Package ‘CondIndTests’

November 12, 2019

Type Package
Title Nonlinear Conditional Independence Tests
Version 0.1.5
Date 2019-11-11
Author Christina Heinze-Deml <heinzedeml@stat.math.ethz.ch>,
Jonas Peters <jonas.peters@math.ku.dk>,
Asbjoern Marco Sinius Munk <fgp998@alumni.ku.dk>
Depends R (>= 3.1.0)
Maintainer Christina Heinze-Deml <heinzedeml@stat.math.ethz.ch>
Description Code for a variety of nonlinear conditional independence tests:
Kernel conditional independence test (Zhang et al., UAI 2011, <arXiv:1202.3775>),
Residual Prediction test (based on Shah and Buehlmann, <arXiv:1511.03334>),
Invariant environment prediction,
Invariant target prediction,
Invariant residual distribution test,
Invariant conditional quantile prediction (all from Heinze-Deml et al., <arXiv:1706.08576>).
License GPL
LazyData TRUE
Imports methods, randomForest, quantregForest, lawstat, RPtests,
caTools, mgcv, MASS, kernlab, pracma, mize
RoxygenNote 6.1.1
Suggests testthat
NeedsCompilation no
Repository CRAN
Date/Publication 2019-11-12 06:50:21 UTC
R topics documented:

CondIndTest ........................................................... 2
fishersTestExceedance ............................................ 3
tTestTargetY ......................................................... 4
InvariantConditionalQuantilePrediction ...................... 4
InvariantEnvironmentPrediction ................................... 6
InvariantResidualDistributionTest ............................ 7
InvariantTargetPrediction ....................................... 9
KCI ................................................................. 10
ksResidualDistributions ........................................... 12
leveneAndWilcoxResidualDistributions ....................... 12
propTestTargetE .................................................... 13
ResidualPredictionTest ......................................... 14
wilcoxTestTargetY ................................................ 15

Index 17

---

CondIndTest  Wrapper function for conditional independence tests.

Description

Tests the null hypothesis that Y and E are independent given X.

Usage

CondIndTest(Y, E, X, method = "KCI", alpha = 0.05,
  parsMethod = list(), verbose = FALSE)

Arguments

Y  An n-dimensional vector or a matrix or dataframe with n rows and p columns.
E  An n-dimensional vector or a matrix or dataframe with n rows and p columns.
X  An n-dimensional vector or a matrix or dataframe with n rows and p columns.
method  The conditional independence test to use, can be one of "KCI", "InvariantConditionalQuantilePrediction",
        "InvariantEnvironmentPrediction", "InvariantResidualDistributionTest",
        "InvariantTargetPrediction", "ResidualPredictionTest".
alpha  Significance level. Defaults to 0.05.
parsMethod  Named list to pass options to method.
verbose  If TRUE, intermediate output is provided. Defaults to FALSE.

Value

A list with the p-value of the test (pvalue) and possibly additional entries, depending on the output
of the chosen conditional independence test in method.
References


Examples

# Example 1
set.seed(1)
n <- 100
Z <- rnorm(n)
X <- 4 + 2 * Z + rnorm(n)
Y <- 3 * X^2 + Z + rnorm(n)
test1 <- CondIndTest(X,Y,Z, method = "KCI")
cat("These data come from a distribution, for which X and Y are NOT cond. ind. given Z.")
cat(paste("The p-value of the test is: ", test1$pvalue))

# Example 2
set.seed(1)
Z <- rnorm(n)
X <- 4 + 2 * Z + rnorm(n)
Y <- 3 + Z + rnorm(n)
test2 <- CondIndTest(X,Y,Z, method = "KCI")
cat("The data come from a distribution, for which X and Y are cond. ind. given Z.")
cat(paste("The p-value of the test is: ", test2$pvalue))

---

fishersTestExceedance  Fishers test to test whether the exceedance of the conditional quantiles is independent of the categorical variable E.

Description

Used as a subroutine in InvariantConditionalQuantilePrediction to test whether the exceedance of the conditional quantiles is independent of the categorical variable E.

Usage

fishersTestExceedance(Y, predicted, E, verbose)

Arguments

Y  An n-dimensional vector.
predicted  A matrix with n rows. The columns contain predictions for different conditional quantiles of Y|X.
E  An n-dimensional vector. E needs to be a factor.
verbose  Set to TRUE if output should be printed.
InvariantConditionalQuantilePrediction

Invariant conditional quantile prediction.

Description
Tests the null hypothesis that Y and E are independent given X.

