Package 'XReg'

April 17, 2009

Version 1.0

Date 2007-06-08

Title Extreme Regression

Author Michael LeBlanc <mleblanc@fhcrc.org>.

Maintainer Michael LeBlanc <mleblanc@fhcrc.org>

Depends R (>= 2.3.1)

Description This is a package that implements extreme regression estimation as described in LeBlanc, Moon and Kooperberg, Biostatistics, 7, 71-84, 2006.

License GPL (>= 2)

Repository CRAN

Date/Publication 2008-01-08 09:08:51

R topics documented:

`level.set` *Level sets for extreme regression models*

Description

`level.set` This function gives the decision rule for a given threshold. It also tells how many observations are indicated by each rule and gives the overall indicator of the observations indicated by the rule. Currently this function is only written for >= rules; however, with it one should be able to specify the desired inequality

Usage

```
level.set(thresh, output,pred=F,xpred=NULL,lower=-Inf,upper=Inf)
```

Arguments

`thresh`	threshold or target output for rule x:f(x)>=thresh
`output`	data structure from XR regression function
`pred`	determine which observations are consistent with rule
`xpred`	if pred=T, then sent observations for which the rule should be applied
`lower`	lower bound for predictions
`upper`	upper bound for predictions

Value

a list containing:

`cutpoints`	cutpoints corresponding to level thresh

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##
  # see XReg
```

`plotXRinvreg` *Plots inverse of extreme regression model*

Description

`plotXRinvreg` Returns a plot of the inverse regression function, as a matrix plot. Rows of matrix are variables and columns correspond to miniumum terms. This representation of the regression function probably only works well as a graphical display for up to about 5 unique variables and 5 terms.

Usage

```
plotXRinvreg(x,xrout,shading=F,quant=F,both=F,qlev=c(.2,.4,.6,.8),
                      spec.ylim=F,mat.ylim=NULL,level.set=NA)
```

Arguments

`x`	matrix of predictors
`xrout`	output from XR regression function XReg
`shading`	if shading=T shade the directions of the rules in panel associated with sequence of rules x:f(x)>=q
`quant`	if quant=T, plot rules in terms of quantiles or ranked predictions. Note, models will not appear linear univarite.
`both`	if both=T, plot in linear form, but put quantiles on the top of plot
`qlev`	if both=T, these are the labels for the quantiles put on the top of the plot
`spec.ylim`	if spec.ylim=F, do not specify the limits of the y axis of the plot. Note, in this case the same variable different plots could have different axes. If spec.ylim=T, then the user must provide the limits y-axis for each of the variables.
`mat.ylim`	This gives the y-axis limits (min,max) for each of the variables, it is a 2xp matrix, where each column corresponds to a predictor variable. It is only used if spec.ylim=T.
`level.set`	If this is set to a numeric value, a highlighted line will be added to the plot to indicate associated rule

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##
  # See XReg
```

plotXRreg *Plots component linear models in extreme regression model*

Description

plotXRreg This function returns a plot of the regression function and data points as a matrix plot. Rows of matrix are variables and columns correspond to miniumum terms. This representation of the regression function probably only works well as a graphical display up to about 5 unique variables and 5 terms. It also tells how many observations are indicated by each rule and gived the overall indicator of the observations indicated by the rule.

Usage

```
plotXRreg(y,x,xrout,allpoints=F)
```

Arguments

y	response
x	matrix of predictors
xrout	output from XR regression function XReg
allpoints	if allpoints=T, plot all points (but highlight those corresponding to the local univariate function

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##
  # See XReg
#
```

predictfun *Predictions for extreme regression models*

Description

predictfun This gives predictions but also give assigments to the active sets for data given in the xpred matrix. The default is not to give assignments.

Usage

```
predictfun(xpred,varlist,beta0list,lower=-Inf,upper=Inf,activeset=F)
```

Arguments

xpred	xmatrix for predictions
varlist	model form list (described in XR regression function)
beta0list	the beta list from XR regression function
lower	lower bound for predictions
upper	upper bound for predictions
activeset	if T denote where observations are in terms of active set

Value

a list containing:

predfit	predicted values

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##
 #See XReg
```

