Package ‘SoftClustering’

February 19, 2015

Title  Soft Clustering Algorithms
Description  It contains soft clustering algorithms, in particular approaches derived from rough set theory: Lingras & West original rough k-means, Peters' refined rough k-means, and PI rough k-means. It also contains classic k-means and a corresponding illustrative demo.

Version  1.1502
Author  G. Peters (Ed.)
Maintainer  G. Peters <peters.projects@outlook.com>
Depends  R (>= 3.0.2)
License  GPL-2
LazyData  true
NeedsCompilation  no
Repository  CRAN
Date/Publication  2015-02-10 13:36:07

R topics documented:

createLowerMShipMatrix ........................................ 2
datatypeInteger ............................................. 2
DemoDataC2D2a ............................................. 3
HardKMeans ................................................... 3
HardKMeansDemo ............................................. 4
initializeMeansMatrix ........................................ 5
initMeansC2D2a ............................................. 6
initMeansC3D2a ............................................. 6
initMeansC4D2a ............................................. 7
initMeansC5D2a ............................................. 7
normalizeMatrix ............................................. 8
plotRoughKMeans ............................................ 8
RoughKMeans_LW .............................................. 9
RoughKMeans_PE ............................................ 10
RoughKMeans_PI ............................................ 12
RoughKMeans_SHELL ......................................... 13

Index  15
createLowerMShipMatrix

*Create Lower Approximation*

**Description**

Creates a lower approximation out of an upper approximation.

**Usage**

```r
createLowerMShipMatrix(upperMShipMatrix)
```

**Arguments**

- `upperMShipMatrix`
  
  An upper approximation matrix.

**Value**

Returns the corresponding lower approximation.

**Author(s)**

G. Peters.

---

datatypeInteger

*Rough k-Means Plotting*

**Description**

Checks for integer.

**Usage**

```r
datatypeInteger(x)
```

**Arguments**

- `x`
  
  As a replacement for `is.integer()`. `is.integer()` delivers FALSE when the variable is numeric (as superset for integer etc.)

**Value**

TRUE if x is integer otherwise FALSE.

**Author(s)**

G. Peters.
**DemoDataC2D2a**

**A Simple Data Set**

**Description**

This is a simple two-dimensional data set with 200 objects in 2 clusters. It is provided as sample data set to easily test the algorithms.

**Usage**

```matlab
data(DemoDataC2D2a)
```

**Format**

A matrix \([\text{nClusters} \times \text{nFeatures}] = [2 \times 2]\).

**Examples**

```matlab
data(DemoDataC2D2a); plot(DemoDataC2D2a)
```

---

**HardKMeans**

**Hard k-Means**

**Description**

HardKMeans performs classic (hard) k-means.

**Usage**

```matlab
HardKMeans(dataMatrix, meansMatrix, nClusters, maxIterations)
```

**Arguments**

- **dataMatrix**: Matrix with the objects to be clustered. Dimension: \([\text{nObjects} \times \text{nFeatures}]\).
- **meansMatrix**: Select means derived from 1 = random (unity interval), 2 = maximum distances, matrix \([\text{nClusters} \times \text{nFeatures}] = \text{self-defined means}\). Default: 2 = maximum distances.
- **nClusters**: Number of clusters: Integer in \([2, \text{nObjects}]\). Note, nCluster must be set even when meansMatrix is a matrix. For transparency, nClusters will not be overridden by the number of clusters derived from meansMatrix. Default: nClusters=2.
- **maxIterations**: Maximum number of iterations. Default: maxIterations=100.
HardKMeansDemo

Value

$upperApprox$: Obtained upper approximations [$nObjects \times nClusters$]. Note: Apply function createLowerMShipMatrix() to obtain lower approximations; and for the boundary: $boundary = upperApprox - lowerApprox$.

$clusterMeans$: Obtained means [$nClusters \times nFeatures$].

$nIterations$: Number of iterations.

Author(s)


References


Examples

# An illustrative example clustering the sample data set DemoDataC2D2a.txt
HardKMeans(DemoDataC2D2a, 2, 2, 100)

<table>
<thead>
<tr>
<th>HardKMeansDemo</th>
<th>Hard k-Means Demo</th>
</tr>
</thead>
</table>

Description

HardKMeansDemo shows how hard k-means performs stepwise. The number of features is set to 2 and the maximum number of iterations is 100.

Usage

HardKMeansDemo(dataMatrix, meansMatrix, nClusters)

Arguments

dataMatrix | Matrix with the objects to be clustered. Dimension: [$nObjects \times nFeatures$]. Default: no default set.
meansMatrix | Select means derived from $1 = $random (unity interval), $2 = $maximum distances, matrix [$nClusters \times nFeatures=2$] = self-defined means. Default: meansMatrix=1 (random).
nClusters | Number of clusters: Integer in [$2, \min(5, nObjects-1)$]. Note, nCluster must be set even when meansMatrix is a matrix. For transparency, nClusters will not be overridden by the number of clusters derived from meansMatrix. Default: nClusters=2.
initializeMeansMatrix

Value
None.

