Package 'NonpModelCheck'

July 2, 2014

Type Package

Title Model Checking and Variable Selection in Nonparametric Regression

Version 1.0

Date 2012-08-03

Author Adriano Zanin Zambom

Maintainer Adriano Zanin Zambom <adriano.zambom@gmail.com>

Depends R (>= 2.15.0), dr, MASS, graphics

Description This package provides tests of significance for covariates (or groups of covariates) in a fully nonparametric regression model and a variable (or group) selection procedure based on False Discovery Rate. In addition, it provides a function for local polynomial regression for any number of dimensions, using a bandwidth specified by the user or automatically chosen by cross validation or an adaptive procedure.

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-09-22 04:33:51

NeedsCompilation yes

R topics documented:

group.npvarselec 2
localpoly.reg 4
npmodelcheck 6
npvarselec 9
plot3d.localpoly.reg 11

Index **13**

`group.npvarselec` *Group variable selection for nonparametric regression*

Description

Performs group variable selection in a completely nonparametric regression model using hypothesis testing for high-dimensional one-way ANOVA and False Discovery Rate (FDR) corrections.

Usage

```
group.npvarselec(X, Y, groups, method = "backward", p = 7, fitSPC = TRUE,
    degree.pol = 0, kernel.type = "epanech", bandwidth = 0, gridsize = 10,
    dim.red = c(1, 10))
```

Arguments

`X`	matrix with observations, rows corresponding to data points and columns correspond to covariates.
`Y`	vector of observed responses.
`groups`	a variable of type "list" containing, in each item, a vector of indices of the covariates in each group.
`method`	type of algorithm to run variable selection, options are "backward", "forward" and "forward2".
`p`	size of the window Wi. See npmodelcheck for details.
`fitSPC`	a logical indicating whether to use the first supervised principal component (SPC) of each group in the local polynomial fitting. See Details.
`degree.pol`	degree of the polynomial to be used in the local fit.
`kernel.type`	kernel type, options are "box", "trun.normal", "gaussian", "epanech", "biweight", "triweight" and "triangular". "trun.normal" is a gaussian kernel truncated between -3 and 3.
`bandwidth`	bandwidth for the local polynomial fit at each step of the elimination (or selection). Options are: 0 for leave-one-out cross validation with criterion of minimum MSE to select a unique bandwidth that will be used for all dimensions; -1 for Generalized Cross Validation to select a unique bandwidth that will be used for all dimensions; -2 for leave-one-out cross validation for each covariate; and -3 for GCV for each covariate. See localpoly.reg.
`gridsize`	number of possible bandwidths to be searched in cross-validation. *Default* is set to 10. If cross-validation is not performed, it is ignored.
`dim.red`	vector with first element indicating 1 for Sliced Inverse Regression (SIR) and 2 for Supervised Principal Components (SPC); the second element of the vector should be number of slices (if SIR), or number of principal components (if SPC). If 0, no dimension reduction is performed. This is used to moderate the curse of dimensionality in the local polynomial estimation at each step of the elimination (or selection). See npmodelcheck for details.

Details

The selection procedure is based on the nonparametric test npmodelcheck, which for testing the significance of group i, uses the residuals of the local polynomial regression of all the other covariates that are not in that group. When "fitSPC" is TRUE, the residuals for each test are computed based on the estimated regression curve m(S_(-i)), where S has d columns, each containing the first SPC of the corresponding group, and S_(-i) is the matrix S without column i.

Backward elimination is done by removing, at each step, the least significant group in the model if its p-value, obtained from the test npmodelcheck, is not significant according to False Discovery Rate (FDR) corrections (Benjamini and Yekutieli, 2001). The final model contains only groups that have significant p-values (based on FDR).

Forward selection is done by adding to the model, at each step, the group with the smallest p-value (when tested with all covariates that are already in the model), if when added, every group in the model is significant according to FDR corrections.

Forward2 selection is as follows: at each step, denote by Z = (Z_1, ..., Z_q) the groups in the model and by W = (W_1, ..., W_r) the groups not in the model (note that (Z,W) = X). Let p_j, j = 1,...r, be the maximum of the set of q+1 p-values obtained from testing each group (Z1,...,Z_q,W_j). Add to the model the group corresponding to the smallest p_j as long as, when added, all the p-values of the groups in the model are significant according to FDR corrections.

See also details of npmodelcheck and localpoly.reg.

Value

`selected`	groups selected
`p_values`	p-values of the tests of the selected groups

Author(s)

Adriano Zanin Zambom <adriano.zambom@gmail.com>

References

Zambom, A. Z. and Akritas, M. G. (2012). a) Nonparametric Model Checking and Variable Selection. arXiv 1205.6761.

