Package 'upre'

July 26, 2016

Title What the Package Does (one line, title case)

Version 0.0.0.9000

Description What the package does (one paragraph).

Depends R (>= 3.2.4), akima, partykit, glmnet, Formula, Matrix, caret

License GPL-2 | GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 5.0.1

R topics documented:

Description

bsnullinteract calculates null interaction models on bootstrapped datasets, for deriving a reference distribution of the test statistic calculated with interact

Usage

```
bsnullinteract(object, nsamp = 10, seed = 42)
```

Arguments

object	an object of class upre
nsamp	the number of bootstrapped null interaction models to be derived
seed	numeric. Random seed to be used in deriving the final ensemble (for reproducability). Defaults to 42.

Details

Can be computationally intensive.

Value

A list of null interactrion models

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data=airquality[complete.cases(airquality),])
 nullmods <- bsnullinteract(airq.ens)

## End(Not run)
```

`coef.upre` *Coefficients for the final prediction rule ensemble*

Description

`coef.upre` gets coefficients for each of the prediction rules and linear terms in the ensemble

Usage

```
## S3 method for class 'upre'
coef(object, penalty.par.val = "lambda.min", print = TRUE,
  ...)
```

Arguments

`object`	an object resulting from application of `upre()`
`penalty.par.val`	character. Should model be selected with lambda giving minimum cv error ("lambda.min"), or lambda giving cv error that is within 1 standard error of the minimum cv error ("lambda.1se")?
`print`	logical. Should the coefficients of the base learners with non-zero coeffcients in the final ensemble be printed to the command line?
`...`	additional arguments to be passed to `coef.glmnet`.

Value

returns a dataframe with 3 columns: coefs (coefficients), rule (rule or variable name) and descriptions (<NA> for linear terms, conditions for rules). In the command line, the non zero coefficients are printed

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data=airquality[complete.cases(airquality),])
 coef(airq.ens)

## End(Not run)
```

importance — *Calculate importances of base learners (rules and linear terms) and input variables*

Description

`importance` calculates importances for rules, linear terms and input variables in the ensemble, and provides a bar plot of variable importances

Usage

```
importance(object, plot = TRUE, ylab = "Importance",
  main = "Variable importances", ...)
```

Arguments

`object`	an object of class `upre`
`plot`	logical. Should variable importances be plotted?
`ylab`	character. Only used when `plot = TRUE`. PLotting label for y-axis.
`...`	further arguments to be passed to `barplot`

Value

A list with two dataframes: $baseimps, giving the importance for each baselearner (not) in the ensemble, and $varimps, giving the importance for each predictor variable (not) in the ensemble

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data=airquality[complete.cases(airquality),])
 importance(airq.ens)

## End(Not run)
```

interaction — *Calculate interaction test statistic for user-specified variable*

Description

`interaction` calculates a statistic for testing whether a user-supplied variable interacts with any other variable in the ensemble.

Usage

```
interaction(object, varname, nullmods = NULL, k = 10)
```

Arguments

object	an object of class upre
varname	character. Variable for which interation test statistic should be calculated.
nullmods	object with bootstrapped null interaction models, resulting from application of bsnullinteract.
k	integer. Calculating interaction test statistics is a computationally intensive, so calculations are split up in several parts to prevent memory allocation errors. If a memory allocation error still occurs, increase k.

Details

Can be computationally intensive, especially when nullmods is specified.

Value

If nullmods is not specified: the test statistic of the interaction strength If nullmods is specified: $H = the test statistic of the interaction strength $nullH = a vector of test statistics of the interaction strength for each of the bootstrapped null interaction models

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data=airquality[complete.cases(airquality),])
 interaction(airq.ens, "Temp")

## End(Not run)
```

---

pairplot *Create partial dependence plot for a pair of predictor variables*

Description

pairplot generates a partial dependence plot to assess the effects of a pair of predictor variables, on the predictions of the ensemble

Usage

```
pairplot(object, varnames, penalty.par.val = "lambda.min", phi = 45,
  theta = 315, col = "cyan", nvals = NULL)
```

Arguments

object	an object of class upre
varnames	character vector of length two
penalty.par.val	character. Should model be selected with lambda giving minimum cv error ("lambda.min"),
phi	numeric. See persp() documentation.
theta	numeric. See persp() documentation.
col	character. Optional color to be used for surface in 3D plot.
nvals	optional numeric vector or length two. For how many values of x1 and x2 should partial dependence be plotted?

Details

By default, partial dependence will be plotted for each combination of unique observed values of the specified predictor variables. When the number of unique observed values is large, this may take a long time to compute. Specifying the nvals argument can substantially reduce computing time. When the nvals argument is supplied, values for the minimum, maximum, and nvals - 2 intermediate values of the predictor variable will be plotted.

Providing the names of two variables that do not appear in the final prediction rule ensemble will result in an error.