Author(s)
G. Peters.

References

Examples

```r
# Clustering the data set DemoDataC2D2a.txt (nClusters=2, random initial means)
HardKMeansDemo(DemoDataC2D2a,1,2)
# Clustering the data set DemoDataC2D2a.txt (nClusters=2,3,4; initially set means)
HardKMeansDemo(DemoDataC2D2a,initMeansC2D2a,3)
HardKMeansDemo(DemoDataC2D2a,initMeansC3D2a,3)
HardKMeansDemo(DemoDataC2D2a,initMeansC4D2a,4)
# Clustering the data set DemoDataC2D2a.txt (nClusters=5, initially set means)
# It leads to an empty cluster: a (rare) case of termination of k-means.
HardKMeansDemo(DemoDataC2D2a,initMeansC5D2a,5)
```

---

**initializeMeansMatrix**  *Initialize Means Matrix*

**Description**
initializeMeansMatrix delivers an initial means matrix.

**Usage**
initializeMeansMatrix(dataMatrix, nClusters, meansMatrix)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataMatrix</td>
<td>Matrix with the objects as basis for the means matrix.</td>
</tr>
<tr>
<td>nClusters</td>
<td>Number of clusters.</td>
</tr>
<tr>
<td>meansMatrix</td>
<td>Select means derived from 1 = random (unity interval), 2 = maximum distances, matrix [nClusters x nFeatures] = self-defined means (will be returned unchanged). Default: 2 = maximum distances.</td>
</tr>
</tbody>
</table>
Value
   Initial means matrix [nClusters x nFeatures].

Author(s)

InitMeansC2D2a

Description
   This file contains initial means for an example for the HardKMEANSDemo function.

Usage
   data(initMeansC2D2a)

Format
   A data frame with 200 observations on 2 variables.

Examples
   data(initMeansC2D2a); plot(initMeansC2D2a)

InitMeansC3D2a

Description
   This file contains initial means for an example for the HardKMEANSDemo function.

Usage
   data(initMeansC3D2a)

Format
   A matrix [nClusters x nFeatures] = [3 x 2].

Examples
   data(initMeansC3D2a); plot(initMeansC3D2a)
initMeansC4D2a  Initial Means (4 Clusters)

Description
This file contains initial means for an example for the HardKMEANSDemo function.

Usage
data(initMeansC4D2a)

Format
A matrix \([nClusters \times nFeatures] = [4 \times 2]\).

Examples
data(initMeansC4D2a); plot(initMeansC4D2a)

initMeansC5D2a  Initial Means (5 Clusters)

Description
This file contains initial means for an example for the HardKMEANSDemo function.

Usage
data(initMeansC5D2a)

Format
A matrix \([nClusters \times nFeatures] = [5 \times 2]\).

Examples
data(initMeansC5D2a); plot(initMeansC5D2a)
normalizeMatrix  

**Matrix Normalization**

**Description**

normalizeMatrix delivers a normalized matrix.

**Usage**

```r
normalizeMatrix(dataMatrix, normMethod, byrow)
```

**Arguments**

- **dataMatrix**: Matrix with the objects to be normalized.
- **normMethod**: 1 = unity interval, 2 = normal distribution (sample variance), 3 = normal distribution (population variance). Any other value returns the matrix unchanged. Default: `meansMatrix = 1` (unity interval).
- **byrow**: TRUE = rows are normalized, FALSE = columns are normalized. Default: `byrow = TRUE` (rows are normalized).

**Value**

Normalized matrix.

**Author(s)**


---

plotRoughKMeans  

**Rough k-Means Plotting**

**Description**

plotRoughKMeans plots the rough clustering results in 2D. Note: Plotting is limited to a maximum of 5 clusters.

**Usage**

```r
plotRoughKMeans(dataMatrix, upperMShipMatrix, meansMatrix, plotDimensions, colouredPlot)
```
**Arguments**

- `dataMatrix`: Matrix with the objects to be plotted.
- `upperMShipMatrix`: Corresponding matrix with upper approximations.
- `meansMatrix`: Corresponding means matrix.
- `plotDimensions`: An integer vector of the length 2. Defines the to be plotted feature dimensions, i.e., `max(plotDimensions = c(1:2)) <= nFeatures`. Default: `plotDimensions = c(1:2)`.
- `colouredPlot`: Select `TRUE = colouredPlot plot, FALSE = black/white plot`.

