Package ‘quantregForest’

Type Package
Title Quantile Regression Forests
Version 1.0
Date 2015-06-03
Author Nicolai Meinshausen, Lukas Schiesser
Maintainer Nicolai Meinshausen <meinshausen@stat.math.ethz.ch>
Depends randomForest
Imports stats, graphics, grDevices
Suggests gss
Description Quantile Regression Forests is a tree-based ensemble
method for estimation of conditional quantiles. It is
particularly well suited for high-dimensional data. Predictor
variables of mixed classes can be handled. The package is
dependent on the package randomForests, written by Andy Liaw.
License GPL
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-06-03 18:23:21

R topics documented:

importance .................................................. 2
plot.quantregForest ....................................... 3
predict.quantregForest ................................. 5
quantregForest ........................................... 6
varImpPlot.qrf ............................................ 8

Index 11
importance

| importance | Extract variable importance measure |

Description

This is the extractor function for variable importance measures as produced by `quantregForest`.

Usage

```r
## S3 method for class 'quantregForest'
importance(x, quantiles = x$quantiles, ...)
```

Arguments

- `x`: an object of class `quantregForest`
- `quantiles`: A vector of quantiles (with numerical values in [0,1]) for which the variable importance measure should be extracted. Only quantiles for which the measure already has been computed with `quantregForest` are allowed
- `...`: not used

Details

The variable importance measure is computed from permuting OOB data: First, the prediction error on out-of-bag portion of data is recorded (quantile loss function). Then the same is done after permuting each predictor variable. The differences between the two are then averaged over the number of out-of-bag data. This is done for each quantile separately specified by the input `quantiles` in `quantregForest`.

Value

A matrix of importance measure, one row for each predictor variable and one column for each quantile.

Author(s)

Lukas Schiesser

See Also

`quantregForest`, `varImpPlot.qrf`
Examples

```r
# Load air-quality data (and preprocessing)
data(airquality)
set.seed(1)

# remove observations with missing values
airquality <- airquality[!apply(is.na(airquality), 1, any), ]

# number of remaining samples
n <- nrow(airquality)

# divide into training and test data
indextrain <- sample(1:n, round(0.6*n), replace=FALSE)
xtrain <- airquality[indextrain, 2:6]
xtest <- airquality[-indextrain, 2:6]
ytrain <- airquality[indextrain]
ytest <- airquality[-indextrain]

# compute Quantile Regression Forests
qrf <- quantregForest(x=xtrain, y=ytrain, importance=TRUE)

# look at computed importance measure of predictors
importance(qrf)

# print the measure only for one quantile
importance(qrf, quantiles=0.5)
```

---

**Description**

Plots the 90%-prediction intervals on out-of-bag data for a given object of class `quantregForest`.

**Usage**

```r
# S3 method for class 'quantregForest'
plot(x, all = FALSE, obs = 1, ...)
```
Arguments

- **x**: An object of class `quantregForest`
- **all**: A logical value. `all=TRUE` uses all observations for prediction. `all=FALSE` uses only a certain number of observations per node for prediction (set with argument `obs`). The default is `all=FALSE`
- **obs**: An integer number. Determines the maximal number of observations per node used for prediction. The input is ignored for `all=TRUE`. The default is `obs=1`
- **...**: Further arguments (not in use in the current version)

Author(s)

Nicolai Meinshausen, Lukas Schiesser

See Also

`quantregForest`, `predict.quantregForest`

Examples

data(airquality)
set.seed(1)

## remove observations with missing values
airquality <- airquality[!apply(is.na(airquality), 1, any), ]

## number of remaining samples
n <- nrow(airquality)

## divide into training and test data
indextrain <- sample(1:n, round(0.6*n), replace=FALSE)
Xtrain <- airquality[indextrain,2:6]
Xtest <- airquality[-indextrain,2:6]
Ytrain <- airquality[indextrain,1]
Ytest <- airquality[-indextrain,1]

qrf <- quantregForest(x=Xtrain, y=Ytrain)

## plot out-of-bag predictions for the training data
plot(qrf)
**predict.quantregForest**

*Prediction method for class quantregForest*

**Description**

Prediction of test data with quantile regression forests.

