Package ‘akmedoids’

January 10, 2020

Type Package

Title Anchored Kmedoids for Longitudinal Data Clustering

Version 0.1.5

Date 2020-01-09

Author Monsuru Adepeju [cre, aut], Samuel Langton [aut], Jon Bannister [aut]

Maintainer Monsuru Adepeju <monsuur2010@yahoo.com>

Description Advances a novel adaptation of longitudinal k-means clustering technique (Genolini et al. (2015) <doi:10.18637/jss.v065.i04>) for grouping trajectories based on the similarities of their long-term trends and determines the optimal solution based on either the average silhouette width (Rousseeuw P. J. 1987) or the Calinski-Harabatz criterion (Calinski and Harabatz (1974) <doi:10.1080/03610927408827101>). Includes functions to extract descriptive statistics and generate a visualisation of the resulting groups, drawing methods from the 'ggplot2' library (Wickham H. (2016) <doi:10.1007/978-3-319-24277-4>). The package also includes a number of other useful functions for exploring and manipulating longitudinal data prior to the clustering process.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

Imports kml, Hmisc, ggplot2, utils, reshape2, longitudinalData, stats, signal

RoxygenNote 7.0.1

Suggests knitr, rmarkdown, flextable, kableExtra, clusterCrit

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-10 00:30:06 UTC

R topics documented:

`akmedoids.clust` *Anchored k-medoids clustering*

Description

Given a list of trajectories and a functional method, this function clusters the trajectories into a `k` number of groups. If a vector of two numbers is given, the function determines the best solution from those options based on the Calinski-Harabasz criterion.

Usage

```
akmedoids.clust(traj, id_field = FALSE, method = "linear", k = c(3,6), crit="Silhouette")
```

Arguments

`traj` [matrix (numeric)]: longitudinal data. Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time steps.

`id_field` [numeric or character] Whether the first column of the `traj` is a unique (`id`) field. Default: `FALSE`. If `TRUE` the function recognises the second column as the first time points.

`method` [character] The parametric initialisation strategy. Currently, the only available method is a `linear` method, set as `"linear"`. This uses the time-dependent linear regression lines and the resulting groups are order in the order on increasing slopes.

`k` [integer or vector (numeric)] either an exact integer number of clusters, or a vector of length two specifying the minimum and maximum numbers of clusters to be examined from which the best solution will be determined. In either case, the minimum number of clusters is `3`. The default is `c(3,6)`.

crit [character] a string specifying the type of the criterion to use for assessing the quality of the cluster solutions, when k is a vector of two values (as above). Default:crit="Silhouette", use the average Silhouette width (Rousseeuw P. J. 1987). Using the "Silhouette" criterion, the optimal value of k can be determined as the elbow point of the curve. Other valid criterion is the "Calinski_Harabatz" (Calinski T. & Harabatz J. 1974) in which the maximum score represent the point of optimality. Having determined the optimal k, the function can then be re-run, using the exact (optimal) value of k.

Details

This function works by first approximating the trajectories based on the chosen parametric forms (e.g. linear), and then partitions the original trajectories based on the form groupings, in similar fashion to k-means clustering (Genolini et al. 2015). The key distinction of akmedoids compared with existing longitudinal approaches is that both the initial starting points as well as the subsequent cluster centers (as the iteration progresses) are based the selection of observations (medoids) as oppose to centroids.

Value

If k is a vector of two numbers (see param. k details above), the output is a graphical plot of the quality scores of the cluster solutions. If k is an exact integer number of clusters, the function returns trajectory labels indicating the group membership of the corresponding trajectory in the traj object.

References

1. Genolini, C. et al. (2015) kml and kml3d: R Packages to Cluster Longitudinal Data. Journal of Statistical Software, 65(4), 1-34. URL http://www.jstatsoft.org/v65/i04/. 2. Rousseeuw P. J. (1987) Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math 20:53–65. 3. Calinski T, Harabasz J (1974) A dendrite method for cluster analysis. Commun Stat 3:1-27.

