

Package ‘FCPS’

August 20, 2026

Type Package

Title Fundamental Clustering Problems Suite

Version 1.4.1

Date 2026-08-19

Maintainer Michael Thrun <m.thrun@gmx.net>

Description Over sixty clustering algorithms are provided in this package with consistent input and output, which enables the user to try out algorithms swiftly. Additionally, 26 statistical approaches for the estimation of the number of clusters as well as the mirrored density plot (MD-plot) of clusterability are implemented. The package is published in Thrun, M.C., Stier Q.: ``Fundamental Clustering Algorithms Suite" (2021), SoftwareX, <[DOI:10.1016/j.softx.2020.100642](https://doi.org/10.1016/j.softx.2020.100642)>. Moreover, the fundamental clustering problems suite (FCPS) offers a variety of clustering challenges any algorithm should handle when facing real world data, see Thrun, M.C., Ultsch A.: ``Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems" (2020), Data in Brief, <[DOI:10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501)>.

Imports mclust, ggplot2, DataVisualizations, methods

Suggests mlpack, kernlab, cclust, dbscan, kohonen, MCL, cluster, DatabionicSwarm, orclus, flexclust, ABCanalysis, apcluster, pracma, EMCluster, pdfCluster, parallelDist, plotly, ProjectionBasedClustering, GeneralizedUmatrix, mstknnclust, densityClust, parallel, energy, R.utils, tclust, Spectrum, genie, protoclust, fastcluster, clusterability, signal, reshape2, PPCI, clustrd, smacof, rgl, CEC, dendextend, moments, prabclus, VarSelLCM, sparcl, mixtools, HDclassif, clustvarsel, yardstick, knitr, rmarkdown, igraph, leiden, clustMixType, clusterSim, NetworkToolbox, ClusterR, partitionComparison, aricode, pfclust, ConsensusClusterPlus, clue

Depends R (>= 3.5.0)

License GPL-3

LazyData TRUE

LazyLoad yes

URL <https://www.deepbionics.org/>

BugReports <https://github.com/Mthrun/FCPS/issues>

Encoding UTF-8

VignetteBuilder knitr

SystemRequirements Pandoc (>= 1.12.3)

NeedsCompilation no

Author Michael Thrun [aut, cre, cph] (ORCID:
[<https://orcid.org/0000-0001-9542-5543>](https://orcid.org/0000-0001-9542-5543)),
 Peter Nahrgang [ctr, ctb],
 Felix Pape [ctr, ctb],
 Vasył Pihur [ctb],
 Guy Brock [ctb],
 Susmita Datta [ctb],
 Somnath Datta [ctb],
 Luis Winckelmann [com],
 Alfred Ultsch [dtc, ctb],
 Quirin Stier [ctb, rev]

Repository CRAN

Date/Publication 2026-08-20 08:12:18 UTC

Contents

FCPS-package	4
AgglomerativeNestingClustering	5
APclustering	7
Atom	9
AutomaticProjectionBasedClustering	9
Chainlink	12
ClusterabilityMDplot	13
ClusterAlignLabels	15
ClusterAMI	17
ClusterApply	18
ClusterARI	20
ClusterChallenge	22
ClusterCount	23
ClusterCreateClassification	24
ClusterDaviesBouldinIndex	25
ClusterDendrogram	26
ClusterDistances	28
ClusterDunnIndex	29
ClusterEqualWeighting	30
ClusteringAccuracy	31
ClusterInterDistances	33
ClusterMCC	34
ClusterNMI	35

ClusterNoEstimation	37
ClusterNormalize	39
ClusterPlotMDS	40
ClusterRedefine	42
ClusterRename	43
ClusterRenameDescendingSize	44
ClusterShannonInfo	45
ClusterUpsamplingMinority	47
ConsensusClustering	48
CrossEntropyClustering	49
DBSCAN	51
DBScusteringAndVisualization	53
DensityPeakClustering	56
DivisiveAnalysisClustering	58
EngyTime	59
EntropyOfDataField	60
EstimateRadiusByDistance	61
FannyClustering	62
GapStatistic	63
GenieClustering	64
GolfBall	66
HCLclustering	66
HDDClustering	67
Hepta	69
HierarchicalClusterData	70
HierarchicalClusterDists	71
HierarchicalClustering	75
HierarchicalDBSCAN	79
IsDissimilarity	81
kmeansClustering	82
kmeansDist	85
LargeApplicationClustering	86
Leukemia	88
Lsun3D	89
MarkovClustering	89
MeanShiftClustering	91
MinimalEnergyClustering	92
MinimaxLinkageClustering	93
ModelBasedClustering	95
ModelBasedVarSelClustering	96
MoGclustering	98
MSTclustering	100
NetworkClustering	101
NeuralGasClustering	102
OPTICSclustering	103
PAMclustering	105
parApplyClusterAnalysis	106
parApplyDataBasedCA	109

parApplyDistanceBasedCA	111
pdfClustering	113
PFclustering	114
ProjectionPursuitClustering	115
QTclustering	117
RobustTrimmedClustering	118
SharedNearestNeighborClustering	120
SOMclustering	121
SOTAclustering	123
SparseClustering	124
SpectralClustering	126
Spectrum	127
StatPDEdensity	129
SubspaceClustering	129
TandemClustering	131
Target	133
Tetra	134
TwoDiamonds	134
WingNut	135
Index	136

FCPS-package

Fundamental Clustering Problems Suite

Description

Over sixty clustering algorithms are provided in this package with consistent input and output, which enables the user to try out algorithms swiftly. Additionally, 26 statistical approaches for the estimation of the number of clusters as well as the mirrored density plot (MD-plot) of clusterability are implemented. The package is published in Thrun, M.C., Stier Q.: "Fundamental Clustering Algorithms Suite" (2021), SoftwareX, <DOI:10.1016/j.softx.2020.100642>. Moreover, the fundamental clustering problems suite (FCPS) offers a variety of clustering challenges any algorithm should handle when facing real world data, see Thrun, M.C., Ultsch A.: "Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems" (2020), Data in Brief, <DOI:10.1016/j.dib.2020.105501>.

The package consists of many algorithms and fundamental datasets for clustering published in [Thrun/Stier, 2021]. Originally, the 'Fundamental Clustering Problems Suite' (FCPS) offered a variety of clustering problems any algorithm shall be able to handle when facing real world data. Nine of the here presented artificial datasets were priorly named FCPS with a fixed sample size in Ultsch, A.: "Clustering with SOM: U*C", In Workshop on Self-Organizing Maps, 2005. FCPS often served in the paper as an elementary benchmark for clustering algorithms. The FCPS package extends datasets, enables variable sample sizes for these datasets, and provides a standardized and easy access to many clustering algorithms.

<https://www.deepbionics.org/>

Details

FCPS datasets consists of data sets with known a priori classification to be reproduced by the algorithms. All data sets are intentionally created to be simple and might be visualized in two or three dimensions. Each data sets represents a certain problem that is solved by known clustering algorithms with varying success. This is done in order to reveal benefits and shortcomings of algorithms in question. Standard clustering methods, e.g. single-linkage, ward and k-means, are not able to solve all FCPS problems satisfactorily. "Lsun3D and each of the nine artificial data sets of "Fundamental Clustering Problems Suite" (FCPS) were defined separately for a specific clustering problem as cited (in [Thrun/Ultsch, 2020]). The original sample size defined in the respective first publication mentioning the data was used in [Thrun/Ultsch, 2020], but using the R function "ClusterChallenge" (...) any sample size can be drawn for all artificial data sets. [Thrun/Ultsch, 2020]

Index: This package was not yet installed at build time.

Author(s)

Michael Thrun [aut, cre, cph] (ORCID: <<https://orcid.org/0000-0001-9542-5543>>), Peter Nahrgang [ctr, ctb], Felix Pape [ctr, ctb], Vasyl Pihur [ctb], Guy Brock [ctb], Susmita Datta [ctb], Somnath Datta [ctb], Luis Winkelmann [com], Alfred Ultsch [dtc, ctb], Quirin Stier [ctb, rev]

Maintainer: Michael Thrun <m.thrun@gmx.net>

References

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:[10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

[Thrun/Stier, 2021] Thrun, M. C., & Stier, Q.: Fundamental Clustering Algorithms Suite SoftwareX, Vol. 13(C), in press, pp. 100642. doi:[10.1016/j.softx.2020.100642](https://doi.org/10.1016/j.softx.2020.100642), 2021.

[Ultsch, 2005] Ultsch, A.: Clustering with SOM: U*C, In Proc. Workshop on Self-Organizing Maps, pp. 75-82, Paris, France, 2005.

AgglomerativeNestingClustering
AGNES clustering

Description

Agglomerative hierarchical clustering (AGNES) of [Rousseeuw/Kaufman, 1990, pp. 199-252]

Usage

```
AgglomerativeNestingClustering(DataOrDistances, ClusterNo,
PlotIt = FALSE, Standardization = TRUE, ...)
```

Arguments

DataOrDistances	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be built by the algorithm. if ClusterNo=0, the dendrogram is generated instead of a clustering to estimate the numbers of clusters.
PlotIt	Default: FALSE if ClusterNo!=0, If TRUE or ClusterNo=0 plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
Standardization	DataOrDistances is standardized before calculating the dissimilarities. Measurements are standardized for each variable (column), by subtracting the variable's mean value and dividing by the variable's mean absolute deviation. If DataOrDistances is already a distance matrix, then this argument will be ignored.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Dendrogram	Dendrogram of hierarchical clustering algorithm
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Rousseeuw/Kaufman, 1990] Rousseeuw, P. J., & Kaufman, L.: Finding groups in data, Belgium, John Wiley & Sons Inc., ISBN: 0471735787, doi 10.1002/9780470316801, Online ISBN: 9780470316801, 1990.
- [Struyf et al., 1996] Struyf, A., Hubert, M. and Rousseeuw, Peter J.: Clustering in an Object-Oriented Environment, Journal of Statistical Software, Vol. 1, doi: 10.18637/jss.v001.i04, 1996.
- [Struyf et al., 1997] Struyf, A., Hubert, M. and Rousseeuw, P.J.: Integrating Robust Clustering Techniques in S-PLUS, Computational Statistics and Data Analysis, Vol. 26, pp. 17–37, 1997.

See Also

[agnes](#)

Examples

```

data('Hepta')
CA=AgglomerativeNestingClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
## Not run:
ClusterDendrogram(CA$Dendrogram,7,main='AGNES clustering')

print(CA$Object)
plot(CA$Object)

## End(Not run)

```

APclustering

*Affinity Propagation Clustering***Description**

Affinity propagation clustering published by [Frey/Dueck, 2007] and implemented by [Bodenhofer et al., 2011].

Usage

```

APclustering(DataOrDistances,

InputPreference=NA,ExemplarPreferences=NA,

DistanceMethod="euclidean",

Seed=7568,PlotIt=FALSE,Data,...)

```

Arguments

DataOrDistances	[1:n,1:d] with: if d=n and symmetric then distance matrix assumed, otherwise: [1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features. In the latter case the Euclidean distances will be calculated.
InputPreference	Default parameter set, see apcluster
ExemplarPreferences	Default parameter set, see apcluster
DistanceMethod	DistanceMethod as in dist for similarities .
Seed	Set as integervalue to have reproducible results, see apcluster
PlotIt	Default: FALSE, If TRUE and dataset of [1:n,1:d] dimensions then a plot of the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s will be generated.

Data	[1:n,1:d] data matrix in the case that DataOrDistances is missing and partial matching does not work.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

Distancematrix D is converted to similarity matrix S with $S=-(D^2)$.

If data matrix is used, then euclidean similarities are calculated by [similarities](#) and a specified distance method.

The AP algorithm decides the k number of clusters.

Value

List of

Cls [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.

Object Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Frey/Dueck, 2007] Frey, B. J., & Dueck, D.: Clustering by passing messages between data points, Science, Vol. 315(5814), pp. 972-976, <doi:10.1126/science.1136800>, 2007.

[Bodenhofer et al., 2011] Bodenhofer, U., Kothmeier, A., & Hochreiter, S.: APCluster: an R package for affinity propagation clustering, Bioinformatics, Vol. 27(17), pp, 2463-2464, 2011.

See Also

apcluster

Examples

```
data('Hepta')
res=APclustering(Hepta$Data, PlotIt = FALSE)
```

Atom

Atom introduced in [Ultsch, 2004].

Description

Two nested spheres with different variances that are not linear not separable. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Atom")
```

Details

Size 800, Dimensions 3, stored in Atom\$Data

Classes 2, stored in Atom\$Cls

References

[Ultsch, 2004] Ultsch, A.: Strategies for an artificial life system to cluster high dimensional data, Abstracting and Synthesizing the Principles of Living Systems, GWAL-6, U. Brggemann, H. Schaub, and F. Detje, Eds, pp. 128-137. 2004.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:[10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(Atom)
str(Atom)
```

AutomaticProjectionBasedClustering

Automatic Projection-Based Clustering

Description

Projection-based clustering [AutomaticProjectionBasedClustering](#) projects the data (nonlinear) into two dimensions and tries only to preserve relevant neighborhoods prior to clustering. The cluster analysis itself includes the high-dimensional distances in the clustering process. Performs non-interactive projection-based clustering based on non-linear projection methods [Thrun/Ultsch, 2017], [Thrun/Ultsch, 2020a].

Usage

```
AutomaticProjectionBasedClustering(DataOrDistances, ClusterNo, Type="NerV",
    StructureType = TRUE, PlotIt=FALSE, PlotTree=FALSE, PlotMap=FALSE, ...)
```

Arguments

DataOrDistances	Either nonsymmetric [1:n,1:d] numerical matrix of a dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. or symmetric [1:n,1:n] distance matrix, e.g. as <code>matrix(dist(Data,method))</code>
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Type	Type of Projection method, either NerV [Venna et al., 2010] Pswarm [Thrun/Ultsch, 2020b] MDS [Torgerson, 1952] Uwot [McInnes et al., 2018] CCA [Demartines/Herault, 1995] Sammon [Sammon, 1969] t-SNE [Van der Maaten/Hinton, 2008]
StructureType	Either compact (TRUE) or connected (FALSE), see discussion in [Thrun, 2018]
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
PlotTree	TRUE: Plots the dendrogram, FALSE: no plot
PlotMap	Plots the topographic map [Thrun et al., 2016].
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

The first idea of using non-PCA projections for clustering was published by [Bock, 1987] as a definition. However, to the knowledge of the author, it was not applied to any data. The coexistence of projection and clustering was introduced in [Thrun/Ultsch, 2017].

Projection-based clustering is based on a nonlinear projection of high-dimensional data into a two-dimensional space [Thrun/Ultsch, 2020b]. Typical projection-methods like t-distributed stochastic neighbor embedding (t-SNE) [Van der Maaten/Hinton, 2008], or neighbor retrieval visualizer (NerV) [Venna et al., 2010] are used project data explicitly into two dimensions disregarding the subspaces of higher dimension than two and preserving only relevant neighborhoods in high-dimensional data. In the next step, the Delaunay graph [Delaunay, 1934] between the projected points is calculated, and each vertex between two projected points is weighted with the high-dimensional distance between the corresponding high-dimensional data points. Thereafter the shortest path between every pair of points is computed using the Dijkstra algorithm [Dijkstra, 1959]. The shortest paths are then used in the clustering process, which involves two choices depending on

the structure type in the high-dimensional data [Thrun/Ultsch, 2020b]. This Boolean choice can be decided by looking at the topographic map of high-dimensional structures [Thrun/Ultsch, 2020a]. In a benchmarking of 34 comparable clustering methods, projection-based clustering was the only algorithm that always was able to find the high-dimensional distance or density-based structure of the dataset [Thrun/Ultsch, 2020b].

It should be noted that it is preferable to use a visualization for the Generalized U-Matrix like the topographic map [plotTopographicMap](#) of [Thrun et al., 2016] to evaluate the choice of the boolean parameter `StructureType` and the clustering, improve it or set the number of clusters appropriately. A comparison with 32 clustering algorithms showed that PBC is always able to find the correct cluster structure while the best of the 32 clustering algorithms varies depending on the dataset [Thrun/Ultsch, 2020].

The first systematic comparison to other DR clustering methods like Projection-Pursuit Methods [ProjectionPursuitClustering](#), supspace clustering methods [SubspaceClustering](#), and CA-based clustering methods can be found in [Thrun/Ultsch, 2020a]. For PCA-based clustering methods please see [TandemClustering](#).

Value

List of

<code>Cls</code>	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. . Points which cannot be assigned to a cluster will be reported with 0.
<code>Object</code>	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Bock, 1987] Bock, H.: On the interface between cluster analysis, principal component analysis, and multidimensional scaling, *Multivariate statistical modeling and data analysis*, (pp. 17-34), Springer, 1987.
- [Thrun/Ultsch, 2017] Thrun, M. C., & Ultsch, A.: Projection based Clustering, *Proc. International Federation of Classification Societies (IFCS)*, pp. 250-251, Tokai University, Japanese Classification Society (JCS), Tokyo, Japan August 7-10, 2017.
- [Thrun/Ultsch, 2020a] Thrun, M. C., & Ultsch, A.: Using Projection based Clustering to Find Distance and Density based Clusters in High-Dimensional Data, *Journal of Classification*, in press, doi 10.1007/s00357-020-09373-2, 2020.
- [Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A.: Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, Vol. 24, pp. 7-16, Plzen, <http://wscg.zcu.cz/wscg2016/short/A43-full.pdf>, 2016.
- [McInnes et al., 2018] McInnes, L., Healy, J., & Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction, *arXiv preprint arXiv:1802.03426*, 2018.

[Demartines/Herault, 1995] Demartines, P., & Herault, J.: CCA:" Curvilinear component analysis", Proc. 15 Colloque sur le traitement du signal et des images, Vol. 199, GRETSI, Groupe d Etudes du Traitement du Signal et des Images, France 18-21 September, 1995.

[Sammon, 1969] Sammon, J. W.: A nonlinear mapping for data structure analysis, IEEE Transactions on computers, Vol. 18(5), pp. 401-409. doi doi:10.1109/t-c.1969.222678, 1969.

[Thrun/Ultsch, 2020b] Thrun, M. C., & Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, Journal of Artificial Intelligence, Vol. in press, pp. doi 10.1016/j.artint.2020.103237, 2020.

[Torgerson, 1952] Torgerson, W. S.: Multidimensional scaling: I. Theory and method, Psychometrika, Vol. 17(4), pp. 401-419. 1952.

[Venna et al., 2010] Venna, J., Peltonen, J., Nybo, K., Aidos, H., & Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization, The Journal of Machine Learning Research, Vol. 11, pp. 451-490. 2010.

[Van der Maaten/Hinton, 2008] Van der Maaten, L., & Hinton, G.: Visualizing Data using t-SNE, Journal of Machine Learning Research, Vol. 9(11), pp. 2579-2605. 2008.

Examples

```
data('Hepta')
out=AutomaticProjectionBasedClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

Chainlink

Chainlink introduced in [Ultsch et al., 1994; Ultsch, 1995].

Description

Two chains of rings. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Chainlink")
```

Details

Size 1000, Dimensions 3, stored in Chainlink\$Data

Classes 2, stored in Chainlink\$Cls

References

[Ultsch et al., 1994] Ultsch, A., Guimaraes, G., Korus, D., & Li, H.: Knowledge extraction from artificial neural networks and applications, Parallele Datenverarbeitung mit dem Transputer, (pp. 148-162), Springer, 1994.

[Ultsch, 1995] Ultsch, A.: Self organizing neural networks perform different from statistical k-means clustering, Proc. Society for Information and Classification (GFKL), Vol. 1995, Basel 8th-10th March 1995.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

Examples

```
data(Chainlink)
str(Chainlink)
```

ClusterabilityMDplot *Clusterability MDplot*

Description

Clusterability mirrored-density plot. Clusterability aims to quantify the degree of cluster structures [Adolfsson et al., 2019]. A dataset has a high probability to possess cluster structures, if the first component of the PCA projection is multimodal [Adolfsson et al., 2019]. As the dip test is less exact than the MDplot [Thrun et al., 2020], p-values above 0.05 can be given for MDplots which are clearly multimodal.

An alternative investigation of clusterability can be performed by inspecting the topographic map of the Generalized U-Matrix for a specific projection method using the **ProjectionBasedClustering** and **GeneralizedUmatrix** packages on CRAN, see [Thrun/Ultsch, 2021] for details.