**Usage**

```r
## S3 method for class 'quantregForest'
predict(object, newdata = NULL, quantiles = c(0.1, 0.5, 0.9), all = FALSE, obs = 1, ...)
```

**Arguments**

- `object`: An object of class quantregForest
- `newdata`: A data frame or matrix containing new data. If not given, the out-of-bag prediction in `object` is returned
- `quantiles`: A vector of quantiles (with numerical values in [0,1]) for which the quantile estimates should be returned
- `all`: A logical value. `all=TRUE` uses all observations for prediction. `all=FALSE` uses only a certain number of observations per node for prediction (set with argument `obs`). The default is `all=FALSE`
- `obs`: An integer number. Determines the maximal number of observations per node to use for prediction. The input is ignored for `all=TRUE`. The default is `obs=1`
- `...`: Further arguments (not in use in the current version)

**Details**

It might be useful to try `all=TRUE` if the function is slow when given a high number of observations to fit the model and predicting the quantiles only for few new sample points.

**Value**

A matrix. The first column contains the conditional quantile estimates for the first entry in the vector `quantiles`. The second column contains the estimates for the second entry of `quantiles` and so on.

**Author(s)**

Nicolai Meinshausen, Lukas Schiesser

**See Also**

`quantregForest`, `predict.quantregForest`
Examples

```r
data(airquality)
set.seed(1)

## remove observations with missing values
airquality <- airquality[ !apply(is.na(airquality), 1, any), ]

## number of remaining samples
n <- nrow(airquality)

## divide into training and test data
indextrain <- sample(1:n, round(0.6*n), replace=FALSE)
xtrain <- airquality[ indextrain, 2:6]
xtest <- airquality[-indextrain, 2:6]
ytrain <- airquality[ indextrain, 1]
ytest <- airquality[-indextrain, 1]

qrf <- quantregforest(x=xtrain, y=ytrain)

## plot out-of-bag predictions for the training data
plot(qrf)

## compute out-of-bag predictions
quant.outofbag <- predict(qrf)

## predict test data using all observations per node for prediction
quant.newdata <- predict(qrf, newdata= xtest, all=TRUE)
```

**quantregForest**  
**Quantile Regression Forests**

Description

Quantile Regression Forests infer conditional quantile functions from data

Usage

```r
quantregforest(x,y, mtry= ceiling(ncol(x)/3) , nodesize= 10, ntree= 100,
importance=FALSE ,quantiles=c(0.1,0.5,0.9))
```

Arguments

- **x** A matrix or data.frame containing the predictor variables
- **y** The response variable; a numerical vector
- **mtry** The number of variables to try for each split; same default setting as for Random Forests
quantregForest

### Parameters

- **nodesize**: The minimal number of instances in each terminal node; the default setting is slightly higher than for Random Forests.
- **ntree**: The number of trees to be grown.
- **importance**: If `true`, importance of predictors is assessed.
- **quantiles**: A vector of quantiles (with numerical values in [0,1]) for which the variable importance measure should be computed. (Ignored if `importance=FALSE`)

### Details

It might be useful to try various values of `mtry` and see which one works best; however, results are typically not heavily dependent on this parameter.

### Value

A value of class `quantregForest`, for which `print`, `plot`, `predict` and `importance` methods are available. Class `quantregForest` is a list of the following components additional to the ones given by class `randomForest`:

- **call**: the original call to `quantregForest`.
- **origNodes**: the nodes for the original data in each tree.
- **origObs**: the values of the response variable used to fit the tree.
- **importance**: if `importance=TRUE` a matrix. The first column contains the mean increase in prediction error for the first entry in the vector `quantiles`, the second column contains the same for the second entry of `quantiles` and so on.
- **quantiles**: list of quantiles for which the variable importance measure was computed if `importance=TRUE`.

