Package ‘inTrees’

April 23, 2024

Title Interpret Tree Ensembles
Version 1.4
Date 2024-04-22
Imports RRF, arules, gbm, xtable, xgboost, data.table, methods
Description For tree ensembles such as random forests, regularized random forests and gradient boosted trees, this package provides functions for: extracting, measuring and pruning rules; selecting a compact rule set; summarizing rules into a learner; calculating frequent variable interactions; formatting rules in latex code. Reference: Interpreting tree ensembles with inTrees (Houtao Deng, 2019, <doi:10.1007/s41060-018-0144-8>).

Maintainer Houtao Deng <softwaredeng@gmail.com>

BugReports https://github.com/softwaredeng/inTrees/issues

License GPL (>= 3)

NeedsCompilation no

Repository CRAN

Date/Publication 2024-04-23 06:10:03 UTC

Author Houtao Deng [aut, cre],
Xin Guan [aut],
Vadim Khotilovich [aut]

R topics documented:

applyLearner .......................................................... 2
buildLearner .......................................................... 2
dataSimulate .......................................................... 3
dicretizeVector ......................................................... 4
extractRules .......................................................... 4
GBM2List .............................................................. 5
getFreqPattern ......................................................... 6
getRuleMetric ......................................................... 7
presentRules .......................................................... 8
pruneRule ............................................................ 8
RF2List .............................................................. 9
applyLearner

Description
apply STEL to data and get predictions

Usage
applyLearner(learner, X)

Arguments
learner a matrix with rules ordered by priority
X predictor variable matrix

Value
predictions for the data

See Also
buildLearner

buildLearner

Description
Build a simplified tree ensemble learner (STEL). Currently works only for classification problems.

Usage
buildLearner(ruleMetric, X, target, minFreq = 0.01)

Arguments
ruleMetric a matrix including the conditions, predictions, and and metrics
X predictor variable matrix
target target variable
minFreq minimum frequency of a rule condition in order to be included in STEL.
**Value**

a matrix including the conditions, prediction, and metrics, ordered by priority.

**Author(s)**

Houtao Deng

**References**


**Examples**

data(iris)
library(RRF)
X <- iris[,1:(ncol(iris)-1)]
target <- iris[,"Species"]
rf <- RRF(X,as.factor(target),ntree=100) # build an ordinary RF
treeList <- RF2List(rf)
ruleExec <- extractRules(treeList,X)
ruleExec <- unique(ruleExec)
ruleMetric <- getRuleMetric(ruleExec,X)
ruleMetric <- pruneRule(ruleMetric,X,target) # prune each rule
#ruleMetric <- selectRuleRRF(ruleMetric,X,target) # rule selection
learner <- buildLearner(ruleMetric,X,target)
pred <- applyLearner(learner,X)
read <- presentRules(learner,colnames(X)) # more readable format

# format the rule and metrics as a table in latex code
library(xtable)
print(xtable(read), include.rownames=FALSE)
print(xtable(ruleMetric[1:2,]), include.rownames=FALSE)

---

dataSimulate  

**Simulate data**

**Description**

Simulate data

**Usage**

dataSimulate(flag = 1, nCol = 20, nRow = 1000)

**Arguments**

flag  

nCol  
the number of columns in the data set. must >= 2.

nRow  
the number of rows in the data set.
**Value**

predictor variable matrix and target variable

**Examples**

```r
res <- dataSimulate(flag=1)
X <- res$X;
target <- res$target
```

---

**discretizeVector**

*discretize a variable*

**Description**

discretize a variable

**Usage**

```r
discretizeVector(v, K = 3)
```

**Arguments**

- `v`: vector
- `K`: discretize into up to K levels with equal frequency

**Value**

discretized levels for v

**Examples**

```r
data(iris)
discretizeVector(iris[,1],3)
```

---

**extractRules**

*Extract rules from a list of trees*

**Description**

Extract rule conditions from a list of trees. Use functions RF2List/GBM2List to transform RF/GBM objects to list of trees.

**Usage**

```r
extractRules(treeList, X, ntree = 100, maxdepth = 6, random = FALSE, digits = NULL)
```
GBM2List

Arguments

- `treeList` tree list
- `X` predictor variable matrix
- `ntree` conditions are extracted from the first ntree trees
- `maxdepth` conditions are extracted from the top maxdepth levels from each tree
- `random` the max depth for each tree is an integer randomly chosen between 1 and maxdepth
- `digits` digits for rounding

Value

a set of rule conditions

Examples

```r
library(RRF)
data(iris)
X <- iris[,1:(ncol(iris)-1)]
target <- iris[,"Species"]
rf <- RRF(X,as.factor(target),ntree=100) # build an ordinary RF
treeList <- RF2List(rf)
ruleExec <- extractRules(treeList,X,digits=4) # transform to R-executable rules
ruleExec <- unique(ruleExec)
```

Description

Transform gbm object to a list of trees that can be used for rule condition extraction

Usage

`GBM2List(gbm1,X)`

Arguments

- `gbm1` gbm object
- `X` predictor variable matrix

Value

a list of trees in an inTrees-required format

See Also

`RF2List`
**getFreqPattern**

*calculate frequent variable interactions*

**Description**

calculate frequent variable interactions

**Usage**

getFreqPattern(ruleMetric, minsup = 0.01, minconf = 0.5, minlen = 1, maxlen = 4)

**Arguments**

- **ruleMetric**: a matrix including conditions, predictions, and the metrics
- **minsup**: minimum support of conditions in a tree ensemble
- **minconf**: minimum confidence of the rules
- **minlen**: minimum length of the conditions
- **maxlen**: max length of the conditions