Zambom, A. Z. and Akritas, M. G. (2012). b) Signicance Testing and Group Variable Selection. arXiv 1205.6843.

Benjamini, Y. and Yekutieli, D. (2001) The control of false discovery rate in multiple testing under dependency. Annals of Statistics, 29, 1165-1188.

See Also

`npmodelcheck`, `localpoly.reg`, `npvarselec`

Examples

```
groups = vector("list",7)

groups[[1]] = c(3,8,5,12,14)
```

```
groups[[2]] = c(6,7,9,10)
groups[[3]] = 13
groups[[4]] = c(1,2,4,11)
groups[[5]] = c(15,16, 20)
groups[[6]] = 17
groups[[7]] = c(18,19)

X = matrix(1,100,20)
for (i in 1:20)
X[,i] = rnorm(100)

Y = X[,13]^3 + X[,7] + X[,15]^2 + X[,16] + rnorm(100)

group.npvarselec(X,Y,groups)
```

`localpoly.reg` *Local Polynomial Regression Fitting*

Description

Computes the smoothed response or its derivatives in a nonparametric regression using local polynomial fitting.

Usage

```
localpoly.reg(X, Y, points = NULL, bandwidth = 0, gridsize = 30, degree.pol = 0,
    kernel.type = "epanech", deriv = 0)
```

Arguments

`X`	matrix with observations, rows corresponding to data points and columns corresponding to covariates.
`Y`	vector of observed responses.
`points`	points at which to get smoothed values. If NULL, estimation is done on the observations of X.
`bandwidth`	bandwidth, vector or matrix of bandwidths. When X is univariate(vector): options for the bandwidth are: 0 for leave-one-out cross validation with criterion of minimum MSE; -1 for Generalized Cross Validation; -4 for adaptive bandwidth (see details); a vector of same length as X representing a bandwidth that changes with the location of estimation. When X is multivariate(matrix): options for the bandwidth are: 0 for leave-one-out cross validation with criterion of minimum MSE (search is done in a grid margilally for each covariate); -1 for Generalized Cross Validation (search is done in a grid marginally for each covariate); -2 leave-one-out cross validation (search is done in any combination of grids for each covariate); and if -3, GCV for each covariate(search is done in any combination of grids for each covariate). See *Details*.

gridsize	number of possible bandwidths to be searched in cross-validation. If left as *default* 0, gridsize is taken to be 5+as.integer(100/d^3). If cross-validation is not performed, it is ignored.
degree.pol	degree of the polynomial to be used in the local fit. In the univariate case there is no restriction; in the multivariate case, the degree can be 0,1 or 2.
kernel.type	kernel type, options are "box", "trun.normal", "gaussian", "epanech", "biweight", "triweight" and "triangular". "trun.normal" is a gaussian kernel truncated between -3 and 3.
deriv	order of the derivative of the regression function to be estimated.

Details

Computes smoothed values using local polynomial fitting with the specified kernel type. If multidimensional, a multiplicative(product) kernel is used as weight.

In cross validation, for multivariate X and bandwidth options 0 and -1, the procedure searches individually for each covariate, the bandwidth that produces the smallest MSE from a grid of *gridsize* possible bandwidths evenly distributed between the minimum and maximum/2 distance of any of the points of that covariate. In other words, one separate cross-validation is performed in each dimension of X.

In cross validation, for multivariate X and bandwidth options -2 and -3, a d(number of covariates) dimensional grid is created, where each dimension of the grid is a vector of *gridsize* possible bandwidths evenly distributed between the minimum and maximum/2 distance of any of the points in each covariate. Then a search is done crossing all possible combinations of values of each dimension of the grid, where the resulting vector of bandwidths correspond to those which yeild minimum MSE.

Adaptive bandwidth, for univariate X only, is obtained by a similar procedure to the one proposed by Fan and Gijbels (1995). The interval is split into [1.5*n/(10*log(n))] intervals, a leave-one-out cross validation is performed in each interval to obtain a local bandwidth. These bandwidths are then smoothed to obtain the bandwidth for each point in X.

Value

X	the same input matrix
Y	the same input response vector
points	points at which smoothed values were computed
bandwidth	bandwidth used for the polynomial fit
predicted	vector with the predicted(smoothed) values

Author(s)

Adriano Zanin Zambom <adriano.zambom@gmail.com>

References

Fan J. and Gijbels I. (1995). Data-driven bandwidth selection in local polynomial fitting: Variable Bandwidth and Spatial Adaptation. JRSS-B. Vol 57(2), 371-394.

