Package 'ClustAll'

December 30, 2024

Type Package

Title ClustAll: Data driven strategy to robustly identify stratification of patients within complex diseases

Version 1.2.0

Imports FactoMineR, bigstatsr, clValid, doSNOW, parallel, foreach, dplyr, fpc, mice, modeest, flock, networkD3, methods, ComplexHeatmap, cluster, RColorBrewer, circlize, grDevices, ggplot2, grid, stats, utils, pbapply

Suggests RUnit, knitr, BiocGenerics, rmarkdown, BiocStyle, roxygen2

Description Data driven strategy to find hidden groups of patients with complex diseases using clinical data. ClustAll facilitates the unsupervised identification of multiple robust stratifications. ClustAll, is able to overcome the most common limitations found when dealing with clinical data (missing values, correlated data, mixed data types).

Depends R (>= 4.2.0)

License GPL-2

Encoding UTF-8

biocViews Software, StatisticalMethod, Clustering, DimensionReduction, PrincipalComponent

RoxygenNote 7.3.2

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/ClustAll

git_branch RELEASE_3_20

git_last_commit 73c80c0

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2024-12-30

Author Asier Ortega-Legarreta [aut, cre] (<https://orcid.org/0009-0000-3563-5362>), Sara Palomino-Echeverria [aut]

Maintainer Asier Ortega-Legarreta <aortegal@navarra.es>

Contents

`addValidationData` *Add Validation Data to ClustAllObject*

Description

This function adds or updates the validation data (true labels) in a ClustAllObject. It allows users to incorporate known classifications or groupings of samples after the object has been created, enabling subsequent evaluation of clustering results against these true labels.

Usage

```
addValidationData(Object, dataValidation)
```

Arguments

Object	A ClustAllObject-class object created by createClustAll.
dataValidation	A numeric or character vector containing the validation data (true labels) for the samples. The length of this vector must match the number of rows in the original input data used in createClustAll.

Details

Adding validation data to a ClustAllObject is crucial for assessing the performance of clustering results. This function allows for flexible workflow where true labels can be incorporated at any stage of the analysis:

- It can be used to add validation data that was not available during initial object creation.
- It can also update existing validation data with new or corrected labels.
- The added data can be used with functions like validateStratification to evaluate clustering performance.

Key points to consider:

- The length of dataValidation must exactly match the number of samples in the original dataset.
- If the object already contains validation data, this function will overwrite it with the new data.
- Both numeric and character vectors are accepted, allowing for various types of classification schemes.

Value

An updated ClustAllObject-class object with the new validation data added to the dataValidation slot.

Note

This function modifies the 'dataValidation' slot of the ClustAllObject in-place. Always ensure that the provided validation data correctly corresponds to the samples in your dataset to avoid misinterpretation of subsequent analyses.

See Also

createClustAll, dataValidation, validateStratification, ClustAllObject-class

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
label <- as.numeric(as.factor(wdbc$Diagnosis))
wdbc <- wdbc[,-c(1, 2)] # delete patients IDs & label
obj_noNA <- createClustAll(data = wdbc)
obj_noNA <- addValidationData(Object = obj_noNA,
                              dataValidation = label)
```

characterOrNA-class *Class Union: characterOrNA*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a character value, NULL, or a missing value. It is particularly useful when a slot or function parameter might contain a character string but could also be empty, unspecified, or explicitly set to NULL.

Details

The characterOrNA class union includes:

- character: A standard R character string or vector
- NULL: Representing an empty or unset value
- missing: Allowing for unspecified parameters in function calls

This union is useful in scenarios where:

- A function might return a character result or NULL if no result is available
- A slot in an S4 object could contain a character value or be empty
- A function parameter could accept a character input, but also work with default settings if nothing is provided

See Also

setClassUnion, ClustAllObject-class

ClustAllObject-class *ClustAllObject*

Description

The ClustAllObject class is the central data structure of the ClustAll package, designed to store and manage data and results throughout the patient stratification process. It encapsulates the original data, preprocessed data, imputation results, and clustering outcomes, providing a cohesive framework for the entire ClustALL workflow.

Details

The ClustAllObject is designed to efficiently manage all aspects of the data and results throughout the ClustAll pipeline:

- It preserves the original data while storing preprocessed versions for analysis.
- It handles missing data through multiple imputation, storing both original and imputed datasets.
- It maintains separation between data used for clustering and validation data.
- It stores all generated stratifications and identifies robust solutions.
- It provides a framework for comparing different stratification solutions.

This structure allows for a streamlined workflow from data input through preprocessing, imputation (if needed), stratification, and final analysis of results.

Value

ClustAllObject class object.

Slots

`data` A data frame containing the preprocessed input data after applying one-hot encoding to categorical variables and removing the validation column (if present). This is the data used directly in the clustering process.

`dataOriginal` A data frame containing the original, unmodified input data as provided to `createClustAll`. This preserves the initial state of the data for reference and validation purposes.

`dataImputed` If imputation was applied, this slot contains a 'mids' object from the mice package with the imputed datasets. If no imputation was performed, this is NULL.

`dataValidation` A numeric vector containing the reference labels (true labels) of the original dataset, if provided. These labels are used for validation purposes and are not used in the stratification process itself. NULL if no validation data is available.

`nImputation` An integer indicating the number of imputations to be computed. It should be set to 0 if the dataset is already completed, or if it the dataset has been imputed out of ClustAll framework.

`processed` A logical flag. TRUE if `runClustAll` has been executed on the object, FALSE otherwise. Indicates whether the object contains stratification results.

`summary_clusters` A list containing the resulting stratifications for each combination of clustering methods (distance metric + clustering algorithm) and embedding depth. This is populated after `runClustAll` has been executed. NULL otherwise.

`JACCARD_DISTANCE_F` A matrix of Jaccard distances between the robust stratifications that passed the bootstrapping process. Used to assess similarity between different stratification solutions. NULL if `runClustAll` has not been executed.

Note

The ClustAllObject is typically created using the `createClustAll` function and processed using `runClustAll`. Direct manipulation of the object's slots is not recommended as it may lead to inconsistencies in the analysis pipeline.

`cluster2data` *Export Stratification Results with Original Data*

Description

This function combines the original input data with one or more selected stratifications from the ClustALL algorithm results. It allows users to examine how samples are clustered in the context of their original features, facilitating further analysis and interpretation of the stratification results.

Usage

