Package ‘caROC’

October 12, 2022

Type Package

Title Continuous Biomarker Evaluation with Adjustment of Covariates

Version 0.1.5

Author Ziyi Li

Maintainer Ziyi Li <zli16@mdanderson.org>

Description Compute covariate-adjusted specificity at controlled sensitivity level, or covariate-adjusted sensitivity at controlled specificity level, or covariate-adjust receiver operating characteristic curve, or covariate-adjusted thresholds at controlled sensitivity/specificity level. All statistics could also be computed for specific sub-populations given their covariate values. Methods are described in Ziyi Li, Yijian Huang, Datta Patil, Martin G. Sanda (2021+) ``Covariate adjustment in continuous biomarker assessment''.

License GPL-2

Encoding UTF-8

Depends R (>= 4.0), quantreg, RColorBrewer

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2021-04-02 08:20:03 UTC

R topics documented:

caROC *Covariate-adjusted ROC*

Description

Compute covariate-adjusted specificity at controlled sensitivity level, or covariate-adjusted sensitivity at controlled specificity level, or covariate-adjust receiver operating characteristic curve.

Usage

```
caROC(diseaseData, controlData, userFormula, control_sensitivity = NULL,
control_specificity = NULL, mono_resp_method = "ROC",
whichSE = "sample", global_ROC_controlled_by = "sensitivity",
nbootstrap = 100, CI_alpha = 0.95, logit_CI = TRUE,
verbose = TRUE)
```

Arguments

`diseaseData` Data from patients including dependent (biomarker) and independent (covariates) variables.

`controlData` Data from controls including dependent (biomarker) and independent (covariates) variables.

`userFormula` A character string to represent the function for covariate adjustment. For example, let Y denote biomarker, Z1 and Z2 denote two covariates. Then userFormula = "Y ~ Z1 + Z2".

`control_sensitivity` The level(s) of sensitivity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).

`control_specificity` The level(s) of specificity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).

`mono_resp_method` The method used to restore monotonicity of the ROC curve or computed sensitivity/specificity value. It should one from the following: "none", "ROC". "none" is not applying any monotonicity respecting method. "ROC" is to apply ROC-based monotonicity respecting approach. Default value is "ROC".

`whichSE` The method used to compute standard error. It should be one from the following: "sample", "bootstrap", meaning to calculate the standard error using sample-based approach or bootstrap. Default is "sample".

`global_ROC_controlled_by` Whether sensitivity/specificity is used to control when computing global ROC. It should one from the following: "sensitivity", "specificity". Default is "sensitivity".

`nbootstrap` Number of boostrap iterations. Default is 100.

`CI_alpha` Percentage of confidence interval. Default is 0.95.

logit_CI	Whether to apply logit-based confidence interval. Logit-transformed CI has been identified to be more robust near border area.
verbose	Whether to print out messages. Default value is true.

Value

If control_sensitivity or control_specificity is provided, compute covariate-adjusted specificity (sensitivity) at controlled sensitivity (specificity) level.

Estimate	Covariate-adjusted sensitivity/specificity.
SE	Estimated standard error.
CI	Estimated confidence intervals.

If both control_sensitivity and control_specificity are null, compuate covariate-adjusted ROC curve.

sensitivity	Estimated sensitivity.
specificity	Estimated specificity.
mono_adj	Monotonicity adjustment method.

Author(s)