Usage
InvariantConditionalQuantilePrediction(Y, E, X, alpha = 0.05, verbose = FALSE, test = fishersTestExceedance, mtry = sqrt(NCOL(X)), ntree = 100, nodesize = 5, maxnodes = NULL, quantiles = c(0.1, 0.5, 0.9), returnModel = FALSE)

Value
A list with the p-value for the test.
Arguments

Y  
An n-dimensional vector.

E  
An n-dimensional vector. If test = fishersTestExceedance, E needs to be a factor.

X  
A matrix or dataframe with n rows and p columns.

alpha  
Significance level. Defaults to 0.05.

verbose  
If TRUE, intermediate output is provided. Defaults to FALSE.

test  
Unconditional independence test that tests whether exceedence is independent of E. Defaults to fishersTestExceedance.

mtry  
Random forest parameter: Number of variables randomly sampled as candidates at each split. Defaults to \sqrt{\text{NCOL}(X)}.

ntree  
Random forest parameter: Number of trees to grow. Defaults to 100.

nodesize  
Random forest parameter: Minimum size of terminal nodes. Defaults to 5.

maxnodes  
Random forest parameter: Maximum number of terminal nodes trees in the forest can have. Defaults to NULL.

quantiles  
Quantiles for which to test independence between exceedence and E. Defaults to c(0.1,0.5,0.9).

returnModel  
If TRUE, the fitted quantile regression forest model will be returned. Defaults to FALSE.

Value

A list with the following entries:

- pvalue The p-value for the null hypothesis that Y and E are independent given X.
- model The fitted quantile regression forest model if returnModel = TRUE.

Examples

# Example 1
n <- 1000
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantConditionalQuantilePrediction(Y, as.factor(E), X)

# Example 2
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * E + rnorm(n)
InvariantConditionalQuantilePrediction(Y, as.factor(E), X)
InvariantEnvironmentPrediction

Invariant environment prediction.

Description
Tests the null hypothesis that Y and E are independent given X.

Usage
InvariantEnvironmentPrediction(Y, E, X, alpha = 0.05, verbose = FALSE, 
trainTestSplitFunc = caTools::sample.split, 
argsTrainTestSplitFunc = list(Y = E, SplitRatio = 0.8), 
test = propTestTargetE, mtry = sqrt(NCOL(X)), ntree = 100, 
nodesize = 5, maxnodes = NULL, permute = TRUE, 
returnModel = FALSE)

Arguments

Y  An n-dimensional vector.
E  An n-dimensional vector. If test = propTestTargetE, E needs to be a factor.
X  A matrix or data frame with n rows and p columns.
alpha Significance level. Defaults to 0.05.
verbose If TRUE, intermediate output is provided. Defaults to FALSE.
trainTestSplitFunc Function to split sample. Defaults to stratified sampling using caTools::sample.split, assuming E is a factor.
argsTrainTestSplitFunc Arguments for sampling splitting function.
test Unconditional independence test that tests whether the out-of-sample prediction accuracy is the same when using X only vs. X and Y as predictors for E. Defaults to propTestTargetE.
mtry Random forest parameter: Number of variables randomly sampled as candidates at each split. Defaults to sqrt(NCOL(X)).
ntree Random forest parameter: Number of trees to grow. Defaults to 100.
nodesize Random forest parameter: Minimum size of terminal nodes. Defaults to 5.
maxnodes Random forest parameter: Maximum number of terminal nodes trees in the forest can have. Defaults to NULL.
permute Random forest parameter: If TRUE, model that would use X only for predicting Y also includes a random permutation of E. Defaults to TRUE.
returnModel If TRUE, the fitted quantile regression forest model will be returned. Defaults to FALSE.
InvariantResidualDistributionTest

Value

A list with the following entries:

- pvalue The p-value for the null hypothesis that Y and E are independent given X.
- model The fitted models if returnModel = TRUE.

Examples

# Example 1
n <- 1000
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantEnvironmentPrediction(Y, as.factor(E), X)

# Example 2
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * E + rnorm(n)
InvariantEnvironmentPrediction(Y, as.factor(E), X)

# Example 3
E <- rnorm(n)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantEnvironmentPrediction(Y, E, X, test = wilcoxTestTargetY)
InvariantEnvironmentPrediction(Y, X, E, test = wilcoxTestTargetY)

InvariantResidualDistributionTest

Invariant residual distribution test.

Description

Tests the null hypothesis that Y and E are independent given X.

