Package ‘ROCit’

January 31, 2019

Type Package

Title Performance Assessment of Binary Classifier with Visualization

Version 1.1.1

Date 2019-01-21

Description Sensitivity (or recall or true positive rate), false positive rate, specificity, precision (or positive predictive value), negative predictive value, misclassification rate, accuracy, F-score—these are popular metrics for assessing performance of binary classifier for certain threshold. These metrics are calculated at certain threshold values. Receiver operating characteristic (ROC) curve is a common tool for assessing overall diagnostic ability of the binary classifier. Unlike depending on a certain threshold, area under ROC curve (also known as AUC), is a summary statistic about how well a binary classifier performs overall for the classification task. ROCit package provides flexibility to easily evaluate threshold-bound metrics. Also, ROC curve, along with AUC, can be obtained using different methods, such as empirical, binormal and non-parametric. ROCit encompasses a wide variety of methods for constructing confidence interval of ROC curve and AUC. ROCit also features the option of constructing empirical gains table, which is a handy tool for direct marketing. The package offers options for commonly used visualization, such as, ROC curve, KS plot, lift plot. Along with in-built default graphics setting, there are rooms for manual tweak by providing the necessary values as function arguments. ROCit is a powerful tool offering a range of things, yet it is very easy to use.

Imports stats, graphics

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Md Riaz Ahmed Khan [aut, cre],
Thomas Brandenburger [aut]

Maintainer Md Riaz Ahmed Khan <mdriazahmed.khan@jacks.sdstate.edu>

Repository CRAN

Date/Publication 2019-01-30 23:23:35 UTC
**R topics documented:**

cartesian_2D .................................................. 2
ciAUC .............................................................. 3
ciAUC.rocit ....................................................... 4
ciROC .............................................................. 5
ciROC.rocit ....................................................... 5
ciROCbin ........................................................... 7
ciROCemp .......................................................... 8
convertclass ...................................................... 9
Diabetes ............................................................. 10
gainstable .......................................................... 12
gainstable.default ............................................... 12
gainstable.rocit .................................................. 14
getsurvival ........................................................ 15
getptnfpfn ......................................................... 16
invlogit ............................................................ 17
ksplot .............................................................. 17
ksplot.rocit ...................................................... 18
Loan ................................................................. 19
logit ............................................................... 20
measureit ........................................................ 21
measureit.default ............................................... 21
measureit.rocit .................................................. 23
MLestimates ...................................................... 24
plot.gainstable .................................................. 25
plot.rocci ......................................................... 26
plot.rocit ........................................................ 28
print.gainstable .................................................. 29
print.measureit ................................................... 30
print.rocitaucci .................................................. 31
rankorderedata ................................................... 31
rocit ............................................................. 32
summary.rocit .................................................... 34
trapezoidarea ..................................................... 35

---

**Index** 37

cartesian_2D  

*Cartesian Product of Two Vectors*

---

**Description**

Function *cartesian_2D* takes two vectors as input and returns the two dimensional cartesian product.
Usage

cartesian_2D(array_x, array_y)

Arguments

array_x  A vector, indicating the first set.
array_y  A vector, indicating the second set.

Value

A matrix of length(array_x) * length(array_y) rows and two columns. Each row indicates an ordered pair.

Comment

cartesian_2D is used internally in other function(s) of ROCit. Works if matrix/data frames are passed as arguments. However, returns might not be valid if arguments are not one dimensional.

Examples

x <- seq(3)
y <- c(10,20,30)
cartesian_2D(x,y)

---

ciAUC  Confidence Interval of AUC

Description

See ciAUC.rocit.

Usage

ciAUC(object, ...)

Arguments

object  An object of class "rocit", returned by rocit.
...  Arguments to be passed to methods. See ciAUC.rocit.
ciAUC.rocit

Confidence Interval of AUC

Description

ciAUC constructs confidence interval of area under curve (AUC) of receiver operating characteristic (ROC) curve. This is an S3 method defined for object of class "rocit".

Usage

```r
## S3 method for class 'rocit'
ciAUC(object, level = 0.95, delong = FALSE,
      logit = FALSE, nboot = NULL, ...) = NULL)
```

Arguments

- `object` An object of class "rocit", returned by `rocit`.
- `level` Level of confidence, must be within the range (0 1). Default is 0.95.
- `delong` Logical; indicates whether DeLong formula should be used to estimate the variance of AUC. Default is FALSE.
- `logit` Logical; indicates whether confidence interval of logit transformed AUC should be evaluated first. Default is FALSE.
- `nboot` Number of bootstrap samples, if bootstrap method is desired. Default is NULL. If a numeric value is specified, overrides logit and delong arguments.
- `...` NULL. Used for S3 generic/method consistency.

Value

An object of class "rocitauci".

See Also

- `rocit`, `ciROC.rocit`

Examples

```r
data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
                      data = Diabetes,family = "binomial")

score <- logistic.model$fitted.values
class <- logistic.model$y

# Make the rocit objects
rocit_bin <- rocit(score = score, class = class, method = "bin")
# Confidence interval of AUC
ciAUC(rocit_bin, level = 0.9)
ciAUC(rocit_bin, delong = TRUE, logit = TRUE)
```
ciROC

Confidence Interval of ROC curve

Description

See `ciROC rocit`.

Usage

`ciROC(object, ...)`

Arguments

- **object**: An object of class "rocit", returned by `rocit`. Supports "empirical" and "binormal" ROC curve.
- **...**: Arguments to be passed to methods. See `ciROC rocit`.

---

ciROC rocit

Confidence Interval of ROC curve

Description

ciROC constructs confidence interval of receiver operating characteristic (ROC) curve. This is an S3 method defined for object of class "rocit".