Ziyi.li <ziyi.li@emory.edu>

Examples

```
n1 = n0 = 500

## generate data
Z_D <- rbinom(n1, size = 1, prob = 0.3)
Z_C <- rbinom(n0, size = 1, prob = 0.7)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C == 0) + Y_C_Z1 * (Z_C == 1)
M1 <- Y_D_Z0 * (Z_D == 0) + Y_D_Z1 * (Z_D == 1)

diseaseData <- data.frame(M = M1, Z = Z_D)
controlData <- data.frame(M = M0, Z = Z_C)
userFormula = "M~Z"

## calculate covariate-adjusted specificity at
## controlled sensitivity levels (0.2, 0.8, 0.9)
caROC(diseaseData,controlData,userFormula,
      control_sensitivity = c(0.2,0.8, 0.9),
      control_specificity = NULL,mono_resp_method = "ROC",
      whichSE = "bootstrap",nbootstrap = 100,
      CI_alpha = 0.95, logit_CI = TRUE)
```

```
## calculate covariate-adjusted sensitivity at
## controlled specificity levels (0.2, 0.8, 0.9)
caROC(diseaseData,controlData,userFormula,
      control_sensitivity = NULL,
      control_specificity = c(0.7,0.8, 0.9),mono_resp_method = "none",
      whichSE = "sample",nbootstrap = 100,
      CI_alpha = 0.95, logit_CI = TRUE)

## calculate the whole covariate-adjusted ROC curve
ROC1 <- caROC(diseaseData,controlData,userFormula = "M~Z",
                 mono_resp_method = "none")
ROC2 <- caROC(diseaseData,controlData,userFormula = "M~Z",
                 mono_resp_method = "ROC")
```

---

caROC_CB *Get confidence band for covariate-adjusted ROC curve.*

Description

Use this function to compute the confidence band for covariate-adjusted ROC curve, with or without monotonicity respecting methods.

Usage

```
caROC_CB(diseaseData, controlData, userFormula,
mono_resp_method, global_ROC_controlled_by = "sensitivity",
CB_alpha = 0.95, logit_CB = FALSE, nbootstrap = 100,
nbin = 100, verbose = FALSE)
```

Arguments

diseaseData	Data from patients including dependent (biomarker) and independent (covariates) variables.
controlData	Data from controls including dependent (biomarker) and independent (covariates) variables.
userFormula	A character string to represent the function for covariate adjustment. For example, let Y denote biomarker, Z1 and Z2 denote two covariates. Then userFormula = "Y ~ Z1 + Z2".
mono_resp_method	The method used to restore monotonicity of the ROC curve or computed sensitivity/specificity value. It should one from the following: "none", "ROC". "none" is not applying any monotonicity respecting method. "ROC" is to apply ROC-based monotonicity respecting approach. Default value is "ROC".
global_ROC_controlled_by	Whether sensitivity/specificity is used to control when computing global ROC. It should one from the following: "sensitivity", "specificity". Default is "sensitivity".

`CB_alpha`	Percentage of confidence band. Default is 0.95.
`logit_CB`	Whether to use logit-transformed (then transform back) confidence band. Default is FALSE.
`nbootstrap`	Number of boostrap iterations. Default is 100.
`nbin`	Number of bins used for constructing confidence band. Default is 100.
`verbose`	Whether to print out messages during bootstrap. Default value is FALSE.

Value

If global ROC is controlled by sensitivity, a list will be output including the following

`Sensitivity`	Vector of sensitivities;
`Specificity_upper`	Upper confidence band for specificity estimations;
`Specificity_lower`	Lower confidence band for specificity estimations;
`global_ROC_controlled_by`	"sensitivity".

If global ROC is controlled by Specificity, a list will be output including the following

`Specificity`	Vector of specificity;
`Sensitivity_upper`	Upper confidence band for sensitivity estimations;
`Sensitivity_lower`	Lower confidence band for sensitivity estimations;
`global_ROC_controlled_by`	"specificity".

Author(s)

Ziyi.li <ziyi.li@emory.edu>

Examples

```
n1 = n0 = 500

## generate data
Z_D <- rbinom(n1, size = 1, prob = 0.3)
Z_C <- rbinom(n0, size = 1, prob = 0.7)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C == 0) + Y_C_Z1 * (Z_C == 1)
M1 <- Y_D_Z0 * (Z_D == 0) + Y_D_Z1 * (Z_D == 1)
```

```
diseaseData <- data.frame(M = M1, Z = Z_D)
controlData <- data.frame(M = M0, Z = Z_C)
userFormula = "M~Z"

### calculate confidence band by controlling sensitivity
### using different monotonicity respecting methods


ROC_CB1 <- caROC_CB(diseaseData,controlData,userFormula,
                    mono_resp_method = "none",
                    CB_alpha = 0.95,
                    nbin = 100,verbose = FALSE)
ROC_CB2 <- caROC_CB(diseaseData,controlData,userFormula,
                    mono_resp_method = "ROC",
                    CB_alpha = 0.95,
                    nbin = 100,verbose = FALSE)
```

---

caThreshold *Calculate covariate-adjusted threshold.*

Description

This function is used to calculate covariate-adjusted threshold(s) at controlled sensitivity levels or specificity levels.

