Package ‘MDFs’

February 9, 2021

**Title**  MultiDimensional Feature Selection

**Version**  1.2.0

**Date**  2021-02-09

**URL**  https://featureselector.uco.uwb.edu.pl/software/mdfs/

**Description**  Functions for MultiDimensional Feature Selection (MDFS):
- calculating multidimensional information gains, scoring variables,
- finding important variables, plotting selection results.
This package includes an optional CUDA implementation that speeds up information gain calculation using NVIDIA GPGPUs.

**Depends**  R (&gt;= 3.4.0)

**License**  GPL-3

**SystemRequirements**  C++11

**NeedsCompilation**  yes

**Encoding**  UTF-8

**LazyData**  true

**RoxygenNote**  7.1.1

**Author**  Radosław Piliszek [aut, cre],
            Krzysztof Mnich [aut],
            Paweł Tabaszewski [aut],
            Szymon Migacz [aut],
            Andrzej Sulecki [aut],
            Witold Remigiusz Rudnicki [aut]

**Maintainer**  Radoslaw Piliszek &lt;r.piliszek@uwb.edu.pl&gt;

**Repository**  CRAN

**Date/Publication**  2021-02-09 18:00:02 UTC

**R topics documented:**

AddContrastVariables .......................................................... 2
AddContrastVariables

Add contrast variables to data

Description
Add contrast variables to data

Usage
AddContrastVariables(data, n.contrast = max(ncol(data)/10, 30))

Arguments
- data: data organized in matrix with separate variables in columns
- n.contrast: number of contrast variables (defaults to max of 1/10 of variables number and 30)

Value
A list with the following key names:
- indices: vector of indices of input variables used to construct contrast variables
- x: data with contrast variables appended to it
- mask: vector of booleans making it easy to select just contrast variables

Examples
AddContrastVariables(madelon$data)
as.data.frame.MDFS

Description

as.data.frame S3 method implementation for MDFS

Usage

## S3 method for class 'MDFS'
as.data.frame(x, ...)

Arguments

x      an MDFS object
...    ignored

Value

data.frame

ComputeInterestingTuples

Description

Interesting tuples

Usage

ComputeInterestingTuples(
  data,
  decision,
  dimensions = 2,
  divisions = NULL,
  discretizations = 1,
  seed = NULL,
  range = NULL,
  pc.xi = 0.25,
  ig.thr = 0,
  I.lower,
  interesting.vars = vector(mode = "integer"),
  require.all.vars = FALSE,
  return.matrix = FALSE
)
ComputeInterestingTuples

Arguments

data          input data where columns are variables and rows are observations (all numeric)
decision      decision variable as a binary sequence of length equal to number of observations
dimensions    number of dimensions (a positive integer; 5 max) - FIXME: only 2D supported for now!
divisions     number of divisions (from 1 to 15; NULL selects probable optimal number)
discretizations number of discretizations
seed          seed for PRNG used during discretizations (NULL for random)
range         discretization range (from 0.0 to 1.0; NULL selects probable optimal number)
pc.xi          parameter xi used to compute pseudocounts (the default is recommended not to be changed)
ig.thr        IG threshold above which the tuple is interesting (0 and negative mean no filtering)
I.lower       IG values computed for lower dimension (1D for 2D, etc.)
ing.interesting.vars variables for which to check the IGs (none = all)
require.all.vars boolean whether to require tuple to consist of only interesting.vars
return.matrix boolean whether to return a matrix instead of a list (ignored if not using the optimised method variant)

Details

If no filtering is applied, this function is able to run in an optimised fashion. It is recommended to avoid filtering if only it is feasible.

Value

A data.frame or NULL (following a warning) if no tuples are found.

The following columns are present in the data.frame:

- Var – interesting variable index
- Tuple.1, Tuple.2,... – corresponding tuple (up to dimensions columns)
- IG – information gain achieved by var in Tuple.*

Additionally attribute named run.params with run parameters is set on the result.

Examples

ig.1d <- ComputeMaxInfoGains(madelon$data, madelon$decision, dimensions = 1, divisions = 1, range = 0, seed = 0)
ComputeInterestingTuples(madelon$data, madelon$decision, dimensions = 2, divisions = 1, range = 0, seed = 0, ig.thr = 100, I.lower = ig.1d$IG)
ComputeMaxInfoGains

Max information gains

Description

Max information gains

Usage

```
ComputeMaxInfoGains(
  data,
  decision,
  dimensions = 1,
  divisions = NULL,
  discretizations = 1,
  seed = NULL,
  range = NULL,
  pc.xi = 0.25,
  return.tuples = FALSE,
  return.min = FALSE,
  interesting.vars = vector(mode = "integer"),
  require.all.vars = FALSE,
  use.CUDA = FALSE
)
```

