

Package ‘splinetree’

July 18, 2019

Title Longitudinal Regression Trees and Forests

Version 0.2.0

Description Builds regression trees and random forests for longitudinal or functional data using a spline projection method. Implements and extends the work of Yu and Lambert (1999) <doi:10.1080/10618600.1999.10474847>. This method allows trees and forests to be built while considering either level and shape or only shape of response trajectories.

Depends R (>= 3.5.0), rpart, nlme, splines

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports mosaic, ggplot2, treeClust, mclust

RoxygenNote 6.1.1

Suggests R.rsp, knitr, rmarkdown, testthat

VignetteBuilder R.rsp

BugReports <https://github.com/anna-neufeld/splinetree/issues>

URL <https://github.com/anna-neufeld/splinetree>

NeedsCompilation no

Author Anna Neufeld [aut, cre],
Brianna Heggeseth [aut, ths]

Maintainer Anna Neufeld <aneufeld@uw.edu>

Repository CRAN

Date/Publication 2019-07-18 06:36:41 UTC

R topics documented:

avSize	2
getNodeData	2
nodePlot	3

plotImp	3
plotNode	4
predictCoeffs	5
predictCoeffsForest	6
predictY	7
predictYForest	7
projectedR2	8
projectedR2Forest	9
pruneForest	10
spaghettiPlot	10
splineForest	11
splineTree	13
splineTreePlot	14
stPlot	15
stPrint	15
terminalNodeSummary	16
treeSimilarity	17
treeSize	17
treeSummary	18
varImpCoeff	19
varImpY	20
yR2	21
yR2Forest	21

 avSize

Compute the average tree size in a forest

Description

Returns the average number of terminal nodes for trees in a forest

Usage

```
avSize(forest)
```

Arguments

forest A model returned by splineForest()

Value

The average number of terminal nodes in forest

Examples

```
avSize(forest)
```

getNodeData	<i>Retrieve the subset of the data found at a given terminal node</i>
-------------	---

Description

Given a terminal node number, this function returns the data belonging to this terminal node. If the `dataType` argument is 'all', returns all rows of data from the original dataset that fall in this node. Otherwise, the flattened data that belongs to this node is returned (one row of data per ID, original responses replaced by spline coefficients).

Usage

```
getNodeData(tree, node, dataType = "all")
```

Arguments

<code>tree</code>	a model returned from <code>splineTree()</code>
<code>node</code>	The number of the node to retrieve data from. Must be valid number of a terminal node. Node numbers can be seen using <code>stPrint(tree)</code> or <code>treeSummary(tree)</code> .
<code>dataType</code>	If "all", the data returned is from the original dataset (one row per individual observation with original response values). If "flat", the data returned is the flattened data (one row per person/unit), with individual spline coefficients instead of response values.

Value

A dataframe which holds all the data that falls into this node of the tree.

Examples

```
## Not run:
split_formula <- BMI ~ HISP + WHITE + BLACK + SEX +
  Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, 'ID', nlsySample, degree=1,
  df=3, intercept=TRUE, cp=0.006, minNodeSize=20)

## End(Not run)
node6data <- getNodeData(tree, 6, dataType = 'all')
plot(BMI~AGE, data=node6data)
```

nodePlot	<i>Plots the trajectories of each terminal node side by side.</i>
----------	---

Description

Corresponds to plotting only the second panel of `stPlot()`. If `model$intercept==FALSE`, estimated intercepts are added to each trajectory so that the trajectories are plotted at the level of reasonable response values.

Usage

```
nodePlot(model, colors = NULL)
```

Arguments

model	A model returned from <code>splineTree()</code>
colors	A list of colors to use. By default, uses colors drawn from a rainbow.

plotImp	<i>Create a barplot of relative variable importance scores.</i>
---------	---

Description

Given a named vector of variable importance measures, this function makes a barplot of the relative importances. The importances are scaled to sum to 1. An appropriate input is one column of the output from `varImpY()` or `varImpCoeff()`.

