Package 'klic'

October 13, 2022

Title Kernel Learning Integrative Clustering

Version 1.0.4

Author Alessandra Cabassi [aut, cre] (<https://orcid.org/0000-0003-1605-652X>),
Paul DW Kirk [ths] (<https://orcid.org/0000-0002-5931-7489>),
Mehmet Gonen [ctb] (<https://orcid.org/0000-0002-2483-075X>)

Description Kernel Learning Integrative Clustering (KLIC) is an algorithm that allows to combine multiple kernels, each representing a different measure of the similarity between a set of observations. The contribution of each kernel on the final clustering is weighted according to the amount of information carried by it. As well as providing the functions required to perform the kernel-based clustering, this package also allows the user to simply give the data as input: the kernels are then built using consensus clustering. Different strategies to choose the best number of clusters are also available. For further details please see Cabassi and Kirk (2020) <doi:10.1093/bioinformatics/btaa593>.

Depends R (>= 3.5.0)

License MIT + file LICENSE

URL http://github.com/acabassi/klic

BugReports http://github.com/acabassi/klic/issues

Encoding UTF-8

LazyData true

Imports Matrix, cluster, coca, RColorBrewer, pheatmap, utils

Suggests Rmosek, tikzDevice, mclust, grDevices, graphics, knitr,
markdown

SystemRequirements MOSEK (http://www.mosek.com) and MOSEK license.

RoxygenNote 7.1.0

VignetteBuilder knitr

NeedsCompilation no

Maintainer Alessandra Cabassi <alessandra.cabassi@mrc-bsu.cam.ac.uk>

Repository CRAN

Date/Publication 2020-07-06 16:50:03 UTC

R topics documented:

`copheneticCorrelation` *Cophenetic correlation coefficient*

Description

Compute the cophenetic correlation coefficient of a kernel matrix, which is a measure of how faithfully hierarchical clustering would preserve the pairwise distances between the original data points.

Usage

```
copheneticCorrelation(kernelMatrix)
```

Arguments

`kernelMatrix`	kernel matrix.

Value

This functions returns the cophenetic correlation coefficient of the kernel matrix provided as input.

Author(s)

Alessandra Cabassi <`alessandra.cabassi@mrc-bsu.cam.ac.uk`>

References

Cabassi, A. and Kirk, P. D. W. (2019). Multiple kernel learning for integrative consensus clustering of genomic datasets. arXiv preprint. arXiv:1904.07701.

Sokal, R.R. and Rohlf, F.J., 1962. The comparison of dendrograms by objective methods. Taxon, 11(2), pp.33-40.

Examples

```
# Load kernel matrix
consensus_matrix <- as.matrix(read.csv(system.file('extdata',
'consensus_matrix1.csv', package = 'klic'), row.names = 1))

# Compute cophenetic correlation
coph_corr_coeff <- copheneticCorrelation(consensus_matrix)
cat(coph_corr_coeff)
```

kkmeans *Kernel k-means*

Description

Perform the training step of kernel k-means.

Usage

```
kkmeans(K, parameters)
```

Arguments

`K`	Kernel matrix.
`parameters`	A list containing the number of clusters `number_count`.

Value

This function returns a list containing:

`clustering`	the cluster labels for each element (i.e. row/column) of the kernel matrix.
`objective`	the value of the objective function for the given clustering.
`parameters`	same parameters as in the input.

Author(s)

Mehmet Gonen

References

Gonen, M. and Margolin, A.A., 2014. Localized data fusion for kernel k-means clustering with application to cancer biology. In Advances in Neural Information Processing Systems (pp. 1305-1313).

Examples

```
# Load one dataset with 100 observations, 2 variables, 4 clusters
data <- as.matrix(read.csv(system.file("extdata", "dataset1.csv",
package = "klic"), row.names = 1))
# Compute consensus clustering with K=4 clusters
cm <- coca::consensusCluster(data, 4)
# Shift eigenvalues of the matrix by a constant: (min eigenvalue) * (coeff)
km <- spectrumShift(cm, coeff = 1.05)
# Initalize the parameters of the algorithm
parameters <- list()
# Set the number of clusters
parameters$cluster_count <- 4
# Perform training
state <- kkmeans(km, parameters)
# Display the clustering
print(state$clustering)
```

klic *Kernel learning integrative clustering*

Description

This function allows to perform Kernel Learning Integrative Clustering on M data sets relative to the same observations. The similarities between the observations in each data set are summarised into M different kernels, that are then fed into a kernel k-means clustering algorithm. The output is a clustering of the observations that takes into account all the available data types and a set of weights that sum up to one, indicating how much each data set contributed to the kernel k-means clustering.

