Package ‘Daim’

February 19, 2015

Title Diagnostic accuracy of classification models.

Version 1.1.0

Author Sergej Potapov, Werner Adler, Benjamin Hofner and Berthold Lausen.

Description Several functions for evaluating the accuracy of classification models. The package provides the following performance measures: repeated k-fold cross-validation, 0.632 and 0.632+ bootstrap estimation of the misclassification rate, sensitivity, specificity and AUC. If an application is computationally intensive, parallel execution can be used to reduce the computational effort.

Depends R (>= 2.15.1), parallel

Imports rms

Suggests MASS, rpart, ipred, e1071, randomForest, mboost, TH.data

Date 02.09.2013

License GPL-2

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

Repository CRAN

Date/Publication 2013-09-13 14:18:09

NeedsCompilation yes

R topics documented:

Daim-package .......................................................... 2
auc ................................................................. 3
auc.Daim ........................................................... 4
Daim ................................................................. 5
Daim.Class ......................................................... 8
Daim.control ....................................................... 10
Daim.data 1 & 2 .................................................... 11
Daim.data 3 ......................................................... 12
deLong.test ......................................................... 12
performDaim ....................................................... 13
Daim-package

Several functions for evaluating the accuracy of classification models. The package provides the following performance measures: repeated k-fold cross-validation, 0.632 and 0.632+ bootstrap estimation of the misclassification rate, sensitivity, specificity and AUC. If an application is computationally intensive, parallel execution can be used to reduce the computational effort.

Details

Package: Daim
Type: Package
Version: 1.1-0
Date: 2013-10-09
License: GPL (>= 2)

See the help files for the following functions for more information:
Daim, performDaim, auc.Daim

Author(s)
Sergej Potapov, Werner Adler, Benjamin Hofner and Berthold Lausen
Maintainer: Sergej Potapov <sergej.potapov@gmail.com>

References
Werner Adler and Berthold Lausen (2009).
Bootstrap Estimated True and False Positive Rates and ROC Curve.

An introduction to ROC analysis.
Pattern Recognition Letters, 27, (8).
The area under the ROC curve (AUC)

Description

This function computes the area under an ROC curve.

Usage

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

Arguments

- `x` vector consisting of sensitivities of a classification procedure/diagnostic test for several cutpoints.
- `y` vector consisting of specificity of a classification procedure/diagnostic test for several cutpoints.
- `...` additional parameters.

Value

a scalar number, the area under the ROC curve.

See Also

`auc.Daim`, `roc.area.Daim`, `Daim`

Examples

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

###
### compute the 0.632+ estimation of the AUC.
###

auc(perform$roc$sens632p, perform$roc$spec632p)
```
The area under the ROC curve (AUC)

Description

This function computes the area under an ROC curve for Daim-class objects.

Usage

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

Arguments

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

Value

A list with following components:

- `auc.632p` the 0.632+ estimation of the AUC.
- `auc.632` the 0.632 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

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

auc(perform)  
roc.area(perform)
```
Diagnostic accuracy of classification models.

Description

Estimation of misclassification rate, sensitivity, specificity and AUC 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,
      multicore = FALSE, ...)

Arguments

formula formula of the form y ~ x1 + x2 + ..., where y must be a factor and
          x1, x2, ... are numeric or factor.
model function. Modelling technique whose error rate is to be estimated. The function
         model returns the predicted probability for each observation.
data an optional data frame containing the variables in the model (training data).
control See Daim.control.
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:
       "cv" - the optimal cut-point corresponding to cv estimation of the sensitivity
       and the specificity.
       "loob" - the optimal cut-point corresponding to loob estimation of the sensitiv-
       ity and the specificity.
       "0.632" - the optimal cut-point corresponding to 0.632 estimation of the sen-
       sitivity and the specificity.
       "0.632+" - the optimal cut-point corresponding to 0.632+ estimation of the
       sensitivity and the specificity.
labpos a character string of the response variable that defines a "positive" event. The
        labels of the "positive" events will be set to "pos" and others to "neg".
returnSample a logical value for saving the data from each sample.
cluster the name of the cluster, if parallel computing is used.
seed.cluster an integer value used as seed for the RNG.
multicore a logical indicating whether multiple cores (if available) should be used for the
          computations.
... additional parameters passed to clusterApplyLB or mclapply.

