Package 'DET'

May 7, 2026

Type Package

Title Representation of DET Curve with Confidence Intervals

Version 3.0.2

Description Builds both ROC (Receiver Operating Characteristic) and DET (Detection Error Trade-off) curves from a set of predictors, which are the
results of a binary classification system. The curves give a general vision of the performance of the classifier,
and are useful for comparing performance of different systems.

License GPL-2

Encoding UTF-8

Depends R (>= 3.5.0)

Imports pROC, doParallel, parallel, methods

RoxygenNote 7.3.2

NeedsCompilation no

URL https://github.com/jmcurran/DET

BugReports https://github.com/jmcurran/DET/issues

Author García-Ródenas, Álvaro [aut],
Franco, Manuel [aut],
Vivo, Juana-María [aut],
Fernández-Breis, Jesualdo T. [aut],
Font, Roberto [aut],
Curran, James [cre]

Maintainer ``Curran, James" <j.curran@auckland.ac.nz>

Repository CRAN

Date/Publication 2025-08-28 17:40:10 UTC

Contents

`detc` *DET Curve calculation*

Description

From a response and predictors, the function calculates the DET curve for each pair (response, predictor). Optionally, it can compute this curve with a Confidence Interval (CI). Instead of a response and predictors, it can also receive a 'DETs' object to extract the results of the DET curves and compute the CIs.

Usage

```
detc(
  response = NULL,
  predictors = NULL,
  dets = NULL,
  names = NULL,
  conf = NULL,
  positive = "",
  parallel = FALSE,
  ncores = detectCores(),
  nboot = NULL,
  plot = FALSE,
  ...
)
```

Arguments

`response`	A factor, typically encoded with 0 (non-target) and 1 (target). Also it can be a dicothomous variable which will be treated as a factor. By default, the level of response is taken as target.
`predictors`	A matrix which columns represent the values of each predictor.
`dets`	A 'DETs' object which will be used to compute the DET curves.
`names`	A character vector that will be used to set the names of the DET Curves. It will also appear on the graphic legend when is plotted.

conf	If present, it represents the confidence level of the CI of the DET Curve, within [0,1]. Default: The CI will not be computed.
positive	A string with the name of the 'positive' level which is setting as reference level of 'response'.
parallel	If TRUE, the bootstrap method to calculated the CI is processed in parallel, using the backend provided by plyr (foreach).
ncores	The number of nodes to be forked for the parallel computation of the CI. Default: the maximum available. None used if parallel = FALSE.
nboot	The number of bootstrap replicates to be used for the computation of the CI. Default: 2000.
plot	If TRUE, the DETs curves will be plotted. Default: FALSE.
...	Further attributes that will be passed to the plot function.

Value

A 'DETs' object. This object contains in the attribute 'detCurves' the list of DET curves, one per classifier. Each DET curve is an object of class "DET". This object contains the parameters of the DET curve (false positive ratio, false negative ratio, and the thresholds used), along with the Equal Error Rate (EER). If the CI was calculated, it includes the median, upper and lower extremes of the CI of the false negative ratio and EER. Check the 'show' function to know more details on the outcomes saved in a 'DETs' object.

Examples

```
library(DET)
n = 500
#Predictors with normal distribution
set.seed(1235)
scoreNegative1 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(5321)
scoreNegative2 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive1 = rnorm(n, mean = 0.55, sd = 0.125)
set.seed(54321)
scorePositive2 = rnorm(n, mean = 0.65, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor1 = c(scoreNegative1, scorePositive1)
predictor2 = c(scoreNegative2, scorePositive2)
predictors = matrix(c(predictor1, predictor2), ncol = 2)
colnames(predictors) = c("DET1", "DET2")
detCurves = detc(
  response,
  predictors,
  positive = "target",
  names = colnames(predictors)
)

#Run in parallel for a faster execution activating logical argument 'parallel'
#and setting the number of cores of your computer
```

```
#logical argument 'parallel'
detCurvesWithConfidenceInterval = detc(
  response,
  predictors,
  positive = "target",
  names = colnames(predictors),
  conf = 0.95,
  parallel = TRUE,
  ncores = 2
)
```

`detc.ci` *DET Curve calculation with CI*

Description

From a 'DETs' object, the function extracts either computes the confidence interval (CI) of each DET curve of the object.

