Package 'parboost'

May 4, 2015

Title Distributed Model-Based Boosting

Version 0.1.4

Date 2015-05-03

Description Distributed gradient boosting based on the mboost package. The parboost package is designed to scale up component-wise functional gradient boosting in a distributed memory environment by splitting the observations into disjoint subsets, or alternatively using bootstrap samples (bagging). Each cluster node then fits a boosting model to its subset of the data. These boosting models are combined in an ensemble, either with equal weights, or by fitting a (penalized) regression model on the predictions of the individual models on the complete data.

Author Ronert Obst <ronert.obst@gmail.com>

Maintainer Ronert Obst <ronert.obst@gmail.com>

Depends R (>= 3.0.1), parallel, mboost, party, iterators

Imports plyr, caret, glmnet, doParallel

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2015-05-04 01:24:31

R topics documented:

`coef.parboost` *Print coefficients for base learners with a notion of coefficients*

Description

Extract coefficients from base learners that have a notion of coefficients

Usage

```
## S3 method for class 'parboost'
coef(object, which = NULL, aggregate = c("sum", "cumsum",
  "none"), ...)
```

Arguments

`object`	Object of class parboost.
`which`	Optionally a subset of base learners to evaluate as an integer or character vector.
`aggregate`	a character specifying how to aggregate predictions or coefficients of single base learners. The default returns the prediction or coefficient for the final number of boosting iterations. `"cumsum"` returns a matrix with the predictions for all iterations simultaneously (in columns). `"none"` returns a list with matrices where the *j*th columns of the respective matrix contains the predictions of the base learner of the *j*th boosting iteration (and zero if the base learner is not selected in this iteration).
`...`	Additional arguements passed to callies.

Details

Extract the coefficients of base learners which have a notion of coefficients from each boosting model. Weighs the coefficients by the postprocessed submodel weights.

Value

Returns a list of coefficients

Author(s)

Ronert Obst

References

T. Hothorn, P. Buehlmann, T. Kneib, M. Schmid, and B. Hofner (2013). mboost: Model-Based Boosting, R package version 2.2-3, http://CRAN.R-project.org/package=mboost.

`friedman2` *Benchmark Problem Friedman 2*

Description

Dataset taken from **mlbench**. The inputs are 4 independent variables uniformly distributed over the ranges

$$0 \leq x1 \leq 100$$

$$40\pi \leq x2 \leq 560\pi$$

$$0 \leq x3 \leq 1$$

$$1 \leq x4 \leq 11$$

The outputs are created according to the formula

$$y = (x1^2 + (x2x3 - (1/(x2x4)))^2)^{0.5} + e$$

where e is $N(0, sd)$.

Format

A data frame with 100 rows and 5 variables

Source

http://cran.r-project.org/web/packages/mlbench/index.html

References

Breiman, Leo (1996) Bagging predictors. Machine Learning 24, pages 123-140.

Friedman, Jerome H. (1991) Multivariate adaptive regression splines. The Annals of Statistics 19 (1), pages 1-67.

Friedrich Leisch & Evgenia Dimitriadou (2010). mlbench: Machine Learning Benchmark Problems. R package version 2.1-1.

parboost *Distributed gradient boosting based on the* **mboost** *package.*

Description

The parboost package implements distributed gradient boosting based on the **mboost** package. When should you use parboost instead of mboost? There are two use cases: 1. The data takes too long to fit as a whole 2. You want to bag and postprocess your boosting models to get a more robust ensemble parboost is designed to scale up component-wise functional gradient boosting in a distributed memory environment by splitting the observations into disjoint subsets. Alternatively, parboost can generate and use bootstrap samples of the original data. Each cluster node then fits a boosting model to its subset of the data. These boosting models are combined in an ensemble, either with equal weights, or by fitting a (penalized) regression model on the predictions of the individual models on the complete data. All other functionality of mboost is left untouched for the moment.

Distributed gradient boosting based on the **mboost** package. Gaussian, Binomial and Poisson families are currently supported.