Arguments

data  input data where columns are variables and rows are observations (all numeric)
decision  decision variable as a binary sequence of length equal to number of observations
dimensions  number of dimensions (a positive integer; 5 max)
divisions  number of divisions (from 1 to 15; additionally limited by dimensions if using CUDA; NULL selects probable optimal number)
discretizations  number of discretizations
seed  seed for PRNG used during discretizations (NULL for random)
range  discretization range (from 0.0 to 1.0; NULL selects probable optimal number)
pc.xi  parameter xi used to compute pseudocounts (the default is recommended not to be changed)
return.tuples  whether to return tuples (and relevant discretization number) where max IG was observed (one tuple and relevant discretization number per variable) - not supported with CUDA nor in 1D
return.min  whether to return min instead of max (per tuple, always max per discretization) - not supported with CUDA
interesting.vars
variables for which to check the IGs (none = all) - not supported with CUDA
require.all.vars
boolean whether to require tuple to consist of only interesting.vars
use.CUDA
whether to use CUDA acceleration (must be compiled with CUDA)

Value

A `data.frame` with the following columns:

- **IG** – max information gain (of each variable)
- **Tuple.1, Tuple.2, ...** – corresponding tuple (up to dimensions columns, available only when `return.tuples == T`)
- **Discretization.nr** – corresponding discretization number (available only when `return.tuples == T`)

Additionally attribute named `run.params` with run parameters is set on the result.

Examples

```r
ComputeMaxInfoGains(madelon$data, madelon$decision, dimensions = 2, divisions = 1,
range = 0, seed = 0)
```

---

**ComputePValue**

*Compute p-values from information gains and return MDFS*

**Description**

Compute p-values from information gains and return MDFS

**Usage**

```r
ComputePValue(
  IG,            
  dimensions,    
  divisions,     
  response.divisions = 1,     
  df = NULL,     
  contrast.mask = NULL,      
  ig.in.bits = TRUE,        
  ig.doubled = FALSE,       
  one.dim.mode = "exp",      
  irr.vars.num = NULL,     
  ign.low.ig.vars.num = NULL,   
  min.irr.vars.num = NULL,
```
max.ign.low.ig.vars.num = NULL,
search.points = 8,
level = 0.05
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IG</td>
<td>max conditional information gains</td>
</tr>
<tr>
<td>dimensions</td>
<td>number of dimensions</td>
</tr>
<tr>
<td>divisions</td>
<td>number of divisions</td>
</tr>
<tr>
<td>response.divisions</td>
<td>number of response divisions (i.e. categories-1)</td>
</tr>
<tr>
<td>df</td>
<td>vector of degrees of freedom for each variable (optional)</td>
</tr>
<tr>
<td>contrast.mask</td>
<td>boolean mask on IG specifying which variables are contrast variables (or NULL if none, otherwise at least 3 variables must be marked)</td>
</tr>
<tr>
<td>ig.in.bits</td>
<td>TRUE if input is in binary log (as opposed to natural log)</td>
</tr>
<tr>
<td>ig.doubled</td>
<td>TRUE if input is doubled (to follow the chi-squared distribution)</td>
</tr>
<tr>
<td>one.dim.mode</td>
<td>'exp' for exponential distribution, 'lin' for linear function of chi-squared or 'raw' for raw chi-squared</td>
</tr>
<tr>
<td>irr.vars.num</td>
<td>if not NULL, number of irrelevant variables, specified by the user</td>
</tr>
<tr>
<td>ign.low.ig.vars.num</td>
<td>if not NULL, number of ignored low IG variables, specified by the user</td>
</tr>
<tr>
<td>min.irr.vars.num</td>
<td>minimum number of irrelevant variables (NULL selects probable optimal number)</td>
</tr>
<tr>
<td>max.ign.low.ig.vars.num</td>
<td>maximum number of ignored low IG variables (NULL selects probable optimal number)</td>
</tr>
<tr>
<td>search.points</td>
<td>number of points in search procedure for the optimal number of ignored variables</td>
</tr>
<tr>
<td>level</td>
<td>acceptable error level of goodness-of-fit one-sample Kolmogorov-Smirnov test (used only for warning)</td>
</tr>
</tbody>
</table>

Value

A data.frame with class set to MDFS. Can be coerced back to data.frame using as.data.frame.

The following columns are present:

- IG – information gains (input copy)
- chi.squared.p.value – chi-squared p-values
- p.value – theoretical p-values

Additionally the following attributes are set:

- run.params – run parameters
• sq.dev – vector of square deviations used to estimate the number of irrelevant variables
• dist.param – distribution parameter
• err.param – squared error of the distribution parameter
• fit.p.value – p-value of fit

Examples
ComputePValue(madelon$IG.2D, dimensions = 2, divisions = 1)

Discretize variable on demand

Description
Discretize variable on demand

Usage
Discretize(data, variable.idx, divisions, discretization.nr, seed, range)

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>input data where columns are variables and rows are observations (all numeric)</td>
</tr>
<tr>
<td>variable.idx</td>
<td>variable index (as it appears in data)</td>
</tr>
<tr>
<td>divisions</td>
<td>number of divisions</td>
</tr>
<tr>
<td>discretization.nr</td>
<td>discretization number (positive integer)</td>
</tr>
<tr>
<td>seed</td>
<td>seed for PRNG</td>
</tr>
<tr>
<td>range</td>
<td>discretization range</td>
</tr>
</tbody>
</table>

Value
Discretized variable.

