Package 'cvAUC'

May 8, 2026

Type Package

Title Cross-Validated Area Under the ROC Curve Confidence Intervals

Version 1.1.4

Date 2022-01-17

Author Erin LeDell, Maya Petersen, Mark van der Laan

Maintainer Erin LeDell <oss@ledell.org>

Description

Tools for working with and evaluating cross-validated area under the ROC curve (AUC) estimators. The primary functions of the package are ci.cvAUC and ci.pooled.cvAUC, which report cross-validated AUC and compute confidence intervals for cross-validated AUC estimates based on influence curves for i.i.d. and pooled repeated measures data, respectively. One benefit to using influence curve based confidence intervals is that they require much less computation time than bootstrapping methods. The utility functions, AUC and cvAUC, are simple wrappers for functions from the ROCR package.

License Apache License (== 2.0)

Imports ROCR, data.table

URL https://github.com/ledell/cvAUC

BugReports https://github.com/ledell/cvAUC/issues

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2022-01-17 22:42:41 UTC

Contents

---

cvAUC-package *Cross-Validated Area Under the ROC Curve Confidence Intervals*

Description

Tools for working with and evaluating cross-validated area under the ROC curve (AUC) estimators. The primary functions of the package are `ci.cvAUC` and `ci.pooled.cvAUC`, which compute confidence intervals for cross-validated AUC estimates based on influence curves of both regular i.i.d and pooled repeated measures data. One benefit to using influence function based confidence intervals is that they require much less computation time than bootstrapping methods. The utility function, `cvAUC`, which computes cross-validated AUC, is a wrapper for functions from the ROCR package.

Details

Package:	cvAUC
Type:	Package
Version:	1.1.4
Date:	2022-01-17
License:	Apache License (== 2.0)

See the help files for the following functions for more information:

`cvAUC`, `ci.cvAUC`, `ci.pooled.cvAUC`

Note

This work was supported by the Doris Duke Charitable Foundation Grant No. 2011042.

Author(s)

Erin LeDell, Maya Petersen, Mark van der Laan

Maintainer: Erin LeDell <oss@ledell.org>

References

LeDell, Erin; Petersen, Maya; van der Laan, Mark. Computationally efficient confidence intervals for cross-validated area under the ROC curve estimates. Electron. J. Statist. 9 (2015), no. 1, 1583–1607. doi:10.1214/15-EJS1035. http://projecteuclid.org/euclid.ejs/1437742107.

M. J. van der Laan and S. Rose. Targeted Learning: Causal Inference for Observational and Experimental Data. Springer Series in Statistics. Springer, first edition, 2011.

Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. ROCR: Visualizing classifier performance in R. Bioinformatics, 21(20):3940-3941, 2005.

See Also

https://cran.r-project.org/package=ROCR

---

adherence *Data set: Simulated Pooled Repeated Measures Data*

Description

This is a simulated pooled repeated measures data set of patient medication adherence summaries with a binary outcome. The independent units are the patients, and each patient has one or more measurements made at different time points in treatment, each associated with a binary outcome that could represent a diagnostic test result. This data set is meant to be used with the `ci.pooled.cvAUC` function.

Usage

```
data(adherence)
```

Format

A data frame. The first column, `id`, is the patient id. Columns 2-5 represent medication adherence (as a percentage) averaged over the past 2, 7, 14, 21 and 28 days, respectively. The last column is a binary outcome that could represent a test result.

See Also

`ci.pooled.cvAUC`

---

admissions *Data set: Simulated Admissions Data with Binary Outcome*

Description

This is a simulated data set that represents admissions information for a graduate program in the sciences. The binary outcome is 1 for admitted and 0 for not admitted. This data set is meant to be used with the `ci.cvAUC` function.

Usage

```
data(admissions)
```

Format

A data frame. The five predictor variables are: `quant`, `verbal`, `gpa`, `toptier` and `research`. We can treat `quant` and `verbal`, which represent quantitative and verbal GRE scores, as continuous variables. The binary indicator variables, `toptier` and `research`, indicate whether the application is coming from a "top tier" institution and whether or not they have prior research experience. The binary indicator, `Y`, represents admitted `(Y=1)` vs. not admitted `(Y=0)`.

AUC *Area Under the ROC Curve*

Description

This function calculates Area Under the ROC Curve (AUC). The AUC can be defined as the probability that the fit model will score a randomly drawn positive sample higher than a randomly drawn negative sample. This is also equal to the value of the Wilcoxon-Mann-Whitney statistic. This function is a wrapper for functions from the ROCR package.

