Package: PRROC (via r-universe)

September 8, 2024

Type Package

Title Precision-Recall and ROC Curves for Weighted and Unweighted Data

Version 1.3.1

Date 2018-06-19

Author Jan Grau and Jens Keilwagen

Maintainer Jan Grau <grau@informatik.uni-halle.de>

Description Computes the areas under the precision-recall (PR) and ROC curve for weighted (e.g., soft-labeled) and unweighted data. In contrast to other implementations, the interpolation between points of the PR curve is done by a non-linear piecewise function. In addition to the areas under the curves, the curves themselves can also be computed and plotted by a specific S3-method. References: Davis and Goadrich (2006) <doi:10.1145/1143844.1143874>; Keilwagen et al. (2014) <doi:10.1371/journal.pone.0092209>; Grau et al. (2015) <doi:10.1093/bioinformatics/btv153>.

License GPL-3

Suggests testthat, ggplot2, ROCR

NeedsCompilation no

Date/Publication 2018-06-19 10:42:55 UTC

Repository https://jgraux.r-universe.dev

RemoteUrl https://github.com/cran/PRROC

RemoteRef HEAD

RemoteSha 1d539299caec60badeb6aa1f4508dceeb258327e

Contents

PRROC-package *Compute and plot PR and ROC curves and the areas under the curves for weighted and unweighted data*

Description

This package computes the areas under the precision-recall (PR) and receiver operating characteristics (ROC) curve for weighted (e.g., soft-labeled) and unweighted data. In contrast to other implementations, the interpolation between points of the PR curve is done by a non-linear piecewise function. In addition to the areas under the curves, the curves themselves can also be computed and plotted by a specific S3-method.

Details

Package:	PRROC
Type:	Package
Version:	1.3
Date:	2017-04-21
License:	GPL-3

Author(s)

Jan Grau and Jens Keilwagen

Maintainer: Jan Grau <grau@informatik.uni-halle.de>

References

J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, New York, NY, USA, 2006. ACM.

T. Fawcett, An introduction to ROC analysis, Pattern Recognition Letters (27) 8, 861-874, 2006.

J. Keilwagen, I. Grosse, and J. Grau. Area under precision-recall curves for weighted and unweighted data, PLOS ONE (9) 3, 2014.

J. Grau, I. Grosse, and J. Keilwagen. PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R. Bioinformatics, 31(15):2595-2597, 2015.

See Also

`pr.curve`

`roc.curve`

`plot.PRROC`

`print.PRROC`

Examples

```
# create artificial scores as random numbers
x <- rnorm( 1000 );
y <- rnorm( 1000, -1 );

# compute area under PR curve
pr <- pr.curve( x, y );
print( pr );

# compute area under ROC curve
roc <- roc.curve( x, y );
print( roc );

# compute PR curve and area under curve
pr <- pr.curve( x, y, curve = TRUE );
# plot curve
plot(pr);

# compute ROC curve and area under curve
roc <- roc.curve( x, y, curve = TRUE );
# plot curve
plot(roc);

# create artificial weights
x.weights <- runif( 1000 );
y.weights <- runif( 1000 );

# compute PR curve and area under curve
pr <- pr.curve( x, y, x.weights, y.weights, curve = TRUE );
# plot curve
plot(pr);

# compute ROC curve and area under curve
roc <- roc.curve( x, y, x.weights, y.weights, curve = TRUE );
# plot curve
plot(roc);
```

`plot.PRROC` *Plotting PRROC objects*

Description

Plots the PR or ROC curves of a PRROC object. To obtain such curves, `pr.curve` or `roc.curve` must be called with argument `curve=TRUE`.

Usage

```
## S3 method for class 'PRROC'
plot(x, xlim=c(0,1), ylim=c(0,1), auc.main=TRUE,
```

```
auc.type=c("integral","davis.goadrich"),
legend=ifelse(is.logical(color) & color==TRUE,4,NA), xlab=NULL, ylab=NULL,
main=NULL, color=TRUE, lwd=3, add=FALSE,
scale.color=hsv(h=seq(0,1,length=100)*0.8, s=1, v=1),
max.plot = FALSE, min.plot = FALSE, rand.plot = FALSE,
fill.area = (max.plot & min.plot), maxminrand.col = grey(0.5),
fill.color = grey(0.95), ...)
```

