Package: stray (via r-universe)

November 14, 2024

Type Package

Title Anomaly Detection in High Dimensional and Temporal Data

Version 0.1.1

Depends R (>= 3.4.0)

Imports FNN, ggplot2, colorspace, pcaPP, stats, ks

Description This is a modification of 'HDoutliers' package. The 'HDoutliers' algorithm is a powerful unsupervised algorithm for detecting anomalies in high-dimensional data, with a strong theoretical foundation. However, it suffers from some limitations that significantly hinder its performance level, under certain circumstances. This package implements the algorithm proposed in Talagala, Hyndman and Smith-Miles (2019) <arXiv:1908.04000> for detecting anomalies in high-dimensional data that addresses these limitations of 'HDoutliers' algorithm. We define an anomaly as an observation that deviates markedly from the majority with a large distance gap. An approach based on extreme value theory is used for the anomalous threshold calculation.

BugReports https://github.com/pridiltal/stray/issues

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Repository https://robjhyndman.r-universe.dev

RemoteUrl https://github.com/pridiltal/stray

RemoteRef HEAD

RemoteSha 519b2e05b76ef9644131386f039767acf52b8124

Contents

`data_a` *A dataset with an outlier*

Description

A bivariate dataset with an outlier

Usage

```
data_a
```

Format

A data frame with 1001 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

`data_b` *A bimodal dataset with a micro cluster*

Description

A bivariate dataset with two typical classes and a micor cluster

Usage

```
data_b
```

Format

A data frame with 2003 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

---

data_c *A dataset with local anomalies and micro clusters*

Description

A bivariate dataset with local anomalies and two micro clusters

Usage

```
data_c
```

Format

A data frame with 1009 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

---

data_d *A wheel dataset with two inliers*

Description

A bivariate dataset with two inliers. The inliers are very close to one another

Usage

```
data_d
```

Format

A data frame with 1002 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

data_e *A bimodal dataset with an inlier*

Description

A bimodal dataset with an inlier. one typical class is a very dense cluster,

Usage

```
data_e
```

Format

A data frame with 2001 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

data_f *A dataset with an outlier*

Description

A dataset with an outlier. The typical class is a very dense cluster.

Usage

```
data_f
```

Format

A data frame with 2001 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

display_HDoutliers *Display outliers with a scatterplot*

Description

Provide a 2D scatterplot of data for visual exploration. For data with more than two dimensions, two dimensional scatterplot is produced using the first two pricipal components.

Usage

```
display_HDoutliers(data, out)
```

Arguments

data	A vector, matrix, or data frame consisting of numerical variables.
out	A list containing output values produced by find_HDoutliers

Value

A ggplot object of data space with detected outliers (if any).

Examples

```
data <- c(rnorm(100), 7, 7.5, rnorm(100, 20), 45)
output <- find_HDoutliers(data, knnsearchtype = "kd_tree")
display_HDoutliers(data, out = output)


data <- rbind(matrix(rnorm(96), ncol = 2), c(10,12),c(3,7))
output <- find_HDoutliers(data, knnsearchtype = "brute")
display_HDoutliers(data, out = output)


data <- rbind(matrix(rnorm(144), ncol = 3), c(10,12,10),c(3,7,10))
output <- find_HDoutliers(data, knnsearchtype = "brute")
display_HDoutliers(data, out = output)
```

find_HDoutliers *Detect Anomalies in High Dimensional Data.*

Description

Detect anomalies in high dimensional data. This is a modification of HDoutliers.

Usage

```
find_HDoutliers(
  data,
  alpha = 0.01,
  k = 10,
  knnsearchtype = "brute",
  normalize = "unitize",
  p = 0.5,
  tn = 50
)
```

Arguments

data	A vector, matrix, or data frame consisting of numerical variables.
alpha	Threshold for determining the cutoff for outliers. Observations are considered outliers if they fall in the $(1 - alpha)$ tail of the distribution of the nearest-neighbor distances between exemplars.
k	Number of neighbours considered.
knnsearchtype	A character vector indicating the search type for k- nearest-neighbors.
normalize	Method to normalize the columns of the data. This prevents variables with large variances having disproportional influence on Euclidean distances. Two options are available "standardize" or "unitize". Default is set to "unitize"
p	Proportion of possible candidates for outliers. This defines the starting point for the bottom up searching algorithm. Default is set to 0.5.
tn	Sample size to calculate an emperical threshold. Default is set to 50.