Usage

```
plotImp(importance_vector, ...)
```

Arguments

importance_vector	a named vector where the names are the variables and the vector stores the importances.
...	additional arguments to plot, such as "main", "cex", etc.

Examples

```
imp <- varImpCoeff(forest)[,3]
plotImp(imp, main="Standardized Variable Importance")
```

plotNode *Plot the predicted trajectory for a single node*

Description

Creates a simple plot of the predicted trajectory at a given node. Option to include the data that falls in the node on the same plot.

Usage

```
plotNode(tree, node, includeData = FALSE, estimateIntercept = TRUE)
```

Arguments

tree	A model returned from splineTree()
node	A node number. Must be a valid terminal node for the given spline tree. To view valid terminal node numbers, use stPrint() or treeSummary().
includeData	Would you like to see the data from the node plotted along with the predicted trajectory?
estimateIntercept	If the tree was built without an intercept, should the average starting response of all the individuals in the node be added to the trajectory to give the plot interpretable values? Or should the shape of the trajectory be plotted without any regard to the intercept?

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

plotNode(tree, 6, includeData=TRUE)
```

predictCoeffs *Predict spline coefficients for a testset using a spline tree*

Description

Returns a matrix of spline coefficients for each observation in the testset. If no testset is provided, returns predicted coefficients for the individuals in training set; in this case, the columns of the returned predictions correspond to the rows of the flattened training dataset (found in tree\$parms\$flat_data).

Usage

```
predictCoeffs(tree, testset = tree$parms$flat_data)
```

Arguments

tree	A model created with splineTree()
testset	The dataset to predict coefficients for. Default is the flattened dataset used to make the tree.

Details

```
importFrom treeClust rpart.predict.leaves
```

Value

A matrix of spline coefficients. The dimension of the matrix is the degrees of freedom of the spline by the number of units in the test set. The *i*th column of the matrix holds the predicted coefficients for the *i*th row in the testset.

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

preds <- predictCoeffs(tree)
```

```
predictCoeffsForest
```

Predict spline coefficients for a testset using a splineforest.

Description

Uses the forest to predict spline coefficients. Returns a matrix of predicted spline coefficients where the columns of the returned matrix correspond to rows of the testdata. The number of rows of the returned matrix is equal to the degrees of freedom of the forest. If no testdata is provided, forest\$flat_data is used. When testdata is not provided, predictions will be made according to one of three methods. The "method" parameter must be either "oob", "itb", or "all". This parameter specifies which trees are used in making a prediction for a certain datapoint. This parameter is not relevant when predicting for a testset that is distinct from the training set.

Usage

```
predictCoeffsForest(forest, method = "oob", testdata = NULL)
```

Arguments

forest	A model returned from splineForest()
method	A string; either "oob", "itb", or "all". If "oob" (the default), predictions for a given data point are made only using trees for which this data point was "out of the bag" (not in the random subsample). If "itb", predictions for a given data point are made using only the trees for which this datapoint was "in the bag" (in the random subsample). If "all", all trees are used for every datapoint.
testdata	The test data to make predictions for. If this is provided, then all trees are used for all datapoints.

Value

A matrix of predicted spline coefficients. The dimensions are forest\$df x nrow(testdata). Each column of the matrix corresponds to a row of the testdata.

Examples

```
trainingSetPreds <- predictCoeffsForest(forest)
newData <- data.frame("WHITE" = 0, "BLACK"=1, "HISP"=0, "Num_sibs"=3,
  "HGC_MOTHER"=12, "HGC_FATHER"=12, "SEX"=1)
predictCoeffsForest(forest, testdata = newData)
```

predictY

Predictions from a spline tree

Description

Returns a vector of predicted responses for the testData. If testData is omitted, returns predictions for the training data. This function is most meaningful if model\$intercept==TRUE.

