Package ‘diversityForest’

January 29, 2020

Type Package
Title Diversity Forests
Version 0.2.0
Date 2020-01-29
Author Roman Hornung [aut, cre], Marvin N. Wright [ctb, cph]
Maintainer Roman Hornung <hornung@ibe.med.uni-muenchen.de>
Description Implements diversity forests as described in an upcoming paper by the author of the package.
This package is a fork of the R package ‘ranger’ (main author: Marvin N. Wright) that implements random forests using an efficient C++ implementation. More precisely, ‘diversityForest’ was written by modifying the code of ‘ranger’, version 0.11.0. Therefore, details on further functionalities of the code that are not presented in the help pages of ‘diversityForest’ are found in the help pages of ‘ranger’ (version 0.11.0). The code in the example sections of the ‘diversityForest’ manual can be used as a template for all basic application scenarios with respect to classification, regression and survival prediction using univariate, binary splitting. Some function arguments adopted from the ‘ranger’ package are not be useable with diversity forests (for the current package version).
SystemRequirements C++11
Encoding UTF-8
License GPL-3
Imports Rcpp (>= 0.11.2), Matrix
LinkingTo Rcpp, RcppEigen
Depends R (>= 3.1)
Suggests survival, testthat
RoxygenNote 6.1.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2020-01-29 19:00:02 UTC

1
Construct a Diversity Forest prediction rule

Description

Implements diversity forests as presented in Hornung (inprep). Currently, classification, regression and survival prediction are possible. Moreover, the current version of the package only supports univariate, binary splitting, but future versions will allow using specific other split procedures. Because 'diversityForest' is a fork of the 'ranger' R package (see below for details), the documentation is largely taken from 'ranger', where large parts of the documentation do not apply to (the current version of) the 'diversityForest' package. Moreover, divfor contains function arguments that are currently not supported, but will be in future versions of the package. However, the package is fully functional with respect to applying diversity forest using univariate, binary splitting. See the example section for all basic application scenarios.

Usage

divfor(formula = NULL, data = NULL, num.trees = 500, mtry = NULL, importance = "none", write.forest = TRUE, probability = FALSE, min.node.size = NULL, max.depth = NULL, replace = TRUE, sample.fraction = ifelse(replace, 1, 0.632), case.weights = NULL, class.weights = NULL, splitrule = NULL, num.random.splits = 1, alpha = 0.5, minprop = 0.1, split.select.weights = NULL, always.split.variables = NULL, respect.unordered.factors = NULL, scale.permutation.importance = FALSE, keep.inbag = FALSE, inbag = NULL, holdout = FALSE, quantreg = FALSE, oob.error = TRUE, num.threads = NULL, save.memory = FALSE, verbose = TRUE, seed = NULL, dependent.variable.name = NULL, status.variable.name = NULL, classification = NULL, nsplits = 30, proptry = 1)
Arguments

formula Object of class formula or character describing the model to fit. Interaction terms supported only for numerical variables.
data Training data of class data.frame, matrix, dgCMatrix (Matrix) or gwaa.data (GenABEL).
num.trees Number of trees. Default is 500.
mtry Artefact from 'ranger'. NOT needed for diversity forests.
importance Variable importance mode, one of 'none', 'impurity', 'impurity_corrected', 'permutation'. The 'impurity' measure is the Gini index for classification, the variance of the responses for regression and the sum of test statistics (see splitrule) for survival. NOTE: Currently, only "permutation" (and "none") work for diversity forests.
write.forest Save divfor.forest object, required for prediction. Set to FALSE to reduce memory usage if no prediction intended.
probability Grow a probability forest as in Malley et al. (2012).
min.node.size Minimal node size. Default 1 for classification.
max.depth Maximal tree depth. A value of NULL or 0 (the default) corresponds to unlimited depth, 1 to tree stumps (1 split per tree).
replace Sample with replacement.
sample.fraction Fraction of observations to sample. Default is 1 for sampling with replacement and 0.632 for sampling without replacement. For classification, this can be a vector of class-specific values.

