

Package ‘CompClassMetrics’

January 19, 2026

Title Classification Measures when Subclasses are Involved

Version 1.0.0

Description Accuracy metrics are commonly used to assess the discriminating ability of diagnostic tests or biomarkers. Among them, metrics based on the ROC framework are particularly popular. When classification involves subclasses, the package 'CompClassMetrics' includes functions that can provide the point estimate, confidence interval as well as true values if a parametric setting is known. For more details see Nan and Tian (2025) <doi:10.1177/09622802251343600>, Nan and Tian (2023) <doi:10.1002/sim.9908>, Feng and

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports plot3D, pracma, cubature, stats

NeedsCompilation no

Author Nan Nan [aut, cre]

Maintainer Nan Nan <nannan@buffalo.edu>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2026-01-18 23:30:18 UTC

Contents

adni2	2
auco_func	3
cdf_min_given_max_partial_upper	3
cdf_min_max_partial	4
cdf_order_r	5
CI.func	5
cvus_func	6
f_order_max	6
f_order_min	7
get_max_min_permutations	8
humc_dynamic	8

humc_fourclass	9
humc_min	10
humc_npc1	10
humc_standard	11
plco	11
rocc_curve	12
rocc_curve_emp	13
rocc_surface	13
rocc_surface_emp	14
Index	15

adni2	adni2
-------	-------

Description

Description of adni2.

Format

A data frame with 317 rows and 7 columns:

- RID** Participant ID
- DX.bl** The disease class label
- FDG** Numeric, value of FDG
- AV45** Numeric, value of AV45
- ABETA** Numeric, value of ABETA
- TAU.x** Numeric, value of TAU from CSF
- PTAU** Numeric, value of PTAU from CSF

Source

This is a subset of ADNI2 dataset, available at <https://adni.loni.usc.edu>

auco_func	<i>R function that calculates the true values of AUCo when distribution is known</i>
-----------	--

Description

R function that calculates the true values of AUCo when distribution is known

Usage

```
auco_func(k1, k2, distribution, arg1, arg2)
```

Arguments

k1	number of subclasses in main class-1
k2	number of subclasses in main class-2
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input arg1 parameters
arg2	if distribution is gamma input variance parameter, if gamma input arg2 parameters

Value

The true value of AUCo under given distribution and parameters

cdf_min_given_max_partial_upper	<i>R function that calculates the conditional probability of minimum greater than y_min given maximum equals to y_max of random variables (upper tail probability of minimum given maximum)</i>
---------------------------------	---

Description

R function that calculates the conditional probability of minimum greater than y_min given maximum equals to y_max of random variables (upper tail probability of minimum given maximum)

Usage

```
cdf_min_given_max_partial_upper(y_min, y_max, distribution, arg1, arg2)
```

Arguments

y_min	the value of y_min
y_max	the value of y_max
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is normal input variance parameter, if gamma input rate parameters

Value

The conditional probability of minimum given maximum of random variables

cdf_min_max_partial	<i>R function that calculates the partial of joint probability of min and max over max of NIND random variables</i>
---------------------	---

Description

R function that calculates the partial of joint probability of min and max over max of NIND random variables

Usage

```
cdf_min_max_partial(y_min, y_max, distribution, arg1, arg2)
```

Arguments

y_min	the value of y_min
y_max	the value of y_max
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is normal input variance parameter, if gamma input rate parameters

Value

The partial of joint probability of min and max over max

cdf_order_r	<i>R function that calculates the probability of r-th order statistics of normal random variables (CDF of r-th order statistics)</i>
-------------	--

Description

R function that calculates the probability of r-th order statistics of normal random variables (CDF of r-th order statistics)

Usage

```
cdf_order_r(x, r, distribution, arg1, arg2)
```

Arguments

x	the value of x
r	r-th order statistics
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is normal input variance parameter, if gamma input rate parameters

Value

The probability of r-th order statistics of random variables smaller or equal to x

CI.func	<i>R function that calculates percentile confidence interval given an array of estimates</i>
---------	--

Description

This function provides percentile confidence interval

Usage

```
CI.func(x)
```

Arguments

x	an array of calculated estimates
---	----------------------------------

Value

The percentile confidence interval of given values

cvus_func	<i>R function that calculates the true values of VUSC when distribution is known</i>
-----------	--

