Package 'crossval'

July 8, 2015

Version 1.0.3

Date 2015-07-08

Title Generic Functions for Cross Validation

Author Korbinian Strimmer.

Maintainer Korbinian Strimmer <strimmerlab@gmail.com>

Depends R (>= 3.0.2)

Imports stats

Suggests MASS, sda, care, binda

Description Contains generic functions for performing cross validation and for computing diagnostic errors.

License GPL (>= 3)

URL http://cran.r-project.org/package=crossval

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-08 13:21:53

R topics documented:

crossval-package 2
confusionMatrix 2
crossval 3
diagnosticErrors 6

Index **8**

crossval-package *The crossval Package*

Description

The "crossval" package implements generic functions for performing cross validation and for computing diagnostic errors.

Author(s)

Korbinian Strimmer (http://strimmerlab.org/)

References

Website: http://cran.r-project.org/package=crossval

See Also

crossval, confusionMatrix, diagnosticErrors.

confusionMatrix *Compute Confusion Matrix*

Description

confusionMatrix computes the confusion matrix, i.e. it counts the number of false positives (FP), true positives (TP), true negatives (TN), and false negatives (FN).

Despite its name the functions returns a vector rather than an actual matrix for easier use with the crossval function.

Usage

```
confusionMatrix(actual, predicted, negative="control")
```

Arguments

actual	a vector containing the actual correct labels for each sample (e.g. "cancer" or "control").
predicted	a vector containing the predicted labels.
negative	the label of a negative "null" sample (default: "control").

Value

confusionMatrix returns a vector of length 4 containing the counts for FP, TP, TN, and FN.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

diagnosticErrors.

Examples

```
# load crossval library
library("crossval")

# true labels
a = c("cancer", "cancer", "control", "control", "cancer", "control", "control")

# predicted labels
p = c("cancer", "control", "control", "control", "cancer", "control", "cancer")

# confusion matrix (a vector)
cm = confusionMatrix(a, p, negative="control")
cm
# FP TP TN FN
# 1  2  3  1
# attr(,"negative")
# [1] "control"

# corresponding accuracy, sensitivity etc.
diagnosticErrors(cm)
#       acc      sens      spec       ppv       npv       lor
# 0.7142857 0.6666667 0.7500000 0.6666667 0.7500000 1.7917595
# attr(,"negative")
# [1] "control"
```

---

crossval *Generic Function for Cross Valdidation*

Description

crossval performs K-fold cross validation with B repetitions. If Y is a factor then balanced sampling is used (i.e. in each fold each category is represented in appropriate proportions).

Usage

```
crossval(predfun, X, Y, K=10,  B=20, verbose=TRUE, ...)
```

Arguments

predfun	Prediction function (see details).
X	Matrix of predictors (columns correspond to variables).
Y	Univariate response variable.
K	Number of folds.
B	Number of repetitions.
verbose	If verbose=TRUE then status messages appear during cross validation.
...	optional arguments for predfun

Details

The argument predfun must be a function of the form predfun(Xtrain, Ytrain, Xtest, Ytest, ...).

Value

crossval returns a list with three entries:

stat.cv: the statistic returned by predfun for each cross validation run.

stat: the statistic returned by predfun averaged over all cross validation runs.

stat.se: the corresponding standard error.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

confusionMatrix.