Usage

```
klic(
  data,
  M,
  individualK = NULL,
  individualMaxK = 6,
  individualClAlgorithm = "kkmeans",
  globalK = NULL,
  globalMaxK = 6,
  B = 1000,
  C = 100,
  scale = FALSE,
  savePNG = FALSE,
  fileName = "klic",
  verbose = TRUE,
  annotations = NULL,
  ccClMethods = "kmeans",
  ccDistHCs = "euclidean",
  widestGap = FALSE,
  dunns = FALSE,
  dunn2s = FALSE
)
```

Arguments

`data` List of M datasets, each of size N X P_m, m = 1, ..., M.

`M` number of datasets.

`individualK` Vector containing the number of clusters in each dataset. Default is NULL. If the number of clusters is not provided, then all the possible values between 2 and individualMaxK are considered and the best value is chosen for each dataset by maximising the silhouette.

`individualMaxK` Maximum number of clusters considered for the individual data. Default is 6.

`individualClAlgorithm`
Clustering algorithm used for clustering of each dataset individually if is required to find the best number of clusters.

`globalK` Number of global clusters. Default is NULL. If the number of clusters is not provided, then all the possible values between 2 and globalMaxK are considered and the best value is chosen by maximising the silhouette.

`globalMaxK` Maximum number of clusters considered for the final clustering. Default is 6.

`B` Number of iterations for consensus clustering. Default is 1000.

`C` Maximum number of iterations for localised kernel k-means. Default is 100.

`scale` Boolean. If TRUE, each dataset is scaled such that each column has zero mean and unitary variance.

`savePNG` Boolean. If TRUE, a plot of the silhouette is saved in the working folder. Default is FALSE.

`fileName` If `savePNG` is TRUE, this is the name of the png file. Can be used to specify the folder path too. Default is "klic".

`verbose` Boolean. Default is TRUE.

`annotations` Data frame containing annotations for final plot.

`ccClMethods` The i-th element of this vector goes into the `clMethod` argument of consensusCluster() for the i-th dataset. If only one string is provided, then the same method is used for all datasets.

`ccDistHCs` The i-th element of this vector goes into the `dist` argument of `consensusCluster()` for the i-th dataset.

`widestGap` Boolean. If TRUE, compute also widest gap index to choose best number of clusters. Default is FALSE.

`dunns` Boolean. If TRUE, compute also Dunn's index to choose best number of clusters. Default is FALSE.

`dunn2s` Boolean. If TRUE, compute also alternative Dunn's index to choose best number of clusters. Default is FALSE.

Value

The function returns a list contatining:

`consensusMatrices`
an array containing one consensus matrix per data set.

`weights` a vector containing the weights assigned by the kernel k-means algorithm to each consensus matrix.

`weightedKM` the weighted kernel matrix obtained by taking a weighted sum of all kernels, where the weights are those specified in the `weights` matrix.

`globalClusterLabels`
a vector containing the cluster labels of the observations, according to kernel k-means clustering done on the kernel matrices.

`bestK` a vector containing the best number of clusters between 2 and `maxIndividualK` for each kernel. These are chosen so as to maximise the silhouette and only returned if the number of clusters `individualK` is not provided.

`globalK` the best number of clusters for the final (global) clustering. This is chosen so as to maximise the silhouette and only returned if the final number of clusters `globalK` is not provided.

Author(s)

Alessandra Cabassi <`alessandra.cabassi@mrc-bsu.cam.ac.uk`>

References

Cabassi, A. and Kirk, P. D. W. (2019). Multiple kernel learning for integrative consensus clustering of genomic datasets. arXiv preprint. arXiv:1904.07701.

