Package 'HRTnomaly'

November 25, 2025

Type Package

Classification/MSC-2010 62G86

Title Historical, Relational, and Tail Anomaly-Detection Algorithms

Version 25.11.22

Date 2025-11-22

Maintainer Luca Sartore <drwolf85@gmail.com>

Description The presence of outliers in a dataset can substantially bias the results of statistical analyses. To correct for outliers, micro edits are manually performed on all records. A set of constraints and decision rules is typically used to aid the editing process. However, straightforward decision rules might overlook anomalies arising from disruption of linear relationships. Computationally efficient methods are provided to identify historical, tail, and relational anomalies at the data-entry level (Sartore et al., 2024; <doi:10.6339/24-JDS1136>). A score statistic is developed for each anomaly type, using a distribution-free approach motivated by the Bienaymé-Chebyshev's inequality, and fuzzy logic is used to detect cellwise outliers resulting from different types of anomalies. Each data entry is individually scored and individual scores are combined into a final score to determine anomalous entries. In contrast to fuzzy logic, Bayesian bootstrap and a Bayesian test based on empirical likelihoods are also provided as studied by Sartore et al. (2024; <doi:10.3390/stats7040073>). These algorithms allow for a more nuanced approach to outlier detection, as it can identify outliers at data-entry level which are not obviously distinct from the rest of the data.

This research was supported in part by the U.S. Department of Agriculture, National Agriculture Statistics Service. The findings and conclusions in this publication are those of the authors and should not be construed to represent any official USDA, or US Government determination or policy.

License AGPL-3

Depends R (>= 4.0.0)

Imports dplyr, purrr, tidyr

Suggests knitr, rmarkdown, cellWise

Encoding UTF-8

LazyLoad yes

NeedsCompilation yes

ByteCompile TRUE

Author Luca Sartore [aut] (ORCID = ``0000-0002-0446-1328''),
Luca Sartore [cre] (ORCID = ``0000-0002-0446-1328''),
Lu Chen [aut] (ORCID = ``0000-0003-3387-3484''),
Justin van Wart [aut],
Andrew Dau [aut] (ORCID = ``0009-0008-9482-5316''),
Valbona Bejleri [aut] (ORCID = ``0000-0001-9828-968X'')

Repository CRAN

Date/Publication 2025-11-25 11:32:22 UTC

Contents

`HRTnomaly-package` *HRTnomaly*

Description

Enhanced Anomaly Detection for Historical, Relational, and Tail Cellwise Outlier

Details

Package:	HRTnomaly
Type:	Package
License:	AGPL-3

The presence of outliers in a dataset can substantially bias the results of statistical analyses. To correct for outliers, micro edits are manually performed on all records. A set of constraints and decision rules is typically used to aid the editing process. However, straightforward decision rules might overlook anomalies arising from disruption of linear relationships. This package provides a computationally efficient method to identify historical, tail, and relational anomalies at the data-entry level. A score statistic is developed for each anomaly type, using a distribution-free approach motivated by the Bienaymé-Chebyshev's inequality, and fuzzy logic is used to detect cellwise outliers resulting from different types of anomalies. Each data entry is individually scored and individual scores are combined into a final score to determine anomalous entries. The HRTnomaly package has proven to be a powerful tool for identifying outliers that are not easily detectable using other traditional methods.

For a complete list of exported functions, use `library(help = "HRTnomaly")`.

This research was supported by the U.S. Department of Agriculture, National Agriculture Statistics Service. The findings and conclusions in this publication are those of the authors and should not be construed to represent any official USDA or U.S. Government determination or policy.

Author(s)

Luca Sartore, Lu Chen, Justin van Wart, Andrew Dau, and Valbona Bejleri

Maintainer: Luca Sartore <drwolf85@gmail.com>

References

Agostinelli C, Leung A, Yohai VJ, Zamar RH (2015). Robust estimation of multivariate location and scatter in the presence of cellwise and casewise contamination. *Test*, **24**(3): 441–461.

Alqallaf F, Van Aelst S, Yohai VJ, Zamar RH (2009). Propagation of outliers in multivariate data. *The Annals of Statistics*, 311–331.

