Package 'D2C'

February 19, 2015

Type Package

Title Predicting Causal Direction from Dependency Features

Version 1.2.1

Date 2015-01-14

Author Gianluca Bontempi, Catharina Olsen, Maxime Flauder

Maintainer Catharina Olsen <colsen@ulb.ac.be>

Description The relationship between statistical dependency and causality lies at the heart of all statistical approaches to causal inference. The D2C package implements a supervised machine learning approach to infer the existence of a directed causal link between two variables in multivariate settings with n>2 variables. The approach relies on the asymmetry of some conditional (in)dependence relations between the members of the Markov blankets of two variables causally connected. The D2C algorithm predicts the existence of a direct causal link between two variables in a multivariate setting by (i) creating a set of of features of the relationship based on asymmetric descriptors of the multivariate dependency and (ii) using a classifier to learn a mapping between the features and the presence of a causal link

License Artistic-2.0

Depends R(>= 2.10.0), randomForest

Imports gRbase, lazy, RBGL, MASS, corpcor, methods, Rgraphviz, foreach

LazyData true

Suggests knitr

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2015-01-21 00:23:55

R topics documented:

`alarm` *Alarm dataset*

Description

contains the adjacency matrix of the Alarm DAG (`true.net`) and the related measured dataset (`dataset`). See the vignette for an utilization of the dataset

contains the adjacency matrix of the Alarm DAG (`true.net`) and the related measured dataset (`dataset`). See the vignette for an utilization of the dataset

Details

Dataset alarm

Benchmark alarm

References

Aliferis C, Statnikov A, Tsamardinos I, Mani S, Koutsoukos X Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classif ication Part II: Analysis and Extensions' by ; JMLR 2010'

Aliferis C, Statnikov A, Tsamardinos I, Mani S, Koutsoukos X Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classif ication Part II: Analysis and Extensions' by ; JMLR 2010'

BER *Balanced Error Rate*

Description

The balanced error rate is the average of the errors on each class: BER = 0.5*(FP/(TN+FP) + FN/(FN+TP)).

Usage

```
BER(Ytrue, Yhat)
```

Arguments

Ytrue	: binary numeric vector (made of 0 or 1) of real classes
Yhat	: binary numeric vector (made of 0 or 1) of predicted classes

Value

Balanced Error Rate $0 \leq \text{BER} \leq 1$

compute,DAG.network-method *compute N samples according to the network distribution*

Description

compute N samples according to the network distribution

Usage

```
## S4 method for signature 'DAG.network'
compute(object, N = 50)
```

Arguments

object	a DAG.network object
N	numeric. the number of samples generated according to the network

Value

a N*nNodes matrix

D2C-class *An S4 class to store the RF model trained on the basis of the descriptors of NDAG DAGs*

Description

An S4 class to store the RF model trained on the basis of the descriptors of NDAG DAGs

DAG.network-class *An S4 class to store DAG.network*

Description

An S4 class to store DAG.network

Arguments

network : object of class "graph"

dataset *Dataset of the Alarm benchmark*

Description

contains the measured dataset. See the vignette for an utilization of the dataset

Details

Dataset of the Alarm benchmark

References

Aliferis C, Statnikov A, Tsamardinos I, Mani S, Koutsoukos X Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classif ication Part II: Analysis and Extensions' by ; JMLR 2010'

`descriptor` *compute descriptor*

Description

compute descriptor

Usage

```
descriptor(D, ca, ef, ns = min(4, NCOL(D) - 2), lin = FALSE, acc = TRUE,
  struct = TRUE, pq = c(0.1, 0.25, 0.5, 0.75, 0.9), bivariate = FALSE)
```

Arguments

`D`	: the observed data matrix of size [N,n], where N is the number of samples and n is the number of nodes
`ca`	: node index ($1 \leq ca \leq n$) of the putative cause
`ef`	: node index ($1 \leq ef \leq n$) of the putative effect
`ns`	: size of the Markov Blanket
`lin`	: TRUE OR FALSE. if TRUE it uses a linear model to assess a dependency, otherwise a local learning algorithm
`acc`	: TRUE OR FALSE. if TRUE it uses the accuracy of the regression as a descriptor
`struct`	: TRUE or FALSE to use the ranking in the markov blanket as a descriptor
`pq`	: a vector of quantiles used to compute de descriptor
`bivariate`	: TRUE OR FALSE. if TRUE it includes the descriptors of the bivariate dependency

