Package ‘HighDimOut’

April 2, 2015

Type Package

Title Outlier Detection Algorithms for High-Dimensional Data

Version 1.0.0

Date 2015-03-30

Author Cheng Fan <raja8885@hotmail.com>

Maintainer Cheng Fan <raja8885@hotmail.com>

Depends R (>= 3.0.1)

Imports foreach, DMwR, plyr, proxy, FNN, ggplot2

Suggests knitr

VignetteBuilder knitr

LazyData true

Description Three high-dimensional outlier detection algorithms and a outlier unification scheme are implemented in this package. The angle-based outlier detection (ABOD) algorithm is based on the work of Kriegel, Schubert, and Zimek [2008]. The subspace outlier detection (SOD) algorithm is based on the work of Kriegel, Kroger, Schubert, and Zimek [2009]. The feature bagging-based outlier detection (FBOD) algorithm is based on the work of Lazarevic and Kumar [2005]. The outlier unification scheme is based on the work of Kriegel, Kroger, Schubert, and Zimek [2011].

License GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2015-04-02 08:02:28

R topics documented:

HighDimOut-package *Outlier Detection Algorithms for High-Dimensional Data*

Description

Three high-dimensional outlier detection algorithms and a outlier unification scheme are implemented in this package. The angle-based outlier detection (ABOD) algorithm is based on the work of Kriegel, Schubert, and Zimek [2008]. The subspace outlier detection (SOD) algorithm is based on the work of Kriegel, Kroger, Schubert, and Zimek [2009]. The feature bagging-based outlier detection (FBOD) algorithm is based on the work of Lazarevic and Kumar [2005]. The outlier unification scheme is based on the work of Kriegel, Kroger, Schubert, and Zimek [2011].

Details

Package:	HighDimOut
Type:	Package
Version:	1.0
Date:	2015-03-30
License:	MIT

Author(s)

Cheng Fan

Maintainer: Cheng Fan <raja8885@hotmail.com>

References

Hans-Peter Kriegel, Matthias Schubert, Arthur Zimek. Angle-based outlier detection in high-dimensional data. KDD 2008, 444-452.

Hans-Peter Kriegel, Peer Kroger, Erich Schubert, Arthur Zimek. Interpreting and Unifying Outlier Scores. SDM 2011, 13-24.

Hans-Peter Kriegel, Peer Kroger, Erich Schubert, Arthur Zimek. Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data. PAKDD 2009, 831-838.

Aleksandar Lazarevic, Vipin Kumar. Feature bagging for outlier detection. KDD 2005, 157-166.

Func.ABOD *Angle-based outlier detection (ABOD) algorithm*

Description

This function performs the basic and aprroximated version of angle-based outlier detection algorithm. The ABOD method is especially useful for high-dimensional data, as angle is a more robust measure than distance in high-dimensional space. The basic version calculate the angle variance based on the whole data. The results obtained are more reliable. However, the speed can be very slow. The approximated version calculate the angle variance based on a subset of data and thereby, increasing the calculation speed. This function is based on the work of Krigel, H.P., Schubert, M., Zimek, A., Angle-based outlier detection in high dimensional data, 2008.

Usage

```
Func.ABOD(data, basic = FALSE, perc)
```

Arguments

`data`	is the data frame containing the observations. Each row represents an observation and each variable is stored in one column.
`basic`	is a logical value, indicating whether the basic method is used. The speed of basic version can be very slow if the data size is large.
`perc`	defines the percentage of data to use when calculating the angle variance. It is only needed when basic=F.

Value

The function returns the vector containing the angle variance for each observation

Examples

```
library(ggplot2)
res.ABOD <- Func.ABOD(data=TestData[,1:2], basic=FALSE, perc=0.2)
data.temp <- TestData[,1:2]
data.temp$Ind <- NA
data.temp[order(res.ABOD, decreasing = FALSE)[1:10],"Ind"] <- "Outlier"
data.temp[is.na(data.temp$Ind),"Ind"] <- "Inlier"
data.temp$Ind <- factor(data.temp$Ind)
ggplot(data = data.temp) + geom_point(aes(x = x, y = y, color=Ind, shape=Ind))
```

Func.FBOD *Feature-bagging outlier detection (FBOD) algorithm*

Description

This function performs feature-bagging based outlier detection algorithm The implemented method is based on the work of "Lazarevic, A., Kumar, V., Feature bagging for outlier detection, 2005" This method can be regarded as an ensemble method, which based on the results of local outlier factor (LOF). During each iteration, a random subset of variables, whose size is randomly chosen between d/2 to d (where d is the dimensionality of the input data), is selected. The LOF method is applied to calculate the LOF scores based on the selected data subset. The final score of FBOD is the cumulative sum of each iteration.