Examples

```
if(requireNamespace("Rmosek", quietly = TRUE) &&
(!is.null(utils::packageDescription("Rmosek")$Configured.MSK_VERSION))){

# Load synthetic data
data1 <- as.matrix(read.csv(system.file('extdata',
'dataset1.csv', package = 'klic'), row.names = 1))
data2 <- as.matrix(read.csv(system.file('extdata',
'dataset2.csv', package = 'klic'), row.names = 1))
data3 <- as.matrix(read.csv(system.file('extdata',
'dataset3.csv', package = 'klic'), row.names = 1))
data <- list(data1, data2, data3)

# Perform clustering with KLIC assuming to know the
# number of clusters in each individual dataset and in
# the final clustering
klicOutput <- klic(data, 3, individualK = c(4, 4, 4),
globalK = 4, B = 30, C = 5)

# Extract cluster labels
klic_labels <- klicOutput$globalClusterLabels

cluster_labels <- as.matrix(read.csv(system.file('extdata',
'cluster_labels.csv', package = 'klic'), row.names = 1))
# Compute ARI
ari <- mclust::adjustedRandIndex(klic_labels, cluster_labels)
}
```

lmkkmeans *Localised multiple kernel k-means*

Description

Perform the training step of the localised multiple kernel k-means.

Usage

```
lmkkmeans(Km, parameters, verbose = FALSE)
```

Arguments

`Km` An array of size N x N x M containing M different N x N kernel matrices.

`parameters` A list of parameters containing the desired number of clusters, `cluster_count`, and the number of iterations of the algorithm to be run, `iteration_count`.

`verbose` Boolean flag. If TRUE, at each iteration the iteration number is printed. Default is FALSE.

Value

This function returns a list containing:

`clustering` the cluster labels for each element (i.e. row/column) of the kernel matrix.

`objective` the value of the objective function for the given clustering.

`parameters` same parameters as in the input.

`Theta` N x M matrix of weights, each row corresponds to an observation and each column to one of the kernels.

Author(s)

Mehmet Gonen

References

Gonen, M. and Margolin, A.A., 2014. Localized data fusion for kernel k-means clustering with application to cancer biology. In Advances in Neural Information Processing Systems (pp. 1305-1313).

Examples

```
if(requireNamespace("Rmosek", quietly = TRUE) &&
(!is.null(utils::packageDescription("Rmosek")$Configured.MSK_VERSION))){

# Initialise 100 x 100 x 3 array containing M kernel matrices
# representing three different types of similarities between 100 data points
km <- array(NA, c(100, 100, 3))
# Load kernel matrices
km[,,1] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix1.csv', package = 'klic'), row.names = 1))
km[,,2] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix2.csv', package = 'klic'), row.names = 1))
km[,,3] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix3.csv', package = 'klic'), row.names = 1))

# Initalize the parameters of the algorithm
parameters <- list()
# Set the number of clusters
parameters$cluster_count <- 4
# Set the number of iterations
parameters$iteration_count <- 10

# Perform training
state <- lmkkmeans(km, parameters)

# Display the clustering
print(state$clustering)
# Display the kernel weights
print(state$Theta)
}
```

lmkkmeans_missingData *Localised multiple kernel k-means*

Description

Perform the training step of the localised multiple kernel k-means.

Usage

```
lmkkmeans_missingData(Km, parameters, missing = NULL, verbose = FALSE)
```

Arguments

Km	Array of size N X N X M containing M different N x N kernel matrices.
parameters	A list of parameters containing the desired number of clusters, cluster_count, and the number of iterations of the algorithm to be run, iteration_count.
missing	Matrix of size N X M containing missingness indicators, i.e. missing[i,j] = 1 (or = TRUE) if observation i is missing in dataset j, missing[i,j] = 0 (or = FALSE).

`verbose` Boolean flag. If TRUE, at each iteration the iteration number is printed. Defaults to FALSE.

Value

This function returns a list containing:

`clustering` the cluster labels for each element (i.e. row/column) of the kernel matrix.

`objective` the value of the objective function for the given clustering.

`parameters` same parameters as in the input.

`Theta` N x M matrix of weights, each row corresponds to an observation and each column to one of the kernels.

Author(s)

Mehmet Gonen, Alessandra Cabassi

References

Gonen, M. and Margolin, A.A., 2014. Localized data fusion for kernel k-means clustering with application to cancer biology. In Advances in Neural Information Processing Systems (pp. 1305-1313).