Wand M. P. and Jones M. C. (1995). Kernel Smoothing. Chapman and Hall.

See Also

npvarselec, npmodelcheck

Examples

```
X = rnorm(100)
Y = X^3 + rnorm(100)

localpoly.reg(X, Y, degree.pol = 0, kernel.type = "box", bandwidth = 0)
localpoly.reg(X, Y, degree.pol = 1, kernel.type = "box", bandwidth = 0)
##--
X = runif(100,-3,3)
Y = sin(1/2*pi*X) + rnorm(100,0,.5)

localpoly.reg(X, Y, degree.pol = 0, kernel.type = "gaussian", bandwidth = 0)
localpoly.reg(X, Y, degree.pol = 1, kernel.type = "gaussian", bandwidth = 0)
```

---

npmodelcheck *Hypothesis Testing for Covariate or Group effect in Nonparametric Regression*

Description

Tests the significance of a covariate or a group of covariates in a nonparametric regression based on residuals from a local polynomial fit of the remaining covariates using high dimensional one-way ANOVA.

Usage

```
npmodelcheck(X, Y, ind_test, p = 7, degree.pol = 0, kernel.type = "epanech",
    bandwidth = 0, gridsize = 30, dim.red = c(1, 10))
```

Arguments

X	matrix with observations, rows corresponding to data points and columns correspond to covariates.
Y	vector of observed responses.
ind_test	index or vector with indices of covariates to be tested.
p	size of the window Wi. See Details
degree.pol	degree of the polynomial to be used in the local fit.
kernel.type	kernel type, options are "box", "trun.normal", "gaussian", "epanech", "biweight", "triweight" and "triangular". "trun.normal" is a gaussian kernel truncated between -3 and 3.

`bandwidth` bandwidth, vector or matrix of bandwidths for the local polynomial fit. If a vector of bandwidths, it must correspond to each covariate of X_-(ind_test), that is, the covariates not being tested. If 0, leave-one-out cross validation with criterion of minimum MSE is performed to select a unique bandwidth that will be used for all dimensions of X_-(ind_test); if -1, Generalized Cross Validation is performed to select a unique bandwidth that will be used for all dimensions of X_-(ind_test); if -2 leave-one-out cross validation for each covariate of X_-(ind_test); and if -3, GCV for each covariate of X_-(ind_test). It can be a matrix of bandwidths (not to be confused with bandwidth matrix H), where each row is a vector of the same dimension of the columns of X_-(ind_test), representing a bandwidth that changes with the location of estimation for multidimensional X. See localpoly.reg.

`gridsize` number of possible bandwidths to be searched in cross-validation. If left as *default* 0, gridsize is taken to be 5+as.integer(100/d^3). If cross-validation is not performed, it is ignored.

`dim.red` vector with first element indicating 1 for Sliced Inverse Regression (SIR) and 2 for Supervised Principal Components (SPC); the second element of the vector should be number of slices (if SIR), or number of principal components (if SPC). If 0, no dimension reduction is performed. See Details.

Details

To test the significance of a single covariate, say X_j, assume that its observations X_ij, i = 1,...n, define the factor levels of a one-way ANOVA. To construct the ANOVA, each of these factor levels is augmented by including residuals from nearby covariate values. Specifically, cell "i" is augmented by the values of the residuals corresponding to observations X_ij for "i" in W_i (W_i defines the neighborhood, and has size "p"). These residuals are obtained from a local polynomial fit of the remaining covariates X_-(j). Then, the test for the significance of X_j is the test for no factor effects in the high-dimensional one-way ANOVA. See references for further details.

When testing the significance of a group of covariates, the window W_i is defined using the fist supervised principal component (SPC) of the covariates in that group; and the local polynomial fit uses the remaining covariates X_-(ind_test).

Dimension reduction (SIR or SPC) is applied on the remaining covariates (X_-(ind_test)), which are used on the local polynomial fit. This reduction is used to moderate the effect of the curse of dimensionality when fitting nonparametric regression for several covariates. For SPC, the supervision is done in the following way: only covariates with p-values (from univariate "npmodelcheck" test with Y) < 0.3 can be selected to compose the principal components. If no covariate has p-value < 0.3, then the most significant covariate will be the only component. For SIR, the size of the effective dimension reduction space is selected automatically through sequential testing (see references for details).

Value

`bandwidth` bandwidth used for the local polynomial fit

`predicted` vector with the predicted values with the remaining covariates

`p-value` p-value of the test

Author(s)

Adriano Zanin Zambom <adriano.zambom@gmail.com>

References

Zambom, A. Z. and Akritas, M. G. (2012). a) Nonparametric Model Checking and Variable Selection. arXiv 1205.6761.