Usage

```
parboost(cluster_object = NULL, mc.cores = NULL, data = NULL,
  path_to_data = "", data_import_function = NULL,
  split_data = c("disjoint", "bagging"), nsplits, preprocessing = NULL,
  seed = NULL, formula, baselearner = c("bbs", "bols", "btree", "bss",
  "bns"), family = c("gaussian", "binomial", "poisson"),
  control = boost_control(), tree_controls = NULL, cv = TRUE,
  cores_cv = detectCores(), folds = 8, stepsize_mstop = 1,
  postprocessing = c("none", "glm", "lasso", "ridge", "elasticnet"))
```

Arguments

`cluster_object`	Cluster object from the **parallel** package to carry out distributed computations.
`mc.cores`	If not `NULL`, `parboost` uses mclapply for shared memory parallelism.
`data`	A data frame containing the variables in the model. It is recommended to use path_to_data instead for IO efficiency. Defaults to NULL
`path_to_data`	A string pointing to the location of the data. `parboost` assumes that the data is located at the same location on every cluster node. This parameter is ignored if you pass a data frame to the data argument.
`data_import_function`	Function used to import data. Defaults to `read.csv`. This parameter is ignored if you pass a data frame to the data argument.
`split_data`	String determening the way the data should be split. `disjoint` splits the data into disjoint subsets. `bootstrap` draws a bootstrap sample instead.
`nsplits`	Integer determining the number of disjoint sets the data should be split into. If `split_data` is set to `bootstrap`, `nsplits` determines the number of bootstrap samples.

preprocessing	Optional preprocessing function to apply to the data. This is useful if you cannot modify the data on the cluster nodes.
seed	Integer determining the random seed value for reproducible results.
formula	Formula to be passed to mboost.
baselearner	Character string to determine the type of baselearner to be used for boosting. See mboost for details.
family	A string determining the family. Currently gaussian, binomial and poisson are implemented.
control	An object of type boost_control for controlling mboost. See boost_control in the mboost for details.
tree_controls	Optional object of type TreeControl. See ctree_control in the party documentation for detailos. Used to set hyperparameters for tree base learners.
cv	Logical to activate crossvalidation to determine m_{stop}. Defaults to TRUE.
cores_cv	Integer determining the number of CPU cores used for cross-validation on each node (or locally). Defaults to maximum available using detectCores.
folds	Integer determening the number of folds used during cross-validation on each cluster node. Defaults to 8. It is computationally more efficient to set the value of of folds to a multiple of the number of cores on each cluster node.
stepsize_mstop	Integer denoting the stepsize used during cross-validation for tuning the value of m_{stop}.
postprocessing	String to set the type of postprocessing. Defaults to "none" for a simple average of the ensemble components.

Details

Generally gradient boosting offers more flexibility and better predictive performance than random forests, but is usually not used for large data sets because of its iterative nature. parboost is designed to scale up component-wise functional gradient boosting in a distributed memory environment by splitting the observations into disjoint subsets, or alternatively by bootstrapping the original data. Each cluster node then fits a boosting model to its subset of the data. These boosting models are combined in an ensemble, either with equal weights, or by fitting a (penalized) regression model on the predictions of the individual models on the complete data. The motivation behind parboost is to offer a boosting framework that is as easy to parallelize and thus scalable as random forests.

If you want to modify the boosting parameters, please take a look at the mboost package documentation and pass the appropriate parameters to tree_control and boost_control.

Value

An object of type parboost with print, summary, predict, coef and selected methods.

Author(s)

Ronert Obst

References

Peter Buehlmann and Bin Yu (2003), Boosting with the L2 loss: regression and classification. *Journal of the American Statistical Association*, **98**, 324–339.

Peter Buehlmann and Torsten Hothorn (2007), Boosting algorithms: regularization, prediction and model fitting. *Statistical Science*, **22**(4), 477–505.

Torsten Hothorn, Peter Buehlmann, Thomas Kneib, Mattthias Schmid and Benjamin Hofner (2010), Model-based Boosting 2.0. *Journal of Machine Learning Research*, **11**, 2109–2113.