**Value**

2D-plot of clustering results. The boundary objects are represented by stars (*).

**Author(s)**

G. Peters.

---

**RoughKMeans_LW**

*Lingras & West's Rough k-Means*

**Description**

`RoughKMeans_LW` performs Lingras & West's k-means clustering algorithm. The commonly accepted relative threshold is applied.

**Usage**

`RoughKMeans_LW(dataMatrix, meansMatrix, nClusters, maxIterations, threshold, weightLower)`

**Arguments**

- `dataMatrix`: Matrix with the objects to be clustered. Dimension: `[nObjects x nFeatures]`.
- `meansMatrix`: Select means derived from `1 = random (unity interval), 2 = maximum distances, matrix [nClusters x nFeatures] = self-defined means`. Default: `2 = maximum distances`.
- `nClusters`: Number of clusters: Integer in `[2, nObjects)`. Note, `nCluster` must be set even when `meansMatrix` is a matrix. For transparency, `nClusters` will not be overridden by the number of clusters derived from `meansMatrix`. Default: `nClusters=2`.
- `maxIterations`: Maximum number of iterations. Default: `maxIterations=100`.
- `threshold`: Relative threshold in rough k-means algorithms (threshold >= 1.0). Default: `threshold = 1.5`.
- `weightLower`: Weight of the lower approximation in rough k-means algorithms (0.0 <= weightLower <= 1.0). Default: `weightLower = 0.7`. 
Value

$\text{upperApprox}$: Obtained upper approximations [nObjects x nClusters]. Note: Apply function $\text{createLowerMShipMatrix()}$ to obtain lower approximations; and for the boundary: $\text{boundary} = \text{upperApprox} - \text{lowerApprox}$.

$\text{clusterMeans}$: Obtained means [nClusters x nFeatures].

$\text{nIterations}$: Number of iterations.

Author(s)


References


Examples

```r
# An illustrative example clustering the sample data set DemoDataC2D2a.txt
RoughKMeans_PELW(DemoDataC2D2a, 2, 2, 100, 1.5, 0.7)
```

<table>
<thead>
<tr>
<th>RoughKMeans_PE</th>
<th>Peters’ Rough k-Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Description

RoughKMeans_PE performs Peters’ k-means clustering algorithm.

Usage

RoughKMeans_PE(dataMatrix, meansMatrix, nClusters, maxIterations, threshold, weightLower)
**Arguments**

- **datamatrix**: Matrix with the objects to be clustered. Dimension: \([n\text{Objects} \times n\text{Features}]\).
- **meansMatrix**: Select means derived from 1 = random (unity interval), 2 = maximum distances, matrix \([n\text{Clusters} \times n\text{Features}]\) = self-defined means. Default: 2 = maximum distances.
- **nClusters**: Number of clusters: Integer in \([2, n\text{Objects})\). Note, nCluster must be set even when meansMatrix is a matrix. For transparency, nClusters will not be overridden by the number of clusters derived from meansMatrix. Default: nClusters=2.
- **maxIterations**: Maximum number of iterations. Default: maxIterations=100.
- **threshold**: Relative threshold in rough k-means algorithms (threshold \(\geq 1.0\)). Default: threshold = 1.5.
- **weightLower**: Weight of the lower approximation in rough k-means algorithms (0.0 \(\leq\) weightLower \(\leq 1.0\)). Default: weightLower = 0.7.

**Value**

- $upperApprox$: Obtained upper approximations \([n\text{Objects} \times n\text{Clusters}]\). Note: Apply function createLowerMShipMatrix() to obtain lower approximations; and for the boundary: boundary = upperApprox - lowerApprox.
- $clusterMeans$: Obtained means \([n\text{Clusters} \times n\text{Features}]\).
- $nIterations$: Number of iterations.

**Author(s)**


**References**


**Examples**

# An illustrative example clustering the sample data set DemoDataC2D2a.txt

```
RoughKMeans_PE(DemoDataC2D2a, 2, 2, 100, 1.5, 0.7)
```
RoughKMeans_PI  \hspace{1cm} PI Rough k-Means

Description

RoughKMeans_PI performs pi k-means clustering algorithm in its standard case. Therefore, weights are not required.

Usage

RoughKMeans_PI(dataMatrix, meansMatrix, nClusters, maxIterations, threshold)

Arguments

dataMatrix \hspace{1cm} Matrix with the objects to be clustered. Dimension: \([n\text{Objects} \times n\text{Features}]\).

meansMatrix \hspace{1cm} Select means derived from 1 = random (unity interval), 2 = maximum distances, matrix \([n\text{Clusters} \times n\text{Features}] = \text{self-defined means}\). Default: 2 = maximum distances.

nClusters \hspace{1cm} Number of clusters: Integer in \([2, n\text{Objects}]\). Note, nCluster must be set even when meansMatrix is a matrix. For transparency, nClusters will not be overridden by the number of clusters derived from meansMatrix. Default: nClusters=2.

maxIterations \hspace{1cm} Maximum number of iterations. Default: maxIterations=100.

threshold \hspace{1cm} Relative threshold in rough k-means algorithms (threshold \(\geq 1.0\)). Default: threshold = 1.5.