---

stepXR — *Forward stepwise function for adding terms to a extreme regression model*

Description

stepXR This functions calls builds extreme regression models in a step-wise fashion

Usage

```
stepXR(y, x, ypred = y, xpred = x, kstep = 5, varlist = NULL, outtemp = NULL, niter
        niterfull = 20, step.sizefull = 0.2, kbest = 1, limsize = 0.05, step.adapt
         = min(ypred), predict = T, randinit=0,penalty = 2, update.all = T,survival
```

Arguments

y	response variable
x	matrix of predictors
xpred	xmatrix for predictions
ypred	y values on test data set
varlist	is a list specifying the form of the extreme regression model. EXAMPLE: varlist=list(1, c(2,3),c(3,4)) –> max (a1+b1x1,min(a2+b2x2,a3+b3x3),min(a4+b4x3,a5+b5x4))

lower	minimum prediction
outtemp	current output model
step.adapt	If T the step size is adapted so that the sum of squares is reduced at each step. This slows down the algorithm.
predict	make predictions on the xpred values
update.all	update all coefficients as terms are added
upper	maximum prediction EXAMPLE lower=min(y) and upper=max(y) predictions will always be in the range of y.
limsize	the targeted smallest number of observations used to estimate a univariate function.
niter	number of estimations steps
step.adapt	If T the step size is adapted so that the sum of squares is reduced at each step. This slows down the algorithm.
survival	If yes, the expoential survival model is used.
times	If survival=T, time under observation
status	If survival=T, survival status (1=dead, 0=alive)
randinit	If randinit>0 some noise is added to the initial estimates. Randinit adds noise of standard deviation form randinit/sqrt(n). This avoids bad initial starts when the same variable is involved in multiple terms.
kstep	the number of linear components to add
penalty	penalty used in model selection
step.size	step.size for each term considered for addition
niterfull	number of steps after term has been selected
step.sizefull	as above
kbest	used in addeval function and is the number of variables considered for selection. It is the kbest variables with highest univariate correlation.

Value

a list containing:

stepcoef	list of coefficients for the stepwise process
steplist	list of variables for the stepwise process
stepfit	fit for training data for each model
stepfitpred	fit for test data (if given) for each model
bestgcv	best model by gcv

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##
  # See XReg
```

XReg *Extreme Regression*

Description

XReg This function estimates parameters in an extreme regression model

Usage

```
XReg(y, x, varlist, lower =  - Inf, upper = Inf,
limsize = (min(c(50,max(c(0.05*length(y),25)))))/length(y), niter = 5, betainit = N
var.step = F,step.adapt = F, step.size = 1, adaptcntlim = 6, randinit = 0,
update.terms = as.matrix(c(0, 0)), survival=F,times=y,status=y)
```

Arguments

y	response variable
x	matrix of predictors
varlist	is a list specifying the form of the extreme regression model. EXAMPLE: varlist=list(1, c(2,3),c(3,4)) –> max (a1+b1x1,min(a2+b2x2,a3+b3x3),min(a4+b4x3,a5+b5x4))
lower	minimum prediction
upper	maximum prediction EXAMPLE lower=min(y) and upper=max(y) predictions will always be in the range of y.
limsize	the targeted smallest number of observations used to estimate a univariate function.
niter	number of estimations steps
var.step	variable step-size
betainit	initial estimates for regression parameters. This is in list form the same as output of the regression parameters.
step.adapt	If T the step size is adapted so that the sum of squares is reduced at each step. This slows down the algorithm.
adaptcntlim	This limits the number of reduction steps in the adaptive step selection to reduce sums of squares. Larger the numbers can slow algorithm.
step.size	Fixed step size. A size of 1 moves directly to the least squares solution. A value of 1 is almost always too large. Better results are obtained with fixed step sizes about .2-.5. The larger the step size the more rapidly the estimates change, but may also may cause lack of convergence.
update.terms	Only update the specified terms. EXAMPLE: Suppose the varlist=list(1,2,3,c(1,2),c(1,3),c(2,3)). One needs to give it an betainit list corresponding to the varlist above but then to specify only to update terms c(1,2),c(1,3),c(2,3) above one would specify update.terms=cbind(c(4,1),c(4,2),c(5,1),c(5,2),c(6,2),c(6,3)). Update components of the 4th, 5th and 6th terms in the list.
survival	If yes, the expoential survival model is used.

times	If survival=T, time under observation
status	If survival=T, survival status (1=dead, 0=alive)
randinit	If randinit>0 some noise is added to the initial estimates. Randinit adds noise of standard deviation form randinit/sqrt(n). This avoids bad initial starts when the same variable is involved in multiple terms.

Value

a list containing:

beta0list	list of parameter estimates
pred0list	list of predictions from each univariate model
prop0list	proportions of observations in each subregion
actset	observations identified in each subregion
actset0	observation used in the estimation of each univariate model
predicted	overall XR model prediction
varlist	varlist as input to the function
outmat	gives the current activeset and change with each iteration. Useful for diagnosing convergence issues.

Author(s)

Michael LeBlanc ⟨mleblanc@fhcrc.org⟩

Examples