Value

The indexes of the observations determined to be outliers.

References

Wilkinson, L. (2018), 'Visualizing big data outliers through distributed aggregation', IEEE transactions on visualization and computer graphics 24(1), 256-266.

Examples

```
require(ggplot2)
set.seed(1234)
data <- c(rnorm(1000, mean = -6), 0, rnorm(1000, mean = 6))
outliers <- find_HDoutliers(data, knnsearchtype = "kd_tree")



set.seed(1234)
n <- 1000 # number of observations
nout <- 10 # number of outliers
typical_data <- matrix(rnorm(2 * n), ncol = 2, byrow = TRUE)
```

```
out <- matrix(5 * runif(2 * nout, min = -5, max = 5), ncol = 2, byrow = TRUE)
data <- rbind(out, typical_data)
outliers <- find_HDoutliers(data, knnsearchtype = "brute")
```

`find_threshold` *Find Outlier Threshold*

Description

Find Outlier Threshold

Usage

```
find_threshold(
  outlier_score,
  alpha = 0.01,
  outtail = c("max", "min"),
  p = 0.5,
  tn = 50
)
```

Arguments

`outlier_score`	A vector of outlier scores. Can be a named vector or a vector with no names.
`alpha`	Threshold for determining the cutoff for outliers. Observations are considered outliers if they fall in the $(1 - alpha)$ tail of the distribution of the nearest-neighbor distances between exemplars.
`outtail`	Direction of the outlier tail.
`p`	Proportion of possible candidates for outliers. This defines the starting point for the bottom up searching algorithm.
`tn`	Sample size to calculate an empirical threshold

Value

The indexes (or names, if the input is named vactor) of the observations determined to be outliers.

ped_data *Dataset with pedestrian counts*

Description

A dataset with hourly pedestrian counts at 43 locations in the city Melbourne, australia, from 1 December, 2018 to 1, January, 2019.

Usage

```
ped_data
```

Format

A data frame with 33024 rows and 5 variables:

Sensor Sensor location

Date_Time Time and date

Date Date

Time Time

Count Pedestrian count

stray *stray: A package for robust anomaly detection in data streams with concept drift*

Description

This package is a modification of HDoutliers package. HDoutliers is a powerful algorithm for the detection of anomalous observations in a dataset, which has (among other advantages) the ability to detect clusters of outliers in multi-dimensional data without requiring a model of the typical behavior of the system. However, it suffers from some limitations that affect its accuracy. In this package, we propose solutions to the limitations of HDoutliers, and propose an extension of the algorithm to deal with data streams that exhibit non-stationary behavior. The results show that our proposed algorithm improves the accuracy, and enables the trade-off between false positives and negatives to be better balanced.

Note

The name stray comes from Search and TRace AnomalY

References

Talagala, P. D., Hyndman, R. J., & Smith-Miles, K. (2019). Anomaly Detection in High Dimensional Data. https://www.monash.edu/business/ebs/research/publications/ebs/wp20-2019.pdf

Wilkinson, L. (2017). Visualizing big data outliers through distributed aggregation. IEEE transactions on visualization and computer graphics, 24(1), 256-266. https://www.cs.uic.edu/~wilkinson/Publications/outliers.pdf

See Also

The core functions in this package: find_HDoutliers, display_HDoutliers

Full documentation and demos:

---

use_KNN *Find outliers using kNN distance with maximum gap*

Description

Find outliers using kNN distance with maximum gap

Usage

```
use_KNN(data, alpha, k, knnsearchtype, p, tn)
```

Arguments

data	A vector, matrix, or data frame consisting of numeric and/or categorical variables.
alpha	Threshold for determining the cutoff for outliers. Observations are considered outliers outliers if they fall in the $(1 - alpha)$ tail of the distribution of the nearest-neighbor distances between exemplars.
k	Number of neighbours considered.
knnsearchtype	A character vector indicating the search type for k- nearest-neighbors.
p	Proportion of possible candidates for outliers. This defines the starting point for the bottom up searching algorithm.
tn	Sample size to calculate an emperical threshold. Default is set to 50.

Value

The indexes of the observations determined to be outliers and the outlying scores.

`wheel1` *wheel data set with inlier and outlier.*

Description

A bivariate dataset with an inlier and anoutlier

Usage

```
wheel1
```

Format

A data frame with 1002 rows and 3 variables:

x numerical variable

y numerical variable

type Type of a data point : Typical or Outlier

Index