

Package ‘Daim’

June 24, 2009

Version 1.0.0

Title Diagnostic accuracy of classification models.

Author Sergej Potapov, Werner Adler and Berthold Lausen.

Description Several functions for evaluating the accuracy of classification models. The package provide the following performance measures: “cv”, “0.632” and “0.632+” estimation of the misclassification rate, sensitivity, specificity and auc. If an application is computationally intensive, parallel execution can be used to reduce the time taken.

Maintainer Sergej Potapov <sergej.potapov@googlemail.com>

Depends R (>= 2.6.0)

Suggests snow, multicore, MASS, rpart, ipred, e1071, randomForest, mboost

Date 24.06.2009

License GPL-2

Repository CRAN

Date/Publication 2009-06-24 14:04:34

R topics documented:

auc	2
auc.Daim	3
Daim	4
Daim.control	7
Daim.data1	8
Daim.data3	9
performDaim	10
plot.Daim	11
plot.roc.Daim	12
print.Daim	13
roc	14
roc.area	15
summary.Daim	16

auc

The area under the ROC curve (AUC)

Description

This function computes the area under an ROC curve.

Usage

```
## S3 method for class 'numeric':  
auc(x, y, ...)
```

Arguments

x	sensitivity.
y	specificity.
...	additional parameters.

Value

a scalar number

See Also

[auc.Daim](#), [roc.area.Daim](#), [Daim](#)

Examples

```
data(Daim.data1)  
perform <- performDaim(Daim.data1$prob.oob, Daim.data1$labels,  
  Daim.data1$prob.app)  
  
####  
#### compute the .632+ estimation of the AUC.  
####  
  
auc(perform$roc$sens632p, perform$roc$spec632p)
```

auc.Daim	<i>The area under the ROC curve (AUC)</i>
----------	---

Description

This function computes the area under an ROC curve.

Usage

```
## S3 method for class 'Daim':  
auc(x, ...)
```

Arguments

x	an object of class Daim.
...	additional parameters.

Value

a list with following components :

auc.632p	the .632+ estimation of the AUC.
auc.632	the .632 estimation of the AUC.
auc.loob	the LOOB estimation of the AUC.
auc.app	the apparent estimation of the AUC.
auc.samples	the AUC values for each bootstrap sample.

See Also

[auc.numeric](#), [Daim](#), [plot.Daim](#), [performDaim](#)

Examples

```
data(Daim, data1)  
perform <- performDaim(Daim.data1$prob.oob, Daim.data1$labels,  
                      Daim.data1$prob.app)  
  
auc(perform)  
roc.area(perform)
```

Description

Estimation of prediction error based on cross-validation (CV) or various bootstrap techniques.

Usage

```
Daim(formula, model=NULL, data=NULL, control = Daim.control(),
      thres = seq(0,1,by=0.01), cutoff = 0.5,
      labpos = "1", returnSample = FALSE,
      cluster = NULL, seed.cluster = NULL, ...)
```

Arguments

<code>formula</code>	formula of the form $y \sim x_1 + x_2 + \dots$, where y must be a factor and x_1, x_2, \dots are numeric or factor.
<code>model</code>	function. Modelling technique whose error rate is to be estimated. The function <code>model</code> returns the predicted probability for each observation.
<code>data</code>	an optional data frame containing the variables in the model (training data).
<code>control</code>	See Daim.control .
<code>thres</code>	a numeric vector with the cutoff values.
<code>cutoff</code>	the cutoff value for error estimation. This can be a numeric value or a character string. If the <code>cutoff</code> set to: <code>".632"</code> - the estimated cut-point corresponding the <code>.632</code> estimation of the sensitivity and the specificity. <code>".632+"</code> - the estimated cut-point corresponding the <code>.632+</code> estimation of the sensitivity and the specificity.
<code>labpos</code>	a character string of the response variable that defines a "positive" event. The labels of the "positive" events will be set to "pos" and other to "neg".
<code>returnSample</code>	a logical value for saving the data from each sample.
<code>cluster</code>	the name of the cluster, if parallel computing will be used.
<code>seed.cluster</code>	an integer value used as seed for the RNG.
<code>...</code>	additional parameters.