Arguments

x	a PRROC object obtained from pr.curve or roc.curve
xlim	as in plot
ylim	as in plot
auc.main	TRUE to show the area under curve in the title
auc.type	the area under the curve shown in the title (see also pr.curve). Ignored if auc.main=FALSE or x$type=="ROC".
legend	if TRUE (and color==TRUE), a legend for the color scheme for the scores is shown on the right side of the main plot. If legend is a number between 1 and 4, the legend is drawn on the correspond side of the main plot (see axis). If legend is FALSE or NA, no legend is drawn.
xlab	the label of the x-axis. If NULL, label is chosen according the kind of the curve.
ylab	the label of the y-axis. If NULL, label is chosen according the kind of the curve.
main	the title of the plot. If NULL, title is chosen according the kind of the curve.
color	if TRUE, curve is plotted in colors reflecting score thresholds, if FALSE, the curve is plotted in black, if a color value (e.g., 2 or "red" for red) the curve is plotted in that color. For everything different from TRUE, the legend is omitted irrespective of the value of argument legend
lwd	the line width of the curve
add	if TRUE, the curve is added to an existing plot. Only works correctly, if in the previous call (with add==FALSE), no legend has been added to the plot.
scale.color	vector of colors that are used to reflect score thresholds, compare color
max.plot	if TRUE and x has been computed including the maximum curve, plot this maximum curve (ignored if add=TRUE)
min.plot	if TRUE and x has been computed including the minimum curve, plot this maximum curve (ignored if add=TRUE)
rand.plot	if TRUE and x has been computed including the curve of a random classifier, plot this curve (ignored if add=TRUE)
fill.area	fill the area between maximum and minimum curve (given both have been computed for x) (ignored if add=TRUE)
maxminrand.col	the plot color for the maximum, minimum, and random curves
fill.color	the fill color for the area between minimum and maximum curve
...	see plot

Details

The `plot` method for PRROC objects can be used in different ways.

The first is to plot a visualization of a single ROC or PR curve that also represents the classification thresholds of individual points on the curve by a color scale. In this case, a `PRROC` object must be provided as `x`, `add` must be `FALSE`, and `color` must be `TRUE`. If, in addition, `legend` is set to `TRUE`, a legend translating colors to numerical threshold values is included to the right of the curve plot itself. The layout of curve plot and legend is accomplished using `layout()`, which means that this type of ROC/PR plot cannot be combined with other/complex layouts.

The second application of the `plot` method is to compare the performance of different classifiers (typically on the same data set). To do so, `plot` must be called with `add=FALSE` and `color` set to one specific color (e.g., 2, "red",...) for the first `PRROC` object provided as `x`. Subsequent calls of `plot` with `add=TRUE` can be used to add further curves to the first plot, where different colors may be specified by the `color` parameter.

In both cases, the first (or only) call to `plot` also allows for including plots of the maximum and minimum curve, highlighting the area between minimum and maximum, and the curve of a random classifier. For this purpose, the `PRROC` object needs to be created (using `pr.curve` or `roc.curve`) with the corresponding parameters (e.g., `max.compute`) set to `TRUE`.

Additional examples for the different use cases and corresponding plot commands are given in the documentations of `pr.curve` and `roc.curve`.

Author(s)

Jan Grau and Jens Keilwagen

See Also

`pr.curve`

`roc.curve`

Examples

```
# create artificial scores as random numbers
x <- rnorm( 1000 );
y <- rnorm( 1000, -1 );

# compute PR curve
pr <- pr.curve( x, y, curve = TRUE );

# standard plot of PR curve
plot( pr );

# compute ROC curve
roc <- roc.curve( x, y, curve = TRUE );

# standard plot of ROC curve
plot( roc );

# create another set of scores
```

```
x.2 <- rnorm( 1000 );
y.2 <- rnorm( 1000, -2 );

# compute PR curve
pr.2 <- pr.curve( x.2, y.2, curve=TRUE );
# and ROC curve
roc.2 <- roc.curve( x.2, y.2, curve=TRUE );

# plot PR curve in red, without legend
plot( pr, color = "red", auc.main=FALSE );
# add second PR curve in green
plot( pr.2, color = 3, add = TRUE );


# plot ROC curve in red, without legend
plot( roc, color = "red", auc.main=FALSE);
# add second ROC curve in green
plot( roc.2, color = 3, add = TRUE );

# plot PR curve with legend below the main plot
plot( pr, legend=1 );

# compute PR curve with minimum and maximum curve, and random classifier
pr <- pr.curve( x, y, curve = TRUE, max.compute = TRUE,
  min.compute = TRUE, rand.compute = TRUE);

# plot PR curve with area between minimum and
# maximum curve in green and random classifier in blue
plot(pr, rand.plot = TRUE, fill.area = TRUE, fill.color = rgb(0.8,1,0.8),
  maxminrand.col = "blue" );
```

`pr.curve` *PR curve*

Description

Computes the area under the precision-recall (PR) curve for weighted and unweighted data. In contrast to other implementations, the interpolation between points of the PR curve is done by a non-linear piecewise function. In addition to the area under the curve, the curve itself can be obtained by setting argument `curve` to `TRUE`.