Usage

```
AUC(predictions, labels, label.ordering = NULL)
```

Arguments

`predictions`	A vector of predictions, or predicted probabilities, for each observation.
`labels`	A binary vector containing the true values for each observation. Must have the same length as `predictions`.
`label.ordering`	The default ordering of the classes can be changed by supplying a vector containing the negative and the positive class label (negative label first, positive label second).

Value

The value returned is the Area Under the ROC Curve (AUC).

Author(s)

Erin LeDell <oss@ledell.org>

References

References to the underlying ROCR code, used to calculate area under the ROC curve, can be found on the ROCR homepage at: https://ipa-tys.github.io/ROCR/

See Also

`prediction`, `performance`, `cvAUC`, `ci.cvAUC`, `ci.pooled.cvAUC`

Examples

```
library(cvAUC)
library(ROCR)  #load example data

data(ROCR.simple)
auc <- AUC(ROCR.simple$predictions, ROCR.simple$labels)
# [1] 0.8341875
```

`ci.cvAUC` *Confidence Intervals for Cross-validated Area Under the ROC Curve (AUC) Estimates*

Description

This function calculates influence curve based confidence intervals for cross-validated area under the ROC curve (AUC) estimates.

Usage

```
ci.cvAUC(predictions, labels, label.ordering = NULL, folds = NULL, confidence = 0.95)
```

Arguments

`predictions`	A vector, matrix, list, or data frame containing the predictions.
`labels`	A vector, matrix, list, or data frame containing the true class labels. Must have the same dimensions as `predictions`.
`label.ordering`	The default ordering of the classes can be changed by supplying a vector containing the negative and the positive class label (negative label first, positive label second).
`folds`	If specified, this must be a vector of fold ids equal in length to `predictions` and `labels`, or a list of length V (for V-fold cross-validation) of vectors of indexes for the observations contained in each fold. The `folds` argument must only be specified if the `predictions` and `labels` arguments are vectors.
`confidence`	A number between 0 and 1 that represents confidence level.

Details

See the documentation for the `prediction` function in the ROCR package for details on the `predictions`, `labels` and `label.ordering` arguments.

Value

A list containing the following named elements:

`cvAUC`	Cross-validated area under the curve estimate.
`se`	Standard error.
`ci`	A vector of length two containing the upper and lower bounds for the confidence interval.
`confidence`	A number between 0 and 1 representing the confidence.

Author(s)

Erin LeDell <oss@ledell.org>

Maya Petersen <mayaliv@berkeley.edu>

Mark van der Laan <laan@berkeley.edu>

References

LeDell, Erin; Petersen, Maya; van der Laan, Mark. Computationally efficient confidence intervals for cross-validated area under the ROC curve estimates. Electron. J. Statist. 9 (2015), no. 1, 1583–1607. doi:10.1214/15-EJS1035. http://projecteuclid.org/euclid.ejs/1437742107.

M. J. van der Laan and S. Rose. Targeted Learning: Causal Inference for Observational and Experimental Data. Springer Series in Statistics. Springer, first edition, 2011.

Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. ROCR: Visualizing classifier performance in R. Bioinformatics, 21(20):3940-3941, 2005.