Value

a list with the following components :

<code>call</code>	the matched call.
<code>formula</code>	the formula supplied.
<code>method</code>	the list of control parameters.

<code>err632p</code>	the .632+ estimation of the misclassification error.
<code>err632</code>	the .632 estimation of the misclassification error.
<code>errloob</code>	the LOOB estimation of the misclassification error.
<code>errapp</code>	the apparent error.
<code>sens632p</code>	the .632+ estimation of the sensitivity.
<code>spec632p</code>	the .632+ estimation of the specificity.
<code>sens632</code>	the .632 estimation of the sensitivity.
<code>spec632</code>	the .632 estimation of the specificity.
<code>sensloob</code>	the LOOB estimation of the sensitivity.
<code>specloob</code>	the LOOB estimation of the specificity.
<code>sensapp</code>	the apparent sensitivity.
<code>specapp</code>	the apparent specificity.
<code>roc</code>	a data frame with estimated values of sensitivity and specificity for a variety of cutoffs.
<code>sample.roc</code>	a list in which each entry contains the values of the ROC curve of this special sample or cross-validation run.
<code>sample.data</code>	a data frame with the results of this particular sample or cross-validation run.

References

Werner Adler and Berthold Lausen (2009).
Bootstrap Estimated True and False Positive Rates and ROC Curve.
Computational Statistics & Data Analysis, **53**, (3), 718–729.

Tom Fawcett (2006).
An introduction to ROC analysis.
Pattern Recognition Letters, **27**, (8).

Bradley Efron and Robert Tibshirani (1997).
Improvements on cross-validation: The .632+ bootstrap method.
Journal of the American Statistical Association, **92**, (438), 548–560.

See Also

[plot.Daim](#), [performDaim](#), [auc.Daim](#), [roc.area.Daim](#)

Examples

```
library(ipred)
data(GlaucomaM)
head(GlaucomaM)

mylda <- function(formula, train, test) {
```

```

        model <- lda(formula,train)
        predict(model,test)$posterior[, "pos"]
    }

ACC <- Daim(Class~.,model=mylda,data=GlaucomaM,labpos="glaucoma")
ACC
summary(ACC)

####
#### for parallel computing with snow cluster
####

# library(snow)
###
### create cluster with two slave nodes

# cl <- makeCluster(2)

###
### Load used library on all slaves and execute the Daim in parallel
###

# clusterEvalQ(cl, library(ipred))
# ACC <- Daim(Class~.,model=mylda,data=GlaucomaM,labpos="glaucoma",cluster=cl)
# ACC

####
#### for parallel computing with multicore package
#### you need only to load this library
####

# library(multicore)
# ACC <- Daim(Class~.,model=mylda,data=GlaucomaM,labpos="glaucoma")
# ACC

library(randomForest)

myRF <- function(formula,train,test){
    model <- randomForest(formula,train)
    predict(model,test,type="prob")[, "pos"]
}

ACC2 <- Daim(Class~.,model=myRF,data=GlaucomaM,labpos="glaucoma",
    control=Daim.control(number=25))
ACC2
summary(ACC2)

####
#### for parallel computing with snow cluster
####

# library(snow)
###

```

```

### create cluster with two slave nodes

# cl <- makeCluster(2)

###
### Load used library on all slaves and execute the Daim in parallel
###

# clusterEvalQ(cl, library(randomForest))
# ACC2 <- Daim(Class~.,model=myRF,data=GlaucomaM,labpos="glaucoma",cluster=cl)
# ACC2

####
#### for parallel computing with multicore package
####

# library(multicore)
# ACC2 <- Daim(Class~.,model=myRF,data=GlaucomaM,labpos="glaucoma")
# ACC2

```

Daim.control

Control parameters for the diagnostic accuracy of models.

