# Package 'praznik'

May 8, 2018

**Type** Package

**Title** Collection of Information-Based Feature Selection Filters

**Version** 5.0.0

**Depends** R (>= 2.10)

**License** GPL-3

**Description** A collection of feature selection filters performing greedy optimisation of mutual information-based usefulness criteria, inspired by the overview by Brown, Pocock, Zhao and Lujan (2012) <http://www.jmlr.org/papers/v13/brown12a.html>.
Implements, among other, minimum redundancy maximal relevancy ('mRMR') method by Peng, Long and Ding (2005) <doi:10.1109/TPAMI.2005.159>; joint mutual information ('JMI') method by Yang and Moody (1999) <http://papers.nips.cc/paper/1779-data-visualization-and-feature-selection-new-algorithms-for-nongaussian-data>; double input symmetrical relevance ('DISR') method by Meyer and Bontempi (2006) <doi:10.1007/11732242_9> as well as joint mutual information maximisation ('JMIM') method by Bennasar, Hicks and Setchi (2015) <doi:10.1016/j.eswa.2015.07.007>.

**BugReports** https://github.com/mbq/praznik/issues

**URL** https://github.com/mbq/praznik

**RoxygenNote** 6.0.1.9000

**Suggests** testthat

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Miron B. Kursa [aut, cre] (<https://orcid.org/0000-0001-7672-648X>)

**Maintainer** Miron B. Kursa <M.Kursa@icm.edu.pl>

**Repository** CRAN

**Date/Publication** 2018-05-08 15:43:35 UTC

## R topics documented:

---

`praznik-package` *Collection of Information-Based Feature Selection Filters*

---

## Description

Praznik is a collection of feature selection filters performing greedy optimisation of mutual information-based usefulness criteria.

## Details

In a nutshell, an algorithm of this class requires an information system $(X, Y)$ and a predefined number of features to selected $k$, and works like this. To start, it estimates mutual information between each feature and the decision, find a feature with maximal such score and stores it as a first on a list of selected features, $S$. Then, it estimates a value of a certain criterion function $J(X, Y, S)$ for each feature $X$; this function also depends on $Y$ and the set of already selected features $S$. As in the first step, the previously unselected feature with a greatest value of the criterion function is selected next. This is repeated until the method would gather $k$ features, or, in case of some methods, when no more informative features can be found. The methods implemented in praznik consider the following criteria.

The mutual information maximisation filter, `MIM`, simply selects top-$k$ features of best mutual information, that is

$$J_{MIM} = I(X;Y).$$

The minimal conditional mutual information maximisation proposed by F. Fleauret, `CMIM`, uses

$$J_{CMIM}(X) = \min_{W \in S} I(X;Y|W);$$

this method is also effectively identical to the information fragments method.

The minimum redundancy maximal relevancy proposed by H. Peng et al., `MRMR`, uses

$$J_{MRMR} = I(X;Y) - \frac{1}{|S|} \sum_{W \in S} I(X;W).$$

The joint mutual information filter by H. Yang and J. Moody, JMI, uses

$$J_{JMI} = \sum_{W \in S} I(X, W; Y).$$

The double input symmetrical relevance filter by P. Meyer and G. Bontempi, DISR, uses

$$J_{DISR}(X) = \sum_{W \in S} \frac{I(X, W; Y)}{H(X, W, Y)}.$$

The minimal joint mutual information maximisation filter by M. Bennasar, Y. Hicks and R. Setchi, JMIM, uses

$$J_{JMIM} = \min_{W \in S} I(X, W; Y).$$

The minimal normalised joint mutual information maximisation filter by the same authors, NJMIM, uses

$$J_{NJMIM} = \min_{W \in S} \frac{I(X, W; Y)}{H(X, W, Y)}.$$

While CMIM, JMIM and NJMIM consider minimal value over already selected features, they may use a somewhat more sophisticated and faster algorithm.

The package also provides functions for scoring features.

miScores returns

$$I(X; Y).$$

cmiScores returns, for a given condition vector $Z$,

$$I(X; Y|Z).$$

jmiScores returns

$$I(X, Z; Y).$$

njmiScores returns

$$\frac{I(X, Z; Y)}{H(X, Y, Z)}.$$

Estimation of mutual information and its generalisations is a hard task; still, praznik aims at speed and simplicity and hence only offers basic, maximum likelihood estimator applicable on discrete data. For convenience, praznik automatically and silently coerces non-factor inputs into about ten equally-spaced bins, following the heuristic often used in literature.

