Package ‘twosamples’

July 19, 2020

Type Package
Title Fast Permutation Based Two Sample Tests
Version 1.1.1
Author Connor Dowd
Maintainer Connor Dowd <cdowd@chicagobooth.edu>
Description Fast randomization based two sample tests.
    Testing the hypothesis that two samples come from the same distribution using randomiza-
    tion to create p-values. Included tests are: Kolmogorov-Smirnov, Kuiper, Cramer-
    von Mises, Anderson-Darling, Wasserstein, and DTS. The de-
    fault test (two_sample) is based on the DTS test statistic, as it is the most power-
    ful, and thus most useful to most users.
    The DTS test statistic builds on the Wasserstein distance by using a weight-
    ing scheme like that of Anderson-Darling. See the companion pa-
    per at <arXiv:2007.01360> or <https://codowd.com/public/DTS.pdf> for details of that test statis-
    tic, and non-standard uses of the package (parallel for big N, weighted observations, one sam-
    ple tests, etc). We also include the permutation scheme to make test building simple for others.
License GPL (>= 2)
Encoding UTF-8
LazyData true
Imports Rcpp (>= 0.12.17)
LinkingTo Rcpp
RoxygenNote 7.0.2
URL https://github.com/cdowd/twosamples
BugReports https://github.com/cdowd/twosamples/issues
NeedsCompilation yes
Repository CRAN
Date/Publication 2020-07-19 14:10:02 UTC
Description
A two-sample test based on the Anderson-Darling test statistic (\texttt{ad_stat}).

Usage
\begin{verbatim}
ad_stat(a, b, power = 2)
ad_test(a, b, nboots = 2000, p = default.p)
\end{verbatim}

Arguments
\begin{verbatim}
a b power nboots p
\end{verbatim}

\begin{description}
\item[a] a vector of numbers
\item[b] a vector of numbers
\item[power] power to raise test stat to
\item[nboots] Number of bootstrap iterations
\item[p] power to raise test stat to
\end{description}

Details
The AD test compares two ECDFs by looking at the weighted sum of the squared differences between them – evaluated at each point in the joint sample. The weights are determined by the variance of the joint ECDF at that point, which peaks in the middle of the joint distribution (see figure below). Formally – if \( E \) is the ECDF of sample 1, \( F \) is the ECDF of sample 2, and \( G \) is the ECDF of the joint sample then

\[
AD = \sum_{x \in k} \frac{|E(x) - F(x)|^p}{G(x)(1 - G(x))}
\]

where \( k \) is the joint sample. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample. Intuitively the AD test improves on the CVM test by giving lower weight to noisy observations.
In the example plot below, we see the variance of the joint ECDF over the range of the data. It clearly peaks in the middle of the joint sample.

In the example plot below, the AD statistic is the weighted sum of the heights of the vertical lines, where weights are represented by the shading of the lines.

**Value**

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the p-value.

**Functions**

- ad_stat: Anderson-Darling Test statistic
- ad_test: Permutation based two sample Anderson-Darling test
See Also

dts_test() for a more powerful test statistic. See cvm_test() for the predecessor to this test statistic. See dts_test() for the natural successor to this test statistic.

Examples

vec1 = rnorm(20)
vec2 = rnorm(20,4)
ad_test(vec1,vec2)

cvm_stat

Cramer-von Mises Test

Description

A two-sample test based on the Cramer-Von Mises test statistic (cvm_stat).

Usage

cvm_stat(a, b, power = 2)
cvm_test(a, b, nboots = 2000, p = default.p)

Arguments

a       a vector of numbers
b       a vector of numbers
power   power to raise test stat to
nboots  Number of bootstrap iterations
p       power to raise test stat to

Details

The CVM test compares two ECDFs by looking at the sum of the squared differences between them – evaluated at each point in the joint sample. Formally – if E is the ECDF of sample 1 and F is the ECDF of sample 2, then

\[ CVM = \sum_{x \in k} |E(x) - F(x)|^p \]

where k is the joint sample. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample. Intuitively the CVM test improves on KS by using the full joint sample, rather than just the maximum distance – this gives it greater power against shifts in higher moments, like variance changes.
In the example plot below, the CVM statistic is the sum of the heights of the vertical black lines.

Value

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the pvalue.