```
cluster2data(Object, stratificationName)
```

Arguments

`Object`	A processed `ClustAllObject-class` object. The object must have been processed by `runClustAll` before using this function.
`stratificationName`	A character vector specifying the names of one or more stratifications to be exported. These names should correspond to stratifications generated by the ClustALL algorithm and stored in the Object.

Details

The cluster2data function serves some important purposes in the ClustALL workflow:

1. Data Integration: It combines clustering results with the original feature data, allowing for comprehensive analysis of how cluster assignments relate to input variables.

2. Preparation for External Analysis: The resulting data frame can be easily exported for use in other analytical tools or visualization software.

The function is particularly useful for:

- Identifying features that distinguish different clusters
- Comparing how samples are grouped across different stratifications
- Preparing data for cluster-specific statistical analyses
- Creating visualizations that incorporate both cluster assignments and original features

Value

A data.frame that includes the original data with additional column(s) containing the selected stratification(s). Each selected stratification will be added as a separate column to the original dataset.

Note

- This function requires a processed ClustAllObject. Ensure `runClustAll` has been executed before using cluster2data.
- The stratificationName parameter accepts multiple stratification names, allowing for simultaneous export of multiple clustering solutions.
- Stratification names can be obtained from the results of `resStratification` or by examining the names in the summary_clusters slot of the ClustAllObject.
- The original data in the returned data frame includes all preprocessing steps applied during the creation of the ClustAllObject, such as one-hot encoding of categorical variables.

See Also

`runClustAll`, `resStratification`, `ClustAllObject-class`, `createClustAll`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)

obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
resStratification(Object = obj_noNA1, population = 0.05,
```

```
                    stratification_similarity = 0.88, all = FALSE)
df <- cluster2data(Object = obj_noNA1,
                   stratificationName = c("cuts_a_1","cuts_b_5","cuts_a_5"))
```

`createClustAll` *Create ClustAllObject for Patient Stratification Analysis*

Description

This function initializes a ClustAllObject, which serves as the core data structure for the ClustAll package. It preprocesses the input data, handles missing values through imputation if necessary, and prepares the data for subsequent clustering analysis.

Usage

```
createClustAll(data, nImputation = NULL, dataImputed = NULL,
                       colValidation = NULL)
```

Arguments

`data`	A data frame containing the input data. It may include missing values (NAs) and can contain both numeric and categorical variables.
`nImputation`	An integer specifying the number of imputations to perform if the data contains missing values. Default is NULL, which means no imputation unless missing values are detected.
`dataImputed`	A 'mids' object created by the mice package, containing pre-computed imputations. If provided, this will be used instead of performing new imputations. Default is NULL.
`colValidation`	A character string specifying the name of the column in 'data' that contains validation labels (true classifications). This column will be extracted and stored separately. Default is NULL.

Details

The createClustAll function performs several key steps in preparing data for the ClustAll analysis pipeline:

1. Data Preprocessing:
   - Applies one-hot encoding to categorical variables, converting them to a format suitable for clustering algorithms.
   - Extracts the validation column (if specified) and stores it separately.
2. Missing Data Handling:
   - If the data contains missing values and no imputed data is provided, it performs multiple imputation using the mice package.
   - The number of imputations is determined by the 'nImputation' parameter or set to a default if not specified.
3. Object Initialization:

- Creates a new ClustAllObject with slots for original data, processed data, imputed data (if applicable), and validation data (if provided).

The function accommodates three main scenarios:

- Data without missing values: Preprocessing is applied, no imputation needed.
- Data with missing values, no pre-computed imputations: Automatic imputation is performed.
- Data with missing values and pre-computed imputations: The provided imputations are used.

Value

An unprocessed ClustAllObject without stratification results. Use runClustAll on this object to execute the ClustAll pipeline.

Note

- Categorical variables in the input data should be coded as factors.
- If 'dataImputed' is provided, it must correspond exactly to the input data in terms of dimensions and variable names.
- The 'colValidation' parameter allows for the incorporation of known classifications, which can be used later for validating clustering results.
- After creating the ClustAllObject, use `runClustAll` to perform the actual clustering analysis.

See Also

`runClustAll`, `ClustAllObject-class`, `addValidationData`

Examples

```
# Scenario 1: data does not contain missing values
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- wdbc[,-c(1,2)]
obj_noNA <- createClustAll(data = wdbc)

# Scenario 2: data contains NAs and there is no imputed data.
# Then it performs the imputations automatically
data("BreastCancerWisconsinMISSING", package = "ClustAll")

obj_NA <- createClustAll(wdbcNA, nImputation = 2)

# Scenario 3: data contains NAs and imputed data is provided manually
data("BreastCancerWisconsinMISSING", package = "ClustAll")
ini <- mice::mice(wdbcNA, maxit = 0, print = FALSE)
pred <- ini$pred # predictor matrix
pred["radius1", c("perimeter1", "area1", "smoothness1")] <- 0 # example of
# how to remove predictors

imp <- mice::mice(wdbcNA, m=5, pred=pred, maxit=5, seed=1234, print=FALSE)
obj_imp <- createClustAll(data=wdbcNA, dataImputed = imp)
```

`dataImputed` *Retrieve Imputed Data from ClustAllObject*

Description

This method extracts and returns the imputed datasets stored in a ClustAllObject. It provides access to the multiple imputed versions of the data generated to handle missing values, if imputation was conducted and included in the object.

Usage

```
dataImputed(Object)
```

Arguments

`Object`	A `ClustAllObject-class` object created by `createClustAll`.

Details

The dataImputed method provides access to the imputed data used in the ClustALL analysis pipeline when missing values are present. Key aspects of this data include:

1. Imputation Structure:
   - Returns a 'mids' object, which contains multiple imputed datasets.
   - Each imputed dataset is a complete version of the original data with missing values filled in.
   - The number of imputed datasets corresponds to the 'nImputation' parameter.
2. Imputation Method:
   - Imputation is performed using the mice (Multivariate Imputation by Chained Equations) package.
   - The specific imputation methods used for each variable can be examined in the returned 'mids' object.
3. Data Consistency:
   - The imputed datasets maintain the same structure (rows and columns) as the original data.
   - Only variables with missing values are imputed; complete variables remain unchanged.

#'

Value

A 'mids' object from the mice package containing the imputed datasets. If no imputation was performed, the method returns NULL.

See Also

`createClustAll`, `showData`, `dataOriginal`, `ClustAllObject-class`,

Examples

```
data("BreastCancerWisconsinMISSING", package = "ClustAll")
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)
obj_NA <- createClustAll(data = wdbcNA, colValidation = "Diagnosis",
                         dataImputed = wdbcMIDS)
dataImputed(obj_NA)
```

`dataOriginal` *Retrieve Original Unprocessed Data from ClustAllObject*

Description

This method extracts and returns the original, unmodified data that was used to create the ClustAllObject. It provides access to the raw input data before any preprocessing steps were applied, including one-hot encoding, validation column removal, or imputation.

Usage

```
dataOriginal(Object)
```

Arguments

`Object`	A `ClustAllObject-class` object created by `createClustAll`.

Details

The dataOriginal method serves as a reference point for the initial state of the data in the ClustALL analysis pipeline. Key aspects of this data include:

1. Data Integrity:
   - Contains all original variables, including any that may have been removed or transformed during preprocessing.
   - Preserves original data types (e.g., factors for categorical variables).
   - Includes the validation column if it was present in the original data.
2. Missing Data:
   - Reflects the original state of missing values (NAs) in the dataset.
3. Data Structure:
   - Maintains the original row and column order of the input data.
   - No transformations or encodings have been applied.

