

Package ‘HandTill2001’

October 3, 2025

Type Package
Title Multiple Class Area under ROC Curve
Version 1.0.3
Description An S4 implementation of Eq. (3) and Eq. (7) by
David J. Hand and Robert J. Till (2001) <[DOI:10.1023/A:1010920819831](https://doi.org/10.1023/A:1010920819831)>.
License BSD_2_clause + file LICENSE
URL <https://gitlab.com/fvafrcu/HandTill2001>
Depends R (>= 2.14)
Imports methods, utils
Suggests caTools, devtools, knitr, MASS, mda, nnet, pkgload,
rmarkdown, ROCR, rpart, rprojroot, RUnit, testthat
VignetteBuilder utils
RoxygenNote 7.3.2
NeedsCompilation no
Author Andreas Dominik Cullmann [aut, cre]
Maintainer Andreas Dominik Cullmann <fvafrcu@mailbox.org>
Repository CRAN
Date/Publication 2025-10-03 05:20:25 UTC

Contents

HandTill2001-package	2
auc-methods	3
bincap	4
bincap-class	5
ht01.multipleclass	6
ht01.twoclass	7
multcap	8
multcap-class	9
Index	10

HandTill2001-package *Multiple Class Area under ROC Curve*

Description

A very lean package implementing merely M given by *Hand and Till (2001)*, Eq. (7).

Details

M given by *Hand and Till (2001)* defines a multiple class version of the area under curve of the receiver operating characteristic.

Author(s)

Maintainer: Andreas Dominik Cullmann <fvafrcu@mailbox.org>

References

David J. Hand and Robert J. Till (2001). *A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems*. *Machine Learning* **45**(2), p. 171–186. DOI: [doi:10.1023/A:1010920819831](https://doi.org/10.1023/A:1010920819831).

See Also

`help(package="HandTill2001")`, especially `?HandTill2001::auc`; various packages that calculate binary class AUC ([ROCR](#)) or multiple class AUC ([pROC](#), [caTools](#)).

Examples

```
library(HandTill2001)
data(ht01.multipleclass)
auc(
  multcap(
    response = ht01.multipleclass$observed,
    predicted = as.matrix(ht01.multipleclass[, levels(ht01.multipleclass$observed)])
  )
)
```

Description

Calculate area under curve of the receiver operating characteristic for two or more prediction classes.

Usage

```
## S4 method for signature 'bincap'
auc(object)

## S4 method for signature 'multcap'
auc(object)
```

Arguments

object An object of class *bincap* or *multcap*.

Details

Depending on whether object is of class *bincap* or of class *multcap*, a binary class or multiple class AUC is calculated.

Value

An object of class "numeric".

Methods

signature(object = "bincap") calculates the AUC statistic for a binary class response following *Hand and Till (2001)*, Eq. (3).

signature(object = "multcap") calculates the AUC statistic for a multiple class response following *Hand and Till (2001)*, Eq. (7).

References

David J. Hand and Robert J. Till (2001). A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning* **45**(2), p. 171–186. DOI: [doi:10.1023/A:1010920819831](https://doi.org/10.1023/A:1010920819831).

See Also

[class?bincap](#), [class?multcap](#)

Examples

```
data(ht01.twoclass, package = "HandTill2001")
message(" == AUC for a binary class response")
message(" == HandTill2001 result:")
HandTill2001::auc(HandTill2001::bincap(
  response = as.factor(ht01.twoclass[["observed"]]),
  predicted = ht01.twoclass[["predicted"]],
  true = "1"
))
## Not run:
message(" == ROCR result:")
ROCR::performance(ROCR::prediction(
  labels = ht01.twoclass[["observed"]],
  predictions = ht01.twoclass[["predicted"]]
),
measure = "auc"
)@y.values

## End(Not run)
data(ht01.multipleclass, package = "HandTill2001")
message(" == AUC for a multiple class response")
predicted <- as.matrix(ht01.multipleclass[, levels(ht01.multipleclass[["observed"]])])
HandTill2001::auc(HandTill2001::multcap(
  response = ht01.multipleclass[["observed"]],
  predicted = predicted
))
```

bincap

A Constructor for Objects of Class bincap

Description

bincap(...) is an alias to new("bincap", ...).

Usage

```
bincap(response, predicted, true = "1")
```

Arguments

response	Object of class factor.
predicted	Object of class numeric.
true	Object of class character.

Details

There is no casting or conversion of data. bincap(...) is just an alias to new("bincap", ...).

Value

An object of class bincap.

See Also

[class?HandTill2001::bincap](#)

Examples

```
library(HandTill2001)
data(ht01.twoclass)
str(ht01.twoclass$observed)
message("note that ht01.twoclass$observed is not a factor; we have to convert it.")
bincap(
  response = as.factor(ht01.twoclass$observed),
  predicted = ht01.twoclass$predicted,
  true = c("1")
)
```

bincap-class

Binary Class and Prediction Objects

Description

S4 class for a binary class response and corresponding (predicted) probabilities.

Objects from the Class

Objects can be created by calls of the form `new("bincap", ...)`. They are used to store a binary class response (one of the two levels of which is supposed to be true), the information which of the two levels of the binary class response is thought of as 'true'/'positive'/'present' (the other one would then be thought of as 'false'/'negative'/'absence') and the predicted probabilities that response is true.

Note

No defaults are set. Especially, you have to explicitly initialize `true`, there is no trying to guess it from the levels of response.

See Also

[class?HandTill2001::cap](#), [class?HandTill2001::multcap](#), [?HandTill2001::bincap](#)

Examples

```
showClass("bincap")
```

ht01.multipleclass *Example Data for Multiple Classes*

Description

Multiple class data and probability predictions thereof.

