Package 'RRBoost'

October 12, 2022

Type Package

Title A Robust Boosting Algorithm

Version 0.1

Date 2020-09-28

Description An implementation of robust boosting algorithms for regression in R. This includes the RRBoost method proposed in the paper ``Robust Boosting for Regression Problems" (Ju X and Salibian-Barrera M. 2020) <doi:10.1016/j.csda.2020.107065> (to appear in Computational Statistics and Data Science). It also implements previously proposed boosting algorithms in the simulation section of the paper: L2Boost, LADBoost, MBoost (Friedman, J. H. (2001) <doi:10.1214/aos/1013203451>) and Robloss (Lutz et al. (2008) <doi:10.1016/j.csda.2007.11.006>).

Depends R (>= 3.5.0)

Imports stats, rpart, RobStatTM

License GPL (>= 3)

LazyData True

RoxygenNote 7.1.1

Encoding UTF-8

NeedsCompilation no

Author Xiaomeng Ju [aut, cre],
Matias Salibian-Barrera [aut]

Maintainer Xiaomeng Ju <xiaomeng.ju@stat.ubc.ca>

Repository CRAN

Date/Publication 2020-10-13 13:10:06 UTC

R topics documented:

`airfoil` *Airfoil data*

Description

Here goes a description of the data.

Usage

```
data(airfoil)
```

Format

An object of class `"data.frame"`.

Details

Here goes a more detailed description of the data. There are 1503 observations and 6 variables: `y`, `frequency`, `angle`, `chord_length`, `velocity`, and `thickness`.

Source

The UCI Archive https://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise,

References

Brooks, T. F., Pope, D. S., and Marcolini, M. A. (1989). Airfoil self-noise and prediction. NASA Reference Publication-1218, document id: 9890016302.

Examples

```
data(airfoil)
```

`Boost` *Robust Boosting for regression*

Description

This function implements the RRBoost robust boosting algorithm for regression, as well as other robust and non-robust boosting algorithms for regression.

Usage

```
Boost(
  x_train,
  y_train,
  x_val,
  y_val,
  x_test,
  y_test,
  type = "RRBoost",
  error = c("rmse", "aad"),
  niter = 200,
  y_init = "LADTree",
  max_depth = 1,
  tree_init_provided = NULL,
  control = Boost.control()
)
```

Arguments

x_train	predictor matrix for training data (matrix/dataframe)
y_train	response vector for training data (vector/dataframe)
x_val	predictor matrix for validation data (matrix/dataframe)
y_val	response vector for validation data (vector/dataframe)
x_test	predictor matrix for test data (matrix/dataframe, optional, required when make_prediction in control is TRUE)
y_test	response vector for test data (vector/dataframe, optional, required when make_prediction in control is TRUE)
type	type of the boosting method: "L2Boost", "LADBoost", "MBoost", "Robloss", "SBoost", "RRBoost" (character string)
error	a character string (or vector of character strings) indicating the type of error metrics to be evaluated on the test set. Valid options are: "rmse" (root mean squared error), "aad" (average absolute deviation), and "trmse" (trimmed root mean squared error)
niter	number of boosting iterations (for RRBoost: T_1,max + T_2,max) (numeric)
y_init	a string indicating the initial estimator to be used. Valid options are: "median" or "LADTree" (character string)
max_depth	the maximum depth of the tree learners (numeric)
tree_init_provided	an optional pre-fitted initial tree (an rpart object)
control	a named list of control parameters, as returned by Boost.control

Details

This function implements a robust boosting algorithm for regression (RRBoost). It also includes the following robust and non-robust boosting algorithms for regression: L2Boost, LADBoost, MBoost,

Robloss, and SBoost. This function uses the functions available in the `rpart` package to construct binary regression trees.

Value

A list with the following components:

`type`	which boosting algorithm was run. One of: "L2Boost", "LADBoost", "MBoost", "Robloss", "SBoost", "RRBoost" (character string)
`control`	the list of control parameters used
`niter`	number of iterations for the boosting algorithm (for RRBoost T_1,max + T_2,max) (numeric)
`error`	if `make_prediction = TRUE` in argument `control`, a vector of prediction errors evaluated on the test set at early stopping time. The length of the vector matches that of the `error` argument in the input.
`tree_init`	if `y_init = "LADTree"`, the initial tree (an object of class `rpart`)
`tree_list`	if `save_tree = TRUE` in `control`, a list of trees fitted at each boosting iteration
`f_train_init`	a vector of the initialized estimator of the training data
`alpha`	a vector of base learners' coefficients
`early_stop_idx`	early stopping iteration
`when_init`	if `type = "RRBoost"`, the early stopping time of the first stage of RRBoost
`loss_train`	a vector of training loss values (one per iteration)
`loss_val`	a vector of validation loss values (one per iteration)
`err_val`	a vector of validation aad errors (one per iteration)
`err_train`	a vector of training aad errors (one per iteration)
`err_test`	a matrix of test errors before and at the early stopping iteration (returned if make_prediction = TRUE in control); the matrix dimension is the early stopping iteration by the number of error types (matches the `error` argument in the input); each row corresponds to the test errors at each iteration
`f_train`	a matrix of training function estimates at all iterations (returned if save_f = TRUE in control); each column corresponds to the fitted values of the predictor at each iteration
`f_val`	a matrix of validation function estimates at all iterations (returned if save_f = TRUE in control); each column corresponds to the fitted values of the predictor at each iteration
`f_test`	a matrix of test function estimatesbefore and at the early stopping iteration (returned if save_f = TRUE and make_prediction = TRUE in control); each column corresponds to the fitted values of the predictor at each iteration
`var_select`	a vector of variable selection indicators (one per explanatory variable; 1 if the variable was selected by at least one of the base learners, and 0 otherwise)
`var_importance`	a vector of permutation variable importance scores (one per explanatory variable, and returned if cal_imp = TRUE in control)

Author(s)

Xiaomeng Ju, <xmengju@stat.ubc.ca>

See Also

Boost.validation, Boost.control.

Examples

```
data(airfoil)
n <- nrow(airfoil)
n0 <- floor( 0.2 * n )
set.seed(123)
idx_test <- sample(n, n0)
idx_train <- sample((1:n)[-idx_test], floor( 0.6 * n ) )
idx_val <- (1:n)[ -c(idx_test, idx_train) ]
xx <- airfoil[, -6]
yy <- airfoil$y
xtrain <- xx[ idx_train, ]
ytrain <- yy[ idx_train ]
xval <- xx[ idx_val, ]
yval <- yy[ idx_val ]
xtest <- xx[ idx_test, ]
ytest <- yy[ idx_test ]
model_RRBoost_LADTree = Boost(x_train = xtrain, y_train = ytrain,
    x_val = xval, y_val = yval, x_test = xtest, y_test = ytest,
    type = "RRBoost", error = "rmse", y_init = "LADTree",
    max_depth = 1, niter = 10, ## to keep the running time low
    control = Boost.control(max_depth_init = 2,
    min_leaf_size_init = 20, make_prediction =  TRUE,
    cal_imp = FALSE))
```

Boost.control *Tuning and control parameters for the robust boosting algorithm*

Description

Tuning and control parameters for the RRBoost robust boosting algorithm, including the initial fit.

Usage

```
Boost.control(
  n_init = 100,
  eff_m = 0.95,
  bb = 0.5,
  trim_prop = NULL,
  trim_c = 3,
  max_depth_init = 3,
```

```
  min_leaf_size_init = 10,
  cal_imp = TRUE,
  save_f = FALSE,
  make_prediction = TRUE,
  save_tree = FALSE,
  precision = 4,
  shrinkage = 1,
  trace = FALSE
)
```

Arguments

n_init	number of iterations for the SBoost step of RRBoost (T_1,max) (int)
eff_m	scalar between 0 and 1 indicating the efficiency (measured in a linear model with Gaussian errors) of Tukey's loss function used in the 2nd stage of RRBoost.
bb	breakdown point of the M-scale estimator used in the SBoost step (numeric)
trim_prop	trimming proportion if 'trmse' is used as the performance metric (numeric). 'trmse' calculates the root-mean-square error of residuals (r) of which |r| < quantile(|r|, 1-trim_prop) (e.g. trim_prop = 0.1 ignores 10% of the data and calculates RMSE of residuals whose absolute values are below 90% quantile of |r|). If both trim_prop and trim_c are specified, trim_c will be used.
trim_c	the trimming constant if 'trmse' is used as the performance metric (numeric, defaults to 3). 'trmse' calculates the root-mean-square error of the residuals (r) between median(r) + trim_c mad(r) and median(r) - trim_c mad(r). If both trim_prop and trim_c are specified, trim_c will be used.
max_depth_init	the maximum depth of the initial LADTtree (numeric, defaults to 3)
min_leaf_size_init	the minimum number of observations per node of the initial LADTtree (numeric, defaults to 10)
cal_imp	logical indicating whether to calculate variable importance (defaults to TRUE)
save_f	logical indicating whether to save the function estimates at all iterations (defaults to FALSE)
make_prediction	logical indicating whether to make predictions using x_test (defaults to TRUE)
save_tree	logical indicating whether to save trees at all iterations (defaults to FALSE)
precision	number of significant digits to keep when using validation error to calculate early stopping time (numeric, defaults to 4)
shrinkage	shrinkage parameter in boosting (numeric, defaults to 1 which corresponds to no shrinkage)
trace	logical indicating whether to print the number of completed iterations and for RRBoost the completed combinations of LADTree hyperparameters for monitoring progress (defaults to FALSE)

Details

Various tuning and control parameters for the RRBoost robust boosting algorithm implemented in the function Boost, including options for the initial fit.