Value

A data frame containing the original, unprocessed input data exactly as it was provided when creating the ClustAllObject.

See Also

`createClustAll`, `showData`, `dataImputed`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)
obj_noNA <- createClustAll(data = wdbc, colValidation = "Diagnosis")
dataOriginal(obj_noNA)
```

---

dataValidation *Retrieve Validation Data from ClustAllObject*

Description

This method extracts and returns the validation data (true labels) stored in a ClustAllObject. These labels represent known classifications or groupings of samples, which can be used to assess the performance of clustering results.

Usage

```
dataValidation(Object)
```

Arguments

Object	A ClustAllObject-class object created by createClustAll or modified by addValidationData.

Details

The dataValidation method provides access to the ground truth or reference classifications for the samples in the dataset:

1. Validation Data Content:
   - Contains known classifications or groupings of samples.
   - Typically represents biological, clinical, or other meaningful categorizations.
   - Used as benchmarking to evaluate clustering performance.
2. Data Characteristics:
   - Returned as a numeric vector.
   - Length matches the number of samples in the original dataset.
   - Each element corresponds to a sample's true label or classification.
3. Availability and Source:
   - May be added during object creation via the colValidation parameter in createClustAll.
   - Can be added later using the addValidationData function.
   - If not available, the method returns NULL.

Value

A numeric vector containing the validation data (true labels) if available. Returns NULL if no validation data has been added to the object.

Note

- This method returns the data stored in the 'dataValidation' slot of the ClustAllObject.
- The validation data is not used in the clustering process itself; it is solely for evaluation purposes.
- If validation data was not provided during object creation or added later, this method will return NULL.
- Always check for NULL before using the returned data in calculations or visualizations.
- The interpretation and use of validation data depend on the specific context of your study.

See Also

createClustAll, addValidationData, validateStratification, ClustAllObject-class

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)
obj_noNA <- createClustAll(data = wdbc, colValidation="Diagnosis")
dataValidation(obj_noNA)
```

---

extractData *extractData*

Description

This function retrieves all data components stored in a ClustAllObject-class, including the preprocessed data, original input data, and imputed datasets (if applicable). It provides a comprehensive view of the data at different stages of the ClustALL pipeline.

Usage

```
extractData(Object)
```

Arguments

Object	A ClustAllObject-class object created by createClustAll.

Details

The extractData function provides access to data at various stages of the ClustALL pipeline:

- Data: Reflects preprocessing steps such as one-hot encoding for categorical variables and removal of the validation column.
- DataOriginal: The exact data as input to createClustAll, useful for reference and verifying preprocessing steps.
- Data_imputed: Multiple imputed datasets if missing data was present and imputation was performed.

This function is particularly useful for:

- Verifying preprocessing steps and their effects on the data.

- Accessing original data for additional analyses or visualizations.
- Examining imputed datasets to understand how missing data was handled.
- Exporting data at different stages for use in other analyses or packages.

Value

A list containing three elements:

- Data_modified: A data frame of the preprocessed data used for clustering.
- Data_original: A data frame of the original, unmodified input data.
- Data_imputed: A 'mids' object from the mice package containing imputed datasets, if imputation was performed. NULL otherwise.

See Also

createClustAll, ClustAllObject-class, runClustAll

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)

obj_noNA <- createClustAll(data = wdbc, colValidation = "Diagnosis")
extractData(obj_noNA)
```

---

extractResults *Extract Clustering Results from ClustAllObject*

Description

This function retrieves the complete set of clustering results from a processed ClustAllObject, including all generated stratifications and the subset of statistically robust stratifications. It provides comprehensive access to the outcomes of the ClustALL algorithm.

Usage

```
extractResults(Object)
```

Arguments

Object	A processed ClustAllObject-class object. The object must have been processed by runClustAll before using this function.

Details

The extractResults function provides comprehensive access to the clustering outcomes of the ClustALL algorithm:

- All_clusters: Contains every stratification generated, regardless of statistical robustness. Each element is a vector of cluster assignments for the samples.
- Robust_clusters: A subset of All_clusters, containing only those stratifications that passed the bootstrapping process for population-based robustness.

This function is particularly useful for:

- Comprehensive analysis of all generated clustering solutions.
- Comparing robust stratifications with non-robust ones.
- Exporting clustering results for further analysis in other tools.
- Assessing the impact of robustness criteria on stratification selection.

Value

A list containing two elements:

- All_clusters: A list of all stratifications generated by the ClustALL algorithm.
- Robust_clusters: A list of statistically robust stratifications that passed the bootstrapping process.

If `runClustAll` has not been executed, the function returns a list with a single NULL element.

Note

This function will return a list with a single NULL element if `runClustAll` has not been executed on the object. Always check if the returned list contains valid results before proceeding with analysis.

See Also

`runClustAll`, `summary_clusters`, `resStratification`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)

obj_noNA <- createClustAll(data = wdbc, colValidation = "Diagnosis")
obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
extractResults(obj_noNA1)
```

heart_data *Heart Disease Dataset*

Description

The dataset comprises both categorical and numerical features derived from medical examinations and patient history. Each row represents a patient, characterized by 13 attributes, with the target variable indicating the presence or absence of heart disease.

Usage

```
data("heart_data", package = "ClustAll")
```

Format

A data frame with 918 rows and 12 variables:

Age Numeric. Age of the patient in years.

Sex Categorical. Patient's gender (M = Male, F = Female).

ChestPainType Categorical. Type of chest pain experienced (TA = Typical Angina, ATA = Atypical Angina, NAP = Non-Anginal Pain, ASY = Asymptomatic).

RestingBP Numeric. Resting blood pressure in mm Hg.

Cholesterol Numeric. Serum cholesterol in mg/dl.

FastingBS Binary. Fasting blood sugar > 120 mg/dl (1 = true; 0 = false).

RestingECG Categorical. Resting electrocardiogram results (Normal, ST = having ST-T wave abnormality, LVH = showing probable or definite left ventricular hypertrophy by Estes' criteria).

MaxHR Numeric. Maximum heart rate achieved.

ExerciseAngina Categorical. Exercise-induced angina (Y = Yes, N = No).

Oldpeak Numeric. ST depression induced by exercise relative to rest.

ST_Slope Categorical. The slope of the peak exercise ST segment (Up, Flat, Down).

HeartDisease Binary. Output class (1 = heart disease, 0 = normal).

Details

This dataset contains various medical and lifestyle attributes of patients, along with their heart disease diagnosis status. It is commonly used for predicting the presence of heart disease in patients.

This dataset is valuable for developing and testing machine learning models for heart disease prediction. It includes a mix of demographic information, vital signs, and results from various medical tests, making it a comprehensive resource for studying factors associated with heart disease.