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data = airquality[complete.cases(airquality),])
 pairplot(airq.ens, c("Temp", "Wind"))

## End(Not run)
```

---

predict.upre *Predicted values based on the final prediction rule ensemble*

Description

`predict.upre` generates predictions based on the final ensemble for training, or for new test observations

Usage

```
## S3 method for class 'upre'
predict(object, newdata = NULL,
  penalty.par.val = "lambda.min", ...)
```

Arguments

`object`	an object resulting from application of final()
`newdata`	optional dataframe of new observations, including all predictor variables used to generate the initial ensemble
`penalty.par.val`	character. Should model be selected with lambda giving minimum cv error ("lambda.min"),
`...`	additional arguments to be passed (currently not used).

Details

When newdata is not provided, training data included in the specified object is used.

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data = airquality[complete.cases(airquality),])
 predict(airq.ens, newdata = airquality[complete.cases(airquality),])
 predict(airq.ens)

## End(Not run)
```

singleplot *Create partial dependence plot for a single variable*

Description

singleplot generates a partial dependence plot to assess the effect of a single predictor variable, on the predictions of the ensemble

Usage

```
singleplot(object, varname, penalty.par.val = "lambda.min", nvals = NULL)
```

Arguments

object	an object of class upre
varname	character vector of length one, specifying the variable for which the partial dependence plot should be created.
penalty.par.val	character. Should model be selected with lambda giving minimum cv error ("lambda.min"),
nvals	optional numeric vector or length one. For how many values of x should the partial dependence plot be created?

Details

By default, a partial dependence plot will be created for each unique observed value of the specified predictor variable. When the number of unique observed values is large, this may take a long time to compute. Specifying the nvals argument can substantially reduce computing time. When the nvals argument is supplied, values for the minimum, maximum, and nvals - 2 intermediate values of the predictor variable will be plotted. Providing the name of a variable that does not appear in the final prediction rule ensemble will result in an error.

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data = airquality[complete.cases(airquality),])
 singleplot(airq.ens, "Temp")

## End(Not run)
```

upre *Derive an unbiased prediction rule ensemble*

Description

upre derives a sparse ensemble of rules and/or linear functions for prediction

Usage

```
upre(formula, data, type = "both", weights = rep(1, times = nrow(data)),
  sampfrac = 0.5, ntrees = 500, seed = 42, maxdepth = 3,
  learnrate = 0.01, removeduplicates = TRUE, maxrules = 2000, alpha = 1,
  dfmax = 5 * ncol(data), mtry = Inf, thres = 1e-07,
  standardize = FALSE, winsfrac = 0.025, normalize = TRUE, nfolds = 10,
  mod.sel.crit = "deviance", verbose = TRUE)
```

Arguments

formula	regression formula; a symbolic description of the model to be fit.
data	matrix or data frame containing the variables in the model.
type	character. Type of base learners to be included in ensemble. Defaults to "both" (intial ensemble included both rules and linear functions). Other option may be "rules" (for prediction rules only) or "linear" (for linear functions only).
weights	an optional vector of observation weights to be used for deriving the ensemble.
sampfrac	numeric value greater than 0, and smaller than or equal to 1. Fraction of randomly selected training observations used to produce each tree. Setting this to values < 1 will result in subsamples being drawn without replacement (i.e., subsampling). Setting this equal to 1 will result in bootstrap sampling.
ntrees	numeric. Total number of trees to be generated.
seed	numeric. Random seed to be used in deriving the final ensemble (for reproducability). Defaults to 42.
maxdepth	numeric. Maximal depth of trees to be grown. Defaults to 3, resulting in trees with max 15 nodes (8 terminal and 7 inner nodes), and therefore max 15 rules.
learnrate	numeric. Learning rate for sequentially induced trees.
removeduplicates	logical. Remove rules from the ensemble which have the exact same support in training data?
maxrules	numeric. Approximate maximum number of rules to be generated. The number of rules in the final ensemble will be smaller, due to the omission of rules with identical conditions or support.
alpha	numeric. Elastic net mixing parameter.
dfmax	numeric. Maximal number of terms in the final model.
mtry	numeric. Number of randomly selected predictor variables for creating each split in each tree.
thres	numeric. Threshold for convergence.
standardize	logical. Standardize rules and predictor variables before estimating the regression model?
winsfrac	numeric. Quantiles of data distribution to be used for winsorizing linear predictors. When set to 0, no winsorizing is performed.
normalize	logical. Normalize linear variables before estimating the regression model? Normalizing gives linear terms the same a priori influence as a typical rule.
nfolds	numeric. Number of folds to be used in performing cross validation for determining penalty parameter

mod.sel.crit	character. Model selection criterion to be used for deriving the final ensemble. The default is type.measure="deviance", which uses squared-error for gaussian models (a.k.a type.measure="mse"). type.measure="mse" or type.measure="mae" (mean absolute error) measure the deviation from the fitted mean to the response.
verbose	logical. Should information on the initial and final ensemble be printed to the command line?

Details

Note that variable names supplied in the formula may not start with the word 'rule'

Value

a list with many elements

Examples

```
## Not run:
 airq.ens <- upre(Ozone ~ ., data=airquality[complete.cases(airquality),])

## End(Not run)
```