Usage

```
detc.ci(
  dets = NULL,
  conf = 0.95,
  positive = "",
  parallel = TRUE,
  ncores = detectCores(),
  nboot = 2000,
  plot = FALSE,
  ...
)
```

Arguments

`dets`	A 'DETs' object which will be used to extract or compute the CIs of the DET curves.
`conf`	A single numeric value into the (0,1) interval, which represents the confidence level of the CI of the DET Curve. Default: `conf = 0.95`.
`positive`	A string with the name of the 'positive' level which is setting as reference level of 'response'.
`parallel`	Boolean. By default `parallel = TRUE`. If TRUE, the bootstrap method to calculated the CI is processed in parallel, using the backend provided by `plyr` (foreach).
`ncores`	The number of nodes to be forked for the parallel computation of the CI. Default: the maximum available. None used if `parallel = FALSE`.

nboot	The number of bootstrap replicates to be used for the computation of the CI. Default: nboot = 2000.
plot	If TRUE, the CIs will be plotted for the DET curves. Default: plot = FALSE.
...	Further attributes that will be passed to the plot function.

Value

A 'DETs' object containing the list of DET curves with their CIs, one per classifier.

Examples

```
library(DET)
n = 500
#Predictors with normal distribution
set.seed(1235)
scoreNegative1 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(5321)
scoreNegative2 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive1 = rnorm(n, mean = 0.55, sd = 0.125)
set.seed(54321)
scorePositive2 = rnorm(n, mean = 0.65, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor1 = c(scoreNegative1, scorePositive1)
predictor2 = c(scoreNegative2, scorePositive2)
predictors = matrix(c(predictor1, predictor2), ncol = 2)
colnames(predictors) = c("DET1", "DET2")
detCurves = detc(
  response,
  predictors,
  positive = "target",
  names = colnames(predictors)
)

#Run in parallel for a faster execution activating logical argument 'parallel'
#and setting the number of cores of your computer
#logical argument 'parallel'
detCurvesWithConfidenceInterval = detc.ci(
  dets = detCurves,
  positive = "target",
  names = colnames(predictors),
  conf = 0.95,
  parallel = TRUE,
  ncores = 2
)
```

EER *Equal Error Rate computation*

Description

From two vectors of false positive and false negative rates which define the points of the curve, the function computes the Equal Error Rate (EER).

Usage

```
EER(fpr, fnr)
```

Arguments

fpr	A numeric vector representing the False Positive Rates.
fnr	A numeric vector representing the False Negative Rates.

Value

The Equal Error Rate (EER).

minDcf *Minimum of the Detection Cost Function computation*

Description

From a 'DET' object, the function computes the minimum value of the Detection Cost Function (minDCF).

Usage

```
minDcf(det, p = 0.01, cFp = 1, cFn = 10)
```

Arguments

det	An object of class "DET".
p	A single numeric value into the (0, 1) intervalrepresenting the prior probability of positive class.
cFp	A single numeric value representing the cost of False Positives.
cFn	A single numeric value representing the cost of False Negatives.

Value

A 'data.frame' with two attributes:

- 'minDcfValue': the computed minDCF.

- 'minDcfIndex': the index of the fpr and fnr in which the minimum is reached.

ovarianCancer *Data on Ovarian Cancer (NCI PBSII Data)*

Description

The database used correspond to proteomic spectra, generated by mass spectroscopy. This data dates from 6-19-02, and includes 91 controls (Normal) and 162 ovarian cancers. The raw spectral data of each sample contains the relative amplitude of the intensity at each molecular mass/charge (M/Z) identity. There are total 15154 M/Z identities. The intensity values were normalized according to the formula: $NV = (V - Min)/(Max - Min)$ where NV is the normalized value, V the raw value, Min the minimum intensity and Max the maximum intensity. The normalization is done over all the 253 samples for all 15154 M/Z identities. After the normalization, each intensity value falls within the range of 0 to 1.

Usage

```
data(ovarianCancer)
```

Format

An object of class "data.frame".

References

E. F. Petricoin, A. M. Ardekani, B. A. Hitt, P. J. Levine, V. A. Fusaro, S. M. Steinberg, G. B. Mills, C. Simone, D. A. Fishman, E. C. Kohn, L. A. Liotta (2002). Use of proteomic patterns in serum to identify ovarian cancer. The Lancet, 359(9306), 572–577. doi:10.1016/S01406736(02)077462

Examples

```
library(DET)
data(ovarianCancer)
response = as.factor(ovarianCancer$response)
predictors = matrix(c(as.numeric(ovarianCancer[[2]]),
                      as.numeric(ovarianCancer[[3]])), ncol = 2)
colnames(predictors) = c("Protein 1689", "Protein 1737")
detCurves =
  detc(
    response,
    predictors,
    names = colnames(predictors),
    positive = "Cancer"
  )
plot(detCurves, main = "Proteomic patterns")
```

plot.DET *DET Curve plot*

Description

From a 'DET' object, this function plots the DET curve included in the object.

