Package 'IndepTest'

August 23, 2018

Type Package

Title Nonparametric Independence Tests Based on Entropy Estimation

Version 0.2.0

Date 2018-08-23

Author
Thomas B. Berrett <t.berrett@statslab.cam.ac.uk> [aut], Daniel J. Grose <dan.grose@lancaster.ac.uk> [cre,ctb],
worth <r.samworth@statslab.cam.ac.uk> [aut]

Maintainer Daniel Grose <dan.grose@lancaster.ac.uk>

Description Implementations of the weighted Kozachenko-Leonenko entropy estimator and independence tests based on this estimator, (Kozachenko and Leonenko (1987) <http://mi.mathnet.ru/eng/ppi797>). Also includes a goodness-of-fit test for a linear model which is an independence test between covariates and errors.

Depends FNN,mvtnorm

Imports Rdpack

RdMacros Rdpack

License GPL

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-08-23 14:04:28 UTC

R topics documented:

KLentropy *KLentropy*

Description

Calculates the (weighted) Kozachenko–Leonenko entropy estimator studied in Berrett, Samworth and Yuan (2018), which is based on the k-nearest neighbour distances of the sample.

Usage

```
KLentropy(x, k, weights = FALSE, stderror = FALSE)
```

Arguments

x	The $n \times d$ data matrix.
k	The tuning parameter that gives the maximum number of neighbours that will be considered by the estimator.
weights	Specifies whether a weighted or unweighted estimator is used. If a weighted estimator is to be used then the default (weights=TRUE) results in the weights being calculated by L2OptW, otherwise the user may specify their own weights.
stderror	Specifies whether an estimate of the standard error of the weighted estimate is calculated. The calculation is done using an unweighted version of the variance estimator described on page 7 of Berrett, Samworth and Yuan (2018).

Value

The first element of the list is the unweighted estimator for the value of 1 up to the user-specified k. The second element of the list is the weighted estimator, obtained by taking the inner product between the first element of the list and the weight vector. If stderror=TRUE the third element of the list is an estimate of the standard error of the weighted estimate.

References

Berrett, T. B., Samworth, R. J. and Yuan, M. (2018). "Efficient multivariate entropy estimation via k-nearest neighbour distances." *Annals of Statistics, to appear*.

Examples

```
n=1000; x=rnorm(n); KLentropy(x,30,stderror=TRUE)   # The true value is 0.5*log(2*pi*exp(1)) = 1.42.
n=5000; x=matrix(rnorm(4*n),ncol=4)              # The true value is 2*log(2*pi*exp(1)) = 5.68
KLentropy(x,30,weights=FALSE)                        # Unweighted estimator
KLentropy(x,30,weights=TRUE)                         # Weights chosen by L2OptW
w=runif(30); w=w/sum(w); KLentropy(x,30,weights=w)  # User-specified weights
```

`L2OptW` *L2OptW*

Description

Calculates a weight vector to be used for the weighted Kozachenko–Leonenko estimator. The weight vector has minimum L_2 norm subject to the linear and sum-to-one constraints of (2) in Berrett, Samworth and Yuan (2018).

Usage

```
L2OptW(k, d)
```

Arguments

`k`	The tuning parameter that gives the number of neighbours that will be considered by the weighted Kozachenko–Leonenko estimator.
`d`	The dimension of the data.

Value

The weight vector that is the solution of the optimisation problem.

References

Berrett, T. B., Samworth, R. J. and Yuan, M. (2018). "Efficient multivariate entropy estimation via k-nearest neighbour distances." *Annals of Statistics, to appear*.

Examples

```
# When d < 4 there are no linear constraints and the returned vector is (0,0,...,0,1).
L2OptW(100,3)
w=L2OptW(100,4)
plot(w,type="l")
w=L2OptW(100,8);
# For each multiple of 4 that d increases an extra constraint is added.
plot(w,type="l")
w=L2OptW(100,12)
plot(w, type="l") # This can be seen in the shape of the plot
```

MINTauto *MINTauto*

Description

Performs an independence test without knowledge of either marginal distribution using permutations and using a data-driven choice of k.