Value

An object of class Daim-class.
References

Werner Adler and Berthold Lausen (2009).
Bootstrap Estimated True and False Positive Rates and ROC Curve.

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

Improvements on cross-validation: The .632+ bootstrap method.

See Also

`plot.Daim`, `performDaim`, `auc.Daim`, `roc.area.Daim`

Examples

```r
# Evaluation of LDA

library(TH.data)
library(MASS)
data(GlaucomaM)
head(GlaucomaM)

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

set.seed(1102013)
ACC <- Daim(Class~., model=mylda, data=GlaucomaM, labpos="glaucoma",
            control=Daim.control(method="boot", number=50))

ACC
summary(ACC)

# Not run:
# just because of checking time on CRAN

####
#### optimal cut point determination
```
```
```r
set.seed(1102013)
ACC <- Daim(Class=., model=mlda, data=GlaucomaM, labpos="glaucoma",
control=Daim.control(method="boot", number=50), cutoff="0.632+")
ACC
summary(ACC)

###
### for parallel execution on multicore CPUs and computer clusters
###
library(parallel)
###
### create cluster with two slave nodes
cl <- makeCluster(2)
###
### Load used package on all slaves and execute Daim in parallel
###
clusterEvalQ(cl, library(ipred))
ACC <- Daim(Class=., model=mlda, data=GlaucomaM, labpos="glaucoma", cluster=cl)
ACC

###
### for parallel computing on multicore CPUs
###
ACC <- Daim(Class=., model=mlda, data=GlaucomaM, labpos="glaucoma", multicore=TRUE)
ACC

#########################################################################
## Evaluation of  ##
## randomForrest  ##
#########################################################################

library(randomForest)

myRF <- function(formula, train, test){
  model <- randomForest(formula, train)
  predict(model,test,type="prob")[,"pos"]
}
```
```r
daim_class <- Daim(Class~., model=myRF, data=GlaucomaM, labpos="glaucoma", control=Daim.control(number=50))
daim_class
summary(daim_class)

#####
##### optimal cut point determination
#####

set.seed(1102013)
daim_class <- Daim(Class~., model=myRF, data=GlaucomaM, labpos="glaucoma", control=Daim.control(method="boot", number=50), cutoff="0.632+")
daim_class
summary(daim_class)

#####
##### for parallel execution on multicore CPUs and computer clusters
#####

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

c1 <- makeCluster(2)
###
### Load used package on all slaves and execute Daim in parallel
###
clusterEvalQ(c1, library(randomForest))
daim_class <- Daim(Class~., model=myRF, data=GlaucomaM, labpos="glaucoma", cluster=c1)
daim_class

#####
##### for parallel computing on multicore CPUs
#####

daim_class <- Daim(Class~., model=myRF, data=GlaucomaM, labpos="glaucoma", multicore=TRUE)
daim_class