Value

heart dataset

Source

Kaggle: https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset

`initialize,ClustAllObject-method` *initializeClustAllObject*

Description

constuctor for `ClustAllObject-class`

Usage

```
## S4 method for signature 'ClustAllObject'
initialize(
  .Object,
  data,
  dataOriginal,
  dataImputed,
  dataValidation,
  nImputation,
  processed,
  summary_clusters,
  JACCARD_DISTANCE_F
)
```

Arguments

`.Object`	initializing object
`data`	Data Frame of the input data after applying one-hot encoding to the categorical variables and extracting the validation (true label) column.
`dataOriginal`	Data Frame of the input data.
`dataImputed`	Mids object derived from the mice package that stores the imputed data, in case imputation was applied. Otherwise NULL.
`dataValidation`	A vector in the case there is a validation (true label) column in the input data. This information can be added later with `addValidationData`. Otherwise NULL.
`nImputation`	Number of imputations performed.
`processed`	A boolean. TRUE if `runClustAll` has been executed. Otherwise FALSE.
`summary_clusters`	List with the resulting stratifications for each combination of clustering methods (distance + clustering algorithm) and depth, in case `runClustAll` has been executed previously. Otherwise NULL.
`JACCARD_DISTANCE_F`	Matrix containing the Jaccard distances derived from the robust stratifications after applying the bootstrapping if `runClustAll` has been executed previously. Otherwise NULL.

Value

ClustAllObject class object.

JACCARD_DISTANCE_F *Retrieve Jaccard Distance Matrix for Robust Stratifications*

Description

This method extracts and returns the matrix of Jaccard distances between robust stratifications identified by the ClustALL algorithm. It provides a quantitative measure of similarity between different clustering solutions that have passed the bootstrapping process for population-based robustness.

Usage

```
JACCARD_DISTANCE_F(Object)
```

Arguments

Object	A processed ClustAllObject-class object. The object must have been processed by runClustAll before using this method.

Details

The JACCARD_DISTANCE_F method provides crucial information about the similarity structure of robust clustering solutions:

1. Jaccard Distance:
   - A measure of dissimilarity between sample sets, calculated as 1 minus the Jaccard coefficient.
   - Ranges from 0 (identical stratifications) to 1 (completely different stratifications).
   - Lower values indicate higher similarity between stratifications.
2. Matrix Structure:
   - Symmetric matrix with stratification names as row and column labels.
   - Diagonal elements are always 0 (each stratification is identical to itself).
   - Off-diagonal elements represent pairwise Jaccard distances.
3. Robust Stratifications:
   - Only includes stratifications that passed the bootstrapping process.
   - Represents the most stable and reliable clustering solutions.

This method is particularly useful for:

- Identifying groups of similar stratifications.
- Assessing the diversity of robust clustering solutions.
- Selecting representative stratifications for further analysis.
- Visualizing the relationships between different clustering outcomes.
- Input for further clustering or dimensionality reduction of stratifications.

Value

A square matrix where each element represents the Jaccard distance between two robust stratifications. The row and column names correspond to the names of the robust stratifications. Returns NULL if runClustAll has not been executed on the object.

Note

- This method returns the data stored in the 'JACCARD_DISTANCE_F' slot of the ClustAllObject.
- It will return NULL if `runClustAll` has not been executed on the object.
- The Jaccard distance is calculated based on the co-occurrence of samples in clusters, not on the specific cluster labels.
- This matrix is often used as input for the `plotJACCARD` function to visualize the similarity structure of stratifications.

See Also

`runClustAll`, `plotJACCARD`, `resStratification`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)
obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = FALSE)
JACCARD_DISTANCE_F(obj_noNA1)
```

`listOrNULL-class` *Class Union: listOrNULL*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a list, NULL, or a missing value. It is particularly useful when a slot or function parameter might contain a list of elements but could also be empty or unspecified.

Details

The listOrNULL class union includes:

- list: A standard R list object
- NULL: Representing an empty or unset value
- missing: Allowing for unspecified parameters in function calls

This union is useful in scenarios where:

- A function might return a list of results or NULL if no results are available
- A slot in an S4 object could contain a list of elements or be empty
- A function parameter could accept a list of options, but also work with default settings if nothing is provided

See Also

`setClassUnion`, `ClustAllObject-class`

logicalOrNA-class *Class Union: logicalOrNA*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a logical value, NULL, or a missing value. It is particularly useful when a slot or function parameter might contain a boolean flag but could also be empty, unspecified, or explicitly set to NULL.

Details

The logicalOrNA class union includes:

- logical: A standard R logical value (TRUE or FALSE)
- NULL: Representing an empty or unset value
- missing: Allowing for unspecified parameters in function calls

This union is useful in scenarios where:

- A function might return a logical result or NULL if no result is available
- A slot in an S4 object could contain a logical flag or be empty
- A function parameter could accept a logical input, but also work with default settings if nothing is provided

See Also

setClassUnion, ClustAllObject-class

matrixOrNULL-class *Class Union: matrixOrNULL*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a matrix or NULL. It is particularly useful when a slot or function parameter might contain a matrix of data but could also be empty or explicitly set to NULL.

Details

The matrixOrNULL class union includes:

- matrix: A standard R matrix object
- NULL: Representing an empty or unset value

This union is useful in scenarios where:

- A function might return a matrix of results or NULL if no results are available
- A slot in an S4 object could contain a matrix of data or be empty
- A function parameter could accept a matrix input, but also work with default settings if nothing is provided

See Also

setClassUnion, ClustAllObject-class

---

nImputation *Retrieve Number of Imputations from ClustAllObject*

Description

This method returns the number of imputations performed when creating or processing the ClustAllObject. It provides information about how missing data was handled in the ClustALL pipeline.

Usage

```
nImputation(Object)
```

Arguments

Object	A ClustAllObject-class object created by createClustAll.

Details

The nImputation method provides insight into the multiple imputation strategy used in the ClustALL analysis pipeline:

This method is particularly useful for:

- Verifying whether imputation was performed on the dataset.
- Understanding the extent of the imputation process.
- Assessing the potential impact of imputation on subsequent analyses.
- Reporting the methodology used in handling missing data.

Value

An integer indicating the number of imputations performed. Returns 0 if no imputations were required or performed.

Note

- This method returns the value stored in the 'nImputation' slot of the ClustAllObject.
- A return value of 0 does not necessarily mean the original data had no missing values; it could also indicate that imputation was explicitly skipped.
- The number of imputations is typically set during the creation of the ClustAllObject with the createClustAll function.
- For accessing the actual imputed datasets, use the dataImputed method.

#'

See Also

createClustAll, dataImputed, ClustAllObject-class, mice

Examples