Usage

```
ClusterabilityMDplot(DataOrDistance, Method,
na.rm=FALSE, PlotIt=TRUE, ...)
```

Arguments

DataOrDistance	Either a dataset[1:n,1:d] of n cases and d features or a symmetric distance matrix [1:d,1:d] or multiple data sets or distances in a list
Method	"none" performs no dimension reduction. "pca" uses the scores from the first principal component. "distance" computes pairwise distances (using distance_metric as the metric).
na.rm	Statistical testing will not work with missing values, if TRUE values are imputed with averages
PlotIt	TRUE: print plot, otherwise do not plot directly, instead use Handle for further adjustment
...	Further arguments for functionMDplot4multiplevectors of package DataVisualizations like "main", and "Ordering"

Details

Use the method of [Adolfsson et al., 2019] specified as `pca plus dip-test` (PCA dip) per default without scaling or standardization of data because this step should never be done automatically. In [Thrun, 2020] the standardization and scaling did not improve the results.

If `list` is named, then the names of the list will be used and the MDplots will be re-ordered according to multimodality in the plot, otherwise only the pvalues of [Adolfsson et al., 2019] will be the names and the ordering of the MDplots is the same as the list.

Beware, as shown below, this test fails for almost touching clusters of Tetra and is difficult to interpret on WingNut but with overlaid with a robustly estimated unimodal Gaussian distribution it can be interpreted as multimodal). However, it does not fail for chaining data contrary to the claim in [Adolfsson et al., 2019].

Based on [Thrun, 2020], the author of this function disagrees with [Adolfsson et al., 2019] as to the preference which clusterability method should be used because the approach "distance" is not preferable for density-based cluster structures.

Value

List of

Handle GGobject, plotter handle of **ggplot2**

Pvalue One or more p-values of dip test depending on `DataOrDistance`

Note

"none" seems to call `dip.test` in `clusterabilitytest` with high-dimensional data. In that case `dip.test` just vectorizes the matrix of the data which does not make any sense. Since this could be a bug, the "none" option should not be used.

Imputation does not work for distance matrices. Imputation is still experimental. It is advised to impute missing values before using this function

Author(s)

Michael Thrun

References

[Adolfsson et al., 2019] Adolfsson, A., Ackerman, M., & Brownstein, N. C.: To cluster, or not to cluster: An analysis of clusterability methods, *Pattern Recognition*, Vol. 88, pp. 13-26. 2019.

[Thrun et al., 2020] Thrun, M. C., Gehlert, T. & Ultsch, A.: Analyzing the Fine Structure of Distributions, *PLoS ONE*, Vol. 15(10), pp. 1-66, DOI [doi:10.1371/journal.pone.0238835](https://doi.org/10.1371/journal.pone.0238835), 2020.

[Thrun/Ultsch, 2021] Thrun, M. C., and Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, *Artificial Intelligence*, Vol. 290, pp. 103237, [doi:10.1016/j.artint.2020.103237](https://doi.org/10.1016/j.artint.2020.103237), 2021.

[Thrun, 2020] Thrun, M. C.: Improving the Sensitivity of Statistical Testing for Clusterability with Mirrored-Density Plot, in Archambault, D., Nabney, I. & Peltonen, J. (eds.), *Machine Learning Methods in Visualisation for Big Data*, The Eurographics Association, <https://diglib.org:443/handle/10.2312/mlvis20201102>, Norrköping, Sweden, May, 2020.

See Also[MDplot](#)**Examples**

```
##one dataset
data(Hepta)

ClusterabilityMDplot(Hepta$Data)

##multiple datasets
data(Atom)
data(Chainlink)
data(Lsun3D)
data(GolfBall)
data(EngyTime)
data(Target)
data(Tetra)
data(WingNut)
data(TwoDiamonds)

DataV = list(
  Atom = Atom$Data,
  Chainlink = Chainlink$Data,
  Hepta = Hepta$Data,
  Lsun3D = Lsun3D$Data,
  GolfBall = GolfBall$Data,
  EngyTime = EngyTime$Data,
  Target = Target$Data,
  Tetra = Tetra$Data,
  WingNut = WingNut$Data,
  TwoDiamonds = TwoDiamonds$Data
)

ClusterabilityMDplot(DataV)
```

ClusterAlignLabels	<i>Align Cluster Labels to a Reference Clustering</i>
--------------------	---

Description

Aligns the numeric labels of a candidate clustering to those of a reference clustering by solving a one-to-one assignment problem that maximizes the total number of overlapping observations.

Because cluster labels are arbitrary identifiers, two equivalent partitions may use different numeric labels. This function finds the optimal relabeling of the candidate clustering without changing observation order.

Usage

```
ClusterAlignLabels(Cls_reference, Cls_candidate)
```

Arguments

- Cls_reference** Numeric vector of length n containing the reference cluster labels. These labels define the target labeling.
- Cls_candidate** Numeric vector of length n containing the candidate cluster labels that should be aligned to **Cls_reference**.

Details

The function first determines the sorted unique labels in the reference and candidate clusterings. A one-to-one relabeling is only possible if both clusterings contain the same number of distinct labels. A contingency matrix is then constructed with `table()`. Its rows represent candidate labels and its columns represent reference labels.

The optimal label permutation is obtained with

```
clue::solve_LSAP(overlap_matrix, maximum = TRUE),
```

which solves the linear sum assignment problem and maximizes the sum of the selected overlap counts.

The resulting assignment provides exactly one reference label for each candidate label. Candidate labels are then replaced through this mapping using `match()`, preserving the original observation order.

The returned agreement statistics allow direct comparison of label agreement before and after alignment.

Value

A list with components:

- Cls_aligned** Numeric vector of length n containing the optimally relabeled candidate clustering. Observation order is unchanged.
- Mapping** Data frame with one row per candidate label and columns `candidate_label`, `reference_label`, and `matched_observations`.
- Matches_before** Integer count of observations for which reference and candidate labels were already identical before relabeling.
- Matches_after** Integer count of observations for which the reference label equals the aligned candidate label after optimal relabeling.
- Agreement_before** Numeric proportion of observations with identical labels before alignment.
- Agreement_after** Numeric proportion of observations with identical labels after alignment.
- Overlap_matrix** Integer contingency matrix. Rows correspond to candidate labels and columns to reference labels. Entry `[i, j]` contains the number of observations assigned simultaneously to candidate class `i` and reference class `j`.

Author(s)

Michael Thrun

See Also[solve_LSAP](#)**Examples**

```
## Not run:
reference <- c(1, 1, 1, 2, 2, 3, 3)
candidate <- c(2, 2, 2, 3, 3, 1, 1)

result <- ClusterAlignLabels(
  Cls_reference = reference,
  Cls_candidate = candidate
)

result$Cls_aligned
result$Mapping
result$Agreement_before
result$Agreement_after

## End(Not run)
```

ClusterAMI

*Adjusted Mutual Information [Vinh et al., 2009]***Description**

Mutual information measures the amount of information that two clusterings share: the higher the MI, the more information shared, indicating a higher similarity between the two clusterings [Cover and Thomas, 1991].

Usage

```
ClusterAMI(Cls1, Cls2)
```

Arguments

Cls1	1:n numerical vector of numbers defining the classification as the main output of the first clustering or trial for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Cls2	1:n numerical vector of numbers defining the classification as the main output of the second clustering algorithm trial for the n cases of data. It has p unique numbers representing the arbitrary labels of the clustering.

Details

Adjusted mutual information ranges from 0 (no similarity between true labels and cluster assignments) to 1 (perfect similarity). It adjusts for chance associations between true labels and cluster assignments.

Value

value of adjusted mutual information

Author(s)

Michael Thrun (Wrapper only)

References

[Vinh et al., 2009] Vinh, N. X.; Epps, J.; Bailey, J.: Information theoretic measures for clusterings comparison, Proceedings of the 26th Annual International Conference on Machine Learning (ICML), doi:10.1145/1553374.1553511, 2009.

[Cover and Thomas, 1991] Cover, T. and Thomas, J.A.: Elements of Information Theory. Wiley, 1991.

Examples

```
data(Hepta)
#compare to baseline
Cls2=kmeansClustering(Hepta$Data,7,Type = "Steinley")$Cls
ClusterAMI(Hepta$Cls,Cls2)
#compare different solutions
Cls3=kmeansClustering(Hepta$Data,5)$Cls
ClusterAMI(Cls3,Cls2)
```

ClusterApply

Applies a function over grouped data

Description

Applies a given function to each dimension d of data separately for each cluster

Usage

```
ClusterApply(DataOrDistances,FUN,Cls,Simple=FALSE,...)
```

Arguments

DataOrDistances	[1:n,1:d] with: if d=n and symmetric then distance matrix assumed, otherwise: [1:n,1:d] matrix of defining the dataset that consists of n cases or d-dimensional data points. Every case has d attributes, variables or features.
FUN	Function to be applied to each cluster of data and each column of data
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Simple	Boolean, if TRUE, simplifies output
...	Additional parameters to be passed on to FUN

Details

Applies a given function to each feature of each cluster of data using the clustering stored in Cls which is the cluster identifiers for all rows in data. If missing, all data are in first cluster, The main output is FUNPerCluster[i] which is the result of FUN for the data points in cluster of UniqueClusters[i] named with the function's name used.

In case of a distance matrix an automatic classical multidimensional scaling transformation of distances to data is computed. Number of dimensions is selected by the minimal stress w.r.t. the possible output dimensions of cmdscale.

If FUN has not function name, then ResultPerCluster is given back.

Value

if(Simple==FALSE) List with

UniqueClusters The unique clusters in Cls

FUNPerCluster a matrix of [1:k,1:d] of d features and k clusters, the list element is named by the function FUN used

if(Simple==TRUE)

a matrix of [1:k,1:d] of d features and k clusters

Author(s)

Felix Pape, Michael Thrun

Examples

```
##one dataset
data(Hepta)
Data=Hepta$Data
Cls=Hepta$Cls
#mean per cluster
ClusterApply(Data,mean,Cls)

#Simplified
```

```

ClusterApply(Data,mean,Cls,Simple=TRUE)

# Mean per cluster of MDS transformation
# Beware, this is not the same!

ClusterApply(as.matrix(dist(Data)),mean,Cls)

## Not run:
Iris=datasets::iris
Distances=as.matrix(Iris[,1:4])
SomeFactors=Iris$Species
V=ClusterCreateClassification(SomeFactors)
Cls=V$Cls
V$ClusterNames
ClusterApply(Distances,mean,Cls)

## End(Not run)
#special case of identity
## Not run:
suppressPackageStartupMessages(library('prabclus',quietly = TRUE))
data(tetragonula)
#Generated Specific Distance Matrix
ta <- alleleconvert(strmatrix=as.matrix(tetragonula[1:236,]))
tai <- alleleinit(allelematrix=ta,distance="none")
Distance=alleledist((unbuild.charmatrix(tai$charmatrix,236,13)),236,13)

MDStrans=ClusterApply(Distance,identity)$identityPerCluster

## End(Not run)

```

ClusterARI

Adjusted Rand index

Description

Adjusted Rand index for two clusterings that should be compared to each other. This index has expected value zero for independant clusterings and maximum value 1 (for identical clusterings).

Usage

```
ClusterARI(Cls1, Cls2,Fast=TRUE)
```

Arguments

Cls1	1:n numerical vector of numbers defining the classification as the main output of the first clustering or trial for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
------	---

Cls2	1:n numerical vector of numbers defining the classification as the main output of the second clustering algorithm trial for the n cases of data. It has p unique numbers representing the arbitrary labels of the clustering.
Fast	TRUE:uses mclust package which maybe does not integrate all published insights about ARI FALSE: uses partitionComparison package

Details

"The expected value of the Rand Index of two random partitions does not take a constant value (e.g. zero). Thus, Hubert and Arabie proposed an adjustment [Hubert & Arabie] which assumes a generalized hypergeometric distribution as null hypothesis: the two clusterings are drawn randomly with a fixed number of clusters and a fixed number of elements in each cluster (the number of clusters in the two clusterings need not be the same). Then the adjusted Rand Index is the (normalized) difference of the Rand Index and its expected value under the null hypothesis. The significance of this measure has to be put into question because of the strong assumptions it makes on the distribution. Meila [Meila, 2003] notes, that some pairs of clusterings may result in negative index values" [Wagner and Wagner, 2007].

Value

value of adjusted rand index

Note

the equation of adjusted random index ignores the labels themselves and measures only the agreement. Hence, one can compare clustering solutions for $k \neq p$ unique numbers that represent the labels, see second example

Author(s)

Michael Thrun (Wrapper only)

References

- [Rand, 1971] Rand, W. M.: Objective criteria for the evaluation of clustering methods, Journal of the American Statistical Association, Vol. 66(336), pp. 846-850, 1971.
- [Hubert & Arabie] Hubert, L. and Arabie, P.: Comparing partitions, Journal of Classification. Vol. 2 (1), pp. 193-218. doi:10.1007/BF01908075, 1985.
- [Ball/Geyer-Schulz, 2018] Ball, F., & Geyer-Schulz, A.: Invariant Graph Partition Comparison Measures, Symmetry, Vol. 10(10), pp. 1-27, 2018.
- [Meila, 2003] Meila, Marina: Comparing Clusterings. COLT 2003.
- [Wagner and Wagner, 2007] Wagner, Silke; Wagner, Dorothea. Comparing clusterings: an overview. Karlsruhe: Universitaet Karlsruhe, Fakultaeet für Informatik, 2007.

See Also

[adjustedRandIndex](#)

Examples

```

data(Hepta)
#compare to baseline
Cls2=kmeansClustering(Hepta$Data,7,Type = "Steinley")$Cls
ClusterARI(Hepta$Cls,Cls2)
#compare different solutions
Cls3=kmeansClustering(Hepta$Data,5)$Cls
ClusterARI(Cls3,Cls2)

```

ClusterChallenge	<i>Generates a Fundamental Clustering Challenge based on specific artificial datasets.</i>
------------------	--

Description

Lsun3D and FCPS datasets were introduced in various publications for a specific fixed size. This function generalizes them for any sample size.

Usage

```

ClusterChallenge(Name, SampleSize,

PlotIt=FALSE, PointSize=1, Plotter3D="rgl", ...)

```

Arguments

Name	string, either 'Atom', 'Chainlink', 'EngyTime', 'GolfBall', 'Hepta', 'Lsun3D', 'Target' 'Tetra' 'TwoDiamonds' 'WingNut
SampleSize	Size of Sample higher than 300, preferable above 500
PlotIt	TRUE: Plots the challenge with ClusterPlotMDS
PointSize	If PlotIt=TRUE: see ClusterPlotMDS
Plotter3D	If PlotIt=TRUE: see ClusterPlotMDS
...	If PlotIt=TRUE: further arguments for ClusterPlotMDS

Details

A detailed description of the datasets can be found in [Thrun/Ultsch 2020]. Sampling works by combining Pareto Density Estimation with rejection sampling.

Value

LIST, with

Name	[1:SampleSize,1:d] data matrix
Cls	[1:SampleSize] numerical vector of classification

Author(s)

Michael Thrun

References

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. in press, pp. 105501, [doi:10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

See Also

[ClusterPlotMDS](#)

Examples

```
## Not run:
ClusterChallenge("Chainlink",2000,PlotIt=TRUE)

## End(Not run)
```

ClusterCount	<i>ClusterCount</i>
--------------	---------------------

Description

Calulates statistics for clustering in each group of the data points

Usage

```
ClusterCount(Cls,Ordered=TRUE,NonFinite=9999)
```

Arguments

Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Ordered	Optional, boolean, if TRUE: the ouput is ordered increasingly by cluster labels in UniqueClusters.
NonFinite	Optional, If non finite values are given in the numerical vector, they are set to the scalar value defined here

Details

The ordering of the output is defined by the first occurence of every cluster label in Cls in the setting of Ordered=FALSE.

The function can be overloaded with non-numerical vectors. In this case, a cast via `as.character()` is applied to Cls, a warning is stated, and the statistics are still computed.

Value

UniqueClusters [1:k] numerical vector of the k unique clusters in Cls

CountPerCluster
Named vector [1:k] with the number of data points in the corresponding unique clusters. Names are the UniqueClusters

NumberOfClusters
The number of clusters k

ClusterPercentages
[1:k] numerical vector of the percentages of datapoints belonging to a cluster for each cluster

Author(s)

Michael Thrun

Examples

```
data('Hepta')
Cls=Hepta$Cls
ClusterCount(Cls)
```

ClusterCreateClassification

Create Classification for Cluster.. functions

Description

Creates a Cls from arbitrary list of objects

Usage

```
ClusterCreateClassification(Objects,Decreasing,na.rm=FALSE)
```

Arguments

Objects	Listed objects, for example factor
Decreasing	Boolean that can be missing. If given, sorts ClusterNames with either decreasing or increasing.
na.rm	Boolean, if TRUE all non finite values of Cls are set to 0. FALSE: NaN and Inf and -Inf are set to different numeric labels, NA is set to NaN

Details

ClusterNames can be sorted before the classification stored Cls is created. See example.

Value

LIST, with

Cls	[1:n] numerical vector with n numbers defining the labels of the classification. It has 1 to k unique numbers representing the arbitrary labels of the classification.
ClusterNames	ClusterNames defined which names belongs to which unique number

Author(s)

Michael Thrun

Examples

```
## Not run:
Iris=datasets::iris
SomeFactors=Iris$Species
V=ClusterCreateClassification(SomeFactors)
Cls=V$Cls
V$ClusterNames
table(Cls,SomeFactors)

#Increasing alphabetical order
V=ClusterCreateClassification(SomeFactors,Decreasing=FALSE)
Cls=V$Cls
V$ClusterNames
table(Cls,SomeFactors)

## End(Not run)
```

ClusterDaviesBouldinIndex

Davies Bouldin Index

Description

Internal (i.e. without prior classification) cluster quality measure called Davies Bouldin index for a given clustering published in [Davies/Bouldin, 1979].

Usage

```
ClusterDaviesBouldinIndex(Cls, Data,...)
```

Arguments

Cls	[1:n] numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Data	matrix, [1:d,1:n] dataset of d variables and n cases
...	Further arguments passed on to the index.DB function of clusterSim

Details

Wrapper for [index.DB](#). Davies Bouldin index is defined in [Davies/Bouldin, 1979]. Best clustering scheme essentially minimizes the Davies-Bouldin index because it is defined as the function of the ratio of the within cluster scatter, to the between cluster separation.[Davies/Bouldin, 1979].

Value

List of
DaviesBouldinIndex
scalar,Davies Bouldin index
Object further information stored in [index.DB](#)

Author(s)

Michael Thrun

References

[Davies/Bouldin, 1979] Davies, D. L., & Bouldin, D. W.: A cluster separation measure, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 1(2), pp. 224-227. doi 10.1109/TPAMI.1979.4766909, 1979.

Examples

```
data("Hepta")
Cls=kmeansClustering(Hepta$Data,ClusterNo = 7,Type="Hartigan")$Cls
ClusterDaviesBouldinIndex(Cls,Hepta$Data)[1]

data("Hepta")
ClsWellSeperated=kmeansClustering(Hepta$Data,ClusterNo = 7,Type="Steinley")$Cls
ClusterDaviesBouldinIndex(ClsWellSeperated,Hepta$Data)[1]
```

ClusterDendrogram	<i>Cluster Dendrogram</i>
-------------------	---------------------------

Description

Presents a dendrogram of a given tree using a colorsequence for the branches defined from the highest cluster size to the lowest cluster size.

Usage

```
ClusterDendrogram(TreeOrDendrogram, ClusterNo,
Colorsequence,main,xlab,ylab)
```

Arguments

TreeOrDendrogram	Either object of <code>hclust</code> defining the tree, third list element of hierarchical cluster algorithms of this package or Object of class <code>dendrogram</code> , second list element of hierarchical cluster algorithms.
ClusterNo	k number of clusters for <code>cutree</code> .
Colorsequence	[1:k] character vector of colors, per default the <code>colorsequence</code> defined in the Data Visualizations is used
main	title of plot
xlab	x label of plot
ylab	ylabel of plot

Details

Requires the package **dendextend** to work correctly.

Value

In mode invisible:

[1:n] numerical vector defining the clustering of k clusters; this classification is the main output of the algorithm.

Author(s)

Michael Thrun

See Also

[cutree](#), [hclust](#)

Examples

```
data(Lsun3D)
listofh=HierarchicalClustering(Lsun3D$Data,0,'SingleL')
Tree=listofh$Object
#given colors are per default:
#"magenta" "yellow" "black" "red"
ClusterDendrogram(Tree, 4,main='Single Linkage Clustering')

listofh=HierarchicalClustering(Lsun3D$Data,4)
ClusterCount(listofh$Cls)
#c1 is magenta, c2 is red, c3 is yellow, c4 is black
#because the order of the cluster sizes is
#c1,c3,c4,c2
```

ClusterDistances	<i>ClusterDistances</i>
------------------	-------------------------

Description

Computes intra-cluster distances which are the distance in-between each cluster.