Usage

```
predictY(model, testData = NULL)
```

Arguments

model	A model created with splineTree()
testData	The data to return predictions for. If omitted, uses the training data.

Value

A vector of predictions with rows corresponding to the testdata.

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

plot(predictY(tree), tree$parms$data[[tree$parms$yvar]])
```

predictYForest *Predict responses for a testset using a splineforest.*

Description

Uses the forest to make predictions of responses for individuals. This method should only be used on forests where `forest$intercept=TRUE`. If the `testdata` parameter is null, makes predictions for each row of the training data. In this case, the `methods` parameter (which should be set to "oob", "itb", or "all") determines the method used for prediction. If the `testdata` parameter is not null, the `methods` parameter is ignored and all trees are used for the prediction of every datapoint.

Usage

```
predictYForest(forest, method = "oob", testdata = NULL)
```

Arguments

<code>forest</code>	A model returned from <code>splineForest()</code>
<code>method</code>	A string. Must be either "oob", "itb", or "all". Only relevant when <code>testdata</code> is NULL. The default value is "oob". If "oob", predictions for a given data point are made only using trees for which this data point was "out of the bag" (not in the random subsample). If "itb", predictions for a given data point are made using only the trees for which this datapoint was in the bag (in the random subsample). If "all", all trees are used for every datapoint.
<code>testdata</code>	the Test data to make predictions for. If this is provided, then all trees are used for all datapoints.

Value

A vector of predicted responses. The indices of the vector correspond to rows of the `testdata`.

Examples

```
trainingSetPreds <- predictYForest(forest)
newData <- data.frame("AGE"=21, "WHITE" = 0, "BLACK"=1, "HISP"=0,
  "Num_sibs"=3, "HGC_MOTHER"=12, "HGC_FATHER"=12, "SEX"=1)
predictYForest(forest, testdata = newData)
```

projectedR2	<i>Computes percent of variation in projected response explained by a splinetree.</i>
-------------	---

Description

Computes an R^2 measure for a splinetree based on the projected sum of squared errors. Returns $1 - \text{SSE}/\text{SST}$. SSE is the sum of projection squared errors between individual smoothed trajectories and predicted smoothed trajectories evaluated on a fixed grid. SST is the sum of projection squared errors between individual smoothed trajectories and the overall population mean trajectory, evaluated on the same fixed grid. If `model$intercept==TRUE`, then there is the option to ignore the intercept coefficient when computing this metric. When the intercept is ignored, the metric captures how well the model explains variation in shape, and ignores any variation in intercept explained by the model.

Usage

```
projectedR2(model, includeIntercept = FALSE)
```

Arguments

<code>model</code>	a model created with <code>splineTree()</code>
<code>includeIntercept</code>	If <code>FALSE</code> and if the model was built with an intercept, the projected squared errors are computed while ignoring the intercept. If the model was built without an intercept, this parameter does not do anything.

Value

The percentage of variation in projected trajectory explained by the model. Computed as $1 - \text{SSE}/\text{SST}$. See description.

Examples

```
r2 <- projectedR2(tree)
```

projectedR2Forest	<i>Computes a level-based or shape-based evaluation metric for a spline-forest.</i>
-------------------	---

Description

Computes an R-squared-like evaluation metric for a spline forest. Goal is to see how well the predicted spline coefficients for each individual match the spline coefficients obtained when fitting a spline only to this individual's data (we call these coefficients the true coefficients). Computes $1 - \text{SSE}/\text{SST}$, where SSE is the total sum of squared projection errors of the true coefficients compared to the predicted coefficients, and SST is the total sum of squared projection errors of the true coefficients compared to the population mean coefficients. If this is an intercept forest, have the option to compute these sum of squares either with the intercept included or with the intercept ignored to isolate the shape.

Usage