```
data("BreastCancerWisconsinMISSING", package = "ClustAll")
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)
obj_NA <- createClustAll(data = wdbcNA, colValidation = "Diagnosis",
                         dataImputed = wdbcMIDS)
nImputation(obj_NA)
```

---

numericOrCharacter-class *Class Union: numericOrCharacter*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a numeric or character value. It is particularly useful when a slot or function parameter might contain either numeric data or character strings representing numbers or categories.

Details

The numericOrCharacter class union includes:

- numeric: A standard R numeric value or vector
- character: A standard R character string or vector

This union is useful in scenarios where:

- A function might accept or return either numeric values or their string representations
- A slot in an S4 object could contain either numeric data or categorical labels
- A function needs to handle both numeric and character input flexibly

See Also

setClassUnion, ClustAllObject-class

---

numericOrNA-class *Class Union: numericOrNA*

Description

This class union allows for flexibility in method signatures and slot definitions by accepting either a numeric value, NULL, or a missing value. It is particularly useful when a slot or function parameter might contain a numeric value but could also be empty, unspecified, or explicitly set to NULL.

Details

The numericOrNA class union includes:

- numeric: A standard R numeric value or vector
- NULL: Representing an empty or unset value
- missing: Allowing for unspecified parameters in function calls

This union is useful in scenarios where:

- A function might return a numeric result or NULL if no result is available
- A slot in an S4 object could contain a numeric value or be empty
- A function parameter could accept a numeric input, but also work with default settings if nothing is provided

See Also

setClassUnion, ClustAllObject-class

---

obj_noNA1 *obj_noNA1: Processed wdbc dataset for testing purposed*

Description

The "obj_noNA1" dataset is a processed version of the Breast Cancer Wisconsin (Diagnostic) dataset (wdbc), prepared for testing purposes and used in the ClustAll package vignette. It has been processed using the ClustAll package methods, which include data preprocessing, feature transformation, and the application of clustering algorithms.

Usage

```
data("testData", package = "ClustAll")
```

Format

A processed ClustAllObject

Details

The "obj_noNA1" dataset can be used as a reference for users who want to understand the ClustAll package workflow and replicate the analysis presented in the vignette.

For more information on the original dataset, please refer to the documentation of the "wdbc" (wdbc) dataset.

Value

ClustAllObject Object

obj_noNA1simplify *obj_noNA1simplify: Processed wdbc dataset for testing purposed*

Description

The "obj_noNA1simplify" dataset is a processed version of the Breast Cancer Wisconsin (Diagnostic) dataset (wdbc), prepared for testing purposes and used in the ClustAll package vignette. It has been processed using the ClustAll package methods with the 'simplify' parameter set to TRUE, which reduces the computational complexity of the clustering algorithms by considering only a subset of the dendrogram depths.

Usage

```
data("testData", package = "ClustAll")
```

Format

A processed ClustAllObject

Details

The "obj_noNA1simplify" dataset can be used as a reference for users who want to understand the ClustAll package workflow and replicate the simplified analysis presented in the vignette.

For more information on the original dataset and the full processed version, please refer to the documentation of the "wdbc" (wdbc).

Value

ClustAllObject Object

obj_noNAno1Validation *obj_noNAno1Validation: Processed wdbc dataset for testing purposed*

Description

The "obj_noNAno1Validation" dataset is a processed version of the Breast Cancer Wisconsin (Diagnostic) dataset (wdbc), prepared for testing purposes and used in the ClustAll package vignette. It has been processed using the ClustAll package methods, but with the validation data removed.

Usage

```
data("testData", package = "ClustAll")
```

Format

A processed ClustAllObject

Details

The "obj_noNAno1Validation" dataset can be used as a reference for users who want to understand the ClustAll package workflow and replicate the analysis presented in the vignette, focusing on unsupervised learning aspects.

For more information on the original dataset and the processed version with validation data, please refer to the documentation of the "wdbc" (`wdbc`).

Value

ClustAllObject Object

`plotJACCARD` *Visualize Jaccard Distances Between Robust Stratifications*

Description

This function generates a heatmap visualization of the Jaccard distances between robust stratifications identified by the ClustALL algorithm. It provides a visual representation of the similarity between different clustering solutions, helping to identify groups of related stratifications.

Usage

```
plotJACCARD(Object, paint = TRUE, stratification_similarity = 0.7)
```

Arguments

`Object`	A processed `ClustAllObject-class` object. The object must have been processed by `runClustAll` before using this function.
`paint`	Logical. If TRUE (default), the function highlights groups of similar stratifications on the heatmap with red squares. This helps in visually identifying clusters of similar stratifications.
`stratification_similarity`	Numeric value between 0 and 1. Sets the threshold Jaccard distance for considering two stratifications as similar. Default is 0.7. Higher values result in more stringent similarity criteria.

Details

The plotJACCARD function visualizes the similarity between robust stratifications using a heatmap of Jaccard distances:

- The heatmap color scale represents Jaccard distances, with darker colors indicating higher similarity (lower distance).
- Stratifications are ordered based on hierarchical clustering of their Jaccard distances.
- The 'paint' option highlights groups of similar stratifications, making it easier to identify clusters of related solutions.
- The 'stratification_similarity' parameter allows fine-tuning of what is considered "similar" for the purpose of highlighting.

The function provides annotations for each stratification, including:

- Distance metric used (e.g., Correlation, Gower)
- Clustering method employed (e.g., H-Clustering, K-Means, K-Medoids)
- Depth of the dendrogram cut used in the Data Complexity Reduction step

This visualization is particularly useful for:

- Identifying groups of similar stratifications
- Assessing the overall diversity of robust solutions
- Guiding the selection of representative stratifications for further analysis

Value

A plot displaying a correlation matrix heatmap that shows the Jaccard Distances between population-based robust stratifications. The heatmap visually distinguishes groups of similar stratifications according to the specified "stratification_similarity" threshold.

Note

- This function requires a processed ClustAllObject. Ensure `runClustAll` has been executed before using plotJACCARD.
- The 'paint' feature may not be visible if there are no groups of stratifications meeting the similarity threshold.
- For exploring stratifications, it's recommended to start with a high 'stratification_similarity' value and gradually decrease it to examine various levels of stratification grouping.

See Also

`runClustAll`, `resStratification`, `ClustAllObject-class`, `JACCARD_DISTANCE_F`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)

obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
plotJACCARD(obj_noNA1, paint = TRUE, stratification_similarity = 0.9)
```

---

plotSANKEY *Visualize Stratification Comparisons with Sankey Diagram*

Description

This function generates a Sankey diagram to visualize the relationships between different stratifications or between a stratification and the true labels (if available). It provides an intuitive representation of how samples are distributed across clusters in different stratifications or how they align with known stratifications.

Usage

```
plotSANKEY(Object, clusters, validationData = FALSE)
```

Arguments

`Object`	A processed `ClustAllObject-class` object. The object must have been processed by `runClustAll` before using this function.
`clusters`	A character vector specifying the names of stratifications to compare. If validationData is FALSE, exactly two stratification names should be provided. If validationData is TRUE, provide one stratification name to compare with true labels.
`validationData`	Logical. If TRUE, compares the specified stratification with the true labels (validation data) if available. Default is FALSE.