Details

This function is the core of the D2C algorithm. Given two candidate nodes, (`ca`, putative cause and `ef`, putative effect) it first infers from the dataset D the Markov Blankets of the variables indexed by `ca` and `ef` (`MBca` and `MBef`) by using the mimr algorithm (Bontempi, Meyer, ICML10). Then it computes a set of (conditional) mutual information terms describing the dependency between the variables ca and ef. These terms are used to create a vector of descriptors. If `acc=TRUE`, the vector contains the descriptors related to the asymmetric information theoretic terms described in the paper. If `struct=TRUE`, the vector contains descriptors related to the positions of the terms of the MBef in MBca and viceversa. The estimation of the information theoretic terms require the estimation of the dependency between nodes. If `lin=TRUE` a linear assumption is made. Otherwise the local learning estimator, implemented by the R package lazy, is used.

References

Gianluca Bontempi, Maxime Flauder (2014) From dependency to causality: a machine learning approach. Under submission

Bontempi G., Meyer P.E. (2010) Causal filter selection in microarray data. ICML'10

M. Birattari, G. Bontempi, and H. Bersini (1999) Lazy learning meets the recursive least squares algorithm. Advances in Neural Information Processing Systems 11, pp. 375-381. MIT Press.

G. Bontempi, M. Birattari, and H. Bersini (1999) Lazy learning for modeling and control design. International Journal of Control, 72(7/8), pp. 643-658.

`example` *stored D2C object*

Description

small D2C object for testing D2C functionalities

Details

Dataset example

Examples

```
require(RBGL)
require(gRbase)
data(example)
print(example@mod)
## Random Forest
print(dim(example@X))
## dimension of the training set
```

`initialize,D2C-method` *creation of a D2C object which preprocesses the list of DAGs and observations contained in sDAG and fits a Random Forest classifier*

Description

creation of a D2C object which preprocesses the list of DAGs and observations contained in sDAG and fits a Random Forest classifier

Usage

```
## S4 method for signature 'D2C'
initialize(.Object, sDAG, descr = new("D2C.descriptor"),
  verbose = TRUE, ratioMissingNode = 0, ratioEdges = 1,
  max.features = 20, goParallel = FALSE)
```

Arguments

.Object	: the D2C object
sDAG	: simulateDAG object
descr	: D2C.descriptor object containing the parameters of the descriptor
verbose	: if TRUE it prints the state of progress
ratioMissingNode	: percentage of existing nodes which are not considered. This is used to emulate latent variables.
ratioEdges	: percentage of existing edges which are added to the training set
max.features	: maximum number of features used by the Random Forest classifier randomForest. The features are selected by the importance returned by the function importance.
goParallel	: if TRUE it uses parallelism

References

Gianluca Bontempi, Maxime Flauder (2014) From dependency to causality: a machine learning approach. Under submission

Examples

```
require(RBGL)
require(gRbase)
 require(foreach)
descr=new("D2C.descriptor")
descr.example<-new("D2C.descriptor",bivariate=FALSE,ns=3,acc=TRUE)
trainDAG<-new("simulatedDAG",NDAG=2, N=50,noNodes=10,
             functionType = "linear", seed=0,sdn=0.5)
example<-new("D2C",sDAG=trainDAG, descr=descr.example)
```

---

initialize,D2C.descriptor-method

creation of a D2C.descriptor

Description

creation of a D2C.descriptor

Usage

```
## S4 method for signature 'D2C.descriptor'
initialize(.Object, lin = TRUE, acc = TRUE,
  struct = TRUE, pq = c(0.1, 0.25, 0.5, 0.75, 0.9), bivariate = FALSE,
  ns = 4)
```