Usage

```
ClusterDistances(FullDistanceMatrix, Cls,
Names, PlotIt = FALSE)

ClusterIntraDistances(FullDistanceMatrix, Cls,
Names, PlotIt = FALSE)
```

Arguments

FullDistanceMatrix	[1:n,1:n] symmetric distance matrix
Cls	[1:n] numerical vector of k classes
Names	Optional [1:k] character vector naming k classes
PlotIt	Optional, Plots if TRUE

Details

Cluster distances are given back as a matrix, one column per cluster and the vector of the full distance matrix without the diagonal elements and the upper half of the symmetric matrix. Details and definitons can be found in [Thrun, 2021].

Value

Matrix [1:m,1:(k+1)] of k clusters, each columns consists of the distances in a cluster, filled up with NaN at the end to be of the same length as the vector of the upper triangle of the complete distance matrix.

Author(s)

Michael Thrun

References

[Thrun, 2021] Thrun, M. C.: The Exploitation of Distance Distributions for Clustering, International Journal of Computational Intelligence and Applications, Vol. 20(3), pp. 2150016, DOI: [doi:10.1142/S1469026821500164](https://doi.org/10.1142/S1469026821500164), 2021.

See Also

[MDplot](#)
[ClusterInterDistances](#)

Examples

```
data(Hepta)
Distance=as.matrix(dist(Hepta$Data))

interdists=ClusterDistances(Distance,Hepta$Cls)
```

ClusterDunnIndex	<i>Dunn Index</i>
------------------	-------------------

Description

Internal (i.e. without prior classification) cluster quality measure called Dunn index for a given clustering published in [Dunn, 1974].

Usage

```
ClusterDunnIndex(Cls,DataOrDistances,

DistanceMethod="euclidean",Silent=TRUE,Force=FALSE,...)
```

Arguments

Cls	[1:n] numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
DataOrDistances	matrix, DataOrDistance[1:n,1:n] symmetric matrix of dissimilarities, if variable unsymmetric DataOrDistance[1:d,1:n] is assumed as a dataset and the euclidean distances are calculated of d variables and n cases
DistanceMethod	Optional, one of 39 distance methods of parDist of package parallelDist, if Data matrix is chosen above
Silent	TRUE: Warnings are shown
Force	TRUE: force computing in case of numerical instability
...	Further arguments passed on to the parDist function, e.g. user_defined distance functions

Details

Dunn index is defined as $Dunn = \min(InterDist) / \max(IntraDist)$. Well seperated clusters have usually a dunn index above 1, for details please see [Dunn, 1974].

Value

List of	
Dunn	scalar, Dunn Index
IntraDist	[1:k] numerical vector of minimal intra cluster distances per given cluster
InterDist	[1:k] numerical vector of minimal inter cluster distances per given cluster

Author(s)

Michael Thrun

References

[Dunn, 1974] Dunn, J. C.: Well_separated clusters and optimal fuzzy partitions, Journal of cybernetics, Vol. 4(1), pp. 95-104. 1974.

Examples

```
data("Hepta")
Cls=kmeansClustering(Hepta$Data,ClusterNo = 7,Type="Hartigan")$Cls
ClusterDunnIndex(Cls,Hepta$Data)

data("Hepta")
ClsWellSeperated=kmeansClustering(Hepta$Data,ClusterNo = 7,Type="Steinley")$Cls
ClusterDunnIndex(ClsWellSeperated,Hepta$Data)
```

ClusterEqualWeighting	<i>ClusterEqualWeighting</i>
-----------------------	------------------------------

Description

Weights clusters equally

Usage

```
ClusterEqualWeighting(Cls, Data, MinClusterSize)
```

Arguments

Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Data	Optional, [1:n,1:d] matrix of dataset consisting of n cases of d-dimensional data points. Every case has d attributes, variables or features.
MinClusterSize	Optional, scalar defining the number of cases m that each cluster should have

Details

Balance clusters such that their sizes are the same by subsampling the larger cluster. If `MinClusterSize` is missing the number of cases per cluster is set to the smallest cluster size. For clusters sizes smaller than `MinClusterSize`, sampling with replacement is turned on, i.e. up sampling. For clusters sizes equal to `MinClusterSize`, no sampling is performed.

Value

List of

BalancedCls	Vector of Cls such that all clusters have the same sizes spezified by <code>MinClusterSize</code>
BalancedInd	index such that <code>BalancedCls = Cls[BalancedInd]</code>
BalancedData	NULL if missing, otherwise, <code>Data[BalancedInd,]</code>

Author(s)

Alfred Ultsch (matlab), reimplemented by Michael Thrun

Examples

```
data(Hepta)
ClusterEqualWeighting(Hepta$Cls,Hepta$Data,5)
```

ClusteringAccuracy	<i>ClusterAccuracy</i>
--------------------	------------------------

Description

ClusterAccuracy

Usage

```
ClusterAccuracy(PriorCls,CurrentCls,Fast=TRUE,K=9)
```

Arguments

PriorCls	Ground truth,[1:n] numerical vector with n numbers defining the classification. It has k unique numbers representing the arbitrary labels of the clustering.
CurrentCls	Main output of the clustering, [1:n] numerical vector with n numbers defining the classification. It has k unique numbers representing the arbitrary labels of the clustering.
Fast	TRUE: useses <code>ClusterAlignLabels</code> in which clue solves a one-to-one assignment problem FALSE: best of all permutation of labels
K	Only for FALSE: Maximal number of classes for computation.

Details

Here, accuracy is defined as the normalized sum over all true positive labeled data points of a clustering algorithm. The best of all permutation of labels with the highest accuracy is selected in every trial because algorithms arbitrarily define the labels [Thrun et al., 2018]. Beware that in contrast to [ClusterMCC](#), the labels can be arbitrary. However, accuracy is a only a valid quality measure if the clusters are balanced (of) nearly equal size). Ohterwise please use [ClusterMCC](#).

In contrast to the F-measure, "Accuracy tends to be naturally unbiased, because it can be expressed in terms of a binomial distribution: A success in the underlying Bernoulli trial would be defined as sampling an example for which a classifier under consideration makes the right prediction. By definition, the success probability is identical to the accuracy of the classifier. The i.i.d. assumption implies that each example of the test set is sampled independently, so the expected fraction of correctly classified samples is identical to the probability of seeing a success above. Averaging over multiple folds is identical to increasing the number of repetitions of the Binomial trial. This does not affect the posterior distribution of accuracy if the test sets are of equal size, or if we weight each estimate by the size of each test set." [Forman/Scholz, 2010]

Value

Single scalar of Accuracy between zero and one

Author(s)

Michael Thrun

References

[Thrun et al., 2018] Michael C. Thrun, Felix Pape, Alfred Ultsch: Benchmarking Cluster Analysis Methods in the Case of Distance and Density-based Structures Defined by a Prior Classification Using PDE-Optimized Violin Plots, ECDA, Potsdam, 2018

[Forman/Scholz, 2010] Forman, G., and Scholz, M.: Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement, ACM SIGKDD Explorations Newsletter, Vol. 12(1), pp. 49-57. 2010.

See Also

[ClusterMCC](#) [ClusterAlignLabels](#)

Examples

```
#Influence of random sets/ random starts on k-means

data('Hepta')
Cls=kmeansClustering(Hepta$Data,7,Type = "Hartigan",nstart=1)
table(Cls$Cls,Hepta$Cls)
ClusterAccuracy(Hepta$Cls,Cls$Cls)

data('Hepta')
```

```

Cls=kmeansClustering(Hepta$Data,7,Type = "Hartigan",nstart=100)
table(Cls$Cls,Hepta$Cls)
ClusterAccuracy(Hepta$Cls,Cls$Cls)

```

ClusterInterDistances *Computes Inter-Cluster Distances*

Description

Computes inter-cluster distances which are the distance between each cluster and all other clusters

Usage

```

ClusterInterDistances(FullDistanceMatrix, Cls,

Names,PlotIt=FALSE)

```

Arguments

FullDistanceMatrix	[1:n,1:n] symmetric distance matrix
Cls	[1:n] numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Names	Optional [1:k] character vector naming k classes
PlotIt	Optional, Plots if TRUE

Details

Cluster distances are given back as a matrix, one column per cluster and the vector of the full distance matrix without the diagonal elements and the upper half of the symmetric matrix. Details and definitons can be found in [Thrun, 2021].

Value

Matrix [1:m,1:(k+1)] of k clusters, each columns consists of the distances between a cluster and all other clusters, filled up with NaN at the end to be of the same lenght as the vector of the upper triangle of the complete distance matrix.

Author(s)

Michael Thrun

References

[Thrun, 2021] Thrun, M. C.: The Exploitation of Distance Distributions for Clustering, International Journal of Computational Intelligence and Applications, Vol. 20(3), pp. 2150016, DOI: [doi:10.1142/S1469026821500164](https://doi.org/10.1142/S1469026821500164), 2021.

See Also

[MDplot](#)
ClusterDistances

Examples

```
data(Hepta)
Distance=as.matrix(dist(Hepta$Data))

interdists=ClusterInterDistances(Distance,Hepta$Cls)
```

ClusterMCC	<i>Matthews Correlation Coefficient (MCC)</i>
------------	---

Description

Matthews correlation coefficient eneralized to the multiclass case (a.k.a. R_K statistic).

Usage

```
ClusterMCC(PriorCls, CurrentCls,Force=TRUE)
```

Arguments

PriorCls	Ground truth,[1:n] numerical vector with n numbers defining the classification. It has k unique numbers representing the labels of the clustering.
CurrentCls	Main output of the clustering, [1:n] numerical vector with n numbers defining the classification. It has k unique numbers representing the labels of the clustering.
Force	Boolean, if is TRUE: forces code even if one or more than one of the k numbers given in PriorCls is missing in CurrentCls or vice versa. In this case, one label per missing number is added ad the end of the vectors.

Details

Contrary to accuracy, the MCC is balanced measure which can be used even if the classes are of very different sizes. When there are more than two labels the MCC will no longer range between -1 and +1. Instead the minimum value will be between -1 and 0 depending on the true distribution. The maximum value is always +1. Beware that in contrast to [ClusterAccuracy](#), the labels cannot be arbitrary. Instead each label of PriorCls and CurrentCls has to be mapped to the same cluster of data points. Typically this has to be ensured manually.

Value

Single scalar of MCC in a range described in details.

Note

If No. of Clusters is not equivalent, internally the number is alligned with zero datapoints belonging to the missing clusters.

Author(s)

Michael Thrun

References

Matthews, B. W.: Comparison of the predicted and observed secondary structure of T4 phage lysozyme, *Biochimica et Biophysica Acta (BBA), Protein Structure*, Vol. 405(2), pp. 442-451, 1975.

Boughorbel, S.B: Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric, *PLOS ONE*, Vol. 12(6), pp. e0177678, 2017.

Chicco, D.; Toetsch, N. and Jurman, G.: The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two_class confusion matrix evaluation. *BioData Mining*. Vol. 14., 2021.

See Also

[ClusterAccuracy](#)

Examples

```
#Beware that algorithm arbitrary defines the labels
data(Hepta)
V=kmeansClustering(Hepta$Data,Type = "Hartigan",7)
table(V$Cls,Hepta$Cls)
#result is only valid if the above issue is resolved manually
ClusterMCC(Hepta$Cls,V$Cls)
```

ClusterNMI

Normalized Mutual Information [Strehl et al., 2003]

Description

Mutual information measures the amount of information that two clusterings share: the higher the MI, the more information shared, indicating a higher similarity between the two clusterings [Cover and Thomas, 1991].

Usage

```
ClusterNMI(Cls1, Cls2,Variant="max")
```

Arguments

Cls1	1:n numerical vector of numbers defining the classification as the main output of the first clustering or trial for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Cls2	1:n numerical vector of numbers defining the classification as the main output of the second clustering algorithm trial for the n cases of data. It has p unique numbers representing the arbitrary labels of the clustering.
Variant	Optional, string, default "max", alternatives are different variants of NMI: ("min", "sqrt", "sum", "joint")

Details

Normalized mutual information scales the result to a fixed range [0, 1], where 0 indicates no mutual information (completely independent clusterings) and 1 indicates perfect correlation (the clusterings are identical). It is normalized by the entropy of both the true labels and the cluster assignments, which makes it more interpretable and less sensitive to the number of clusters. NMI is often used when you have prior knowledge about the number of clusters or when comparing different clustering results.

Value

value of normalized mutual information

Author(s)

Michael Thrun (Wrapper only)

References

[Strehl et al., 2003] Strehl, Alexander; Ghosh, Joydeep: Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions, The Journal of Machine Learning Research, Vol. 3, pp. 583–617. doi:10.1162/153244303321897735, 2003

[Cover and Thomas, 1991] Cover, T, and Thomas, J.A.: Elements of Information Theory. Wiley, 1991.

Examples

```
data(Hepta)
#compare to baseline
Cls2=kmeansClustering(Hepta$Data,7,Type = "Steinley")$Cls
ClusterNMI(Hepta$Cls,Cls2)
#compare different solutions
Cls3=kmeansClustering(Hepta$Data,5)$Cls
ClusterNMI(Cls3,Cls2)
```

ClusterNoEstimation *Estimates Number of Clusters using up to 26 Indicators*

Description

Calculation of up to 26 indicators and the recommendations based on them for the number of clusters in data sets. For a given dataset and clusterings for this dataset, key indicators mentioned in details are calculated and based on this a recommendation regarding the number of clusters is given for each indicator.

An alternative estimation of the cluster number can be done by counting the valleys of the topographic map of the generalized U-Matrix for a specific projection method using the **ProjectionBasedClustering** and **GeneralizedUmatrix** packages on CRAN, see [Thrun/Ultsch, 2021] for details.

Usage

```
ClusterNoEstimation(DataOrDistances, ClsMatrix = NULL, MaxClusterNo,
ClusterIndex = "all", Method = NULL, MinClusterNo = 2,
Silent = TRUE, PlotIt=FALSE, SelectByABC=TRUE, Colorsequence,...)
```

Arguments

DataOrDistances	Either [1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. or Symmetric [1:n,1:n] distance matrix
ClsMatrix	[1:n,1:(MaxClusterNo)] matrix of clusterings each columns is defined as: 1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering. (see also details (2) and (3)), must be specified if method = NULL
MaxClusterNo	Highest number of clusters to be checked
Method	Cluster procedure, with which the clusterings are created (see details (4) for possible methods), must be specified if ClsMatrix = NULL
Optional:	
ClusterIndex	String or vector of strings with the indicators to be calculated (see details (1)), default = "all"
MinClusterNo	Lowest number of clusters to be checked, default = 2
Silent	If TRUE status messages are output, default = FALSE
PlotIt	If TRUE plots fanplot with proposed cluster numbers

SelectByABC	If PlotIt=TRUE, TRUE: Plots group A of ABCanalysis of the most important ones (highest overlap in indicators), FALSE: plots all indicators
Colorsequence	Optional, character vector of sufficient length of colors for the fan plot.If the sequence is too long the first part of the sequence is used.
...	Optional, further arguments used if clustering methods if Method is set.

Details

Each column of `ClsMatrix` has to have at least two unique clusters defined. Otherwise the function will stop.

(1)

The following 26 indicators can be calculated: "ball", "beale", "calinski", "ccc", "cindex", "db", "duda", "dunn", "frey", "friedman", "hartigan", "kl", "marriot", "mcclain", "pseudot2", "ptbiserial", "ratkowsky", "rubin", "scott", "sdbw", "sdindex", "silhouette", "ssi", "tracew", "trcovw", "xuindex".

These can be specified individually or as a vector via the parameter `index`. If you enter 'all', all key figures are calculated.

(2)

The indicators `kl`, `duda`, `pseudot2`, `beale`, `frey` and `mcclain` require a clustering for `MaxClusterNo+1` clusters. If these key figures are to be calculated, this clustering must be specified in `cls`.

(3)

The indicator `kl` requires a clustering for `MinClusterNo-1` clusters. If this key figure is to be calculated, this clustering must also be specified in `cls`. For the case `MinClusterNo = 2` no clustering for 1 has to be given.

(4)

The following methods can be used to create clusterings:

"kmeans", "DBSclustering", "DivisiveAnalysisClustering", "FannyClustering", "ModelBasedClustering", "SpectralClustering" or all methods found in [HierarchicalClustering](#).

(5)

The indicators `duda`, `pseudot2`, `beale` and `frey` are only intended for use in hierarchical cluster procedures.

If a distances matrix is given, then **ProjectionBasedClustering** is required to be accessible.

Value

Indicators	A table of the calculated indicators except Duda, Pseudot2 and Beale
ClusterNo	The recommended number of clusters for each calculated indicator
ClsMatrix	[1:n,MinClusterNo:(MaxClusterNo)] Output of the clusterings used for the calculation
HierarchicalIndicators	Either NULL or the values for the indicators Duda, Pseudot2 and Beale in case of hierarchical cluster procedures, if calculated

Note

Code of "calinski", "cindex", "db", "hartigan", "ratkowsky", "scott", "marriot", "ball", "trcovw", "tracew", "friedman", "rubin", "ssi" of package cclust ist adapted for the purpose of this function.

Colorsequence works if **DataVisualizations** 1.1.13 is installed (currently only on github available).

Author(s)

Peter Nahrgang, revised by Michael Thrun (2021)

References

Charrad, Malika, et al. "Package 'NbClust', J. Stat. Soft Vol. 61, pp. 1-36, 2014.

Dimtriadou, E. "cclust: Convex Clustering Methods and Clustering Indexes." R package version 0.6-16, URL <https://CRAN.R-project.org/package=cclust>, 2009.

[Thrun/Ultsch, 2021] Thrun, M. C., and Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, Artificial Intelligence, Vol. 290, pp. 103237, doi:10.1016/j.artint.2020.103237, 2021.

Examples

```
# Reading the iris dataset from the standard R-Package datasets
data <- as.matrix(iris[,1:4])
MaxClusterNo = 7
# Creating the clusterings for the data set
#(here with method complete) for the number of clusters 2 to 8
hc <- hclust(dist(data), method = "complete")
clsm <- matrix(data = 0, nrow = dim(data)[1],

ncol = MaxClusterNo)
for (i in 2:(MaxClusterNo+1)) {
  clsm[,i-1] <- cutree(hc,i)
}

# Calculation of all indicators and recommendations for the number of clusters
indicatorsList=ClusterNoEstimation(Data = data,

ClsMatrix = clsm, MaxClusterNo = MaxClusterNo)

# Alternatively, the same calculation as above can be executed with the following call
ClusterNoEstimation(Data = data, MaxClusterNo = 7, Method = "CompleteL")
# In this variant, the function clusterumbers also takes over the clustering
```

ClusterNormalize

Cluster Normalize

Description

Values in Cls are consistently recoded to positive consecutive integers

Usage

```
ClusterNormalize(Cls)
```

Arguments

Cls [1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.

Details

For recoding depending on cluster size please see [ClusterRenameDescendingSize](#).

Value

The renamed classification. A vector of clusters recoded to positive consecutive integers.

Author(s)

.

See Also

[ClusterRenameDescendingSize](#)

Examples

```
data('Lsun3D')
Cls=Lsun3D$Cls
#not descending cluster numbers
Cls[Cls==1]=543
Cls[Cls==4]=1

# Now ordered consecutively
ClusterNormalize(Cls)
```

Description

This function uses a projection method to perform dimensionality reduction (DR) on order to visualize the data as 3D data points colored by a clustering.

Usage

```
ClusterPlotMDS(DataOrDistances, Cls, main = "Clustering",
  DistanceMethod = "euclidean", OutputDimension = 3,
  PointSize=1, Plotter3D="rgl", Colorsequence, ...)
```

Arguments

DataOrDistances	Either nonsymmetric [1:n,1:d] datamatrix of n cases and d features or symmetric [1:n,1:n] distance matrix
Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
main	String, title of plot
DistanceMethod	Method to compute distances, default "euclidean"
OutputDimension	Either two or three depending on user choice
PointSize	Scalar defining the size of points
Plotter3D	In case of 3 dimensions, choose either "plotly" or "rgl",
Colorsequence	[1:k] character vector of colors, per default the colorsequence defined in the DataVisualizations is used
...	Please see Plot3D in DataVisualizations

Details

If dataset has more than 3 dimensions, mds is performed as defined in the **smacof** [De Leeuw/Mair, 2011]. If **smacof** package is not installed, classical metric MDS (see Def. in [Thrun, 2018]) is performed. In both cases, the first OutputDimension are visualized. Points are colored by the labels (Cls).

In the special case that the dataset has not more than 3 dimensions, all dimensions are visualized and no DR is performed.

Value

The rgl or plotly plot handler depending on Plotter3D

Note

If **DataVisualizations** is not installed a 2D plot using native plot function is shown.

If **MASS** is not installed, classical metric MDS is used, see [Thrun, 2018] for definition.

Author(s)

Michael Thrun

References

[De Leeuw/Mair, 2011] De Leeuw, J., & Mair, P.: Multidimensional scaling using majorization: SMACOF in R, Journal of statistical Software, Vol. 31(3), pp. 1-30. 2011.

