

Package: Ecume (via r-universe)

January 23, 2025

Type Package

Title Equality of 2 (or k) Continuous Univariate and Multivariate Distributions

Version 0.9.2

Description We implement (or re-implements in R) a variety of statistical tools. They are focused on non-parametric two-sample (or k-sample) distribution comparisons in the univariate or multivariate case. See the vignette for more info.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

VignetteBuilder knitr

biocViews Software, Infrastructure

Imports stats, spatstat.univar, magrittr, caret, dplyr, e1071, methods, pbapply, kernlab, transport

RoxygenNote 7.3.1

Suggests testthat, covr, knitr, rmarkdown

Config/pak/sysreqs libicu-dev

Repository <https://hectorrdb.r-universe.dev>

RemoteUrl <https://github.com/hectorrdb/ecume>

RemoteRef HEAD

RemoteSha 465831aed91a4aaf0c233faa868a1d13643d136e

Contents

classifier_test	2
ks_test	3
mmd_test	4
stouffer_zscore	6
wasserstein_permut	7

Index**8**

classifier_test	<i>Classifier k-sample test</i>
-----------------	---------------------------------

Description

Classifier k-sample test

Usage

```
classifier_test(
  x,
  y,
  split = 0.7,
  thresh = 0,
  method = "knn",
  control = caret::trainControl(method = "cv"),
  ...
)
```

Arguments

x	Samples from the first distribution or a list of samples from k distribution
y	Samples from the second distribution. Only used if x is a vector.
split	How to split the data between training and test. Default to .7
thresh	Value to add to the null hypothesis. See details.
method	Which model(s) to use during training. Default to knn.
control	Control parameters when fitting the methods. See trainControl
...	Other parameters passed to train

Details

See Lopez-Paz et .al for more background on those tests.

Value

A list containing the following components:

- *statistic* the value of the test statistic.
- *p.value* the p-value of the test.

References

Lopez-Paz, D., & Oquab, M. (2016). Revisiting Classifier Two-Sample Tests, 1–15. Retrieved from <http://arxiv.org/abs/1610.06545>

Examples

```
x <- matrix(c(runif(100, 0, 1),
              runif(100, -1, 1)),
            ncol = 2)
y <- matrix(c(runif(100, 0, 3),
              runif(100, -1, 1)),
            ncol = 2)
classifier_test(x, y)
```

ks_test

*Weighted KS Test***Description**

Weighted Kolmogorov-Smirnov Two-Sample Test with threshold

Usage

```
ks_test(x, y, thresh = 0.05, w_x = rep(1, length(x)), w_y = rep(1, length(y)))
```

Arguments

x	Vector of values sampled from the first distribution
y	Vector of values sampled from the second distribution
thresh	The threshold needed to clear between the two cumulative distributions
w_x	The observation weights for x
w_y	The observation weights for y

Details

The usual Kolmogorov-Smirnov test for two vectors **X** and **Y**, of size m and n rely on the empirical cdfs E_x and E_y and the test statistic

$$D = \sup_{t \in (X, Y)} |E_x(x) - E_y(x)|$$

. This modified Kolmogorov-Smirnov test relies on two modifications.

- Using observation weights for both vectors **X** and **Y**: Those weights are used in two places, while modifying the usual KS test. First, the empirical cdfs are updates to account for the weights. Secondly, the effective sample sizes are also modified. This is inspired from <https://stackoverflow.com/a/55664242/13768995>, using Monahan (2011).
- Testing against a threshold: the test statistic is thresholded such that $D = \max(D - \text{thresh}, 0)$. Since $0 \leq D \leq 1$, the value of the threshold is also between 0 and 1, representing an effect size for the difference.

Value

A list with class "htest" containing the following components:

- *statistic* the value of the test statistic.
- *p.value* the p-value of the test.
- *alternative* a character string describing the alternative hypothesis.
- *method* a character string indicating what type of test was performed.
- *data.name* a character string giving the name(s) of the data.