Examples

```
if(requireNamespace("Rmosek", quietly = TRUE) &&
(!is.null(utils::packageDescription("Rmosek")$Configured.MSK_VERSION))){

# Intialise 100 x 100 x 3 array containing M kernel matrices
# representing three different types of similarities between 100 data points
km <- array(NA, c(100, 100, 3))
# Load kernel matrices
km[,,1] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix1.csv', package = 'klic'), row.names = 1))
km[,,2] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix2.csv', package = 'klic'), row.names = 1))
km[,,3] <- as.matrix(read.csv(system.file('extdata',
'kernel_matrix3.csv', package = 'klic'), row.names = 1))
# Introduce some missing data
km[76:80, , 1] <- NA
km[, 76:80, 1] <- NA

# Define missingness indicators
missing <- matrix(FALSE, 100, 3)
missing[76:80,1] <- TRUE

# Initalize the parameters of the algorithm
parameters <- list()
# Set the number of clusters
parameters$cluster_count <- 4
# Set the number of iterations
parameters$iteration_count <- 10

# Perform training
state <- lmkkmeans_missingData(km, parameters, missing)

# Display the clustering
print(state$clustering)
# Display the kernel weights
print(state$Theta)
}
```

plotSimilarityMatrix *Plot similarity matrix with pheatmap*

Description

Plot similarity matrix with pheatmap

Usage

```
plotSimilarityMatrix(
  X,
  y = NULL,
  clusLabels = NULL,
  colX = NULL,
  colY = NULL,
  myLegend = NULL,
  fileName = "posteriorSimilarityMatrix",
  savePNG = FALSE,
  semiSupervised = FALSE,
  showObsNames = FALSE,
  clr = FALSE,
  clc = FALSE,
  plotWidth = 500,
  plotHeight = 450
)
```

Arguments

`X`	Similarity matrix.
`y`	Vector
`clusLabels`	Cluster labels
`colX`	Colours for the matrix
`colY`	Colours for the response
`myLegend`	Vector of strings with the names of the variables

`fileName` If `savePNG` is TRUE, this is the string containing the name of the output file. Can be used to specify the folder path too. Default is "posteriorSimilarityMatrix". The extension ".png" is automatically added to this string.

`savePNG` Boolean: if TRUE, the plot is saved as a png file. Default is FALSE.

`semiSupervised` Boolean flag: if TRUE, the response is plotted next to the matrix.

`showObsNames` Boolean. If TRUE, observation names are shown in the plot. Default is FALSE.

`clr` Boolean. If TRUE, rows are ordered by hierarchical clustering. Default is FALSE.

`clc` Boolean. If TRUE, columns are ordered by hierarchical clustering. Default is FALSE.

`plotWidth` Plot width. Default is 500.

`plotHeight` Plot height. Default is 450.

Value

No return value. This function plots the similarity matrix either to screen or to a png file.

Author(s)

Alessandra Cabassi <`alessandra.cabassi@mrc-bsu.cam.ac.uk`>

Examples

```
# Load one dataset with 100 observations, 2 variables, 4 clusters
data <- as.matrix(read.csv(system.file("extdata", "dataset1.csv",
package = "klic"), row.names = 1))
# Load cluster labels
cluster_labels <- as.matrix(read.csv(system.file("extdata",
"cluster_labels.csv", package = "klic"), row.names = 1))

# Compute consensus clustering with K=4 clusters
cm <- coca::consensusCluster(data, 4)

# Plot consensus (similarity) matrix
plotSimilarityMatrix(cm)

# Plot consensus (similarity) matrix with response
names(cluster_labels) <- as.character(1:100)
rownames(cm) <- names(cluster_labels)
plotSimilarityMatrix(cm, y = cluster_labels)
```

spectrumShift *Spectrum shift*

Description

Make a symmetric matrix positive semi-definite.

Usage

```
spectrumShift(kernelMatrix, coeff = 1.2, shift = NULL, verbose = FALSE)
```

Arguments

kernelMatrix	symmetric matrix
coeff	Coefficient by which the minimum eigenvalue is multiplied when shifting the eigenvalues, in order to avoid numeric problems. Default is 1.2.
shift	Value of the constant added to the diagonal, if known a priori. Default is NULL.
verbose	Boolean flag: if TRUE, information about the shift is printed to screen. Default is FALSE.

Value

This function returns the matrix `kernelMatrix` after applying the required spectrum shift.

Author(s)

Alessandra Cabassi <alessandra.cabassi@mrc-bsu.cam.ac.uk>

Examples

```
# Load one dataset with 300 observations, 2 variables, 6 clusters
data <- as.matrix(read.csv(system.file("extdata", "dataset1.csv",
package = "klic"), row.names = 1))

# Compute consensus clustering with K=4 clusters
cm <- coca::consensusCluster(data, 4)

# Shift eigenvalues of the matrix by a constant: (min eigenvalue) * (coeff)
km <- spectrumShift(cm, coeff = 1.05)
```

Index