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, ISBN: 978-3-658-20539-3, Heidelberg, 2018.

See Also

[Plot3D](#)

Examples

```
data(Hepta)
ClusterPlotMDS(Hepta$Data,Hepta$Cls)

data(Leukemia)
ClusterPlotMDS(Leukemia$DistanceMatrix,Leukemia$Cls)
```

ClusterRedefine	<i>Redfines Clustering</i>
-----------------	----------------------------

Description

Redfines some or all Clusters of Clustering such that the names of the numerical vectors are defined by

Usage

```
ClusterRedefine(Cls, NewLabels,OldLabels,Silent=FALSE)
```

Arguments

Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
NewLabels	[1:p], p<=k labels (identifiers) of clusters to be changed with
OldLabels	Optional, [1:p], p<=k labels(identifiers) of clusters to be changed, default [1:k] unique cluster Ids of Cls
Silent	Optional, TRUE: No warnign messages are printed out

Details

The same ordering of NewLabels and OldLabels is assumend, i.e., the mapping is defined by OldLabels[i] -> NewLabels[i] with i in [1:p]. NewLabels can also be a vector for strings, for example for plotting.

Value

Cls[1:n] numerical vector named after the row names of data

Author(s)

Michael Thrun

Examples

```
data('Lsun3D')
Cls=Lsun3D$Cls
Data=Lsun3D$Data#
#prior
ClsNew=unique(Cls)+10
#Redfined Clustering
NewCls=ClusterRedefine(Cls,ClsNew)

table(Cls,NewCls)

#require(DataVisualizations)
n=length(unique(Cls))
NewCls=ClusterRedefine(Cls,LETTERS[1:n])
#DataVisualizations package required
if(requireNamespace("DataVisualizations"))
  DataVisualizations::Classplot(Data[,1],Data[,2],
    Cls,Names=NewCls,Plotter="ggplot",Size =1.5)
```

ClusterRename	<i>Renames Clustering</i>
---------------	---------------------------

Description

Renames Clustering such that the names of the numerical vectors are the row names of DataOrDistances

Usage

```
ClusterRename(Cls, DataOrDistances)
```

Arguments

- Cls 1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
- DataOrDistances Either nonsymmetric [1:n,1:d] datamatrix of n cases and d features or symmetric [1:n,1:n] distance matrix

Details

If DataOrDistances is missing or if inconsistent length, nothing is done.

Value

Cls[1:n] numerical vector named after the row names of data

Author(s)

Michael Thrun

Examples

```
data('Hepta')
Cls=Hepta$Cls
Data=Hepta$Data#
#prior
Cls
#Named Clustering
ClusterRename(Cls,Data)
```

ClusterRenameDescendingSize

Cluster Rename Descending Size

Description

Renames the clusters of a classification in descending order.

Usage

```
ClusterRenameDescendingSize(Cls,
  ProvideClusterNames=FALSE)
```

Arguments

Cls	[1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
ProvideClusterNames	TRUE: Provides in seperate output new and old k numbers, FALSE: simple output

Details

Beware: output changes in this function depending on ProvideClusterNames in order to be congruent to prior code in a large varierity of other packages.

Value

ProvideClusterNames==FALSE:

RenamedCls The renamed classification. A vector of clusters, were the largest cluster is C1 and so forth

ProvideClusterNames==TRUE: List V with

RenamedCls The renamed classification. A vector of clusters, were the largest cluster is C1 and so forth

ClusterName [1:k,1:2] matrix of k new numbers and prior numbers

Author(s)

Michael Thrun, Alfred Ultsch

See Also

[ClusterNormalize](#)

Examples

```
data('Lsun3D')
Cls=Lsun3D$Cls
#not desceending cluster numbers
Cls[Cls==1]=543
Cls[Cls==4]=1

# Now ordered per cluster size and descending
ClusterRenameDescendingSize(Cls)
```

ClusterShannonInfo	<i>Shannon Information</i>
--------------------	----------------------------

Description

Shannon Information [Shannon, 1948] for each column in ClsMatrix.

Usage

ClusterShannonInfo(ClsMatrix)

Arguments

ClsMatrix [1:n,1:C] matrix of C clusterings each columns is defined as:
1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.

Details

$\text{Info}[1:d] = \text{sum}(-p * \log(p)/\text{MaxInfo})$ for all unique cases with probability p in $\text{ClsMatrix}[,c]$ for a column with k clusters $\text{MaxInfo} = -(1/k)*\log(1/k)$

The function measures how balanced and diverse the clusters are in each clustering solution (column):

Balanced clusters (e.g., 3 clusters with equal size) => entropy close to 1.

Skewed clusters (e.g., 95 percent in one cluster) => entropy closer to 0.

It normalizes entropy by the theoretical max entropy for k clusters, giving a percentage of "information efficiency" in the clustering.

Value

Info	[1:max.nc,1:C] matrix of Shannin informaton as defined in details, each column represents one Cls of ClsMatrix,each row yields the information of one cluster up the ClusterNo k , if $k < \text{max.nc}$ (highest number of clusters) then NaN are filled.
ClusterNo	Number of Clusters k found for each Cls respectively
MaxInfo	max per column of Info
MinInfo	min per column of Info
MedianInfo	median per column of Info
MeanInfo	mean per column of Info

Author(s)

Michael Thrun

References

[Shannon, 1948] Shannon, C. E.: A Mathematical Theory of Communication, Bell System Technical Journal, Vol. 27(3), pp. 379-423. doi doi:10.1002/j.1538-7305.1948.tb01338.x, 1948.

Examples

```
# Reading the iris dataset from the standard R-Package datasets
data <- as.matrix(iris[,1:4])
max.nc = 7
# Creating the clusterings for the data set
#(here with method complete) for the number of classes 2 to 8
hc <- hclust(dist(data), method = "complete")
clsm <- matrix(data = 0, nrow = dim(data)[1],

ncol = max.nc)
for (i in 2:(max.nc+1)) {
  clsm[,i-1] <- cutree(hc,i)
}

ClusterShannonInfo(clsm)
```

ClusterUpsamplingMinority

Cluster Up Sampling using SMOTE for minority cluster

Description

Wrapper for one specific internal function of L. Torgo who implemented there the relevant part of the SMOTE algorithm [Chawla et al., 2002].

Usage

```
ClusterUpsamplingMinority(Cls, Data, MinorityCluster,
Percentage = 200, knn = 5, PlotIt = FALSE)
```

Arguments

Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Data	[1:n,1:d] datamatrix of n cases and d features
MinorityCluster	scalar defining the number of the cluster to be upsampeled
Percentage	percentage above 100 of who many samples should be taken
knn	k nearest neighbors of SMOTE algorithm
PlotIt	TRUE: plots the result using ClusterPlotMDS

Details

the number of items m is defined by the scalar Percentage and the up sampling is combined with the Data and the Cls to DataExt and ClsExt such that the sample is placed thereafter.

Value

List with	
ClsExt	1:(n+m) numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
DataExt	[1:(n+m),1:d] datamatrix of n cases and d features

.

Author(s)

L. Torgo

References

[Chawla et al., 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P.: SMOTE: synthetic minority over-sampling technique, Journal of artificial intelligence research, Vol. 16, pp. 321-357. 2002.

Examples

```
data(Lsun3D)
Data=Lsun3D$Data
Cls=Lsun3D$Cls
table(Cls)

V=ClusterUpsamplingMinority(Cls,Data,4,1000)
table(V$ClsExt)
```

ConsensusClustering	<i>Consensus Clustering</i>
---------------------	-----------------------------

Description

Performs consensus clustering using the **ConsensusClusterPlus** package. The input may be a data matrix or a symmetric distance matrix.

Usage

```
ConsensusClustering(
  Data,
  ClusterNo = NULL,
  PlotIt = FALSE,
  PlotConsensus = FALSE,
  ...
)
```

Arguments

Data	Data set with n observations and d features
ClusterNo	Integer or NULL. Number of clusters to search for. If NULL, the complete object returned by ConsensusClusterPlus() is returned.
PlotIt	Logical. If TRUE, plots the final clustering using ClusterPlotMDS(). Default is FALSE.
PlotConsensus	FALSE, NULL, "png", "pdf", or "pngBMP".
...	Further arguments passed to ConsensusClusterPlus::ConsensusClusterPlus().

Value

If ClusterNo is not NULL, a list with components:

Cls	Numeric vector of cluster assignments.
Object	Complete object returned by <code>ConsensusClusterPlus::ConsensusClusterPlus()</code> .

If ClusterNo = NULL, the complete result from `ConsensusClusterPlus::ConsensusClusterPlus()` is returned directly.

If **ConsensusClusterPlus** is unavailable, a fallback list is returned with all observations assigned to cluster 1 and an explanatory message in Object.

Author(s)

Michael Thrun

References

Wilkerson, D. M. and Hayes, N. D.: ConsensusClusterPlus: a class discovery tool with confidence assessments and item tracking, *Bioinformatics*, Vol. **26**(12), pp. 1572–1573, 2010.

Examples

```
# Requires the Bioconductor package ConsensusClusterPlus

# set.seed(1)
# Data <- matrix(rnorm(100), ncol = 5)
#
# result <- ConsensusClustering(
#   Data = Data,
#   ClusterNo = 3,
#   PlotIt = FALSE,
#   reps = 100,
#   clusterAlg = "hc",
#   distance = "euclidean",
#   seed = 123
# )
#
# result$Cls
```

CrossEntropyClustering

Cross-Entropy Clustering

Description

Cross-entropy clustering published by [Tabor/Spurek, 2014] and implemented by [Spurek et al., 2017].

Usage

```
CrossEntropyClustering(Data, ClusterNo, PlotIt=FALSE, ...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

Contrary to most of the other implemented algorithms in this package, the results on the easiest clustering challenge of Hepta are unstable for cross-entropy clustering in the sense that the clustering is not always correct. Reproducibility experiments should be performed (see [Tabor/Spurek, 2014]).

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Spurek et al., 2017] Spurek, P., Kamieniecki, K., Tabor, J., Misztal, K., & Śmieja, M.: R package cec, Neurocomputing, Vol. 237, pp. 410-413. 2017.

[Tabor/Spurek, 2014] Tabor, J., & Spurek, P.: Cross-entropy clustering, Pattern Recognition, Vol. 47(9), pp. 3046-3059. 2014.

Examples

```
data('Hepta')
out=CrossEntropyClustering(Hepta$Data, ClusterNo=7, PlotIt=FALSE)
```

DBSCAN	<i>DBSCAN</i>
--------	---------------

Description

Density-Based Spatial Clustering of Applications with Noise of [Ester et al., 1996].

Usage

```
DBSCAN(Data,Radius,minPts,Rcpp=TRUE,  
  
PlotIt=FALSE,UpperLimitRadius,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
Radius	Eps [Ester et al., 1996, p. 227] neighborhood in the R-ball graph/unit disk graph), size of the epsilon neighborhood. If NULL, automatic estimation is performed using insights of [Ultsch, 2005].
minPts	Number of minimum points in the eps region (for core points). In principle minimum number of points in the unit disk, if the unit disk is within the cluster (core) [Ester et al., 1996, p. 228]. If NULL, 2.5 percent of points is selected.
Rcpp	If TRUE: fast Rcpp implementation of mlpack is used. FALSE uses dbscan package.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in CIs
UpperLimitRadius	Limit for radius search, experimental
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

To simplify, the radius works as follows. When DBSCAN looks at a data point, it draws a circle around it with a Radius. If there are enough data points inside that circle, they become a group. Choosing the right Radius depends on how spread out your data points are.

This is the minimum number of data points minPts that should be inside the circle for DBSCAN to consider them a group. If there are fewer data points than this number, the data point might be considered as noise or an outlier. The minPts value helps control how strict or lenient the algorithm is when forming groups.

Value

List of

Cls [1:n] numerical vector defining the clustering; this classification is the main output of the algorithm. Points which cannot be assigned to a cluster will be reported as members of the noise cluster with 0.

Object Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., & Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, Proc. Kdd, Vol. 96, pp. 226-231, 1996.

[Ultsch, 2005] Ultsch, A.: Pareto density estimation: A density estimation for knowledge discovery, In Baier, D. & Wernicke, K. D. (Eds.), Innovations in classification, data science, and information systems, (Vol. 27, pp. 91-100), Berlin, Germany, Springer, 2005.

Examples

```
data('Hepta')

out=DBSCAN(Hepta$Data,Radius=NULL,minPts=NULL,PlotIt=FALSE)

## Not run:
#search for right parameter setting by grid search
data("WingNut")
Data = WingNut$Data
DBSGrid <- expand.grid(
  Radius = seq(from = 0.01, to = 0.3, by = 0.02),
  minPts = seq(from = 1, to = 50, by = 2)
)
BestAcc = c()
for (i in seq_len(nrow(DBSGrid))) {
  parameters <- DBSGrid[i,]
  Cls9 = DBSCAN(
    Data,
    minPts = parameters$minPts,

    Radius = parameters$Radius,
    PlotIt = F,

    UpperLimitRadius = parameters$Radius
  )$Cls
  if (length(unique(Cls9)) < 5)
    BestAcc[i] = ClusterAccuracy(WingNut$Cls,
                                Cls9) * 100
  else
```

```

        BestAcc[i] = 50
    }
    max(BestAcc)
    which.max(BestAcc)
    parameters <- DBSGrid[13,]

    Cls9 = DBSCAN(
        Data,
        minPts = parameters$minPTs,
        Radius = parameters$Radius,
        UpperLimitRadius = parameters$Radius,
        PlotIt = TRUE
    )$Cls

    ## End(Not run)

```

DBScusteringAndVisualization

Databionic Swarm (DBS) Clustering and Visualization

Description

Swarm-based clustering by exploiting self-organization, emergence, swarm intelligence and game theory published in [Thrun/Ultsch, 2021].

Usage

```

DatabionicSwarmClustering(DataOrDistances, ClusterNo = 0,
    StructureType = TRUE, DistancesMethod = NULL,
    PlotTree = FALSE, PlotMap = FALSE, PlotIt=FALSE, Parallel = FALSE)

```

Arguments

DataOrDistances	Either nonsymmetric [1:n,1:d] numerical matrix of a dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. or symmetric [1:n,1:n] distance matrix, e.g. as <code>matrix(dist(Data,method))</code>
ClusterNo	Number of Clusters, if zero a the topographic map is plotted. Number of valleys equals number of clusters.
StructureType	Either TRUE or FALSE, has to be tested against the visualization. If colored points of clusters a divided by mountain ranges, parameter is incorrect.
DistancesMethod	Optional, if data matrix given, annon Euclidean distance can be selected
PlotTree	Optional, if TRUE: dendrogram is plotted.

PlotMap	Optional, if TRUE: topographic map is plotted if GeneralizedUmatrix is installed. See details.
PlotIt	Default: FALSE, If TRUE and dataset of [1:n,1:d] dimensions then a plot of the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls will be generated.
Parallel	FALSE: default implementation, TRUE faster Cpp parallel implementation, for this the subsequent packages have to be installed from github, as they are not available on CRAN yet.

Details

This function does not enable the user first to project the data and then to test the Boolean parameter defining the type of structure contrary to the **DatabionicSwarm** which is an inappropriate approach in case of exploratory data analysis.

Instead, this function is implemented for the purpose of automatic benchmarking because in such a case nobody will investigate many trials with one visualization per trial.

If one would like to perform a clustering exploratively (in the sense that a prior clustering is not given for evaluation purposes), then please use the **DatabionicSwarm** package directly and read the vignette there. Databionic swarm is like k-means a stochastic algorithm meaning that the clustering and visualization may change between trials.

If PlotMap==TRUE and ClusterNo=0 a topview of the topographic map is shown, in which the points are not labeled, i.e. colored by the same color. If PlotMap==TRUE and ClusterNo>0, then the points are colored by their cluster labels. If you would like to look an 3D topographic map that can be interactively rotated or use 3D printing of the high-dimensional structures [Thrun et al., 2016], please see [plotTopographicMap](#) for further details.

Value

List of

Cls	1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.
Object	List of further output of DBS

Note

Current implementation is not efficient enough to cluster more than N=4000 cases as in that case it takes longer than a day for a result.

Author(s)

Michael Thrun

References

[Thrun/Ultsch, 2021] Thrun, M. C., and Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, Artificial Intelligence, Vol. 290, pp. 103237, doi:10.1016/j.artint.2020.103237, 2021.

[Thrun/Ultsch, 2021] Thrun, M. C., & Ultsch, A.: Swarm Intelligence for Self-Organized Clustering (Extended Abstract), in Bessiere, C. (Ed.), 29th International Joint Conference on Artificial Intelligence (IJCAI), Vol. IJCAI-20, pp. 5125–5129, doi:10.24963/ijcai.2020/720, Yokohama, Japan, Jan., 2021.

[Thrun et al., 2016] Thrun, M. C., Lerch, F., Lötsch, J., & Ultsch, A. : Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Vol. 24, Plzen, 2016.

See Also

[Pswarm](#), [DBScustering](#), [GeneratePswarmVisualization](#)

Examples

```
# Generate random but small non-structured data set
data = cbind(
  sample(1:100, 300, replace = TRUE),
  sample(1:100, 300, replace = TRUE),
  sample(1:100, 300, replace = TRUE)
)
# Make sure there are no structures
# (sample size is small and still could generate structures randomly)
if(requireNamespace('DataVisualizations',quietly = TRUE)){
  Data = DataVisualizations::RobustNormalization(data, Centered = TRUE)
  #DataVisualizations::Plot3D(Data)

  # No structures are visible
  # Topographic map looks like "egg carton"
  # with every point in its own valley
  ClsV = DatabionicSwarmClustering(Data, 0, PlotMap = TRUE)
}else{
  # only for testing purposes of CRAN!
  # in case CRAN tests with no suggest packages available
  # please use always some kind of standardization!
  ClsV = DatabionicSwarmClustering(data, 0, PlotMap = TRUE)
}

# Distance based cluster structures
# 7 valleys are visible, thus ClusterNo=7

data(Hepta)
#DataVisualizations::Plot3D(Hepta$Data)

ClsV = DatabionicSwarmClustering(Hepta$Data, 0, PlotMap = TRUE)

#entagled, complex, and non-linear seperable structures
```

```
## Not run:
#takes too long for CRAN tests
data(Chainlink)
#DataVisualizations::Plot3D(Chainlink$Data)

# 2 valleys are visible, thus ClusterNo=2
ClsV = DatabionicSwarmClustering(Chainlink$Data, 0, PlotMap = TRUE)

# Experiment with parameter StructureType only
# reveals that clustering is appropriate
# if StructureType=FALSE
ClsV2 = DatabionicSwarmClustering(Chainlink$Data,
                                   2,
                                   StructureType = FALSE,
                                   PlotMap = TRUE)

# Here clusters (colored points)
# are not separated by valleys
ClsV = DatabionicSwarmClustering(Chainlink$Data,
                                   2,
                                   StructureType = TRUE,
                                   PlotMap = TRUE)

## End(Not run)
```

DensityPeakClustering *Density Peak Clustering algorithm using the Decision Graph*

Description

Density peaks clustering of [Rodriguez/Laio, 2014] is here implemented by [Pedersen et al., 2017] with estimation of [Wang et al, 2015] meaning its non adaptive.

Usage

```
DensityPeakClustering(DataOrDistances, Rho,Delta,Dc,Knn=7,
  DistanceMethod = "euclidean", PlotIt = FALSE, Data, ...)
```

Arguments

DataOrDistances	Either [1:n,1:n] symmetric distance matrix or [1:n,1:d] non symmetric data matrix of n cases and d variables
Rho	Local density of a point, see [Rodriguez/Laio, 2014] for explanation
Delta	Minimum distance between a point and any other point, see [Rodriguez/Laio, 2014] for explanation

Dc	Optional, cutoff distance, will either be estimated by [Pedersen et al., 2017] or [Wang et al, 2015] (see example below)
Knn	Optional k nearest neighbors
DistanceMethod	Optional distance method of data, default is euclid, see parDist for details
PlotIt	Optional TRUE: Plots 2d or 3d result with clustering
Data	[1:n,1:d] data matrix in the case that DataOrDistances is missing and partial matching does not work.
...	Optional, further arguments for densityClust

Details

The densityClust algorithm does not decide the k number of clusters, this has to be done by the parameter setting.

The plot shows the density peaks (Cluster centers). Set Rho and Delta as boundaries below the number of relevant cluster centers for your problem. (see example below).

Value

If Rho and Delta are set:

list of

Cls [1:n numerical vector of numbers defining the classification as the main output of the clustering algorithm for the n cases of data. It has k unique numbers representing the arbitrary labels of the clustering.

Object output of [Pedersen et al., 2017] algorithm

If Rho and Delta are missing:

p object of [plot_ly](#) for the decision graph is returned

Author(s)

Michael Thrun

References

[Wang et al., 2015] Wang, S., Wang, D., Li, C., & Li, Y.: Comment on "Clustering by fast search and find of density peaks", arXiv preprint arXiv:1501.04267, 2015.