**Value**

a matrix including frequent variable interactions (in a form of conditions), predictions, length, support, and confidence.
getRuleMetric

Examples

```r
library(RRF)
library(arules)
data(iris)
X <- iris[,1:(ncol(iris)-1)]
target <- iris[,"Species"]
rf <- RRF(X,as.factor(target),ntree=100) # build an ordinary RF
treeList <- RF2List(rf)
ruleExec <- extractRules(treeList,X) # transform to R-executable rules
ruleMetric <- getRuleMetric(ruleExec,X,target)
freqPattern <- getFreqPattern(ruleMetric)
freqPatternMetric <- getRuleMetric(freqPattern,X,target)
```

getRuleMetric

Assign outcomes to a conditions, and measure the rules

Description

Assign outcomes to a conditions, and measure the rules

Usage

```r
getRuleMetric(ruleExec, X, target)
```

Arguments

- `ruleExec` a set of rule conditions
- `X` predictor variable matrix
- `target` target variable

Value

a matrix including the conditions, predictions, and metrics

References


Examples

```r
library(RRF)
data(iris)
X <- iris[,1:(ncol(iris)-1)]
target <- iris[,"Species"]
rf <- RRF(X,as.factor(target),ntree=100) # build an ordinary RF
treeList <- RF2List(rf)
ruleExec <- extractRules(treeList,X) # transform to R-executable rules
ruleExec <- unique(ruleExec)
ruleMetric <- getRuleMetric(ruleExec,X,target) # measure rules
```
pruneRule

### Description

Prune irrelevant variable-value pair from a rule condition

### Usage

```r
pruneRule(rules, X, target, maxDecay = 0.05, typeDecay = 2)
```

### Arguments

- **rules**: A matrix including the rules and metrics
- **X**: predictor variable matrix
- **target**: target variable vector
- **maxDecay**: threshold of decay
- **typeDecay**: 1: relative error; 2: error; default: 2

---

presentRules

**Present a learner using column names instead of X[i,]**

### Description

Present a learner using column names instead of X[i,]

### Usage

```r
presentRules(rules, colN, digits)
```

### Arguments

- **rules**: a set of rules
- **colN**: a vector including the column names
- **digits**: digits for rounding

### Value

A matrix including the conditions (with column names), etc.

### See Also

- buildLearner

### Examples

```r
# See function "buildLearner"
```
Value
A matrix including the rules each being pruned, and metrics

Author(s)
Houtao Deng

References

See Also
buildLearner

Examples

# see function “buildLearner”

---

**RF2List**  
*Transform a random forest object to a list of trees*

Description
Transform a random forest object to a list of trees

Usage
RF2List(rf)

Arguments
rf  
random forest object

Value
a list of trees

See Also
GBM2List
Examples

```r
library(RRF)
data(iris)
X <- iris[,1:(ncol(iris)-1)]
target <- iris[,"Species"]
rf <- RRF(X,as.factor(target),ntree=100) # build an ordinary RF
treeList <- RF2List(rf)
ruleExec <- extractRules(treeList,X) # transform to R-executable rules
```

---

selectRuleRRF

select a set of relevant and non-redundant rules

Description

select a set of relevant and non-redundant rules using regularized random forests

Usage

```r
selectRuleRRF(ruleMetric, X, target)
```

Arguments

- `ruleMetric`: a matrix including the rules and metrics
- `X`: predictor variable matrix
- `target`: response variable

Value

a matrix including a set of relevant and non-redundant rules, and their metrics

Author(s)

Houtao Deng

See Also

buildLearner

Examples

```r
# See function "buildLearner:
```
**XGB2List**

Transform an xgboost object to a list of trees

**Description**

Transform an xgboost object to a list of trees

**Usage**

`XGB2List(xgb, X)`

**Arguments**

- **xgb**: xgboost object
- **X**: predictor variable matrix

**Value**

a list of trees in an inTrees-required format

**See Also**

`XGB2List`

**Examples**

```r
library(data.table)
library(xgboost)
# test data set 1: iris
X <- within(iris, rm("Species")); Y <- iris[, "Species"]
X <- within(iris, rm("Species")); Y <- iris[, "Species"]
model_mat <- model.matrix(~ . - 1, data=X)
xgb <- xgboost(model_mat, label = as.numeric(Y) - 1, nrounds = 20,
objective = "multi:softprob", num_class = 3)
tree_list <- XGB2List(xgb, model_mat)
```
Index

* STEL
  * buildLearner, 2
* apply
  * applyLearner, 2
* discretize
  * discretizeVector, 4
* extract
  * extractRules, 4
* gbm
  * GBM2List, 5
* learner
  * buildLearner, 2
* measure
  * getRuleMetric, 7
* predict
  * applyLearner, 2
* present
  * presentRules, 8
* prune
  * pruneRule, 8
* randomforest
  * RF2List, 9
* rank
  * getRuleMetric, 7
* select
  * selectRuleRRF, 10
* simulate
  * dataSimulate, 3
* variable interaction
  * getFreqPattern, 6
* xgboost
  * XGB2List, 11

applyLearner, 2
buildLearner, 2, 8–10
dataSimulate, 3
discretizeVector, 4
eextractRules, 4

GBM2List, 5, 9
ggetFreqPattern, 6
ggetRuleMetric, 7

presentRules, 8
pruneRule, 8
RF2List, 5, 9
selectRuleRRF, 10
XGB2List, 11, 11