```
projectedR2Forest(forest, method = "oob", removeIntercept = TRUE)
```

Arguments

<code>forest</code>	The output of a call to <code>splineForest()</code>
<code>method</code>	How would you like to compute this metric? The choices are "oob", "itb", or "all". "oob" means that predictions for a datapoint can only be made using trees for which that datapoint was "out of the bag" (not in the random subsample). "all" means that all trees are used in the prediction for every datapoint. "itb" means that predictions for a datapoint are made using only the trees for which this datapoint was IN the random subsample.
<code>removeIntercept</code>	If true, the projection sum of squared error is computed while ignoring the intercept coefficient. This will help capture the tree's performance at clustering based on shape, not based on level. This parameter is only meaningful if this forest was built using an intercept.

Value

Returns $1 - \text{SSE}/\text{SST}$, where SSE is the total sum of squared projection errors of the true coefficients compared to the predicted coefficients, and SST is the total sum of squared projection errors of the true coefficients compared to the population mean coefficients.

Examples

```
projectedR2Forest(forest, method="all", removeIntercept=TRUE)
projectedR2Forest(forest, method="all", removeIntercept=FALSE)
```

pruneForest	<i>Prune each tree in forest using a given complexity parameter.</i>
-------------	--

Description

Prunes each tree in the list forest\$Trees according to the provided complexity parameter. Returns a new forest.

Usage

```
pruneForest(forest, cp)
```

Arguments

forest	A model returned by splineForest()
cp	The complexity parameter that will be used to prune each tree (see rpart package documentation for detailed description of complexity parameter)

Value

A new spline forest model (named list) where each tree has been pruned to the desired level.

Examples

```
print(avSize(forest))
print(avSize(pruneForest(forest, cp=0.007)))
print(avSize(pruneForest(forest, cp=0.01)))
```

spaghettiPlot	<i>Create a faceted spaghetti plot of a splinetree model</i>
---------------	--

Description

Uses ggplot to create a paneled spaghetti plot of the data, where each panel corresponds to a terminal node in the tree. Allows users to visualize homogeneity of trajectories within the terminal nodes of the tree while also looking at the trajectories of different nodes side by side.

Usage

```
spaghettiPlot(model, colors = NULL)
```

Arguments

model	a model returned from splineTree()
colors	optional argument specifying colors to be used for each panel.

Examples

```
nlsySubset <- nlsySample[nlsySample$ID %in% sample(unique(nlsySample$ID), 400),]
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySubset, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

spaghettiPlot(tree)
```

splineForest *Build a spline random forest.*

Description

Builds an ensemble of regression trees for longitudinal or functional data using the spline projection method. The resulting model contains a list of spline trees along with some additional information. All parameters are used in the same way that they are used in the `splineTree()` function. The additional parameter `ntree` specifies how many trees should be in the ensemble, and `prob` controls the probability of selecting a given variable for split consideration at a node. This method may take several minutes to run- saving the forest after building it is recommended.

Usage

