

Package ‘arulesNBMiner’

April 17, 2009

Version 0.1-0

Date 2008-03-26

Title Mining NB-Frequent Itemsets and NB-Precise Rules

Author Michael Hahsler

Maintainer Michael Hahsler <michael@hahsler.net>

Description NBMiner is an implementation of the model-based mining algorithm for mining NB-frequent itemsets presented in “Michael Hahsler. A model-based frequency constraint for mining associations from transaction data. *Data Mining and Knowledge Discovery*, 13(2):137-166, September 2006.” In addition an extension for NB-precise rules is implemented.

Depends arules (>= 0.6-6), rJava (>= 0.6-0)

Imports methods

License GPL-2

Copyright All code is Copyright (C) Michael Hahsler

Repository CRAN

Date/Publication 2009-03-27 10:44:18

R topics documented:

Agrawal	2
NBMiner	3
NBMinerParameters	4
Index	6

Description

This dataset is generated by the method described by Agrawal and Srikant (1994) using the reimplementation in **arules** which also retains the patterns used in the generation process.

Usage

```
data(Agrawal)
```

Format

The format is: transactions Agrawal.db itemsets Agrawal.pat

Details

Agrawal.db contains the dataset (1000 items/20000 transactions) and Agrawal.pat contains the patterns that were used to create the dataset.

References

Rakesh Agrawal and Ramakrishnan Srikant (1994). Fast algorithms for mining association rules in large databases. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB*, pages 487-499, Santiago, Chile.

Examples

```
data(Agrawal)

summary(Agrawal.pat)
summary(Agrawal.db)

## the data sets was generated with the following code
## Not run:
Agrawal.pat <- random.patterns(1000, nPats = 2000, method = "agrawal",
  lPats = 2, corr = 0.5, cmean = 0.5, cvar = 0.1, iWeight = NULL,
  verbose = FALSE)
Agrawal.db <- random.transactions(1000, 20000, method="agrawal",
  patterns = Agrawal.pat)
## End(Not run)
```

Description

Calls the Java implementation of the depth first search algorithm described in the paper in the references section to mine NB-frequent itemsets of NB-precise rules.

Usage

```
NBMiner(data, parameter, control = NULL)
```

Arguments

<code>data</code>	object of class <code>transactions</code> .
<code>parameter</code>	a list of parameters (automatically converted into an object of class <code>NBMinerParameter</code>). Reasonable parameters can be obtained using <code>NBMinerParameters</code> (see details section).
<code>control</code>	a list of control options (automatically converted into an object of class <code>NBMinerControl</code>). Currently only "verbose" and "debug" (both logical) are available.

Details

The parameters can be estimated from the data using `NBMinerParameters`.

Value

An object of class `itemsets` or `rules` (depending on the rules entry in parameter). The estimated precision is stored in the quality slot.

References

Michael Hahsler. A model-based frequency constraint for mining associations from transaction data. *Data Mining and Knowledge Discovery*, 13(2):137-166, September 2006.

See Also

[NBMinerParameters](#), [transactions-class](#), [itemsets-class](#), [rules-class](#)

Examples

```
data("Agrawal")

## mine
param <- NBMinerParameters(Agrawal.db, pi=0.99, theta=0.5, maxlen=5,
  minlen=1, trim = 0, verb = TRUE, plot=TRUE)
itemsets_NB <- NBMiner(Agrawal.db, parameter = param,
  control = list(verb = TRUE, debug=FALSE))
```

```

inspect(head(itemsets_NB))

## remove patterns of length 1 (noise)
i_NB <- itemsets_NB[size(itemsets_NB)>1]
patterns <- Agrawal.pat[size(Agrawal.pat)>1]

## how many found itemsets are subsets of the patterns used in the db?
table(rowSums(is.subset(i_NB,patterns))>0)

## compare with the same number of the most frequent itemsets
itemsets_supp <- eclat(Agrawal.db, parameter=list(supp=0.001))
i_supp <- itemsets_supp[size(itemsets_supp) >1]
i_supp <- head(sort(i_supp, by = "support"), length(i_NB))
table(rowSums(is.subset(i_supp,patterns))>0)

## mine NB-precise rules
param <- NBMinerParameters(Agrawal.db, pi=0.99, theta=0.5, maxlen=5,
  rules=TRUE, minlen=1, trim = 0)
rules_NB <- NBMiner(Agrawal.db, parameter = param,
  control = list(verb = TRUE, debug=FALSE))

inspect(head(rules_NB))

```

NBMinerParameters *Estimate Global Model Parameters from Data*

Description

Estimate the global negative binomial data model used by the NBMiner and create an appropriate parameter object.

Usage

```

NBMinerParameters(data, trim = 0.01, pi = 0.99, theta = 0.5,
  minlen = 1, maxlen = 5, rules = FALSE,
  plot = FALSE, verbose = FALSE, getdata = FALSE)

```

Arguments

data	the data as a object of class transactions.
trim	fraction of incidences to trim off the tail of the frequency distribution of the data.
pi	precision threshold π .
theta	pruning parameter θ .
minlen	minimum number of items in found itemsets (default: 1).
maxlen	maximal number of items in found itemsets (default: 5).
rules	mine NB-precise rules instead of NB-frequent itemsets?

plot	plot the model?
verbose	use verbose output for the estimation procedure.
getdata	get also the observed and estimated counts.

Details

Uses the EM algorithm to estimate the global NB model for the data. The EM algorithm is used since the zero class (items which do not occur in the dataset) is not included in the data. The result are the two NB parameters k and a , where a is rescaled by dividing it by the number of incidences in the data (this is needed by the NBMiner). Also the real number of items n is a result of the estimation.

`theta` and `pi` are just taken and added to the resulting parameter object.

Value

an object of class `NBMinerParameter` for `NBMiner`.

References

Michael Hahsler. A model-based frequency constraint for mining associations from transaction data. *Data Mining and Knowledge Discovery*,13(2):137-166, September 2006.

See Also

[NBMiner](#), [transaction-class](#)

Examples

```
data("Epub")

param <- NBMinerParameters(Epub, trim = 0.001, plot = TRUE, verbose = TRUE)
param
```

Index

*Topic **datasets**

Agrawal, [2](#)

*Topic **models**

NBMiner, [3](#)

NBMinerParameters, [4](#)

Agrawal, [2](#)

itemsets-class, [3](#)

NBMiner, [3](#), [5](#)

NBMinerControl-class (*NBMiner*), [3](#)

NBMinerParameter-class (*NBMiner*),
[3](#)

NBMinerParameters, [3](#), [4](#)

rules-class, [3](#)

transaction-class, [5](#)

transactions-class, [3](#)