Description

Control of resampling methods.

Usage

```

Daim.control(method="boot", number = 100, replace = TRUE,
             boot.size = 1, k = 10, k.runs = 10,
             dependency = list(var = NULL, keep.id = FALSE))

```

Arguments

method	The resampling method: boot, cv
number	the number of bootstrap samples.
replace	a logical indicating whether sampling of observations is done with or without replacement.
boot.size	percentage of observations ($0 < \text{boot.size} < 1$) to draw without replacement (only relevant if <code>replace = FALSE</code>). In this case subagging (Buehlmann and Yu, 2002) without replacement is performed.
k	the number of folds
k.runs	the number of runs of k-fold cross-validations
dependency	...

References

Leo Breiman (1996b), Out-Of-Bag Estimation. *Technical Report*.
<ftp://ftp.stat.berkeley.edu/pub/users/breiman/OOBestimation.ps.Z>.

Peter Buehlmann and Bin Yu (2002), Analyzing Bagging.
The Annals of Statistics **30**(4), 927–961.

See Also

[Daim](#)

Examples

```
###  
### for bootstrap  
###  
  
Daim.control(method="boot", number=100)  
  
###  
### for cross-validation  
###  
  
Daim.control(method="cv", k=10, k.runs=10)  
  
###  
### for subbagging or subsampling  
###  
  
Daim.control(method="boot", number=100, replace=FALSE,  
             boot.size=0.9)
```

Daim.data1

Data set: Artificial bootstrap data for use with Daim

Description

The data set containing 100 sets of predictions, id's and corresponding labels were obtained from 100 bootstrap samples.

Usage

```
data(Daim.data1)
```

Format

A three element list. The first element, `Daim.data1$prob.oob`, is itself a matrix where rows are observations and columns are the (bootstrap) samples. Each of these 100 columns is a numerical prediction for each bootstrap sample. The second list entry, `Daim.data1$prob.app` is a vector of numerical apparent predictions. The third list entry, `Daim.data1$labels` is a 100 element list in which each element is a vector of true class labels corresponding to the predictions.

Examples

```
data(Daim.data1)
perform <- performDaim(Daim.data1$prob.oob, Daim.data1$labels,
  Daim.data1$prob.app)
perform
plot(perform)
```

`Daim.data3`*Data set: Artificial data for use with Daim*

Description

The Artificial data set.

Usage

```
data(Daim.data1)
```

Format

A `data.frame` containing 4 variables and 196 observations.

Examples

```
data(Daim.data3)
head(Daim.data3)
M <- roc(Daim.data3[,-1], Daim.data3[,1], labpos="pos")
summary(M)
```

 performDaim

Plotting method for Daim Objects

Description

Plot an Daim object generated by [Daim](#) function.

Usage

```
## S3 method for class 'matrix':
performDaim(x, labels, prediction=NULL,
            thres=seq(0,1,by=0.01),
            cutoff=0.5, labpos="1", ...)
## S3 method for class 'data.frame':
performDaim(x, labels, prediction=NULL,
            thres=seq(0,1,by=0.01),
            cutoff=0.5, labpos="1", ...)
```

Arguments

x	a matrix or data frame containing the predictions.
labels	a vector containing the true class labels. It can be a factor or character vector.
prediction	a vector containing the prediction probability obtained by a model: see Daim .
thres	a numeric vector with the cutoff values.
cutoff	the cutoff value for error estimation. This can be a numeric value or a character string. If the <code>cutoff</code> set to: ".632" - the estimated cut-point corresponding the .632 estimation of the sensitivity and the specificity. ".632+" - the estimated cut-point corresponding the .632+ estimation of the sensitivity and the specificity.
labpos	a character string of the variable <code>labels</code> that defines a "positive" event.
...	additional parameters.