Usage

```r
## S3 method for class 'rocit'
ciROC(object, level = 0.95, nboot = 500, ... = NULL)
```

Arguments

- **object**: An object of class "rocit", returned by `rocit`. Supports "empirical" and "binormal" ROC curve.
- **level**: Level of confidence, must be within the range (0 1). Default is 0.95.
- **nboot**: Number of bootstrap samples, used to estimate `var(A), var(B), cov(A,B)`. Only used for method = "binomial". See 'Details'.
- **...**: NULL. Used for S3 generic/method consistency.
Details

For large values of $n_Y$ and $n_{\bar{Y}}$, the distribution of $TPR(c)$ at $FPR(c)$ can be approximated as a normal distribution with following mean and variance:

$$
\mu_{TPR(c)} = \frac{n_Y}{n_Y} \sum_{i=1}^{n_Y} I(D_{Y_i} \geq c) / n_Y
$$

$$
V(TPR(c)) = \frac{TPR(c)(1 - TPR(c))}{n_Y} + \frac{\left(\frac{g(c^*)}{f(c^*)}\right)^2}{n_{\bar{Y}}} K
$$

where $K = \frac{FPR(c)(1 - FPR(c))}{n_Y}$, $g$ and $f$ are the probability distribution functions of the diagnostic variable in positive and negative groups (with corresponding cumulative distribution functions $G$ and $F$), $c^* = S_{D_X}^{-1}(FPR(c))$, and $S$ is the survival function given by: $S(t) = P(T > t) = 1 - F(t)$. density and approxfun were used to approximate PDF and CDF of the diagnostic score in the two groups and the inverse survival of the diagnostic in the negative responses.

For "binomial" type, variance of $A + BZ_x$ is given by $V(A) + Z_x^2V(B) + 2Z_xCov(A, B)$. Bootstrap method was used to estimate $V(A), V(B)$ and $Cov(A, B)$. The lower and upper limit of $A + BZ_x$ are inverse probit transformed to obtain the confidence interval of the ROC curve.

Value

A list of class "rocci", having following elements:

- **'roc estimation method'**
  The method applied to estimate ROC curve in the rocit object.

- **'confidence level'**
  Level of confidence as supplied as argument.

- **FPR**
  An array containing all the FPR values, for which TPR and confidence interval of TPR were estimated.

- **TPR**
  Array containing the TPR values associated with the FPR values.

- **LowerTPR**
  Lower limits of the TPR values. Forced to zero for type = "empirical", where empirical TPR is zero.

- **UpperTPR**
  Upper limits of the TPR values. Forced to one for type = "empirical", where empirical TPR is one.

References


See Also

plot.rocci, rocit, ciAUC.rocit
ciROCBin

Examples

```r
data("Loan")
score <- Loan$Score
class <- ifelse(Loan$Status == "CO", 1, 0)
rocit_emp <- rocit(score = score, class = class, method = "emp")
# -----------------------------------------------
ciROC_emp90 <- ciROC(rocit_emp, level = 0.9)
plot(ciROC_emp90, egend = TRUE)
```

---

**ciROCBin**

**Confidence Interval of Binormal ROC Curve**

**Description**

Function ciROCBin estimates confidence interval of binormally estimated ROC curve.

**Usage**

```r
ciROCBin(rocit_bin, level, nboot)
```

**Arguments**

- `rocit_bin` An object of class rocit, (method = "binormal").
- `level` Desired level of confidence to be estimated.
- `nboot` Number of bootstrap samples, used to estimate var(A), var(B), cov(A,B). See `ciROC rocit`.

**Value**

A list object containing TPR, upper and lower bound of TPR at certain FPR values.

**Comment**

ciROCBin is used internally in `ciROC rocit` of ROCit.

**See Also**

rocit, ciROC, plot.rocci
Examples

data("Loan")
score <- Loan$Score
class <- ifelse(Loan$Status == "CO", 1, 0)
rocit_bin <- rocit(score = score, class = class, method = "bin")
ciROC_bin90 <- ciROCbin(rocit_bin, level = 0.9, nboot = 300)
TPR <- ciROC_bin90$TPR
FPR <- ciROC_bin90$FPR
Upper90 <- ciROC_bin90$UpperTPR
Lower90 <- ciROC_bin90$LowerTPR
plot(TPR~FPR, type = "l")
lines(Upper90~FPR, lty = 2)
lines(Lower90~FPR, lty = 2)
ggrid()
legend("bottomright", c("Binormal ROC curve", "90% CI"), lty = c(1,2))

---

**ciROCemp**  
*Confidence Interval of Empirical ROC Curve*

Description

Function ciROCemp estimates confidence interval of empirically estimated ROC curve.

Usage

`ciROCemp(rocit_emp, level)`

Arguments

- `rocit_emp` An object of class `rocit`, (method = "empirical").
- `level` Desired level of confidence to be estimated.

Value

A list object containing TPR, upper and lower bound of TPR at certain FPR values.

Comment

`ciROCemp` is used internally in `ciROC_rocit` of `ROCit`.

See Also

`rocit.ciROC, plot.rocci`
**convertclass**

**Examples**

```r
cat <- c("cat", "dog", "cat")
convertclass(cat) # by default, "cat" is converted to 0
convertclass(cat, reference = "dog")
```

```r
set.seed(100)
x <- round(runif(10, 2, 3))
convertclass(x, reference = 3)
# numeric reference can be supplied as character
convertclass(x, reference = "3") # same result
```

**Description**

`convertclass` converts a binary variable with any response into 1/0 response. It is used internally in other functions of package `ROCit`.

**Usage**

```r
convertclass(x, reference = NULL)
```

**Arguments**

- `x`: A vector of exactly two unique values.
- `reference`: The reference value. Depending on the class of `x`, it can be numeric or character type. If specified, this value is converted to 0 and other is converted to 1. If `NULL`, reference is set alphabetically.

**Value**

A numeric vector of 1 and 0.

**Comment**

`convertclass` is used internally in other function(s) of `ROCit`.
Description

These data are courtesy of Dr. John Schorling, Department of Medicine, University of Virginia School of Medicine.