Bienaymé IJ (1867). Considérations à l'appui de la découverte de Laplace sur la loi de probabilité dans la méthode des moindres carrés. *Journal de Mathématiques Pures et Appliquées*, **2**(12): 158–176.

Chepulis MA, Shevlyakov G (2020). On outlier detection with the Chebyshev type inequalities. *Journal of the Belarusian State University. Mathematics and Informatics*, **3**: 28–35.

Filzmoser P, Gregorich M (2020). Multivariate outlier detection in applied data analysis: Global, local, compositional and cellwise outliers. *Mathematical Geosciences*, **52**(8): 1049–1066.

Gupta MM, Qi J (1991). Theory of t-norms and fuzzy inference methods. *Fuzzy Sets and Systems*, **40**(3): 431–450.

Huber PJ, Ronchetti EM (1981). *Robust statistics*. John Wiley & Sons, New York.

O'Gorman TJ (1994). The effect of cosmic rays on the soft error rate of a DRAM at ground level. *IEEE Transactions on Electron Devices*, **41**(4): 553–557.

Rousseeuw PJ, Van den Bossche W (2018). Detecting deviating data cells. *Technometrics*, **60**(2): 135–145.

Sandqvist AP (2016). Identifizierung von Ausreissern in eindimensionalen gewichteten Umfragedaten. *KOF Analysen*, **2016**(2): 45–56.

Sartore L, Chen L, Bejleri V (2024). Empirical Inferences Under Bayesian Framework to Identify Cellwise Outliers. *Stats*, **7**: 1244–1258.

Sartore L, Chen L, van Wart J, Dau A, Bejleri V (2024). Identifying Anomalous Data Entries in Repeated Surveys. *Journal of Data Science*, **22**(3): 436–455.

Tchebichef P (1867). Des valeurs moyennes. *Journal de Mathématiques Pures et Appliquées*, **2**(12): 177–184.

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers using robust regression
res_c <- cellwise(toy[sample.int(33), ], epochs = 10L)
# Detect cellwise outliers using Bayesian testing
res_g <- bayesHRT(toy[sample.int(33), ], prior = 0.5)

# Detect record level outliers using Deep Isolation Forests
res_t <- dif(iris)
```

`bayesHRT` *Calculate Cellwise Flags for Anomaly Detection Using Bayesian Testing*

Description

The function uses the predictive posterior distribution based on empirical likelihoods to determine if a data entry is an outlier or not. The function takes a long-format `data.frame` object as input and returns it with two appended vectors. The first vector contains the posterior probabilities as numbers between zero and one, and the second vector provides a set of logical values indicating whether the data entry is an outlier (`TRUE`) or not (`FALSE`).

Usage

```
bayesHRT(a, prior = NULL)
```

Arguments

`a`	A long-format `data.frame` object with survey data. For details see information on the data format.

`prior` A numerical value or vector of cell-level prior probabilities of observing an outlier. It is `NULL` by default. If false, the function searches for a column named `"prior"` within the dataset. If such column is not provided in the dataset, a `0.5` non-informative value is used for all cells.

Details

The argument `a` is provided as an object of class `data.frame`. This object is considered as a long-format `data.frame`, and it must have at least five columns with the following names:

`"strata"` a `character` or `factor` column containing the information on the stratification.

`"unit_id"` a `character` or `factor` column containing the ID of the statistical unit in the survey sample(x, size, replace = FALSE, prob = NULL).

`"master_varname"` a `character` column containing the name of the observed variable.

`"current_value_num"` a `numeric` the observed value, i.e., a data entry

`"pred_value"` a `numeric` a value observed on a previous survey for the same variable if available. If not available, the value can be set to `NA` or `NaN`. When working with longitudinal data, the value can be set to a time-series forecast or a filtered value.

`"prior"` a `numeric` a value of prior probabilities of observing an outlier for the cell. If this column is omitted in the dataset provided, the function will use the values provided through the argument `prior`.

The `data.frame` object in input can have more columns, but the extra columns would be ignored in the analyses. However, these extra columns would be preserved in the system memory and returned along with the results from the cellwise outlier-detection analysis. The use of the R-packages `dplyr`, `purrr`, and `tidyr` is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

A data frame with the same columns as the input data frame, plus the following additional columns:

post_prob The posterior probability of the value being an outlier.

outlier A boolean indicating whether the value is an outlier.

anomaly_flag A character string indicating the type of anomaly detected, if any.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers
res <- bayesHRT(toy[sample.int(100), ], prior = 0.5)
```

bayeswise *Calculate Cellwise Flags for Anomaly Detection Using Robust Bayesian Methods*