See Also

[Daim](#), [plot.Daim](#)

Examples

```
data(Daim.data1)
perform <- performDaim(Daim.data1$prob.oob,
                      Daim.data1$labels, Daim.data1$prob.app)
perform
```

plot.Daim

Plotting method for Daim Objects

Description

Plot a Daim object generated by the [Daim](#) function.

Usage

```
## S3 method for class 'Daim':
plot(x, method=NULL, all.roc=FALSE, color="red",
      alpha=0.25, type="b", xlab="False positive rate",
      ylab="True positive rate", main=NULL, add=FALSE, ...)
```

Arguments

x	an object of class Daim.
method	kind of the estimation of the ROC curve: '.632+', '.632', 'loob', 'cv', 'sample'
all.roc	logical. Should ROC curves from all samples be plotted ?
color	the color used to draw the ROC curve.
alpha	Semi-transparent color: see rgb .
type	what type of plot should be drawn: see argument 'type' by the function plot .
xlab	a title for the x axis: see title .
ylab	a title for the y axis: see title .
main	a main title for the plot, see also title .
add	logical specifying if roc-area should be added to an already existing plot.
...	graphical parameters can be given as arguments to 'plot'.

See Also

[Daim](#), [roc.area.Daim](#)

Examples

```
library(Daim)
data(Daim.data1)
perform <- performDaim(Daim.data1$prob.oob, Daim.data1$labels,
                      Daim.data1$prob.app)
summary(perform)

par(mfrow=c(2,2))
plot(perform,method=".632+")
plot(perform,method="sample")
plot(perform,method=".632+",main="Comparison between methods")
plot(perform,method=".632",col="blue",add=TRUE)
```

```

plot(perform,method="loob",col="green",add=TRUE)
legend("bottomright",c(".632+", ".632", "loob"),
      col=c("red", "blue", "green"),lty=1,inset=0.01)
plot(perform,all.roc=TRUE)

####
#### If your device don't support the semi-transparent colors use
#### the PDF Graphics Device.
####

pdf("plot-Daim.pdf",version=1.4)
plot(perform,method=".632+")
plot(perform,method="sample")
plot(perform,method=".632+",main="Comparison between methods")
plot(perform,method=".632",col="blue",add=TRUE)
plot(perform,method="loob",col="green",add=TRUE)
legend("bottomright",c(".632+", ".632", "loob"),
      col=c("red", "blue", "green"),lty=1,inset=0.01)
plot(perform,all.roc=TRUE)
dev.off()

```

plot.roc.Daim

Plotting method for roc.Daim Objects

Description

Plot a roc.Daim object generated by the `roc` function.

Usage

```

## S3 method for class 'Daim.vector':
plot(x, color="blue", type="l", bty="n",
     xlab="False positive rate", ylab="True positive rate",
     main="ROC curve", ...)
## S3 method for class 'Daim.list':
plot(x, color=rgb(1,0,0,alpha=0.5),
     xlab="False positive rate", ylab="True positive rate",
     main="ROC curves", legend=TRUE, ...)

```

Arguments

<code>x</code>	an object of class <code>roc.Daim</code> .
<code>color</code>	the color used to draw the ROC curve.
<code>type</code>	what type of plot should be drawn: see argument 'type' by the function <code>plot</code> .
<code>bty</code>	the type of box to be drawn around the legend: see <code>legend</code> .
<code>xlab</code>	a title for the x axis: see <code>title</code> .
<code>ylab</code>	a title for the y axis: see <code>title</code> .

main	a main title for the plot, see also title .
legend	Should a legend be added?
...	graphical parameters can be given as arguments to 'plot'.

See Also

[Daim](#), [roc.area.Daim](#)

Examples

```
library(Daim)
data(Daim.data3)

M <- roc(Daim.data3[,2], Daim.data3[,1], "pos")
plot(M)

M <- roc(Daim.data3[,-1], Daim.data3[,1], "pos")
plot(M, color=1:4)
```

print.Daim	<i>Print Method for Daim Object</i>
------------	-------------------------------------

Description

Print object of class Daim.

