Package ‘lowmemtkmeans’

January 8, 2017

Type Package
Title Low Memory Use Trimmed K-Means
Version 0.1.2
Author Andrew Thomas Jones, Hien Duy Nguyen
Maintainer Andrew Thomas Jones <andrewthomasjones@gmail.com>
Description Performs the trimmed k-means clustering algorithm with lower memory use. It also provides a number of utility functions such as BIC calculations.
License GPL (>= 3)
LazyData TRUE
LinkingTo Rcpp, RcppArmadillo
Imports Rcpp (>= 0.12.5)
SystemRequirements C++11
RoxygenNote 5.0.1
Suggests testthat
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-01-08 12:07:33

R topics documented:

  cluster_BIC ................................................................. 2
  nearest_cluster ........................................................... 2
  scale_mat_inplace ......................................................... 3
  tkmeans ................................................................. 4

Index 6
cluster_BIC  \hspace{1cm} \textit{Calculates BIC for a given clustering.}

\section*{Description}
Computes Bayesian information criterion for a given clustering of a data set.

\section*{Usage}
cluster_BIC(data, centres)

\section*{Arguments}
\begin{description}
\item[data] a matrix (n x m). Rows are observations, columns are predictors.
\item[centres] matrix of cluster means (k x m), where k is the number of clusters.
\end{description}

\section*{Details}
Bayesian information criterion (BIC) is calculated using the formula, BIC = -2 * log(L) + k*log(n). k is the number of free parameters, in this case is m*k + k - 1. n is the number of observations (rows of data). L is the likelihood for the given set of cluster centres.

\section*{Value}
BIC value

\section*{Examples}
\begin{verbatim}
iris_mat <- as.matrix(iris[,1:4])
iris_centres2 <- kmeans(iris_mat, 2, 0.1, c(1,1,1,1), 1, 10, 0.001) # 2 clusters
iris_centres3 <- kmeans(iris_mat, 3, 0.1, c(1,1,1,1), 1, 10, 0.001) # 3 clusters
cluster_BIC(iris_mat, iris_centres2)
cluster_BIC(iris_mat, iris_centres3)
\end{verbatim}

\begin{center}
\begin{table}
\hline
\textbf{nearest_cluster}  \hspace{1cm} \textit{Allocates each rw (observation) in data to the nearest cluster centre.}
\hline
\end{table}
\end{center}

\section*{Description}
For each observation the euclidean distance to each of the cluster centres is calculated and cluster with the smallest distance is return for that observation.

\section*{Usage}
\begin{verbatim}
nearest_cluster(data, centres)
\end{verbatim}
scale_mat_inplace

Arguments

- **data**: a matrix (n x m) to be clustered
- **centres**: matrix of cluster means (k x m), where k is the number of clusters.

Value

- vector of cluster allocations, n values ranging from 1 to k.

Examples

```r
iris_mat <- as.matrix(iris[,1:4])
centres <- tkmeans(iris_mat, 3, 0.2, c(1,1,1,1), 1, 10, 0.001)
nearest_cluster(iris_mat, centres)
```

scale_mat_inplace

Rescales a matrix in place.

Description

Rescales matrix so that each column has a mean of 0 and a standard deviation of 1. The original matrix is overwritten in place. The function returns the means and standard deviations of each column used to rescale it.

Usage

```r
scale_mat_inplace(M)
```

Arguments

- **M**: matrix of data (n x m)

Details

The key advantage of this method is that it can be applied to very large matrices without having to make a second copy in memory and the original can still be restored using the saved information.

Value

- Returns a matrix of size (2 x m). The first row contains the column means. The second row contains the column standard deviations. NOTE: The original matrix, M, is overwritten.

Examples

```r
m = matrix(rnorm(24, 1, 2),4, 6)
scale_params = scale_mat_inplace(m)
sweep(sweep(m,2,scale_params[2,], '+'), 2, scale_params[1,], '+') # original matrix restored
```
tkmeans  

Trimmed k-means clustering

Description
Perform trimmed k-means clustering algorithm [1] on a matrix of data. Each row in the data is an observation, each column is a variable. For optimal use columns should be scaled to have the same means and variances using scale_mat_inplace.

Usage
tkmeans(M, k, alpha, weights = rep(1, ncol(M)), nstart = 1L, iter = 10L, 
tol = 1e-04, verbose = FALSE)

Arguments
- M: matrix (n x m). Rows are observations, columns are predictors.
- k: number of clusters
- alpha: proportion of data to be trimmed
- weights: weightings for variables (columns).
- nstart: number of restarts
- iter: maximum number of iterations
- tol: criteria for algorithm convergence
- verbose: If true will output more information on algorithm progress.

Details
k is the number of clusters. alpha is the proportion of data that will be excluded in the clustering. Algorithm will halt if either maximum number of iterations is reached or the change between iterations drops below tol.
When n_starts is greater than 1, the algorithm will run multiple times and the result with the best BIC will be returned. The centres are intialised by picking k observations.
The function only returns the k cluster centres. To calculate the nearest cluster centre for each observation use the function nearest_cluster.

Value
Returns a matrix of cluster means (k x m).

References
Examples

iris_mat <- as.matrix(iris[,1:4])
scale_params <- scale_mat_inplace(iris_mat)
iris_cluster <- tkmeans(iris_mat, 2, 0.1, c(1,1,1,1), 1, 10, 0.001) # 2 clusters
Index

cluster_BIC, 2
nearest_cluster, 2
scale_mat_inplace, 3
tkmeans, 4