Usage

```
Func.FBOD(data, iter, k.nn)
```

Arguments

`data`	is the data frame containing the observations. Each row represents an observation and each variable is stored in one column.
`iter`	is the iteration used.
`k.nn`	is the value used for calculating the LOF score

Value

The function returns a vector containing the FBOD outlier scores for each observation

Examples

```
library(ggplot2)
res.FBOD <- Func.FBOD(data = TestData[,1:2], iter=10, k.nn=5)
data.temp <- TestData[,1:2]
data.temp$Ind <- NA
data.temp[order(res.FBOD, decreasing = TRUE)[1:10],"Ind"] <- "Outlier"
data.temp[is.na(data.temp$Ind),"Ind"] <- "Inlier"
data.temp$Ind <- factor(data.temp$Ind)
ggplot(data = data.temp) + geom_point(aes(x = x, y = y, color=Ind, shape=Ind))
```

`Func.SNN` *A function to calculate the shared nearest neighbors (SNN)*

Description

This function calculate the shared nearest neighbors (SNN). SNN is reported to be more robust than k nearest neighbors. Firstly, the k nearest neighbor distances for each observation is calculated. Then, the shared nearest neighbor similarity is calculated based on the result of k nearest neighbor. Note that k.nn should be greater than k.sel.

Usage

```
Func.SNN(data, k.nn, k.sel)
```

Arguments

`data`	is the data frame containing the observations (should be numeric data). Each row represents an observation and each variable is stored in one column.
`k.nn`	specifies the value used for calculating the shared nearest neighbors.
`k.sel`	specifies the number of shared nearest neighbors

Value

The function returns the matrix containing the indices of top k shared nearest neighbors for each observation

Examples

```
Func.SNN(data=TestData[,1:2], k.nn=5, k.sel=3)
```

`Func.SOD` *Subspace outlier detection (SOD) algorithm*

Description

This function performs suspace outlier detection algorithm The implemented method is based on the work of Krigel, H.P., Kroger, P., Schubert, E., Zimek, A., Outlier detection in axis-parallel subspaces of high dimensional data, 2009.

Usage

```
Func.SOD(data, k.nn, k.sel, alpha = 0.8)
```

Arguments

`data` is the data frame containing the observations. Each row represents an observation and each variable is stored in one column.

`k.nn` specifies the value used for calculating the shared nearest neighbors. Note that k.nn should be greater than k.sel.

`k.sel` specifies the number shared nearest neighbors. It can be interpreted as the number of reference set for constructing the subspace hyperplane.

`alpha` specifies the lower limit for selecting subspace. 0.8 is set as default as suggested in the original paper.

Value

The function returns a vector containing the SOD outlier scores for each observation

Examples

```
library(ggplot2)
res.SOD <- Func.SOD(data = TestData[,1:2], k.nn = 10, k.sel = 5, alpha = 0.8)
data.temp <- TestData[,1:2]
data.temp$Ind <- NA
data.temp[order(res.SOD, decreasing = TRUE)[1:10],"Ind"] <- "Outlier"
data.temp[is.na(data.temp$Ind),"Ind"] <- "Inlier"
data.temp$Ind <- factor(data.temp$Ind)
ggplot(data = data.temp) + geom_point(aes(x = x, y = y, color=Ind, shape=Ind))
```

`Func.trans` *Outlier score transformation*

Description

This function calculate the transformed outlier scores, with the aim of unifying the results from different methods. The method is based on the work of Kriegel, H.P., Kroger, P., Schubert, E., Zimek, A., Interpreting and unifying outlier scores, 2011. It consists of two steps, regularization and normalization. For the ABOD scores, logarithmic inversion is used for regularization For the SOD scores, no action is taken to perform regularization For the FBOD method, the basic regularization, i.e., score-1, is used for regularization For the normalization step, the gaussian scaling method is used. The final output can be interpreted as the outlier probability, ranging from 0 to 1.

Usage

```
Func.trans(raw.score, method)
```

Arguments

`raw.score` is the scores returned by each method

`method` should be a character specifying the method used to generate the raw score. It has 3 possible values, "ABOD", "SOD", and "FBOD".

Value

The function returns the transformed outlier scores

`GoldenStatesWarriors` *The player statistics of the Golden States Warriors in the season 2013-2014*

Description

The data contains the statistics of the players in one NBA team, i.e., Golden States Warriors, during the season 2013-2014. It can be obtained from the following website: http://www.basketball-reference.com. The data has 18 rows since there were 18 players shown in the lineups for the Golden States Warriors during that season. The data has 27 columns, including the player names, age, games played, games started, minutes played, field goal made, field goal attemps, field goal percentage, 3-pointers made, 3-pointer attemps, 3-pointer percentage, 2-pointers made, 2-pointer attemps, 2-pointer percentages, effective field goal percentage, free throws made, free throw attemps, free throw percentage, offensive rebounds, defensive rebounds, total rebounds, assists, steals, blocks, turnovers, personal fouls, and total points.

Usage

```
data(TestData)
```

Format

A data frame with 18 rows and 27 variables:

`TestData` *Testing data for testing the performance of different algorithms*

Description

The data are generated according to the example in the paper "Kriegel, Kroger, Schubert, and Arthur Zimek, 2009, Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data". The data has 60 rows and 3 variables. The first two variables are the x and y coordinates. The third variable indicates the type of data, i.e., "Pattern_1", "Pattern_2", or "Outlier". The first 25 observations belong to "Pattern_1". Another 25 observations represent "Pattern_2". The other 10 observations are "Outliers".

Usage

```
data(TestData)
```

Format

A data frame with 60 rows and 3 variables:

Examples

```
library(ggplot2)
data(TestData)
ggplot(data = TestData, aes(x = x, y = y, shape=Lab, color=Lab)) + geom_point()
```

Index