Package ‘omada’

May 29, 2024

Type Package

Title Machine learning tools for automated transcriptome clustering analysis

Version 1.6.0

Description Symptomatic heterogeneity in complex diseases reveals differences in molecular states that need to be investigated. However, selecting the numerous parameters of an exploratory clustering analysis in RNA profiling studies requires deep understanding of machine learning and extensive computational experimentation. Tools that assist with such decisions without prior field knowledge are nonexistent and further gene association analyses need to be performed independently. We have developed a suite of tools to automate these processes and make robust unsupervised clustering of transcriptomic data more accessible through automated machine learning based functions. The efficiency of each tool was tested with four datasets characterised by different expression signal strengths. Our toolkit’s decisions reflected the real number of stable partitions in datasets where the subgroups are discernible. Even in datasets with less clear biological distinctions, stable subgroups with different expression profiles and clinical associations were found.

Depends pdfCluster (>= 1.0-3), kernlab (>= 0.9-29), R (>= 4.2), fpc (>= 2.2-9), Repp (>= 1.0.7), diceR (>= 0.6.0), geplot2 (>= 3.3.5), reshape (>= 0.8.8), genieclust (>= 1.1.3), clValid (>= 0.7), glmnet (>= 4.1.3), dplyr (>= 1.0.7), stats (>= 4.1.2), clValid(>= 0.7)

Suggests rmarkdown, knitr, testthat

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.1

VignetteBuilder knitr

biocViews Software, Clustering, RNASeq, GeneExpression

LazyData true

git_url https://git.bioconductor.org/packages/omada
Contents

clusteringMethodSelection .................................................. 3
clusterVoting ................................................................. 4
feasibilityAnalysis ........................................................... 4
feasibilityAnalysisDataBased ............................................... 5
featureSelection ............................................................... 6
get_agreement_scores ......................................................... 6
get_average_feature_k_stabilities ........................................ 7
get_average_stabilities_per_k ............................................. 8
get_average_stability ......................................................... 8
get_cluster_memberships_k ................................................ 9
get_cluster_voting_k_votes ................................................ 9
get_cluster_voting_memberships ......................................... 10
get_cluster_voting_metric_votes ....................................... 10
get_cluster_voting_scores ................................................ 11
get_feature_selection_optimal_features ................................ 11
get_feature_selection_optimal_number_of_features .................. 12
get_feature_selection_scores .............................................. 13
get_generated_dataset ....................................................... 13
get_internal_metric_scores ............................................... 14
get_max_stability ............................................................ 14
get_metric_votes_k ......................................................... 15
get_optimal_features ....................................................... 15
get_optimal_memberships .................................................. 16
get_optimal_number_of_features ....................................... 16
get_optimal_parameter_used ............................................. 17
get_optimal Stability_score ............................................... 17
get_partition_agreement_scores ......................................... 18
get_sample_memberships ................................................... 19
get_vote_frequencies_k ..................................................... 19
omada ............................................................... 20
optimalClustering ............................................................ 20
partitionAgreement .......................................................... 21
plot_average_stabilities .................................................. 22
plot_cluster_voting ......................................................... 23
plot_feature_selection ................................................... 23
plot_partition_agreement ................................................ 24
clusteringMethodSelection

Method Selection through intra-method Consensus Partition Consistency

Description

Method Selection through intra-method Consensus Partition Consistency

Usage

clusteringMethodSelection(data, method.upper.k = 5, number.of.comparisons = 3)

Arguments

data A dataframe, where columns are features and rows are data points
method.upper.k The number of clusters, k, up to which the average agreements will be calculated
number.of.comparisons The number of comparisons to average over per k

Value

An object of class "methodSelection" containing a dataframe of partition agreement scores for a set of random parameters clustering runs across different methods and the corresponding plot

Examples

clusteringMethodSelection(toy_genes, method.upper.k = 3, number.of.comparisons = 2)
clusterVoting

Estimating number of clusters through internal exhaustive ensemble majority voting

Description
Estimating number of clusters through internal exhaustive ensemble majority voting

Usage
clusterVoting(data, min.k, max.k, algorithm)