```
splineForest(splitFormula, tformula, idvar, data, knots = NULL,
  df = NULL, degree = 3, intercept = FALSE, nGrid = 7,
  gridPoints = NULL, ntree = 50, prob = 0.3, cp = 0.001,
  minNodeSize = 1, bootstrap = FALSE)
```

Arguments

<code>splitFormula</code>	Formula specifying the longitudinal response variable and the time-constant variables that will be used for splitting in the tree.
<code>tformula</code>	Formula specifying the longitudinal response variable and the variable that acts as the time variable.
<code>idvar</code>	The name of the variable that serves as the ID variable for grouping observations. Must be in quotes
<code>data</code>	dataframe that contains all variables specified in the formulas- in long format.
<code>knots</code>	Specified locations for internal knots in the spline basis. Defaults to <code>NULL</code> , which corresponds to no internal knots.
<code>df</code>	Degrees of freedom of the spline basis. If this is specified but the <code>knots</code> parameter is <code>NULL</code> , then the appropriate number of internal knots will be added at quantiles of the training data. If both <code>df</code> and <code>knots</code> are unspecified, the spline basis will have no internal knots.
<code>degree</code>	Specifies degree of spline basis used in the tree.

<code>intercept</code>	Specifies whether or not the splitting process will consider the intercept coefficient of the spline projections. Defaults to FALSE, which means that the tree will split based on trajectory shape, ignoring response level.
<code>nGrid</code>	Number of grid points to evaluate projection sum of squares at. If <code>gridPoints</code> is not supplied, then this is the number of grid points that will be automatically placed at quantiles of the time variable. The default is 7.
<code>gridPoints</code>	Optional. A vector of numbers that will be used as the grid on which to evaluate the projection sum of squares. Should fall roughly within the range of the time variable.
<code>ntree</code>	Number of trees in the forest.
<code>prob</code>	Probability of selecting a variable to included as a candidate for each split.
<code>cp</code>	Complexity parameter passed to the <code>rpart</code> building process. Default is the <code>rpart</code> default of 0.01
<code>minNodeSize</code>	Minimum number of observational units that can be in a terminal node. Controls tree size and helps avoid overfitting. Default is 10.
<code>bootstrap</code>	Boolean specifying whether bootstrap sampling should be used when choosing data to use for each tree. When set to FALSE (the default), sampling without replacement is used and 63.5 is used for each tree. When set to TRUE, a bootstrap sample is used for each tree.

Details

The ensemble method is highly similar to the random forest methodology of Breiman (2001). Each tree in the ensemble is fit to a random sample of 63.5 the subset of variables considered at each node is determined by a random process. The `prob` parameter specifies the probability that a given variable will be selected at a certain node. Because the method is based on probability, the same number of variables are not considered for splitting at each node (as in the `randomForest` package). Note that if `prob` is small and the number of variables in the `splitFormula` is also small, there is a high probability that no variables will be considered for splitting at a certain node, which is problematic. The fewer total variables there are, the larger `prob` should be to ensure good results.

Value

A spline forest model, which is a named list with 15 components. The list stores a list of trees (in `model$Trees`), along with information about the spline basis used (`model$intercept`, `model$innerKnots`, `model$boundaryKnots`, etc.), and information about which datapoints were used to build each tree (`model$soob_indices` and `model$index`). Note that each element in `model$Trees` is an `rpart` object but it is not the same as a model returned from `splineTree()` because it does not store all relevant information in `model$parms`.

Examples

```
nlsySubset <- nlsySample[nlsySample$ID %in% sample(unique(nlsySample$ID), 400),]
splitForm <- ~HISP+WHITE+BLACK+HGC_MOTHER+HGC_FATHER+SEX+Num_sibs
sampleForest <- splineForest(splitForm, BMI~AGE, 'ID', nlsySubset, degree=1, cp=0.005, ntree
```

splineTree

Build a splinetree model.

Description

Builds a regression tree for longitudinal or functional data using the spline projection method. The underlying tree building process uses the `rpart` package, and the resulting spline tree is an `rpart` object with additional stored information. The parameters `df`, `knots`, `degree`, `intercept` allow for flexibility in customizing the spline basis used for projection. The parameters `nGrid` and `gridPoints` allow for flexibility in the grid on which the projection sum of squares is evaluated. The parameters `minNodeSize` and `cp` allow for flexibility in controlling the size of the final tree.

Usage