[Pedersen et al., 2017] Thomas Lin Pedersen, Sean Hughes and Xiaojie Qiu: densityClust: Clustering by Fast Search and Find of Density Peaks. R package version 0.3. <https://CRAN.R-project.org/package=densityClust>, 2017.

[Rodriguez/Laio, 2014] Rodriguez, A., & Laio, A.: Clustering by fast search and find of density peaks, Science, Vol. 344(6191), pp. 1492-1496. 2014.

See Also

[densityClust](#)

Examples

```

data(Hepta)
H=EntropyOfDataField(Hepta$Data, seq(from=0,to=1.5,by=0.05),PlotIt=FALSE)
Sigmin=names(H)[which.min(H)]
Dc=3/sqrt(2)*as.numeric(names(H)[which.min(H)])
# Look at the plot and estimate rho and delta

DensityPeakClustering(Hepta$Data, Knn = 7,Dc=Dc)
Cls=DensityPeakClustering(Hepta$Data,Dc=Dc,Rho = 0.028,

Delta = 22,Knn = 7,PlotIt = TRUE)$Cls

```

DivisiveAnalysisClustering

Large DivisiveAnalysisClustering Clustering

Description

Divisive Analysis Clustering (diana) of [Rousseeuw/Kaufman, 1990, pp. 253-279]

Usage

```

DivisiveAnalysisClustering(DataOrDistances, ClusterNo,

PlotIt=FALSE,Standardization=TRUE,PlotTree=FALSE,Data,...)

```

Arguments

DataOrDistances	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be build by the algorithm. if ClusterNo=0 and PlotTree=TRUE, the dendrogram is generated instead of a clustering to estimate the numbers of clusters.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
Standardization	DataOrDistances Is standardized before calculating the dissimilarities. Measurements are standardized for each variable (column), by subtracting the variable's mean value and dividing by the variable's mean absolute deviation.If DataOrDistances Is already a distance matrix, then this argument will be ignored.
PlotTree	TRUE: Plots the dendrogram, FALSE: no plot
Data	[1:n,1:d] data matrix in the case that DataOrDistances is missing and partial matching does not work.

... Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of

`Cls` [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.

`Dendrogram` Dendrogram of hierarchical clustering algorithm

`Object` Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Rousseeuw/Kaufman, 1990] Rousseeuw, P. J., & Kaufman, L.: Finding groups in data, Belgium, John Wiley & Sons Inc., ISBN: 0471735787, doi: 10.1002/9780470316801, Online ISBN: 9780470316801, 1990.

Examples

```
data('Hepta')
CA=DivisiveAnalysisClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)

print(CA$Object)
plot(CA$Object)
ClusterDendrogram(CA$Dendrogram,7,main='DIANA')
```

EngyTime

EngyTime introduced in [Baggenstoss, 2002].

Description

Gaussian mixture. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("EngyTime")
```

Details

Size 4096, Dimensions 2, stored in `EngyTime$Data`

Classes 2, stored in `EngyTime$Cls`

References

- [Baggenstoss, 2002] Baggenstoss, P. M.: Statistical modeling using gaussian mixtures and hmms with matlab, Naval Undersea Warfare Center, Newport RI, 2002.
- [Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

Examples

```
data(EngyTime)
str(EngyTime)
```

EntropyOfDataField *Entropy Of a Data Field [Wang et al., 2011].*

Description

Calculates the Potential Entropy Of a Data Field for a given ranges of impact factors sigma

Usage

```
EntropyOfDataField(Data,
    sigmarange = c(0.01, 0.1, 0.5, 1, 2, 5, 8, 10, 100)
    , PlotIt = FALSE)
```

Arguments

Data	[1:n,1:d] data matrix
sigmarange	Numeric vector [1:s] of relevant sigmas
PlotIt	FALSE: disable plot, TRUE: Plot with upper boundary of H after [Wang et al., 2011].

Details

In theory there should be a curve with a clear minimum of Entropy [Wang et al.,2011]. Then the choice for the impact factor sigma is the minimum of the entropy to define the correct data field. It follows, that the influence radius is $3/\sqrt{2} \cdot \sigma$ (3B rule of gaussian distribution) for clustering algorithms like density peak clustering [Wang et al.,2011].

Value

[1:s] named vector of the Entropy of data field. The names are the impact factor sigma.

Author(s)

Michael Thrun

References

[Wang et al., 2015] Wang, S., Wang, D., Li, C., & Li, Y.: Comment on "Clustering by fast search and find of density peaks", arXiv preprint arXiv:1501.04267, 2015.

[Wang et al., 2011] Wang, S., Gan, W., Li, D., & Li, D.: Data field for hierarchical clustering, International Journal of Data Warehousing and Mining (IJDWM), Vol. 7(4), pp. 43-63. 2011.

Examples

```
data(Hepta)
H=EntropyOfDataField(Hepta$Data,PlotIt=FALSE)
Sigmin=names(H)[which.min(H)]
Dc=3/sqrt(2)*as.numeric(names(H)[which.min(H)])
```

EstimateRadiusByDistance

Estimate Radius By Distance

Description

Published in [Thrun et al, 2016] for the case of automatically estimating the radius of the P-matrix. Can also be used to estimate the radius parameter for distance based clustering algorithms.

Usage

```
EstimateRadiusByDistance(DistanceMatrix)
```

Arguments

DistanceMatrix [1:n,1:n] symmetric distance Matrix of n cases

Details

For density-based clustering algorithms like [DBSCAN](#) it is not always usefull.

Value

Numerical scalar defining the radius

Note

Symmetric matrix is assumed.

Author(s)

Michael Thrun

References

[Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A.: Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Vol. 24, pp. 7-16, Plzen, <http://wscg.zcu.cz/wscg2016/short/A43-full.pdf>, 2016.

See Also[GeneratePmatrix](#)**Examples**

```
data('Hepta')
DistanceMatrix=as.matrix(dist(Hepta$Data))
Radius=EstimateRadiusByDistance(DistanceMatrix)
```

FannyClustering

*Fuzzy Analysis Clustering [Rousseeuw/Kaufman, 1990, p. 253-279]***Description**

...

Usage

```
FannyClustering(DataOrDistances,ClusterNo,
PlotIt=FALSE,Standardization=TRUE,...)
```

Arguments

DataOrDistances

[1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix

ClusterNo A number k which defines k different clusters to be build by the algorithm.

PlotIt Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls

Standardization

DataOrDistances is standardized before calculating the dissimilarities. Measurements are standardized for each variable (column), by subtracting the variable's mean value and dividing by the variable's mean absolute deviation. If DataOrDistances is already a distance matrix, then this argument will be ignored.

... Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

...

Value

List of
Cls [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.
Object Object defined by clustering algorithm as the second output of this algorithm

Author(s)

Michael Thrun

References

[Rousseeuw/Kaufman, 1990] Rousseeuw, P. J., & Kaufman, L.: Finding groups in data, Belgium, John Wiley & Sons Inc., ISBN: 0471735787, doi: 10.1002/9780470316801, Online ISBN: 9780470316801, 1990.

Examples

```
data('Hepta')
out=FannyClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

GapStatistic	<i>Gap Statistic</i>
--------------	----------------------

Description

Gap Statistic

Usage

```
GapStatistic(Data, ClusterNoMax, ClusterFun, ...)
```

Arguments

Data [1:n,1:d] data matrix
ClusterNoMax max no of clusters to be investigated
ClusterFun which clustering algorithm to investigate
... further arguments passed on

Details

does not work on hepta, see example

Value

tobedocumented

Note

Wrapper only

Author(s)

Michael Thrun

References

Tibshirani, R., Walther, G. and Hastie, T: Estimating the number of data clusters via the Gap statistic, Journal of the Royal Statistical Society B, Vol. 63, pp. 411-423, 2003.

Examples

```
data(Hepta)
#GapStatistic(Hepta$Data,10,ClusterFun = kmeans)
```

GenieClustering	<i>Genie Clustering by Gini Index</i>
-----------------	---------------------------------------

Description

Outlier Resistant Hierarchical Clustering Algorithm of [Gagolewski/Bartoszuk, 2016].

Usage

```
GenieClustering(DataOrDistances, ClusterNo = 0, ColorTreshold = 0,
DistanceMethod="euclidean",PlotIt=FALSE,...)
```

Arguments

- DataOrDistances [1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
- ClusterNo A number k which defines k different clusters to be build by the algorithm.
- DistanceMethod See [parDist](#), for example 'euclidean', 'mahalanobis', 'manhattan' (cityblock), 'fJaccard', 'binary', 'canberra', 'maximum'. Any unambiguous substring can be given.

ColorTreshold	Draws cutline w.r.t. dendrogram y-axis (height), height of line as scalar should be given
...	furter argument to genie like:
	thresholdGini Single numeric value in [0,1], threshold for the Gini index, 1 gives the standard single linkage algorithm
PlotIt	If TRUE and ClusterNo>0 plots dendrogramm using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogramm using base R plot FALSE: No plot

Details

Wrapper for Genie algorithm.

Value

List of

Cls	If, ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise for ClusterNo=0: NULL
Dendrogram	Dendrogram of hierarchical clustering algorithm
Object	Ultrametric tree of hierarchical clustering algorithm

Author(s)

Michael Thrun

References

[Gagolewski/Bartoszuk, 2016] Gagolewski M., Bartoszuk M., Cena A., Genie: A new, fast, and outlier-resistant hierarchical clustering algorithm, Information Sciences, Vol. 363, pp. 8-23, 2016.

See Also

[HierarchicalClustering](#)

Examples

```
data('Hepta')

Clust=GenieClustering(Hepta$Data,ClusterNo=7)
```

 GolfBall

GolfBall introduced in [Ultsch, 2005]

Description

No clusters at all. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("GolfBall")
```

Details

Size 4002, Dimensions 3, stored in GolfBall\$Data

Classes 1, stored in GolfBall\$Cls

References

[Ultsch, 2005] Ultsch, A.: Clustering with SOM: U* C, Proc. Proceedings of the 5th Workshop on Self-Organizing Maps, Vol. 2, pp. 75-82, 2005.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, [doi:10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(GolfBall)
str(GolfBall)
```

 HCLclustering

On-line Update (Hard Competitive learning) method

Description

Hard Competitive learning clustering published by [Ripley, 2007].

Usage

```
HCLclustering(Data, ClusterNo, PlotIt=FALSE, ...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be build by the algorithm.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Dimitriadou, 2002] Dimitriadou, E.: cclust-convex clustering methods and clustering indexes. R package, 2002,
- [Ripley, 2007] Ripley, B. D.: Pattern recognition and neural networks, Cambridge university press, ISBN: 0521717701, 2007.

Examples

```
data('Hepta')
out=HCLclustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

HDDClustering	<i>HDD clustering is a model-based clustering method of [Bouveyron et al., 2007].</i>
---------------	---

Description

HDD clustering is based on the Gaussian Mixture Model and on the idea that the data lives in subspaces with a lower dimension than the dimension of the original space. It uses the EM algorithm to estimate the parameters of the model [Berge et al., 2012].

Usage

```
HDDClustering(Data, ClusterNo, PlotIt=F,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	Optional, Numeric indicating either the number of cluster or a vector of 1:k to indicate the maximal expected number of clusters.
PlotIt	(optional) Boolean. Default = FALSE = No plotting performed.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used, see hddc for details.

Details

HDD clustering maximises the BIC criterion for a range of possible number of cluster up to ClusterNo. Per default the most general model is used, alternatively the parameter `model="ALL"` can be used to evaluate all possible models with BIC [Berge et al., 2012]. If specific properties of Data are known priorly please see [hddc](#) for specific model selection.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Quirin Stier

References

[Berge et al., 2012] L. Berge, C. Bouveyron and S. Girard, HDclassif: an R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data, Journal of Statistical Software, vol. 42 (6), pp. 1-29, 2012.

[Bouveyron et al., 2007] Bouveyron, C. Girard, S. and Schmid, C: High-Dimensional Data Clustering, Computational Statistics and Data Analysis, vol. 52 (1), pp. 502-519, 2007.

Examples

```
# Hepta
data("Hepta")
Data = Hepta$Data
#Non-default parameter model
#can be set to evaluate all possible models
V = HDDClustering(Data=Data,ClusterNo=7,model="ALL")
Cls = V$Cls

ClusterAccuracy(Hepta$Cls, Cls)
```

```
## Not run:
library(HDclassif)
data(Crabs)
Data = Crabs[,-1]
V = HDDClustering(Data=Data,ClusterNo=4,com_dim=1)

## End(Not run)
```

Hepta

Hepta introduced in [Ultsch, 2003]

Description

Clearly defined clusters, different variances. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Hepta")
```

Details

Size 212, Dimensions 3, stored in Hepta\$Data

Classes 7, stored in Hepta\$Cls

References

[Ultsch, 2003] Ultsch, A.: Maps for the visualization of high-dimensional data spaces, Proc. Workshop on Self organizing Maps (WSOM), pp. 225-230, Kyushu, Japan, 2003.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:[10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(Hepta)
str(Hepta)
```

HierarchicalClusterData

Internal function of Hierarchical Clusterering of Data

Description

Please use [HierarchicalClustering](#). Hierarchical cluster analysis on a set of dissimilarities and methods for analyzing it. Uses stats package function 'hclust'.

Usage

```
HierarchicalClusterData(Data,ClusterNo=0,
  Type="ward.D2",DistanceMethod="euclidean",
  ColorTreshold=0,Fast=FALSE,PlotIt=FALSE,
  Cls=NULL,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be build by the algorithm.
Type	Methode der Clusterung: "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median" or "centroid", "MinEnergy", "Minimax", and "Sparse" or "Gini" or its synonym "Genie"
DistanceMethod	see parDist , for example 'euclidean', 'mahalanobis', 'manhattan' (cityblock), 'fJaccard', 'binary', 'canberra', 'maximum'. Any unambiguous substring can be given.
ColorTreshold	Draws outline w.r.t. dendrogram y-axis (height), height of line as scalar should be given
Fast	If TRUE and fastcluster installed, then a faster implementation of the methods above can be used
PlotIt	If TRUE and ClusterNo>0 plots dendrogramm using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogramm using base R plot FALSE: No plot
Cls	If PlotIt==TRUE and ClusterNo==0: [1:n] classification vector for coloring of dendrogram in plot
...	IF PlotIt==TRUE and ClusterNo==0 In case of plotting further argument for plot, see as.dendrogram

Value

List of

Cls	If, ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise for ClusterNo=0: NULL
Dendrogram	Dendrogram of hierarchical clustering algorithm
Object	Ultrametric tree of hierarchical clustering algorithm

Author(s)

Michael Thrun

See Also[HierarchicalClusterData](#)[HierarchicalClusterDists](#)[HierarchicalClustering](#)**Examples**

```
data('Hepta')
#out=HierarchicalClusterData(Hepta$Data,ClusterNo=7)
```

HierarchicalClusterDists

Internal Hierarchical Clustering from Precomputed Dissimilarities

Description

Performs agglomerative hierarchical clustering from a precomputed dissimilarity object.

The function accepts either a dist object or a validated square dissimilarity matrix. If ClusterNo > 0, the hierarchy is cut into the requested number of clusters. If ClusterNo = 0, the complete dendrogram is plotted and returned.

This is primarily an internal helper. For the usual user interface, see [HierarchicalClustering](#).

Usage

```
HierarchicalClusterDists(
  pDist,
  ClusterNo = 0,
  Type = "ward.D2",
  ColorTreshold = 0,
  Fast = FALSE,
  PlotIt=FALSE,
  ...
)
```

Arguments

pDist	Precomputed pairwise dissimilarities supplied either as an object of class "dist" or as a finite, non-negative, symmetric numeric $n \times n$ matrix with zero diagonal. A raw feature matrix is not converted to distances by this function. Distances must be calculated beforehand, for example with <code>stats::dist()</code> . For Type = "ward.pseudo", pDist must contain the original unsquared dissimilarities.
ClusterNo	Integer between zero and the number of observations. If ClusterNo = 0, the complete hierarchy is plotted and Cls = NULL is returned. If ClusterNo > 0, the hierarchy is cut into exactly ClusterNo clusters using <code>stats::cutree()</code> .
Type	Character scalar specifying the agglomeration method. Supported values are "ward.D", "ward.D2", "ward.pseudo", "single", "complete", "average", "mcquitty", "median", and "centroid", "MinEnergy", "Minimax", and "Sparse" or "Gini" or its synonym "Genie" See Details for the distinction between the three Ward variants.
ColorTreshold	Finite numeric scalar. If nonzero and ClusterNo = 0, the value is interpreted on the dendrogram height scale. For a monotone ultrametric portion of given dissimilarity, <code>stats::rect.hclust()</code> draws rectangles at this height and classification vector Cls is return based on the cut using the threshold. If these fusion heights are non-monotone, only a horizontal reference line is drawn and Cls=NULL Height scales depend on Type; in particular, a threshold used for "ward.D2" cannot be transferred unchanged to "ward.pseudo".
Fast	Logical scalar. If TRUE and package fastcluster is installed, <code>fastcluster::hclust()</code> is used. Otherwise <code>stats::hclust()</code> is used. If Fast = TRUE but fastcluster is unavailable, a warning is issued.
PlotIt	If TRUE and ClusterNo>0 plots dendrogramm using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogramm using base R plot FALSE: No plot
...	IF PlotIt==TRUE and ClusterNo==0 In case of plotting further argument for plot, see as.dendrogram

Details**Input validation**

If pDist is supplied as a matrix or data frame, it is converted to a matrix and checked to be numeric, square, at least 2×2 , finite, symmetric, non-negative within floating-point tolerance, and to have a zero diagonal. Negligible asymmetry and small negative round-off values are corrected before conversion to a dist object.

If a dist object is supplied directly, its size, expected vector length, finiteness, and non-negativity are validated.

Ward variants

The Ward methods have distinct interpretations:

- "ward.D" applies R's legacy Ward Lance-Williams update directly to the supplied dissimilarities. It should not be interpreted as the generalized Ward pseudo-inertia criterion for arbitrary Minkowski dissimilarities.
- "ward.D2" uses R's Ward.D2 implementation. The supplied dissimilarities are squared internally before the Ward update. For arbitrary dissimilarities, this can be interpreted as a Ward-like hierarchy based on squared dissimilarities. Reported heights are on a square-root, distance-like scale.
- "ward.pseudo" implements the dissimilarity-based generalized Ward criterion described by Chavent et al. (2018).

For "ward.pseudo", let w_i denote the observation weights and $\mu_C = \sum_{i \in C} w_i$ the weight of cluster C . The dissimilarity-based pseudo-inertia is

$$I_D(C) = \frac{\sum_{i,j \in C} w_i w_j d_{ij}^2}{2\mu_C}.$$

At every agglomeration step, clusters A and B minimizing

$$\Delta_D(A, B) = I_D(A \cup B) - I_D(A) - I_D(B)$$

are merged.

This implementation uses equal normalized weights

$$w_i = \frac{1}{n}.$$

Therefore the singleton aggregation costs are

$$\Delta_{ij} = \frac{d_{ij}^2}{2n}.$$

These transformed aggregation costs, not the original dissimilarities, are passed internally to the "ward.D" Lance-Williams update together with `members = rep(1/n, n)`.

The generalized pseudo-inertia criterion is defined for potentially non-Euclidean dissimilarities.

For equal observation weights, "ward.pseudo" and "ward.D2" applied to the same original dissimilarities produce the same sequence of cluster mergers, apart from possible ordering differences caused by exact numerical ties. However, their height scales differ according to

$$h_{\text{ward.pseudo}} = \frac{h_{\text{ward.D2}}^2}{2n}.$$

allowing for a better visual identification of an appropriate cut in the dendrogram.

The "ward.pseudo" option is not the Ward_p algorithm of de Amorim (2015), which additionally uses cluster-dependent feature weights and L_p cluster centers.

Plotting and non-monotone heights

When `ClusterNo = 0`, the dendrogram is plotted. For "ward.pseudo", the y-axis represents the increase in pseudo within-cluster inertia; for other methods it is the ordinary fusion height.

If `ColorThreshold != 0` and fusion heights are monotone, `rect.hclust()` is used. If they are non-monotone, a horizontal red line is drawn instead and a warning recommends using `ClusterNo` rather than a height cut to obtain a partition.

Value

A list with components:

Cls	If, ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise for ClusterNo=0: NULL
Dendrogram	Object of class "dendrogram" representing the complete hierarchy.
Object	Object of class "hclust". For Type = "ward.pseudo", Object\$height contains the successive increases in normalized pseudo within-cluster inertia.

Note

A raw $n \times d$ feature matrix must not be passed as pDist. Compute a dissimilarity object beforehand.