Usage

InvariantResidualDistributionTest(Y, E, X, alpha = 0.05,
   verbose = FALSE, fitWithGam = TRUE,
   test = leveneAndWilcoxResidualDistributions, colNameNoSmooth = NULL,
   mtry = sqrt(NCOL(X)), ntree = 100, nodesize = 5, maxnodes = NULL,
   returnModel = FALSE)
**Arguments**

- **Y** An n-dimensional vector.
- **E** An n-dimensional vector. E needs to be a factor.
- **X** A matrix or dataframe with n rows and p columns.
- **alpha** Significance level. Defaults to 0.05.
- **verbose** If TRUE, intermediate output is provided. Defaults to FALSE.
- **fitWithGam** If TRUE, a GAM is used for the nonlinear regression, else a random forest is used. Defaults to TRUE.
- **test** Unconditional independence test that tests whether residual distribution is invariant across different levels of E. Defaults to leveneAndWilcoxResidDistributions.
- **colNameNoSmooth** Gam parameter: Name of variables that should enter linearly into the model. Defaults to NULL.
- **mtry** Random forest parameter: Number of variables randomly sampled as candidates at each split. Defaults to $\sqrt{\text{NCOL}(X)}$.
- **ntree** Random forest parameter: Number of trees to grow. Defaults to 100.
- **nodesize** Random forest parameter: Minimum size of terminal nodes. Defaults to 5.
- **maxnodes** Random forest parameter: Maximum number of terminal nodes trees in the forest can have. Defaults to NULL.
- **returnModel** If TRUE, the fitted quantile regression forest model will be returned. Defaults to FALSE.

**Value**

A list with the following entries:

- **pvalue** The p-value for the null hypothesis that Y and E are independent given X.
- **model** The fitted model if returnModel = TRUE.

**Examples**

```r
# Example 1
n <- 1000
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantResidualDistributionTest(Y, as.factor(E), X)
InvariantResidualDistributionTest(Y, as.factor(E), X, test = ksResidualDistributions)

# Example 2
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * E + rnorm(n)
InvariantResidualDistributionTest(Y, as.factor(E), X)
InvariantResidualDistributionTest(Y, as.factor(E), X, test = ksResidualDistributions)
```
InvariantTargetPrediction

Invariant target prediction.

Description

Tests the null hypothesis that Y and E are independent given X.

Usage

InvariantTargetPrediction(Y, E, X, alpha = 0.05, verbose = FALSE, fitWithGam = TRUE, trainTestSplitFunc = caTools::sample.split, argsTrainTestSplitFunc = NULL, test = fTestTargetY, colNameNoSmooth = NULL, mtry = sqrt(NCOL(X)), ntree = 100, nodesize = 5, maxnodes = NULL, permute = TRUE, returnModel = FALSE)

Arguments

Y An n-dimensional vector.
E An n-dimensional vector or an nxq dimensional matrix or dataframe.
X A matrix or dataframe with n rows and p columns.
alpha Significance level. Defaults to 0.05.
verbose If TRUE, intermediate output is provided. Defaults to FALSE.
fitWithGam If TRUE, a GAM is used for the nonlinear regression, else a random forest is used. Defaults to TRUE.
trainTestSplitFunc Function to split sample. Defaults to stratified sampling using caTools::sample.split, assuming E is a factor.
argsTrainTestSplitFunc Arguments for sampling splitting function.
test Unconditional independence test that tests whether the out-of-sample prediction accuracy is the same when using X only vs. X and E as predictors for Y. Defaults to fTestTargetY.
colNameNoSmooth Gam parameter: Name of variables that should enter linearly into the model. Defaults to NULL.
mtry Random forest parameter: Number of variables randomly sampled as candidates at each split. Defaults to sqrt(NCOL(X)).
ntree Random forest parameter: Number of trees to grow. Defaults to 100.
nodesize Random forest parameter: Minimum size of terminal nodes. Defaults to 5.
maxnodes Random forest parameter: Maximum number of terminal nodes trees in the forest can have. Defaults to NULL.
permute Random forest parameter: If TRUE, model that would use X only for predicting Y also includes a random permutation of E. Defaults to TRUE.

returnModel If TRUE, the fitted quantile regression forest model will be returned. Defaults to FALSE.

Value
A list with the following entries:

- pvalue The p-value for the null hypothesis that Y and E are independent given X.
- model The fitted models if returnModel = TRUE.

Examples

```r
# Example 1
n <- 1000
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantTargetPrediction(Y, as.factor(E), X)
InvariantTargetPrediction(Y, as.factor(E), X, test = wilcoxTestTargetY)

# Example 2
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * E + rnorm(n)
InvariantTargetPrediction(Y, as.factor(E), X)
InvariantTargetPrediction(Y, as.factor(E), X, test = wilcoxTestTargetY)

# Example 3
E <- rnorm(n)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
InvariantTargetPrediction(Y, E, X)
InvariantTargetPrediction(Y, X, E)
InvariantTargetPrediction(Y, E, X, test = wilcoxTestTargetY)
InvariantTargetPrediction(Y, X, E, test = wilcoxTestTargetY)
```

KCI Kernel conditional independence test.