Examples

```
traj <- traj
print(traj)
traj <- dataImputation(traj, id_field = TRUE, method = 2, replace_with = 1, fill_zeros = FALSE)
traj <- props(traj, id_field = TRUE)
print(traj)
output <- akmedoids.clust(traj, id_field = TRUE, method = "linear", k = c(3))
print(output)  #type 'as.vector(output$memberships)'
```

---

alphaLabel *Numerics ids to alphabetical ids*

Description

Function to transform a list of numeric ids to alphabetic ids

Usage

```
alphaLabel(x)
```

Arguments

`x` A vector of numeric ids

Details

Given a vector of numeric cluster ids, ‘alphaLabel‘ converts each id to its corresponding alphabets. It combines alphabets for ids greater than 26.

Value

A vector of alphabetical ids.

Examples

```
ids <- sample(1:100, 10, replace=FALSE)
ids_alphab <- alphaLabel(ids)
```

`dataImputation` *Data imputation for longitudinal data*

Description

This function fills any missing entries (`NA`, `Inf`, `null`) in a matrix or dataframe, according to a specified method. By default, `'0'` is considered a value.

Usage

```
dataImputation(traj, id_field = FALSE, method = 2, replace_with = 1, fill_zeros = FALSE)
```

Arguments

`traj` [`matrix (numeric)`]: longitudinal data. Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time points.

`id_field` [numeric or character] Whether the first column of the `traj` is a unique (`id`) field. Default: `FALSE`. If `TRUE` the function recognises the second column as the first time step.

`method` [an integer] indicating a method for calculating the missing values. Options are: `'1': arithmetic` method, and `'2': regression` method. The default is `'1': arithmetic` method `replace_with` [an integer from 1 to 6] indicating the technique, based on a specified `method`, for calculating the missing entries. `'1': arithmetic` method, `replace_with` options are: `'1'`: Mean value of the corresp column; `'2'`: Minimum value of corresp column; `'3'`: Maximum value of corresp column; `'4'`: Mean value of corresp row; `'5'`: Minimum value of corresp row, or `'6'`: Maximum value of corresp row. For `'2'`: regression method: the available option for the `replace_with` is: `'1': linear`. The regression method fits a linear regression line to a trajectory with missing entry(s) and estimates the missing data values from the regression line. Note: only the missing data points derive their new values from the regression line while the rest of the data points retain their original values. The function terminates if there are trajectories with only one observation. The default is `'1'`: Mean value of the corresp column

`fill_zeros` [TRUE or FALSE] whether to consider zeros `0` as missing values when `2: regression` method is used. The default is `FALSE`.

Details

Given a matrix or data.frame with some missing values indicated by (`NA`, `Inf`, `null`), this function impute the missing value by using either an estimation from the corresponding rows or columns, or to use a regression method to estimate the missing values.

Value

A data.frame with missing values (`NA`, `Inf`, `null`) imputed according to the a specified technique.

Examples

```
print(traj)
dataImputation(traj, id_field = TRUE, method = 1, replace_with = 1, fill_zeros = FALSE)
```

`elbowPoint` *Determine the elbow point on a curve*

Description

Given a list of x, y coordinates on a curve, function determines the elbow point of the curve.

Usage

```
elbowPoint(x, y)
```

Arguments

`x` vector of x coordinates of points on the curve

`y` vector of y coordinates of points on the curve

Details

highlight the maximum curvature to identify the elbow point (credit: ’github.com/agentlans’)

Value

an x, y coordinates of the elbow point.

Examples

```
# Generate some curve
x <- runif(100, min=-2, max=3)
y <- -exp(-x) * (1+rnorm(100)/3)
plot(x, y)
# Plot elbow points
abline(v=elbowPoint(x,y)$y, col=”blue”, pch=20, cex=3)
```

`outlierDetect` *Outlier detection and replacement*

Description

This function identifies outlier observations in the trajectories, and allows users to replace the observations or remove trajectories entirely.

Usage

```
outlierDetect(traj, id_field = FALSE, method = 1, threshold = 0.95,
count = 1, replace_with = 1)
```

Arguments

`traj` [matrix (numeric)]: longitudinal data. Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time points.

`id_field` [numeric or character] Whether the first column of the `traj` is a unique (`id`) field. Default: `FALSE`. If `TRUE` the function recognises the second column as the first time step.

`method` [integer (numeric)] indicating the method for identifying the outlier. Options are: `'1'`: quantile method (`default`), and `'2'`: manual method. The `manual` method requires a user-defined value.

`threshold` [numeric] A cut-off value for outliers. If the `method` parameter is set as `'1'`:quantile, the `threshold` should be a numeric vector of probability between `[0,1]`, whilst if the `method` is set as `'2'`: `manual`, the `threshold` could be any numeric vector.