Usage

```
## S3 method for class 'DET'
plot(x, ...)
```

Arguments

x	An object of class "DET".
...	Further graphical arguments passed to the plot function.

Examples

```
library(DET)
n = 5000
#Predictors with normal distribution
set.seed(1235)
scoreNegative = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive = rnorm(n, mean = 0.55, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor = matrix(c(scoreNegative, scorePositive), ncol = 1)
colnames(predictor) = c("DET")
detCurve = detc(response,
                predictor,
                names = colnames(predictor),
                positive = "target")
plot(detCurve@detCurves$DET,
     main = "Example")
```

plot.DETs *DET Curves plot*

Description

From a 'DETs' object generated with the detc function, this function plots the different DET curves included in the object. It includes the Confidence band in case the DETs curves were calculated with a confidence interval.

Usage

```
## S3 method for class 'DETs'
plot(x, ...)
```

Arguments

x	An object of class "DETs".
...	Further graphical arguments passed to the plot function.

Details

It accepts plot personalization with graphical parameters (see plot, for more details):

- 'xlim': a numeric vector of length 2, giving the x coordinate range of the plot.

- 'ylim': a numeric vector of length 2, giving the y coordinate range of the plot.

- 'col': a vector of colors, specifying the color for each DET curve.

- 'labels_x': a numeric vector indicating the labels of the X axis.

- 'labels_y': a numeric vector indicating the labels of the Y axis.

- 'xlab': a main label for the X axis.

- 'ylab': a main label for the Y axis.

- 'panel.first': a background grid is plotted. It can be used for modifying the background style of the graphic.

Examples

```
library(DET)
n = 5000
#Predictors with normal distribution
set.seed(1235)
scoreNegative1 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(5321)
scoreNegative2 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(6987)
scoreNegative3 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive1 = rnorm(n, mean = 0.55, sd = 0.125)
set.seed(54321)
scorePositive2 = rnorm(n, mean = 0.65, sd = 0.125)
set.seed(65987)
scorePositive3 = rnorm(n, mean = 0.75, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor1 = c(scoreNegative1, scorePositive1)
predictor2 = c(scoreNegative2, scorePositive2)
predictor3 = c(scoreNegative3, scorePositive3)
predictors = matrix(c(predictor1, predictor2, predictor3), ncol = 3)
colnames(predictors) = c("DET1", "DET2", "DET3")
detCurves = detc(
  response,
  predictors,
```

```
  positive = "target",
  names = colnames(predictors)
)
plot(detCurves,
     main = "Example",
     col = c("black", "blue", "red"))
```

---

plotROC *ROC Curve plot*

Description

From a 'DET' object, this function plots the ROC curve associated with the DET curve of the object. It also draws the confidence band when it is available in the object.

Usage

```
plotROC(dets, ...)
```

Arguments

dets	A 'DET' object from the list of a 'DETs' object computed by the detc function.
...	Further graphical arguments passed to the plot function.

Examples

```
library(DET)
n = 5000
#Predictors with normal distribution
set.seed(1235)
scoreNegative = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive = rnorm(n, mean = 0.55, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor = matrix(c(scoreNegative, scorePositive), ncol = 1)
colnames(predictor) = c("DET")
detCurve = detc(response,
                predictor,
                names = colnames(predictor),
                positive = "target")
plotROC(detCurve@detCurves$DET,
         main = "Example")
```

plotROCs *ROC Curves plot*

Description

From a 'DETs' object, this function plots the ROC curves associated with the DETs curves of the object. It includes the Confidence band when CIs were computed for the DETs curves.

Usage

```
plotROCs(dets, ...)
```

Arguments

dets	An object of class "DETs".
...	Further graphical arguments passed to the plot function.

Details

It accepts plot personalization with graphical parameters (see plot, for more details):

- 'xlim': a numeric vector of length 2, giving the x and y coordinate ranges of the plot.

- 'col': a vector of colors, specifying the color for each DET curve.

- 'labels_x': a numeric vector indicating the labels of the X and Y axes.

- 'xlab': a main label for the X axis.

- 'ylab': a main label for the Y axis.

- 'panel.first': a background grid is plotted. It can be used for modifying the background style of the graphic.

