sDistance

January 18, 2018

`sDistance` *Function to compute the pairwise distance for a given data matrix*

Description

`sDistance` is supposed to compute and return the distance matrix between the rows of a data matrix using a specified distance metric

Usage

```
sDistance(data, metric = c("pearson", "spearman", "kendall",
"euclidean",
"manhattan", "cos", "mi"))
```

Arguments

`data`	a data frame or matrix of input data
`metric`	distance metric used to calculate a symmetric distance matrix. See 'Note' below for options available

Value

- `dist`: a symmetric distance matrix of nRow x nRow, where nRow is the number of rows of input data matrix

Note

The distance metrics are supported:

- "pearson": Pearson correlation. Note that two curves that have identical shape, but different magnitude will still have a correlation of 1
- "spearman": Spearman rank correlation. As a nonparametric version of the pearson correlation, it calculates the correlation between the ranks of the data values in the two vectors (more robust against outliers)
- "kendall": Kendall tau rank correlation. Compared to spearman rank correlation, it goes a step further by using only the relative ordering to calculate the correlation. For all pairs of data points (x_i, y_i) and (x_j, y_j), it calls a pair of points either as concordant (Nc in total) if $(x_i - x_j) * (y_i - y_j) > 0$, or as discordant ($Nd$ in total) if $(x_i - x_j) * (y_i - y_j) < 0$. Finally, it calculates gamma coefficient $(Nc - Nd)/(Nc + Nd)$ as a measure of association which is highly resistant to tied data

- "euclidean": Euclidean distance. Unlike the correlation-based distance measures, it takes the magnitude into account (input data should be suitably normalized
- "manhattan": Cityblock distance. The distance between two vectors is the sum of absolute value of their differences along any coordinate dimension
- "cos": Cosine similarity. As an uncentered version of pearson correlation, it is a measure of similarity between two vectors of an inner product space, i.e., measuring the cosine of the angle between them (using a dot product and magnitude)
- "mi": Mutual information (MI). MI provides a general measure of dependencies between variables, in particular, positive, negative and nonlinear correlations. The caclulation of MI is implemented via applying adaptive partitioning method for deriving equal-probability bins (i.e., each bin contains approximately the same number of data points). The number of bins is heuristically determined (the lower bound): $1 + log2(n)$, where n is the length of the vector. Because MI increases with entropy, we normalize it to allow comparison of different pairwise clone similarities: $2 * MI/[H(x) + H(y)]$, where $H(x)$ and $H(y)$ stand for the entropy for the vector x and y, respectively

See Also

sDmatCluster

Examples

```
# 1) generate an iid normal random matrix of 100x10
data <- matrix( rnorm(100*10,mean=0,sd=1), nrow=100, ncol=10)

# 2) calculate distance matrix using different metric
sMap <- sPipeline(data=data)
# 2a) using "pearson" metric
dist <- sDistance(data=data, metric="pearson")
# 2b) using "cos" metric
# dist <- sDistance(data=data, metric="cos")
# 2c) using "spearman" metric
# dist <- sDistance(data=data, metric="spearman")
# 2d) using "kendall" metric
# dist <- sDistance(data=data, metric="kendall")
# 2e) using "euclidean" metric
# dist <- sDistance(data=data, metric="euclidean")
# 2f) using "manhattan" metric
# dist <- sDistance(data=data, metric="manhattan")
# 2g) using "mi" metric
# dist <- sDistance(data=data, metric="mi")
```