Arguments
- **data**: A dataframe, where columns are features and rows are data points
- **min.k**: Minimum number of clusters for which we calculate stabilities
- **max.k**: Maximum number of clusters for which we calculate stabilities
- **algorithm**: The clustering algorithm to use for the multiple clustering runs to be measured

Value
An object of class "clusterVoting" containing a matrix with metric scores for every k and internal index, cluster memberships for every k, a dataframe with the k votes for every index, k vote frequencies and the frequency barplot of the k votes

Examples
clusterVoting(toy_genes, 4,14,"sc")

feasibilityAnalysis

Simulating dataset and calculate stabilities over different number of clusters

Description
Simulating dataset and calculate stabilities over different number of clusters

Usage
feasibilityAnalysis(classes = 3, samples = 320, features = 400)
feasibilityAnalysisDataBased

Arguments

- **classes**: The number of classes of samples to be reflected in the simulated dataset. Also determines the ks to be considered (classes-2, classes+2)
- **samples**: The number of samples in the simulated dataset
- **features**: The number of features in the simulated dataset

Value

An object of class “feasibilityAnalysis” containing the average stabilities for all number of clusters(k), the average (over all k) and maximum stabilities observed and the generated dataset

Examples

```
feasibilityAnalysis(classes = 2, samples = 20, features = 30)
```

feasibilityAnalysisDataBased

*Simulating dataset based on existing dataset’s dimensions, mean and standard deviation*

Description

Simulating dataset based on existing dataset’s dimensions, mean and standard deviation

Usage

```
feasibilityAnalysisDataBased(data, classes = 3)
```

Arguments

- **data**: The dataset to base the simulation extracting the number of samples, features and numeric
- **classes**: The number of classes of samples to be reflected in the simulated dataset. Also determines the ks to be considered (classes-2, classes+2)

Value

An object of class “feasibilityAnalysis” containing the average stabilities for all numbers of clusters(k), the average (over all k) and maximum stabilities observed and the generated dataset

Examples

```
feasibilityAnalysisDataBased(data = toy_genes, classes = 2)
```
get_agreement_scores

get_agreement_scores

Description

Get a dataframe of partition agreement scores for a set of random parameters clustering runs across different methods

Usage

get_agreement_scores(object)

Arguments

object An object of class "partitionAgreement"

featureSelection

Description

Predictor variable subsampling sets and bootstrapping stability set selection

Usage

featureSelection(data, min.k = 2, max.k = 4, step = 5)

Arguments

data A dataframe, where columns are features and rows are data points
min.k Minimum number of clusters for which we calculate stabilities
max.k Maximum number of clusters for which we calculate stabilities
step The number for additional features each feature set will contain

Value

An object of class "featureSelection" containing the dataframe of average bootstrap stabilities, where rows represent feature sets and columns number of clusters, the corresponding line plot, the number and the names of the selected features

Examples

featureSelection(toy_genes, min.k = 2, max.k = 4, step = 10)
get_average_feature_k_stabilities

Value

A dataframe of partition agreement scores for a set of random parameters clustering runs across different methods

Examples

pa.object <- partitionAgreement(toy_genes, algorithm.1 = "spectral", measure.1 = "rbfdot", algorithm.2 = "kmeans", measure.2 = "Lloyd", number.of.clusters = 3)
get_agreement_scores(pa.object)

get_average_feature_k_stabilities

Get a dataframe of average bootstrap stabilities

Description

Get a dataframe of average bootstrap stabilities

Usage

get_average_feature_k_stabilities(object)

Arguments

object An object of class "featureSelection"

Value

A dataframe of average bootstrap stabilities

Examples

fs.object <- featureSelection(toy_genes, min.k = 3, max.k = 4, step = 10)
get_average_feature_k_stabilities(fs.object)
get_average_stabilities_per_k

*Get average stabilities for all numbers of clusters (k)*

**Description**

Get average stabilities for all numbers of clusters (k)

**Usage**

`get_average_stabilities_per_k(object)`

**Arguments**

- `object` An object of class "feasibilityAnalysis"

**Value**

Average stabilities for all numbers of clusters (k)