```
splineTree(splitFormula, tformula, idvar, data, knots = NULL,
           df = NULL, degree = 3, intercept = FALSE, nGrid = 7,
           gridPoints = NULL, minNodeSize = 10, cp = 0.01)
```

Arguments

<code>splitFormula</code>	Formula specifying the longitudinal response variable and the time-constant variables that will be used for splitting in the tree.
<code>tformula</code>	Formula specifying the longitudinal response variable and the variable that acts as the time variable.
<code>idvar</code>	The name of the variable that serves as the ID variable for grouping observations. Must be a string.
<code>data</code>	dataframe in long format that contains all variables specified in the formulas.
<code>knots</code>	Specified locations for internal knots in the spline basis. Defaults to <code>NULL</code> , which corresponds to no internal knots.
<code>df</code>	Degrees of freedom of the spline basis. If this is specified but the <code>knots</code> parameter is <code>NULL</code> , then the appropriate number of internal knots will be added at quantiles of the training data. If both <code>df</code> and <code>knots</code> are unspecified, the spline basis will have no internal knots. If <code>knots</code> is specified, this parameter will be ignored.
<code>degree</code>	Specifies degree of spline basis used for projection.
<code>intercept</code>	Specifies whether or not the set of basis functions will include the intercept function. Defaults to <code>FALSE</code> , which means that the tree will split based on trajectory shape, ignoring response level.
<code>nGrid</code>	Number of grid points to evaluate projection sum of squares at. If <code>gridPoints</code> is not supplied, this argument will be used and the appropriate number of grid points will be placed at equally spaced quantiles of the time variable. The default is 7.
<code>gridPoints</code>	Optional. A vector of numbers that will be used as the grid on which to evaluate the projection sum of squares. Should fall roughly within the range of the time variable.

<code>minNodeSize</code>	Minimum number of observational units that can be in a terminal node. Controls tree size and helps avoid overfitting. Defaults to 10.
<code>cp</code>	Complexity parameter passed to the <code>rpart</code> building process. Controls tree size. Defaults to the <code>rpart</code> default of 0.01.

Value

An `rpart` object with additional `splinetree`-specific information stored in `model$parms`. The important attributes of the `rpart` object include `model$frame`, `model$where`, and `model$cptable`. `model$frame` holds information about each node in the tree. The `i`th entry in `model$where` tells us which row of `model$frame` describes the node that the `i`th individual in the flattened dataset falls into. `model$parms$flat_data` holds the flattened dataset that was used to build the tree. `model$cptable` displays the complexity parameters that would be needed to prune the tree to various desired sizes. Apart from holding the flattened dataset, `model$parms` holds the boundary knots and the internal knots of the spline basis used to build the tree. These are sometimes important to recover later.

Examples

```
nlsySample_subset <- nlsySample[nlsySample$ID %in% sample(unique(nlsySample$ID), 500),]
splitForm <- ~HISP+WHITE+BLACK+HGC_MOTHER+HGC_FATHER+SEX+Num_sibs
tree1 <- splineTree(splitForm, BMI~AGE, 'ID', nlsySample_subset, degree=3, intercept=TRUE, c
stPrint(tree1)
stPlot(tree1)
```

`splineTreePlot` *Creates a tree plot of a spline tree.*

Description

Creates a tree plot of a spline tree. This corresponds to plotting only the first panel of `stPlot()`. Code for this function was borrowed from the `longRPart` package on github.

Usage

```
splineTreePlot(model, colors = NULL)
```

Arguments

<code>model</code>	a model returned from <code>splineTree()</code>
<code>colors</code>	a list of colors that will be used for the terminal nodes (if <code>NULL</code> , will use a rainbow)

 stPlot

Plots a splinetree.

Description

Creates a two paneled plot of a splinetree that shows both the tree and the trajectories side by side. Note that this function has trouble when the plot window is not wide enough. If nothing shows up in RStudio, try increasing the size of the plot window and trying again. For a tree without an intercept, intercepts are estimated after-the-fact for each node using the average starting value in the data so that the plotted trajectories have reasonable response values.

Usage

```
stPlot(model, colors = NULL)
```

Arguments

model	A model returned from splineTree()
colors	A list of colors that will be used for the trajectories (if NULL, will automatically select colors from rainbow color scheme).

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

stPlot(tree, colors = c("red", "orange", "green", "blue", "cyan", "magenta"))
```

 stPrint

Print a spline tree in the style of print.rpart

Description

The printout provides numbered labels for the terminal nodes, a description of the split at each node, the number of observations found at each node, and the predicted spline coefficients for each node. This code is primarily taken from rpart base code for print.rpart. It has been modified to ensure that the full vector of coefficients is printed for each node.