References

Monahan, J. (2011). *Numerical Methods of Statistics* (2nd ed., Cambridge Series in Statistical and Probabilistic Mathematics). Cambridge: Cambridge University Press. doi:10.1017/CBO9780511977176

Examples

```
x <- runif(100)
y <- runif(100, min = .5, max = .5)
ks_test(x, y, thresh = .001)
```

mmd_test

Perform the Maximum Mean Discrepancy unbiased bootstrap test

Description

Maximum Mean Discrepancy Unbiased Test

Usage

```
mmd_test(
  x,
  y,
  kernel = "rbfdot",
  type = ifelse(min(nrow(x), nrow(y)) < 1000, "unbiased", "linear"),
  null = c("permutation", "exact"),
  iterations = 10^3,
  frac = 1,
  ...
)
```

Arguments

x	d-dimensional samples from the first distribution
y	d-dimensional samples from the first distribution
kernel	A character that must match a known kernel. See details.

type	Which statistic to use. One of 'unbiased' or 'linear'. See Gretton et al for details. Default to 'unbiased' if the two vectors are of length less than 1000 and to 'linear' otherwise.
null	How to asses the null distribution. This can only be set to exact if the type is 'unbiased' and the kernel is 'rbf'.
iterations	How many iterations to do to simulate the null distribution. Default to 10^4. Only used if null is 'permutations'
frac	For the linear statistic, how many points to sample. See details.
...	Further arguments passed to kernel functions

Details

This computes the MMD^{2u} unbiased statistic or the MMDl linear statistic from Gretton et al. The code relies on the pairwise_kernel function from the python module sklearn. To list the available kernels, see the examples.

Value

A list containing the following components:

- *statistic* the value of the test statistic.
- *p.value* the p-value of the test.

References

Gretton, A., Borgwardt, K., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). *A Kernel Two-Sample Test* Journal of Machine Learning Research (2012)

Examples

```
x <- matrix(rnorm(1000, 0, 1), ncol = 10)
y <- matrix(rnorm(1000, 0, 2), ncol = 10)
mmd_test(x, y)
mmd_test(x, y, type = "linear")
x <- matrix(rnorm(1000, 0, 1), ncol = 10)
y <- matrix(rnorm(1000, 0, 1), ncol = 10)
# Set iterations to small number for runtime
# Increase for more accurate results
mmd_test(x, y, iterations = 10^2)
```

stouffer_zscore	<i>Stouffer</i>
-----------------	-----------------

Description

Stouffer's Z-score method

Usage

```
stouffer_zscore(pvals, weights = rep(1, seq_along(pvals)), side = "two")
```

Arguments

pvals	A vector of p-values
weights	A vector of weights
side	How the p-values were generated. One of 'right', 'left' or 'two'.

Details

Given a set of i.i.d p-values and associated weights, it combines the p-values p_i . Letting ϕ be the standard normal cumulative distribution function and $Z_i = \phi^{-1}(1 - p_i)$, the meta-analysis Z-score is

$$Z = (\sum w_i Z_i) * (\sum (w_i)^2)^{-1/2}$$

Value

A list containing the following components:

- *statistic* the value of the test statistic.
- *p.value* the p-value of the test.

References

Samuel Andrew Stouffer. *Adjustment during army life*. Princeton University Press, 1949.

Examples

```
pvals <- runif(100, 0, 1)
weights <- runif(100, 0, 1)
stouffer_zscore(pvals, weights)
```

wasserstein_permut *Permutation test based on Wasserstein distance*

Description

Permutation test based on Wasserstein distance

Usage

```
wasserstein_permut(
  x,
  y,
  iterations = 10^4,
  fast = nrow(x) + nrow(y) > 10^3,
  S = NULL,
  ...
)
```

Arguments

x	Samples from the first distribution
y	Samples from the second distribution. Only used if x is a vector.
iterations	How many iterations to do to simulate the null distribution. Default to 10 ⁴ .
fast	If true, uses the subwasserstein approximate function. Default to true if there are more than 1,000 samples total.
S	Number of samples to use in approximate mode. Must be set if fast=TRUE. See subwasserstein .
...	Other parameters passed to wasserstein or wasserstein1d

Value

A list containing the following components:

- *statistic* the Wasserstein distance between x and y.
- *p.value* the p-value of the permutation test.

Examples

```
x <- matrix(c(runif(100, 0, 1),
               runif(100, -1, 1)),
            ncol = 2)
y <- matrix(c(runif(100, 0, 3),
               runif(100, -1, 1)),
            ncol = 2)
# Set iterations to small number for runtime
# Increase for more accurate results
wasserstein_permut(x, y, iterations = 10^2)
```

Index

classifier_test, [2](#)

ks_test, [3](#)

mmd_test, [4](#)

stouffer_zscore, [6](#)

subwasserstein, [7](#)

train, [2](#)

trainControl, [2](#)

wasserstein, [7](#)

wasserstein1d, [7](#)

wasserstein_permut, [7](#)