Usage

```
MINTauto(x, y, kmax, B1 = 1000, B2 = 1000)
```

Arguments

x	The $n \times d_X$ data matrix of the X values.
y	The response vector of length $n \times d_Y$ data matrix of the Y values.
kmax	The maximum value of k to be considered for estimation of the joint entropy $H(X, Y)$.
B1	The number of repetitions used when choosing k, set to 1000 by default.
B2	The number of permutations to use for the final test, set at 1000 by default.

Value

The p-value corresponding the independence test carried out and the value of k used.

References

Berrett, T. B. and Samworth R. J. (2017). "Nonparametric independence testing via mutual information." *ArXiv e-prints*. 1711.06642.

Examples

```
# Independent univariate normal data
x=rnorm(1000); y=rnorm(1000);
MINTauto(x,y,kmax=200,B1=100,B2=100)
# Dependent univariate normal data
library(mvtnorm)
data=rmvnorm(1000,sigma=matrix(c(1,0.5,0.5,1),ncol=2))
MINTauto(data[,1],data[,2],kmax=200,B1=100,B2=100)
# Dependent multivariate normal data
Sigma=matrix(c(1,0,0,0,0,1,0,0,0,0,1,0.5,0,0,0.5,1),ncol=4)
data=rmvnorm(1000,sigma=Sigma)
MINTauto(data[,1:3],data[,4],kmax=50,B1=100,B2=100)
```

MINTav *MINTav*

Description

Performs an independence test without knowledge of either marginal distribution using permutations and averaging over a range of values of k.

Usage

```
MINTav(x, y, K, B = 1000)
```

Arguments

x	The $n \times d_X$ data matrix of the X values.
y	The $n \times d_Y$ data matrix of the Y values.
K	The vector of values of k to be considered for estimation of the joint entropy $H(X,Y)$.
B	The number of permutations to use for the test, set at 1000 by default.

Value

The p-value corresponding the independence test carried out.

References

Berrett, T. B. and Samworth R. J. (2017). "Nonparametric independence testing via mutual information." *ArXiv e-prints*. 1711.06642.

Examples

```
# Independent univariate normal data
x=rnorm(1000); y=rnorm(1000);
MINTav(x,y,K=1:200,B=100)
# Dependent univariate normal data
library(mvtnorm);
data=rmvnorm(1000,sigma=matrix(c(1,0.5,0.5,1),ncol=2))
MINTav(data[,1],data[,2],K=1:200,B=100)
# Dependent multivariate normal data
Sigma=matrix(c(1,0,0,0,0,1,0,0,0,0,1,0.5,0,0,0.5,1),ncol=4);
data=rmvnorm(1000,sigma=Sigma)
MINTav(data[,1:3],data[,4],K=1:50,B=100)
```

MINTknown *MINTknown*

Description

Performs an independence test when it is assumed that the marginal distribution of Y is known and can be simulated from.

Usage

```
MINTknown(x, y, k, ky, w = FALSE, wy = FALSE, y0)
```

Arguments

x	The $n \times d_X$ data matrix of X values.
y	The $n \times d_Y$ data matrix of Y values.
k	The value of k to be used for estimation of the joint entropy $H(X, Y)$.
ky	The value of k to be used for estimation of the marginal entropy $H(Y)$.
w	The weight vector to used for estimation of the joint entropy $H(X, Y)$, with the same options as for the KLentropy function.
wy	The weight vector to used for estimation of the marginal entropy $H(Y)$, with the same options as for the KLentropy function.
y0	The data matrix of simulated Y values.

Value

The p-value corresponding the independence test carried out.

References

Berrett, T. B. and Samworth R. J. (2017). "Nonparametric independence testing via mutual information." *ArXiv e-prints*. 1711.06642.