Description

The function uses a Bayesian approach to determine if a data entry is an outlier or not. The function takes a long-format `data.frame` object as input and returns it with two appended vectors. The first vector contains the posterior probabilities for a cell to be anomalous, and the second vector provides a set of logical values indicating whether the data entry is an outlier (`TRUE`) or not (`FALSE`).

Usage

```
bayeswise(a, prior = NULL, epochs = 1000L)
```

Arguments

`a`	A long-format `data.frame` object with survey data. For details see information on the data format.
`prior`	A numerical value or vector of cell-level prior probabilities of observing an outlier. It is `NULL` by default. If false, the function searches for a column named `"prior"` within the dataset. If such column is not provided in the dataset, a `0.5` non-informative value is used for all cells.
`epochs`	Number of epochs used to train a nontrivial robust linear model via the lion algorithm. By default, the algorithm will run 1000 iterations.

Details

The argument `a` is provided as an object of class `data.frame`. This object is considered as a long-format `data.frame`, and it must have at least five columns with the following names:

`"strata"` a `character` or `factor` column containing the information on the stratification.

`"unit_id"` a `character` or `factor` column containing the ID of the statistical unit in the survey sample(x, size, replace = FALSE, prob = NULL).

`"master_varname"` a `character` column containing the name of the observed variable.

`"current_value_num"` a `numeric` the observed value, i.e., a data entry

`"pred_value"` a `numeric` a value observed on a previous survey for the same variable if available. If not available, the value can be set to `NA` or `NaN`. When working with longitudinal data, the value can be set to a time-series forecast or a filtered value.

`"prior"` a `numeric` a value of prior probabilities of observing an outlier for the cell. If this column is omitted in the dataset provided, the function will use the values provided through the argument `prior`.

The `data.frame` object in input can have more columns, but the extra columns would be ignored in the analyses. However, these extra columns would be preserved in the system memory and returned along with the results from the cellwise outlier-detection analysis. The use of the R-packages `dplyr`, `purrr`, and `tidyr` is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

A data frame with the same columns as the input data frame, plus the following additional columns:

prior The prior probability used for the cell (either input or derived).

zScore The z-score of the cell.

hScore The h-score of the cell.

rScore The r-score of the cell.

tScore The t-score of the cell.

score The final outlier score of the cell, representing the posterior probability of being anomalous.

outlier A boolean indicating whether the cell is an outlier.

anomaly_flag A character string indicating the type of anomaly detected, if any.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers using Bayesian Analysis
res <- bayeswise(toy[sample.int(100), ], 0.5, 10L)
```

`bootHRT` *Calculate Cellwise Flags for Anomaly Detection Using Bayesian Bootstrap*

Description

The function uses Bayesian bootstrap to determine if a data entry is an outlier or not. The function takes a long-format `data.frame` object as input and returns it with appended vectors. The output includes flags for different quantiles and means of the contamination threshold distribution.

Usage