Details

The plotSANKEY function provides a powerful visualization tool for understanding the relationships between different clustering solutions or between a clustering solution and known classifications:

1. Stratification Comparison (validationData = FALSE):
   - Visualizes how samples are distributed across clusters in two different stratifications.
   - Helps identify similarities and differences between clustering solutions.
   - Useful for understanding how changes in clustering parameters affect sample groupings.
2. Validation Comparison (validationData = TRUE):
   - Compares a single stratification with true labels (if available).
   - Helps assess how well the clustering aligns with known stratifications.
   - Useful for evaluating the biological or clinical relevance of a clustering solution.

The Sankey diagram represents:

- Clusters (or true labels) as nodes on the left and right sides of the diagram.
- Flows between nodes indicating how samples are distributed.
- The width of each flow is proportional to the number of samples it represents.

This visualization is particularly useful for:

- Identifying stable sample groupings across different stratifications.
- Detecting major shifts in cluster assignments between solutions.
- Evaluating the concordance between clustering results and known stratifications.
- Understanding the impact of different clustering approaches on sample groupings.

Value

A Sankey plot showing the composition and flow of clusters between the selected stratifications. If true labels are available and "validationData" is TRUE, the plot will compare the selected stratification against the true labels.

Note

- This function requires a processed ClustAllObject. Ensure runClustAll has been executed before using plotSANKEY.
- When validationData is TRUE, the ClustAllObject must contain validation data (true labels). This can be added using addValidationData if not provided during object creation.
- Stratification names can be obtained from the results of resStratification or by examining the names in the summary_clusters slot of the ClustAllObject.
- The Sankey diagram may become cluttered if there are many clusters or if the clustering solutions are very different. In such cases, consider focusing on specific subsets of clusters or using additional filtering criteria.

See Also

runClustAll, resStratification, addValidationData, ClustAllObject-class

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
label <- as.numeric(as.factor(wdbc$Diagnosis))
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
label <- label[16:30]
obj_noNA <- createClustAll(data = wdbc)

obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
resStratification(Object = obj_noNA1, population = 0.05,
                  stratification_similarity = 0.88, all = FALSE)
plotSANKEY(Object = obj_noNA1, clusters = c("cuts_a_1","cuts_b_5"))

obj_noNA1 <- addValidationData(obj_noNA1, label)
plotSANKEY(Object = obj_noNA1, clusters = "cuts_a_1", validationData=TRUE)
```

---

processed *Check Processing Status of ClustAllObject*

Description

This method retrieves the processing status of a ClustAllObject, indicating whether the ClustALL algorithm has been executed on the object. It provides a quick way to verify if clustering results are available for analysis.

Usage

```
processed(Object)
```

Arguments

Object	A ClustAllObject-class object created by createClustAll.

Details

The processed method serves as a crucial indicator in the ClustALL workflow:

1. Processing Status:
   - Indicates whether the ClustALL algorithm has been applied to the object.
   - A TRUE value means clustering results are available for analysis.
   - A FALSE value indicates the object only contains input data and preprocessing.
2. Workflow Implications:
   - Helps determine which methods and analyses can be performed on the object.
   - Guides users in the correct sequence of operations in the ClustALL pipeline.
3. Data Availability:
   - TRUE status implies availability of:
     - Stratification results (`summary_clusters`)
     - Jaccard distance matrix (`JACCARD_DISTANCE_F`)
     - Other clustering-related outputs
   - FALSE status means only input and preprocessed data are available.

Value

A logical value:

- TRUE if `runClustAll` has been executed on the object.
- FALSE if the object has not yet been processed by `runClustAll`.

Note

- This method checks the 'processed' slot of the ClustAllObject.
- The processed status is automatically set to TRUE when `runClustAll` completes successfully.
- Users should not manually modify this status to ensure consistency between the status and the actual content of the object.
- A FALSE status does not necessarily indicate an error; it simply means `runClustAll` needs to be executed before accessing clustering results.

See Also

`runClustAll`, `createClustAll`, `ClustAllObject-class`, `summary_clusters`, `JACCARD_DISTANCE_F`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)
processed(obj_noNA)
```

`resStratification` *Extract Representative Stratifications from ClustAllObject*

Description

This function retrieves and filters representative stratifications from a processed ClustAllObject. It allows users to explore the most robust and significant clustering solutions generated by the ClustALL algorithm, based on population size criteria and stratification similarity.

Usage

```
resStratification(Object, population = 0.05, all = FALSE,
                          stratification_similarity = 0.7)
```

Arguments

`Object`	A processed `ClustAllObject-class` object. The object must have been processed by `runClustAll` before using this function.
`population`	Numeric value between 0 and 1. Specifies the minimum proportion of the total population that a cluster within a stratification must contain to be considered valid. Default is 0.05.
`all`	A logical value. If set to TRUE, the function will return all stratification representatives that meet the robustness criteria of each group of similar stratifications. If set to FALSE, only the centroid stratification (the central that serves as the representative stratification) for each group of similar stratifications will be returned.
`stratification_similarity`	Numeric value between 0 and 1. Sets the threshold Jaccard distance for considering two stratifications as similar. Default is 0.7. Higher values result in more stringent similarity criteria.

Details

The resStratification function performs several key steps:

1. Filters stratifications based on the 'population' parameter, ensuring that each cluster in a stratification contains at least the specified proportion of the total population.
2. Groups similar stratifications based on their Jaccard distances, using the 'stratification_similarity' threshold.
3. For each group of similar stratifications:
    - If all = FALSE, selects the centroid (most representative) stratification.
    - If all = TRUE, includes all stratifications in the group.

This function is particularly useful for:

- Identifying the most robust and significant clustering solutions.
- Reducing the number of stratifications to a manageable set for further analysis.
- Exploring how different similarity thresholds affect the grouping of stratifications.
- Comparing multiple similar stratifications within each group (when all = TRUE).

Value

A list of representative stratifications and their associated clusters. The structure of the return value depends on the 'all' parameter:

- If all = FALSE: A named list where each element represents the centroid stratification for a group of similar stratifications.
- If all = TRUE: A nested list where each top-level element represents a group of similar stratifications, and contains all stratifications in that group.

Each stratification is represented by a table showing the distribution of patients across clusters.

Note

- This function requires a processed ClustAllObject. Ensure `runClustAll` has been executed before using resStratification.
- The 'population' parameter helps filter out stratifications with very small, potentially insignificant clusters.
- The 'stratification_similarity' parameter allows for fine-tuning the balance between diversity and similarity in the returned stratifications.
- When exploring results, it's often useful to try different combinations of 'population' and 'stratification_similarity' values to understand the characteristics of your clustering solutions.