**Examples**

```r
fa.object <- feasibilityAnalysis(classes = 2, samples = 10, features = 15)
average.sts.k <- get_average_stabilities_per_k(fa.object)
```

get_average_stability

*Get the average stability (over all k)*

**Description**

Get the average stability (over all k)

**Usage**

`get_average_stability(object)`

**Arguments**

- `object` An object of class "feasibilityAnalysis"

**Value**

The average stability (over all k)

**Examples**

```r
fa.object <- feasibilityAnalysis(classes = 2, samples = 10, features = 15)
average.st <- get_average_stability(fa.object)
```
**get_cluster_memberships_k**

*Get cluster memberships for every k*

**Description**
Get cluster memberships for every k

**Usage**

get_cluster_memberships_k(object)

**Arguments**

| object       | An object of class "clusterVoting" |

**Value**
Cluster memberships for every k

**Examples**

cv.object <- clusterVoting(toy_genes, 4,6,"sc")
get_cluster_memberships_k(cv.object)

---

**get_cluster_voting_k_votes**

*Get k vote frequencies*

**Description**
Get k vote frequencies

**Usage**

get_cluster_voting_k_votes(object)

**Arguments**

| object       | An object of class "clusterAnalysis" |

**Value**
Matrix with k vote frequencies
get_cluster_voting_memberships

Get cluster memberships for every k

Description
Get cluster memberships for every k

Usage
get_cluster_voting_memberships(object)

Arguments
object An object of class "clusterAnalysis"

Value
Cluster memberships for every k

Examples
oa.object <- omada(toy_genes, method.upper.k = 4)
get_cluster_voting_memberships(oa.object)

get_cluster_voting_metric_votes

Get a dataframe with the k votes for every index

Description
Get a dataframe with the k votes for every index

Usage
get_cluster_voting_metric_votes(object)

Arguments
object An object of class "clusterAnalysis"
**get_cluster_voting_scores**

**Value**
Dataframe with the k votes for every index

**Examples**
```r
oa.object <- omada(toy_genes, method.upper.k = 4)
get_cluster_voting_metric_votes(oa.object)
```

---

**get_cluster_voting_scores**

*Get a matrix with metric scores for every k and internal index*

**Description**
Get a matrix with metric scores for every k and internal index

**Usage**
```r
get_cluster_voting_scores(object)
```

**Arguments**
- `object`: An object of class "clusterAnalysis"

**Value**
A matrix with metric scores for every k and internal index

**Examples**
```r
oa.object <- omada(toy_genes, method.upper.k = 4)
get_cluster_voting_scores(oa.object)
```

---

**get_feature_selection_optimal_features**

*Get the optimal features*

**Description**
Get the optimal features

**Usage**
```r
get_feature_selection_optimal_features(object)
```
get_feature_selection_optimal_number_of_features

Arguments

object An object of class "clusterAnalysis"

Value

The list of optimal features

Examples

oa.object <- omada(toy_genes, method.upper.k = 4)
get_feature_selection_optimal_features(oa.object)

get_feature_selection_optimal_number_of_features

Get the optimal number of features

Description

Get the optimal number of features

Usage

get_feature_selection_optimal_number_of_features(object)

Arguments

object An object of class "clusterAnalysis"

Value

The optimal number of features

Examples

oa.object <- omada(toy_genes, method.upper.k = 6)
get_feature_selection_optimal_number_of_features(oa.object)
get_feature_selection_scores

Get a dataframe of average bootstrap stabilities

Description
Get a dataframe of average bootstrap stabilities

Usage
get_feature_selection_scores(object)

Arguments
object An object of class "clusterAnalysis"

Value
A dataframe of average bootstrap stabilities

Examples
oa.object <- omada(toy_genes, method.upper.k = 6)
get_feature_selection_scores(oa.object)

get_generated_dataset
Get the simulated dataset

Description
Get the simulated dataset

Usage
generate_dataset(object)

Arguments
object An object of class "feasibilityAnalysis"