For example, a Minkowski $p = 6$ hierarchy using the generalized pseudo-inertia criterion can be obtained from `stats::dist(X, method = "minkowski", p = 6)` followed by `Type = "ward.pseudo"`.

Author(s)

Michael Thrun

References

- Chavent, M., Kuentz-Simonet, V., Labenne, A., and Saracco, J. (2018). ClustGeo: an R package for hierarchical clustering with spatial constraints. *Computational Statistics*, **33**, 1799–1822. doi: 10.1007/s00180-018-0791-1.
- Murtagh, F. and Legendre, P. (2014). Ward's hierarchical agglomerative clustering method: Which algorithms implement Ward's criterion? *Journal of Classification*, **31**, 274–295. doi: 10.1007/s00357-014-9161-z.
- de Amorim, R. C. (2015). Feature relevance in Ward's hierarchical clustering using the Lp norm. *Journal of Classification*, **32**, 46–62. doi: 10.1007/s00357-015-9167-1.

See Also

[HierarchicalClusterData](#), [HierarchicalClustering](#), [ClusterDendrogram](#), [hclust](#), [cutree](#)

Examples

```
## Not run:
data("Hepta")

# Standard Ward.D2 clustering
D <- stats::dist(Hepta$Data)

res <- HierarchicalClusterDists(
  D,
  ClusterNo = 7,
```

```

    Type = "ward.D2"
  )

  res$Cls

  # Minkowski p = 6 with generalized Ward pseudo-inertia
  Xscaled <- scale(Hepta$Data)

  D6 <- stats::dist(
    Xscaled,
    method = "minkowski",
    p = 6
  )

  res_pseudo <- HierarchicalClusterDists(
    D6,
    ClusterNo = 6,
    Type = "ward.pseudo"
  )

  # Plot the complete hierarchy
  HierarchicalClusterDists(
    D,
    ClusterNo = 0,
    Type = "complete"
  )

  ## End(Not run)

```

HierarchicalClustering

Unified Wrapper for Hierarchical Clustering

Description

Provides a unified interface for hierarchical clustering from either raw numeric data or precomputed distances.

The wrapper dispatches to the appropriate lower-level implementation according to the selected clustering Type and the form of DataOrDistances. Ordinary agglomerative methods, generalized Ward pseudo-inertia clustering, and several specialized hierarchical methods are supported.

Usage

```

HierarchicalClustering(
  DataOrDistances,
  ClusterNo = 0,
  Type = "SingleL",
  Fast = TRUE,
  PlotIt=FALSE,

```

```

    Data,
    ...
)

```

Arguments

DataOrDistances

Either nonsymmetric [1:n,1:d] numerical matrix of a dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.

or

symmetric [1:n,1:n] distance matrix, e.g. as `matrix(dist(Data,method))`

ClusterNo

Number of clusters.

For the ordinary hierarchical methods and "WardPseudo", ClusterNo = 0 delegates complete-hierarchy/dendrogram handling to the lower-level function. A positive value cuts the hierarchy into the requested number of groups.

For "HDBSCAN", values greater than one explicitly cut the returned hierarchy, while smaller values use the automatic clustering produced by the HDBSCAN implementation.

Type

Character scalar selecting the hierarchical clustering method.

The following public choices are supported:

- "Ward": Ward hierarchical clustering using the Ward.D2 criterion. This is the historical/default Ward interpretation in the wrapper.
- "WardPseudo": generalized Ward-like clustering based on dissimilarity pseudo-inertia. It is intended for precomputed, potentially non-Euclidean dissimilarities, for example Minkowski $p = 6$ distances. The input must be the original unsquared dissimilarities.
- "SingleL": single-linkage clustering.
- "CompleteL": complete-linkage clustering.
- "AverageL": average-linkage clustering (UPGMA).
- "WPGMA": weighted pair-group method using arithmetic averages.
- "MedianL": median linkage (WPGMC).
- "CentroidL": centroid linkage (UPGMC).
- "Minimax": minimax-linkage clustering through `MinimaxLinkageClustering()`.
- "MinEnergy": minimum-energy hierarchical clustering through `MinimalEnergyClustering()`.
- "Gini": Genie/Gini-based clustering through `GenieClustering()`.
- "Sparse": sparse hierarchical clustering through `SparseClustering(..., Strategy = "Hierarchical")`.
- "HDBSCAN": hierarchical density-based clustering through `HierarchicalDBSCAN()`.

Default is "SingleL".

Fast

Logical scalar. Passed to the ordinary lower-level hierarchical clustering implementations. If supported, a faster implementation such as **fastcluster** may be used. Specialized methods manage their own implementations. Default is TRUE.

PlotIt	If TRUE and ClusterNo>0 plots dendrogram using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogram using base R plot FALSE: No plot
Data	Legacy alias for DataOrDistances. It is used only when DataOrDistances is missing.
...	Further arguments passed to the selected lower-level clustering function.

Details

Input dispatch

The wrapper distinguishes raw data from precomputed distances or dissimilarities.

A dist object is always treated as precomputed distances or dissimilarities. A square matrix or data frame is treated as a distance matrix only when it is numeric, finite, symmetric within numerical tolerance, has a zero diagonal, and is non-negative within tolerance. This reduces the risk of accidentally treating a square raw-data matrix as a distance matrix.

Precomputed distances or dissimilarities are sent to `HierarchicalClusterDists()`. Raw data are sent to `HierarchicalClusterData()`.

WardPseudo

"WardPseudo" exposes the generalized Ward-like dissimilarity pseudo-inertia criterion implemented by `HierarchicalClusterDists()`.

It is especially useful when the desired dissimilarity is not Euclidean. For example, Minkowski $p = 6$ dissimilarities can be calculated explicitly before clustering:

```
D6 <- stats::dist(scale(X), method = "minkowski", p = 6)
```

```
res <- HierarchicalClustering(
  D6,
  ClusterNo = 6,
  Type = "WardPseudo"
)
```

The supplied dissimilarities must be unsquared. "WardPseudo" is distinct from applying the ordinary legacy Ward method directly to raw non-Euclidean distances and is also distinct from de Amorim's Ward_p algorithm.

Specialized methods

The choices "MinEnergy", "Gini", "Minimax", "Sparse", and "HDBSCAN" are dispatched directly to their specialized implementations.

For "HDBSCAN", if `ClusterNo > 1`, the returned hierarchy is cut with `stats::cutree()`. Otherwise, the automatically selected classes from `HierarchicalDBSCAN()` are returned.

Backward compatibility

The argument `Data` remains available as a legacy alias for `DataOrDistances`. If neither is supplied, the function stops.

The public choice "Ward" retains its established meaning corresponding to Ward.D2 behavior. Use "WardPseudo" explicitly when the generalized dissimilarity pseudo-inertia criterion is desired.

Value

A method-specific list.

For ordinary hierarchical methods and "WardPseudo", the result is normally a list with components:

Cls	If, ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise for ClusterNo=0: NULL
Dendrogram	Object of class "dendrogram".
Object	Underlying hierarchical clustering object, typically of class "hclust".

For "HDBSCAN", the result additionally contains `OriginalObject`, which stores the original object returned by `HierarchicalDBSCAN()`.

Other specialized methods return the corresponding method-specific list from their lower-level implementation.

Note

The mathematical details of the generalized Ward pseudo-inertia criterion, including its height scale and relationship to Ward.D2, are documented in [HierarchicalClusterDists](#).

Author(s)

Michael Thrun

See Also

[HierarchicalClusterDists](#), [HierarchicalClusterData](#), [HierarchicalDBSCAN](#)

Examples

```
## Not run:
data("Hepta")

# Raw data with ordinary single linkage
res_single <- HierarchicalClustering(
  DataOrDistances = Hepta$Data,
  ClusterNo = 7,
  Type = "SingleL"
)

# Ward clustering from raw data
res_ward <- HierarchicalClustering(
  Hepta$Data,
  ClusterNo = 7,
  Type = "Ward"
)

# WardPseudo with explicitly computed Minkowski p = 6 dissimilarities
```

```

D6 <- stats::dist(
  scale(Hepta$Data),
  method = "minkowski",
  p = 6
)

res_pseudo <- HierarchicalClustering(
  D6,
  ClusterNo = 6,
  Type = "WardPseudo"
)

# HDBSCAN with automatic cluster-number selection
res_hdbscan <- HierarchicalClustering(
  Hepta$Data,
  Type = "HDBSCAN"
)

## End(Not run)

```

HierarchicalDBSCAN	<i>Hierarchical DBSCAN</i>
--------------------	----------------------------

Description

Hierarchical DBSCAN clustering [Campello et al., 2015].

Usage

```

HierarchicalDBSCAN(DataOrDistances,minPts=4,

PlotTree=FALSE,PlotIt=FALSE,...)

```

Arguments

DataOrDistances	Either a [1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. or a [1:n,1:n] symmetric distance matrix.
minPts	Classic smoothing factor in density estimates [Campello et al., 2015, p.9]
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
PlotTree	Default: FALSE, If TRUE plots the dendrogram. If minPts is missing, PlotTree is set to TRUE.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

"Computes the hierarchical cluster tree representing density estimates along with the stability-based flat cluster extraction proposed by Campello et al. (2013). HDBSCAN essentially computes the hierarchy of all DBSCAN* clusterings, and then uses a stability-based extraction method to find optimal cuts in the hierarchy, thus producing a flat solution." [Hahsler et al., 2019]

It is claimed by the inventors that the minPts parameter is noncritical [Campello et al., 2015, p.35]. minPts is reported to be set to 4 on all experiments [Campello et al., 2015, p.35].

Value

List of

Cls	[1:n] numerical vector defining the clustering; this classification is the main output of the algorithm. Points which cannot be assigned to a cluster will be reported as members of the noise cluster with 0.
Dendrogram	Dendrogram of hierarchical clustering algorithm
Tree	Ultrametric tree of hierarchical clustering algorithm
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Campello et al., 2015] Campello, R. J., Moulavi, D., Zimek, A., & Sander, J.: Hierarchical density estimates for data clustering, visualization, and outlier detection, ACM Transactions on Knowledge Discovery from Data (TKDD), Vol. 10(1), pp. 1-51. 2015.

[Hahsler et al., 2019] Hahsler M, Piekenbrock M, Doran D: dbscan: Fast Density-Based Clustering with R. Journal of Statistical Software, 91(1), pp. 1-30. doi: 10.18637/jss.v091.i01, 2019

Examples

```
data('Hepta')

out=HierarchicalDBSCAN(Hepta$Data,PlotIt=FALSE)

data('Leukemia')
set.seed(1234)
CA=HierarchicalDBSCAN(Leukemia$DistanceMatrix)
#ClusterCount(CA$Cls)
#ClusterDendrogram(CA$Dendrogram,5,main='H-DBscan')
```

IsDissimilarity	<i>Check whether an object represents a dissimilarity</i>
-----------------	---

Description

Tests whether an object can reasonably be interpreted as a dissimilarity or distance representation.

Objects inheriting from class "dist" are accepted directly. Numeric square matrices and data frames are checked for finiteness, approximate symmetry, an approximately zero diagonal, and non-negative entries.

Usage

```
IsDissimilarity(x)
```

Arguments

x	Object to test. Typically a <code>stats::dist</code> object, numeric matrix, data frame, or another R object.
---	---

Details

An object inheriting from class "dist" is returned as TRUE without further matrix validation.

For matrix or data-frame input, x is first converted with `as.matrix()`. It must then satisfy all of the following requirements:

- it is numeric;
- it is two-dimensional;
- it is square;
- it contains at least two observations;
- it contains no missing values;
- all entries are finite.

More reliable than calling `isSymmetric()` directly on every possible input. Within floating-point tolerance the check requires a zero diagonal, symmetric and nonnegative entries. The check also reduces accidental classification of a square, symmetric data matrix as a distance matrix.

The floating-point tolerance is calculated as

$$100\epsilon \max(1, \max |x|),$$

For a distance, the triangle inequality must additionally be satisfied, which is not checked here.

Value

Logical scalar.

TRUE if x is recognized as a distance representation; otherwise FALSE.

Note

Small negative values caused by floating-point round-off are tolerated, but the function does not modify the supplied object.

See Also

[dist](#), [isSymmetric](#)

Examples

```
X <- matrix(
  c(
    0, 1, 2,
    1, 0, 3,
    2, 3, 0
  ),
  nrow = 3,
  byrow = TRUE
)

IsDissimilarity(X)
# TRUE

IsDissimilarity(stats::as.dist(X))
# TRUE

IsDissimilarity(matrix(1:6, nrow = 2))
# FALSE

bad <- X
bad[1, 2] <- 10
IsDissimilarity(bad)
# FALSE
```

kmeansClustering

K-Means Clustering

Description

Perform k-means clustering on a data matrix.

Usage

```
kmeansClustering(DataOrDistances, ClusterNo,
  Type = 'LBG', RandomNo=5000, CategoricalData,
  PlotIt=FALSE, Verbose = FALSE,... )
```

Arguments

DataOrDistances	Either nonsymmetric [1:n,1:d] datamatrix of n cases and d numerical features or symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Type	Choice of Kmeans algorithm, currently either "Hartigan" [Hartigan/Wong, 1979], "LBG" [Linde et al., 1980], "Sparse" sparse k-means proposed in [Witten/Tibshirani, 2010], "Steinley" best method of [Steinley/Brusco, 2007] proposed in Steinley 2003, "Lloyd" [Lloyd, 1982], "Forgy" [Forgy, 1965], MacQueen [MacQueen, 1967], kcentroids [Leisch, 2006], "kprototypes" [Szepannek, 2018], "Pelleg-moore" [Pelleg & Moores, 2000], "Elkan" [Elkan, 2003], "kmeans++" [Arthur & Vassilvitskii], Hamerly [Hamerly, 2010], Dualtree or Dualtree-covertree [Curtin, 2017]"
RandomNo	Only for "Steinley" or in case of distance matrix, number of random initializations with searching for minimal SSE, see [Steinley/Brusco, 2007]
CategoricalData	Only for "kprototypes", [1:n,1:m] matrix of categorical features]
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
Verbose	Print details, if true
...	Further arguments like iter.max, nstart, for kcentroids please see kcca function of the flexclust package, or KMeansSparseCluster

Details

Uses either **stats** package function 'kmeans', **cclust** package implementation, **flexclust** package implementation or own code. In case of a distance matrix, RandomNo should be significantly lower than 5000, otherwise a long computation time is to be expected.

Value

List V of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object of the clustering algorithm used if existent, otherwise SumDistsToCentroids: Vector of within-cluster sum of squares, one component per cluster
Centroids	the final cluster centers.

Note

The version using a distance matrix is still in the test phase and not yet verified.

Author(s)

Michael Thrun

References

- [Hartigan/Wong, 1979] Hartigan, J. A., & Wong, M. A.: Algorithm AS 136: A k-means clustering algorithm, *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, Vol. 28(1), pp. 100-108. 1979.
- [Linde et al., 1980] Linde, Y., Buzo, A., & Gray, R.: An algorithm for vector quantizer design, *IEEE Transactions on communications*, Vol. 28(1), pp. 84-95. 1980.
- [Steinley/Brusco, 2007] Steinley, D., & Brusco, M. J.: Initializing k-means batch clustering: A critical evaluation of several techniques, *Journal of Classification*, Vol. 24(1), pp. 99-121. 2007.
- [Forgy, 1965] Forgy, E. W.: Cluster analysis of multivariate data: efficiency versus interpretability of classifications, *Biometrics*, Vol. 21, pp. 768-769. 1965.
- [MacQueen, 1967] MacQueen, J.: Some methods for classification and analysis of multivariate observations, *Proc. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1, pp. 281-297, Oakland, CA, USA., 1967.
- [Pelleg & Moores, 2000] Pelleg, Dan, and Andrew W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters, *ICML*. Vol. 1. 2000.
- [Elkan, 2003] Elkan, Charles: Using the triangle inequality to accelerate k-means, In Tom Fawcett and Nina Mishra, editors, *ICML*, pages Vol.3, 147-153. AAAI Press, 2003.
- [Lloyd, 1982] Lloyd, S.: Least squares quantization in PCM, *IEEE transactions on information theory*, Vol. 28(2), pp. 129-137. 1982.
- [Leisch, 2006] Leisch, F.: A toolbox for k-centroids cluster analysis, *Computational Statistics & Data Analysis*, Vol. 51(2), pp. 526-544. 2006.
- [Arthur & Vassilvitskii] Arthur, David, and Vassilvitskii, Sergei: K-means++ the advantages of careful seeding, *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 2007
- [Witten/Tibshirani, 2010] Witten, D. and Tibshirani, R.: A Framework for Feature Selection in Clustering. *Journal of the American Statistical Association*, Vol. 105(490), pp. 713-726, 2010.
- [Hamerly, 2010] Hamerly, Greg: Making k-means even faster, *Proceedings of the 2010 SIAM international conference on data mining*, Society for Industrial and Applied Mathematics, pp. 130-140, 2010.
- [Szepannek, 2018] Szepannek, G.: clustMixType: User-Friendly Clustering of Mixed-Type Data in R, *The R Journal*, Vol. 10/2, pp. 200-208, doi:10.32614/RJ2018048, 2018.
- [Curtin, 2017] Curtin, Ryan R: A dual-tree algorithm for fast k-means clustering with large k, *Proceedings of the 2017 SIAM International Conference on Data Mining*, Society for Industrial and Applied Mathematics, 2017.

Examples

```
data('Hepta')
out=kmeansClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

```
data('Leukemia')
# As expected does not perform well
# For non-spherical cluster structures:
out=kmeansClustering(Leukemia$DistanceMatrix,ClusterNo=6,RandomNo =10,PlotIt=TRUE)
```

```
data('Hepta')
out=kmeansClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE,Type="Steinley")
```

```
data('Hepta')
out=kmeansClustering(Hepta$Data,ClusterNo = 7,
Type = "kprototypes",CategoricalData = as.matrix(Hepta$Cls))
```

kmeansDist	<i>k-means Clustering using a distance matrix</i>
------------	---

Description

Perform k-means clustering on a distance matrix

Usage

```
kmeansDist(Distance, ClusterNo=2,Centers=NULL,
RandomNo=1,maxIt = 2000,
PlotIt=FALSE,verbose = F)
```

Arguments

Distance	Distance matrix. For n data points of the dimension n x n
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Centers	Default(NULL) a set of initial (distinct) cluster centres.
RandomNo	If>1: Number of random initializations with searching for minimal SSE is defined by this scalar
maxIt	Optional: Maximum number of iterations before the algorithm terminates.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
verbose	Optional: Algorithm always outputs current iteration.

Value

`Cls[1:n]` [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.

`centerids[1:k]` Indices of the centroids from which the cluster CIs was created

Note

Currently an experimental version

Author(s)

Felix Pape, Michael Thrun

Examples

```
data('Hepta')
#out=kmeansDist(as.matrix(dist(Hepta$Data)),ClusterNo=7,PlotIt=FALSE,RandomNo = 10)

## Not run:
data('Leukemia')
#as expected does not perform well
#for non-spherical cluster structures:
#out=kmeansDist(Leukemia$DistanceMatrix,ClusterNo=6,PlotIt=TRUE,RandomNo=10)

## End(Not run)
```

LargeApplicationClustering

Large Application Clustering

Description

Clustering Large Applications (clara) of [Rousseeuw/Kaufman, 1990, pp. 126-163]

Usage

```
LargeApplicationClustering(Data, ClusterNo,
PlotIt=FALSE,Standardization=TRUE,Samples=50,Random=TRUE,...)
```

Arguments

`Data` [1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.

`ClusterNo` A number k which defines k different clusters to be built by the algorithm.

PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
Standardization	Data is standardized before calculating the dissimilarities. Measurements are standardized for each variable (column), by subtracting the variable's mean value and dividing by the variable's mean absolute deviation.
Samples	Integer, say N, the number of samples to be drawn from the dataset. Default value set as recommended by documentation of clara
Random	Logical indicating if R's random number generator should be used instead of the primitive clara()-builtin one.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

It is recommended to use `set.seed` if clustering output should be always the same instead of setting `Random=FALSE` in order to use the primitive `clara()`-builtin random number generator.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Rousseeuw/Kaufman, 1990] Rousseeuw, P. J., & Kaufman, L.: Finding groups in data, Belgium, John Wiley & Sons Inc., ISBN: 0471735787, doi 10.1002/9780470316801, Online ISBN: 9780470316801, 1990.

Examples

```
data('Hepta')
out=LargeApplicationClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

Leukemia

*Leukemia distance matrix and classification used in [Thrun, 2018]***Description**

Data is anonymized. Original dataset was published in [Haferlach et al., 2010]. Original dataset had around 12.000 dimensions. Detailed description of preprocessed dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Leukemia")
```

Details

554x554 distance matrix. Cls defines the following clusters:

1= APL Outlier

2=APL

3=Healthy

4=AML

5=CLL

6=CLL Outlier

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, [doi:10.1007/9783658205409](https://doi.org/10.1007/9783658205409), 2018.

[Haferlach et al., 2010] Haferlach, T., Kohlmann, A., Wieczorek, L., Basso, G., Te Kronnie, G., Bene, M.-C., . . . Mills, K. I.: Clinical utility of microarray-based gene expression profiling in the diagnosis and subclassification of leukemia: report from the International Microarray Innovations in Leukemia Study Group, Journal of Clinical Oncology, Vol. 28(15), pp. 2529-2537. 2010.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, [doi:10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(Leukemia)
str(Leukemia)
Cls=Leukemia$Cls
Distance=Leukemia$DistanceMatrix
isSymmetric(Distance)
```

Lsun3D

*Lsun3D inspired by FCPS introduced in [Thrun, 2018]***Description**

Clearly defined clusters, different variances. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Lsun3D")
```

Details

Size 404, Dimensions 3

Dataset defines discontinuities, where the clusters have different variances. Three main clusters, and four outliers (in cluster 4). For a more detailed description see [Thrun, 2018].

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, Heidelberg, ISBN: 978-3-658-20539-3, doi:[10.1007/9783658205409](https://doi.org/10.1007/9783658205409), 2018.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:[10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(Lsun3D)
str(Lsun3D)
Cls=Lsun3D$Cls
Data=Lsun3D$Data
```

MarkovClustering

*Markov Clustering***Description**

Graph clustering algorithm introduced by [van Dongen, 2000].