Usage

```
pr.curve( scores.class0, scores.class1=scores.class0, weights.class0=NULL,
    weights.class1 = {if(is.null(weights.class0)){NULL}else{1-weights.class0}},
    sorted = FALSE, curve = FALSE,
   minStepSize=min(1,ifelse(is.null(weights.class0),1,sum(weights.class0)/100)),
    max.compute=F, min.compute=F, rand.compute=F,dg.compute=T)
```

Arguments

scores.class0	the classification scores of i) all data points or ii) only the data points belonging to the positive class. In the first case, scores.class1 should not be assigned an explicit value, but left at the default (scores.class1=scores.class0). In addition, weights.class0 needs to contain the class labels of the data points (1 for positive class, 0 for negative class) or the soft-labels for the positive class, i.e., the probability for each data point to belong to the positive class. Accordingly, weights.class1 should be left at the default value (1-weights.class0). In the second case, the scores for the negative data points need to be provided in scores.class1. In this case, weights.class0 and weights.class1 need to be provided only for soft-labelling and should be of the same length as scores.class0 and scores.class1, respectively.
scores.class1	the scores of the negative class if provided separately (see scores.class0)
weights.class0	the weights for the data points of the positive class in same ordering as scores.class0 (optional)
weights.class1	the weights for the data points of the negative class in same ordering as scores.class1 (optional)
sorted	TRUE if the scores are already sorted
curve	TRUE if the curve should also be returned, FALSE otherwise
minStepSize	the minimum step size between intermediate points of the curve, does not affect the computation of AUC-PR
max.compute	TRUE if the maximum PR curve given the supplied weights should be computed
min.compute	TRUE if the minimum PR curve given the supplied weights should be computed
rand.compute	TRUE if the PR curve of a random classifier given the supplied weights should be computed
dg.compute	TRUE if the area under the curve according to the interpolation of Davis and Goadrich should be computed. Reduces runtime if switched off.

Details

This function computes the area under a precision-recall curve and, optionally, the curve itself and returns it as a PRROC object (see below). It can be used under different scenarios:

1. Standard, hard-labeled classification problems:

Each data point is uniquely assigned to one out of two possible classes. In this case, the classification scores may be either provided separately for the data points of each of the classes, i.e., as scores.class0 for the data points from the positive/foreground class and as scores.class1 for the data points of the negative/background class; or the classification scores for all data points are provided as scores.class0 and the labels are provided as numerical values (1 for the positive class, 0 for the negative class) as weights.class0.

2. Weighted, hard-labeled classification problems:

Each data point is uniquely assigned to one out of two possible classes, where each data points additionally has a weight assigned, for instance multiplicities in the original data set. In this case,

the classification scores need to be provided separately for the data points of each of the classes, i.e., as `scores.class0` for the data points from the positive/foreground class and as `scores.class1` for the data points of the negative/background class. In addition, the weights for the data points must be provided as `weights.class0` and `weights.class1`, respectively.

3. Soft-labeled classification problems:

Each data point belongs to both of the two classes with a certain probability, where for each data point, these two probabilities add up to 1. In this case, the classification scores for all data points need to be provided only once as `scores.class0` and only the positive/foreground weights for each data point need to be provided in `weights.class0`, while the converse probability for the negative class is automatically set to `weights.class1=1.0-weights.class0`.

Value

`type`	always `"PR"`
`auc.integral`	area under the curve computed by integration of the piecewise function
`auc.davis.goadrich`	area under the curve computed using the interpolation of Davis & Goadrich (2006). Is `NA` if weights are provided and different from `1`.
`curve`	the PR curve as a matrix, where the first column contains recall, the second contains precision, and the third contains the corresponding threshold on the scores.
`max`	the maximum PR curve (if `max.compute=TRUE`)
`min`	the minimum PR curve (if `min.compute=TRUE`)
`rand`	the PR curve of a random classifier (if `rand.compute=TRUE`)

Author(s)

Jan Grau and Jens Keilwagen

References

J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, New York, NY, USA, 2006. ACM.

J. Keilwagen, I. Grosse, and J. Grau. Area under precision-recall curves for weighted and unweighted data, PLOS ONE (9) 3, 2014.

J. Grau, I. Grosse, and J. Keilwagen. PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R. Bioinformatics, 31(15):2595-2597, 2015.

