgrid(r = 0.3,rl = -Inf,ru = Inf,p = fakeData$p,y = fakeData$y,
                cvParm = "lambda",delta=1,rl.raw = 0.25,ru.raw = 0.35)
```

`snbRecalPlot` *Plot of Potential sNB Under Recalibration*

Description

Function for producing a graphical device to help assess the potential for recalibration to improve the clinical utility of a risk model.

This function plots the sNB for a given risk score (p) and cost benefit ratio (R/1-R) against the threshold used to perscribe intervention. The peak of the curve estimates the maxmium possible sNB that can be obtained via recalibration. The estimated sNB of the original risk score and the recalibrated risk score under standard logistic recalibration are also plotted on the curves on the curves. The dotted lined "stdErrThresh" controls. The plot includes a dotted horizontal line one standard error (or any number of standard errors, controlled by stdErrThresh) below the maximum, to help guage much room for improvement in sNB there is for original or standard recalibrated risk score from maximum possible sNB.

Usage

```
snbRecalPlot(p,p.std,y,r,stdErrThresh=1,ylim=NULL,
            titlePlot = "Potential sNB Under Recalibration",risk.model.std=TRUE)
```

Arguments

`p`	Vector of risk score values
`p.std`	Vector of risk score values after standard logistic recalibration
`y`	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
`r`	Clinically relevant risk threshold
`stdErrThresh`	Indicates how many standard errors line drawn below the maximum of the sNB curve should be
`ylim`	Y axis limits
`titlePlot`	Title for plot
`risk.model.std`	Plot standard error bars for sNB of plotted risk models

Value

Prints plot of potential sNB of risk score p under recalibration

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

Examples

```
data("fakeData")

### get standard recalibrated risk score
stdRecal.res <- stdRecal(y = fakeData$y,p = fakeData$p)
p.std <- stdRecal.res$p.std


## make plot
snbRecalPlot(p = fakeData$p,p.std = p.std,y = fakeData$y,r = 0.3)

## both original and std logistic recalibrated risk score are below 1 std err from maximum
## indicating that alternative recalibration methods could improve
```

---

stdRecal *Standard Logistic Recalibration*

Description

Produces recalibration intercept, slope and corresponding recalibrated risk scores using standard logistic rexalibration. Recalibration slope and intercept using logistic recalibration method, develop by Cox (1958), then used to calculated the recalibrated risk score

Usage

```
stdRecal(y,p)
```

Arguments

y	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
p	Vector of risk score values

Value

stdRisk	Vector of recalibrated risks under standard logistic recalibration
alpha	Two-element vector containing logistic recalibration intercept and slope

Author(s)

Anu Mishra

References

Cox, D. R. (1958). Two further applications of a model for binary regression. Biometrika, 45(3/4), 562-565.

Harrell Jr, F. E. (2015). Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis. Springer.

Examples

```
#load data
data(fakeData)

res <- stdRecal(p=fakeData$p, y =fakeData$y)
p.std <- res$stdRecal
coef.dt <- res$alpha
```

`wtRecal` *Weighted Logistic Recalibration*

Description

When recalibrating a risk model, where the intended purpose of the risk model is to prescribe an intervention to those deemed at high risk, it is desirable to have good calibration at the clinically relevant threshold used to define high risk (i.e. threshold used to identify who recieves treatment). This function realibrates risk scores (predicting binary outcome) using the weighted logistic recalibration method (Mishra et al. [2020]).

Under this method, a recalibration intercept and slope are estimated via weighted logistic regression. Weights are constructed such that observations further from the clinically relevant risk threshold are down-weighted, meaning observations closer to the clinically relevant risk threshold have higher contribution to the risk threshold. The resulting estimated weighted recalibration slope and intercept are used to scale and shift the existing risk score, producing better calibrated risk scores near the risk threhold and potentially increaseing the net benefit of the risk score.

Usage