Usage

```
MarkovClustering(DataOrDistances=NULL,Adjacency=NULL,
Radius=TRUE,DistanceMethod="euclidean",addLoops = TRUE,PlotIt=FALSE,...)
```

Arguments

DataOrDistances	NULL or: Either [1:n,1:n] symmetric distance matrix or [1:n,1:d] not symmetric data matrix of n cases and d variables
Adjacency	Used if Data is NULL, matrix [1:n,1:n] defining which points are adjacent to each other by the number 1; not adjacent: 0
Radius	Scalar, Radius for unit disk graph (r-ball graph) if adjacency matrix is missing. Automatic estimation can be done either with =TRUE [Ultsch, 2005] or FALSE [Thrun et al., 2016] if Data instead of Distances are given.
DistanceMethod	Optional distance method of data, default is euclid, see parDist for details
addLoops	Logical; if TRUE, self-loops with weight 1 are added to each vertex of x (see <code>mcl</code> of CRAN package MCL).
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

DataOrDistances is used to compute the Adjacency matrix if this input is missing. Then a unit-disk (R-ball) graph is calculated.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [van Dongen, 2000] van Dongen, S.M. Graph Clustering by Flow Simulation. Ph.D. thesis, University of Utrecht. Utrecht University Repository: <http://dspace.library.uu.nl/handle/1874/848>, 2000
- [Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A. : Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Vol. 24, Plzen, 2016.
- [Ultsch, 2005] Ultsch, A.: Pareto density estimation: A density estimation for knowledge discovery, In Baier, D. & Werrnecke, K. D. (Eds.), Innovations in classification, data science, and information systems, (Vol. 27, pp. 91-100), Berlin, Germany, Springer, 2005.

Examples

```
data('Hepta')
out=MarkovClustering(Data=Hepta$Data,PlotIt=FALSE)
```

MeanShiftClustering	<i>Mean Shift Clustering</i>
---------------------	------------------------------

Description

Mean Shift Clustering of [Cheng, 1995]

Usage

```
MeanShiftClustering(Data,
PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

the radius used for search can be specified with the "radius" parameter. The maximum number of iterations before algorithm termination is controlled with the "max_iterations" parameter.

If the distance between two centroids is less than the given radius, one will be removed. A radius of 0 or less means an estimate will be calculated and used for the radius. Default value "0" (numeric).

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Cheng, 1995] Cheng, Yizong: Mean Shift, Mode Seeking, and Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17 (8), pp. 790-799, doi:10.1109/34.400568, 1995.

Examples

```
data('Hepta')
out=MeanShiftClustering(Hepta$Data,PlotIt=FALSE,radius=1)
```

MinimalEnergyClustering

Minimal Energy Clustering

Description

Hierarchical Clustering using the minimal energy approach of [Szekely/Rizzo, 2005].

Usage

```
MinimalEnergyClustering(DataOrDistances, ClusterNo = 0,
  ColorTreshold = 0,
  DistanceMethod="euclidean", Data,PlotIt=FALSE,...)
```

Arguments

DataOrDistances	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be build by the algorithm.
DistanceMethod	See parDist , for example 'euclidean', 'mahalanobis', 'manhattan' (cityblock), 'fJaccard', 'binary', 'canberra', 'maximum'. Any unambiguous substring can be given.
ColorTreshold	Draws outline w.r.t. dendrogram y-axis (height), height of line as scalar should be given
Data	[1:n,1:d] data matrix in the case that DataOrDistances is missing and partial matching does not work.
PlotIt	If TRUE and ClusterNo>0 plots dendrogramm using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogramm using base R plot FALSE: No plot
...	In case of plotting further argument for plot, see as.dendrogram

Value

List of

Cls	If ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise ClusterNo=0: NULL
Dendrogram	Dendrogram of hierarchical clustering algorithm
Object	Ultrametric tree of hierarchical clustering algorithm

Author(s)

Michael Thrun

References

[Szekely/Rizzo, 2005] Szekely, G. J. and Rizzo, M. L.: Hierarchical Clustering via Joint Between-Within Distances: Extending Ward's Minimum Variance Method, Journal of Classification, 22(2) 151-183.<http://dx.doi.org/10.1007/s00357-005-0012-9>, 2005.

See Also[HierarchicalClustering](#)**Examples**

```
data('Hepta')
out=MinimalEnergyClustering(Hepta$Data,ClusterNo=7)
```

MinimaxLinkageClustering

Minimax Linkage Hierarchical Clustering

Description

In the minimax linkage hierarchical clustering every cluster has an associated prototype element that represents that cluster [Bien/Tibshirani, 2011].

Usage

```
MinimaxLinkageClustering(DataOrDistances, ClusterNo = 0,
ColorTreshold = 0,
DistanceMethod="euclidean", PlotIt=FALSE,...)
```

Arguments

DataOrDistances	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be build by the algorithm.
DistanceMethod	See parDist , for example 'euclidean', 'mahalanobis', 'manhattan' (cityblock), 'fJaccard', 'binary', 'canberra', 'maximum'. Any unambiguous substring can be given.
ColorTreshold	Draws cutline w.r.t. dendrogram y-axis (height), height of line as scalar should be given
PlotIt	If TRUE and ClusterNo>0 plots dendrogramm using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogramm using base R plot FALSE: No plot
...	In case of plotting further argument for plot, see as.dendrogram

Value

List of	
Cls	If, ClusterNo>0: [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Otherwise for ClusterNo=0: NULL
Dendrogram	Dendrogram of hierarchical clustering algorithm
Object	Ultrametric tree of hierarchical clustering algorithm

Author(s)

Michael Thrun

References

[Bien/Tibshirani, 2011] Bien, J., and Tibshirani, R.: Hierarchical Clustering with Prototypes via Minimax Linkage, The Journal of the American Statistical Association, Vol. 106(495), pp. 1075-1084, 2011.

See Also

[HierarchicalClustering](#)

Examples

```
data('Hepta')
out=MinimaxLinkageClustering(Hepta$Data,ClusterNo=7)
```

ModelBasedClustering *Model Based Clustering*

Description

Calls Model based clustering of [Fraley/Raftery, 2006] which models a Mixture Of Gaussians (MoG).

Usage

```
ModelBasedClustering(Data, ClusterNo=2, PlotIt=FALSE, ...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

see [Thrun, 2017, p. 23] or [Fraley/Raftery, 2002] and [Fraley/Raftery, 2006].

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Note

MoGclustering used in [Thrun, 2017] was renamed to [ModelBasedClustering](#) in this package.

Author(s)

Michael Thrun

References

[Thrun, 2017] Thrun, M. C.: A System for Projection Based Clustering through Self-Organization and Swarm Intelligence, (Doctoral dissertation), Philipps-Universitaet Marburg, Marburg, 2017.

[Fraley/Raftery, 2002] Fraley, C., and Raftery, A. E.: Model-based clustering, discriminant analysis, and density estimation, Journal of the American Statistical Association, Vol. 97(458), pp. 611-631. 2002.

[Fraley/Raftery, 2006] Fraley, C., and Raftery, A. E. MCLUST version 3: an R package for normal mixture modeling and model-based clustering, DTIC Document, 2006.

See Also

[MoGclustering](#)

Examples

```
data('Hepta')
out=ModelBasedClustering(Hepta$Data,PlotIt=FALSE)
```

ModelBasedVarSelClustering

Model Based Clustering with Variable Selection

Description

Model-based clustering with variable selection and estimation of the number of clusters which is either based on [Marbac/Sedki, 2017], [Marbac et al., 2020], or on [Scrucca and Raftery, 2014].

Usage

```
ModelBasedVarSelClustering(Data,ClusterNo,Type,PlotIt=FALSE, ...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	Numeric which defines number of cluster to search for.
Type	String, either VarSelLCM [Marbac/Sedki, 2017], [Marbac et al., 2020], or clustvarsel [Scrucca and Raftery, 2014].
PlotIt	(optional) Boolean. Default = FALSE = No plotting performed.
...	Further arguments passed on to VarSelCluster or clustvarsel .

Value

List of

Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Quirin Stier, Michael Thrun

References

- [Marbac/Sedki, 2017] Marbac, M. and Sedki, M.: Variable selection for model-based clustering using the integrated complete-data likelihood. *Statistics and Computing*, 27(4), pp. 1049-1063, 2017.
- [Marbac et al., 2020] Marbac, M., Sedki, M., & Patin, T.: Variable selection for mixed data clustering: application in human population genomics, *Journal of Classification*, Vol. 37(1), pp. 124-142, 2020.

Examples

```
# Hepta
data("Hepta")
Data = Hepta$Data
V = ModelBasedVarSelClustering(Data, ClusterNo=7, Type="VarSelLCM")
Cls = V$Cls
ClusterAccuracy(Hepta$Cls, Cls, K = 7)

V = ModelBasedVarSelClustering(Data, ClusterNo=7, Type="clustvarsel")
Cls = V$Cls
ClusterAccuracy(Hepta$Cls, Cls, K = 7)

## Not run:
# Hearts
heart=VarSelLCM::heart
ztrue <- heart[, "Class"]
Data <- heart[, -13]
V <- ModelBasedVarSelClustering(Data, 2, Type="VarSelLCM")
Cls = V$Cls
ClusterAccuracy(ztrue, Cls, K = 2)

## End(Not run)
```

MoGclustering

*Mixture of Gaussians Clustering using EM***Description**

MixtureOfGaussians (MoG) clustering based on Expectation Maximization (EM) of [Chen et al., 2012] or algorithms closely resembling EM of [Benaglia/Chauveau/Hunter, 2009].

Usage

```
MoGclustering(Data,ClusterNo=2,Type,PlotIt=FALSE,Silent=TRUE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Type	string defining approach to select: initialization approach of "EM" or "kmeans" of [Chen et al., 2012], or other methods "mvnormalmixEM" [McLachlan/Peel, 2000], "npEM"[Benaglia et al., 2009] or its extension "mvnpEM" [Chauveau/Hoang, 2016].
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
Silent	(optional) Boolean: print output or not (Default = FALSE = no output)
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used, see package mixtools EMCluster or mixtools for details.

Details

Algorithms for clustering through EM or its close resembles.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Note

MoG used in [Thrun, 2017] was renamed to [ModelBasedClustering](#) in this package. Type="mvnormalmixEM" sometimes fails

Author(s)

Michael Thrun

References

[Chen et al., 2012] Chen, W., Maitra, R., & Melnykov, V.: EMCluster: EM Algorithm for Model-Based Clustering of Finite Mixture Gaussian Distribution, R Package, URL <http://cran.r-project.org/package=EMCluster>, 2012.

[Chauveau/Hoang, 2016] Chauveau, D., & Hoang, V. T. L.: Nonparametric mixture models with conditionally independent multivariate component densities, Computational Statistics & Data Analysis, Vol. 103, pp. 1-16. 2016.

[Benaglia et al., 2009] Benaglia, T., Chauveau, D., and Hunter, D. R.: An EM-like algorithm for semi-and nonparametric estimation in multivariate mixtures. Journal of Computational and Graphical Statistics, 18(2), pp. 505-526, 2009.

[McLachlan/Peel, 2000] D. McLachlan, G. J. and Peel, D.: Finite Mixture Models, John Wiley and Sons, Inc, 2000.

See Also

[ModelBasedClustering](#)

Examples

```
data('Hepta')
Data = Hepta$Data
out=MoGclustering(Data,ClusterNo=7,Type="EM",PlotIt=FALSE)
V=out$Cls

V1 = MoGclustering(Data,ClusterNo=7,Type="mvnpEM")
Cls1 = V1$Cls

V2 = MoGclustering(Data,ClusterNo=7,Type="npEM")
Cls2 = V2$Cls

## Not run:
#does not work always
V3 = MoGclustering(Data,ClusterNo=7,Type="mvnormalmixEM")
Cls3 = V3$Cls

## End(Not run)
```

MSTclustering

MST-kNN clustering algorithm [Inostroza-Ponta, 2008].

Description

Performs the MST-kNN clustering algorithm which generate a clustering solution with automatic k determination using two proximity graphs: Minimal Spanning Tree (MST) and k-Nearest Neighbor (kNN) which are recursively intersected.

Usage

```
MSTclustering(DataOrDistances, DistanceMethod = "euclidean", PlotIt=FALSE, ...)
```

Arguments

DataOrDistances	Either [1:n,1:n] symmetric distance matrix or [1:n,1:d] not symmetric data matrix of n cases and d variables
DistanceMethod	Optional distance method of data, default is euclid, see parDist for details
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Optional, further arguments for mst.knn

Details

Does not work on Hepta with euclidean distances.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Inostroza-Ponta, 2008] Inostroza-Ponta, M.: An integrated and scalable approach based on combinatorial optimization techniques for the analysis of microarray data, University of Newcastle, ISBN, 2008

See Also[mst.knn](#)**Examples**

```
data(Hepta)

MSTclustering(Hepta$Data)
```

NetworkClustering	<i>Network Clustering</i>
-------------------	---------------------------

Description

Either leiden [Traag et al., 2019] or louvain [Blondel et al., 2008] clustering

Usage

```
NetworkClustering(DataOrDistances=NULL,Adjacency=NULL,

Type="louvain",Radius=FALSE,PlotIt=FALSE,...)
```

Arguments

DataOrDistances	NULL or: [1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
Adjacency	Used if DataOrDistances is NULL, matrix [1:n,1:n] defining which points are adjacent to each other by the number 1; not adjacent: 0
Type	Either "louvain" or "leiden"
Radius	Scalar, Radius for unit disk graph (r-ball graph) if adjacency matrix is missing. Automatic estimation can be done either with =TRUE [Ultsch, 2005] or FALSE [Thrun et al., 2016]
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

DataOrDistances is used to compute the Adjacency matrix if this input is missing. Then a unit-disk (R-ball) graph is calculated. Radius=TRUE only works if data matrix is given.

Value

List of

Cls [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.

Object Object defined by clustering algorithm as the other output of this algorithm

Note

leiden requires igraph package and an installed python version. automatic installation may not work. manual call in console has to be in this case `conda install -c conda-forge leidenalg`

Author(s)

Michael Thrun

References

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E.: Fast unfolding of communities in large networks, Journal of statistical mechanics: theory and experiment, Vol. 2008(10), pp. P10008. 2008.

[Traag et al., 2019] Traag, V. A., Waltman, L., & van Eck, N. J.: From Louvain to Leiden: guaranteeing well-connected communities, Scientific reports, Vol. 9(1), pp. 1-12. 2019.

Examples

```
data('Hepta')
#out=NetworkClustering(Hepta$Data,PlotIt=FALSE)
```

NeuralGasClustering *Neural gas algorithm for clustering*

Description

Neural gas clustering published by [Martinetz et al., 1993]] and implemented by [Bodenhofer et al., 2011].

Usage

```
NeuralGasClustering(Data, ClusterNo,PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Dimitriadou, 2002] Dimitriadou, E.: cclust-convex clustering methods and clustering indexes. R package, 2002,
- [Martinetz et al., 1993] Martinetz, T. M., Berkovich, S. G., & Schulten, K. J.: 'Neural-gas' network for vector quantization and its application to time-series prediction, IEEE Transactions on Neural Networks, Vol. 4(4), pp. 558-569. 1993.

Examples

```
data('Hepta')
out=NeuralGasClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

OPTICSclustering

OPTICS Clustering

Description

OPTICS (Ordering points to identify the clustering structure) clustering algorithm [Ankerst et al.,1999].

Usage

```
OPTICSclustering(Data, MaxRadius,RadiusThreshold, minPts = 5, PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
MaxRadius	Upper limit neighborhood in the R-ball graph/unit disk graph), size of the epsilon neighborhood (eps) [Ester et al., 1996, p. 227]. If NULL, automatic estimation is done using insights of [Ultsch, 2005].
RadiusThreshold	Threshold to identify clusters (RadiusThreshold <= MaxRadius), if NULL 0.9*MaxRadius is set.
minPts	Number of minimum points in the eps region (for core points). In principle minimum number of points in the unit disk, if the unit disk is within the cluster (core) [Ester et al., 1996, p. 228]. If NULL, its 2.5 percent of points.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

...

Value

List of	
Cls	[1:n] numerical vector defining the clustering; this classification is the main output of the algorithm. Points which cannot be assigned to a cluster will be reported as members of the noise cluster with 0.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Ankerst et al.,1999] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, Joerg Sander: OPTICS: Ordering Points To Identify the Clustering Structure, ACM SIGMOD international conference on Management of data, ACM Press, pp. 49-60, 1999.
- [Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., & Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, Proc. Kdd, Vol. 96, pp. 226-231, 1996.
- [Ultsch, 2005] Ultsch, A.: Pareto density estimation: A density estimation for knowledge discovery, In Baier, D. & Werrnecke, K. D. (Eds.), Innovations in classification, data science, and information systems, (Vol. 27, pp. 91-100), Berlin, Germany, Springer, 2005.

See Also[optics](#)

Examples

```
data('Hepta')
out=OPTICSclustering(Hepta$Data,MaxRadius=NULL,RadiusThreshold=NULL,minPts=NULL,PlotIt = FALSE)
```

PAMclustering	<i>Partitioning Around Medoids (PAM)</i>
---------------	--

Description

Partitioning (clustering) of the data into k clusters around medoids, a more robust version of k-means [Rousseeuw/Kaufman, 1990, p. 68-125] .

Usage

```
PAMclustering(DataOrDistances,ClusterNo,
PlotIt=FALSE,Standardization=TRUE,Data,...)
```

Arguments

DataOrDistances	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. Alternatively, symmetric [1:n,1:n] distance matrix
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
Standardization	DataOrDistances is standardized before calculating the dissimilarities. Measurements are standardized for each variable (column), by subtracting the variable's mean value and dividing by the variable's mean absolute deviation.If DataOrDistances is already a distance matrix, then this argument will be ignored.
Data	[1:n,1:d] data matrix in the case that DataOrDistances is missing and partial matching does not work.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

[Rousseeuw/Kaufman, 1990, chapter 2] or [Reynolds et al., 1992].

Value

List of

Cls [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.

Object Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Rousseeuw/Kaufman, 1990] Rousseeuw, P. J., & Kaufman, L.: Finding groups in data, Belgium, John Wiley & Sons Inc., ISBN: 0471735787, doi:10.1002/9780470316801, Online ISBN: 9780470316801, 1990.

[Reynolds et al., 1992] Reynolds, A., Richards, G., de la Iglesia, B. and Rayward-Smith, V.: Clustering rules: A comparison of partitioning and hierarchical clustering algorithms, Journal of Mathematical Modelling and Algorithms 5, 475-504, DOI:10.1007/s10852-005-9022-1, 1992.

Examples

```
data('Hepta')
out=PAMclustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

parApplyClusterAnalysis

Repeated Application of a Clustering Method

Description

Repeatedly applies a clustering method to data or distances. Trials can be executed serially or on a worker cluster from the recommended **parallel** package. A single input or a list of inputs can be processed. It was used for the benchmarking of clustering algorithms in [Thrun, 2018; Thrun/Ultsch, 2020; Thrun, 2021].