Value

A list of all input parameters

Author(s)

Xiaomeng Ju, <xmengju@stat.ubc.ca>

Examples

```
data(airfoil)
n <- nrow(airfoil)
n0 <- floor( 0.2 * n )
set.seed(123)
idx_test <- sample(n, n0)
idx_train <- sample((1:n)[-idx_test], floor( 0.6 * n ) )
idx_val <- (1:n)[ -c(idx_test, idx_train) ]
xx <- airfoil[, -6]
yy <- airfoil$y
xtrain <- xx[ idx_train, ]
ytrain <- yy[ idx_train ]
xval <- xx[ idx_val, ]
yval <- yy[ idx_val ]
xtest <- xx[ idx_test, ]
ytest <- yy[ idx_test ]
my.control <- Boost.control(max_depth_init = 2,
    min_leaf_size_init = 20, make_prediction =  TRUE,
    cal_imp = FALSE)
model_RRBoost_LADTree = Boost(x_train = xtrain, y_train = ytrain,
    x_val = xval, y_val = yval, x_test = xtest, y_test = ytest,
    type = "RRBoost", error = "rmse", y_init = "LADTree",
    max_depth = 1, niter = 10, ## to keep the running time low
    control = my.control)
```

`Boost.validation` *Robust Boosting for regression with initialization parameters chosen on a validation set*

Description

A function to fit RRBoost (see also `Boost`) where the initialization parameters are chosen based on the performance on the validation set.

Usage

```
Boost.validation(
  x_train,
  y_train,
  x_val,
```

```
  y_val,
  x_test,
  y_test,
  type = "RRBoost",
  error = c("rmse", "aad"),
  niter = 1000,
  max_depth = 1,
  y_init = "LADTree",
  max_depth_init_set = c(1, 2, 3, 4),
  min_leaf_size_init_set = c(10, 20, 30),
  control = Boost.control()
)
```

Arguments

x_train	predictor matrix for training data (matrix/dataframe)
y_train	response vector for training data (vector/dataframe)
x_val	predictor matrix for validation data (matrix/dataframe)
y_val	response vector for validation data (vector/dataframe)
x_test	predictor matrix for test data (matrix/dataframe, optional, required when make_prediction in control is TRUE)
y_test	response vector for test data (vector/dataframe, optional, required when make_prediction in control is TRUE)
type	type of the boosting method: "L2Boost", "LADBoost", "MBoost", "Robloss", "SBoost", "RRBoost" (character string)
error	a character string (or vector of character strings) indicating the types of error metrics to be evaluated on the test set. Valid options are: "rmse" (root mean squared error), "aad" (average absulute deviation), and "trmse" (trimmed root mean squared error)
niter	number of iterations (for RRBoost T_1,max + T_2,max) (numeric)
max_depth	the maximum depth of the tree learners (numeric)
y_init	a string indicating the initial estimator to be used. Valid options are: "median" or "LADTree" (character string)
max_depth_init_set	a vector of possible values of the maximum depth of the initial LADTree that the algorithm choses from
min_leaf_size_init_set	a vector of possible values of the minimum observations per node of the initial LADTree that the algorithm choses from
control	a named list of control parameters, as returned by Boost.control

Details

This function runs the RRBoost algorithm (see Boost) on different combinations of the parameters for the initial fit, and chooses the optimal set based on the performance on the validation set.

Value

A list with components

`the components of model`
: an object returned by Boost that is trained with selected initialization parameters

`param`
: a vector of selected initialization parameters (return (0,0) if selected initialization is the median of the training responses)

Author(s)

Xiaomeng Ju, <xmengju@stat.ubc.ca>

See Also

Boost, Boost.control.