The data contains information on 403 subjects from 1046 subjects who were interviewed in a study to understand the prevalence of obesity, diabetes, and other cardiovascular risk factors in central Virginia for African Americans. According to Dr. John Hong, Diabetes Mellitus Type II (adult onset diabetes) is associated most strongly with obesity. The waist/hip ratio may be a predictor in diabetes and heart disease. DM II is also associated with hypertension - they may both be part of "Syndrome X". The 403 subjects were the ones who were actually screened for diabetes. Glycosylated hemoglobin > 7.0 is usually taken as a positive diagnosis of diabetes.

Usage

Diabetes

Format

A data frame with 403 rows and 22 variables (See "Note"):

- id  Subject id
- chol  Total cholesterol
- stab.glu  Stabilized glucose
- hdl  High density lipoprotein
- ratio  Cholesterol/hdl ratio
- glyhb  Glycosylated hemoglobin
- location  A factor with levels Buckingham and Louisa
- age  Age (years)
- gender  Gender, male or female
- height  Height (inches)
- weight  Weight (pounds)
- frame  A factor with levels small, medium and large
- bp.1s  First systolic blood pressure
- bp.1d  First diastolic blood pressure
- bp.2s  Second systolic blood pressure
- bp.2d  Second diastolic blood pressure
- waist  Waist (inches)
- hip  Hip (inches)
time.ppn  Postprandial time when labs were drawn in minutes

bmi  Body mass index

dtest  An indicator whether glyhb is greater than 7 or not

whr  Waist to hip ratio

Note

The last three variables (bmi, dtest, whr) were created. For bmi, following formula was used:

\[ bmi = 703 \times \left(\frac{\text{weight (lbs)}}{\text{height (inches)}}\right)^2 \]

Source

staff.pubhealth.ku.dk/~tag/Teaching/share/data/Diabetes.html#sec-2

References


Examples

data("Diabetes")
plot(Diabetes$hd1-Diabetes$weight, pch = 16,
    col = ifelse(Diabetes$gender=="male",1,2))

### density plot
femaleBMI <- density(subset(Diabetes, gender == "female")$bmi, na.rm = TRUE)

maleBMI <- density(subset(Diabetes, gender == "male")$bmi, na.rm = TRUE)

plot(NULL, ylim = c(0,0.08), xlim = c(10,60),
    xlab = "BMI", ylab = "Density", main = "")
grid(col = 1)
polygon(maleBMI, col = rgb(0,0,1,0.2), border = 4)
polygon(femaleBMI, col = rgb(1,0,0,0.2), border = 2)
abline(h = 0)
legend("topright", c("Male", "Female"), pch = 15,
    col = c(rgb(0,0,1,0.2), rgb(1,0,0,0.2)), bty = "n")

# logistic model
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
    data = Diabetes,family = "binomial")
summary(logistic.model)

class <- logistic.model$y
score <- logistic.model$fitted.values
rocit.object <- rocit(score = score, class = class)
gainstable

### Description

Default S3 method to create gains table from a vector of diagnostic score and the class of observations.

### Usage

```r
# Default S3 method:
gainstable(score, class, negref = NULL, ngroup = 10,
            breaks = NULL, ... = NULL)
```

### Arguments

- **score**: An numeric array of diagnostic score. Same as in `rocit`.
- **class**: An array of equal length of score, containing the class of the observations. Same as in `rocit`.
- **negref**: The reference value, same as the reference in `convertclass`. Depending on the class of x, it can be numeric or character type. If specified, this value is converted to 0 and other is converted to 1. If NULL, reference is set alphabetically. Same as in `rocit`.
- **ngroup**: Number of desired groups in gains table. Ignored if `breaks` is specified. See "Details".
- **breaks**: Percentiles (in percentage) at which observations should be separated to form groups. If specified, `ngroup` is ignored. See "Details".
- **...**: NULL. Used for S3 generic/method consistency.

---

gainstable.default

### Description

Gains Table for Binary Classifier

### Usage

```r
summary(rocit_object)
plot(rocit_object)
```
The `gainstable` function creates a gains table containing `ngroup` number of groups or buckets. The algorithm first orders the score variable with respect to the score variable. In case of a tie, it class becomes the ordering variable, keeping the positive responses first. The algorithm calculates the ending index in each bucket as 
\[ \text{round}\left(\frac{\text{length}(\text{score})}{\text{ngroup}} \times (1 : \text{ngroup})\right) \]. Each bucket should have at least 5 observations.

If buckets’ end index are to be ended at desired level of population, then `breaks` should be specified. If specified, it overrides `ngroup` and `ngroup` is ignored. `breaks` by default always includes 100. If a whole number does not exist at specified population, nearest integers are considered.

The function returns a list of class "gainstable". It has the following components:

- **Bucket**: The serial number of buckets or groups.
- **Obs**: Number of observation in the group.
- **CObs**: Cumulative number of observations up to the group.
- **Depth**: Cumulative population depth up to the group.
- **Resp**: Number of (positive) responses in the group.
- **CResp**: Cumulative number of (positive) responses up to the group.
- **RespRate**: (Positive) response rate in the group.
- **CRespRate**: Cumulative (positive) response rate up to the group.
- **CCapRate**: Cumulative overall capture rate of (positive) responses up to the group.
- **Lift**: Lift index in the group. Calculated as \( \frac{\text{GroupResponseRate}}{\text{OverallResponseRate}} \).
- **CLift**: Cumulative lift index up to the group.

The algorithm is designed for complete cases. If NA(s) found in either `score` or `class`, they are removed.