Functions

- `cvm_stat`: Cramer-Von Mises Test statistic
- `cvm_test`: Permutation based two sample Cramer-Von Mises test

See Also

- `dts_test()` for a more powerful test statistic. See `ks_test()` or `kuiper_test()` for the predecessors to this test statistic. See `wass_test()` and `ad_test()` for the successors to this test statistic.

Examples

```r
vec1 = rnorm(20)
vec2 = rnorm(20,4)
cvm_test(vec1,vec2)
```

**dts_stat**

**DTS Test**

**Description**

A two-sample test based on the DTS test statistic (`dts_stat`). This is the recommended two-sample test in this package because of its power. The DTS statistic is the reweighted integral of the distance between the two ECDFs.
Usage

dts_stat(a, b, power = 1)
dts_test(a, b, nboots = 2000, p = default.p)
two_sample(a, b, nboots = 2000, p = default.p)

Arguments

- **a**: a vector of numbers
- **b**: a vector of numbers
- **power**: also the power to raise the test stat to
- **nboots**: Number of bootstrap iterations
- **p**: power to raise test stat to

Details

The DTS test compares two ECDFs by looking at the reweighted Wasserstein distance between the two. See the companion paper at arXiv:2007.01360 or https://codowd.com/public/DTS.pdf for details of this test statistic, and non-standard uses of the package (parallel for big N, weighted observations, one sample tests, etc).

If the `wass_test()` extends `cvm_test()` to interval data, then `dts_test()` extends `ad_test()` to interval data. Formally – if E is the ECDF of sample 1, F is the ECDF of sample 2, and G is the ECDF of the combined sample, then

\[ DTS = \int_{x \in \mathbb{R}} \frac{|E(x) - F(x)|^p}{G(x)(1 - G(x))} \]

for all x. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample. Intuitively the DTS test improves on the AD test by allowing more extreme observations to carry more weight. At a higher level – CVM/AD/KS/etc only require ordinal data. DTS (and Wasserstein) gain power because they take advantages of the properties of interval data – i.e. the distances have some meaning. However, DTS, like Anderson-Darling (AD) also downweights noisier observations relative to Wass, thus (hopefully) giving it extra power.

In the example plot below, the DTS statistic is the shaded area between the ECDFs, weighted by the
variances – shown by the color of the shading.

Value

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the pvalue.

Functions

• dts_stat: Test statistic based on a weighted area between ECDFs
• dts_test: Permutation based two sample test
• two_sample: Recommended two-sample test

See Also

wass_test(), ad_test() for the predecessors of this test statistic. arXiv:2007.01360 or https://codowd.com/public/DTS.pdf for details of this test statistic

Examples

vec1 = rnorm(20)
vec2 = rnorm(20,4)
dts_stat(vec1,vec2)
dts_test(vec1,vec2)
two_sample(vec1,vec2)
**ks_stat**  
*Kolmogorov-Smirnov Test*

**Description**
A two-sample test using the Kolmogorov-Smirnov test statistic (ks_stat).

**Usage**
- `ks_stat(a, b, power = 1)`
- `ks_test(a, b, nboots = 2000, p = default.p)`

**Arguments**
- `a`: a vector of numbers
- `b`: a vector of numbers
- `power`: power to raise test stat to
- `nboots`: Number of bootstrap iterations
- `p`: power to raise test stat to

**Details**
The KS test compares two ECDFs by looking at the maximum difference between them. Formally – if $E$ is the ECDF of sample 1 and $F$ is the ECDF of sample 2, then

$$KS = \max |E(x) - F(x)|^p$$

for values of $x$ in the joint sample. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample.

In the example plot below, the KS statistic is the height of the vertical black line.
Value

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the pvalue.

Functions

- ks_stat: Kolmogorov-Smirnov test statistic
- ks_test: Permutation based two sample Kolmogorov-Smirnov test

See Also

dts_test() for a more powerful test statistic. See kuiper_test() or cvm_test() for the natural successors to this test statistic.

Examples

vec1 = rnorm(20)
vec2 = rnorm(20,4)
ks_test(vec1,vec2)
Details

The Kuiper test compares two ECDFs by looking at the maximum positive and negative difference between them. Formally – if E is the ECDF of sample 1 and F is the ECDF of sample 2, then

\[
KUIPER = |\max_x E(x) - F(x)|^p + |\max_x F(x) - E(x)|^p
\]

. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample.

