Package 'EHRsourceVariability'

May 20, 2019

Type Package

Title Measurement and visualization of biomedical data heterogeneity across data sources

Version 0.0.1

Maintainer Carlos Sáez <carsaesi@upv.es>

Description The 'EHRsourceVariability' package contains functions to measure and visualize biomedical data heterogeneity across data sources (such as locations, hospitals, professionals, etc.). The multi-source variability measurement methods include two metrics. The first measures the dissimilarity of a data source to a global central tendency of sources, namely the Source Probabilistic Outlyingness (SPO) metric. The second measures the global variability among all the data sources in a repository, namely the Global Probabilistic Deviation (GPD) metric. The metrics are complemented with an exploratory visualization of the variability among data sources, namely the Multi-Source Variability (MSV) plot. These methods serve to highlight anomalous behaviors in the data of specific sources, detect groups of sources with similar data, or provide an indicator of concordance among data sources.

License Apache License 2.0 | file LICENSE

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 6.1.1

Imports plotly, dplyr

Author Carlos Sáez [aut, cre],
Juan M García-Gómez [aut],
Biomedical Data Science Lab, Universitat Politècnica de València (Spain) [cph]

R topics documented:

`estimateMSVmetrics`	*Estimates the GPD and SPOmulti-source variability metrics and the corresponding source projection vertices from a matrix of probability distributions of different data sources*

Description

Estimates the GPD and SPOmulti-source variability metrics and the corresponding source projection vertices from a matrix of probability distributions of different data sources

Usage

```
estimateMSVmetrics(msvMetrics, nBySource, idSource)
```

Arguments

`probabilities`	m-by-n matrix containing the probability mass of n data sources on m distribution bins

Value

A list containing the following results: GPD the value of the Global Probabilistic Deviation metric, where 0 means equal distributions and 1 means non-overlapping distributions SPOs the values of Source Probabilistic Outlyingness for each data source, where 0 means equal to central tendency and 1 completely non-overlapping Vertices a n-by-(n-1) matrix containing the coordinates of each data source in the projected probabilistic space conserving their dissimilarities, e.g, for 3D projections the 3 first columns can be used

`plotMSV`	*Estimates the GPD and SPOmulti-source variability metrics and the corresponding source projection vertices from a matrix of probability distributions of different data sources*

Description

Estimates the GPD and SPOmulti-source variability metrics and the corresponding source projection vertices from a matrix of probability distributions of different data sources

Usage

```
estimateMSVmetrics(msvMetrics, nBySource, idSource)
```

Arguments

`msvMetrics`	the output of estimateMSVmetrics
`nBySource`	number of individuals for each source
`idSource`	identifier for each source (character array)

Value

a plotly plot object

Examples

```
## Not run:

library("PCAmixdata")

# We are going to estimate the MSV metrics and plot an MSV plot of the three first PCA coordinates of a dataset co
# We assume 'data' is a data.frame including numerical and categorical variables

# We get the indices of numerical and categorical data

quantidx = sapply(data,class) %in% c("numeric","integer")
qualiidx = sapply(data,class) %in% c("factor","character")

# We estimate a PCA projection using PCAmix for both numerical and categorical data
mca = PCAmix(X.quanti = NULL, X.quali = datasetVarsC2, ndim = 3, rename.level = TRUE, graph = FALSE)
coords = mca$ind$coord

# 'ID_SOURCE' contains the data source tag for each row in the data
# We get a kernel density estimation for the distributions of each source, removing those NULL estimations next

kdeData = by(coords[,1],ID_SOURCE, density, n = 100, from = min(coords[,1]), to = max(coords[,1]))
kdeData = lapply(kdeData,function(x) x$y)
kdeNull = sapply(kdeData,is.null)
kdeNotNull = kdeData[!kdeNull]
kdeDataNotNull = matrix(unlist(kdeNotNull), ncol = length(kdeNotNull), byrow = FALSE)
probMatrix = sweep(kdeDataNotNull, 2, colSums(kdeDataNotNull), FUN="/")

# We estimate the MSV metrics

msvMetrics = multisourcestability(probMatrix)
idSource = levels(ID_SOURCE)

nBySource = table(ID_SOURCE)

plotMSV(msvMetrics, nBySource, idSource)


## End(Not run)
```

Index