Weights for sampling of training observations. Observations with larger weights will be selected with higher probability in the bootstrap (or subsampled) samples for the trees.

class.weights Weights for the outcome classes (in order of the factor levels) in the splitting rule (cost sensitive learning). Classification and probability prediction only. For classification the weights are also applied in the majority vote in terminal nodes.
splitrule Splitting rule. For classification and probability estimation "gini" or "extratrees" with default "gini". For regression "variance", "extratrees" or "maxstat" with default "variance". For survival "logrank", "extratrees", "C" or "maxstat" with default "logrank". NOTE: For diversity forests currently only the default splitting rules are supported.

Artefact from 'ranger'. NOT needed for diversity forests.
alpha For "maxstat" splitrule: Significance threshold to allow splitting. NOT needed for diversity forests.
minprop For "maxstat" splitrule: Lower quantile of covariate distribution to be considered for splitting. NOT needed for diversity forests.
split.select.weights Numeric vector with weights between 0 and 1, representing the probability to select variables for splitting. Alternatively, a list of size num.trees, containing split select weight vectors for each tree can be used.
always.split.variables
Currently not useable. Character vector with variable names to be always selected.

respect.unordered.factors
Handling of unordered factor covariates. One of ‘ignore’ and ‘order’ (the option ‘partition’ possible in ‘ranger’ is not (yet) possible with diversity forests). Default is ‘ignore’. Alternatively TRUE (= ‘order’) or FALSE (= ‘ignore’) can be used.

scale.permutation.importance
Scale permutation importance by standard error as in (Breiman 2001). Only applicable if permutation variable importance mode selected.

keep.inbag
Save how often observations are in-bag in each tree.
inbag
Manually set observations per tree. List of size num.trees, containing inbag counts for each observation. Can be used for stratified sampling.

holdout
Hold-out mode. Hold-out all samples with case weight 0 and use these for variable importance and prediction error.

quantreg
Prepare quantile prediction as in quantile regression forests (Meinshausen 2006). Regression only. Set keep.inbag = TRUE to prepare out-of-bag quantile prediction.

oob.error
Compute OOB prediction error. Set to FALSE to save computation time, e.g. for large survival forests.

num.threads
Number of threads. Default is number of CPUs available.

save.memory
Use memory saving (but slower) splitting mode. No effect for survival and GWAS data. Warning: This option slows down the tree growing, use only if you encounter memory problems. NOT needed for diversity forests.

verbose
Show computation status and estimated runtime.

seed
Random seed. Default is NULL, which generates the seed from R. Set to 0 to ignore the R seed.

dependent.variable.name
Name of dependent variable, needed if no formula given. For survival forests this is the time variable. NOTE: Currently, diversity forests are only possible using the formula interface; thus, dependent.variable.name must not be specified.

status.variable.name
Name of status variable, only applicable to survival data and needed if no formula given. Use 1 for event and 0 for censoring. NOTE: Currently, diversity forests are only possible using the formula interface; thus, status.variable.name must not be specified.

classification
Only needed if data is a matrix. Set to TRUE to grow a classification forest.
nstarts
Number of candidate splits to sample for each split. Default is 30.

proptry
parameter that restricts the number of candidate splits considered for small nodes. If nstarts is larger than proptry times the number of all possible splits, the number of candidate splits to draw is reduced to the largest integer smaller than proptry times the number of all possible splits. Default is 1, which corresponds to always using nstarts candidate splits.
**Details**

As noted above, 'diversityForest' is a fork of the R package 'ranger' that implements random forests using an efficient C++ implementation. More precisely, 'diversityForest' was written by modifying the code of 'ranger', version 0.11.0. Therefore, details on further functionalities of the code that are not presented in the help pages of 'diversityForest' are found in the help pages of 'ranger' (version 0.11.0). The code in the example sections of `divfor` and `tunedivfor` can be used as a template for all basic application scenarios with respect to classification, regression and survival prediction using univariate, binary splitting. Some function arguments adopted from the 'ranger' package are not useable with diversity forests (for the current package version).