```
##


# Simple MAX-MIN  example

ffsimp=function(x) {
        eta=pmax(pmin(1*x[,1],.5*x[,2]),pmin(x[,3],x[,4]))
        return(eta) }

# training data
set.seed(123)
n=500
x=matrix(rnorm(n*5),ncol=5)
eta=ffsimp(x)
y=eta+rnorm(n)*1

# For survival data would need to load SURVIVAL library

# XR regression
# specify form of model.
varlist=list(c(1,2),c(3,4))
outxr1=XReg(y,x,varlist,step.size=.5,niter=20)
```

```
# XR regression using adaptive step-size

outxr2=XReg(y,x,varlist,step.size=.5,step.adapt=TRUE,niter=20)

# ------------------------------------------------------------------------
# XR Survival analysis

# load packages
library(survival)
library(splines)

# MAKE SURVIVAL DATA
tt=-log(runif(n))/exp(eta-mean(eta))  # make uncensored survival times
cc=runif(n)*4   # make censoring times

status=1*(tt<cc)
times=status*tt+(1-status)*cc

# Exponential extreme regression model

outxr4=XReg(y,x,varlist,step.size=.5,step.adapt=FALSE,niter=20,survival=TRUE,times=times,sta

#-------------------------------------------------------------------------------
# Find the rule correspondint to a cutpoint eta=0

ooo=level.set(0,outxr1,pred=TRUE,xpred=x)

print(ooo$cutpoints)
#-------------------------------------------------------------------------------
# EXAMPLES OF PLOTTING FUNCTIONS
# Plot the regression function

plotXRreg(y,x,outxr1,allpoints=FALSE) # only show active points
plotXRreg(y,x,outxr1,allpoints=TRUE) # show all data points

# Plot the inverse regression function

plotXRinvreg(x,outxr1,shading=TRUE,quant=TRUE) # plot against quantiles
# Plot against regression function (but put quantiles on upper horizontal axis
plotXRinvreg(x,outxr1,shading=TRUE,quant=FALSE,both=TRUE,qlev=(1:9)/10)

#-------------------------------------------------------------------------------
# STEPWISE BUIDING FOR XR MODEL
# this example builds models on martingale residuals

outstepxr1=stepXR(y,x,ypred=y,xpred=x,niterfull=10,niter=5,step.size=.3,kstep=8,kbest=4,pena

# pick model with best GCV
varlist=outstepxr1$steplist[[outstepxr1$bestgcv]]
# fit final model with larger number of interations.
outxrs=XReg(y=y,x=x,varlist=varlist,step.adapt=TRUE,niter=20,limsize=.05,step.size=.5,
            survival=FALSE)
```

```
# Get predicted values from the model
etaxr=predictfun(xpred=x,varlist=varlist,outxrs$beta0list)$predicted

# get rules for upper 90
levelout=level.set(quantile(etaxr,.90),outxrs,pred=TRUE,xpred=x)
print(levelout$cutpoints)

# These are the observations corresponding to that group
group=levelout$selectobs
```

Index