Format

A data frame with 214 observations on the following 7 variables.

observed a factor with levels Con Head Tabl Veh WinF WinNF

WinF a numeric vector

WinNF a numeric vector

Veh a numeric vector

Con a numeric vector

Tabl a numeric vector

Head a numeric vector

Details

Multiple class data ('observed': MASS::fgl\$type) and probability predictions (predict(fgl.rp4), cf. Venables and Ripley (2002), p. 264 and 'Source') from rpart::rpart.

Source

```
## From: Forensic glass example Venables and Ripley
(2002) pp. 261--265 library(MASS);library(rpart);data(fgl);set.seed(123)
fgl.rp4 <- rpart(type ~ ., data = fgl, cp = 0.03 , parms = list(split =
"information")) ht01.multipleclass <- data.frame(observed = fgl$type,
predict(fgl.rp4)) write.table(ht01.multipleclass, file =
"ht01.multipleclass.txt")
```

References

Venables, W. N and Ripley, B. D. (2002), *Modern Applied Statistics with S* (4th edition). Springer, ISBN 0-387-95457-0

Examples

```
library(HandTill2001)
data(ht01.multipleclass)
str(ht01.multipleclass)
```

ht01.twoclass*Example Data for Binary Classes*

Description

Binary class data and probability predictions thereof.

Format

A data frame with 189 observations on the following 2 variables.

observed a numeric vector

predicted a numeric vector

Details

Binary class data ('observed': MASS::birthwt\$low) and probability predictions (predict(birthwt.step2, type = "response")), cf. Venables and Ripley (2002), pp. 195f and 'Source') from stats::glm.

Source

```
## From: A binary class data example Venables and
Ripley pp. 194--199 library(MASS); data("birthwt"); attach(birthwt) race <-
(factor(race, labels = c("white", "black", "other"))) ptd <- factor(ptl > 0)
ftv <- factor(ftv) levels(ftv)[-1:2] <- "2+" bwt <- data.frame(low =
factor(low), age, lwt, race, smoke = (smoke > 0) , ptd, ht = (ht > 0), ui =
(ui > 0), ftv) detach(birthwt) birthwt.glm <- glm(low ~ .,
family=binomial(link=logit), data=bwt) birthwt.step2 <- stepAIC(birthwt.glm,
~ .^2 + I(scale(age)^2) + I(scale(lwt)^2), trace = F ) ht01.twoclass <-
data.frame(observed = bwt$low , predicted = predict(birthwt.step2 , type =
"response")) write.table(ht01.twoclass, file = "ht01.twoclass.txt")
```

References

Venables, W. N and Ripley, B. D. (2002), *Modern Applied Statistics with S* (4th edition). Springer, ISBN 0-387-95457-0

Examples

```
library(HandTill2001)
data(ht01.twoclass)
str(ht01.twoclass)
```

multcap

A Constructor for Objects of Class multcap

Description

multcap(...) is an alias to new("multcap", ...).

Usage

```
multcap(response, predicted)
```

Arguments

response	Object of class factor.
predicted	Object of class matrix.

Details

There is no casting or conversion of data. multcap(...) is just an alias to new("multcap", ...).

Value

An object of class multcap.

See Also

[class?HandTill2001::multcap](#)

Examples

```
library(HandTill2001)
data(ht01.multipleclass)
str(ht01.multipleclass$observed)
message("note that ht01.multipleclass$observed is a factor; we do not have to convert it.")
multcap(
  response = ht01.multipleclass$observed,
  predicted = as.matrix(ht01.multipleclass[, levels(ht01.multipleclass$observed)])
)
```

`multcap-class`*Multiple Class and Prediction Objects*

Description

S4 class for a multiple class response and corresponding (predicted) probabilities.

Objects from the Class

Objects can be created by calls of the form `new("multcap", ...)`. They are used to store a multiple class response and the predicted probabilities for each of the `levels(response)`.

See Also

`class?HandTill2001::cap`, `class?HandTill2001::bincap`, `?HandTill2001::multcap`

Examples

```
showClass("multcap")
```

Index

- * **AUC**
 - HandTill2001-package, [2](#)
- * **ROC**
 - HandTill2001-package, [2](#)
- * **auc**
 - auc-methods, [3](#)
- * **classes**
 - bincap-class, [5](#)
 - multcap-class, [9](#)
- * **datasets**
 - ht01.multipleclass, [6](#)
 - ht01.twoclass, [7](#)
- * **methods**
 - auc-methods, [3](#)
- * **ui-constructor**
 - bincap, [4](#)
 - multcap, [8](#)
- ?HandTill2001::auc, [2](#)
- ?HandTill2001::bincap, [5](#)
- ?HandTill2001::multcap, [9](#)

- auc (auc-methods), [3](#)
- auc, bincap-method (auc-methods), [3](#)
- auc, multcap-method (auc-methods), [3](#)
- auc-methods, [3](#)

- bincap, [4](#)
- bincap-class, [5](#)

- caTools, [2](#)
- class?bincap, [3](#)
- class?HandTill2001::bincap, [5](#), [9](#)
- class?HandTill2001::cap, [5](#), [9](#)
- class?HandTill2001::multcap, [5](#), [8](#)
- class?multcap, [3](#)

- HandTill2001 (HandTill2001-package), [2](#)
- HandTill2001-package, [2](#)
- ht01.multipleclass, [6](#)
- ht01.twoclass, [7](#)

- multcap, [8](#)
- multcap-class, [9](#)

- ROCR, [2](#)