Value
Simulated dataset

Examples
fa.object <- feasibilityAnalysis(classes = 4, samples = 50, features = 15)
generated_ds <- get_generated_dataset(fa.object)
get_internal_metric_scores

Get a matrix with metric scores for every k and internal index

Description

Get a matrix with metric scores for every k and internal index

Usage

get_internal_metric_scores(object)

Arguments

object An object of class "clusterVoting"

Value

A matrix with metric scores for every k and internal index

Examples

cv.object <- clusterVoting(toy_genes, 4,6,"sc")
get_internal_metric_scores(cv.object)

get_max_stability

Get the maximum stability

Description

Get the maximum stability

Usage

get_max_stability(object)

Arguments

object An object of class "feasibilityAnalysis"

Value

The maximum stability

Examples

fa.object <- feasibilityAnalysis(classes = 2, samples = 10, features = 15)
maximum.st <- get_max_stability(fa.object)
get_metric_votes_k

Description
Get a dataframe with the k votes for every index

Usage
get_metric_votes_k(object)

Arguments
object
An object of class "clusterVoting"

Value
Dataframe with the k votes for every index

Examples
cv.object <- clusterVoting(toy_genes, 4,6,"sc")
get_metric_votes_k(cv.object)

get_optimal_features

Description
Get the optimal features

Usage
get_optimal_features(object)

Arguments
object
An object of class "featureSelection"

Value
The list of optimal features

Examples
fs.object <- featureSelection(toy_genes, min.k = 3, max.k = 6, step = 10)
get_optimal_features(fs.object)
get_optimal_memberships

*Get a dataframe with the memberships of the samples found in the input data*

Description
Get a dataframe with the memberships of the samples found in the input data

Usage
get_optimal_memberships(object)

Arguments
- object: An object of class "optimalClustering"

Value
A dataframe with the memberships of the samples found in the input data

Examples
oc.object <- optimalClustering(toy_genes, 4, "spectral")
get_optimal_memberships(oc.object)

get_optimal_number_of_features

*Get the optimal number of features*

Description
Get the optimal number of features

Usage
get_optimal_number_of_features(object)

Arguments
- object: An object of class "featureSelection"

Value
The optimal number of features
get_optimal_parameter_used

Examples

fs.object <- featureSelection(toy_genes, min.k = 3, max.k = 6, step = 10)
get_optimal_number_of_features(fs.object)

Description

Get the optimal parameter used

Usage

get_optimal_parameter_used(object)

Arguments

object An object of class "optimalClustering"

Value

The optimal parameter used

Examples

oc.object <- optimalClustering(toy_genes, 4, "spectral")
get_optimal_parameter_used(oc.object)

get_optimal_stability_score

Description

Get the optimal stability score

Usage

get_optimal_stability_score(object)

Arguments

object An object of class "optimalClustering"
**get_partition_agreement_scores**

**Value**

The optimal stability score

**Examples**

```r
oc.object <- optimalClustering(toy_genes, 4, "spectral")
get_optimal_stability_score(oc.object)
```

---

**get_partition_agreement_scores**

*Get a dataframe of partition agreement scores for a set of random parameters clustering runs across different methods*

---

**Description**

Get a dataframe of partition agreement scores for a set of random parameters clustering runs across different methods

---

**Usage**

```r
get_partition_agreement_scores(object)
```

---

**Arguments**

- `object` An object of class "clusterAnalysis"

---

**Value**

A dataframe of partition agreement scores for a set of random parameters clustering runs across different methods

---

**Examples**

```r
ms.object <- clusteringMethodSelection(toy_genes, method.upper.k = 3, number.of.comparisons = 2)
get_partition_agreement_scores(ms.object)
oa.object <- omada(toy_genes, method.upper.k = 4)
get_partition_agreement_scores(oa.object)
```
get_sample_memberships

Get a dataframe with the memberships of the samples found in the input data

Description

Get a dataframe with the memberships of the samples found in the input data

Usage

get_sample_memberships(object)

Arguments

object An object of class "clusterAnalysis"

Value

A dataframe with the memberships of the samples found in the input data

Examples

oa.object <- omada(toy_genes, method.upper.k = 4)
get_sample_memberships(oa.object)

get_vote_frequencies_k

Get k vote frequencies

Description

Get k vote frequencies

Usage

get_vote_frequencies_k(object)