Usage

```
stPrint(t, cp, digits = getOption("digits"))
```


Arguments

t	A model returned by splineTree()
cp	Optional- if provided, a pruned version of the tree will be printed. The tree will be pruned using the provided cp as the complexity parameter.
digits	Specifies how many digits of each coefficient should be printed

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

stPrint(tree)
```

```
terminalNodeSummary
```

Prints a summary of a terminal node in a tree

Description

If no argument is provided for the parameter `node`, summaries are printed for every terminal node. Otherwise, the summary of just the requested node is printed.

Usage

```
terminalNodeSummary(tree, node = NULL)
```

Arguments

tree	A model returned by splineTree().
node	The number of the node that you want summarized. To see which nodes correspond to which numbers, see <code>stPrint(tree)</code> or <code>treeSummary(tree)</code> . If this parameter is provided, must correspond to a valid terminal node in the tree.

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

terminalNodeSummary(tree)
```

`treeSimilarity` *Returns a measure of how similar the two trees are.*

Description

Computes the Adjusted Rand Index of the clusterings of the population created by the two trees. In the case of correlated covariates, two trees that split on entirely different variables may actually describe similar partitions of the population. This metric allows us to detect when two trees are partitioning the population similarly. A value close to 1 indicates a similar clustering.

Usage

```
treeSimilarity(tree1, tree2)
```

Arguments

`tree1` a model returned from `splineTree()`
`tree2` a model returned from `splineTree()`

Value

The Adjusted Rand Index of the clusterings created by the two trees.

See Also

`mclust::adjustedRandIndex`

Examples

```
splitForm <- ~SEX+Num_sibs+HGC_MOTHER+HGC_FATHER
nlsySubset <- nlsySample[nlsySample$ID %in% sample(unique(nlsySample$ID), 400),]
tree1 <- splineTree(splitForm, BMI~AGE, "ID", nlsySubset, degree=1, df=2, intercept=FALSE, c
tree2 <- splineTree(splitForm, BMI~AGE, "ID", nlsySubset, degree=1, df=3, intercept=TRUE, cp
treeSimilarity(tree1, tree2)
```

`treeSize` *Returns number of terminal nodes in a tree.*

Description

Returns number of terminal nodes in a tree.

Usage

```
treeSize(model)
```

Arguments

model A model returned by splineTree(). Also works on any rpart object

Value

The number of terminal nodes in the tree

Examples

```
## Not run:
split_formula <- ~ HISP + WHITE + BLACK + SEX + HGC_FATHER + HGC_MOTHER + Num_sibs
tree <- splineTree(split_formula, BMI~AGE, 'ID', nlsySample, degree=1,
  df=3, intercept=TRUE, cp=0.006, minNodeSize=20)

## End(Not run)
treeSize(tree)
```

treeSummary *Returns the tree frame.*

Description

Provides a similar output to model\$frame, but with the redundant information of yval and yval2 removed. Also omits the deviance, the complexity, and the weight. Useful for viewing node numbers and for extracting coefficients for a given node.

Usage

```
treeSummary(model)
```

Arguments

model A model built with splineTree()

Value

A dataframe. The number of rows is the same as the number of nodes in the tree. The row names display the node labels of each node. The "var" attribute either displays the split variable selected at each node, or <leaf> if this node is a terminal node. The "n" attribute displays the number of individuals in the node. The "dev" attribute reports the projected sum of squares at this node; terminal nodes have the smallest values for "dev" because this is what the tree building process is supposed to minimize. The "coeffs" attribute displays the coefficients predicted for each node.