**See Also**

- `gainstable rocit`, `plot.gainstable rocit`

**Examples**

```r
data("Loan")
class <- Loan$Status
score <- Loan$Score
# ------------------------------------------
gtable15 <- gainstable(score = score, class = class,
                        negref = "FP", ngroup = 15)
gtable_custom <- gainstable(score = score, class = class,
                            negref = "FP", breaks = seq(1,100,15))
# ------------------------------------------```
gainstable.rocit

Description

S3 method to create gains table from object of class "rocit".

Usage

```r
 gainstable(x, ngroup = 10, breaks = NULL, ... = NULL)
```

Arguments

- `x` A "rocit" object, created with `rocit`.
- `ngroup` Number of desired groups in gains table. See `gainstable.default`.
- `breaks` Percentiles (in percentage) at which observations should be separated to form groups. See `gainstable.default`.
- `...` NULL. Used for S3 generic/method consistency.

Details

gainstable.rocit calls `gainstable.default`. It creates the score and class variables from the supplied "rocit" object internally. See `gainstable.default` for details.

Value

A list of class "gainstable", same as returned by `gainstable.default`.

See Also

`gainstable.default`, `plot.gainstable`, `rocit`
getSurvival

Examples

data("Loan")
class <- Loan$Status
score <- Loan$Score
rocit_emp <- rocit(score = score, class = class, negref = "FP")
# -------------------------------------------------------------
gtable15 <- gtablestable(rocit_emp, ngroup = 15)
gtable_custom <- gtablestable(rocit_emp, breaks = seq(1, 100, 15))
print(gtable15)
print(gtable_custom)
# -------------------------------------------------------------
plot(gtable15)
plot(gtable15)
plot(gtable_custom, type = 2)
plot(gtable_custom, type = 3)

<table>
<thead>
<tr>
<th>getSurvival</th>
<th>Survival Probability</th>
</tr>
</thead>
</table>

Description

Function getSurvival calculates survival probability from an object of class "density" at specified value.

Usage

getSurvival(x, cutoff)

Arguments

x An object of class "density".
cutoff Value at which survival probability will be calculated.

Details

The survival function S, of a random variable X is defined by,

\[ S(X = x) = 1 - F(X = x) \]

where \( F \) is the cumulative density function (CDF) of \( X \).

Value

Survival probability.

Comment

getSurvival is used internally in other function(s) of ROCit.
Examples

data("Loan")
k <- density(Loan$Income)
  # What portion have income over 100,000
getsurvival(k,100000)

Description

Function `gettptnfpfn` calculates the number of true positive (TP), false positive (FP), true negative (TN) and false negative (FN), if certain depth of the observations are treated as positive. Generally it applies to the observations that are rank-ordered with respect to some diagnostic score.

Usage

`gettptnfpfn(x, depth, npos = sum(x), nneg = length(x) - npos)`

Arguments

- `x` A response vector, consisting of only 1's and 0's.
- `depth` Depth, up to which the positive predictions are made.
- `npos` Number of positive responses.
- `nneg` Number of negative responses.

Details

Function `gettptnfpfn` calculates the number of TP, FP, TN, FN at specified depth. This is designed for numeric 1/0 coding of the responses. Usually it applies to the observations, which are rank-ordered with respect to some diagnostic score.

Value

A numeric vector of length 4 indicating number of TP, FP, TN, FN, respectively.

Comment

`gettptnfpfn` is used internally in other function(s) of `ROCit`. Invalid results are produced for `depth = 0`. 
invlogit

Examples

```r
k <- c(1,1,0,1,0,0,1,0,0,1,0,0,0,0,0)
# get TP, FP, TN, FN if first 6 are predicted to be positives
getptnfpfn(k, 6)
```

invlogit  
*Logistic Transformation*

Description

invlogit returns the logistic transformed value.

Usage

```r
invlogit(x)
```

Arguments

- `x` A number or numeric vector from \((-\infty, \infty)\).

Value

Logistic transformed value of `x`, \(1/(1 + e^{-x})\)

Examples

```r
invlogit(0.5)

invlogit(c(-Inf, Inf))

set.seed(10)
invlogit(runif(10, -3, 3))
```

ksplot  
*KS Plot*

Description

See `ksplot.rocit`.

Usage