Yoav Freund and Robert E. Schapire (1996), Experiments with a new boosting algorithm. In *Machine Learning: Proc. Thirteenth International Conference*, 148–156.

Jerome H. Friedman (2001), Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, **29**, 1189–1232.

Benjamin Hofner, Andreas Mayr, Nikolay Robinzonov and Matthias Schmid (2012). Model-based Boosting in R: A Hands-on Tutorial Using the R Package mboost. *Department of Statistics, Technical Report No. 120*.
http://epub.ub.uni-muenchen.de/12754/

T. Hothorn, P. Buehlmann, T. Kneib, M. Schmid, and B. Hofner (2013). mboost: Model-Based Boosting, R package version 2.2-3, http://CRAN.R-project.org/package=mboost.

Examples

```
## Run parboost on a cluster (example not run)
# data(friedman2)
# library(parallel)
# cl <- makeCluster(2)
# parboost_model <- parboost(cluster_object = cl, data = friedman2,
#                            nsplits = 2, formula = y ~ .,
#                            baselearner="bbs", postprocessing = "glm",
#                            control = boost_control(mstop=10))
# stopCluster(cl)
# print(parboost_model)
# summary(parboost_model)
# head(predict(parboost_model))
#
# ## Run parboost serially for testing/debugging purposes
# parboost_model <- parboost(data = friedman2, nsplits = 2, formula
# = y ~ ., baselearner="bbs", postprocessing = "glm", control =
# boost_control(mstop=10))
```

`predict.parboost` *Generate predictions from parboost object*

Description

Predict method for `parboost` objects

Usage

```
## S3 method for class 'parboost'
predict(object, newdata = NULL, type = c("response",
  "link"), ...)
```

Arguments

object	Object of class parboost
newdata	Optionally a data frame with new data to predict
type	String determining the type of prediction. The default "response" is on the scale of the response variable and "link" is on the scale of the predictors.
...	Additional parameters passed to predict.mboost.

Details

If no new data is passed to predict, predict outputs the fitted values. If you pass a data frame with new values to predict, it will generate the predictions for them.

Value

Numeric vector of fitted values

Author(s)

Ronert Obst

References

T. Hothorn, P. Buehlmann, T. Kneib, M. Schmid, and B. Hofner (2013). mboost: Model-Based Boosting, R package version 2.2-3, http://CRAN.R-project.org/package=mboost.

---

print.parboost *Prints a short description of a parboost object.*

Description

Print a parboost object

Usage

```
## S3 method for class 'parboost'
print(x, ...)
```

Arguments

x	a parboost object.
...	Additional arguements passed to callies.

Details

Prints a basic description of a parboost object

Value

Prints a short descritpion of a parboost object.

Author(s)

Ronert Obst

`print.summary.parboost`

Prints a summary of a parboost object.

Description

Print a summary of a parboost object

Usage

```
## S3 method for class 'summary.parboost'
print(x, ...)
```

Arguments

`x`	a parboost object.
`...`	Additional arguements passed to callies.

Details

Prints a basic summary of a parboost object

Value

Prints a summary of a parboost object.

Author(s)

Ronert Obst

selected.parboost *Selected base learners*

Description

Display selected base learners

Usage

```
## S3 method for class 'parboost'
selected(object, ...)
```

Arguments

object	Object of class parboost
...	Parameters passed to selected.mboost

Details

Displays the selected base learners from all submodels.

Value

Numeric vector of selected base learners.

Author(s)

Ronert Obst

References

T. Hothorn, P. Buehlmann, T. Kneib, M. Schmid, and B. Hofner (2013). mboost: Model-Based Boosting, R package version 2.2-3, http://CRAN.R-project.org/package=mboost.

summary.parboost *Prints a summary of a parboost object.*

Description

Print a summary of a parboost object

Usage

```
## S3 method for class 'parboost'
summary(object, ...)
```

Arguments

`object`	a parboost object.
`...`	Additional arguements passed to callies.

Details

Prints a basic summary of a parboost object

Value

Prints a summary of a parboost object.

Author(s)

Ronert Obst

Index