**Value**

Object of class `divfor` with elements

- **forest**: Saved forest (If write.forest set to TRUE). Note that the variable IDs in the `split.varIDs` object do not necessarily represent the column number in R.
- **predictions**: Predicted classes/values, based on out-of-bag samples (classification and regression only).
- **variable.importance**: Variable importance for each independent variable.
- **prediction.error**: Overall out-of-bag prediction error. For classification this is the fraction of misclassified samples, for probability estimation the Brier score, for regression the mean squared error and for survival one minus Harrell's C-index.
- **r.squared**: R squared. Also called explained variance or coefficient of determination (regression only). Computed on out-of-bag data.
- **confusion.matrix**: Contingency table for classes and predictions based on out-of-bag samples (classification only).
- **unique.death.times**: Unique death times (survival only).
- **chf**: Estimated cumulative hazard function for each sample (survival only).
- **survival**: Estimated survival function for each sample (survival only).
- **call**: Function call.
- **num.trees**: Number of trees.
- **num.independent.variables**: Number of independent variables.
- **min.node.size**: Value of minimal node size used.
- **treetype**: Type of forest/tree. classification, regression or survival.
- **importance.mode**: Importance mode used.
- **num.samples**: Number of samples.
- **splitrule**: Splitting rule.
- **replace**: Sample with replacement.
- **nsplits**: Value of nsplits used.
- **proptry**: Value of proptry used.
Author(s)

Roman Hornung, Marvin N. Wright

References


See Also

predict.divfor

Examples

library("diversityForest")

## Set seed to obtain reproducible results:
set.seed(1234)

## Diversity forest with default settings (NOT recommended)
# Classification:
divfor(Species ~ ., data = iris, num.trees = 20)
# Regression:
iris2 <- iris; iris2$Species <- NULL; iris2$Y <- rnorm(nrow(iris2))
divfor(Y ~ ., data = iris2, num.trees = 20)
# Survival:
library("survival")
divfor(Surv(time, status) ~ ., data = veteran, num.trees = 20, respect.unordered.factors = "order")
# NOTE: num.trees = 20 is specified too small for practical purposes - the prediction performance of the resulting forest will be suboptimal!!
# In practice, num.trees = 500 (default value) or a larger number should be used.

## Diversity forest with specified values for nsplits and proptry (NOT recommended)
divfor(Species ~ ., data = iris, nsplits = 10, proptry = 0.4, num.trees = 20)
# NOTE again: num.trees = 20 is specified too small for practical purposes.

## Applying diversity forest after optimizing the values of nsplits and proptry (recommended)
tuneres <- tunedivfor(formula = Species ~ ., data = iris, num.trees.pre = 20)
# NOTE: num.trees.pre = 20 is specified too small for practical purposes - the out-of-bag error estimates of the forests constructed during optimization will be much too variable!!
# In practice, num.trees.pre = 500 (default value) or a
# larger number should be used.
divfor(Species ~ ., data = iris, nsplits = tuneres$nsplitsopt,
              proptry = tuneres$proptryopt, num.trees = 20)
# NOTE again: num.trees = 20 is specified too small for practical purposes.

## Prediction
train.idx <- sample(nrow(iris), 2/3 * nrow(iris))
iris.train <- iris[train.idx, ]
iris.test <- iris[-train.idx, ]
tuneres <- tunedivfor(formula = Species ~ ., data = iris.train, num.trees.pre = 20)
# NOTE again: num.trees.pre = 20 is specified too small for practical purposes.
rg.iris <- divfor(Species ~ ., data = iris.train, nsplits = tuneres$nsplitsopt,
              proptry = tuneres$proptryopt, num.trees = 20)
# NOTE again: num.trees = 20 is specified too small for practical purposes.
pred.iris <- predict(rg.iris, data = iris.test)
table(iris.test$Species, pred.iris$predictions)

## Variable importance
rg.iris <- divfor(Species ~ ., data = iris, importance = "permutation", num.trees = 20)
# NOTE again: num.trees = 20 is specified too small for practical purposes.
rg.iris$variable.importance

importance.divfor

_Diversity Forest variable importance_

**Description**

Extract variable importance of _divfor_ object.