Zambom, A. Z. and Akritas, M. G. (2012). b) Signicance Testing and Group Variable Selection. arXiv 1205.6843.

Li, K. C. (1991). Sliced Inverse Regression for Dimension Reduction. Journal of the American Statistical Association, 86, 316-327.

Bair E., Hastie T., Paul D. and Tibshirani R. (2006). Prediction by supervised principal components. Journal of the American Statistical Association, 101, 119-137.

See Also

localpoly.reg, npvarselec

Examples

```
X = matrix(1,100,5)

X[,1] = rnorm(100)
X[,2] = rnorm(100)
X[,3] = rnorm(100)
X[,4] = rnorm(100)
X[,5] = rnorm(100)
Y = X[,3]^3 + rnorm(100)

npmodelcheck(X, Y, 2, p = 9, degree.pol = 0, kernel.type = "trun.normal",
bandwidth = -1,  dim.red = 0)

npmodelcheck(X, Y, 3, p = 7, degree.pol = 0, kernel.type = "trun.normal",
bandwidth = 0,  dim.red = c(2,2))

npmodelcheck(X, Y, c(1,2), p = 11, degree.pol = 0, kernel.type = "box",
bandwidth = 0,  dim.red = c(1,10))

npmodelcheck(X, Y, c(3,4), p = 5, degree.pol = 0, kernel.type = "box",
bandwidth = 0,  dim.red = c(1,20))

npmodelcheck(rnorm(100), rnorm(100), 1, p = 5, degree.pol = 1, kernel.type = "box",
bandwidth = 0,  dim.red = c(1,20))
```

`npvarselec` *Variable selection for covariates in nonparametric regression*

Description

Performs variable selection using hypothesis tests of covariates in high-dimensional one-way ANOVA for a completely nonparametric regression model.

Usage

```
npvarselec(X, Y, method = "backward", p = 7, degree.pol = 0,
    kernel.type = "epanech", bandwidth = 0, gridsize = 10, dim.red = c(1, 10))
```

Arguments

`X`	matrix with observations, rows corresponding to data points and columns correspond to covariates.
`Y`	vector of observed responses.
`method`	type of algorithm to run variable selection, options are "backward", "forward" and "forward2".
`p`	size of the window Wi. See npmodelcheck for details.
`degree.pol`	degree of the polynomial to be used in the local fit.
`kernel.type`	kernel type, options are "box", "trun.normal", "gaussian", "epanech", "biweight", "triweight" and "triangular". "trun.normal" is a gaussian kernel truncated between -3 and 3.
`bandwidth`	bandwidth for the local polynomial fit at each step of the elimination (or selection). Options are: 0 for leave-one-out cross validation with criterion of minimum MSE to select a unique bandwidth that will be used for all dimensions; -1 for Generalized Cross Validation to select a unique bandwidth that will be used for all dimensions; -2 for leave-one-out cross validation for each covariate; and -3 for GCV for each covariate. See localpoly.reg.
`gridsize`	number of possible bandwidths to be searched in cross-validation. *Default* is set to 10. If cross-validation is not performed, it is ignored.
`dim.red`	vector with first element indicating 1 for Sliced Inverse Regression (SIR) and 2 for Supervised Principal Components (SPC); the second element of the vector should be number of slices (if SIR), or number of principal components (if SPC). If 0, no dimension reduction is performed. This is used to moderate the curse of dimensionality in the local polynomial estimation at each step of the elimination (or selection). See npmodelcheck for details.

Details

Backward elimination is done by removing, at each step, the least significant covariate in the model if its p-value, obtained from the the test npmodelcheck, is not significant according to False Discovery Rate (FDR) corrections (Benjamini and Yekutieli, 2001). The precudere continues until all covariates left have significant p-values based on FDR.

Forward selection is done by adding to the model, at each step, the covariate with the smallest p-value (when tested with all covariates that are already in the model), if when added, every covariate in the model is significant according to FDR corrections.

Forward2 selection as follows: at each step, denote by Z = (Z_1, ..., Z_q) the covariates in the model and by W = (W_1, ..., W_r) the covariates not in the model (note that (Z,W) = X). Let p_j, j = 1,...r, be the maximum of the set of q+1 p-values obtained from testing each the covariates (Z1,...,Z_q,W_j). Add to the model the covariate corresponding to the smallest p_j as long as, when added, all the p-values of the covariates in the model are significant according to FDR corrections.

See also details of npmodelcheck.