See Also

prediction, performance, cvAUC, ci.pooled.cvAUC

Examples

```
# This i.i.d. data example does the following:

# Load a data set with a binary outcome. For the i.i.d. case we use a simulated data set of
# 500 observations, included with the package, of graduate admissions data.
#
# Divide the indices randomly into 10 folds, stratifying by outcome.  Stratification is not
# necessary, but is commonly performed in order to create validation folds with similar
# distributions.  Store this information in a list called folds.
#
# Define a function to fit a model on the training data and to generate predicted values
# for the observations in the validation fold, for a single iteration of the cross-validation
# procedure.  We use a logistic regression fit.
#
# Apply this function across all folds to generate predicted values for each validation fold.
# The concatenated version of these predicted values is stored in vector called predictions.
# The outcome vector, Y, is the labels argument.


iid_example <- function(data, V = 10){

  .cvFolds <- function(Y, V){  #Create CV folds (stratify by outcome)
    Y0 <- split(sample(which(Y==0)), rep(1:V, length = length(which(Y==0))))
    Y1 <- split(sample(which(Y==1)), rep(1:V, length = length(which(Y==1))))
    folds <- vector("list", length=V)
    for (v in seq(V)) {folds[[v]] <- c(Y0[[v]], Y1[[v]])}
    return(folds)
  }
  .doFit <- function(v, folds, data){  #Train/test glm for each fold
    fit <- glm(Y~., data = data[-folds[[v]],], family = binomial)
    pred <- predict(fit, newdata = data[folds[[v]],], type = "response")
    return(pred)
  }
  folds <- .cvFolds(Y = data$Y, V = V)  #Create folds
 predictions <- unlist(sapply(seq(V), .doFit, folds = folds, data = data)) #CV train/predict
  predictions[unlist(folds)] <- predictions  #Re-order pred values
  # Get CV AUC and confidence interval
```

```
  out <- ci.cvAUC(predictions = predictions, labels = data$Y,
                  folds = folds, confidence = 0.95)
  return(out)
}


# Load data
library(cvAUC)
data(admissions)

# Get performance
set.seed(1)
out <- iid_example(data = admissions, V = 10)


# The output is given as follows:
# > out
# $cvAUC
# [1] 0.9046473
#
# $se
# [1] 0.01620238
#
# $ci
# [1] 0.8728913 0.9364034
#
# $confidence
# [1] 0.95
```

`ci.pooled.cvAUC` *Confidence Intervals for Cross-validated Area Under the ROC Curve (AUC) Estimates for Pooled Repeated Measures Data*

Description

This function calculates influence curve based confidence intervals for cross-validated area under the curve (AUC) estimates, for a pooled repeated measures data set.

Usage

```
ci.pooled.cvAUC(predictions, labels, label.ordering = NULL,
  folds = NULL, ids, confidence = 0.95)
```

Arguments

`predictions`	A vector, matrix, list, or data frame containing the predictions.
`labels`	A vector, matrix, list, or data frame containing the true class labels. Must have the same dimensions as `predictions`.

label.ordering	The default ordering of the classes can be changed by supplying a vector containing the negative and the positive class label (negative label first, positive label second).
folds	If specified, this must be a vector of fold ids equal in length to predictions and labels, or a list of length V (for V-fold cross-validation) of vectors of indexes for the observations contained in each fold. The folds argument must only be specified if the predictions and labels arguments are vectors.
ids	A vector, matrix, list, or data frame containing cluster or entity ids. All observations from the same entity (i.e. patient) that have been pooled must have the same id. Must have the same dimensions as 'predictions'.
confidence	A number between 0 and 1 that represents confidence level.

Details

See the documentation for the prediction function in the ROCR package for details on the predictions, labels and label.ordering arguments.

In pooled repeated measures data, the clusters (not the individual observations) are the independent units. Each observation has a corresponding binary outcome. This data structure arises often in clinical studies where each patient is measured, and an outcome is recorded, at various time points. Then the observations from all patients are pooled together. See the Examples section below for more information.

Value

A list containing the following named elements:

cvAUC	Cross-validated area under the curve estimate.
se	Standard error.
ci	A vector of length two containing the upper and lower bounds for the confidence interval.
confidence	A number between 0 and 1 representing the confidence.

Author(s)

Erin LeDell <oss@ledell.org>

Maya Petersen <mayaliv@berkeley.edu>

Mark van der Laan <laan@berkeley.edu>

References

LeDell, Erin; Petersen, Maya; van der Laan, Mark. Computationally efficient confidence intervals for cross-validated area under the ROC curve estimates. Electron. J. Statist. 9 (2015), no. 1, 1583–1607. doi:10.1214/15-EJS1035. http://projecteuclid.org/euclid.ejs/1437742107.

M. J. van der Laan and S. Rose. Targeted Learning: Causal Inference for Observational and Experimental Data. Springer Series in Statistics. Springer, first edition, 2011.

Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. ROCR: Visualizing classifier performance in R. Bioinformatics, 21(20):3940-3941, 2005.