### Author(s)

Nicolai Meinshausen, Lukas Schiesser

### References


### See Also

`predict.quantregForest`, `plot.quantregForest`, `importance.quantregForest`, `varImpPlot.qrf`

### Examples

```r
################################################
## load air-quality data (and preprocessing) ##
################################################

data(airquality)
s.set.seed(1)
```
## remove observations with missing values
`airquality <- airquality[!apply(is.na(airquality), 1, any),]`

## number of remaining samples
`n <- nrow(airquality)`

## divide into training and test data
`indextrain <- sample(1:n, round(0.6*n), replace=FALSE)`
`xtrain <- airquality[indextrain,2:6]`
`xtest <- airquality[-indextrain,2:6]`
`ytrain <- airquality[indextrain,1]`
`ytest <- airquality[-indextrain,1]`

### compute Quantile Regression Forests

```r
qrf <- quantregForest(x=xtrain, y=ytrain, importance=TRUE)
```

## plot out-of-bag predictions for the training data
`plot(qrf)`

## compute out-of-bag predictions
`quant.outofbag <- predict(qrf)`

## predict test data using all observations per node for prediction
`quant.newdata <- predict(qrf, newdata=xtest, all=TRUE)`

## look at computed importance measure of predictors
`importance(qrf)`

---

**varImpPlot.qrf**

**Variable Importance Plot**

### Description

Dotchart of variable importance as measured by a Quantile Regression Forest

### Usage

```r
varImpPlot.qrf(x, quantiles=x$quantiles, symbols=TRUE, color=TRUE, sort=TRUE, which.sort=1, n.var=min(30, nrow(x$importance)), main=deparse(substitute(x)), ...)
```
Arguments

- **x**: An object of class `quantregForest`.
- **quantiles**: A vector of quantiles (with numerical values in [0,1]) for which the variable importance measure should be extracted. Only quantiles for which the measure already has been computed with `quantregForest` are allowed.
- **symbols**: A logical value. If TRUE symbols are used to distinguish the values for the different quantiles.
- **color**: A logical value. If TRUE colors are used to distinguish the values for the different quantiles.
- **sort**: Should the variables be sorted in decreasing order of importance?
- **which.sort**: A number between 1 and length(quantiles). Specifies which quantile is used to order the predictors for the plot (Ignored if sort=FALSE).
- **n.var**: How many variables to show? (Ignored if sort=FALSE).
- **main**: Plot title.
- **...**: Other graphical parameters to be passed on to `dotchart`.

Value

Invisibly, the importance of the variables that were plotted.

Author(s)

Lukas Schiesser

See Also

`quantregForest`, `importance.quantregForest`

Examples

```r
# Load air-quality data (and preprocessing)

#--------------------------------------------------------------
# Load air-quality data (and preprocessing) #
#--------------------------------------------------------------

data(airquality)
set.seed(1)

# remove observations with missing values
airquality <- airquality[!apply(is.na(airquality), 1, any), ]

# number of remaining samples
n <- nrow(airquality)

# divide into training and test data
indextrain <- sample(1:n, round(0.6*n), replace=FALSE)
Xtrain    <- airquality[indextrain, 2:6]
```
Xtest <- airquality[-indextrain,2:6]
Ytrain <- airquality[indextrain,1]
Ytest <- airquality[-indextrain,1]

################################################
## compute quantile regression forests
################################################
qrf <- quantregforest(x=Xtrain, y=Ytrain, importance=TRUE)

## produce a simple variable importance plot
varImpPlot.qrf(qrf)

## plot only one quantile
varImpPlot.qrf(qrf, quantile=0.5)

## use second quantile to order values
varImpPlot.qrf(qrf, which.sort=2)
Index

* Topic regression
  quantregForest, 6
* Topic tree
  quantregForest, 6

dotchart, 9

importance, 2
importance.quantregForest, 7, 9

plot.quantregForest, 3, 7
predict.quantregForest, 4, 5, 5, 7

quantregForest, 2, 4, 5, 6, 9

varImpPlot.qrf, 2, 7, 8