Usage

```
caThreshold(userFormula, new_covariates, diseaseData = NULL,
controlData = NULL, control_sensitivity = NULL,
control_specificity = NULL)
```

Arguments

`userFormula` A character string to represent the function for covariate adjustment. For example, let Y denote biomarker, Z1 and Z2 denote two covariates. Then userFormula = "Y ~ Z1 + Z2".

`new_covariates` A data frame containing covariates for new data. For example, if my userFormula is "Y ~ Z1 + Z2", new_covariates could be data.frame(Z1 = rnorm(100), Z2 = rnorm(100)).

`diseaseData` Data from patients including dependent (biomarker) and independent (covariates) variables.

`controlData` Data from controls including dependent (biomarker) and independent (covariates) variables.

`control_sensitivity` The level(s) of sensitivity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).

control_specificity
: The level(s) of specificity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).

Value

A vector of covariate-adjusted threshold for all subjects if a scalar sensitivity/specificity is given. A data matrix of covariate-adjusted thresholds for all subjects if a vector of sensitivity/specificity is given.

Author(s)

Ziyi Li <ziyi.li@emory.edu>

Examples

```
n1 = n0 = 500

## generate data
Z_D <- rbinom(n1, size = 1, prob = 0.3)
Z_C <- rbinom(n0, size = 1, prob = 0.7)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C == 0) + Y_C_Z1 * (Z_C == 1)
M1 <- Y_D_Z0 * (Z_D == 0) + Y_D_Z1 * (Z_D == 1)

diseaseData <- data.frame(M = M1, Z = Z_D)
controlData <- data.frame(M = M0, Z = Z_C)
userFormula = "M~Z"

### generate new covariates
new_covariates <- data.frame(Z = rbinom(20, size = 1, prob = 0.5))

### calculate covariate-adjusted thresholds at controlled
### sensitivity level 0.7, 0.8, 0.9
caThreshold(userFormula, new_covariates,
            diseaseData = diseaseData,
            controlData = NULL,
            control_sensitivity = c(0.7,0.8,0.9),
            control_specificity = NULL)

### calculate covariate-adjusted thresholds at controlled
### sensitivity level 0.7
caThreshold(userFormula,new_covariates,
            diseaseData = diseaseData,
            controlData = NULL,
            control_sensitivity = 0.7,
            control_specificity = NULL)
```

```
### calculate covariate-adjusted thresholds at controlled
### specificity level 0.7, 0.8, 0.9
caThreshold(userFormula,new_covariates,
            diseaseData = NULL,
            controlData = controlData,
            control_sensitivity = NULL,
            control_specificity = c(0.7,0.8,0.9))

### calculate covariate-adjusted thresholds at controlled
### specificity level 0.7
caThreshold(userFormula,new_covariates,
            diseaseData = NULL,
            controlData = controlData,
            control_sensitivity = NULL,
            control_specificity = 0.7)
```

---

plot_caROC *Plot covariate-adjusted ROC.*

Description

Function to plot the ROC curve generated from caROC().

Usage

```
plot_caROC(myROC, ...)
```

Arguments

myROC	ROC output from caROC() function.
...	Arguments to tune generated plots.

Details

This function can be used to plot other ROC curve, as long as the input contains two components "sensitivity" and "specificity".

Value

Plot the ROC curve.