Arguments

.Object	: the D2C.descriptor object
lin	: TRUE OR FALSE: if TRUE it uses a linear model to assess a dependency, otherwise a local learning algorithm
acc	: TRUE OR FALSE: if TRUE it uses the accuracy of the regression as a descriptor
struct	: TRUE or FALSE to use the ranking in the markov blanket as a descriptor
pq	:a vector of quantiles used to compute the descriptors
bivariate	:TRUE OR FALSE: if TRUE it includes also the descriptors of the bivariate dependence
ns	: size of the Markov Blanket returned by the mIMR algorithm

References

Gianluca Bontempi, Maxime Flauder (2014) From dependency to causality: a machine learning approach. Under submission

Examples

```
require(RBGL)
require(gRbase)
require(foreach)
descr.example<-new("D2C.descriptor",bivariate=FALSE,ns=3,acc=TRUE)
trainDAG<-new("simulatedDAG",NDAG=2, N=50,noNodes=10,
             functionType = "linear", seed=0,sdn=0.5)
```

---

initialize,DAG.network-method

creation of a DAG.network

Description

creation of a DAG.network

Usage

```
## S4 method for signature 'DAG.network'
initialize(.Object, network, sdn = 0.5,
  sigma = function(x) return(rnorm(n = 1, sd = sdn)), H = function(x)
  return(H_Rn(1)))
```

Arguments

.Object	: DAG.network object
network	: object of class "graph"
sdn	: standard deviation of aditive noise.
sigma	: function returning the additive noise
H	: function describing the type of the dependency.

---

initialize,simulatedDAG-method

creation of a "simulatedDAG" containing a list of DAGs and associated observations

---

Description

creation of a "simulatedDAG" containing a list of DAGs and associated observations

Usage

```
## S4 method for signature 'simulatedDAG'
initialize(.Object, NDAG = 1,
  noNodes = sample(10:20, size = 1), functionType = "linear",
  quantize = FALSE, verbose = TRUE, N = sample(100:500, size = 1),
  seed = 1234, sdn = 0.5, goParallel = FALSE)
```

Arguments

.Object	: simulatedDAG object
NDAG	: number of DAGs to be created and simulated
noNodes	: number of Nodes of the DAGs. If it is a two-valued vector , the value of Nodes is randomly sampled in the interval
functionType	: type of the dependency. It is of class "character" and is one of ("linear", "quadratic","sigmoid")
quantize	: if TRUE it discretize the observations into two bins. If it is a two-valued vector [a,b], the value of quantize is randomly sampled in the interval [a,b]
verbose	: if TRUE it prints out the state of progress
N	: number of sampled observations for each DAG. If it is a two-valued vector [a,b], the value of N is randomly sampled in the interval [a,b]
seed	: random seed
sdn	: standard deviation of aditive noise. If it is a two-valued vector, the value of N is randomly sampled in the interval
goParallel	: if TRUE it uses parallelism

References

Gianluca Bontempi, Maxime Flauder (2014) From dependency to causality: a machine learning approach. Under submission

Examples

```
require(RBGL)
require(gRbase)
require(foreach)
descr=new("D2C.descriptor")
descr.example<-new("D2C.descriptor",bivariate=FALSE,ns=3,acc=TRUE)
trainDAG<-new("simulatedDAG",NDAG=10, N=c(50,100),noNodes=c(15,40),
             functionType = "linear", seed=0,sdn=c(0.45,0.75))
```

`mimr` *mIMR (minimum Interaction max Relevance) filter*

Description

Filter based on information theory which aims to prioritise direct causal relationships in feature selection problems where the ratio between the number of features and the number of samples is high. The approach is based on the notion of interaction which is informative about the relevance of an input subset as well as its causal relationship with the target.