Description

R function that calculates the true values of VUSC when distribution is known

Usage

```
cvus_func(k1, k2, k3, distribution, arg1, arg2)
```

Arguments

k1	number of subclasses in main class-1
k2	number of subclasses in main class-2
k3	number of subclasses in main class-3
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input arg1 parameters
arg2	if distribution is gamma input variance parameter, if gamma input arg2 parameters

Value

The true value of VUSc under given distribution and parameters

f_order_max	<i>R function that calculates the probability density of maximum of NIND random variables (PDF)</i>
-------------	---

Description

R function that calculates the probability density of maximum of NIND random variables (PDF)

Usage

```
f_order_max(y_max, distribution, arg1, arg2)
```

Arguments

y_max	the value of y_max
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is normal input variance parameter, if gamma input rate parameters

Value

The probability density of maximum of random variables

f_order_min	<i>R function that calculates the probability density of minimum of NIND random variables (PDF)</i>
-------------	---

Description

R function that calculates the probability density of minimum of NIND random variables (PDF)

Usage

```
f_order_min(y_min, distribution, arg1, arg2)
```

Arguments

y_min	the value of y_min
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is normal input variance parameter, if gamma input rate parameters

Value

The probability density of minimum of NIND random variables

get_max_min_permutations

R function for obtaining all combinations of maximum and minimum of a given dataset

Description

R function for obtaining all combinations of maximum and minimum of a given dataset

Usage

```
get_max_min_permutations(df)
```

Arguments

df Given dataset, in list

Value

A list of all combinations of maximum and minimum of df

humc_dynamic

R function that calculates empirical estimates of HUMcm

Description

This function provides empirical estimates of HUMcm

Usage

```
humc_dynamic(dat, num_sub)
```

Arguments

dat test values in list, each element represents biomarker values for a disease group, ordered in ascending severity

num_sub a vector of number of subclasses in each subclass

Value

The empirical estimate of HUMcm based on given data and num_sub

Examples

```
# Create a list of example data
Y1 <- c(0.9316, 0.9670, 1.3856, 1.3505, 1.0316, 1.1764, 0.7435, 0.5813, 0.4695, 0.3249)
Y2 <- c(1.63950, 1.36535, 1.79859, 0.47961, 1.50978, 1.36525, 0.13515, 2.11275, 0.45659)
Y3 <- c(1.89856, 1.30920, 2.38615, 2.34785, 2.92493, 2.71615, 2.75243, 0.95060, 0.38964)
Y4 <- c(2.580, 2.570, 2.143, 3.079, 1.765, 3.081, 2.175, 2.306, 2.918, 2.507, 4.261, 3.033, 1.836, 2.321)
Y5 <- c(3.969, 3.044, 3.318, 2.862, 3.655, 1.523, 3.722, 4.074, 3.662, 3.571, 5.177, 6.321, 4.932, 4.129)
Y.dat <- list(Y1, Y2, Y3, Y4, Y5)
num_sub <- c(1, 3, 1)
# calculate HUMcm of Y.dat and num_sub
humc_dynamic(Y.dat, num_sub)
```

humc_fourclass	<i>R function that calculates the true values of HUMcm when distribution is known</i>
----------------	---

Description

R function that calculates the true values of HUMcm when distribution is known

Usage

```
humc_fourclass(distribution, arg1, arg2, num_sub)
```

Arguments

distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input arg1 parameters
arg2	if distribution is gamma input variance parameter, if gamma input arg2 parameters
num_sub	the vector of number of subclasses in each main class

Value

The true value of HUMcm under given distribution and parameters

humc_min	<i>R function that calculates the minimum of HUMcm under given structure</i>
----------	--

Description

R function that calculates the minimum of HUMcm under given structure

Usage

```
humc_min(num_sub)
```

Arguments

num_sub the vector of number of subclasses in each main class

Value

the minimum of HUMcm

humc_npci	<i>R function that calculates non-parametric bootstrap percentile confidence interval</i>
-----------	---

Description

This function provides non-parametric bootstrap percentile confidence interval of HUMcm