Author(s)

Ziyi Li <zli16@mdanderson.org>

Examples

```
n1 = n0 = 500

## generate data
Z_D <- rbinom(n1, size = 1, prob = 0.3)
Z_C <- rbinom(n0, size = 1, prob = 0.7)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C == 0) + Y_C_Z1 * (Z_C == 1)
M1 <- Y_D_Z0 * (Z_D == 0) + Y_D_Z1 * (Z_D == 1)

diseaseData <- data.frame(M = M1, Z = Z_D)
controlData <- data.frame(M = M0, Z = Z_C)
userFormula = "M~Z"

ROC1 <- caROC(diseaseData,controlData,userFormula,
                 mono_resp_method = "none")
ROC2 <- caROC(diseaseData,controlData,userFormula,
                 mono_resp_method = "ROC")

plot_caROC(ROC1)
plot_caROC(ROC2, col = "blue")
```

`plot_caROC_CB` *Plot confidence band of covariate-adjusted ROC.*

Description

A function to plot the confidence band of covariate-adjusted ROC.

Usage

```
plot_caROC_CB(myROC_CB, add = TRUE, ...)
```

Arguments

`myROC_CB`	Output from caROC_CB() function.
`add`	Whether to add confidence band to existing plot (TRUE) or draw a new one (FALSE). Default is TRUE.
`...`	Any parameters related with the plot.

Value

No values will be return. This function is for plotting only.

Author(s)

Ziyi Li<ziyi.li@emory.edu>

Examples

```
library(caROC)
n1 = n0 = 100

## generate data
Z_D <- rbinom(n1, size = 1, prob = 0.3)
Z_C <- rbinom(n0, size = 1, prob = 0.7)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C == 0) + Y_C_Z1 * (Z_C == 1)
M1 <- Y_D_Z0 * (Z_D == 0) + Y_D_Z1 * (Z_D == 1)

diseaseData <- data.frame(M = M1, Z = Z_D)
controlData <- data.frame(M = M0, Z = Z_C)
formula = "M~Z"

ROC_CB1 <- caROC_CB(diseaseData,controlData,formula,
                       mono_resp_method = "none",
                       CB_alpha = 0.95,
                       nbin = 100,verbose = FALSE)
### plot confidence band individually
plot_caROC_CB(ROC_CB1, add = FALSE, lty = 2, col = "blue")

### plot confidence band together with the ROC curve
ROC1 <- caROC(diseaseData,controlData,formula,
                 mono_resp_method = "none", verbose = FALSE)
plot_caROC(ROC1)
plot_caROC_CB(ROC_CB1, add = TRUE, lty = 2, col = "blue")
```

`plot_sscaROC` *Plot covariate-adjusted ROC for specific subpopulations.*

Description

Function to plot the ROC curve generated from sscaROC().

Usage

```
plot_sscaROC(myROC, ...)
```

Arguments

`myROC`	ROC output from sscaROC() function.
`...`	Arguments to tune generated plots.

Details

This function can be used to plot other ROC curve, as long as the input contains two components "sensitivity" and "specificity".

Value

Plot the ROC curve.

Author(s)

Ziyi Li <zli16@mdanderson.org>

Examples

```
n1 = n0 = 1000

## generate data
Z_D1 <- rbinom(n1, size = 1, prob = 0.3)
Z_D2 <- rnorm(n1, 0.8, 1)

Z_C1 <- rbinom(n0, size = 1, prob = 0.7)
Z_C2 <- rnorm(n0, 0.8, 1)

Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C1 == 0) + Y_C_Z1 * (Z_C1 == 1) + Z_C2
M1 <- Y_D_Z0 * (Z_D1 == 0) + Y_D_Z1 * (Z_D1 == 1) + 1.5 * Z_D2

diseaseData <- data.frame(M = M1, Z1 = Z_D1, Z2 = Z_D2)
controlData <- data.frame(M = M0, Z1 = Z_C1, Z2 = Z_C2)
userFormula = "M~Z1+Z2"

target_covariates = c(1, 0.7, 0.9)


myROC <- sscaROC(diseaseData,
                 controlData,
                 userFormula,
                 target_covariates,
                 global_ROC_controlled_by = "sensitivity",
                 mono_resp_method = "none")
plot_sscaROC(myROC, lwd = 1.6)
```

plot_sscaROC_CB — *Plot confidence band of covariate-adjusted ROC in specific subpopulations.*

Description

A function to plot the confidence band of covariate-adjusted ROC in specific subpopulations.

