Package ‘rCBA’

October 14, 2022

Title CBA Classifier
Version 0.4.3
Author Jaroslav Kuchar [aut, cre],
Tomas Kliegr [ctb]
Maintainer Jaroslav Kuchar <jaroslav.kuchar@gmail.com>
URL https://github.com/jaroslav-kuchar/rCBA
BugReports https://github.com/jaroslav-kuchar/rCBA/issues
Description Provides implementations of a classifier based on the
``Classification Based on Associations'' (CBA). It can be used for building
classification models from association rules. Rules are pruned in the order of
precedence given by the sort criteria and a default rule is added. The final
classifier labels provided instances. CBA was originally proposed by Liu,
B. Hsu, W. and Ma, Y. Integrating Classification and Association Rule
070-7).
Depends R (>= 3.1.3), rJava, arules
Imports R.utils, TunePareto, methods, stats, utils
License Apache License (== 2.0)
LazyData true
SystemRequirements Java (>= 8)
RoxygenNote 6.1.1
Encoding UTF-8
Collate 'init.R' 'build.R' 'buildFPGrowth.R' 'classification.R'
 'fpgrowth.R' 'pruning.R' 'utils.R'
NeedsCompilation no
Repository CRAN
Date/Publication 2019-05-29 21:50:03 UTC
R topics documented:

build ......................................................... 2
buildFPGrowth ........................................... 3
classification ............................................. 3
fpgrowth ................................................... 4
frameToRules .............................................. 5
pruning ..................................................... 6

Index

build 7

Build classifier function (Apriori-based)

Description

Automatic build of the classification model using the Apriori algorithm from the arules

Usage

build(trainData, className = NA, pruning = TRUE, sa = list(),
      verbose = TRUE, parallel = TRUE)

Arguments

trainData data.frame or transactions from arules with input data
className column name with the target class - default is the last column
pruning performing pruning while building the model
sa simulated annealing setting. Default values: list(temp=100.0, alpha=0.05, tabu-
      RuleLength=5, timeout=10)
verbose verbose indicator
parallel parallel indicator

Value

list with parameters and model as data.frame with rules

Examples

library("rCBA")
data("iris")

output <- rCBA::build(iris, sa = list(alpha=0.5), parallel=FALSE) # speeding up the cooling
model <- output$model
predictions <- rCBA::classification(iris, model)
table(predictions)
sum(as.character(iris$Species)==as.character(predictions), na.rm=TRUE) / length(predictions)
buildFPGrowth

Build classifier function (FP-Growth-based)

Description
Automatic build of the classification model using the FP-Growth algorithm

Usage
```
buildFPGrowth(train, className = NULL, verbose = TRUE,
              parallel = TRUE)
```

Arguments
- `train`: data.frame or transactions from arules with input data
- `className`: column name with the target class - default is the last column
- `verbose`: verbose indicator
- `parallel`: parallel indicator

Value
- list with parameters and model as data.frame with rules

Examples
```
library("rCBA")
data("iris")

output <- rCBA::buildFPGrowth(iris[sample(nrow(iris), 10),], "Species",
                              parallel=FALSE, verbose=TRUE)
inspect(output$model)
```

classification

A classification function

Description
A classification function

Usage
```
classification(test, rules, verbose = TRUE)
```
Arguments

- `test`: data.frame or transactions from arules with input data
- `rules`: data.frame with rules
- `verbose`: verbose indicator

Value

vector with classifications

Examples

```r
library("arules")
library("rCBA")
data("iris")

train <- sapply(iris, as.factor)
train <- data.frame(train, check.names=FALSE)
trans <- as(train, "transactions")

rules = apriori(trans, parameter=list(support=0.03, confidence=0.03, minlen=2),
               appearance = list(rhs=c("Species=setosa", "Species=versicolor", "Species=virginica"),default="lhs"))

predictions <- rCBA::classification(train, rules)
table(predictions)
sum(as.character(train$Species)==as.character(predictions),na.rm=TRUE)/length(predictions)
```

Description

FP-Growth algorithm - Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. SIGMOD Rec. 29, 2 (2000) <doi:10.1145/335191.335372>

Usage

```r
fpgrowth(train, support = 0.01, confidence = 1, maxLength = 5,
          consequent = NULL, verbose = TRUE, parallel = TRUE)
```

Arguments

- `train`: data.frame or transactions from arules with input data
- `support`: minimum support
- `confidence`: minimum confidence
- `maxLength`: maximum length
- `consequent`: filter consequent - column name with consequent/target class
- `verbose`: verbose indicator
- `parallel`: parallel indicator
Examples

```r
library("rCBA")
data("iris")

train <- sapply(iris, as.factor)
train <- data.frame(train, check.names=FALSE)

txns <- as(train, "transactions")

rules = rCBA::fpgrowth(txns, support=0.03, confidence=0.03, maxLength=2, consequent="Species", parallel=FALSE)

predictions <- rCBA::classification(train, rules)
table(predictions)
sum(as.character(train$Species)==as.character(predictions), na.rm=TRUE)/length(predictions)

prunedRules <- rCBA::pruning(train, rules, method="m2cba", parallel=FALSE)
predictions <- rCBA::classification(train, prunedRules)
table(predictions)
sum(as.character(train$Species)==as.character(predictions), na.rm=TRUE)/length(predictions)
```

---

frameToRules  

**Conversion of data.frame to rules from arules**

Description

Conversion of data.frame to rules from arules

Usage

`frameToRules(model)`

Arguments

- `model`: data.frame with rules

Value

arules rules representation

Examples

```r
library("rCBA")

model <- data.frame("rules" = c("{X=1} => {Y=1}","{X=0} => {Y=0}"),
                     "support" = c(0.5,0.5),
                     "confidence" = c(0.5,0.5),
                     "lift" = c(1.0,1.0))

rules <- rCBA::frameToRules(model)
```
pruning

inspect(rules)

---

**Description**

A Pruning function

**Usage**

pruning(train, rules, method = "m2cba", verbose = TRUE, parallel = TRUE)

**Arguments**

- **train**: trainData data.frame or transactions from arules with input data
- **rules**: data.frame with rules
- **method**: pruning method m2cba(default)|m1cba|dcbrcba
- **verbose**: verbose indicator
- **parallel**: parallel indicator

**Value**

data.frame with pruned rules

**Examples**

```r
library("arules")
library("rCBA")
data("iris")

train <- sapply(iris,as.factor)
train <- data.frame(train, check.names=FALSE)
 txns <- as(train,"transactions")

rules = apriori(txns, parameter=list(support=0.03, confidence=0.03, minlen=2), appearance = list(rhs=c("Species=setosa","Species=versicolor","Species=virginica"),default="lhs"))

print(length(rules))
prunedRules <- rCBA::pruning(train, rules, method="m2cba", parallel=FALSE)
print(length(prunedRules))
```
Index

build, 2
buildFPGrowth, 3

classification, 3

fpgrowth, 4
frameToRules, 5

pruning, 6