**Usage**

```
## S3 method for class 'divfor'
importance(x, ...)
```

**Arguments**

- `x` _divfor_ object.
- `...` Further arguments passed to or from other methods.

**Value**

Variable importance measures.

**Author(s)**

Marvin N. Wright

**See Also**

- _divfor_
**predict.divfor**

*Diversity Forest prediction*

**Description**

Prediction with new data and a saved forest from *divfor*.

**Usage**

```r
def predict(object, data = NULL, predict.all = FALSE, num.trees = object$num.trees, type = "response", se.method = "infjack", quantiles = c(0.1, 0.5, 0.9), seed = NULL, num.threads = NULL, verbose = TRUE, ...)
```

**Arguments**

- `object`: `divfor` object.
- `data`: New test data of class `data.frame` or `gwaa.data` (GenABEL).
- `predict.all`: Return individual predictions for each tree instead of aggregated predictions for all trees. Return a matrix (sample x tree) for classification and regression, a 3d array for probability estimation (sample x class x tree) and survival (sample x time x tree).
- `num.trees`: Number of trees used for prediction. The first `num.trees` in the forest are used.
- `type`: Type of prediction. One of 'response', 'se', 'terminalNodes', 'quantiles' with default 'response'. See below for details.
- `se.method`: Method to compute standard errors. One of 'jack', 'infjack' with default 'infjack'. Only applicable if `type` = 'se'. See below for details.
- `quantiles`: Vector of quantiles for quantile prediction. Set type = 'quantiles' to use.
- `seed`: Random seed. Default is `NULL`, which generates the seed from R. Set to `0` to ignore the R seed. The seed is used in case of ties in classification mode.
- `num.threads`: Number of threads. Default is number of CPUs available.
- `verbose`: Verbose output on or off.
- `...`: Further arguments passed to or from other methods.

**Details**

This package is a fork of the R package 'ranger' that implements random forests using an efficient C++ implementation. More precisely, 'diversityForest' was written by modifying the code of 'ranger', version 0.11.0. Therefore, details on further functionalities of the code that are not presented in the help pages of 'diversityForest' are found in the help pages of 'ranger' (version 0.11.0). The code in the example sections of *divfor* and *tunedivfor* can be used as a template for all common application scenarios with respect to classification, regression and survival prediction using univariate, binary splitting. Some function arguments adopted from the 'ranger' package are not be useable with diversity forests (for the current package version).
Value

Object of class divfor.prediction with elements

- **predictions**: Predicted classes/values (only for classification and regression)
- **unique.death.times**: Unique death times (only for survival).
- **chf**: Estimated cumulative hazard function for each sample (only for survival).
- **survival**: Estimated survival function for each sample (only for survival).
- **num.trees**: Number of trees.
- **num.independent.variables**: Number of independent variables.
- **treetype**: Type of forest/tree. Classification, regression or survival.
- **num.samples**: Number of samples.

Author(s)

Marvin N. Wright

References


See Also

divfor

Description

Extract training data predictions of divfor object.

Usage

```r
## S3 method for class 'divfor'
predictions(x, ...)
```

Arguments

- **x**: divfor object.
- **...**: Further arguments passed to or from other methods.
Value

Predictions: Classes for Classification forests, Numerical values for Regressions forests and the estimated survival functions for all individuals for Survival forests.

Author(s)

Marvin N. Wright

See Also

divfor

diversity Forest predictions

Description

Extract predictions of divfor.prediction object.

Usage

## S3 method for class 'divfor.prediction'
predictions(x, 
...

Arguments

x divfor.prediction object.
...
Further arguments passed to or from other methods.