Arguments

object An object of class "clusterVoting"

Value

Matrix with k vote frequencies
omada

A wrapper function that utilizes all tools to produce the optimal sample memberships

Description

A wrapper function that utilizes all tools to produce the optimal sample memberships

Usage

omada(data, method.upper.k = 5)

Arguments

data A dataframe, where columns are features and rows are data points
method.upper.k The upper limit of clusters, k, to be considered. Must be more than 2

Value

An object of class "clusterAnalysis" containing partition.agreement.scores, partition.agreement.plot, feature.selection.scores, feature.selection.plot, feature.selection.optimal.features, feature.selection.optimal.number.of.features, cluster.voting.scores, cluster.voting.cluster.memberships, cluster.voting.metric.votes,

Examples

omada(toy_genes, method.upper.k = 3)

optimalClustering

Clustering with the optimal parameters estimated by these tools

Description

Clustering with the optimal parameters estimated by these tools

Usage

optimalClustering(data, clusters, algorithm)

Arguments

data A dataframe, where columns are features and rows are data points
clusters Number of clusters to be generated by this clustering
algorithm The clustering algorithm to be used
Value
An object of class "optimalClustering" containing a dataframe with the memberships of the samples found in the input data, the optimal stability score and parameter used.

Examples
optimalClustering(toy_genes, 2,"kmeans")

Description
Calculate the agreement (0,1) between two partitioning generated by two clustering runs using the adjust Rand Index. We can use three clustering algorithms (spectral, kmeans and hierarchical) along with the following parameters for each:

Usage
partitionAgreement(
data, 
algorithm.1 = "hierarchical", 
measure.1 = "canberra", 
hier.agglo.algorithm.1 = "average", 
algorithm.2 = "hierarchical", 
measure.2 = "manhattan", 
hier.agglo.algorithm.2 = "average", 
number.of.clusters = 5
)

Arguments
data A dataframe, where columns are features and rows are data points
algorithm.1 Second algorithm to be used (spectral/kmeans/hierarchical)
measure.1 Concerns the first algorithm to be used and represents a kernel for Spectral/kmeans or a distance measure for hierarchical clustering
hier.agglo.algorithm.1 Concerns the first algorithm to be used and represents the agglomerative method for hierarchical clustering (not used in spectral/kmeans clustering)
algorithm.2 First algorithm to be used (spectral/kmeans/hierarchical)
measure.2 Concerns the second algorithm to be used and represents a kernel for Spectral/kmeans or a distance measure for hierarchical clustering
hier.agglo.algorithm.2 Concerns the second algorithm to be used and represents the agglomerative method for hierarchical clustering (not used in spectral/kmeans clustering)
number_of_clusters
The upper limit of clusters to form starting from 2

Details
Spectral kernels: rbfdot, polydot, vanilladot, tanhdot, laplacedot, besseldot, anovadot, splinedot
K-means kernels: "Hartigan-Wong", Lloyd, Forgy, MacQueen
Hierarchical Agglomeration methods: average, ward.D, ward.D2, single, complete, mcquitty, median, centroid
Distance measures: euclidean, manhattan, canberra, minkowski, maximum

Value
An object of class "partitionAgreement" containing agreements (Rand Indexes) from 1 cluster (ARI=0) up to the number of clusters requested

Examples

    partitionAgreement(toy_genes, algorithm.1 = "hierarchical", measure.1 = "canberra", hier.agglo.algorithm.1 = "average", algorithm.2 = "hierarchical", measure.2 = "manhattan", hier.agglo.algorithm.2 = "average", number.of.clusters = 3)

    partitionAgreement(toy_genes, algorithm.1 = "spectral", measure.1 = "rbfdot", algorithm.2 = "kmeans", measure.2 = "Lloyd", number.of.clusters = 5)

plot_average_stabilities
Plot the average bootstrap stabilities

Description
Plot the average bootstrap stabilities

Usage
plot_average_stabilities(object)

Arguments

    object An object of class "featureSelection"