```r
ksplot(object, ...)
```
Arguments

object An object of class "rocit", returned by rocit function.
...
Arguments to be passed to methods. See ksplot.rocit.

ksplot.rocit  KS Plot

Description

Generates cumulative density of diagnostic variable in positive and negative responses.

Usage

## S3 method for class 'rocit'
ksplot(object, col = c("#26484F", "#BEEine", "#FFA54F"),
        lty = c(1, 1, 1), legend = T, legendpos = "bottomright",
        values = T, ... = NULL)

Arguments

object An object of class "rocit", returned by rocit function.
col Colors to be used for plot. Minimum three colors need to be supplied for F(c), G(c) and KS Stat mark.
lty Line types of the plots.
legend A logical value indicating whether legends to appear in the plot.
legendpos Position of the legend. A single keyword from "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center", as in legend. Ignored if legend is FALSE.
values A logical value, indicating whether values to be returned.
...
NULL. Used for S3 generic/method consistency.

Details

This function plots the cumulative density functions $F(c)$ and $G(c)$ of the diagnostic variable in the negative and positive populations. If the positive population have higher value then negative curve ($F(c)$) ramps up quickly. The KS statistic is the maximum difference of $F(c)$ and $G(c)$.

Value

If values = TRUE, then Cutoff, F(c), G(c), KS stat, KS Cutoff are returned silently.

Note

Customized plots can be made by using the returned values of the function.
Examples

data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
  data = Diabetes, family = "binomial")
class <- logistic.model$y
score <- logit(logistic.model$fitted.values)
# -----------------------------------------------
roc_emp <- roc(score = score, class = class) # default method empirical
# -----------------------------------------------
kplot1 <- ksplot(roc_emp)
message("KS Stat (empirical) : ", kplot1$'KS stat')
message("KS Stat (empirical) cutoff : ", kplot1$'KS Cutoff')

Loan

Loan Data

Description

A data containing information about 900 borrowers. It is a modified version of publicly available real data.

Usage

Loan

Format

A data frame with 900 rows and 9 variables:

Amount  Amount of loan, shown as percentage of a certain amount.
Term  The number of payments on the loan. Values are in months.
IntRate  Interest rate.
ILR  Ratio of installment amount and total loan amount.
EmpLen  Employment length, categorized.
  • A: 0-2 years
  • B: 3-5 years
  • C: 7-8 years
  • D: 8+ years
  • U: Unknown
Home  Status of home ownership.
Income  Annual income.
Status  A factor indicating whether the loan was fully paid (FP) or charged off (C0) after full term.
Score  A risk score calculated from loan amount, interest rate and annual income. The log-odds of logistic regression were transformed into scores using PDO = 30, OddsBase = 20 and ScoreBase = 400. See "References".
logit

Description

logit transforms probability value(s) into log-odds.

Usage

logit(x)

Arguments

x A number or numeric vector from [01].

Value

Log-odds of x, \(\log(x/(1-x))\).

Examples

logit(0.2)

set.seed(1)
logit(runif(10, 0, 1))
**Description**

See `measureit.default`, `measureit.rocit`

**Usage**

`measureit(...)`

**Arguments**

- `score` An numeric array of diagnostic score.
- `class` An array of equal length of score, containing the class of the observations.
- `negref` The reference value, same as the reference in `convertclass`. Depending on the class of x, it can be numeric or character type. If specified, this value is converted to 0 and other is converted to 1. If NULL, reference is set alphabetically.
- `measure` The performance metrics to be evaluated. See "Details" for available options.
- `...` NULL. Used for S3 generic/method consistency.
Details

Various performance metrics for binary classifier are available that are cutoff specific. For a certain
cutoff value, all the observations having score equal or greater are predicted as positive. Following
metrics can be called for via `measure` argument:

- **ACC**: Overall accuracy of classification = \(P(Y = \hat{Y}) = \frac{TP + TN}{TP + FP + TN + FN}\)
- **MIS**: Misclassification rate = \(1 - ACC\)
- **SENS**: Sensitivity = \(P(\hat{Y} = 1|Y = 1) = \frac{TP}{TP + FN}\)
- **SPEC**: Specificity = \(P(\hat{Y} = 0|Y = 0) = \frac{TN}{TN + FP}\)
- **PREC**: Precision = \(P(Y = 1|\hat{Y} = 1) = \frac{TP}{TP + FP}\)
- **REC**: Recall. Same as sensitivity.
- **PPV**: Positive predictive value. Same as precision
- **NPV**: Positive predictive value = \(P(Y = 0|\hat{Y} = 0) = \frac{TN}{TN + FN}\)
- **TPR**: True positive rate. Same as sensitivity.
- **FPR**: False positive rate. Same as \(1 - specificity\).
- **TNR**: True negative rate. Same as specificity.
- **FNR**: False negative rate = \(P(\hat{Y} = 0|Y = 1) = \frac{FN}{FN + TP}\)
- **pDLR**: Positive diagnostic likelihood ratio = \(\frac{TPR}{FPR}\)
- **nDLR**: Negative diagnostic likelihood ratio = \(\frac{FNR}{TNR}\)
- **FSCR**: F-score, defined as \(2 \times \frac{PPV \times TPR}{PPV + TPR}\)

*Exact match* is required. If the values passed in the `measure` argument do not match with the
available options, then ignored.

Value

An object of class "`measureit`". By default it contains the followings:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutoff</td>
<td>Cutoff at which metrics are evaluated.</td>
</tr>
<tr>
<td>Depth</td>
<td>What portion of the observations fall on above the cutoff.</td>
</tr>
<tr>
<td>TP</td>
<td>Number of true positives, when the observations having score equal or greater than cutoff are predicted positive.</td>
</tr>
<tr>
<td>FP</td>
<td>Number of false positives, when the observations having score equal or greater than cutoff are predicted positive.</td>
</tr>
<tr>
<td>TN</td>
<td>Number of true negatives, when the observations having score equal or greater than cutoff are predicted positive.</td>
</tr>
<tr>
<td>FN</td>
<td>Number of false negatives, when the observations having score equal or greater than cutoff are predicted positive.</td>
</tr>
</tbody>
</table>

When other metrics are called via `measure`, those also appear in the return in the order they are
listed above.
Note

The algorithm is designed for complete cases. If NA(s) found in either score or class, then removed.

Internally sorting is performed, with respect to the score. In case of tie, sorting is done with respect to class.

Author(s)

Riaz Khan, <mdriazahmed.khan@jacks.sdstate.edu>

See Also

measureit.rocit, print.measureit

Examples

data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
  data = Diabetes,family = "binomial")
class <- logistic.model$y
score <- logistic.model$fitted.values
# ---------------------------------------------------------------
measure <- measureit(score = score, class = class,
  measure = c("ACC", "SENS", "FSCR"))
names(measure)
plot(measure$ACC~measure$Cutoff, type = "l")
plot(measure$TP~measure$FP, type = "l")

---

**measureit.rocit**  
**Performance Metrics of Binary Classifier**

Description

This is an S3 method for object of class "rocit". It computes various performance metrics at different cutoff values.

Usage

```
## S3 method for class 'rocit'
measureit(x, measure = c("ACC", "SENS"), ... = NULL)
```

Arguments

- `x`  
  An object of class "rocit" created with `rocit`.
- `measure`  
  The performance metrics to be evaluated. See "Details" for available options.
- `...`  
  NULL. Used for S3 generic/method consistency.
Details

This function calls `measureit.default`. From the components of "rocit" objects, it calculates the score and class variables internally. See `measureit.default` for other details and available options for measure argument.

Value

An object of class "measureit", same as returned by `measureit.default`.

Note

See `measureit.default`.

See Also

`measureit.default, print.measureit`

Examples