Usage

```
## S3 method for class 'Daim':
print(x, digits=max(3, getOption("digits") - 3), ...)
```

Arguments

x	object of class Daim.
digits	a non-null value for digits specifies the minimum number of significant digits to be printed in values.
...	additional arguments.

Details

An object of class Daim is printed.

See Also

[summary.Daim](#), [Daim](#), [plot.Daim](#)

Examples

```
library(Daim)
data(Daim.data1)
perform <- performDaim(Daim.data1$prob.oob,
                      Daim.data1$labels, Daim.data1$prob.app)
perform
```

roc

Compute a ROC curve

Description

This function computes sensitivity and specificity for a variety of cut-points.

Usage

```
## S3 method for class 'integer':
roc(x, ...)
## S3 method for class 'numeric':
roc(x, labels, labpos, thres=NULL, ...)
## S3 method for class 'matrix':
roc(x, labels, labpos, thres=NULL, ...)
## S3 method for class 'data.frame':
roc(x, ...)
```

Arguments

<code>x</code>	an object (vector, matrix, data.frame) used for prediction.
<code>labels</code>	a vector containing the true class labels. This can be a factor or character vector.
<code>labpos</code>	a character string of the variable <code>labels</code> that defines a "positive" event.
<code>thres</code>	a numeric vector with the cutoff values. By default, the <code>x</code> define the grid of cut-points.
<code>...</code>	additional parameters.

See Also

[plot.Daim](#), [auc.Daim](#)

Examples

```
data(Daim.data3)

M <- roc(Daim.data3[,2:5], Daim.data3$Gold, "pos")
summary(M)
plot(M, color=c("black", "blue", "green3", "red"))
```

```
roc.area(M)
```

```
roc.area
```

Plot the area under the ROC curve

Description

This function plots the ROC curve and fills the area under this curve.

Usage

```
## S3 method for class 'Daim':
roc.area(x, method=NULL, col="red", area.color=rgb(1,0,0,alpha=0.5),
         xlab="False positive rate", ylab="True positive rate",
         density=NULL, angle=4, border=NULL, add=FALSE, ...)
## S3 method for class 'Daim.list':
roc.area(x, col="black", area.color=rgb(1,0,0,alpha=0.5),
         xlab="False positive rate", ylab="True positive rate",
         main="ROC curves", density=NULL, angle=4, border=NULL,
         add=FALSE, ...)
## S3 method for class 'Daim.vector':
roc.area(x, col="red", area.color=rgb(1,0,0,alpha=0.5),
         xlab="False positive rate", ylab="True positive rate",
         main="ROC curve", density=NULL, angle=4, border=NULL,
         add=FALSE, ...)
```

Arguments

<code>x</code>	an object of class <code>Daim</code> , <code>Daim.list</code> or <code>Daim.vector</code> .
<code>method</code>	kind of the estimation of the ROC curve: <code>' .632+</code> , <code>' .632'</code> , <code>' loob'</code> , <code>' cv'</code> , <code>' sample'</code>
<code>col</code>	the color used to draw the ROC curve.
<code>area.color</code>	the color for filling the area.
<code>xlab</code>	a title for the x axis: see <code>' title'</code> .
<code>ylab</code>	a title for the y axis: see <code>' title'</code> .
<code>main</code>	a main title for the plot, see also title .
<code>density</code>	the density of shading lines, in lines per inch. The default value of <code>' NULL'</code> means that no shading lines are drawn. A zero value of <code>' density'</code> means no shading nor filling whereas negative values (and <code>' NA'</code>) suppress shading (and so allow color filling).
<code>angle</code>	the slope of shading lines, given as an angle in degrees (counter-clockwise).
<code>border</code>	the color to draw the border. The default, <code>' NULL'</code> , means to use <code>' par("fg")'</code> . Use <code>' border = NA'</code> to omit borders.
<code>add</code>	logical specifying if roc-area should be added to an already existing plot.
<code>...</code>	graphical parameters can be given as arguments to <code>' plot'</code> .