See Also

`roc.curve`

`plot.PRROC`

Examples

```
# create artificial scores as random numbers
x <- rnorm( 1000 );
y <- rnorm( 1000, -1 );
# compute area under PR curve for the hard-labeled case
pr <- pr.curve( x, y );
print( pr );

# compute PR curve and area under curve
pr <- pr.curve( x, y, curve = TRUE );
# plot curve
plot(pr);

# create artificial weights
x.weights <- runif( 1000 );
y.weights <- runif( 1000 );

# compute PR curve and area under curve for weighted, hard-labeled data
pr <- pr.curve( x, y, x.weights, y.weights, curve = TRUE );
# and plot the curve
plot(pr);


# compute PR curve and area under curve,
# and maximum, minimum, and random PR curve for weighted, hard-labeled data
pr <- pr.curve(x, y, x.weights, y.weights, curve = TRUE, max.compute = TRUE,
  min.compute = TRUE, rand.compute = TRUE);
# plot all three curves
plot(pr, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE)


# concatenate the drawn scores
scores<-c(x,y);
# and create artificial soft-labels
weights<-c(runif(1000, min = 0.5, max = 1), runif(1000, min = 0, max = 0.5))

# compute PR curve and area under curve,
# and maximum, minimum, and random PR curve for soft-labeled data
pr<-pr.curve(scores.class0 = scores, weights.class0 = weights, curve = TRUE,
  max.compute = TRUE, min.compute = TRUE, rand.compute = TRUE);
# plot all three curves
plot(pr, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE)

# print the areas under the curves
print(pr);


# generate classification scores of a second classifier
scores.2<-c(rnorm( 1000 ),rnorm( 1000, -2 ));
# and compute the PR curve
pr.2<-pr.curve(scores.class0 = scores.2, weights.class0 = weights, curve = TRUE)
# plot all three curves for the first classifier in red
```

```
plot(pr, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE,
  color="red", auc.main=FALSE)
# and add the curve for the second classifier
plot(pr.2, add=TRUE, color="green")
```

`print.PRROC` *printing PRROC objects*

Description

Prints a PRROC object.

Usage

```
## S3 method for class 'PRROC'
print(x, ...)
```

Arguments

`x`	a PRROC object obtained from `pr.curve` or `roc.curve`
`...`	see `print`

Details

The print method for PRROC objects prints the area under the (PR or ROC) curve, and (if `curve=TRUE` in `pr.curve` or `roc.curve`) the range of classification scores. If also `max.compute=TRUE`, `min.compute=TRUE`, and/or `rand.compute=TRUE` when the PRROC object has been computes using `pr.curve` or `roc.curve`, a relative area under curve is reported, i.e., the minimal AUC subtracted from the original AUC and the result divided by the difference of maximum and minimum AUC.

Author(s)

Jan Grau and Jens Keilwagen

See Also

`pr.curve`

`roc.curve`

Examples

```
# create artificial scores as random numbers
x <- rnorm( 1000 );
y <- rnorm( 1000, -1 );
# compute area under PR curve
pr <- pr.curve( x, y );
print( pr );
```

```
# compute area under ROC curve
roc <- roc.curve( x, y );
print( roc );
```

`roc.curve` *ROC curve*

Description

Computes the area under the receiver operating characteristics (ROC) curve for weighted and unweighted data. In addition to the area under the curve, the curve can be obtained by setting argument `curve` to `TRUE`.

Usage

```
roc.curve( scores.class0, scores.class1=scores.class0, weights.class0=NULL,
    weights.class1 = {if(is.null(weights.class0)){NULL}else{1-weights.class0}},
    sorted = FALSE, curve = FALSE,
    max.compute=F, min.compute=F, rand.compute=F)
```

Arguments

`scores.class0`	the classification scores of i) all data points or ii) only the data points belonging to the positive class. In the first case, scores.class1 should not be assigned an explicit value, but left at the default (scores.class1=scores.class0). In addition, weights.class0 needs to contain the class labels of the data points (1 for positive class, 0 for negative class) or the soft-labels for the positive class, i.e., the probability for each data point to belong to the positive class. Accordingly, weights.class1 should be left at the default value (1-weights.class0). In the second case, the scores for the negative data points need to be provided in scores.class1. In this case, weights.class0 and weights.class1 need to be provided only for soft-labelling and should be of the same length as scores.class0 and scores.class1, respectively.
`scores.class1`	the scores of the negative class if provided separately (see scores.class0)
`weights.class0`	the weights for the data points of the positive class in same ordering as `scores.class0` (optional)
`weights.class1`	the weights for the data points of the negative class in same ordering as `scores.class1` (optional)
`sorted`	`TRUE` if the scores are already sorted
`curve`	`TRUE` if the curve should also be returned, `FALSE` otherwise
`max.compute`	`TRUE` if the maximum ROC curve given the supplied weights should be computed
`min.compute`	`TRUE` if the minimum ROC curve given the supplied weights should be computed
`rand.compute`	`TRUE` if the ROC curve of a random classifier given the supplied weights should be computed