Additionally, praznik has a limited, experimental support for replacing entropic statistics with Gini impurity-based; in such framework, entropy is replaced by Gini impurity

$$g(X) := 1 - \sum_x p_x^2,$$

which leads to an impurity gain

$$G(X; Y) := g(Y) - E(g(Y)|X) = \sum_{xy} \frac{p_{xy}^2}{p_x} - \sum_y p_y^2,$$

a counterpart of mutual information or information gain. It does not possess most of elegant properties of mutual information, yet values of both are usually highly correlated; moreover, Gini gain is computationally easier to calculate, hence it often replaces MI in performance-sensitive applications, like optimising splits in decision trees.

In a present version, praznik includes `impScores` for generating values of $G$ for all features (an analog of `miScores`, as well as `JIM`, a Gini gain-based feature selection method otherwise identical to `JMI`.

## Author(s)

**Maintainer**: Miron B. Kursa <M.Kursa@icm.edu.pl> (0000-0001-7672-648X)

## References

"Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection" G. Brown et al. JMLR (2012).

## See Also

Useful links:

- https://github.com/mbq/praznik
- Report bugs at https://github.com/mbq/praznik/issues

---

`CMIM` *Minimal conditional mutual information maximisation filter*

---

## Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \min_{W \in S} I(X;Y|W),$$

where $S$ is the set of already selected attributes.

## Usage

```
CMIM(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `k` | Number of attributes to select. Must not exceed `ncol(X)`. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: `selection`, a vector of indices of the selected features in the selection order, and `score`, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in `X`. Both vectors will be at most of a length `k`, as the selection will stop as soon as all the remaining features will have a score of zero. This may happen during initial selection, in which case both vectors will be empty.

## Note

CMIM is identical to the Informative Fragments (IF) method.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## References

"Fast Binary Feature Selection using Conditional Mutual Information Maximisation" F. Fleuret, JMLR (2004)

"Object recognition with informative features and linear classification" M. Vidal-Naquet and S. Ullman, IEEE Conference on Computer Vision and Pattern Recognition (2003).

## Examples

```
data(MadelonD)
CMIM(MadelonD$X,MadelonD$Y,20)
```

---

# cmiScores *Calculate conditional mutual information of all features*

---

## Description

Calculates mutual information between each attributes and the decision, that is

$$I(X, Y|Z).$$

## Usage

```
cmiScores(X, Y, Z, threads = 0)
```

### Arguments

| | |
|---|---|
| X | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). NAs are not allowed. |
| Y | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. NAs are not allowed. |
| Z | Condition; should be given as a factor, but other options are accepted, as for attributes. |
| threads | Number of threads to use; default value, 0, means all available to OpenMP. |

### Value

A numerical vector with conditional mutual information scores, with names copied from X.

### Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

### Examples

```
cmiScores(iris[,-5],iris$Species,iris$Sepal.Length)
```

---

## DISR *Double input symmetrical relevance filter*

---

### Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \sum_{W \in S} \frac{I(X, W; Y)}{H(X, W, Y)},$$

where $S$ is the set of already selected attributes.

## Usage

```
DISR(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| X | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). NAs are not allowed. |
| Y | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. NAs are not allowed. |
| k | Number of attributes to select. Must not exceed ncol(X). |
| threads | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: selection, a vector of indices of the selected features in the selection order, and score, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in X. Both vectors will either have a length k or zero, when all features turn out to have zero mutual information with the decision.

## Note

DISR is a normalised version of JMI; JMIM and NJMIM are modifications of JMI and DISR in which minimal joint information over already selected attributes is used instead of a sum.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## References

"On the Use of Variable Complementarity for Feature Selection in Cancer Classification" P. Meyer and G. Bontempi, (2006)

## Examples

```
data(MadelonD)
DISR(MadelonD$X,MadelonD$Y,20)
```

`impScores` *Calculate Gini impurity scores of all features*

## Description

Calculates Gini impurity between each attribute and the decision, that is

$$G(X;Y) = \sum_{xy} \frac{p_{xy}^2}{p_x} - \sum_y p_y^2.$$

## Usage

```
impScores(X, Y, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A numerical vector with Gini impurity scores, with names copied from `X`.

## Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## Examples

```
impScores(iris[,-5],iris$Species)
```

JIM *Joint impurity filter*

## Description

The method starts with an attribute of a maximal impurity gain with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \sum_{W \in S} G(X, W; Y),$$

where $S$ is the set of already selected attributes, and

$$G(X; Y) = \sum_{xy} \frac{p_{xy}^2}{p_x} - \sum_{y} p_y^2$$

is the Gini impurity gain from partitioning $Y$ according to $X$.

## Usage

```
JIM(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `k` | Number of attributes to select. Must not exceed `ncol(X)`. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: `selection`, a vector of indices of the selected features in the selection order, and `score`, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in `X`. Both vectors will either have a length `k` or zero, when all features turn out to have zero mutual information with the decision.