Usage

```
mimr(X, Y, nmax = 5, init = FALSE, lambda = 0.5, spouse.removal = TRUE,
  caus = 1)
```

Arguments

`X`	: input matrix
`Y`	: output vector
`nmax`	: number of returned features
`init`	: if TRUE it makes a search in the space of pairs of features to initialize the ranking, otherwise the first ranked feature is the one with the highest mutual information with the output
`lambda`	: weight $0 \leq \lambda \leq 1$ of the interaction term
`spouse.removal`	: TRUE OR FALSE. if TRUE it removes the spouses before ranking
`caus`	: if `caus =1` it prioritizes causes otherwise (`caus=-1`) it prioritizes effects

Value

ranked vector of `nmax` indices of features

References

Bontempi G., Meyer P.E. (2010) Causal filter selection in microarray data. ICML10

Examples

```
set.seed(0)
N<-500
n<-5
X<-array(rnorm(N*n),c(N,n))
Y<-X[,1]-3*X[,3]+4*X[,2]+rnorm(N,sd=0.5)
Z1<-Y+rnorm(N,sd=0.5)
## effect 1
Z2<-2*Y+rnorm(N,sd=0.5)
## effect 2
most.probable.causes<-mimr(cbind(X,Z1,Z2),Y,nmax=3,init=TRUE,spouse=FALSE,lambda=1)
## causes are in the first three columns of the feature dataset
most.probable.effects<-mimr(cbind(X,Z1,Z2),Y,nmax=3,init=TRUE,spouse=FALSE,lambda=1,caus=-1)
## effects are in the last two columns of the feature dataset
```

---

predict,D2C-method *predict if there is a connection between node i and node j*

Description

predict if there is a connection between node i and node j

Usage

```
## S4 method for signature 'D2C'
predict(object, i, j, data)
```

Arguments

`object`	: a D2C object
`i`	: index of putative cause ($1 \leq i \leq n$)
`j`	: index of putative effect ($1 \leq j \leq n$)
`data`	: dataset of observations from the DAG

Value

list with response and prob of the prediction

References

Gianluca Bontempi, Maxime Flauder (2014) From dependency to causality: a machine learning approach. Under submission

Examples

```
require(RBGL)
require(gRbase)
require(foreach)
data(example)
## load the D2C object
testDAG<-new("simulatedDAG",NDAG=1, N=50,noNodes=5,
           functionType = "linear", seed=1,sdn=c(0.25,0.5))
## creates a simulatedDAG object for testing
plot(testDAG@list.DAGs[[1]])
## plot the topology of the simulatedDAG
predict(example,1,2, testDAG@list.observationsDAGs[[1]])
## predict if the edge 1->2 exists
predict(example,4,3, testDAG@list.observationsDAGs[[1]])
## predict if the edge 4->3 exists
predict(example,4,1, testDAG@list.observationsDAGs[[1]])
## predict if the edge 4->1 exists
```

---

simulatedDAG-class *An S4 class to store a list of DAGs and associated observations*

Description

An S4 class to store a list of DAGs and associated observations

Arguments

list.DAGs	: list of stored DAGs
list.observationsDAGs	: list of observed datasets, each sampled from the corresponding member of list.DAGs
NDAG	: number of DAGs.
functionType	: type of the dependency. It is of class "character" and is one of ("linear", "quadratic","sigmoid")
seed	: random seed

---

true.net *Adjacency matrix of the Alarm dataset*

Description

contains the adjacency matrix of the Alarm DAG. See the vignette for an utilization of the dataset

Details

Adjacency matrix of the Alarm benchmark

References

Aliferis C, Statnikov A, Tsamardinos I, Mani S, Koutsoukos X Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classif ication Part II: Analysis and Extensions' by ; JMLR 2010'

`update,simulatedDAG-method` *update of a "simulatedDAG" with a list of DAGs and associated observations*

Description

update of a "simulatedDAG" with a list of DAGs and associated observations

Usage

```
## S4 method for signature 'simulatedDAG'
update(object, list.DAGs, list.observationsDAGs)
```

Arguments

`object`	: simulatedDAG to be updated
`list.DAGs`	: list of stored DAGs
`list.observationsDAGs`	: list of observed datasets, each sampled from the corresponding member of list.DAGs

---

`updateD2C,D2C-method` *update of a "D2C" with a list of DAGs and associated observations*

Description

update of a "D2C" with a list of DAGs and associated observations

Usage

```
## S4 method for signature 'D2C'
updateD2C(object, sDAG, verbose = TRUE, goParallel = FALSE)
```

Arguments

`object`	: D2C to be updated
`sDAG`	: simulatedDAG object to update D2C
`verbose`	: TRUE or FALSE
`goParallel`	: if TRUE it uses parallelism

Index