Details

This function computes the area under a receiver-operating characteristic (ROC) curve and, optionally, the curve itself and returns it as a PRROC object (see below). It can be used under different scenarios:

1. Standard, hard-labeled classification problems:

Each data point is uniquely assigned to one out of two possible classes. In this case, the classification scores may be either provided separately for the data points of each of the classes, i.e., as scores.class0 for the data points from the positive/foreground class and as scores.class1 for the data points of the negative/background class; or the classification scores for all data points are provided as scores.class0 and the labels are provided as numerical values (1 for the positive class, 0 for the negative class) as weights.class0.

2. Weighted, hard-labeled classification problems:

Each data point is uniquely assigned to one out of two possible classes, where each data points additionally has a weight assigned, for instance multiplicities in the original data set. In this case, the classification scores need to be provided separately for the data points of each of the classes, i.e., as scores.class0 for the data points from the positive/foreground class and as scores.class1 for the data points of the negative/background class. In addition, the weights for the data points must be provided as weights.class0 and weights.class1, respectively.

3. Soft-labeled classification problems:

Each data point belongs to both of the two classes with a certain probability, where for each data point, these two probabilities add up to 1. In this case, the classification scores for all data points need to be provided only once as scores.class0 and only the positive/foreground weights for each data point need to be provided in weights.class0, while the converse probability for the negative class is automatically set to weights.class1=1.0-weights.class0.

Value

type	always "ROC"
auc	area under the curve
curve	the ROC curve as a matrix, where the first column contains the false-positive rate, the second contains recall (sensitivity), and the third contains the corresponding threshold on the scores.
max	the maximum ROC curve (if max.compute=TRUE)
min	the minimum ROC curve (if min.compute=TRUE)
rand	the ROC curve of a random classifier (if rand.compute=TRUE)

Author(s)

Jan Grau and Jens Keilwagen

References

J. Keilwagen, I. Grosse, and J. Grau. Area under precision-recall curves for weighted and unweighted data, PLOS ONE (9) 3, 2014.

J. Grau, I. Grosse, and J. Keilwagen. PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R. Bioinformatics, 31(15):2595-2597, 2015.

See Also

pr.curve

plot.PRROC

Examples

```
# create artificial scores as random numbers
x <- rnorm( 1000 );
y <- rnorm( 1000, -1 );
# compute area under ROC curve for the hard-labeled case
roc <- roc.curve( x, y );
print( roc );

# compute ROC curve and area under curve
roc <- roc.curve( x, y, curve = TRUE );
# plot curve
plot(roc);


# create artificial weights
x.weights <- runif( 1000 );
y.weights <- runif( 1000 );

# compute ROC curve and area under curve for weighted, hard-labeled data
roc <- roc.curve( x, y, x.weights, y.weights, curve = TRUE );
# and plot the curve
plot(roc);


# compute ROC curve and area under curve,
# and maximum, minimum, and random ROC curve for weighted, hard-labeled data
roc <- roc.curve(x, y, x.weights, y.weights, curve = TRUE, max.compute = TRUE,
  min.compute = TRUE, rand.compute = TRUE);
# plot all three curves
plot(roc, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE)


# concatenate the drawn scores
scores<-c(x,y);
# and create artificial soft-labels
weights<-c(runif(1000, min = 0.5, max = 1), runif(1000, min = 0, max = 0.5))

# compute ROC curve and area under curve,
# and maximum, minimum, and random ROC curve  for soft-labeled data
roc<-roc.curve(scores.class0 = scores, weights.class0 = weights, curve = TRUE,
  max.compute = TRUE, min.compute = TRUE, rand.compute = TRUE);
# plot all three curves
plot(roc, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE)

# print the areas under the curves
print(roc);
```

```
# generate classification scores of a second classifier
scores.2<-c(rnorm( 1000 ),rnorm( 1000, -2 ));
# and compute the ROC curve
roc.2<-roc.curve(scores.class0 = scores.2, weights.class0 = weights, curve = TRUE)
# plot all three curves for the first classifier in red
plot(roc, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE, fill.area = TRUE,
  color="red", auc.main=FALSE)
# and add the curve for the second classifier
plot(roc.2, add=TRUE, color="green")
```

Index