## End(Not run)
```
Description

Object includes the results of a performance evaluation from several cross-validation or bootstrapping runs.

Objects from the Class

Objects can be created by using the Daim or performDaim functions.

Slots

call: the matched call.
formula: the formula supplied.
method: the list of control parameters.
err632p: the 0.632+ estimation of the misclassification error.
err632: the 0.632 estimation of the misclassification error.
errloob: the LOOB estimation of the misclassification error.
errapp: the apparent error.
sens632p: the 0.632+ estimation of the sensitivity.
spec632p: the 0.632+ estimation of the specificity.
sens632: the 0.632 estimation of the sensitivity.
spec632: the 0.632 estimation of the specificity.
sensloob: the LOOB estimation of the sensitivity.
specloob: the LOOB estimation of the specificity.
sensapp: the apparent sensitivity.
specapp: the apparent specificity.
roc: a data frame with estimated values of sensitivity and specificity for a variety of cutoffs.
sample.roc: a list in which each entry contains the values of the ROC curve of this special sample or cross-validation run.
sample.data: a data frame with the results of this particular sample or cross-validation run.

See Also

Daim, performDaim, auc.Daim, roc.area.Daim
**Daim.control**

*Control parameters for the diagnostic accuracy of models.*

**Description**

Control of resampling methods.

**Usage**

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

**Arguments**

- `method` the resampling method:
  - `boot` - bootstrapping,
  - `cv` - k-fold cross-validation.
- `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 `replace = FALSE`). In this case subsampling is performed.
- `k` the number of folds
- `k.runs` the number of runs of k-fold cross-validations
- `dependency` to handle paired data structures

**References**


Ensemble classification of paired data. *Computational Statistics and Data Analysis*.  

**See Also**

- Daim

**Examples**

```r
###
### for bootstrap
###
```
Daim.control(method="boot", number=100)

###
### for cross-validation  
###
Daim.control(method="cv", k=10, k.runs=10)

###
### for subagging or subsampling  
###
Daim.control(method="boot", number=100, replace=FALSE, boot.size=0.9)

---

**Description**

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

**Usage**

data(Daim.data1)

data(Daim.data2)

**Format**

A three element list. The first element, `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, `prob.app` is a vector of numerical apparent predictions. The third list entry, `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)

```r
(perform <- performDaim(Daim.data1$prob.oob, Daim.data1$labels, Daim.data1$prob.app))
plot(perform)
```
Daim.data 3 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[,2], labpos="pos")
summary(M)

deLong.test DeLong test

Description
A nonparametric test for comparing AUC of two or more correlated ROC curves.

Usage
deLong.test(x, labels, labpos, ref=NULL, conf.level=0.95)

Arguments
x at least 2 dimensional matrix containing the observations of numeric predictors.
labels a vector containing the true class labels. It can be a factor or character vector.
labpos a character string of the variable labels that defines a "positive" event.
ref an index of reference predictors. Relevant only by pairwise comparisons.
conf.level confidence level of the interval.

References
Function to evaluate predictions

Perform the evaluation of predictions.

Usage

```r
# 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", ...)
```

Examples

```r
y <- rbinom(100, 1, .5)
x1 <- rnorm(100) + 1.5 * y
x2 <- rnorm(100) + .5 * y
x3 <- rnorm(100) + 2.5 * y
x <- data.frame(x1,x2,x3)
y <- as.factor(y)

(M1 <- deLong.test(x, labels=y, labpos="1"))

## Not run:
## just because of checking time on CRAN

data(iris)
(M2 <- deLong.test(iris[,1:4], labels=iris[,5], labpos="versicolor"))

## End(Not run)
```
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 cutoff set to:
  - "looob" - the optimal cut-point corresponding to loob estimation of the sensitivity and the specificity.
  - "0.632" - the optimal cut-point corresponding to 0.632 estimation of the sensitivity and the specificity.
  - "0.632+" - the optimal cut-point corresponding to 0.632+ estimation of the sensitivity and the specificity.
- **labpos**: a character string of the variable `labels` that defines a "positive" event.
- **...**: additional parameters.

See Also

- `Daim`, `plot.Daim`

Examples

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

---

**plot.Daim**

*Plotting method for Daim Objects*

Description

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

Usage

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

- **x**: an object of class Daim.
- **method**: kind of the estimation of the ROC curve: '0.632+', '0.632', '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.
- **legend**: add legend to plot.

... graphical parameters can be given as arguments to 'plot'.

See Also

Daim, roc.area.Daim

Examples