Usage

```
plot_sscaROC_CB(myROC_CB, add = TRUE, ...)
```

Arguments

`myROC_CB`	Output from sscaROC_CB() function.
`add`	Whether to add confidence band to existing plot (TRUE) or draw a new one (FALSE). Default is TRUE.
`...`	Any parameters related with the plot.

Value

No values will be return. This function is for plotting only.

Author(s)

Ziyi Li<zli16@mdanderson.org>

Examples

```
n1 = n0 = 500

## generate data
Z_D1 <- rbinom(n1, size = 1, prob = 0.3)
Z_D2 <- rnorm(n1, 0.8, 1)
Z_C1 <- rbinom(n0, size = 1, prob = 0.7)
Z_C2 <- rnorm(n0, 0.8, 1)
Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C1 == 0) + Y_C_Z1 * (Z_C1 == 1) + Z_C2
M1 <- Y_D_Z0 * (Z_D1 == 0) + Y_D_Z1 * (Z_D1 == 1) + 1.5 * Z_D2

diseaseData <- data.frame(M = M1, Z1 = Z_D1, Z2 = Z_D2)
controlData <- data.frame(M = M0, Z1 = Z_C1, Z2 = Z_C2)

userFormula = "M~Z1+Z2"
target_covariates = c(1, 0.7, 0.9)
```

```
# example that takes more than a minute to run
myROC <- sscaROC(diseaseData,
                 controlData,
                 userFormula,
                 target_covariates,
                 global_ROC_controlled_by = "sensitivity",
                 mono_resp_method = "none")

# default nbootstrap is 100
# set nboostrap as 10 here to improve example speed
myROCband <- sscaROC_CB(diseaseData,
                        controlData,
                        userFormula,
                        mono_resp_method = "none",
                        target_covariates,
                        global_ROC_controlled_by = "sensitivity",
                        CB_alpha = 0.95,
                        logit_CB = FALSE,
                        nbootstrap = 10,
                        nbin = 100,
                        verbose = FALSE)

plot_sscaROC(myROC, lwd = 1.6)
plot_sscaROC_CB(myROCband, col = "purple", lty = 2)
```

---

sscaROC *Covariate-adjusted continuous biomarker evaluations for specific population.*

Description

Provides evalution for continuous biomarkers at controlled sensitivity/specificity level, or ROC curve in specified sub-population.

Usage

```
sscaROC(diseaseData, controlData, userFormula, target_covariates,
control_sensitivity = NULL, control_specificity = NULL, mono_resp_method = "ROC",
whichSE = "sample", global_ROC_controlled_by = "sensitivity", nbootstrap = 100,
CI_alpha = 0.95, logit_CI = TRUE, verbose = TRUE)
```

Arguments

diseaseData	Data from patients including dependent (biomarker) and independent (covariates) variables.

`controlData`	Data from controls including dependent (biomarker) and independent (covariates) variables.
`userFormula`	A character string to represent the function for covariate adjustment. For example, let Y denote biomarker, Z1 and Z2 denote two covariates. Then userFormula = "Y ~ Z1 + Z2".
`target_covariates`	Covariates of the interested sub-population. It could be a vector, e.g. c(1, 0.5, 0.8), or a matrix, e.g. target_covariates = matrix(c(1, 0.7, 0.9, 1, 0.8, 0.8), 2, 3, byrow = TRUE)
`control_sensitivity`	The level(s) of sensitivity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).
`control_specificity`	The level(s) of specificity to be controlled at. Could be a scalar (e.g. 0.7) or a numeric vector (e.g. c(0.7, 0.8, 0.9)).
`mono_resp_method`	The method used to restore monotonicity of the ROC curve or computed sensitivity/specificity value. It should one from the following: "none", "ROC". "none" is not applying any monotonicity respecting method. "ROC" is to apply ROC-based monotonicity respecting approach. Default value is "ROC".
`whichSE`	The method used to compute standard error. It should be one from the following: "sample", "bootstrap", meaning to calculate the standard error using sample-based approach or bootstrap. Default is "sample".
`global_ROC_controlled_by`	Whether sensitivity/specificity is used to control when computing global ROC. It should one from the following: "sensitivity", "specificity". Default is "sensitivity".
`nbootstrap`	Number of boostrap iterations. Default is 100.
`CI_alpha`	Percentage of confidence interval. Default is 0.95.
`logit_CI`	Whether to apply logit-based confidence interval. Logit-transformed CI has been identified to be more robust near border area.
`verbose`	Whether to print out messages. Default value is true.