```
bootHRT(a, contamination = 0.08, boot_max_it = 1000L)
```

Arguments

a	A long-format data.frame object with survey data. For details see information on the data format.
contamination	A number between zero and one used as a threshold when identifying outliers from the fuzzy scores. By default, the algorithm will identify approximately 8% of the data entries as anomalies.
boot_max_it	An integer number determining the iterations performed by Bayesian bootstrap algorithm. It is set to 1000 by default.

Details

The argument a is provided as an object of class data.frame. This object is considered as a long-format data.frame, and it must have at least five columns with the following names:

"strata" a character or factor column containing the information on the stratification.

"unit_id" a character or factor column containing the ID of the statistical unit in the survey sample(x, size, replace = FALSE, prob = NULL).

"master_varname" a character column containing the name of the observed variable.

"current_value_num" a numeric the observed value, i.e., a data entry

"pred_value" a numeric a value observed on a previous survey for the same variable if available. If not available, the value can be set to NA or NaN. When working with longitudinal data, the value can be set to a time-series forecast or a filtered value.

The data.frame object in input can have more columns, but the extra columns would be ignored in the analyses. However, these extra columns would be preserved in the system memory and returned along with the results from the cellwise outlier-detection analysis. The use of the R-packages dplyr, purrr, and tidyr is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

A data frame with the same columns as the input data frame, plus the following additional columns:

score The raw anomaly score for each cell.

outlier_1qt A boolean indicating if the cell is an outlier based on the first quantile threshold.

outlier_2qt A boolean indicating if the cell is an outlier based on the second quantile (median) threshold.

outlier_3qt A boolean indicating if the cell is an outlier based on the third quantile threshold.

outlier_1mn A boolean indicating if the cell is an outlier based on the mean threshold minus one standard deviation.

outlier_2mn A boolean indicating if the cell is an outlier based on the mean threshold.

outlier_3mn A boolean indicating if the cell is an outlier based on the mean threshold plus one standard deviation.

anomaly_flag A character string indicating the type of anomaly detected, if any (e.g., "h", "t", "r").

The returned object also includes an attribute "thresholds" which is a numeric vector of length boot_max_it containing samples from the posterior distribution of the contamination threshold.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers
res <- bootHRT(toy, boot_max_it = 10)
```

---

cellwise *Calculate Cellwise Flags for Anomaly Detection*

Description

The function uses fuzzy logic to determine if a data entry is an outlier or not. The function takes a long-format `data.frame` object as input and returns it with two appended vectors. The first vector contains the anomaly scores as numbers between zero and one, and the second vector provides a set of logical values indicating whether the data entry is an outlier (`TRUE`) or not (`FALSE`).

Usage

```
cellwise(a, contamination = 0.08, epochs = 1000L)
```

Arguments

`a`	A long-format `data.frame` object with survey data. For details see information on the data format.
`contamination`	A number between zero and one used as a threshold when identifying outliers from the fuzzy scores. By default, the algorithm will identify 8% of the records as anomalies.
`epochs`	Number of epochs used to train a nontrivial robust linear model via the lion algorithm. By default, the algorithm will run 1000 iterations.

Details

The argument `a` is proivded as an object of class `data.frame`. This object is considered as a long-format `data.frame`, and it must have at least five columns with the following names:

`"strata"` a `character` or `factor` column containing the information on the stratification.

`"unit_id"` a `character` or `factor` column containing the ID of the statistical unit in the survey sample(x, size, replace = FALSE, prob = NULL).

`"master_varname"` a `character` column containing the name of the observed variable.

`"current_value_num"` a `numeric` the observed value, i.e., a data entrie

`"pred_value"` a `numeric` a value observed on a previous survey for the same variable if available. If not available, the value can be set to `NA` or `NaN`. When working with longitudinal data, the value can be set to a time-series forecast or a filtered value.

The `data.frame` object in input can have more columns, but the extra columns would be ignored in the analyses. However, these extra columns would be preserved in the system memory and returned along with the results from the cellwise outlier-detection analysis.

The use of the R-packages `dplyr`, `purrr`, and `tidyr` is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

A data frame with the same columns as the input data frame, plus the following additional columns:

zScore The z-score of the cell.

hScore The h-score of the cell.

rScore The r-score of the cell.

tScore The t-score of the cell.

score The final outlier score of the cell.

outlier A boolean indicating whether the cell is an outlier.

anomaly_flag A character string indicating the type of anomaly detected, if any.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers
res <- cellwise(toy[sample.int(100), ], 0.05, 10L)
```

---

class_check *Evaluate the Accuracy of Outlier Classification Results*

Description

The function computes the confusion matrix between the logical output of an outlier detection algorithm and a reference (ground-truth) logical vector. The function also calculates the overall accuracy of the results from the confusion matrix, including recall, precision, and F1-scores for the two classes (regular, versus outlier).

Usage