In the example plot below, the Kuiper statistic is the sum of the heights of the vertical black lines.

Value

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the pvalue.

Functions

- **kuiper_stat**: Kuiper Test statistic
- **kuiper_test**: Permutation based two sample Kuiper test

See Also

dts_test() for a more powerful test statistic. See ks_test() for the predecessor to this test statistic, and cvm_test() for its natural successor.

Examples

```r
vec1 = rnorm(20)
vec2 = rnorm(20,4)
kuiper_test(vec1, vec2)
```
order_stl

Order function in C++ using the STL

Description
Simply finds the order of a vector in C++. Mostly for internals.

Usage
order_stl(x)

Arguments
x numeric vector

Value
same length vector of integers representing order of input vector

Examples
vec = c(1,4,3,2)
order_stl(vec)

permutation_test_builder

Permutation Test Builder

Description
This function takes a simple two-sample test statistic and produces a function which performs ran-
domization tests (sampling with replacement) using that test stat.

Usage
permutation_test_builder(test_stat_function, default.p = 2)

Arguments
test_stat_function
  a function of two vectors producing a positive number, intended as the test-
  statistic to be used.

default.p This allows for some introduction of defaults and parameters. Typically used to
control the power functions raise something to.
test_stat_function must be structured to take two separate vectors, and then a third value. i.e. (fun = function(vec1, vec2, val1) ...). See examples.

Value

This function returns a function which will perform permutation tests on given test stat.

Functions

• permutation_test_builder: Takes a test statistic, returns a testing function.

See Also

two_sample()

Examples

mean_stat = function(a, b, p) abs(mean(a) - mean(b))**p
myfun = permutation_test_builder(mean_stat, 2.0)
vec1 = rnorm(20)
vec2 = rnorm(20, 4)
myfun(vec1, vec2)

wass_stat

Wasserstein Distance Test

Description

A two-sample test based on Wasserstein’s distance (wass_stat).

Usage

wass_stat(a, b, power = 1)
wass_test(a, b, nboots = 2000, p = default.p)

Arguments

a a vector of numbers
b a vector of numbers
power power to raise test stat to
nboots Number of bootstrap iterations
p power to raise test stat to
Details

The Wasserstein test compares two ECDFs by looking at the Wasserstein distance between the two. This is of course the area between the two ECDFs. Formally – if E is the ECDF of sample 1 and F is the ECDF of sample 2, then

\[ WASS = \int_{x \in \mathbb{R}} |E(x) - F(x)|^p \]

across all x. The test p-value is calculated by randomly resampling two samples of the same size using the combined sample. Intuitively the Wasserstein test improves on CVM by allowing more extreme observations to carry more weight. At a higher level – CVM/AD/KS/etc only require ordinal data. Wasserstein gains its power because it takes advantages of the properties of interval data – i.e. the distances have some meaning.

In the example plot below, the Wasserstein statistic is the shaded area between the ECDFs.

Value

Output is a length 2 Vector with test stat and p-value in that order. That vector has 3 attributes – the sample sizes of each sample, and the number of bootstraps performed for the p-value.

Functions

- `wass_stat`: Wasserstein metric between two ECDFs
- `wass_test`: Permutation based two sample test using Wasserstein metric

See Also

`dts_test()` for a more powerful test statistic. See `cvm_test()` for the predecessor to this test statistic. See `dts_test()` for the natural successor of this test statistic.

Examples

```r
vec1 = rnorm(20)
```
\texttt{vec2 = rnorm(20,4)}
\texttt{wass_test(vec1, vec2)}
Index

ad_stat, 2
ad_test (ad_stat), 2
ad_test(), 5–7

cvm_stat, 4
cvm_test (cvm_stat), 4
cvm_test(), 4, 6, 9, 10, 13

dts_stat, 5
dts_test (dts_stat), 5
dts_test(), 4–6, 9, 10, 13

ks_stat, 8
ks_test (ks_stat), 8
ks_test(), 5, 10
kuiper_stat, 9
kuiper_test (kuiper_stat), 9
kuiper_test(), 5, 9

order_stl, 11

permutation_test_builder, 11

two_sample (dts_stat), 5
two_sample(), 12

wass_stat, 12
wass_test (wass_stat), 12
wass_test(), 5–7