```
wtRecal(y,p,r,rl,ru,lambda,delta)
```

Arguments

`y`	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
`p`	Vector of risk score values
`r`	Clinically relevant risk threshold

rl	Lower bound of clinically relevant region
ru	Upper bound of clinically relevant region
lambda	Tuning parameter for weights inside relevant region
delta	Weight assigned to observations outside relevant region

Value

y	Vector of binary outcomes, with 1 indicating event (case) and 0 indicating no event (controls)
p.wt	Vector of weighted reclaibrated risk scores
alpha.wt	Weighted recalibration slope and intercept
wt	Calibration weights
wt.conv	Indicator of convergence of the weighted logistic regression model. 1 indicates model convergence, 0 indicates model did not converge

Author(s)

Anu Mishra

References

Mishra, A. (2019). Methods for Risk Markers that Incorporate Clinical Utility (Doctoral dissertation). (Available Upon Request)

See Also

calWt

Examples

```
## Not run:
### Load data ##
data(fakeData)

## Implement standard logistic recalibration
stdRecal.res <- stdRecal(y = fakeData$y,p = fakeData$p)
stdRecal.res$alpha #standard recalibration parameters
p.std <- stdRecal.res$p.std

## Look at potential sNB under recalibration plot
snbRecalPlot(p = fakeData$p,p.std = p.std,y = fakeData$y,r = 0.3)
## both original and std logistic recalibrated risk score are below 1 std err from maximum
## indicating that alternative recalibration methods could improve

### Get grid of tuning parameters  ###
## In this example keeping delta fixed, and selecting lambda

grid <- RAWgrid(r = 0.3,rl = -Inf,ru = Inf,p = fakeData$p,y = fakeData$y,
                cvParm = "lambda",rl.raw = 0.25,ru.raw = 0.35)
```

```
### Select tuning parameter lamba using 5-fold cross-validation repeated 25 times
### with one standard error rule implemented

repCV <- cvRepWtTuning(y = fakeData$y,p = fakeData$p,rl = -Inf,ru = Inf,r = 0.3,
                  kFold = 5,cvRep = 25,cvParm = "lambda",tuneSeq = grid,stdErrRule = TRUE)

## Implement weighted logistic recalibration
wtRecal.res <- wtRecal(y = fakeData$y,p = fakeData$p,r = 0.3,rl = -Inf,ru = Inf,
                       lambda = repCV$cv.lambda,delta=1)
# note that delta here is set to 1 as a place holder, but not used since the clinically
# relevant region is [-Inf, Inf]
wtRecal.res$alpha.wt #weighted recalibration parameters
p.recal <- wtRecal.res$p.wt

## comparing standardized net benefit of the two
nb(y = fakeData$y,p = fakeData$p,r = 0.3)$snb #original
nb(y = stdRecal.res$y,p = stdRecal.res$p.std,r = 0.3)$snb #std recal
nb(y = wtRecal.res$y,p = wtRecal.res$p.wt,r = 0.3)$snb #weighted

### Calibration curve of only original, standard and weighted recalibrated risk score
calCurvPlot(y = fakeData$y,p = fakeData$p,p.std=p.std,p.recal=p.recal,
            stdPlot=TRUE, recalPlot=TRUE,
            xlim=c(0,1),ylim=c(0,1),
            label="Original Risk Score",
            label2 = "Standard Recalibrated Risk Score",
            label3 = "Weighted/Constrained Recalibrated Risk Score",
            legendLab = c("Orig.", "Std.", "Wt."),
            mainTitle="Calibration of Risk Score",
            hist=TRUE,ylimHist = c(0,0.5),
            r=0.3,rl = -Inf, ru = Inf)

## End(Not run)
```

Index

∗ **datasets**
fakeData, 12

calCurvPlot, 2
calWt, 4, *10, 11*, *14*, *18*
constRecal, 6
cvRepWtTuning, 8, *11*, *14*
cvWtTuning, *8*, *10*, 10, *14*

fakeData, 12

nb, *10, 11*, 12

RAWgrid, *10, 11*, 13

snbRecalPlot, 15
stdRecal, 16

wtRecal, *5*, 17