Value

$upper\text{Approx}$: Obtained upper approximations \([n\text{Objects} \times n\text{Clusters}]\). Note: Apply function createLowerMShipMatrix() to obtain lower approximations; and for the boundary: \(\text{boundary} = upper\text{Approx} - lower\text{Approx}\).

$cluster\text{Means}$: Obtained means \([n\text{Clusters} \times n\text{Features}]\).

$n\text{Iterations}$: Number of iterations.

Author(s)


References


Examples

# An illustrative example clustering the sample data set DemoDataC2D2a.txt
RoughKMeans_PI(DemoDataC2D2a, 2, 2, 100, 1.5)

RoughKMeans_SHELL  Rough k-Means Shell

Description

RoughKMeans_SHELL performs rough k-means algorithms with options for normalization and a 2D-plot of the results.

Usage

RoughKMeans_SHELL(clusterAlgorithm, dataMatrix, meansMatrix, nclusters, normalizationMethod, maxIterations, plotDimensions, colouredPlot, threshold, weightLower)

Arguments

clusterAlgorithm
Select 0 = classic k-means, 1 = Lingras & West’s rough k-means, 2 = Peters’ rough k-means, 3 = π rough k-means. Default: clusterAlgorithm = 3 (π rough k-means).

dataMatrix
Matrix with the objects to be clustered. Dimension: [nObjects x nFeatures].

meansMatrix
Select means derived from 1 = random (unity interval), 2 = maximum distances, matrix [nClusters x nFeatures] = self-defined means. Default: 2 = maximum distances.

nClusters
Number of clusters: Integer in [2, nObjects). Note, nCluster must be set even when meansMatrix is a matrix. For transparency, nClusters will not be overridden by the number of clusters derived from meansMatrix. Default: nClusters=2. Note: Plotting is limited to a maximum of 5 clusters.

normalizationMethod
1 = unity interval, 2 = normal distribution (sample variance), 3 = normal distribution (population variance). Any other value returns the matrix unchanged. Default: meansMatrix = 1 (unity interval).

maxIterations
Maximum number of iterations. Default: maxIterations=100.

plotDimensions
An integer vector of the length 2. Defines the to be plotted feature dimensions, i.e., max(plotDimensions = c(1:2)) <= nFeatures. Default: plotDimensions = c(1:2).

colouredPlot
Select TRUE = colouredPlot plot, FALSE = black/white plot.

threshold
Relative threshold in rough k-means algorithms (threshold >= 1.0). Default: threshold = 1.5. Note: It can be ignored for classic k-means.

weightLower
Weight of the lower approximation in rough k-means algorithms (0.0 <= weightLower <= 1.0). Default: weightLower = 0.7. Note: It can be ignored for classic k-means and π rough k-means
Value

2D-plot of clustering results. The boundary objects are represented by stars (*).

$\text{upperApprox}$: Obtained upper approximations $[n\text{Objects} \times n\text{Clusters}]$. Note: Apply function createLowerMShipMatrix() to obtain lower approximations; and for the boundary: $\text{boundary} = \text{upperApprox} - \text{lowerApprox}$.

$\text{clusterMeans}$: Obtained means $[n\text{Clusters} \times n\text{Features}]$.

$\text{nIterations}$: Number of iterations.

Author(s)


Examples

# An illustrative example clustering the sample data set DemoDataC2D2a.txt
RoughKMeans_SHELL(3, DemoDataC2D2a, 2, 2, 1, 100, c(1:2), TRUE, 1.5, 0.7)
Index

*Topic **datasets**
  DemoDataC2D2a, 3
  initMeansC2D2a, 6
  initMeansC3D2a, 6
  initMeansC4D2a, 7
  initMeansC5D2a, 7

createLowerMShipMatrix, 2

datatypeInteger, 2
DemoDataC2D2a, 3

HardKMeans, 3
HardKMeansDemo, 4

initializeMeansMatrix, 5
initMeansC2D2a, 6
initMeansC3D2a, 6
initMeansC4D2a, 7
initMeansC5D2a, 7

normalizeMatrix, 8

plotRoughKMeans, 8

RoughKMeans_LW, 9
RoughKMeans_PE, 10
RoughKMeans_PI, 12
RoughKMeans_SHELL, 13