Examples

```
library(mvtnorm)
x=rnorm(1000); y=rnorm(1000);
# Independent univariate normal data
MINTknown(x,y,k=20,ky=30,y0=rnorm(100000))
library(mvtnorm)
# Dependent univariate normal data
data=rmvnorm(1000,sigma=matrix(c(1,0.5,0.5,1),ncol=2))
# Dependent multivariate normal data
MINTknown(data[,1],data[,2],k=20,ky=30,y0=rnorm(100000))
Sigma=matrix(c(1,0,0,0,0,1,0,0,0,0,1,0.5,0,0,0.5,1),ncol=4)
data=rmvnorm(1000,sigma=Sigma)
MINTknown(data[,1:3],data[,4],k=20,ky=30,w=TRUE,wy=FALSE,y0=rnorm(100000))
```

MINTperm *MINTknown*

Description

Performs an independence test without knowledge of either marginal distribution using permutations.

Usage

```
MINTperm(x, y, k, w = FALSE, B = 1000)
```

Arguments

x	The $n \times d_X$ data matrix of X values.
y	The $n \times d_Y$ data matrix of Y values.
k	The value of k to be used for estimation of the joint entropy $H(X,Y)$.
w	The weight vector to used for estimation of the joint entropy $H(X,Y)$, with the same options as for the KLentropy function.
B	The number of permutations to use, set at 1000 by default.

Value

The p-value corresponding the independence test carried out.

References

Berrett, T. B. and Samworth R. J. (2017). “Nonparametric independence testing via mutual information.” *ArXiv e-prints*. 1711.06642.

Examples

```
# Independent univariate normal data
x=rnorm(1000); y=rnorm(1000)
MINTperm(x,y,k=20,B=100)
# Dependent univariate normal data
library(mvtnorm)
data=rmvnorm(1000,sigma=matrix(c(1,0.5,0.5,1),ncol=2))
MINTperm(data[,1],data[,2],k=20,B=100)
# Dependent multivariate normal data
Sigma=matrix(c(1,0,0,0,0,1,0,0,0,0,1,0.5,0,0,0.5,1),ncol=4)
data=rmvnorm(1000,sigma=Sigma)
MINTperm(data[,1:3],data[,4],k=20,w=TRUE,B=100)
```

MINTregression *MINTregression*

Description

Performs a goodness-of-fit test of a linear model by testing whether the errors are independent of the covariates.

Usage

```
MINTregression(x, y, k, keps, w = FALSE, eps)
```

Arguments

x	The $n \times p$ design matrix.
y	The response vector of length n.
k	The value of k to be used for estimation of the joint entropy $H(X, \epsilon)$.
keps	The value of k to be used for estimation of the marginal entropy $H(\epsilon)$.
w	The weight vector to be used for estimation of the joint entropy $H(X, \epsilon)$, with the same options as for the KLentropy function.
eps	A vector of null errors which should have the same distribution as the errors are assumed to have in the linear model.

Value

The p-value corresponding the independence test carried out.

References

Berrett, T. B. and Samworth R. J. (2017). "Nonparametric independence testing via mutual information." *ArXiv e-prints*. 1711.06642.

Examples

```
# Correctly specified linear model
x=runif(100,min=-1.5,max=1.5); y=x+rnorm(100)
plot(lm(y~x),which=1)
MINTregression(x,y,5,10,w=FALSE,rnorm(10000))
# Misspecified mean linear model
x=runif(100,min=-1.5,max=1.5); y=x^3+rnorm(100)
plot(lm(y~x),which=1)
MINTregression(x,y,5,10,w=FALSE,rnorm(10000))
# Heteroscedastic linear model
x=runif(100,min=-1.5,max=1.5); y=x+x*rnorm(100);
plot(lm(y~x),which=1)
MINTregression(x,y,5,10,w=FALSE,rnorm(10000))
```

```
# Multivariate misspecified mean linear model
x=matrix(runif(1500,min=-1.5,max=1.5),ncol=3)
y=x[,1]^3+0.3*x[,2]-0.3*x[,3]+rnorm(500)
plot(lm(y~x),which=1)
MINTregression(x,y,30,50,w=TRUE,rnorm(50000))
```

Index