```r
data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
    data = Diabetes,family = "binomial")
class <- logistic.model$y
score <- logistic.model$fitted.values
rocit_object <- rocit(score = score, class = class)
# --------------------------------------------------------------
measure <- measureit(rocit_object, measure = c("ACC", "SENS", "FSCR"))
names(measure)
plot(measure$ACC~measure$Cutoff, type = "l")
plot(measure$TP~measure$FP, type = "l")
```

---

**MLEstimates**

**ML Estimate of Normal Parameters**

Description

The function calculates the maximum likelihood (ML) estimates of the two parameters $\mu$ and $\sigma$, when a set of numbers are assumed to be normally distributed.

Usage

`MLEstimates(x)`

Arguments

- `x` A numeric vector.
Details

If a set of observations are assumed to be normally distributed, two parameters, mean \( \mu \) and the variance (the square of \( \sigma \)) are to be estimated. In theory, the ML estimate of \( \mu \) is the mean of the observations. And the ML estimate of square of \( \sigma \) is the mean squared deviation of the observations from the estimated \( \mu \).

Value

A "list" object of two numeric components, \( \mu \) and \( \sigma \).

Comment

MLestimates is used internally in other function(s) of ROCit.

Examples

```r
# Find the two parameters
set.seed(10)
points <- rnorm(200, 10, 5)
ML <- MLestimates(points)
message("The ML estimates are: mean = ", round(ML$mu, 3), "", SD = ", round(ML$sigma, 3))

# Superimpose smooth curve over histogram
set.seed(100)
x <- rnorm(400)
hist(x, probability = TRUE, col = "gray90")
ML <- MLestimates(x)
x <- seq(-3, 3, 0.01)
density <- dnorm(x, mean = ML$mu, sd = ML$sigma)
lines(density~x, lwd = 2)
```

---

**plot.gainstable**

**Plot "gainstable" Object**

Description

An S3 method to make different plots using entries of gains table.

Usage

```r
## S3 method for class 'gainstable'
plot(x, y = NULL, type = 1, col = c("#BEBE9E", "#26484F", "#884580"), legend = TRUE, ...) = NULL)
```
Arguments

x  An object of class "gainstable", created with the function \texttt{gainstable}.
y  NULL.
type  Plot type. See "Details".
col  Colors to be used for plot.
legend  A logical value indicating whether legend to appear. See "Details"

Details

Currently three types are available. type = 1 shows lift and cumulative lift against population depth. type = 2 shows response rate and cumulative response rate against population depth. type = 3 shows cumulative capture rate of positive responses against population depth. For type 1 and 2, three colors and for 3, two colors are required. If more than required specified, then first 3 (for type 1, 2) or 2 (for type 3) colors are used. If less than required specified, then specified colors are repeated. If legend is TRUE, then legend appears in the plot. For type 1 and 2, legend position is "topright", for 3, "bottomright".

See Also

\texttt{gainstable, rocit}

Examples

\begin{verbatim}
data("Loan")
class <- Loan$Status
score <- Loan$Score
rocit_emp <- rocit(score = score, class = class, negref = "FP")
# ---------------------------------------------------------------
gtable <- gainstable(rocit_emp)
# ---------------------------------------------------------------
plot(gtable)
plot(gtable, legend = FALSE)
plot(gtable, col = 2:4)
plot(gtable, type = 2, col = 2:4)
plot(gtable, type = 3, col = 2:3)
\end{verbatim}

\hrulefill

\textbf{plot.rocci}  \hspace{1cm}  \textit{Plot ROC Curve with confidence limits}

Description

This function plots receiver operating characteristic (ROC) curve with confidence limits. This is an S3 method for object of class "rocci", returned by \texttt{ciROC.rocit} function.
plot.rocci

Usage

### S3 method for class 'rocci'
plot(x, col = c("#2F4F4F", "#404040"), lty = c(1, 2),
     lwd = c(2, 1), grid = TRUE, legend = TRUE,
     legendpos = "bottomright", ... = NULL)

Arguments

- **x**: An object of class "rocci", returned by `ciROC_rocit` function.
- **col**: Color(s) to be used for the plot. First two colors are used for the ROC curve and confidence limits if multiple colors supplied. Same color is used if single color supplied.
- **lty**: The line type. Same as in `par`. First two or one are used (like `col`) depending on the length of `lty`.
- **lwd**: The line width. Same as in `par`. First two or one are used (like `col`) depending on the length of `lwd`.
- **grid**: Logical, indicating whether to add rectangular grid. Calls `grid` with default settings.
- **legend**: Logical, indicating whether to add legends to the plot.
- **legendpos**: Position of the legend. A single keyword from "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center", as in `legend`. Ignored if `legend` is FALSE.
- **...**: NULL. Used for S3 generic/method consistency.

See Also

`ciROC`, `rocit`, `plot.rocit`

Examples

```r
score <- c(rnorm(300, 30, 15), rnorm(300, 50, 15))
class <- c(rep(0, 300), rep(1, 300))
rocit_object <- rocit(score = score, class = class, method = "bi")
rocci_object <- ciROC(rocit_object)
# -----------------------------
plot(rocci_object)
plot(rocci_object, col = c(2, 4))
plot(rocci_object, col = c(2, 4), legendpos = "bottom", lty = c(1, 3))
```
Description

This function generates receiver operating characteristic (ROC) curve. This is an S3 method for object of class "rocit", returned by `rocit` function.

Usage

```r
## S3 method for class 'rocit'
plot(x, col = c("#2F4F4F", "#BEBEBE"), legend = TRUE,
     legendpos = "bottomright", YIndex = TRUE, values = TRUE,
     ... = NULL)
```

Arguments

- `x`: An object of class "rocit", returned by `rocit` function.
- `col`: Colors to be used in the plot. If multiple specified, the first color is used for the ROC curve, and the second color is used for the chance line \((y = x)\) line, otherwise single color is used.
- `legend`: A logical value indicating whether legends to appear in the plot.
- `YIndex`: A logical value indicating whether optimal Youden Index (i.e where \(|TPR - FPR|\) is maximum) to be marked in the plot.
- `values`: A logical value, indicating whether values to be returned.
- `...`: NULL. Used for S3 generic/method consistency.

Value

If `values = TRUE`, then AUC, Cutoff, TPR, FPR, optimal Youden Index with associated TPR, FPR, Cutoff are returned silently.

Note

Customized plots can be made by using the returned values of the function.

Examples