## Note

This is an impurity-based version of `JMI`; expect similar results in slightly shorter time.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function.

Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## Examples

```
data(MadelonD)
JIM(MadelonD$X,MadelonD$Y,20)
```

---

`JMI` *Joint mutual information filter*

---

## Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \sum_{W \in S} I(X, W; Y),$$

where $S$ is the set of already selected attributes.

## Usage

```
JMI(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `k` | Number of attributes to select. Must not exceed `ncol(X)`. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: `selection`, a vector of indices of the selected features in the selection order, and `score`, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in `X`. Both vectors will either have a length `k` or zero, when all features turn out to have zero mutual information with the decision.

### Note

DISR is a normalised version of JMI; JMIM and NJMIM are modifications of JMI and DISR in which minimal joint information over already selected attributes is used instead of a sum.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

### References

"Data Visualization and Feature Selection: New Algorithms for Nongaussian Data H. Yang and J. Moody, NIPS (1999)

### Examples

```
data(MadelonD)
JMI(MadelonD$X,MadelonD$Y,20)
```

---

## JMIM *Minimal joint mutual information maximisation filter*

---

### Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \min_{W \in S} I(X, W; Y),$$

where $S$ is the set of already selected attributes.

### Usage

```
JMIM(X, Y, k = 3, threads = 0)
```

### Arguments

| | |
|---|---|
| X | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). NAs are not allowed. |

| | |
|---|---|
| Y | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. NAs are not allowed. |
| k | Number of attributes to select. Must not exceed ncol(X). |
| threads | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: selection, a vector of indices of the selected features in the selection order, and score, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in X. Both vectors will be at most of a length k, as the selection will stop as soon as all the remaining features will have a score of zero. This may happen during initial selection, in which case both vectors will be empty.

## Note

NJMIM is a normalised version of JMIM; JMI and DISR are modifications of JMIM and NJMIM in which a sum of joint information over already selected attributes is used instead of a minimum.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## References

"Feature selection using Joint Mutual Information Maximisation" M. Bennasar, Y. Hicks and R. Setchi, (2015)

## Examples

```
data(MadelonD)
JMIM(MadelonD$X,MadelonD$Y,20)
```

---

jmiScores *Calculate joint mutual information of all features*

---

## Description

Calculated mutual information between each attribute joint with some other vector Z with the decision, that is

$$I(X, Z; Y).$$

This is the same as conditional mutual information between X and Y plus a constant that depends on Y and Z, that is

$$I(X, Z; Y) = I(X; Y|Z) + I(Y; Z).$$

## Usage

```
jmiScores(X, Y, Z, threads = 0)
```

## Arguments

| | |
|---|---|
| X | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). NAs are not allowed. |
| Y | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. NAs are not allowed. |
| Z | Other vector; should be given as a factor, but other options are accepted, as for attributes. |
| threads | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A numerical vector with joint mutual information scores, with names copied from X.

## Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## Examples

```
jmiScores(iris[,-5],iris$Species,iris$Sepal.Length)
```

## MadelonD — *Pre-discretised Madelon dataset*

### Description

Madelon is a synthetic data set from the NIPS 2003 feature selection challenge, generated by Isabelle Guyon. It contains 480 irrelevant and 20 relevant features, including 5 informative and 15 redundant. In this version, the originally numerical attributes have been pre-discetised into 10 bins, as well as their names have been altered to reveal 20 relevant features (as identified by the Boruta method).

### Usage

```
MadelonD
```

### Format

A list with two elements, `X` containing a data frame with predictors, and `Y`, the decision. Features are in the same order as in the original data; the names of relevant ones start with `Rel`, while of irrelevant ones with `Irr`.

### Source

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

## MIM — *Mutual information maximisation filter*

### Description

Calculates mutual information between all attributes and the decision, then returns top k.

### Usage

```
MIM(X, Y, k = 3, threads = 0)
```

### Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `k` | Number of attributes to select. Must not exceed `ncol(X)`. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

### Value

A list with two elements: `selection`, a vector of indices of the selected features in the selection order, and `score`, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in `X`. Both vectors will be at most of a length `k`, as the selection will stop as soon as all the remaining features will have a score of zero. This may happen during initial selection, in which case both vectors will be empty.

### Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

### Examples

```
data(MadelonD)
MIM(MadelonD$X,MadelonD$Y,20)
```

---

## `miScores` *Calculate mutual information of all features*

---

### Description

Calculates mutual information between each attribute and the decision, that is

$$I(X, Y).$$

### Usage

```
miScores(X, Y, threads = 0)
```

### Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A numerical vector with mutual information scores, with names copied from `X`.

## Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## Examples

```
miScores(iris[,-5],iris$Species)
```

---

# MRMR *Minimum redundancy maximal relevancy filter*

---

## Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = I(X;Y) - \frac{1}{|S|}\sum_{W\in S} I(X;W),$$

where $S$ is the set of already selected attributes.

## Usage

```
MRMR(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `k` | Number of attributes to select. Must not exceed `ncol(X)`. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: `selection`, a vector of indices of the selected features in the selection order, and `score`, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in `X`. Both vectors will either have a length `k` or zero, when all features turn out to have zero mutual information with the decision.

## Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## References

"Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy" H. Peng et al. IEEE Pattern Analysis and Machine Intelligence (PAMI) (2005)

## Examples

```
data(MadelonD)
MRMR(MadelonD$X,MadelonD$Y,20)
```

---

# NJMIM *Minimal normalised joint mutual information maximisation filter*

---

## Description

The method starts with an attribute of a maximal mutual information with the decision $Y$. Then, it greedily adds attribute $X$ with a maximal value of the following criterion:

$$J(X) = \min_{W \in S} \frac{I(X, W; Y)}{H(X, W, Y)},$$

where $S$ is the set of already selected attributes.

## Usage

```
NJMIM(X, Y, k = 3, threads = 0)
```

## Arguments

| | |
|---|---|
| X | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). NAs are not allowed. |
| Y | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. NAs are not allowed. |
| k | Number of attributes to select. Must not exceed ncol(X). |
| threads | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A list with two elements: selection, a vector of indices of the selected features in the selection order, and score, a vector of corresponding feature scores. Names of both vectors will correspond to the names of features in X. Both vectors will be at most of a length k, as the selection will stop as soon as all the remaining features will have a score of zero. This may happen during initial selection, in which case both vectors will be empty.

## Note

NJMIM is a normalised version of JMIM; JMI and DISR are modifications of JMIM and NJMIM in which a sum of joint information over already selected attributes is used instead of a minimum. It stops returning features when the best score reaches 0.

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in X and Y, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## References

"Feature selection using Joint Mutual Information Maximisation" M. Bennasar, Y. Hicks and R. Setchi, (2015)

## Examples

```
data(MadelonD)
NJMIM(MadelonD$X,MadelonD$Y,20)
```

`njmiScores` *Calculate normalised joint mutual information of all features*

## Description

Calculated normalised mutual information between each attribute joint with some other vector `Z` with the decision, that is

$$\frac{I(X, Z; Y)}{H(X, Y, Z)}.$$

This is the same as in the criterion used by `DISR` and `NJMIM`.

## Usage

```
njmiScores(X, Y, Z, threads = 0)
```

## Arguments

| | |
|---|---|
| `X` | Attribute table, given as a data frame with either factors (preferred), booleans, integers (treated as categorical) or reals (which undergo automatic categorisation; see below for details). `NA`s are not allowed. |
| `Y` | Decision attribute; should be given as a factor, but other options are accepted, exactly like for attributes. `NA`s are not allowed. |
| `Z` | Other vector; should be given as a factor, but other options are accepted, as for attributes. |
| `threads` | Number of threads to use; default value, 0, means all available to OpenMP. |

## Value

A numerical vector with the normalised joint mutual information scores, with names copied from `X`.

## Note

The method requires input to be discrete to use empirical estimators of distribution, and, consequently, information gain or entropy. To allow smoother user experience, praznik automatically coerces non-factor vectors in `X` and `Y`, which requires additional time and space and may yield confusing results – the best practice is to convert data to factors prior to feeding them in this function. Real attributes are cut into about 10 equally-spaced bins, following the heuristic often used in literature. Precise number of cuts depends on the number of objects; namely, it is $n/3$, but never less than 2 and never more than 10. Integers (which technically are also numeric) are treated as categorical variables (for compatibility with similar software), so in a very different way – one should be aware that an actually numeric attribute which happens to be an integer could be coerced into a $n$-level categorical, which would have a perfect mutual information score and would likely become a very disruptive false positive.

## Examples

```
njmiScores(iris[,-5],iris$Species,iris$Sepal.Length)
```

---

`setOmpThreads` *Control OpenMP thread count*

---

## Description

Does nothing, left to provide warnings to an old code.

## Usage

```
setOmpThreads(threads)
```

## Arguments

| | |
|---|---|
| `threads` | Ignored. |

## Value

Invisible `NULL`.

## Note

Since praznik 4.0, please use `threads` argument of particular algorithm function to control how many threads it will use.

# Index