Package 'ClassNoise'

January 11, 2024

Title Model, generate and validate data with class noise

Version 0.1.0

Description A framework to design class noise models based on datasets you provide. Generate synthetic datasets based on noise models and validate them using information metrics. This package empower the creation of controlled experimental settings to evaluate noise impact on learning tasks.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports bnlearn (>= 4.8.3), dplyr (>= 1.1.2)

Depends R (>= 4.3)

LazyData true

R topics documented:

discF2 *Volume of the overlapping region*

Description

Compute the ratio of overlapped examples in a discrete data frame.

Usage

```
discF2(data, class.idx = ncol(data))
```

Arguments

data	A data frame with discrete variables.
class.idx	An optional numerical index for the position of the class variable in the data frame. It is assumed the last variable in the data.

Value

A float number representing the volume of the overlapping region.

Examples

```
data(qb)
F2 <- discF2(qb)
F2
```

discF4 *Collective Feature Efficiency*

Description

Compute the ratio of examples with overlapping in a nominal data frame.

Usage

```
discF4(data, class.idx = ncol(data))
```

Arguments

data	A data frame with discrete variables.
class.idx	An optional numerical index for the position of the class variable in the data frame. It is assumed the last variable in the data.

Value

A list with the value of F4 (value), a nested list of the best attributes to separate examples (attribs), a nested list of indexes of examples with overlapping (overlapping) and a nested list of indexes of examples with no overlapping (nonOverlapping).

Examples

```
data(qb)
F4 <- discF4(qb[1:15,])
F4$value
F4$attribs
F4$overlapping
F4$nonOverlapping
```

`generateDF` *Generation of synthetic data*

Description

Generate a data frame with an specific number of examples according the distribution of a noise model.

Usage

```
generateDF(model, size)
```

Arguments

`size`	An integer representing the number of generated examples.
`data`	A data frame with discrete variables.

Value

A data frame.

Examples

```
data(qb)
model <- NCAR(data = qb, noise = 0.4S)
df <- generateDF(model$parameters, 10)
df
```

`inconsistency` *Percentage of inconsistent examples*

Description

Compute the percentage of inconsistent examples, i.e., a set of examples with the same attribute values and different class labels.

Usage

```
inconsistency(data, class.idx = ncol(data))
```

Arguments

data	A data frame with discrete variables.
class.idx	An optional numerical index for the position of the class variable in the data frame. It is assumed the last variable in the data.

Value

A list with the percentage of inconsistent examples (pct) and a nested list with the indexes of such examples (idx).

Examples

```
model <- NCAR(data = qb, noise = 0.1)
df <- generateDF(model$parameters, 100)
df <- df |> dplyr::mutate(Error = NULL,
  Class = ObservedClass, ObservedClass = NULL)
result <- inconsistency(df)
result$pct
result$idx
```

---

mux11 *11-input multiplexer data*

Description

Predict the output of a 11-input multiplexer. Creator: D. Martínez-Galicia (davidgalicia@outlook.es). Guided By : Dr. A. Guerra-Hernández (aguerra@uv.mx), Dr. Francisco Grimaldo (francisco.grimaldo@uv.es), Dr. Nicandro Cruz-Ramírez (ncruz@uv.mx), Dr. Xavier Limón (hlimon@uv.mx) Institution : Universidad Veracruzana, Mexico and Universitat de València, Spain.

Usage

```
data(mux11)
```

Format

mux11**:**

A data frame with 10240 rows and 12 columns which :

A0 Select line 0

A1 Select line 1

A2 Select line 2

D0 Channel 0

D1 Channel 1

D2 Channel 2

D3 Channel 3

D4 Channel 4

D5 Channel 5

D6 Channel 6

D7 Channel 7

Class Output

NAR *Noise At Random (NAR) model*

Description

This function generates and fits NAR models where each class label has a specific noise level. Noise levels are introduced as a numeric vector which follows the order of the class factor. By default, it employs a hill-climbing algorithm to discover the Bayesian Network structure. Alternatively, it can accept an bn object with a pre-defined structure. The data argument can receive data frames with numeric and categorical variables. The NAR function also assumes the last variable in the dataset serves as class and automatically renames it as 'Class' during model creation.

Usage

```
NAR(data, noise, class.idx = ncol(data), net = NULL)
```

Arguments

data	A data frame containing the domain variables.
noise	A number or a numeric vector of n elements between 0 and 1 representing the percentage of noise affecting each label. If a number is provided all clases have the same probability of error
class.idx	An optional numeric index for the position of the class variable in the data frame. It is assumed the last variable in the data.
net	A bn object representing the structure of the golden Bayesian Network of the domain.

Value

A list with a bn object representing the structure and a bn.fit object representing the parameterized model.