```r
data("Loan")
score <- Loan$Score
class <- ifelse(Loan$Status == "FP", 0, 1)
rocit_emp <- rocit(score = score, class = class)
# --------------
```
```r
plot(rocit_emp)
plot(rocit_emp, col = c(2,4), legendpos = "bottom",
     YIndex = FALSE, values = FALSE)
# ------------------------
rocit_bin <- rocit(score = score, class = class, method = "bin")
# ------------------------
plot(rocit_emp, col = c(1,"gray50"), legend = FALSE, YIndex = FALSE)
lines(rocit_bin$TPR-rocit_bin$FPR, col = 2, lwd = 2)
legend("bottomright", col = c(1,2),
       c("Empirical ROC", "Binormal ROC"), lwd = 2)
```

**print.gainstable**  
*Print 'gainstable' Object*

### Description
S3 print method to print "gainstable" object.

### Usage
```r
## S3 method for class 'gainstable'
print(x, maxdigit = 3, ... = NULL)
```

### Arguments
- **x**: An object of class "gainstable", created with either `gainstable.default` or `gainstable.rocit`.
- **maxdigit**: How many digits after decimal to be printed.
- **...**: NULL. Used for S3 generic/method consistency.

### Examples
```r
data("Loan")
class <- Loan$Status
score <- Loan$Score
rocit_emp <- rocit(score = score, class = class, negref = "FP")
# ------------------------
gtable8 <- gainstable(rocit_emp, ngroup = 8)
print(gtable8)
print(gtable8, maxdigit = 4)
```
print.measureit

Print 'measureit' Object

Description

S3 method to print object of "measureit" class in organized way.

Usage

## S3 method for class 'measureit'
print(x, n = NULL, ... = NULL)

Arguments

x  An object of class "measureit", created with the function measureit.

n  How many rows of output is desired in the output. If NULL, then prints all the rows. If specified, then first n rows are printed. If specified n is bigger than the number of possible rows, then n is adjusted. If non integer or negative, default (10 or number of possible rows, whichever is smaller) is set. If NULL, all rows printed.

...  NULL. Used for S3 generic/method consistency.

See Also

measureit

Examples

data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
  data = Diabetes,family = "binomial")
class <- logistic.model$y
score <- logistic.model$fitted.values
# ----------------------------------------------
measure <- measureit(score = score, class = class,
measure = c("ACC", "SENS", "FSCR"))
print(measure, n = 5)
print(measure, n = 10)
print.rocitaucci

Print Confidence Interval of AUC

Description

Print Confidence Interval of AUC

Usage

## S3 method for class 'rocitaucci'
print(x, ... = NULL)

Arguments

x

An object of class rocitaucci created with ciAUC.

...  

NULL. Used for S3 generic/method consistency.

Examples

data("Diabetes")
logistic.model <- glm(as.factor(dtest)~chol+age+bmi,
    data = Diabetes,family = "binomial")
score <- logistic.model$fitted.values
class <- logistic.model$y

# Make the rocit objects
rocit_bin <- rocit(score = score, class = class, method = "bin")
obj_1 <- ciAUC(rocit_bin, level = 0.9)
obj_2 <- ciAUC(rocit_bin, delong = TRUE)
obj_3 <- ciAUC(rocit_bin, delong = TRUE, logit = TRUE)
# Print
print(obj_1)
print(obj_2)
print(obj_3)

rankorderdata

Rank order data

Description

Function rankorderdata rank-orders the data with respect to some variable (diagnostic variable).

Usage

rankorderdata(score, class, dec = TRUE)
rocit

**Arguments**

- `score`: A vector containing (diagnostic) scores.
- `class`: A vector containing the class.
- `dec`: Logical. TRUE for descending order, FALSE for ascending order.

**Value**

A dataframe, rank-ordered with respect to the score.

**Comment**

`rankorderdata` is used internally in other function(s) of `ROCit`.

**Author(s)**

Riaz Khan, <mdriazahmed.khan@jacks.sdstate.edu>

**Examples**

```r
score <- c(0.4 * runif(20) + 0.2, 0.4*runif(20))
class <- c(rep("A",20), rep("B",20))
returndata <- rankorderdata(score, class, dec = FALSE)
returndata
```

**Description**

`rocit` is the main function of `ROCit` package. With the diagnostic score and the class of each observation, it calculates true positive rate (sensitivity) and false positive rate (1-Specificity) at convenient cutoff values to construct ROC curve. The function returns "rocit" object, which can be passed as arguments for other S3 methods.

**Usage**