Value

Predictions: Classes for Classification forests, Numerical values for Regressions forests and the estimated survival functions for all individuals for Survival forests.

Author(s)

Marvin N. Wright

See Also

divfor
Description
Print contents of divfor object.

Usage
## S3 method for class 'divfor'
print(x, ...)

Arguments
x Object of class 'divfor'.
...
Further arguments passed to or from other methods.

Author(s)
Marvin N. Wright, Roman Hornung

See Also
divfor

Description
Print contents of divfor forest object.

Usage
## S3 method for class 'divfor.forest'
print(x, ...)

Arguments
x Object of class 'divfor.forest'.
...
further arguments passed to or from other methods.

Author(s)
Marvin N. Wright
print.divfor.prediction

Print divfor prediction

Description

Print contents of divfor prediction object.

Usage

## S3 method for class 'divfor.prediction'
print(x, ...)

Arguments

x 
Object of class 'divfor.prediction'.

... 
further arguments passed to or from other methods.

Author(s)

Marvin N. Wright

print.tunedivfor

Print tunedivfor

Description

Print contents of tunedivfor object.

Usage

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

Arguments

x 
Object of class 'tunedivfor'.

... 
further arguments passed to or from other methods.

Author(s)

Roman Hornung

See Also

tunedivfor
**Description**

First, both for nsplits and proptry a grid of possible values may be provided, where default grids are used if no grids are provided. Second, for each pairwise combination of values from these two grids a forest is constructed. Third, that pair of nsplits and proptry values is used as the optimized set of parameter values that is associated with the smallest out-of-bag prediction error. If several pairs of parameter values are associated with the same smallest out-of-bag prediction error, the pair with the smallest (parameter) values is used.

**Usage**

```r
tunedivfor(formula = NULL, data = NULL, nsplitsgrid = c(2, 5, 10, 30, 50, 100, 200), proptrygrid = c(0.05, 1), num.trees.pre = 500)
```

**Arguments**

- `formula`: Object of class formula or character describing the model to fit. Interaction terms supported only for numerical variables.
- `data`: Training data of class data.frame, matrix, dgCMatrix (Matrix) or gwaa.data (GenABEL).
- `nsplitsgrid`: Grid of values to consider for nsplits. Default grid: 2, 5, 10, 30, 50, 100, 200.
- `proptrygrid`: Grid of values to consider for proptry. Default grid: 0.05, 1.
- `num.trees.pre`: Number of trees used for each forest constructed during tuning parameter optimization. Default is 500.

**Value**

List with elements

- `nsplitsopt`: Optimized value of nsplits.
- `proptryopt`: Optimized value of proptry.
- `tunegrid`: Two-dimensional data.frame, where each row contains one pair of values considered for nsplits (first entry) and proptry (second entry).
- `ooberrs`: The out-of-bag prediction errors obtained for each pair of values considered for nsplits and proptry, where the ordering of pairs of values is the same as in tunegrid (see above).

**Author(s)**

Roman Hornung
References


See Also
divfor

Examples

library("diversityForest")

## Set seed to obtain reproducible results:
set.seed(1234)

## Tuning parameter optimization for the iris data set
tuneres <- tunedivfor(formula = Species ~ ., data = iris, num.trees.pre = 20)
# NOTE: num.trees.pre = 20 is specified too small for practical purposes - the out-of-bag error estimates of the forests constructed during optimization will be much too variable!!
# In practice, num.trees.pre = 500 (default value) or a larger number should be used.

tuneres
tuneres$nsplitsopt
tuneres$proptryopt
tuneres$tunegrid
tuneres$ooberrs
Index

divfor, 2, 5, 7–11, 14

importance (importance.divfor), 7
importance.divfor, 7

predict.divfor, 6, 8
predictions
  (predictions.divfor.prediction), 10
predictions.divfor, 9
predictions.divfor.prediction, 10
print.divfor, 11
print.divfor.forest, 11
print.divfor.prediction, 12
print.tunedivfor, 12

tunedivfor, 5, 8, 12, 13