See Also

prediction, performance, cvAUC, ci.cvAUC

Examples

```
# This example is similar to the ci.cvAUC example, with the excpection that
# this is a pooled repeated measures data set.  The example uses simulated
# data that contains multiple time point observations for 500 patients,
# each observation having a binary outcome.
#
# The cross-validation folds are stratified by ids that have at least one
# positive outcome.  All observations belonging to one patient are
# contained within the save CV fold.


pooled_example <- function(data, ids, V = 10){

  .cvFolds <- function(Y, V, ids){
    #Stratify by outcome & id
    classes <- tapply(1:length(Y), INDEX = Y, FUN = split, 1)
    ids.Y1 <- unique(ids[classes$`1`])  #ids that contain an observation with Y==1
    ids.noY1 <- setdiff(unique(ids), ids.Y1)  #ids that have no Y==1 obvervations
    ids.Y1.split <- split(sample(length(ids.Y1)), rep(1:V, length = length(ids.Y1)))
   ids.noY1.split <- split(sample(length(ids.noY1)), rep(1:V, length = length(ids.noY1)))
    folds <- vector("list", V)
    for (v in seq(V)){
      idx.Y1 <- which(ids %in% ids.Y1[ids.Y1.split[[v]]])
      idx.noY1 <- which(ids %in% ids.noY1[ids.noY1.split[[v]]])
      folds[[v]] <- c(idx.Y1, idx.noY1)
    }
    return(folds)
  }
  .doFit <- function(v, folds, data){  #Train/test glm for each fold
    fit <- glm(Y~., data = data[-folds[[v]],], family = binomial)
    pred <- predict(fit, newdata = data[folds[[v]],], type = "response")
    return(pred)
  }
  folds <- .cvFolds(Y = data$Y, ids = ids, V = V)  #Create folds
 predictions <- unlist(sapply(seq(V), .doFit, folds = folds, data = data)) #CV train/predict
  predictions[unlist(folds)] <- predictions  #Re-order fold indices
  out <- ci.pooled.cvAUC(predictions = predictions, labels = data$Y,
                         folds = folds, ids = ids, confidence = 0.95)
  return(out)
}


# Load data
library(cvAUC)
data(adherence)


# Get performance
```

```
set.seed(1)
out <- pooled_example(data = subset(adherence, select=-c(id)),
                      ids = adherence$id, V = 10)


# The output is given as follows:
# > out
# $cvAUC
# [1] 0.8648046
#
# $se
# [1] 0.01551888
#
# $ci
# [1] 0.8343882 0.8952211
#
# $confidence
# [1] 0.95
```

---

cvAUC *Cross-validated Area Under the ROC Curve (AUC)*

Description

This function calculates cross-validated area under the ROC curve (AUC) esimates. For each fold, the empirical AUC is calculated, and the mean of the fold AUCs is the cross-validated AUC estimate. The area under the ROC curve is equal to the probability that the classifier will score a randomly drawn positive sample higher than a randomly drawn negative sample. This function is a simple wrapper for the AUC functionality inside the ROCR package.

Usage

```
cvAUC(predictions, labels, label.ordering = NULL, folds = NULL)
```

Arguments

`predictions`	A vector, matrix, list, or data frame containing the predictions.
`labels`	A vector, matrix, list, or data frame containing the true class labels. Must have the same dimensions as `predictions`.
`label.ordering`	The default ordering of the classes can be changed by supplying a vector containing the negative and the positive class label (negative label first, positive label second).
`folds`	If specified, this must be a vector of fold ids equal in length to `predictions` and `labels`, or a list of length V (for V-fold cross-validation) of vectors of indexes for the observations contained in each fold. The `folds` argument must only be specified if the `predictions` and `labels` arguments are vectors.

Details

If predictions and labels are provided as vectors and folds is NULL, then this function will return AUC (not cross-validated). See the documentation for the prediction function in the ROCR package for details on the predictions, labels and label.ordering arguments.

Value

perf	An object of class 'performance' from the ROCR package. Can be used to plot the ROC curve.
fold.AUC	A vector containing the AUC estimate for each fold.
cvAUC	Cross-validated area under the curve.

Author(s)

Erin LeDell <oss@ledell.org>

References

Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. ROCR: Visualizing classifier performance in R. Bioinformatics, 21(20):3940-3941, 2005.

See Also

prediction, performance, ci.cvAUC, ci.pooled.cvAUC

Examples

```
# Example of how to get CV AUC and plot the curve.
library(cvAUC)
library(ROCR)  #load example data

data(ROCR.xval)
out <- cvAUC(ROCR.xval$predictions, ROCR.xval$labels)

#Plot fold AUCs
plot(out$perf, col = "grey82", lty = 3, main = "10-fold CV AUC")

#Plot CV AUC
plot(out$perf, col ="red", avg = "vertical", add = TRUE)

# See the ci.cvAUC documentation for an example
# of how to use the `folds` argument.
```

Index