Value

If control_sensitivity or control_specificity is provided, compute covariate-adjusted specificity (sensitivity) at controlled sensitivity (specificity) level.

`Estimate`	Covariate-adjusted sensitivity/specificity.
`SE`	Estimated standard error.
`CI`	Estimated confidence intervals.

If both control_sensitivity and control_specificity are null, compuate covariate-adjusted ROC curve.

`sensitivity`	Estimated sensitivity.
`specificity`	Estimated specificity.
`mono_adj`	Monotonicity adjustment method.

Author(s)

Ziyi.li <zli16@mdanderson.org>

Examples

```
n1 = n0 = 1000
## generate data
Z_D1 <- rbinom(n1, size = 1, prob = 0.3)
Z_D2 <- rnorm(n1, 0.8, 1)
Z_C1 <- rbinom(n0, size = 1, prob = 0.7)
Z_C2 <- rnorm(n0, 0.8, 1)
Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)
M0 <- Y_C_Z0 * (Z_C1 == 0) + Y_C_Z1 * (Z_C1 == 1) + Z_C2
M1 <- Y_D_Z0 * (Z_D1 == 0) + Y_D_Z1 * (Z_D1 == 1) + 1.5 * Z_D2
diseaseData <- data.frame(M = M1, Z1 = Z_D1, Z2 = Z_D2)
controlData <- data.frame(M = M0, Z1 = Z_C1, Z2 = Z_C2)
userFormula = "M~Z1+Z2"
target_covariates = c(1, 0.7, 0.9)
res <- sscaROC(diseaseData,controlData,
               userFormula = userFormula,
               control_sensitivity = c(0.2,0.8, 0.9),
               target_covariates = target_covariates,
               control_specificity = NULL,
               mono_resp_method = "none",
               whichSE = "sample",nbootstrap = 100,
               CI_alpha = 0.95, logit_CI = TRUE)


## bootstrap-based variance estimation
res <- sscaROC(diseaseData,controlData,
               userFormula = userFormula,
               control_sensitivity = c(0.2,0.8, 0.9),
               target_covariates = target_covariates,
               control_specificity = NULL,
               mono_resp_method = "none",
               whichSE = "bootstrap",nbootstrap = 100,
               CI_alpha = 0.95, logit_CI = TRUE)
## monotonization by ROC-based
res <- sscaROC(diseaseData,controlData,
               userFormula = userFormula,
               control_sensitivity = c(0.2,0.8, 0.9),
               target_covariates = target_covariates,
               control_specificity = NULL,
               mono_resp_method = "ROC",
               whichSE = "bootstrap",nbootstrap = 100,
               CI_alpha = 0.95, logit_CI = TRUE)
## control specificity
res <- sscaROC(diseaseData,controlData,
               userFormula = userFormula,
```

```
                control_sensitivity = NULL,
                target_covariates = target_covariates,
                control_specificity = c(0.2,0.8, 0.9),
                mono_resp_method = "ROC",
                whichSE = "bootstrap",nbootstrap = 100,
                CI_alpha = 0.95, logit_CI = TRUE)
### get ROC curves
myROC <- sscaROC(diseaseData,
                 controlData,
                 userFormula,
                 target_covariates,
                 global_ROC_controlled_by = "sensitivity",
                 mono_resp_method = "none")
```

`sscaROC_CB` *Get confidence band for covariate-adjusted ROC curve for specified sub-population.*

Description

Use this function to compute the confidence band for covariate-adjusted ROC curve, with or without monotonicity respecting methods for sub-population.