```r
rocit(score, class, negref = NULL, method = "empirical")
```

**Arguments**

- `score`: An numeric array of diagnostic score.
- `class`: An array of equal length of score, containing the class of the observations.
- `negref`: The reference value, same as the reference in `convertclass`. Depending on the class of x, it can be numeric or character type. If specified, this value is converted to 0 and other is converted to 1. If NULL, reference is set alphabetically.
Details

ROC curve is defined as the set of ordered pairs, \((FPR(c), TPR(c))\), where, \(-\infty < c < \infty\), where, \(FPR(c) = P(D \geq c|Y = 0)\) and \(FPR(c) = P(D \geq c|Y = 1)\) at cutoff \(c\). Alternately, it can be defined as:

\[
y(x) = 1 - G[F^{-1}(1 - x)], 0 \leq x \leq 1
\]

where \(F\) and \(G\) are the cumulative density functions of the diagnostic score in negative and positive responses respectively. rocit evaluates TPR and FPR values at convenient cutoffs.

As the name implies, empirical TPR and FPR values are evaluated for method = "empirical". For "binormal", the distribution of diagnostic are assumed to be normal and maximum likelihood parameters are estimated. If method = "nonparametric", then kernel density estimates (using density) are applied with following bandwidth:

\[
\begin{align*}
    h_Y &= 0.9 \times \min(\sigma_Y, IQR(D_Y)/1.34)/((n_Y)^{1/5}) \\
    h_{\bar{Y}} &= 0.9 \times \min(\sigma_{\bar{Y}}, IQR(D_{\bar{Y}})/1.34)/((n_{\bar{Y}})^{1/5})
\end{align*}
\]

as described in Zou et al. From the kernel estimates of PDFs, CDFs are estimated using trapezoidal rule.

For "empirical" ROC, trapezoidal rule is applied to estimate area under curve (AUC). For "binormal", AUC is estimated by \(\Phi\left(A/\sqrt{1 + B^2}\right)\), where \(A\) and \(B\) are functions of mean and variance of the diagnostic in two groups. For "nonparametric", AUC is estimated as by

\[
\frac{1}{n_Y n_{\bar{Y}}} \sum_{i=1}^{n_Y} \sum_{j=1}^{n_{\bar{Y}}} \Phi\left(\frac{D_{Yj} - D_{\bar{Y}i}}{h_Y^2 + h_{\bar{Y}}^2}\right)
\]

Value

A list of class "rocit", having following elements:

- **method**: The method applied to estimate ROC curve.
- **pos_count**: Number of positive responses.
- **neg_count**: Number of negative responses.
- **pos_d**: Array of diagnostic scores in positive responses.
- **neg_d**: Array of diagnostic scores in negative responses.
- **AUC**: Area under curve. See "Details".
- **Cutoff**: Array of cutoff values at which the true positive rates and false positive rates are evaluated. Applicable for "empirical" and "nonparametric".
- **param**: Maximum likelihood estimates of \(\mu\) and \(\sigma\) of the diagnostic score in two groups. Applicable for "binormal".
- **TPR**: Array of true positive rates (or sensitivities or recalls), evaluated at the cutoff values.
- **FPR**: Array of false positive rates (or 1-specificity), evaluated at the cutoff values.

Note

The algorithm is designed for complete cases. If NA(s) found in either score or class, then removed.
References


See Also

ciROC, ciAUC, plot.rocit, gainstable, ksplot

Examples

```r
# ---------------------
data("Diabetes")
roc_empirical <- rocit(score = Diabetes$chol, class = Diabetes$dtest, negref = "-") # default method empirical
roc_binormal <- rocit(score = Diabetes$chol, class = Diabetes$dtest, negref = "-", method = "bin")

# ---------------------
summary(roc_empirical)
summary(roc_binormal)

# ---------------------
plot(roc_empirical)
plot(roc_binormal, col = c("#00BA37", "#F8766D"), legend = FALSE, YIndex = FALSE)
```

---

**summary.rocit**

Summary of rocit object

Description

Prints the summary of rocit object.

Usage

```r
## S3 method for class 'rocit'
summary(object, ... = NULL)
```

Arguments

- **object** An object of class rocit, returned by rocit function.
- **...** NULL. Used for S3 generic/method consistency.
trapezoidarea

Examples

```r
data("Diabetes")
roc_empirical <- rocit(score = Diabetes$chol, class = Diabetes$dtest,
negref = "-")
# ------------------------
summary(roc_empirical)
```

---

trapezoidarea  
**Approximate Area with Trapezoid Rule**

Description

trapezoidarea calculates the approximated area under curve, using trapezoidal rule.

Usage

trapezoidarea(x, y)

Arguments

- **x**, **y**
  Numeric vectors of same length, representing the x and y coordinates of the points.

Details

The function approximates the area bounded by the following 4 curves:

\[ x = a, x = b, y = 0, y = f(x) \]

\( a \) and \( b \) are set at the min and max value of given x coordinates. \((x, y)\) are some points on the \( y = f(x) \) curve.

Value

Numeric value of the area under curve approximated with trapezoid rule.

Comment

trapezoidarea is used internally in other function(s) of ROCit.
**Examples**

# Area under rectangle ---------------

```r
trapezoidarea(seq(0, 10), rep(1, 11))
```

# Area under triangle ---------------

```r
trapezoidarea(seq(0, 10), seq(0, 10))
```

# Area under normal pdf ---------------

```r
x_vals <- seq(-3, 3, 0.01); y_vals <- dnorm(x_vals)
trapezoidarea(x = x_vals, y = y_vals) # theoretically 1
```
Index

*Topic datasets
  Diabetes, 10
  Loan, 19

cartesian_2D, 2
ciAUC, 3, 31, 34
ciAUC.rocit, 3, 4, 6
ciROC, 5, 7, 8, 27, 34
ciROC.rocit, 4, 5, 7, 8, 26, 27
ciROCbin, 7
ciROComp, 8
convertclass, 9, 12, 21, 32
density, 33
Diabetes, 10
gainstable, 12, 26, 34
gainstable.default, 12, 12, 14, 29
gainstable.rocit, 12, 13, 14, 29
getsurvival, 15
gettpnfpfn, 16
grep, 32
grid, 27

invlogit, 17
ksplot, 17, 34
ksplot.rocit, 17, 18, 18

legend, 18, 27, 28
Loan, 19
logit, 20

measureit, 21, 30
measureit.default, 21, 21, 24
measureit.rocit, 21, 23, 23
MEstimates, 24

par, 27
plot.gainstable, 13, 14, 25
plot.rocN, 6–8, 26

plot.rocit, 27, 28, 34
print.gainstable, 29
print.measureit, 23, 24, 30
print.rocitaucci, 31

rankorderdata, 31
rocit, 3–8, 12–14, 18, 23, 26–28, 32

summary.rocit, 34

trapezoidarea, 35