See Also

[plot.Daim](#), [auc.Daim](#)

Examples

```

data(Daim.data1)
data(Daim.data2)

perform1 <- performDaim(Daim.data1$prob.oob, Daim.data1$labels,
  Daim.data1$prob.app)
perform2 <- performDaim(Daim.data2$prob.oob, Daim.data2$labels,
  Daim.data2$prob.app)

summary(perform1)
summary(perform2)

roc.area(perform2)
roc.area(perform1, area.color=rgb(0,0,1,alpha=0.2), col="blue", add=TRUE)
legend(0.7,0.2,c("Model-1","Model-2"), col=c("red","blue"),lty=1,bg="white")

####
#### If your device don't support the semi-transparent colors use
#### the PDF Graphics Device.
####

pdf("ROC-area.pdf",version=1.4)
roc.area(perform2)
roc.area(perform1, area.color=rgb(0,0,1,alpha=0.2), col="blue", add=TRUE)
legend(0.7,0.2,c("Model 1","Model 2"), col=c("red","blue"),lty=1,bg="white")
dev.off()

```

summary.Daim

Summarizing a Daim Object

Description

summary method for class "Daim".

Usage

```

## S3 method for class 'Daim':
summary(object, ...)
## S3 method for class 'Daim.vector':
summary(object, ...)
## S3 method for class 'Daim.list':
summary(object, ...)

```

Arguments

`object` an object of class `Daim`.
`...` further arguments passed to or from other methods.

See Also

[Daim](#)

Examples

```
library(Daim)
data(Daim.data1)
perform <- performDaim(Daim.data1$prob.oob,
                      Daim.data1$labels, Daim.data1$prob.app)
perform
summary(perform)

data(Daim.data3)
head(Daim.data3)

M <- roc(Daim.data3[,2], Daim.data3[,1], labpos="pos")
summary(M)

M <- roc(Daim.data3[,-1], Daim.data3[,1], labpos="pos")
summary(M)
```

Index

*Topic **classif**

- auc, 1
- auc.Daim, 2
- Daim, 3
- Daim.control, 7
- performDaim, 9
- plot.Daim, 10
- plot.roc.Daim, 11
- roc, 13
- roc.area, 14

*Topic **datasets**

- Daim.data1, 8
- Daim.data3, 8

*Topic **hplot**

- plot.Daim, 10
- plot.roc.Daim, 11
- roc, 13
- roc.area, 14

*Topic **manip**

- auc, 1
- auc.Daim, 2
- print.Daim, 12
- summary.Daim, 15

*Topic **models**

- Daim, 3
- performDaim, 9

*Topic **print**

- print.Daim, 12

*Topic **programming**

- Daim, 3

- auc, 1
- auc (*auc.Daim*), 2
- auc.Daim, 2, 2, 5, 13, 15
- auc.numeric, 3

- Daim, 2, 3, 3, 7, 9, 10, 12, 13, 16
- Daim.control, 3, 6
- Daim.data1, 8
- Daim.data2 (*Daim.data1*), 8

- Daim.data3, 8

- legend, 12

- performDaim, 3, 5, 9
- plot, 10, 12
- plot.Daim, 3, 5, 9, 10, 13, 15
- plot.Daim.list (*plot.roc.Daim*), 11
- plot.Daim.vector (*plot.roc.Daim*), 11
- plot.roc.Daim, 11
- print.Daim, 12

- rgb, 10
- roc, 11, 13
- roc.area, 14
- roc.area.Daim, 2, 5, 10, 12
- roc.area.Daim.list (*roc.area*), 14
- roc.area.Daim.vector (*roc.area*), 14

- summary.Daim, 13, 15

- title, 10, 12, 14