```r
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="0.632+", legend=TRUE)
plot(perform, method="sample")
plot(perform, method="0.632+", main="Comparison between methods")
plot(perform, method="0.632", col="blue", add=TRUE)
plot(perform, method="loob", col="green", add=TRUE)
legend("bottomright", c("0.632+","0.632","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.
###
###
## Not run:
pdf("plot-Daim.pdf")
plot(perform, method="0.632+")
plot(perform, method="sample")
plot(perform, method="0.632+", main="Comparison between methods")
```
plot(perform, method="0.632", col="blue", add=TRUE)
plot(perform, method="loo", col="green", add=TRUE)
legend("bottomright", c("0.632+","0.632","loo"),
       col=c("red","blue","green"), lty=1, inset=0.01)
plot(perform, all.roc=TRUE)
dev.off()

## End(Not run)

---

### plot.roc.Daim

**Plotting method for roc.Daim Objects**

**Description**

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

**Usage**

```r
## 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),
     lty=1, lwd=1, pch=19,
     xlab="False positive rate", ylab="True positive rate",
     main="ROC curves", legend=TRUE, ...)
```

**Arguments**

- **x**: an object of class `roc.Daim`.
- **color**: the color used to draw the ROC curve.
- **type**: what type of plot should be drawn: see argument `type` by the function `plot`.
- **bty**: the type of box to be drawn around the legend: see `legend`.
- **lty**: line type code: see `par`.
- **lwd**: line width: see `par`.
- **pch**: plotting 'character'; see `points`.
- **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`.
- **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)

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

```r
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

```r
## 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

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

See Also

- `plot.Daim, auc.Daim`
Examples

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

```r
## 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**

- **x**: an object of class Daim, Daim.list or Daim.vector.
- **method**: kind of the estimation of the ROC curve: '0.632+', '0.632', 'loob', 'cv', 'sample'
- **col**: the color used to draw the ROC curve.
- **area.color**: the color for filling the area.
- **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.
density the density of shading lines, in lines per inch. The default value of 'NULL' means that no shading lines are drawn. A zero value of 'density' means no shading nor filling whereas negative values (and 'NA') suppress shading (and so allow color filling).
angle the slope of shading lines, given as an angle in degrees (counter-clockwise).
border the color to draw the border. The default, 'NULL', means to use 'par("fg")'. Use 'border = NA' to omit borders.
add logical specifying if roc-area should be added to an already existing plot.

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")

## Not run:
## just because of checking time on CRAN

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

df("ROC-area.pdf")

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**

```r
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 **classes**

Daim Class, 8

*Topic **classif**

classif, 3

daaim class, 8

daaimMclass (daaim class), 8

daaimMpackage, 2

daim.control, 5, 10

daim.data 1 & 2, 11

daim.data 3, 12

daim.data1 (daim.data 1 & 2), 11

daim.data2 (daim.data 1 & 2), 11

daim.data3 (daim.data 3), 12

delong.test, 12

legend, 16

mclapply, 5

par, 16

performDaim, 2, 4, 6, 9, 13, 13

plot, 15, 16

plot.Daim, 4, 6, 13, 14, 14, 17, 18, 20

plot.Daim.list (plot.roc.Daim), 16

plot.Daim.vector (plot.roc.Daim), 16

plot.roc.Daim, 16

points, 16

print.Daim, 17

rgb, 15

roc, 13, 16, 18

roc.area, 19

roc.area.Daim, 3, 6, 9, 15, 17

summary.Daim, 17, 21

title, 15, 16, 20

*Topic **datasets**

Daim.data 1 & 2, 11

Daim.data 3, 12

*Topic **hplot**

plot.Daim, 14

plot.roc.Daim, 16

roc, 18

roc.area, 19

*Topic **manip**

auc, 3

daaim.Daim, 4

print.Daim, 17

summary.Daim, 21

*Topic **models**

Daim, 5

performDaim, 13

*Topic **print**

print.Daim, 17

*Topic **programming**

Daim, 5

auc, 3

auc (auc.Daim), 4

auc.Daim, 2, 3, 4, 6, 9, 18, 20

auc.numeric, 4

clusterApplyLB, 5