Examples

```
# load "crossval" package
library("crossval")

# classification examples

# set up lda prediction function
predfun.lda = function(train.x, train.y, test.x, test.y, negative)
{
  require("MASS") # for lda function

  lda.fit = lda(train.x, grouping=train.y)
  ynew = predict(lda.fit, test.x)$class

  # count TP, FP etc.
  out = confusionMatrix(test.y, ynew, negative=negative)

  return( out )
}
```

```
# Student's Sleep Data
data(sleep)
X = as.matrix(sleep[,1, drop=FALSE]) # increase in hours of sleep
Y = sleep[,2] # drug given
plot(X ~ Y)
levels(Y) # "1" "2"
dim(X) # 20  1

set.seed(12345)
cv.out = crossval(predfun.lda, X, Y, K=5, B=20, negative="1")

cv.out$stat
# FP   TP   TN   FN
#0.60 1.08 1.40 0.92

diagnosticErrors(cv.out$stat)
#     acc      sens      spec       ppv       npv       lor
#0.6200000 0.5400000 0.7000000 0.6428571 0.6034483 1.0076405


## Not run:

# Wine Data - see http://archive.ics.uci.edu/ml/datasets/Wine for details
wine.data = read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data",
          sep=",")
c = wine.data[,1]
X = as.matrix(wine.data[c!=3,-1])
Y = as.factor(c[c!=3])
dim(X) # 130  13
levels(Y) # "1", "2"

set.seed(12345)
cv.out = crossval(predfun.lda, X, Y, K=5, B=50, negative="1")

cv.out$stat
#   FP     TP     TN     FN
# 0.028 14.092 11.772  0.108

diagnosticErrors(cv.out$stat)
#      acc       sens       spec        ppv        npv        lor
# 0.9947692  0.9923944  0.9976271  0.9980170  0.9909091 10.9125059

## End(Not run)


# linear regression example

data("attitude")
y = attitude[,1] # rating variable
x = attitude[,-1] # date frame with the remaining variables
is.factor(y) # FALSE
```

```
summary( lm(y ~ . , data=x) )

# set up lm prediction function
predfun.lm = function(train.x, train.y, test.x, test.y)
{
  lm.fit = lm(train.y ~ . , data=train.x)
  ynew = predict(lm.fit, test.x )

  # compute squared error risk (MSE)
  out = mean( (ynew - test.y)^2 )

  return( out )
}


# prediction MSE using all variables
set.seed(12345)
cv.out = crossval(predfun.lm, x, y, K=5, B=20)
c(cv.out$stat, cv.out$stat.se) # 68.06480  2.91447

# and only two variables
cv.out = crossval(predfun.lm, x[,c(1,3)], y, K=5, B=20)
c(cv.out$stat, cv.out$stat.se) # 52.855325  1.878877



# for more examples (e.g. using cross validation in a regression or classification context)
# see the R packages "sda", "care", or "binda".
```

`diagnosticErrors` *Compute Diagnostic Errors: Accuracy, Sensitivity, Specificity, Positive Predictive Value, Negative Predictive Value, Log Odds Ratio*

Description

`diagnosticErrors` computes various diagnostic errors useful for evaluating the performance of a diagnostic test or a classifier: accuracy (acc), sensitivity (sens), specificity (spec), positive predictive value (ppv), negative predictive value (npv), and log-odds ratio (lor).

Usage

```
diagnosticErrors(cm)
```

Arguments

`cm`	a vector containing the true positives, false positives etc, as computed by `confusionMatrix`.

Details

The diagnostic errors are computed as follows:

acc = (TP+TN)/(FP+TN+TP+FN)

sens = TP/(TP+FN)

spec = TN/(FP+TN)

ppv = TP/(FP+TP)

npv = TN/(TN+FN)

lor = log(TP*TN/(FN*FP))

Value

`diagnostic errors` returns a vector containing various diagnostic errors.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

confusionMatrix.

Examples

```
# load crossval library
library("crossval")

# true labels
a = c("cancer", "cancer", "control", "control", "cancer", "control", "control")

# predicted labels
p = c("cancer", "control", "control", "control", "cancer", "control", "cancer")

# confusion matrix (a vector)
cm = confusionMatrix(a, p, negative="control")
cm
# FP TP TN FN
# 1  2  3  1
# attr(,"negative")
# [1] "control"

# corresponding accuracy, sensitivity etc.
diagnosticErrors(cm)
#       acc      sens      spec       ppv       npv       lor
# 0.7142857 0.6666667 0.7500000 0.6666667 0.7500000 1.7917595
# attr(,"negative")
# [1] "control"
```

Index

∗Topic **multivariate**
crossval, 3
crossval-package, 2
∗Topic **univar**
confusionMatrix, 2
diagnosticErrors, 6

confusionMatrix, *2*, 2, 2, *4*, *6*, *7*
crossval, *2*, 3
crossval-package, 2

diagnosticErrors, *2*, *3*, 6