Value
Line plot of average bootstrap stabilities

Examples

    fs.object <- featureSelection(toy_genes, min.k = 3, max.k = 6, step = 10)
    plot_average_stabilities(fs.object)
**plot_cluster_voting**

*Plot k vote frequencies*

**Description**
Plot k vote frequencies

**Usage**

```r
plot_cluster_voting(object)
```

**Arguments**
- **object**
  
  An object of class "clusterAnalysis"

**Value**
Plot k vote frequencies

**Examples**

```r
oa.object <- omada(toy_genes, method.upper.k = 3)
plot_cluster_voting(oa.object)
```

---

**plot_feature_selection**

*Plot the average bootstrap stabilities*

**Description**
Plot the average bootstrap stabilities

**Usage**

```r
plot_feature_selection(object)
```

**Arguments**
- **object**
  
  An object of class "clusterAnalysis"

**Value**
Line plot of average bootstrap stabilities

**Examples**

```r
oa.object <- omada(toy_genes, method.upper.k = 4)
plot_feature_selection(oa.object)
```
plot_partition_agreement

*Plot of partition agreement scores*

**Description**

Plot of partition agreement scores

**Usage**

```r
plot_partition_agreement(object)
```

**Arguments**

- `object`: An object of class "clusterAnalysis"

**Examples**

```r
ms.object <- clusteringMethodSelection(toy_genes, method.upper.k = 3,
number.of.comparisons = 2)
plot_partition_agreement(ms.object)
oa.object <- omada(toy_genes, method.upper.k = 4)
plot_partition_agreement(oa.object)
```

---

plot_vote_frequencies

*Plot k vote frequencies*

**Description**

Plot k vote frequencies

**Usage**

```r
plot_vote_frequencies(object)
```

**Arguments**

- `object`: An object of class "clusterVoting"
toy_genes

Value

Plot k vote frequencies

Examples

```r
cv.object <- clusterVoting(toy_genes, 4,6,"sc")
plot_vote_frequencies(cv.object)
```

---

toy_genes | Toy gene data for package examples

Description

Columns are genes and rows are samples

Usage

```r
data(toy_genes)
```

Format

An object of class ""cross""; see [qtl::read.cross()].

Source

nope

References

nothing

Examples

```r
data(toy_genes)
```
toy_gene_memberships  
Cluster memberships for toy gene data for package examples

**Description**

Column "id" represents genes and column "memberships" represents their respective clusters. Rows are samples.

**Usage**

```r
data(toy_gene_memberships)
```

**Format**

An object of class "cross"; see `qtl::read.cross()`.

**Source**

nope

**References**

nothing

**Examples**

```r
data(toy_gene_memberships)
```
# Index

* datasets
  * toy_gene_memberships, 26
  * toy_genes, 25

clusteringMethodSelection, 3
clusterVoting, 4

feasibilityAnalysis, 4
feasibilityAnalysisDataBased, 5
featureSelection, 6

get_agreement_scores, 6
get_average_feature_k_stabilities, 7
get_average_stabilities_per_k, 8
get_average_stability, 8
get_cluster_memberships_k, 9
get_cluster_voting_k_votes, 9
get_cluster_voting_memberships, 10
get_cluster_voting_metric_votes, 10
get_cluster_voting_scores, 11
get_feature_selection_optimal_features, 11
get_feature_selection_optimal_number_of_features, 12
get_feature_selection_scores, 13
get_generated_dataset, 13
get_internal_metric_scores, 14
get_max_stability, 14
get_metric_votes_k, 15
get_optimal_features, 15
get_optimal_memberships, 16
get_optimal_number_of_features, 16
get_optimal_parameter_used, 17
get_optimal_stability_score, 17
get_partition_agreement_scores, 18
get_sample_memberships, 19
get_vote_frequencies_k, 19

omada, 20
optimalClustering, 20

partitionAgreement, 21
plot_average_stabilities, 22
plot_cluster_voting, 23
plot_feature_selection, 23
plot_partition_agreement, 24
plot_vote_frequencies, 24

toy_gene_memberships, 26
toy_genes, 25