`count` [integer (numeric)] indicating the number of observations (in a trajectory) that must exceed the `threshold` in order for the trajectory to be considered an `outlier`. Default is `1`.

`replace_with` [integer (numeric)] indicating the technique to use for calculating a replacement for an outlier observation. The remaining observations on the row or the column in which the outlier observation is located are used to calculate the replacement. The replacement options are: `'1'`: Mean value of the column, `'2'`: Mean value of the row and `'3'`: remove the row (trajectory) completely from the data. Default value is the `'1'` option.

Details

Given a matrix, this function identifies outliers that exceed the threshold and replaces the outliers with an estimate calculated using the other observations either the rows or the columns in which the outlier observation is located. Option is also provided to remove the trajectories (containing the outlier) from the data.

Value

A dataframe with outlier observations replaced or removed.

Examples

```
traj <- traj
traj <- dataImputation(traj, id_field=TRUE, method = 1, replace_with = 1)
traj <- props(traj, id_field=TRUE)#remove this later
outlierDetect(traj, id_field = TRUE, method = 1, threshold = 0.95,
count = 1, replace_with = 1)
outlierDetect(traj, id_field = TRUE, method = 2, threshold = 15,
count = 4, replace_with = 3)
```

`population` *sample population (denominator) data*

Description

simulated denominator data for two consecutive census years

Usage

```
population
```

Format

A matrix

props *Conversion of counts (or rates) to 'Proportion'*

Description

This function converts counts or rates to proportions.

Usage

```
props(traj, id_field = TRUE, scale = 1, digits = 4)
```

Arguments

`traj`	[matrix (numeric)]: longitudinal data. Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time points.
`id_field`	[numeric or character] Whether the first column of the `traj` is a unique (`id`) field. Default: `FALSE`. If `TRUE` the function recognises the second column as the first time step.
`scale`	[numeric] To scale the 'propotion' measures. Default: `1`
`digits`	[numeric] Specifying number of digits to approximate the output to. Default: `4`.

Details

Given a matrix of observations (counts or rates), this function converts each observation to a proportion equivalent to the sum of each column. In other words, each observation is divided by the sum of the column where it is located, i.e. `prop = [a cell value] / sum[corresponding column]`

Value

A matrix of proportion measures

Examples

```
traj <- dataImputation(traj, id_field = TRUE, method = 2, replace_with = 1,
fill_zeros = FALSE) #filling the missing values
traj <- props(traj, id_field = TRUE, scale=1, digits=4)
print(traj)
```

rates *Conversion of counts to rates*

Description

Calculates rates from 'observed' count and a denominator data

Usage

```
rates(traj, denomin, id_field, multiplier)
```

Arguments

`traj` [matrix (numeric)] longitudinal (e.g. observed count) data (`m x n`). Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time steps.

`denomin` [matrix (numeric)] longitudinal (denominator) data of the same column as 'traj' (`n`).

`id_field` [numeric or character] Default is `TRUE`. The first column of both the 'traj' and the 'denomin' object must be the unique (`id`) field. If `FALSE`, the function will terminate. The assumption is that columns of both the `traj` and `denominat` corresponds. That is, column2, column3, ... represent time points 2, 3, ..., respectively, in each object.

`multiplier` [numeric] A quantify by which to the ratio `traj/denomin` is expressed. Default is `100`.

Value

A matrix of 'rates' measures

Examples

```
traj2 <- traj
traj2 <- dataImputation(traj2, id_field = TRUE, method = 2, replace_with = 1, fill_zeros = FALSE)
pop <- population #read denominator data
pop2 <- as.data.frame(matrix(0, nrow(population), ncol(traj)))
colnames(pop2) <- names(traj2)
pop2[,1] <- as.vector(as.character(pop[,1]))
pop2[,4] <- as.vector(as.character(pop[,2]))
pop2[,8] <- as.vector(as.character(pop[,3]))
list_ <- c(2, 3, 5, 6, 7, 9, 10) #vector of missing years
#fill the missing fields with 'NA'
for(u_ in 1:length(list_)){
    pop2[,list_[u_]] <- "NA"
}
#estimate missing fields
pop_imp_result <- dataImputation(pop2, id_field = TRUE, method = 2,
replace_with = 1, fill_zeros = FALSE)
```

```
#calculate rates i.e. crimes per 200 population
crime_rates <- rates(traj2, denomin=pop_imp_result, id_field=TRUE, multiplier = 200)
```

statPrint *Descriptive (Change) statistics and plots*

Description

This function perform two tasks: (i) it generate the descriptive and change statistics of groups, particularly suited for the outputs form the akmedoids.clust function, and (ii) generates the plots of the groups (performances).

