Package ‘FractalTools’

January 9, 2018

Type Package

Title Fractality-based tools

Version 1.0

Date 2018-01-05

Author Mohamed Laib and Mikhail Kanevski

Maintainer Mohamed Laib <mohamed.laib@unil.ch>

Description Contains some tools based on the fractality concept such as Local-fractality and cross local fractality.

Imports Rcpp (>= 0.12.14), bigmemory, biganalytics, RcppArmadillo, BH

LinkingTo BH, Rcpp, bigmemory, RcppArmadillo

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 6.0.1.9000

R topics documented:

`FractalTools-package` *FractalTools: Fractality-based tools*

Description

Contains some tools based on the fractality concept such as Local-fractality and cross local fractality.

Details

The algorithm does not deal with missing values, please make sure to remove them. The algorithms use "bigmemory" library, which is optimized to handle big data. However, the algorithm is based on pairwise distances, and according to the computing power of your machine, large number of data points can take much time and needs more memory.

Note

This R code was developed and used for the following research: "Analysis of high dimensional environmental data using local fractality concept and machine learning", the European Geosciences Union General Assembly (EGU 2018).

Author(s)

Mohamed Laib <Mohamed.Laib@unil.ch> and

Mikhail Kanevski <Mikhail.Kanevski@unil.ch>,

References

Mikhail Kanevski, Mario G. Pereira, Local fractality: The case of forest fires in Portugal, In Physica A, Volume 479, 2017, Pages 400-410, ISSN 0378-4371.

Kanevski M., Pozdnoukhov A. and Timonin V. Machine Learning of Spatial Environmental Data. EPFL Press, 2009.

`CrossLocalSB` *Cross Local fractality (Sand-Box method) by fixing radial*

Description

Calculate the cross local fractal dimension between two datasets by using the Sand-Box method.

Usage

```
CrossLocalSB(data1, data2, rad)
```

Arguments

`data1`	First dataset. Data of class: `matrix` or `data.frame`.
`data2`	Second dataset. Data of class: `matrix` or `data.frame`.
`rad`	Vector containning values of the radial.

Value

A data.frame contains: data, columns at each radials. The last presents the Fractal dimension of each points.

Examples

```
## Not run:
D1<-matrix(runif(10*2), ncol=2)
D2<-matrix(runif(100*2), ncol=2)

D1<-as.data.frame(D1)
D2<-as.data.frame(D2)
rad<-seq(0.1,0.9,0.01)
tst<-CrossLocalSB(D1,D2, rad = rad)
tst


## End(Not run)
```

`GlobalSB` *Global fractality by fixing a radial (Sand-Box method)*

Description

Calculate the fractal dimension of a dataset by using the Sand-Box method

Usage

```
GlobalSB(data, rad)
```

Arguments

`data`	Data of class: `matrix` or `data.frame`.
`rad`	Vector containning values of the radial.

Value

A list containing

- `Fdim` Computed fractal dimension.
- `SB` Data frame of log (radial) and the log (number of points).

Examples

```
## Not run:
data <- unidat(n=10000, d=20)
data<-apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.7,0.9,0.01)
A<-GlobalSB(data, rad)
A$Fdim
plot(A[[2]][,1],A[[2]][,2],type="b",pch=16,lwd=2,xlab="log(Rad)",ylab="log(N)",
      main="Sandbox plot",  cex.main=1.5, las=1, axes=F)
axis(1)
axis(2)
grid()
```

```
Regline <- lm(A[[2]][,2]~A[[2]][,1])
abline(Regline, col="red", lwd=2)
legend("bottomright",paste("Fractal dimension:", round(A$Fdim,2)), bty="n")


#### IDmining examples #####
library(IDmining)

#### SwissRoll dataset ####

data<-SwissRoll(N=1000)
data<-apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.1,0.31,0.01)
A<-GlobalSB(data, rad)
A$Fdim
plot(A[[2]][,1],A[[2]][,2],type="b",pch=16,lwd=2,xlab="log(Rad)",ylab="log(N)",
      main="Sandbox plot",  cex.main=1.5, las=1, axes=F)
axis(1)
axis(2)
grid()
Regline <- lm(A[[2]][,2]~A[[2]][,1])
abline(Regline, col="red", lwd=2)
legend("bottomright",paste("Fractal dimension:", round(A$Fdim,2)), bty="n")

#### Butterfly dataset ####
data <- Butterfly(1000)[,-9]
data<-apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.1,0.31,0.01)
A<-GlobalSB(data, rad)
A$Fdim
plot(A[[2]][,1],A[[2]][,2],type="b",pch=16,lwd=2,xlab="log(Rad)",ylab="log(N)",
      main="Sandbox plot",  cex.main=1.5, las=1, axes=F)
axis(1)
axis(2)
grid()
Regline <- lm(A[[2]][,2]~A[[2]][,1])
abline(Regline, col="red", lwd=2)
legend("bottomright",paste("Fractal dimension:",
         round(Regline$coefficient[2],2)), bty="n")


## End(Not run)
```

`LocalSB` *Local fractality (Sand-Box method) by fixing radial*

Description

Calculate the fractal dimension of a dataset by using the Sand-Box method

Usage

```
LocalSB(data, Rad)
```

Arguments

`data`	Data of class: `matrix` or `data.frame`.
`Rad`	Vector containning values of the radial.

Value

A data.frame contains: data, columns at each radials. The last presents the Fractal dimension of each points.

Examples

```
## Not run:

#### Uniform variables ####

data <- unidat(n=10000, d=20)
data<-apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.1,1,0.01)
A<-LocalSB(data, rad)
A$Results$Fdim


#### IDmining examples #####
library(IDmining)

#### SwissRoll dataset ####

data<-SwissRoll(N=1000)
data<-apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.1,1,0.01)
A<-LocalSB(data, rad)
A$Results$Fdim
#### Butterfly dataset ####
data <- Butterfly(1000)[,-9]
data <- apply(data,MARGIN=2,FUN = function(X) (X - min(X))/diff(range(X)))
rad<-seq(0.1,0.31,0.01)
A<-LocalSB(data, rad)
A$Results$Fdim


## End(Not run)
```

`unidat` *Simulated uniform matrix*

Description

Gives a matrix of uniform variables

Usage

```
unidat(n=1000, d=20)
```

Arguments

n	Number of points.
d	Number of variables.

Value

A matrix of n*d

Examples

```
## Not run:
raw_data<-unidat(n=1000, d=10)
dim(raw_data)

## End(Not run)
```

Index