See Also

`runClustAll`, `plotJACCARD`, `cluster2data`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)

obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
resStratification(Object = obj_noNA1, population = 0.05,
                  stratification_similarity = 0.88, all = FALSE)
```

---

runClustAll *Execute ClustALL Algorithm on a ClustAllObject*

Description

This function applies the ClustALL algorithm to the data stored in a ClustAllObject. It performs a comprehensive analysis, generating multiple patient stratifications based on various clustering methods and parameters. The function implements the core steps of the ClustALL framework: Data Complexity Reduction (DCR), Stratification Process (SP), and Consensus-based Stratifications (CbS).

Usage

```
runClustAll(Object, threads = 1, simplify = FALSE)
```

Arguments

Object	An unprocessed ClustAllObject-class created by createClustAll. The object should contain the original input data and any preprocessed versions of it, but no stratification results yet.
threads	An integer specifying the number of CPU cores to use for parallel computing. Default is 1 (single-core processing).
simplify	A logical value. If TRUE, only every fourth depth of the dendrogram will be considered in the DCR and SP steps, reducing execution time but potentially sacrificing detail. Default is FALSE.

Details

The runClustAll function implements the three main steps of the ClustALL framework:

1. Data Complexity Reduction (DCR):
   - Creates multiple data embeddings to replace highly correlated variable sets with lower-dimension projections derived from Principal Component Analysis (PCA).
   - Explores all relevant groupings derived from a hierarchical clustering-based dendrogram.
   - Computes an embedding for each depth in the dendrogram.
2. Stratification Process (SP):
   - Calculates and preliminarily evaluates stratifications for each embedding.
   - Computes a stratification for each feasible combination of embedding, dissimilarity metric, and clustering method.
   - Determines the optimal number of clusters using three internal validation measures: sum-of-squares (WB-ratio), Dunn index, and average silhouette width.
3. Consensus-based Stratifications (CbS):
   - Filters out non-robust stratifications through bootstrapping.
   - Excludes stratifications with less than 85
   - Selects representatives of very similar outcomes from the remaining stratifications.

The 'simplify' parameter, when set to TRUE, reduces computation time by considering only every fourth depth of the dendrogram in the first and second steps. While useful for preliminary results, it's recommended to set it to FALSE for more robust and detailed results.

Value

A processed ClustAllObject-class containing stratification results from the ClustALL pipeline. The object's summary_clusters and JACCARD_DISTANCE_F slots will be populated with results.

Note

- This function modifies the input ClustAllObject in-place, populating it with stratification results.
- The process can be computationally intensive, especially for large datasets or when 'simplify' is set to FALSE.
- Consider using multiple threads for improved performance on multi-core systems.

See Also

createClustAll, resStratification, plotJACCARD, cluster2data, ClustAllObject-class

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]

obj_noNA <- createClustAll(data = wdbc)
obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
obj_noNA1
```

---

show,ClustAllObject-method

Display Summary of ClustAllObject

Description

This method provides a concise summary of a ClustAllObject, displaying key information about its contents and processing status. It offers a quick overview of the object's characteristics without the need to inspect individual slots.

Usage

```
## S4 method for signature 'ClustAllObject'
show(object)
```

Arguments

object	A ClustAllObject-class object to be summarized.

Details

The show method for ClustAllObject displays the following information:

1. Object Class: Confirms that the object is of class ClustAllObject.
2. Data Dimensions:
   - Number of variables (columns) in the processed data.
   - Number of patients (rows) in the dataset.
3. Imputation Status:
   - Indicates whether the data has been imputed.
   - If imputed, shows the number of imputations performed.
4. Processing Status:
   - Indicates whether the ClustALL algorithm has been run on the object.
5. Stratification Results:
   - If processed, displays the number of stratifications generated.
   - If not processed, indicates that stratification results are not available.

This method is particularly useful for:

- Quick verification of object contents after creation or modification.
- Checking the processing status before running analyses.
- Confirming the number of stratifications generated after running the ClustALL algorithm.
- Easily sharing key object characteristics in reports or discussions.

Value

No return value, called for printing to the console.

Note

- This method is automatically called when the object name is entered at the R console.
- It provides a high-level overview and does not display detailed data or results.
- For more detailed information about specific aspects of the object, use dedicated accessor methods or examine individual slots directly.

See Also

`createClustAll`, `runClustAll`, `ClustAllObject-class`

---

showData *Retrieve Processed Data from ClustAllObject*

Description

This method extracts and returns the processed data stored in a ClustAllObject. The data returned is the version used for clustering analysis, which has undergone preprocessing steps such as one-hot encoding for categorical variables and removal of the validation column (if present).

Usage

```
showData(Object)
```

Arguments

`Object`	A `ClustAllObject-class` object created by `createClustAll`.

Details

The showData method provides access to the core dataset used in the ClustALL analysis pipeline. Key aspects of this data include:

1. Preprocessing Applied:
   - Categorical variables have been converted to numeric form via one-hot encoding.
   - The validation column (if specified during object creation) has been removed.
2. Data Structure:
   - All columns are numeric, suitable for use in clustering algorithms.
   - Row order corresponds to the original input data.

3. Missing Data:
    - The returned data may still contain missing values (NAs) if imputation was not performed.
    - For imputed versions of the data, use the `dataImputed` method.

Value

A data frame containing the processed data used for clustering analysis. This data reflects all preprocessing steps applied during object creation but does not include any imputation that may have been performed.

See Also

`createClustAll`, `dataOriginal`, `dataImputed`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=-ID)
obj_noNA <- createClustAll(data = wdbc, colValidation = "Diagnosis")
showData(obj_noNA)
```

---

summary_clusters *Retrieve Summary of Clustering Results from ClustAllObject*

Description

This method extracts and returns a comprehensive summary of all clustering results (stratifications) generated by the ClustALL algorithm. It provides access to the complete set of clustering solutions, including both robust and non-robust stratifications.

Usage

