Package 'r.blip'

October 14, 2022

Title Bayesian Network Learning Improved Project

Version 1.1

Description Allows the user to learn Bayesian networks from datasets containing thousands of variables. It focuses on score-based learning, mainly the 'BIC' and the 'BDeu' score functions. It provides state-of-the-art algorithms for the following tasks: (1) parent set identification - Mauro Scanagatta (2015) <http://papers.nips.cc/paper/5803-learning-bayesian-networks-with-thousands-of-variables>; (2) general structure optimization - Mauro Scanagatta (2018) <doi:10.1007/s10994-018-5701-9>, Mauro Scanagatta (2018) <http://proceedings.mlr.press/v73/scanagatta17a.html>; (3) bounded treewidth structure optimization - Mauro Scanagatta (2016) <http://papers.nips.cc/paper/6232-learning-treewidth-bounded-bayesian-networks-with-thousands-of-variables>; (4) structure learning on incomplete data sets - Mauro Scanagatta (2018) <doi:10.1016/j.ijar.2018.02.004>. Distributed under the LGPL-3 by IDSIA.

Depends R (>= 3.0.0)

Imports foreign, bnlearn (>= 4.0)

SystemRequirements Java (>= 1.5)

License LGPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Author Mauro Scanagatta [aut, cre]

Maintainer Mauro Scanagatta <mauro@idsia.ch>

Repository CRAN

Date/Publication 2019-02-27 19:20:21 UTC

R topics documented:

blip 2
blip.learn 2

blip.learn.tw . . . 3
blip.scorer . . . 4
blip.solver . . . 5
blip.solver.tw . . . 6
child . . . 7
child.jkl . . . 7
read.jkl . . . 8
read.str . . . 8
write.jkl . . . 9

Index **10**

`blip` *Bayesian Learning Package - Main function.*

Description

Used by most of the functions in the r.blip binding, provides access to the included jar file.

Usage

```
blip(args)
```

Arguments

`args`	Vector of arguments to be passed to the jar

Details

The arguments vector is formatted in a system call to the included jar file. Should not be called directly by the user, unless you know exactly what you are doing. In that case, call directly the blip jar.

`blip.learn` *Learns a BN*

Description

Fully learns a Bayesian networks.

Usage

```
blip.learn(dat, scorer.method = "is", solver.method = "winasobs",
  indeg = 6, time = 3600, allocated = 80, scorefunction = "bic",
  alpha = 1, cores = 1, verbose = 0)
```

Arguments

dat	dataframe from which to learn the parent sets.(required)
scorer.method	Method to be used for scoring the parent sets. Possible values: "is" (independence selection), "sq" (sequential selection). (default: is)
solver.method	Method to be used for structure exploration. Possible values: "winasobs", "winobs", "asobs", "obs". (default: winasobs)
indeg	Maximum number of parents (default: 6)
time	Execution time (default: 3600)
allocated	Percentage of the total execution time dedicated to parent set exploration (default: 80)
scorefunction	Chosen score function. Possible choices: BIC, BDeu (default: bic)
alpha	(if BDeu is chosen) equivalent sample size parameter (default: 1.0)
cores	Number of machine cores to use. If 0, all are used. (default: 1)
verbose	Verbose level (default: 0)

Details

The input data is required to be complete and discrete. Accordingly missing values in the input data.frame will be ignored, and all numeric values will be converted to integers.

Value

The learned Bayesian network in the bnlearn format.

Examples

```
bn <- blip.learn(child, time=3)
```

---

blip.learn.tw *Learns a BN with a treewidth bound*

Description

Fully learns a Bayesian networks with a treewidth bound.

Usage

```
blip.learn.tw(dat, scorer.method = "is", solver.method = "kmax",
  treewidth = 5, time = 3600, allocated = 80,
  scorefunction = "bic", alpha = 1, cores = 1, verbose = 0)
```

Arguments

dat	dataframe from which to learn the parent sets.(required)
scorer.method	Method to be used for scoring the parent sets. Possible values: "is" (independence selection), "sq" (sequential selection). (default: is)
solver.method	Method to be used for bounded-treewidth structure exploration. Possible values: "kmax", "kg", "ka". (default: kmax)
treewidth	Maximum treewidth (default: 4)
time	Execution time (default: 3600)
allocated	Percentage of the total execution time dedicated to parent set exploration (default: 80)
scorefunction	Chosen score function. Possible choices: BIC, BDeu (default: bic)
alpha	(if BDeu is chosen) equivalent sample size parameter (default: 1.0)
cores	Number of machine cores to use. If 0, all are used. (default: 1)
verbose	Verbose level (default: 0)

Details

The input data is required to be complete and discrete. Accordingly missing values in the input data.frame will be ignored, and all numeric values will be converted to integers.