Description
Tests the null hypothesis that Y and E are independent given X. The distribution of the test statistic under the null hypothesis equals an infinite weighted sum of chi squared variables. This distribution can either be approximated by a gamma distribution or by a Monte Carlo approach. This version includes an implementation of choosing the hyperparameters by Gaussian Process regression.
KCI

Usage

KCI(Y, E, X, width = 0, alpha = 0.05, unbiased = FALSE, gammaApprox = TRUE, GP = TRUE, nRepBs = 5000, lambda = 0.001, thresh = 1e-05, numEig = NROW(Y), verbose = FALSE)

Arguments

Y A vector of length n or a matrix or dataframe with n rows and p columns.
E A vector of length n or a matrix or dataframe with n rows and p columns.
X A matrix or dataframe with n rows and p columns.
width Kernel width; if it is set to zero, the width is chosen automatically (default: 0).
alpha Significance level (default: 0.05).
unbiased A boolean variable that indicates whether a bias correction should be applied (default: FALSE).
gammaApprox A boolean variable that indicates whether the null distribution is approximated by a Gamma distribution. If it is FALSE, a Monte Carlo approach is used (default: TRUE).
GP Flag whether to use Gaussian Process regression to choose the hyperparameters
nRepBs Number of draws for the Monte Carlo approach (default: 500).
lambda Regularization parameter (default: 1e-03).
thresh Threshold for eigenvalues. Whenever eigenvalues are computed, they are set to zero if they are smaller than thresh times the maximum eigenvalue (default: 1e-05).
numEig Number of eigenvalues computed (only relevant for computing the distribution under the hypothesis of conditional independence) (default: length(Y)).
verbose If TRUE, intermediate output is provided. (default: FALSE).

Value

A list with the following entries:

- testStatistic the statistic \( \text{Tr}(K_{(Y|X)} * K_{(E|X)}) \)
- criticalValue the critical point at the p-value equal to alpha; obtained by a Monte Carlo approach if gammaApprox = FALSE, otherwise obtained by Gamma approximation.
- pvalue The p-value for the null hypothesis that Y and E are independent given X. It is obtained by a Monte Carlo approach if gammaApprox = FALSE, otherwise obtained by Gamma approximation.

Examples

# Example 1
n <- 100
E <- rnorm(n)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
**KCI(Y, E, X)**

**KCI(Y, X, E)**

---

**ksResidualDistributions**

*Kolmogorov-Smirnov test to compare residual distributions*

**Description**

Used as a subroutine in InvariantResidualDistributionTest to test whether residual distribution remains invariant across different levels of E.

**Usage**

```
ksResidualDistributions(Y, predicted, E, verbose)
```

**Arguments**

- **Y**: An n-dimensional vector.
- **predicted**: An n-dimensional vector of predictions for Y.
- **E**: An n-dimensional vector. E needs to be a factor.
- **verbose**: Set to TRUE if output should be printed.

**Value**

A list with the p-value for the test.

---

**leveneAndWilcoxonResidualDistributions**

*Levene and wilcoxon test to compare first and second moments of residual distributions*

**Description**

Used as a subroutine in InvariantResidualDistributionTest to test whether residual distribution remains invariant across different levels of E.

**Usage**

```
leveneAndWilcoxonResidualDistributions(Y, predicted, E, verbose)
```
**propTestTargetE**  

**Arguments**

- **Y**  
  An n-dimensional vector.

- **predicted**  
  An n-dimensional vector of predictions for Y.

- **E**  
  An n-dimensional vector. E needs to be a factor.

- **verbose**  
  Set to TRUE if output should be printed.

**Value**

A list with the p-value for the test.

---

**propTestTargetE**  

*Proportion test to compare two misclassification rates.*

---

**Description**

Used as a subroutine in `InvariantEnvironmentPrediction` to test whether out-of-sample performance is better when using X and Y as predictors for E, compared to using X only.

**Usage**

`propTestTargetE(E, predictedOnlyX, predictedXY, verbose)`

**Arguments**

- **E**  
  An n-dimensional vector.

- **predictedOnlyX**  
  Predictions for E based on predictors in X only.

- **predictedXY**  
  Predictions for E based on predictors in X and Y.

- **verbose**  
  Set to TRUE if output should be printed.

**Value**

A list with the p-value for the test.
ResidualPredictionTest

Residual prediction test.