```
class_check(pred, truth)
```

Arguments

pred	A logical vector with the classification output from an anomaly detection algorithm.
truth	A logical vector with the observed classification as a reference (or ground truth).

Details

The function computes the confusion matrix using the function `table`. True positive and false negative are successively evaluated to compute overall accuracy, recall, precision, and F1-scores.

Value

An S3 class named `checkwise` with the confusion matrix, and other accuracy metrices appended as attribues.

`attr(, "overall")` A numeric value between zero and one with the overall accuracy.

`attr(, "recall")` A numeric vector of values between zero and one with the recall index for regular and outlier cells.

`attr(, "precision")` A numeric vector of values between zero and one with the precision index for regular and outlier cells.

`attr(, "f1-score")` A numeric vector of values between zero and one with the F1-scores for regular and outlier cells.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers using Cellwise Analysis
res <- cellwise(toy[sample.int(100), ], 0.05, 2L)
class_check(res$outlier, res$anomaly_flag != "")

## End(Not run)
```

dif *Deep Isolation Forest*

Description

The function builds a deep isolation forest that uses fuzzy logic to determine if a record is anomalous or not. The function takes a wide-format `data.frame` object as input and returns it with two appended vectors. The first vector contains the anomaly scores as numbers between zero and one, and the second vector provides a set of logical values indicating whether the records are outliers (`TRUE`) or not (`FALSE`).

Usage

```
dif(dta, nt = 100L, nss = NULL, threshold = 0.95)
```

Arguments

`dta`	A wide-format `data.frame` object with records (stored by row).
`nt`	Number of deep isolation trees to build to form the forest. By default, it is set to `100`.
`nss`	Number of subsamples used to build a single deep isolation tree. If set (by default) to `NULL`, the program will randomly select 25% of the records provided to the `dta` argument.
`threshold`	A number between zero and one used as a threshold when identifying outliers from the anomaly scores. By default, this argument is set to `0.95`, so that 5% of the records is going to be classified as anomalous.

Details

The argument `dta` is provided as an object of class `data.frame`. This object is considered as a wide-format `data.frame`. The use of the R-packages `dplyr`, `purrr`, and `tidyr` is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

The wide-format `data.frame` is provided as input data and contains extra columns:

scores A numeric vector of anomaly scores ranging from 0 to 1, where values closer to 1 indicate higher anomaly.

flags A logical vector indicating whether each record is flagged as an outlier (`TRUE`) or not (`FALSE`) based on the specified `threshold`.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Detect outliers in the `iris` dataset
res <- dif(iris)

## End(Not run)
```

`fuzzyHRT` *Calculate Cellwise Flags for Anomaly Detection*

Description

The function uses fuzzy logic to determine if a data entry is an outlier or not. The function takes a long-format `data.frame` object as input and returns it with two appended vectors. The first vector contains the anomaly scores as numbers between zero and one, and the second vector provides a set of logical values indicating whether the data entry is an outlier (`TRUE`) or not (`FALSE`).

Usage

```
fuzzyHRT(a, contamination = 0.08)
```

Arguments

`a`	A long-format `data.frame` object with survey data. For details see information on the data format.
`contamination`	A number between zero and one used as a threshold when identifying outliers from the fuzzy scores. By default, the algorithm will identify 8% of the records as anomalies.

Details

The argument `a` is proivded as an object of class `data.frame`. This object is considered as a long-format `data.frame`, and it must have at least five columns with the following names:

`"strata"` a `character` or `factor` column containing the information on the stratification.

`"unit_id"` a `character` or `factor` column containing the ID of the statistical unit in the survey sample(x, size, replace = FALSE, prob = NULL).

`"master_varname"` a `character` column containing the name of the observed variable.

`"current_value_num"` a `numeric` the observed value, i.e., a data entrie

`"pred_value"` a `numeric` a value observed on a previous survey for the same variable if available. If not available, the value can be set to `NA` or `NaN`. When working with longitudinal data, the value can be set to a time-series forecast or a filtered value.

The data.frame object in input can have more columns, but the extra columns would be ignored in the analyses. However, these extra columns would be preserved in the system memory and returned along with the results from the cellwise outlier-detection analysis. The use of the R-packages dplyr, purrr, and tidyr is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

The long-format data.frame is provided as input data and contains extra columns i.e., anomaly flags and outlier indicators columns.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers
res <- fuzzyHRT(toy[sample.int(100), ])

## End(Not run)
```