Value

The learned Bayesian network in the bnlearn format.

Examples

```
bn <- blip.learn.tw(child, treewidth=4, time=3)
```

---

blip.scorer *Parent set exploration*

Description

Generates the cache of parent sets from a given data source

Usage

```
blip.scorer(dat, method = "is", indeg = 6, time = 3600,
  scorefunction = "bic", alpha = 1, cores = 1, verbose = 0)
```

Arguments

dat	dataframe from which to learn the parent sets.(required)
method	Method to be used. Possible values: "is" (independence selection), "sq" (sequential selection). (default: is)
indeg	Maximum number of parents (default: 6)
time	Maximum Execution time (default: 3600)
scorefunction	Chosen score function. Possible choices: BIC, BDeu (default: bic)
alpha	(if BDeu is chosen) equivalent sample size parameter (default: 1.0)
cores	Number of machine cores to use. If 0, all are used. (default: 1)
verbose	Verbose level (default: 0)

Details

Usually the first step in the learning of a Bayesian network.

The input data is required to be complete and discrete. Accordingly missing values in the input data.frame will be ignored, and all numeric values will be converted to integers.

Value

Cache of parent sets

Examples

```
jkl <- blip.scorer(child, time=3)
```

---

blip.solver *Structure Optimization*

Description

Find an optimal structure from the cache of parent sets

Usage

```
blip.solver(jkl, method = "winasobs", time = 3600, cores = 1,
  verbose = 0)
```

Arguments

jkl	cache of pre-computed parent sets.(required)
method	Method to be used. Possible values: "winasobs", "winobs", "asobs", "obs". (default: winasobs)
time	Maximum Execution time (default: 3600)
cores	Number of machine cores to use. If 0, all are used. (default: 1)
verbose	Verbose level (default: 0)

Details

The input data is required to be complete and discrete. Accordingly missing values in the input data.frame will be ignored, and all numeric values will be converted to integers.

Value

Structure

Examples

```
bn <- blip.solver(child.jkl, time=3)
```

---

blip.solver.tw *Structure Optimization - treewidth bound*

Description

Find an optimal structure from the cache of parent sets

Usage

```
blip.solver.tw(jkl, method = "kmax", treewidth = 4, time = 3600,
  cores = 1, verbose = 0)
```

Arguments

jkl	cache of pre-computed parent sets.(required)
method	Method to be used. Possible values: "kmax", "kg", "ka". (default: kmax)
treewidth	Maximum treewidth (default: 4)
time	Maximum Execution time (default: 3600)
cores	Number of machine cores to use. If 0, all are used. (default: 1)
verbose	Verbose level (default: 0)

Details

The input data is required to be complete and discrete. Accordingly missing values in the input data.frame will be ignored, and all numeric values will be converted to integers.

Value

Structure

Examples

```
bn <- blip.solver.tw(child.jkl, time=3)
```

child *Child dataset*

Description

Dataset generated from the famous "child" network. Provided as an example of input data for learning a Bayesian network.

Usage

```
data("child")
```

Format

The format is: chr "child"

Details

Space separeted, integer values for each variable.

Source

http://www.bnlearn.com/bnrepository/discrete-medium.html#child

Examples

```
data(child)
```

child.jkl *Parent set cache for the child dataset*

Description

Parent set cache, taken from the "child" dataset.

read.jkl *Jkl reader*

Description

Read a Jkl file (parent sets cache)

Usage

```
read.jkl(path, names)
```

Arguments

path	Path of the file to load
names	List of variable names

Value

the cache of parent sets

read.str *Structure reader*

Description

Reads a str file (BN structure)

Usage

```
read.str(path, names)
```

Arguments

path	Path of the file to load
names	List of variable names

Value

the BN structure

`write.jkl` *Jkl writer (with names)*

Description

Write a Jkl file (parent sets cache)

Usage

```
write.jkl(path, jkl)
```

Arguments

`path`	Path of the file to write
`jkl`	parent sets cache to write

Index

∗ **datasets**
child, 7
∗ **parentset**
child.jkl, 7

blip, 2
blip.learn, 2
blip.learn.tw, 3
blip.scorer, 4
blip.solver, 5
blip.solver.tw, 6

child, 7
child.jkl, 7

read.jkl, 8
read.str, 8

write.jkl, 9