Examples

```
library(DET)
n = 5000
#Predictors with normal distribution
set.seed(1235)
scoreNegative1 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(5321)
scoreNegative2 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(6987)
scoreNegative3 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive1 = rnorm(n, mean = 0.55, sd = 0.125)
set.seed(54321)
scorePositive2 = rnorm(n, mean = 0.65, sd = 0.125)
set.seed(65987)
scorePositive3 = rnorm(n, mean = 0.75, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
```

```
predictor1 = c(scoreNegative1, scorePositive1)
predictor2 = c(scoreNegative2, scorePositive2)
predictor3 = c(scoreNegative3, scorePositive3)
predictors = matrix(c(predictor1, predictor2, predictor3), ncol = 3)
colnames(predictors) = c("DET1", "DET2", "DET3")
detCurves = detc(
  response,
  predictors,
  positive = "target",
  names = colnames(predictors)
)
plotROCs(detCurves,
          main = "Example",
          col = c("black", "blue", "red"))
```

`pointsEER` *EER Plot*

Description

From a 'DETs' object, this function plots the EER points of each classifier within the same graph of the DETs curves.

Usage

```
pointsEER(
  dets,
  pch = 19,
  col = c("black", "blue", "red", "green", "yellow"),
  lwd = 3,
  ...
)
```

Arguments

`dets`	An object of class "DETs".
`pch`	Symbol used for plotting the EER points, by default `pch = 19`.
`col`	A vector of colors, specifying the color of the points for each DET curve.
`lwd`	Line width used for drawing symbols, by default `lwd = 3`.
`...`	Further graphical arguments passed to the `points` function. For example, by default `pch = 19` and `lwd = 3` (see `points`, for more details).

Examples

```
library(DET)
n = 5000
#Predictors with normal distribution
set.seed(1235)
```

```
scoreNegative1 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(5321)
scoreNegative2 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(6987)
scoreNegative3 = rnorm(n, mean = 0.25, sd = 0.125)
set.seed(11452)
scorePositive1 = rnorm(n, mean = 0.55, sd = 0.125)
set.seed(54321)
scorePositive2 = rnorm(n, mean = 0.65, sd = 0.125)
set.seed(65987)
scorePositive3 = rnorm(n, mean = 0.75, sd = 0.125)
response = as.factor(c(rep(c("target"), times = n), rep(c("nontarget"), times = n)))
predictor1 = c(scoreNegative1, scorePositive1)
predictor2 = c(scoreNegative2, scorePositive2)
predictor3 = c(scoreNegative3, scorePositive3)
predictors = matrix(c(predictor1, predictor2, predictor3), ncol = 3)
colnames(predictors) = c("DET1", "DET2", "DET3")
detCurves = detc(
  response,
  predictors,
  positive = "target",
  names = colnames(predictors),
  plot = TRUE,
  main = "Example",
  col = c("black", "blue", "red"))

pointsEER(detCurves)
```

---

show.DETs *Show the structure of a DET object*

Description

From a 'DETs' object (generated with the detc function), the function shows the different attributes of each curve with a brief description, which have to be used to get the results for each curve.

Usage

```
show.DETs(dets)
```

Arguments

dets	An object of class "DETs".

speaker *Data on a Speaker Recognition System (Voxceleb verification test)*

Description

For our experiments, we have used the Voxceleb database, which contains more than one hundred thousand utterances extracted from Youtube interview videos. The database includes training and test sets that can be used for speaker recognition system development and performance evaluation respectively. The testing protocol consists of a list of utterance pairs, with the corresponding target or nontarget, and the task is to detect whether the two utterances belong to the same speaker or to different ones.

Usage

```
data(speaker)
```

Format

An object of class "data.frame".

Source

Web Archive

References

Nagraniy A, Chungy JS, Zisserman A (2017). Proceedings of the Annual Conference of the International Speech Communication Association, 950:2616–2620 (Publication)

Examples

```
library(DET)
data(speaker)
scoresLDA = speaker$scoresLDA
scoresPLDA = speaker$scoresPLDA
scoresLDAPLDA = speaker$scoresLDAPLDA
predictors = matrix(c(as.numeric(scoresLDA),
                      as.numeric(scoresPLDA),
                      as.numeric(scoresLDAPLDA)), ncol = 3)
colnames(predictors) = c("LDA", "PLDA", "LDAandPLDA")
response = as.factor(speaker$keys$V3)
detCurves =
  detc(
    response,
    predictors,
    names = colnames(predictors),
    positive = "target"
  )
plot(detCurves, main = "Voxceleb verification test")
```

Index