```
summary_clusters(Object)
```

Arguments

`Object`	A processed `ClustAllObject-class` object. The object must have been processed by `runClustAll` before using this method.

Details

The summary_clusters method provides access to all clustering results generated during the ClustALL analysis:

1. Comprehensive Results:
    - Includes all stratifications generated, regardless of their robustness.
    - Each stratification represents a unique combination of data embedding, distance metric, clustering algorithm, and number of clusters.
2. Stratification Structure:

- Each element in the returned list is named according to the parameters used to generate it (e.g., "cuts_a_1" for the first cut using method 'a').
- Each stratification is a vector of integers, where each integer represents a cluster assignment for a sample.

3. Analysis Possibilities:
   - Allows for comparison of different clustering solutions.
   - Enables examination of how different algorithm parameters affect clustering outcomes.
   - Facilitates identification of consistent patterns across multiple stratifications.

This method is particularly useful for:

- Accessing the full range of clustering solutions for in-depth analysis.
- Comparing robust stratifications (identified by other methods) with the full set of results.
- Extracting specific stratifications for further analysis or visualization.

Value

A list where each element represents a stratification. Each stratification is a vector of cluster assignments for each sample in the dataset. Returns NULL if `runClustAll` has not been executed on the object.

Note

- This method returns the data stored in the 'summary_clusters' slot of the ClustAllObject.
- It will return NULL if `runClustAll` has not been executed on the object.
- The returned list includes all stratifications, not just those deemed robust. For accessing only robust stratifications, consider using the `resStratification` function.
- The order and naming of stratifications in the returned list correspond to the order in which they were generated during the ClustALL process.

See Also

`runClustAll`, `resStratification`, `JACCARD_DISTANCE_F`, `ClustAllObject-class`

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
obj_noNA <- createClustAll(data = wdbc)
obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = FALSE)
summary_clusters(obj_noNA1)
```

validateStratification *Validate Stratification Results Against True Labels*

Description

This function calculates the sensitivity and specificity of a selected stratification by comparing it to the true labels (validation data) stored in the ClustAllObject. It provides a quantitative assessment of how well the clustering aligns with known classifications.

Usage

```
validateStratification(Object, stratificationName)
```

Arguments

Object	A processed ClustAllObject-class object. The object must have been processed by runClustAll and contain validation data (true labels).
stratificationName	A character string specifying the name of the stratification to be validated. This should correspond to a stratification generated by the ClustALL algorithm and stored in the Object.

Details

The validateStratification function provides a crucial step in assessing the biological or clinical relevance of clustering results:

1. Comparison Mechanism:
   - The function compares the cluster assignments of the selected stratification with the true labels provided in the validation data.
   - It treats the problem as a binary classification task, considering one class as the "positive" class and all others as "negative".
2. Sensitivity (True Positive Rate):
   - Measures the proportion of actual positive cases that were correctly identified.
   - Calculated as: (True Positives) / (True Positives + False Negatives)
3. Specificity (True Negative Rate):
   - Measures the proportion of actual negative cases that were correctly identified.
   - Calculated as: (True Negatives) / (True Negatives + False Positives)
4. Interpretation:
   - Higher sensitivity indicates better identification of the positive class.
   - Higher specificity indicates better identification of the negative classes.
   - The function automatically adjusts calculations if necessary to ensure sensitivity and specificity are always higher than 0.5.

This function is particularly useful for:

- Evaluating the clinical or biological relevance of clustering solutions
- Comparing different stratifications based on their alignment with known classifications
- Identifying stratifications that best capture known groupings in the data
- Providing quantitative metrics to support the selection of optimal clustering solutions

Value

A named numeric vector containing two elements:

- sensitivity: The proportion of true positive classifications
- specificity: The proportion of true negative classifications

Note

- This function requires a processed ClustAllObject with validation data. Ensure runClustAll has been executed and validation data has been added using addValidationData if not provided during object creation.
- The function assumes binary classification. For multi-class problems, it effectively treats one class as "positive" and all others as "negative".
- Stratification names can be obtained from the results of resStratification or by examining the names in the summary_clusters slot of the ClustAllObject.
- The function will stop with an error if the ClustAllObject does not contain validation data or if the specified stratification name is not found.

See Also

runClustAll, resStratification, addValidationData, plotSANKEY, ClustAllObject-class

Examples

```
data("BreastCancerWisconsin", package = "ClustAll")
label <- as.numeric(as.factor(wdbc$Diagnosis))
wdbc <- subset(wdbc,select=c(-ID, -Diagnosis))
wdbc <- wdbc[1:15,1:8]
label <- label[16:30]
obj_noNA <- createClustAll(data = wdbc)

obj_noNA1 <- runClustAll(Object = obj_noNA, threads = 1, simplify = TRUE)
resStratification(Object = obj_noNA1, population = 0.05,
                  stratification_similarity = 0.88, all = FALSE)
obj_noNA1 <- addValidationData(Object = obj_noNA1,
                               dataValidation = label)
validateStratification(obj_noNA1, "cuts_a_1")
```

---

wdbc *wdbc: Diagnostic Wisconsin Breast Cancer Database.*

Description

The Breast Cancer Wisconsin (Diagnostic) dataset, also known as "wdbc", contains features computed from digitized images of fine needle aspirates (FNA) of breast masses. The features describe characteristics of the cell nuclei present in each image.

Usage

```
data("BreastCancerWisconsin", package = "ClustAll")
```

Format

A data frame with 660 rows and 31 variables

Details

The dataset includes 569 patients, each characterized by 30 numerical features and one categorical feature. The numerical features are computed from ten different measurements of the cell nuclei, and each measurement is repeated three times, resulting in a total of 30 features per patient (10 features x 3 measurements).

The categorical feature is the diagnosis, which indicates whether the tumor is malignant (M) or benign (B). This feature serves as the target class for classification tasks.

Value

wdbc dataset

Source

<https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic>

- Diagnosis Label says tumor is malingnant or benignant
- radius. Mean of distances from the center to points on the perimeter
- perimeter
- area
- smoothness. Local variation in radius lengths
- compactness. (Perimeter^2 / Area) - 1.0
- concavity. Severity of concave portions of the contour
- concave points. Number of concave portions of the contour
- symmetry.
- fractal dimension. “Coastline approximation” - 1.

`wdbcMIDS` *wdbcMIDS: Diagnostic Wisconsin Breast Cancer Database with imputed values*

Description

The "wdbcMIDS" dataset is an imputed version of the "wdbcNA" dataset, which is a modified version of the Breast Cancer Wisconsin (Diagnostic) dataset (`wdbc`) with missing values introduced at random. The missing values have been imputed using the Multiple Imputation by Chained Equations (MICE) algorithm, resulting in a "mids" object manually.

Usage

```
data("BreastCancerWisconsinMISSING", package = "ClustAll")
```

Format

A data frame with 660 rows and 31 variables

Details

For more information on the original dataset and the version with missing values, please refer to the documentation of the "wdbc" (wdbc).

Value

wdbcMIDS dataset

---

wdbcNA *wdbcNA: Diagnostic Wisconsin Breast Cancer Database with missing values*

Description

The "wdbcNA" dataset is a modified version of the Breast Cancer Wisconsin (Diagnostic) dataset (wdbc), incorporating missing values at random. This dataset is designed to demonstrate and evaluate the handling of missing data in the context of breast cancer diagnosis.

Usage

```
data("BreastCancerWisconsinMISSING", package = "ClustAll")
```

Format

A data frame with 660 rows and 31 variables

Details

The dataset retains the same structure as the original "wdbc" dataset, with 569 patients and 32 variables. However, some values in the numerical features have been randomly replaced with missing values (NA).

For a detailed description of the features and the original dataset, please refer to the documentation of the "wdbc" dataset (wdbc).

Value

wdbcNA dataset

Index