Usage

```
statPrint(
  clustr,
  traj,
  id_field = TRUE,
  reference = 1,
  N.quant = 4,
  type = "lines",
  y.scaling = "fixed"
)
```

Arguments

`clustr` [vector (charater)] A vector of cluster membership (labels). For instance, the result extracted from the akmedoids.clust function.

`traj` [matrix (numeric)]: corresponding longitudinal data used to generate clustr (with rows corresponding to each label of clustr). For example, the first label of clustr is the group label of the first row of traj matrix, and so on.

`id_field` [numeric or character] Whether the first column of the traj is a unique (id) field. Default: FALSE. If TRUE the function recognises the second column as the first time step.

`reference` [numeric] Specifying the reference line from which the direction of each group is measured. Options are: 1: slope of mean trajectory, 2: slope of medoid trajectory, 3: slope of a horizontal line (i.e. slope = 0). Default: 1.

`N.quant` [numeric] Number of equal intervals (quantiles) to create between the reference line (R) and the medoids (M) of the most-diverging groups of both sides of (R). Default is 4 - meaning quartile subdivisions on each side of (R). In this scenario, the function returns the quartile in which the medoid of each group falls. This result can be used to further categorise the groups into 'classes'. For example, groups that fall within the 1st quartile may be classified as 'Stable' groups (Adepeju et al. 2019).

`type` [character] plot type. Available options are: "lines" and "stacked".

`y.scaling` [character] works only if type="lines". y.scaling set the vertical scales of the cluster panels. Options are: "fixed": uses uniform scale for all panels, "free": uses variable scales for panels.

Details

Generates the descriptive and change statistics of the trajectory groupings. Given a vector of group membership (labels) and the corresponding data matrix (or data.frame) indexed in the same order, this function generates all the descriptive and change statistics of all the groups. The function can generate a line and an area stacked plot drawing from the functionalities of the `ggplot2` library. For a more customised visualisation, we recommend that users deploy `ggplot2` directly (`Wickham H. (2016)`).

Value

A plot showing group membership or sizes (proportion) and statistics.

References

1. Adepeju, M. et al. (2019). Anchored k-medoids: A novel adaptation of k-means further refined to measure inequality in the exposure to crime across micro places (Submitted).

`Wickham H. (2016). Elegant graphics for Data Analysis. Spring-Verlag New York (2016)`

Examples

```
print(traj)
traj <- dataImputation(traj, id_field = TRUE, method = 1, replace_with = 1,
fill_zeros = FALSE)
print(traj)
traj <- props(traj, id_field = TRUE)
clustr <- akmedoids.clust(traj, id_field = TRUE, method = "linear", k = 5)
clustr <- as.vector(clustr$memberships)
print(statPrint(clustr, traj, id_field=TRUE, type="lines", y.scaling="fixed"))
print(statPrint(clustr, traj, id_field=TRUE, reference = 1, N.quant = 8, type="stacked"))
```

`traj` *Longitudinal dataset*

Description

Simulated longitudinal datasets with missing values (`NA`, `Inf`, `null`)

Usage

```
traj
```

Format

A matrix

wSpaces *Whitespaces removal*

Description

This function removes all the leading and the trailing whitespaces in data

Usage

```
wSpaces(traj)
```

Arguments

`traj`	[matrix (numeric)]: longitudinal data. Each row represents an individual trajectory (of observations). The columns show the observations at consecutive time points.

Details

Given a matrix suspected to contain whitespaces, this function removes all the whitespaces and returns a cleaned data. ’Whitespaces’ are white characters often introduced into data during data entry, for instance by wrongly pressing the spacebar. For example, neither " A" nor "A " equates "A" because of the whitespaces that exist in them. They can also result from systematic errors in data recording devices.

Value

A matrix with all whitespaces (if any) removed.

References

https://en.wikipedia.org/wiki/Whitespace_character

Examples

```
traj <- traj
wSpaces(traj)
```

Index