---

gif *Generalized Isolation Forest*

Description

The function builds a generalized isolation forest that uses fuzzy logic to determine if a record is anomalous or not. The function takes a wide-format data.frame object as input and returns it with two appended vectors. The first vector contains the anomaly scores as numbers between zero and one, and the second vector provides a set of logical values indicating whether the records are outliers (TRUE) or not (FALSE).

Usage

```
gif(dta, nt = 100L, nss = NULL, threshold = 0.95)
```

Arguments

dta	A wide-format data.frame object with records (stored by row).
nt	Number of generalized isolation trees to build to form the forest. By default, it is set to 100.

nss	Number of subsamples used to build a single generalized isolation tree. If set (by default) to NULL, the program will randomly select 25% of the records provided to the dta argument.
threshold	A number between zero and one used as a threshold when identifying outliers from the anomaly scores. By default, this argument is set to 0.95, so that 5% of the records is going to be classified as anomalous.

Details

The argument dta is provided as an object of class data.frame. This object is considered as a wide-format data.frame. The use of the R-packages dplyr, purrr, and tidyr is highly recommended to simplify the conversion of datasets between long and wide formats.

Value

The wide-format data.frame is provided as input data and contains extra columns:

scores A numeric vector of anomaly scores ranging from 0 to 1, where values closer to 1 indicate higher anomaly.

flags A logical vector indicating whether each record is flagged as an outlier (TRUE) or not (FALSE) based on the specified threshold.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Detect outliers in the `iris` dataset
res <- gif(iris)

## End(Not run)
```

---

pif *Proximity Isolation Forest*

Description

The function builds a proximity isolation forest that uses fuzzy logic to determine if a record is anomalous or not. The function takes a list object as input and returns it with two vectors appended as attributes. The first vector contains the anomaly scores as numbers between zero and one, and the second vector provides a set of logical values indicating whether the records are outliers (TRUE) or not (FALSE).

Usage

```
pif(dta, nt = 100L, nss = NULL, max_depth = 12L, threshold = 0.95,
    proximity_type = c("single", "paired", "pivotal"), dist_fun = NULL)
```

Arguments

dta	A list object with records (stored as individual entries on the list).
nt	Number of deep isolation trees to build to form the forest. By default, it is set to 100.
nss	Number of subsamples used to build a proximity isolation tree in the forest. If set (by default) to NULL, the program will randomly select 25% of the records provided to the dta argument.
max_depth	An integer number corresponding to the maximum depth achieved by a proximity isolation tree in the forest. By default, this argument is set to 12.
threshold	A number between zero and one used as a threshold when identifying outliers from the anomaly scores. By default, this argument is set to 0.95, so that 5% of the records is going to be classified as anomalous.
proximity_type	A character string denoting the number the number of proximity prototypes used by the algorithms (see details for more information). By default, a "single" prototype is randomly chosen to split a branch in the isolation tree.
dist_fun	A function computing the distance between any pair of components in dta. If set (by default) to NULL, the program will select an Euclidean distance for two numerical arrays.

Details

The argument dta is provided as an object of class list. This object is considered as a list of arbitrary R objects that will be analyzed by one of the three algorithms provided with the pif function.

Three algorithms are implemented. The user can choose the proximity type by providing the number of prototypes used to build the isolation trees in the forest. A "single" prototype uses the distance between an input data point to a single randomly selected prototype at each branching node of the tree. Two prototypes (denoted as "paired") are randomly chosen and successively considered as gravitational point of their respective basins of attraction for partitioning the data. An additional "pivotal" point is randomly selected to enhance the algorithm based on two prototypes. In this case, the two distances between a data point and the two prototypes are normalized through the Steinhaus transformation and the pivotal prototype.

Value

The original input list dta with the following attributes appended:

scores A numeric vector of anomaly scores, ranging from 0 to 1, where higher values indicate a higher likelihood of being an outlier.

flag A logical vector indicating whether each element in the input list is flagged as an outlier (TRUE) or not (FALSE) based on the specified threshold.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Personalized distance
my_dst <- function(x, y) {
  xn <- as.numeric(x[[1]][1:4])
  yn <- as.numeric(y[[1]][1:4])
  num <- mean((xn - yn)^2)
  den <- median((xn - yn)^2)
  return(num / (1 + den))
}
# Converting the dataset iris to a list
ir <- apply(iris, 1, list)
# Detect outliers in the `iris` dataset
res_sng <- pif(ir, 5L, 18L, 5L, .85, "single", my_dst)
res_prd <- pif(ir, 5L, 18L, 5L, .85, "paired", my_dst)
res_prx <- pif(ir, 5L, 18L, 5L, .85, "pivotal", my_dst)
# count identified anomalies
print(sum(attr(res_prd, "flag")))

## End(Not run)
```