Usage

```
humc_npci(dat, num_sub, B)
```

Arguments

dat test values in list, each element represents biomarker values for a disease group, ordered in ascending severity

num_sub a vector of number of subclasses in each subclass

B the number of iteration

Value

The non-parametric bootstrap percentile confidence interval of HUMcm

Examples

```
# Create a list of example data
Y1 <- c(0.9316, 0.9670, 1.3856, 1.3505, 1.0316, 1.1764, 0.7435, 0.5813, 0.4695, 0.3249)
Y2 <- c(1.63950, 1.36535, 1.79859, 0.47961, 1.50978, 1.36525,0.13515, 2.11275, 0.45659)
Y3 <- c(1.89856, 1.30920, 2.38615, 2.34785, 2.92493, 2.71615, 2.75243, 0.95060, 0.38964)
Y4 <- c(2.580,2.570,2.143,3.079,1.765,3.081,2.175,2.306,2.918,2.507,4.261,3.033,1.836,2.321)
Y5 <- c(3.969,3.044,3.318,2.862,3.655,1.523,3.722,4.074,3.662,3.571,5.177,6.321,4.932,4.129)
Y.dat <- list(Y1,Y2,Y3,Y4,Y5)
num_sub <- c(1,3,1)
# calculate the non-parametric bootstrap percentile confidence interval
humc_npc1(Y.dat,num_sub,50)
```

humc_standard	<i>R function to calculate the standardized HUMcm under given structure</i>
---------------	---

Description

R function to calculate the standardized HUMcm under given structure

Usage

```
humc_standard(value, num_sub)
```

Arguments

- value the value of HUMcm
- num_sub the vector of number of subclasses in each main class

Value

The standardized HUMcm

plco	<i>PLCO</i>
------	-------------

Description

Description of PLCO.

Format

A data frame with 239 rows and 7 columns:

ID Participant ID

Group The disease class label

CA125 Numeric, value of CA125

CA153 Numeric, value of CA153

CA199 Numeric, value of CA199

KLK6 Numeric, value of KLK6

CA724 Numeric, value of CA724

Source

This is a subset of PLCO dataset, available at <https://edrn.nci.nih.gov>.

rocc_curve	<i>R function for plotting the overall ROC curve and chance curve</i>
------------	---

Description

R function for plotting the overall ROC curve and chance curve

Usage

```
rocc_curve(k1, k2, distribution, arg1, arg2)
```

Arguments

k1	number of subclasses in main class-1
k2	number of subclasses in main class-2
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is gamma input variance parameter, if gamma input rate parameters

Value

The overall ROC curve and chance curve

rocc_curve_emp	<i>R function for plotting the empirical compound ROC curve and chance curve</i>
----------------	--

Description

R function for plotting the empirical compound ROC curve and chance curve

Usage

```
rocc_curve_emp(dat, num_sub)
```

Arguments

dat	values in list, each element represents biomarker values for a disease group, ordered in ascending severity
num_sub	a vector of number of subclasses in each subclass

Value

The empirical compound ROC curve and chance curve

rocc_surface	<i>R function for plotting the compound ROC surface and chance surface</i>
--------------	--

Description

R function for plotting the compound ROC surface and chance surface

Usage

```
rocc_surface(k1, k2, k3, distribution, arg1, arg2)
```

Arguments

k1	number of subclasses in main class-1
k2	number of subclasses in main class-2
k3	number of subclasses in main class-3
distribution	the distribution of marker value follows Normal or Gamma
arg1	if distribution is normal input mean parameters of all subclasses in a vector, if gamma input shape parameters
arg2	if distribution is gamma input variance parameter, if gamma input rate parameters

Value

The compound ROC surface and chance surface

rocc_surface_emp	<i>R function for plotting the empirical compound ROC surface and chance surface</i>
------------------	--

Description

R function for plotting the empirical compound ROC surface and chance surface

Usage

```
rocc_surface_emp(dat, num_sub)
```

Arguments

dat	values in list, each element represents biomarker values for a disease group, ordered in ascending severity
num_sub	a vector of number of subclasses in each subclass

Value

The empirical compound ROC surface and chance surface

Index

adni2, [2](#)
auco_func, [3](#)

cdf_min_given_max_partial_upper, [3](#)
cdf_min_max_partial, [4](#)
cdf_order_r, [5](#)
CI.func, [5](#)
cvus_func, [6](#)

f_order_max, [6](#)
f_order_min, [7](#)

get_max_min_permutations, [8](#)

humc_dynamic, [8](#)
humc_fourclass, [9](#)
humc_min, [10](#)
humc_npc, [10](#)
humc_standard, [11](#)

plco, [11](#)

rocc_curve, [12](#)
rocc_curve_emp, [13](#)
rocc_surface, [13](#)
rocc_surface_emp, [14](#)