Examples
Discretize(madelon$data, 3, 1, 1, 0, 0.5)
madelon

An artificial dataset called MADELON

Description

An artificial dataset containing data points grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled 0/1.

Usage

madelon

Format

A list of two elements:

- **data** 2000 by 500 matrix of 2000 objects with 500 features
- **decision** vector of 2000 decisions (labels 0/1)
- **IG.2D** example 2D IG computed using ComputeMaxInfoGains

Details

The five dimensions constitute 5 informative features. 15 linear combinations of those features are added to form a set of 20 (redundant) informative features. There are 480 distractor features called 'probes' having no predictive power.

Included is the original training set with label -1 changed to 0.

Source

https://archive.ics.uci.edu/ml/datasets/Madelon

---

MDFS

Run end-to-end MDFS

Description

Run end-to-end MDFS
Usage

MDFS(
  data,  
decision, 
n.contrast = max(ncol(data)/10, 30), 
dimensions = 1, 
divisions = NULL, 
discretizations = 1, 
range = NULL, 
pc.xi = 0.25, 
p.adjust.method = "holm", 
level = 0.05, 
seed = NULL, 
use.CUDA = FALSE
)

Arguments

data     input data where columns are variables and rows are observations (all numeric)
decision  decision variable as a boolean vector of length equal to number of observations
n.contrast number of contrast variables (defaults to max of 1/10 of variables number and 30)
dimensions number of dimensions (a positive integer; on CUDA limited to 2–5 range)
divisions number of divisions (from 1 to 15; NULL selects probable optimal number)
discretizations number of discretizations
range     discretization range (from 0.0 to 1.0; NULL selects probable optimal number)
pc.xi     parameter xi used to compute pseudocounts (the default is recommended not to be changed)
p.adjust.method method as accepted by p.adjust ("BY" is recommended for FDR, see Details)
level     statistical significance level
seed      seed for PRNG used during discretizations (NULL for random)
use.CUDA  whether to use CUDA acceleration (must be compiled with CUDA)

Details

In case of FDR control it is recommended to use Benjamini-Hochberg-Yekutieli p-value adjustment method ("BY" in p.adjust) due to unknown dependencies between tests.

Value

A list with the following fields:

- contrast.indices – indices of variables chosen to build contrast variables
- contrast.variables – built contrast variables
plot.MDFS

- MIG.Result – result of ComputeMaxInfoGains
- MDFS – result of ComputePValue (the MDFS object)
- statistic – vector of statistic’s values (IGs) for corresponding variables
- p.value – vector of p-values for corresponding variables
- adjusted.p.value – vector of adjusted p-values for corresponding variables
- relevant.variables – vector of relevant variables indices

Examples

MDFS(madelon$data, madelon$decision, dimensions = 2, divisions = 1,
    range = 0, seed = 0)

plot.MDFS

Plot MDFS details

Description

Plot MDFS details

Usage

## S3 method for class 'MDFS'
plot(x, plots = c("ig", "c", "p"), ...)

Arguments

- x
  - an MDFS object
- plots
  - plots to plot (ig for max IG, c for chi-squared p-values, p for p-values)
- ...
  - passed on to plot

RelevantVariables

Find indices of relevant variables

Description

Find indices of relevant variables

Usage

RelevantVariables(fs, ...)

Arguments

- `fs` feature selector
- `...` arguments passed to methods

Value

- indices of important variables

---

RelevantVariables.MDFS

*Find indices of relevant variables from MDFS*

---

Description

Find indices of relevant variables from MDFS

Usage

```r
## S3 method for class 'MDFS'
RelevantVariables(fs, level = 0.05, p.adjust.method = "holm", ...)
```

Arguments

- `fs` an MDFS object
- `level` statistical significance level
- `p.adjust.method` method as accepted by `p.adjust` ("BY" is recommended for FDR, see Details)
- `...` ignored

Details

In case of FDR control it is recommended to use Benjamini-Hochberg-Yekutieli p-value adjustment method ("BY" in `p.adjust`) due to unknown dependencies between tests.

Value

- indices of relevant variables
Index

* datasets
  - madelon, 9

  AddContrastVariables, 2
  as.data.frame, 7
  as.data.frame.MDFS, 3
  attributes, 7
  ComputeInterestingTuples, 3
  ComputeMaxInfoGains, 5
  ComputePValue, 6

  data.frame, 4, 6, 7
  Discretize, 8

  list, 10

  madelon, 9
  MDFS, 9

  NULL, 4

  p.adjust, 10, 12
  plot, 11
  plot.MDFS, 11

  RelevantVariables, 11
  RelevantVariables.MDFS, 12