Description

Tests the null hypothesis that Y and E are independent given X.

Usage

ResidualPredictionTest(Y, E, X, alpha = 0.05, verbose = FALSE,
                   degree = 4, basis = c("nystrom", "nystrom_poly", "fourier",
                   "polynomial", "provided")[1], resid_type = "OLS", XBasis = NULL,
                   noiseMat = NULL, getnoiseFct = function(n, ...) { rnorm(n) },
                   argsGetNoiseFct = NULL, nSim = 100, funcOfRes = function(x) {
                   abs(x) }, useX = TRUE, returnXbasis = FALSE,
                   nSub = ceiling(NROW(X)/4), ntree = 100, nodesize = 5,
                   maxnodes = NULL)

Arguments

Y An n-dimensional vector.
E An n-dimensional vector or an nxq dimensional matrix or dataframe.
X A matrix or dataframe with n rows and p columns.
alpha Significance level. Defaults to 0.05.
verbose If TRUE, intermediate output is provided. Defaults to FALSE.
degree Degree of polynomial to use if basis="polynomial" or basis="nystrom_poly".
           Defaults to 4.
basis Can be one of "nystrom","nystrom_poly","fourier","polynomial","provided".
        Defaults to "nystrom".
resid_type Can be "Lasso" or "OLS". Defaults to "OLS".
XBasis Basis if basis="provided". Defaults to NULL.
noiseMat Matrix with simulated noise. Defaults to NULL in which case the simulation is
           performed inside the function.
getnoiseFct Function to use to generate the noise matrix. Defaults to function(n,...){rnorm(n)}.
argsGetNoiseFct Arguments for getnoiseFct. Defaults to NULL.
nSim Number of simulations to use. Defaults to 100.
funcOfRes Function of residuals to use in addition to predicting the conditional mean. De-
              faults to function(x){abs(x)}.
useX Set to TRUE if the predictors in X should also be used when predicting the scaled
       residuals with E. Defaults to TRUE.
wilcoxTestTargetY

returnXBasis Set to TRUE if basis expansion should be returned. Defaults to FALSE.
nSub Number of random features to use if basis is one of "nystrom", "nystrom_poly" or "fourier". Defaults to ceiling(NROW(X)/4).
ntree Random forest parameter: Number of trees to grow. Defaults to 500.
nodesize Random forest parameter: Minimum size of terminal nodes. Defaults to 5.
maxnodes Random forest parameter: Maximum number of terminal nodes trees in the forest can have. Defaults to NULL.

Value

A list with the following entries:

- pvalue The p-value for the null hypothesis that Y and E are independent given X.
- XBasis Basis expansion if returnXBasis was set to TRUE.
- fctBasisExpansion Function used to create basis expansion if basis is not "provided".

Examples

# Example 1
n <- 100
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * (X)^2 + rnorm(n)
ResidualPredictionTest(Y, as.factor(E), X)

# Example 2
E <- rbinom(n, size = 1, prob = 0.2)
X <- 4 + 2 * E + rnorm(n)
Y <- 3 * E + rnorm(n)
ResidualPredictionTest(Y, as.factor(E), X)

# not run:
# # Example 3
# E <- rnorm(n)
# X <- 4 + 2 * E + rnorm(n)
# Y <- 3 * (X)^2 + rnorm(n)
# ResidualPredictionTest(Y, E, X)
# ResidualPredictionTest(Y, X, E)

wilcoxTestTargetY Wilcoxon test to compare two mean squared error rates.

Description

Used as a subroutine in InvariantTargetPrediction to test whether out-of-sample performance is better when using X and E as predictors for Y, compared to using X only.
Usage

 wilcoxTestTargetY(Y, predictedOnlyX, predictedXE, verbose, ...)  

Arguments

 Y An n-dimensional vector.  
 predictedOnlyX Predictions for Y based on predictors in X only.  
 predictedXE Predictions for Y based on predictors in X and E.  
 verbose Set to TRUE if output should be printed.  
 ... Argument to allow for coherent interface of fTestTargetY and wilcoxTestTargetY.  

Value

 A list with the p-value for the test.
Index

CondIndTest, 2
fishersTestExceedance, 3
fTestTargetY, 4
InvariantConditionalQuantilePrediction, 4
InvariantEnvironmentPrediction, 6
InvariantResidualDistributionTest, 7
InvariantTargetPrediction, 9
KCI, 10
ksResidualDistributions, 12
leveneAndWilcoxResidualDistributions, 12
propTestTargetE, 13
ResidualPredictionTest, 14
wilcoxTestTargetY, 15