Value

`selected`	selected covariates
`p_values`	p-values of the tests of the selected covariates

Author(s)

Adriano Zanin Zambom <adriano.zambom@gmail.com>

References

Zambom, A. Z. and Akritas, M. G. (2012). a) Nonparametric Model Checking and Variable Selection. arXiv 1205.6761.

Benjamini, Y. and Yekutieli, D. (2001) The control of false discovery rate in multiple testing under dependency. Annals of Statistics, 29, 1165-1188.

See Also

`npmodelcheck`, `localpoly.reg`, `group.npvarselec`

Examples

```
d = 10
X = matrix(1,90,d)

for (i in 1:d)
   X[,i] = rnorm(90)
Y = X[,3]^3 + X[,6]^2 + sin(1/2*pi*X[,9]) + rnorm(90)

npvarselec(X, Y, method = "forward", p = 9, degree.pol = 0,
kernel.type = "trun.normal", bandwidth = 0)
```

`plot3d.localpoly.reg` *3d plot from a local polynomial fit*

Description

Create a 3d plot from a local polynomial fit of two covariates and a response variable.

Usage

```
plot3d.localpoly.reg(X,Y, bandwidth = 0, gridsize = 30, degree.pol = 0,
   kernel.type = "epanech", gridsurface = 30, xlab=expression(X_1),
   ylab=expression(X_2), zlab=expression(Y), theta = 30, phi = 30,
   expand = 0.5, col = "lightblue", ltheta = 120, shade = 0.75,
   ticktype = "detailed", pch = 16,...)
```

Arguments

`X`	n by 2 matrix with observations, rows corresponding to data points and columns correspond to covariates.
`Y`	vector of observed responses.
`bandwidth`	bandwidth, vector or matrix. If 0, leave-one-out cross validation with criterion of minimum MSE is performed to select a unique bandwidth that will be used for all dimensions of X; if -1, Generalized Cross Validation is performed to select a unique bandwidth that will be used for all dimensions of X; if -2 leave-one-out cross validation for each covariate; and if -3, GCV for each covariate. It may be a vector for each dimension of the X; or a matrix of bandwidths (not to be confused with bandwidth matrix H), where each row is a vector of size 2, representing a bandwidth that changes with the location of estimation for the grid. See localpoly.reg.
`gridsize`	number of possible bandwidths to be searched in cross-validation. If left as *default* 0, gridsize is taken to be 5+as.integer(100/d^3). If cross-validation is not performed, it is ignored.
`degree.pol`	degree of the polynomial to be used in the local fit.
`kernel.type`	kernel type, options are "box", "trun.normal", "gaussian", "epanech", "biweight", "triweight" and "triangular". "trun.normal" is a gaussian kernel truncated between -3 and 3.
`gridsurface`	number of points on each axis at which to estimate the local polynomial surface.
`xlab`	parameter for persp
`ylab`	parameter for persp
`zlab`	parameter for persp
`theta`	parameter for persp
`phi`	parameter for persp
`expand`	parameter for persp

col	parameter for persp
ltheta	parameter for persp
shade	parameter for persp
ticktype	parameter for persp
pch	parameter for persp
...	further parameters for plotting persp

Details

Uses function "persp" to plot the estimated surface of a local polynomial fit in a nonparametric model with two covariates. The surface is estimated at points of a grid with size "gridsurface", which are evenly distributed between the minimum and maximum of the observed predictors. It also adds the observed points to the plot.

Value

X	the same input matrix
Y	the same input response vector
points	points at which to get smoothed values
bandwidth	bandwidth used for the polynomial fit
predicted	matrix with the predicted values at grid points

Author(s)

Adriano Zanin Zambom <adriano.zambom@gmail.com>

See Also

localpoly.reg

Examples

```
X = matrix(0,50,2)
X[,1] = runif(50,-2,2)
X[,2] = runif(50,-2,2)
Y = 4*sin(pi*X[,1]) + X[,2] + rnorm(50)

plot3d.localpoly.reg(X,Y, bandwidth=-2, gridsize = 15, degree.pol = 0, gridsurface=20)
```

Index

∗Topic **\textasciitildekwd1**
group.npvarselec, 2
localpoly.reg, 4
npmodelcheck, 6
npvarselec, 9
plot3d.localpoly.reg, 11
∗Topic **\textasciitildekwd2**
group.npvarselec, 2
localpoly.reg, 4
npmodelcheck, 6
npvarselec, 9
plot3d.localpoly.reg, 11

group.npvarselec, 2, *10*

localpoly.reg, *2, 3*, 4, *7–12*

npmodelcheck, *2, 3*, *6*, 6, *9, 10*
npvarselec, *3*, *6*, *8*, 9

plot3d.localpoly.reg, 11