Examples

```
nlsySubset <- nlsySample[nlsySample$ID %in% sample(unique(nlsySample$ID), 400),]
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySubset, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

treeSummary(tree)
```

varImpCoeff

Random Forest Variable Importance based on spline coefficients

Description

Returns the random forest variable importance based on the permutation accuracy measure, which is calculated as the difference in mean squared error between the original data and from randomly permuting the values of a variable.

Usage

```
varImpCoeff(forest, removeIntercept = TRUE, method = "oob")
```

Arguments

forest	a random forest, generated from splineForest()
removeIntercept	a boolean value, TRUE if you want to exclude the intercept in the calculations, FALSE otherwise.
method	the method to be used. This must be one of "oob" (out of bag), "all", "itb" (in the bag).

Value

a matrix of variable importance metrics.

Examples

```
importanceMatrix <- varImpCoeff(forest, removeIntercept=TRUE)
```

`varImpY`*Random Forest Variable Importance based on Y*

Description

Returns the random forest variable importance based on the permutation accuracy measure, which is calculated as the difference in mean squared error between the original data and from randomly permuting the values of a variable.

Usage

```
varImpY(forest, method = "oob")
```

Arguments

<code>forest</code>	a random forest, generated from <code>splineForest()</code>
<code>method</code>	the method to be used. This must be one of "oob" (out of bag), "all", "itb" (in the bag).

Details

The "method" parameter deals with the way in which forest performance should be measured. Since variable importance is based on a change in performance, the "method" parameter is necessary for a variable importance measure. The choices are "oob" (out of bag), "all", or "itb" (in the bag).

Value

A matrix storing variable importance metrics. The rows correspond to split variables. The columns are different methods of measuring importance. The first column is the absolute importance (mean difference in performance between permuted and unpermuted datasets). The second column measures the mean percent difference in performance. The third column standardizes the differences by dividing them by their standard deviation.

Examples

```
importanceMatrix <- varImpY(forest, method="oob")  
plotImp(importanceMatrix[,3])
```

yR2	<i>Computes percent of variation in response explained by spline tree.</i>
-----	--

Description

Computes the percentage of variation in response explained by the spline tree. This metric is only meaningful if `model$intercept==TRUE`. If the tree includes an intercept, the measure will be between 0 and 1.

Usage

```
yR2(model)
```

Arguments

`model` a model created with `splineTree()`

Value

An R^2 goodness measure. $1 - \text{SSE}/\text{SST}$ where SSE is the sum of squared errors between predicted responses and true responses, and SST is sum of squared errors of true responses around population mean. Note that if the tree passed in was built without an intercept, this function will return NULL.

Examples

```
split_formula <- ~HISP + WHITE + BLACK + SEX + Num_sibs + HGC_FATHER + HGC_MOTHER
tree <- splineTree(split_formula, BMI~AGE, idvar = "ID",
  data = nlsySample, degree = 1, df = 3,
  intercept = TRUE, cp = 0.005)

yR2(tree)
```

yR2Forest	<i>Computes a level-based evaluation metric for a splineforest that was built WITH an intercept.</i>
-----------	--

Description

Computes the R-squared metric for a spline forest. Goal is to see how well the predicted response values match the actual response values. Note that this function should only be used on forests where the intercept parameter is TRUE. A simple $1 - \text{SSE}/\text{SST}$ calculation.

Usage

```
yR2Forest(forest, method = "oob")
```

Arguments

forest	The output from a call to splineForest()
method	How would you like to compute this metric? The choices are "oob", "itb", or "all". "oob" means that predictions for a datapoint can only be made using trees for which that datapoint was "out of the bag" (not in the random subsample). "all" means that all trees are used in the prediction for every datapoint. "itb" means that predictions for a datapoint are made using only the trees for which this datapoint was IN the random subsample.

Value

Returns $1 - \text{SSE} / \text{SST}$, where SSE is the total sum of squared errors of the true responses and predicted responses, and SST is the total sum of squared errors of the responses around their mean. If this forest was not built with an intercept, returns NULL.

Examples

```
yR2Forest(forest, method="all")
```