Examples

```
data(airfoil)
n <- nrow(airfoil)
n0 <- floor( 0.2 * n )
set.seed(123)
idx_test <- sample(n, n0)
idx_train <- sample((1:n)[-idx_test], floor( 0.6 * n ) )
idx_val <- (1:n)[ -c(idx_test, idx_train) ]
xx <- airfoil[, -6]
yy <- airfoil$y
xtrain <- xx[ idx_train, ]
ytrain <- yy[ idx_train ]
xval <- xx[ idx_val, ]
yval <- yy[ idx_val ]
xtest <- xx[ idx_test, ]
ytest <- yy[ idx_test ]
model_RRBoost_cv_LADTree = Boost.validation(x_train = xtrain,
      y_train = ytrain, x_val = xval, y_val = yval,
      x_test = xtest, y_test = ytest, type = "RRBoost", error = "rmse",
      y_init = "LADTree", max_depth = 1, niter = 1000,
      max_depth_init_set = 1:5,
      min_leaf_size_init_set = c(10,20,30),
      control = Boost.control(make_prediction =  TRUE,
      cal_imp = TRUE))
```

cal_imp_func *Variable importance scores for the robust boosting algorithm RRBoost*

Description

This function calculates variable importance scores for a previously computed RRBoost fit.

Usage

```
cal_imp_func(model, x_val, y_val, trace = FALSE)
```

Arguments

model	an object returned by Boost
x_val	predictor matrix for validation data (matrix/dataframe)
y_val	response vector for validation data (vector/dataframe)
trace	logical indicating whether to print the variable under calculation for monitoring progress (defaults to FALSE)

Details

This function computes permutation variable importance scores given an object returned by Boost and a validation data set.

Value

a vector of permutation variable importance scores (one per explanatory variable)

Author(s)

Xiaomeng Ju, <xmengju@stat.ubc.ca>

Examples

```
data(airfoil)
n <- nrow(airfoil)
n0 <- floor( 0.2 * n )
set.seed(123)
idx_test <- sample(n, n0)
idx_train <- sample((1:n)[-idx_test], floor( 0.6 * n ) )
idx_val <- (1:n)[ -c(idx_test, idx_train) ]
xx <- airfoil[, -6]
yy <- airfoil$y
xtrain <- xx[ idx_train, ]
ytrain <- yy[ idx_train ]
xval <- xx[ idx_val, ]
yval <- yy[ idx_val ]
```

```
xtest <- xx[ idx_test, ]
ytest <- yy[ idx_test ]
model = Boost(x_train = xtrain, y_train = ytrain,
     x_val = xval, y_val = yval,
     type = "RRBoost", error = "rmse",
     y_init = "LADTree", max_depth = 1, niter = 1000,
     control = Boost.control(max_depth_init = 2,
           min_leaf_size_init = 20, save_tree = TRUE,
           make_prediction =  FALSE, cal_imp = FALSE))
var_importance <-  cal_imp_func(model, x_val = xval, y_val= yval)
```

`cal_predict` *cal_predict*

Description

A function to make predictions and calculate test error given an object returned by Boost and test data

Usage

```
cal_predict(model, x_test, y_test)
```

Arguments

`model`	an object returned by Boost
`x_test`	predictor matrix for test data (matrix/dataframe)
`y_test`	response vector for test data (vector/dataframe)

Details

A function to make predictions and calculate test error given an object returned by Boost and test data

Value

A list with with the following components:

`f_t_test`	predicted values with model at the early stopping iteration using x_test as the predictors
`err_test`	a matrix of test errors before and at the early stopping iteration (returned if make_prediction = TRUE in control); the matrix dimension is the early stopping iteration by the number of error types (matches the `error` argument in the input); each row corresponds to the test errors at each iteration
`f_test`	a matrix of test function estimates at all iterations (returned if save_f = TRUE in control)
`value`	a vector of test errors evaluated at the early stopping iteration

Author(s)

Xiaomeng Ju, <xmengju@stat.ubc.ca>

Examples

```
data(airfoil)
n <- nrow(airfoil)
n0 <- floor( 0.2 * n )
set.seed(123)
idx_test <- sample(n, n0)
idx_train <- sample((1:n)[-idx_test], floor( 0.6 * n ) )
idx_val <- (1:n)[ -c(idx_test, idx_train) ]
xx <- airfoil[, -6]
yy <- airfoil$y
xtrain <- xx[ idx_train, ]
ytrain <- yy[ idx_train ]
xval <- xx[ idx_val, ]
yval <- yy[ idx_val ]
xtest <- xx[ idx_test, ]
ytest <- yy[ idx_test ]
model = Boost(x_train = xtrain, y_train = ytrain,
     x_val = xval, y_val = yval,
     type = "RRBoost", error = "rmse",
     y_init = "LADTree", max_depth = 1, niter = 1000,
     control = Boost.control(max_depth_init = 2,
           min_leaf_size_init = 20, save_tree = TRUE,
           make_prediction =  FALSE, cal_imp = FALSE))
prediction <- cal_predict(model, x_test = xtest, y_test = ytest)
```

Index