---

print.checkwise *A Method to Print the Accuracy of Outlier Classification Results*

Description

The function prints the confusion matrix and accuracy results previously computed with the function class_check.

Usage

```
## S3 method for class 'checkwise'
print(x, confusion = FALSE, ...)
```

Arguments

x	An S3 object of the class checkwise, typically computed with the function class_check.
confusion	A logical value, which is FALSE by default. If TRUE, the confusion matrix is printed after showing all accuracy metrics.
...	Additional arguments to pass to the function cat.

Details

The function computes the confusion matrix using the function `table`. True positive and false negative are successively evaluated to compute overall accuracy, recall, precision, and F1-scores.

Value

An S3 class named `checkwise` with the confusion matrix, and other accuracy metrics appended as attribues.

`attr(, "overall")` A numeric value between zero and one with the overall accuracy.

`attr(, "recall")` A numeric vector of values between zero and one with the recall index for regular and outlier cells.

`attr(, "precision")` A numeric vector of values between zero and one with the precision index for regular and outlier cells.

`attr(, "f1-score")` A numeric vector of values between zero and one with the F1-scores for regular and outlier cells.

Author(s)

Luca Sartore <drwolf85@gmail.com>

Examples

```
## Not run:
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the 'toy' data
data(toy)
# Detect cellwise outliers using Bayesian Analysis
res <- cellwise(toy[sample.int(100), ], 0.5, 10L)
class_check(res$outlier, res$anomaly_flag != "")

## End(Not run)
```

`setCores` *Set the number of CPU cores for HPC*

Description

The function set the number of CPU cores for parallel computation by the use of OpenMP library (https://www.openmp.org/). If the package was not complied with the library OpenMP (>= 3.0), this function is disabled.

Usage

```
setCores(n)
```

Arguments

n	an integer value denoting the number of CPU cores to use; if it exceeds the total number of cores, all of them will be used. If missing, the number of CPU cores in use will be displayed.

Details

When the package is loaded, only one CPU core is used.

Value

The total number of CPU cores in use will be returned and a message will be displayed. If the package was not complied with the library OpenMP (>= 3.0), the value one will be returned.

Author(s)

Luca Sartore <drwolf85@gmail.com>

References

SunTM ONE Studio 8 (2003) *OpenMP API User's Guide*. Sun Microsystems Inc., Santa Clara, U.S.A.

Examples

```
#Display the number of CPU cores in use
setCores()

#Set 2 CPU cores for parallel computation
setCores(2)

#Set 1 CPU core for serial computation
setCores(1)
```

---

toy *Toy dataset*

Description

Synthetically produced data are provided to test the FuzzyHRT algorithm.

Usage

```
data(toy)
```

Format

The `toy` data is a `data.frame` object consisting of 9265 synthetic records and 5 variables:

`strata` a numeric value of the strata

`unit_id` a character string with the record ID

`master_varname` a character string with the names of the variables

`current_value_num` a numeric value with the current synthetic observation

`pred_value` a numeric value with the previously synthetic observation

Examples

```
# Load the package
library(HRTnomaly)
set.seed(2025L)
# Load the toy dataset
data(toy)
# Detect cellwise outliers
res <- fuzzyHRT(toy)
```

Index