Examples

```
data(qb)

# The following examples have P(Error = True | Class = B) = 0.5
# and P(True | Error = Class = NB) = 0
# Running the method without a predefined network
model <- NAR(data = qb, noise = c(0.5, 0))
model$parameters$Error
# Running the method with a predefined network
network <- bnlearn::hc(qb)
fit <- bnlearn::bn.fit(network, qb)
model <- NAR(data = qb, noise = c(0.5, 0), net = fit)
model$parameters$Error
```

NCAR *Noisy Completely At Random (NCAR) model*

Description

This function generates and fits NCAR models with a specified noise level. By default, it employs a hill-climbing algorithm to discover the Bayesian Network structure. Alternatively, it can accept an bn object with a pre-defined structure. The `data` argument can receive data frames with numeric and categorical variables. The NCAR function also assumes the last variable in the dataset serves as class and automatically renames it as 'Class' during model creation.

Usage

```
NCAR(data, noise, class.idx = ncol(data), net = NULL)
```

Arguments

`data`	A data frame containing the domain variables.
`noise`	A number between 0 and 1 representing the noise level.
`class.idx`	An optional numeric index for the position of the class variable in the data frame. It is assumed the last variable in the data.
`net`	A bn object representing the structure of the golden Bayesian Network of the domain.

Value

A list with a bn object representing the structure and a `bn.fit` object representing the parameterized model.

Examples

```
data(qb)

# The following examples have P(Error = True) = 0.1
# Running the method without a predefined network
model <- NCAR(data = qb, noise = 0.1)
model$parameters$Error
# Running the method with a predefined network
network <- bnlearn::hc(qb)
fit <- bnlearn::bn.fit(network, qb)
model <- NCAR(data = qb, noise = 0.1, net = fit)
model$parameters$Error
```

NNAR *Noisy Not At Radom (NNAR) model*

Description

This function generates and fits NNAR models where each instance of an attribute set and the class has a specific noise level. Noise levels are introduced as a numeric vector which follows the order of the Cartesian product of their factors. The number of parameters increases with the arity of the attribute set and the possible values of each attribute. By default, it employs a hill-climbing algorithm to discover the Bayesian Network structure within the domain. Alternatively, it can accept an bn object with an pre-defined structure. The `data` argument just receives categorical data frames. The NNAR function also assumes the last variable in the dataset serves as class and automatically renames it as 'Class' during model creation.

Usage

```
NNAR(data, attrib.set, noise, class.idx = ncol(data), net = NULL)
```

Arguments

`data`	A data frame containing categorical variables.
`attrib.set`	A vector of strings representing the attributes related with Error.
`noise`	A number or a numeric vector of `n` elements between 0 and 1 representing the percentage of noise affecting each label. If a number is provided all possible scenarios have the same probability of error
`class.idx`	An optional numeric index for the position of the class variable in the data frame. It is assumed the last variable in the data.
`net`	A bn object representing the structure of the golden Bayesian Network of the domain.

Value

A list with a bn object representing the structure and a `bn.fit` object representing the parameterized model.

Examples

```
data(qb)

# This examples have P(Error = True | IR = P, Class = B) = 0.4,
# P(Error = True | IR = A, Class = B) = 0),
# P(Error = True | IR = N, Class = B) = 0.2,
# P(Error = True | IR = A, Class = NB) = 0),
# P(Error = True | IR = N, Class = NB) = 0.1,
# and P(Error = True | IR = A, Class = NB) = 0.3)
# Running the method without a predefined network
model <- NNAR(data = qb, attrib.set = c("IR"),
 noise = c(0.4, 0, 0.2, 0, 0.1, 0.3))
model$parameters$Error
# Running the method with a predefined network
network <- bnlearn::hc(qb)
```

```
fit <- bnlearn::bn.fit(network, qb)
model <- NNAR(data = qb, attrib.set = c("IR"),
  noise = c(0.4, 0, 0.2, 0, 0.1, 0.3), net = fit)
model$parameters$Error
```

---

noiseLVL *Noise level*

Description

Compute the noise level of data frame generated with the funtions of noise models.

Usage

```
noiseLVL(data)
```

Arguments

data	A data frame.

Value

A float number.

Examples

```
data(qb)
model <- NCAR(data = qb, noise = 0.4)
df <- generateDF(model$parameters, 1000)
result <- noiseLVL(df)
result
```

---

qb *Qualitative Bankruptcy Database*

Description

Predict the Bankruptcy from Qualitative parameters from experts. Creator: Mr. A. Martin (jayamartin@yahoo.com), Mr. J. Uthayakumar (uthayakumar17691@gmail.com), Mr. M. Nadarajan (nadaraj.muthuvel@gmail.com). Guided By : Dr. V. Prasanna Venkatesan. Institution : Sri Manakula Vinayagar Engineering College and Pondicherry University, India,

Usage

```
data(qb)
```

Format

qb**:**

A data frame with 250 rows and 7 columns:

IR Industrial Risk

MR Management Risk

FF Financial Flexibility

CR Credibility

CO Competitiveness

OP Operation Risk

Class Bankruptcy or Nonbankruptcy

Source

https://doi.org/10.24432/C52889

Index