Usage

```
sscaROC_CB(diseaseData, controlData, userFormula, mono_resp_method = "none",
target_covariates, global_ROC_controlled_by = "sensitivity", CB_alpha = 0.95,
logit_CB = FALSE, nbootstrap = 100, nbin = 100, verbose = FALSE)
```

Arguments

`diseaseData` Data from patients including dependent (biomarker) and independent (covariates) variables.

`controlData` Data from controls including dependent (biomarker) and independent (covariates) variables.

`userFormula` A character string to represent the function for covariate adjustment. For example, let Y denote biomarker, Z1 and Z2 denote two covariates. Then userFormula = "Y ~ Z1 + Z2".

`mono_resp_method`
The method used to restore monotonicity of the ROC curve or computed sensitivity/specificity value. It should one from the following: "none", "ROC". "none" is not applying any monotonicity respecting method. "ROC" is to apply ROC-based monotonicity respecting approach. Default value is "ROC".

`target_covariates`
Covariates of the interested sub-population. It could be a vector, e.g. c(1, 0.5, 0.8), or a matrix, e.g. target_covariates = matrix(c(1, 0.7, 0.9, 1, 0.8, 0.8), 2, 3, byrow = TRUE)

global_ROC_controlled_by
: Whether sensitivity/specificity is used to control when computing global ROC. It should one from the following: "sensitivity", "specificity". Default is "sensitivity".

CB_alpha
: Percentage of confidence band. Default is 0.95.

logit_CB
: Whether to use logit-transformed (then transform back) confidence band. Default is FALSE.

nbootstrap
: Number of boostrap iterations. Default is 100.

nbin
: Number of bins used for constructing confidence band. Default is 100.

verbose
: Whether to print out messages during bootstrap. Default value is FALSE.

Value

If global ROC is controlled by sensitivity, a list will be output including the following

Sensitivity
: Vector of sensitivities;

Specificity_upper
: Upper confidence band for specificity estimations;

Specificity_lower
: Lower confidence band for specificity estimations;

global_ROC_controlled_by
: "sensitivity".

If global ROC is controlled by Specificity, a list will be output including the following

Specificity
: Vector of specificity;

Sensitivity_upper
: Upper confidence band for sensitivity estimations;

Sensitivity_lower
: Lower confidence band for sensitivity estimations;

global_ROC_controlled_by
: "specificity".

Author(s)

Ziyi.li <zli16@mdanderson.org>

Examples

```
n1 = n0 = 500

## generate data
Z_D1 <- rbinom(n1, size = 1, prob = 0.3)
Z_D2 <- rnorm(n1, 0.8, 1)
Z_C1 <- rbinom(n0, size = 1, prob = 0.7)
Z_C2 <- rnorm(n0, 0.8, 1)
Y_C_Z0 <- rnorm(n0, 0.1, 1)
Y_D_Z0 <- rnorm(n1, 1.1, 1)
```

```
Y_C_Z1 <- rnorm(n0, 0.2, 1)
Y_D_Z1 <- rnorm(n1, 0.9, 1)

M0 <- Y_C_Z0 * (Z_C1 == 0) + Y_C_Z1 * (Z_C1 == 1) + Z_C2
M1 <- Y_D_Z0 * (Z_D1 == 0) + Y_D_Z1 * (Z_D1 == 1) + 1.5 * Z_D2

diseaseData <- data.frame(M = M1, Z1 = Z_D1, Z2 = Z_D2)
controlData <- data.frame(M = M0, Z1 = Z_C1, Z2 = Z_C2)

userFormula = "M~Z1+Z2"
target_covariates = c(1, 0.7, 0.9)

# default nbootstrap is 100
# set nboostrap as 10 here to improve example speed

myROCband <- sscaROC_CB(diseaseData,
                       controlData,
                       userFormula,
                       mono_resp_method = "none",
                       target_covariates,
                       global_ROC_controlled_by = "sensitivity",
                       CB_alpha = 0.95,
                       logit_CB = FALSE,
                       nbootstrap = 10,
                       nbin = 100,
                       verbose = FALSE)
```

Index