Usage

```
parApplyClusterAnalysis(
  DataOrDistances,
  FUN,
  NumberOfTrials = 1:100,
  ClusterNo = NULL,
  WorkersOrNo,
  SocketType = "PSOCK",
  SetSeed = TRUE,
  ...
)
```

Arguments

DataOrDistances	Data, distances, dissimilarities, or a list of such inputs. A list that is not a <code>data.frame</code> is treated as a collection of inputs.
FUN	A function or character string naming a clustering function. The clustering function should return a list with a unique element named <code>Cls</code> if a clustering matrix is required.
NumberOfTrials	A positive integer count or a vector of positive integer trial identifiers. If a single value <code>n</code> is supplied, the trial identifiers are <code>seq_len(n)</code> . Default is <code>1:100</code> .
ClusterNo	Number of clusters supplied to FUN. For multiple inputs, this may be one value or one value per input. <code>NULL</code> , an empty value, or <code>NA</code> causes the argument to be omitted from the clustering-function call.
WorkersOrNo	Missing, <code>NULL</code> , a positive integer, or an object inheriting from class <code>"cluster"</code> . If missing, the function attempts to use one fewer than the number of detected logical CPU cores. <code>NULL</code> requests serial execution. A positive integer creates a new worker cluster. An existing cluster object is used without being stopped by this function.
SocketType	Non-empty character scalar passed to the <code>type</code> argument of <code>parallel::makeCluster()</code> . Default is <code>"PSOCK"</code> .
SetSeed	Logical scalar. If <code>TRUE</code> , trial <code>i</code> uses seed <code>1000 + i</code> . If <code>FALSE</code> , no deterministic trial seed is assigned. Default is <code>TRUE</code> .
...	Further arguments passed to FUN. Named arguments in ... may not redefine arguments already supplied by the wrapper.

Details

Default is the number of cores existing minus 1.

In **FCPS** default parameters for each clustering algorithm are used automatically if not specified by the user. `parApplyClusterAnalysis` expects in FUN a function of a clustering algorithm which returns a list of objects of which one is named `Cls`. If not given the whole output of FUN is returned with a warning.

`Cls` is a `[1:N]` numerical vector of of numbers `1:k` of the `k` clusters labeling the data points to the clusters.

Value

For a single input, a list with components:

<code>Cls_Matrix</code>	Matrix containing one clustering vector per trial. Column names identify seeds or trial IDs. It is <code>NULL</code> if at least one result does not have exactly one element named <code>Cls</code> .
<code>ComputationTime</code>	Named numeric vector containing elapsed computation time in seconds for each trial.
<code>Seeds</code>	Named integer vector containing the seeds used for the trials, or <code>NULL</code> when <code>SetSeed = FALSE</code> .

CAobjects List of complete clustering results. For this generic wrapper, it is included when `Cls_Matrix` cannot be constructed.

For multiple inputs, a list containing one result of this form per input is returned. Input names are retained when present.

Author(s)

Michael Thrun

References

Thrun, M. C.: Projection-Based Clustering through Self-Organization and Swarm Intelligence, Springer, Heidelberg, ISBN: 978-3658205393, 2018.

Thrun, M. C., Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, Artificial Intelligence, Vol. 290, pp. 103237, DOI: 10.1016/j.artint.2020.103237, Elsevier, 2021.

Thrun, M. C.: Distance-Based Clustering Challenges for Unbiased Benchmarking Studies, Nature Scientific Reports, Vol. 11, pp. 18988, DOI: 10.1038/s41598-021-98126-1, 2021.

See Also

[clusterApply](#)

Examples

```
simple_clustering <- function(Data, ClusterNo = 2, ...) {
  fit <- stats::kmeans(Data, centers = ClusterNo)
  list(Cls = fit$cluster, Object = fit)
}

DataList=list(
  Dataset1=matrix(rnorm(40, 0), ncol = 2),
  DataSet2=matrix(rnorm(40, 4), ncol = 2)
)

# Serial example
result <- parApplyClusterAnalysis(
  DataOrDistances = DataList,
  FUN = simple_clustering,
  NumberOfTrials = 2,
  ClusterNo = 2,
  WorkersOrNo = NULL
)

result$Cls_Matrix
result$ComputationTime
```

parApplyDataBasedCA *Repeated Application of a Data-Based Clustering Method*

Description

Repeatedly applies a data-based clustering method to one dataset or a collection of datasets. Trials may be executed serially or in parallel.

Usage

```
parApplyDataBasedCA(
  Data,
  FUN,
  NumberOfTrials = 1:100,
  ClusterNo = NULL,
  WorkersOrNo,
  SocketType = "PSOCK",
  SetSeed = TRUE,
  ...
)
```

Arguments

Data	Dataset accepted by FUN, or a list of datasets. A list that is not a <code>data.frame</code> is interpreted as a collection of inputs.
FUN	Function or character string naming a data-based clustering method. The result should normally be a list containing a unique element named <code>C1s</code> .
NumberOfTrials	Positive integer count or vector of positive integer trial identifiers. A scalar <code>n</code> generates <code>seq_len(n)</code> . Default is <code>1:100</code> .
ClusterNo	Number of clusters passed to FUN. For multiple datasets, it may be scalar or provide one value per dataset. <code>NULL</code> or <code>NA</code> omits the argument.
WorkersOrNo	Missing, <code>NULL</code> , a positive integer worker count, or an existing <code>parallel</code> cluster. If missing, one fewer than the detected logical cores is used when possible. <code>NULL</code> requests serial execution.
SocketType	Character scalar passed to <code>parallel::makeCluster()</code> . Default is <code>"PSOCK"</code> .
SetSeed	Logical scalar. If <code>TRUE</code> , trial <code>i</code> uses seed <code>1000 + i</code> . Default is <code>TRUE</code> .
...	Further arguments passed to FUN. Wrapper-supplied arguments may not be re-defined.

Details

The dataset is passed to a formal argument named `Data` when available. Otherwise, `DataOrDistances` or the first inspectable non-dots argument is used.

For multiple datasets, errors raised while processing an individual dataset are converted to warnings. The corresponding list element is returned as NULL, while processing continues for the remaining datasets.

An internally created worker cluster is stopped automatically. A cluster supplied through `WorkersOrNo` is not stopped.

Value

For one dataset, a list with:

<code>Cls_Matrix</code>	Matrix of clustering vectors, one column per trial, or NULL if it cannot be constructed.
<code>ComputationTime</code>	Named numeric vector of elapsed trial times.
<code>Seeds</code>	Named integer vector of seeds, or NULL if <code>SetSeed = FALSE</code> .
<code>CAobjects</code>	Named list containing the complete clustering result from every trial.

For a collection of datasets, returns a list with one result per dataset. Failed datasets are represented by NULL.

Operating-system considerations

The default "PSOCK" cluster type is portable across the operating systems supported by R. Objects required by FUN must be available or serializable on each worker.

Set `WorkersOrNo = NULL` for deterministic serial testing.

Author(s)

Michael Thrun

References

Thrun, M. C.: Distance-Based Clustering Challenges for Unbiased Benchmarking Studies, *Nature Scientific Reports*, Vol. 11, pp. 18988, DOI: 10.1038/s41598-021-98126-1, 2021.

See Also

[parApplyClusterAnalysis](#), [parApplyDistanceBasedCA](#), [makeCluster](#)

Examples

```
simple_clustering <- function(Data, ClusterNo = 2) {
  fit <- stats::kmeans(Data, centers = ClusterNo)
  list(Cls = fit$cluster, Object = fit)
}

set.seed(2)
Data <- matrix(rnorm(80), ncol = 2)

result <- parApplyDataBasedCA(
```

```

    Data = Data,
    FUN = simple_clustering,
    NumberOfTrials = 2,
    ClusterNo = 2,
    WorkersOrNo = NULL
  )

result$Cls_Matrix

```

parApplyDistanceBasedCA

Repeated Application of a Distance-Based Clustering Method

Description

Repeatedly applies a distance-based clustering function to one distance or dissimilarity input, or to a collection of such inputs. Trials may run serially or on a **parallel** worker cluster.

Usage

```

parApplyDistanceBasedCA(
  Distances,
  FUN,
  NumberOfTrials = 1:100,
  ClusterNo = NULL,
  WorkersOrNo,
  SocketType = "PSOCK",
  SetSeed = TRUE,
  ...
)

```

Arguments

Distances	Distance or dissimilarity representation accepted by FUN, or a list of such inputs.
FUN	Function or character string naming a distance-based clustering method. The result should normally be a list containing a unique element named CIs.
NumberOfTrials	Positive integer count or vector of positive integer trial identifiers. A scalar n generates seq_len(n). Default is 1:100.
ClusterNo	Number of clusters passed to FUN. For multiple inputs, this may be scalar or provide one value per input. NULL or NA omits the argument.
WorkersOrNo	Missing, NULL, a positive integer worker count, or an existing parallel cluster. If missing, an automatically selected worker count is used. NULL requests serial execution.
SocketType	Character scalar passed to parallel::makeCluster(). Default is "PSOCK".
SetSeed	Logical scalar. If TRUE, trial i uses seed 1000 + i. Default is TRUE.
...	Further arguments passed to FUN.

Details

The distance input is passed preferentially to a formal argument named `DataOrDistances`, `Distances`, `Distance`, or `Dissimilarities`. If none is present, the first inspectable non-dots argument is used.

An internally created worker cluster is stopped when the function exits. An externally supplied cluster remains under the caller's control.

Value

For one input, a list containing:

<code>Cls_Matrix</code>	Matrix of clustering assignments, one column per trial, or NULL if a complete matrix cannot be constructed.
<code>ComputationTime</code>	Named numeric vector containing elapsed trial times.
<code>Seeds</code>	Named integer vector containing trial seeds, or NULL when <code>SetSeed = FALSE</code> .
<code>CAobjects</code>	Named list containing the complete clustering results.

For a collection of inputs, returns a named list containing one result per distance or dissimilarity input.

Operating-system considerations

The default PSOCK backend is suitable for Windows and is also portable to macOS and Unix-like systems. Use serial execution during package examples and most CRAN tests to avoid platform-dependent worker startup behavior.

Author(s)

Michael Thrun

References

Thrun, M. C.: Distance-Based Clustering Challenges for Unbiased Benchmarking Studies, *Nature Scientific Reports*, Vol. 11, pp. 18988, DOI: 10.1038/s41598-021-98126-1, 2021.

See Also

[parApplyClusterAnalysis](#), [parApplyDataBasedCA](#), [parLapply](#)

Examples

```
distance_clustering <- function(Distances, ClusterNo = 2) {
  fit <- stats::hclust(Distances)
  cls <- stats::cutree(fit, k = ClusterNo)
  list(Cls = cls, Object = fit)
}

set.seed(3)
Data <- matrix(rnorm(40), ncol = 2)
Distances <- stats::dist(Data)
```

```

result <- parApplyDistanceBasedCA(
  Distances = Distances,
  FUN = distance_clustering,
  NumberOfTrials = 2,
  ClusterNo = 2,
  WorkersOrNo = NULL
)

result$Cls_Matrix

```

pdfClustering

*Probability Density Distribution Clustering***Description**

Clustering via non parametric density estimation

Usage

```
pdfClustering(Data, PlotIt = FALSE, ...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

Cluster analysis is performed by the density-based procedures described in Azzalini and Torelli (2007) and Menardi and Azzalini (2014), and summarized in Azzalini and Menardi (2014).

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

Azzalini, A., Menardi, G. (2014). Clustering via nonparametric density estimation: the R package pdfCluster. Journal of Statistical Software, 57(11), 1-26, URL <http://www.jstatsoft.org/v57/i11/>.

Azzalini A., Torelli N. (2007). Clustering via nonparametric density estimation. Statistics and Computing. 17, 71-80.

Menardi, G., Azzalini, A. (2014). An advancement in clustering via nonparametric density estimation. Statistics and Computing. DOI: 10.1007/s11222-013-9400-x.

Examples

```
data('Hepta')
out=pdfClustering(Hepta$Data,PlotIt=FALSE)
```

PFclustering	<i>Power Fuzzy Clustering</i>
--------------	-------------------------------

Description

Performs Power Fuzzy Clustering [Nguyen et al, 2026]

Usage

```
PFclustering(
  Data,
  ClusterNo,
  Fuzzifier = 2,
  DistanceMethod = "Euclidean",
  NoIterations = 100,
  PlotIt = FALSE,
  ...
)
```

Arguments

Data	Numeric data set with n observations and d features.
ClusterNo	Integer or NULL. Number of clusters to search for.
Fuzzifier	Optional, Numeric scalar, strictly greater than 1. Fuzzifier parameter m . Default is 2.
DistanceMethod	Optional, Character scalar. Distance metric to use. Choices are "Minkowski", "Mahalanobis", or "Euclidean".
NoIterations	Optional, Integer scalar. Maximum number of iterations. Default is 100.
PlotIt	Optional, Logical. If TRUE, plots the clustering result via ClusterPlotMDS(). Default is FALSE.
...	Optional, Further parameters passed to <code>pfclust::PFC()</code> , depending on the selected distance. These may include, for example:

- q: distance exponent, must be strictly greater than 0
- p: Minkowski exponent
- alpha, beta: Mahalanobis parameters
- threshold: convergence tolerance

Details

Please see [Nguyen et al, 2026].

Value

A list with components:

- Cls: numeric vector of length nrow(Data) containing cluster assignments.
- Object: object returned by pfclust::PFC().

Author(s)

Michael Thrun

References

P. T. Nguyen, C. Tortora and A. Punzo: “Power Fuzzy Clustering: Flexible Distance Metrics and Inclusion of Covariates,” *IEEE Transactions on Fuzzy Systems*, Vol. 34 (7), pp. 2171–2182, doi: 10.1109/TFUZZ.2026.3683998, 2026.

Examples

```
data('Hepta')

out=PFclustering(Hepta$Data,ClusterNo=7,
PlotIt=FALSE,NoIterations=10)
```

ProjectionPursuitClustering

Cluster Identification using Projection Pursuit as described in [Hofmeyr/Pavlidis, 2019].

Description

Summarizes recent projection pursuit methods for clustering based on [Hofmeyr/Pavlidis, 2015], [Hofmeyr, 2016] and [Pavlidis et al., 2016] .

Usage

```
ProjectionPursuitClustering(Data,ClusterNo,Type="MinimumDensity",

PlotIt=FALSE,PlotSolution=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Type	Either MinimumDensity[Pavlidis et al., 2016] MaximumClusterability[Hofmeyr/Pavlidis, 2015]], or NormalisedCut [Hofmeyr, 2016] or KernelPCA [Hofmeyr/Pavlidis, 2019].
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
PlotSolution	Plots the partitioning solution as a tree as described in
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

The details of the options for projection pursuit and partitioning of data are defined in [Hofmeyr/Pavlidis, 2019].

"KernelPCA" uses additionally the package kernlab and is implemented as given in the fifth example on page 21, section "extension" of [Hofmeyr/Pavlidis, 2019].

The first idea of using non-PCA projections for clustering was published by [Bock, 1987] as an definition. However, to the knowledge of the author it was not applied to any data. The first systematic comparison to Projection-Pursuit Methods [ProjectionPursuitClustering](#) and [AutomaticProjectionBasedClustering](#) can be found in [Thrun/Ultsch, 2018]. For PCA-based clustering methods please see [TandemClustering](#)

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Hofmeyr/Pavlidis, 2015] Hofmeyr, D., & Pavlidis, N.: Maximum clusterability divisive clustering, Proc. 2015 IEEE Symposium Series on Computational Intelligence, pp. 780-786, IEEE, 2015.
- [Hofmeyr/Pavlidis, 2019] Hofmeyr, D., & Pavlidis, N.: PPCI: an R Package for Cluster Identification using Projection Pursuit, The R Journal, 2019.

[Hofmeyr, 2016] Hofmeyr, D. P.: Clustering by minimum cut hyperplanes, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 39(8), pp. 1547-1560. 2016.

[Pavlidis et al., 2016] Pavlidis, N. G., Hofmeyr, D. P., & Tasoulis, S. K.: Minimum density hyperplanes, The Journal of Machine Learning Research, Vol. 17(1), pp. 5414-5446. 2016.

[Thrun/Ultsch, 2018] Thrun, M. C., & Ultsch, A.: Using Projection based Clustering to Find Distance and Density based Clusters in High-Dimensional Data, Journal of Classification, Vol. in revision, 2018.

[Bock, 1987] Bock, H.: On the interface between cluster analysis, principal component analysis, and multidimensional scaling, Multivariate statistical modeling and data analysis, (pp. 17-34), Springer, 1987.

Examples

```
data('Hepta')
out=ProjectionPursuitClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

QTclustering

Stochastic QT Clustering

Description

Stochastic quality clustering of [Heyer et al., 1999] with an improved implementation by [Scharl/Leisch, 2006].

Usage

```
QTclustering(Data,Radius,PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
Radius	Maximum radius of clusters. If NULL, automatic estimation can be done with [Thrun et al., 2016] if not otherwise set.
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of

Cls [1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.

Object Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Heyer et al., 1999] Heyer, L. J., Kruglyak, S., & Yooseph, S.: Exploring expression data: identification and analysis of coexpressed genes, *Genome research*, Vol. 9(11), pp. 1106-1115. 1999.

[Scharl/Leisch, 2006] Scharl, T., & Leisch, F.: The stochastic QT-clust algorithm: evaluation of stability and variance on time-course microarray data, in Rizzi, A. & Vichi, M. (eds.), *Proc. Proceedings in Computational Statistics (Compstat)*, pp. 1015-1022, Physica Verlag, Heidelberg, Germany, 2006.

[Thrun et al., 2016] Thrun, M. C., Lerch, F., Loetsch, J., & Ultsch, A.: Visualization and 3D Printing of Multivariate Data of Biomarkers, in Skala, V. (Ed.), *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, Vol. 24, Plzen, 2016.

[Ultsch, 2005] Ultsch, A.: Pareto density estimation: A density estimation for knowledge discovery, In Baier, D. & Werrnecke, K. D. (Eds.), *Innovations in classification, data science, and information systems*, (Vol. 27, pp. 91-100), Berlin, Germany, Springer, 2005.

Examples

```
data('Hepta')
out=QTclustering(Hepta$Data,Radius=NULL,PlotIt=FALSE)
```

RobustTrimmedClustering

Robust Trimmed Clustering

Description

Robust Trimmed Clustering invented by [Garcia-Escudero et al., 2008] and implemented by [Fritz et al., 2012].

Usage

```
RobustTrimmedClustering(Data, ClusterNo,
Alpha=0.05,PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
Alpha	No trimming is done equals to $\alpha = 0$, otherwise proportion of datapoints to be trimmed, tclust uses 0.05 as default.
...	Further arguments to be set for the clustering algorithm, e.g. <code>.nstart</code> (number of random initializations), <code>iter.max</code> (maximum number of concentration steps), <code>restr</code> and <code>restr.fact</code> described in details. If not set, default arguments are used.

Details

"This iterative algorithm initializes k clusters randomly and performs "concentration steps" in order to improve the current cluster assignment. The number of maximum concentration steps to be performed is given by `iter.max`. For approximately obtaining the global optimum, the system is initialized `nstart` times and concentration steps are performed until convergence or `iter.max` is reached. When processing more complex data sets higher values of `nstart` and `iter.max` have to be specified (obviously implying extra computation time). ... The larger `restr.fact` is chosen, the looser is the restriction on the scatter matrices, allowing for more heterogeneity among the clusters. On the contrary, small values of `restr.fact` close to 1 imply very equally scattered clusters. This idea of constraining cluster scatters to avoid spurious solutions goes back to Hathaway (1985), who proposed it in mixture fitting problems" [Fritz et al., 2012]. The type of constraint `restr` can be set to "eigen", "deter" or "sigma.". Please see [tclust](#) for further parameter description.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [Garcia-Escudero et al., 2008] Garcia-Escudero, L. A., Gordaliza, A., Matran, C., & Mayo-Iscar, A.: A general trimming approach to robust cluster analysis, *The annals of Statistics*, Vol. 36(3), pp. 1324-1345. 2008.
- [Fritz et al., 2012] Fritz, H., Garcia-Escudero, L. A., & Mayo-Iscar, A.: `tclust`: An R package for a trimming approach to cluster analysis, *Journal of statistical Software*, Vol. 47(12), pp. 1-26. 2012.

Examples

```
data('Hepta')

out=RobustTrimmedClustering(Hepta$Data,ClusterNo=7,Alpha=0,PlotIt=FALSE)
```

SharedNearestNeighborClustering
<i>SNN clustering</i>

Description

Shared Nearest Neighbor Clustering of [Ertoz et al., 2003].

Usage

```
SharedNearestNeighborClustering(Data,Knn,

Radius,minPts,PlotIt=FALSE,

UpperLimitRadius,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
Knn	Number of neighbors to consider to calculate the shared nearest neighbors.
Radius	Eps [Ester et al., 1996, p. 227] neighborhood in the R-ball graph/unit disk graph), size of the epsilon neighborhood. If NULL, automatic estimation is done using insights of [Ultsch, 2005].
minPts	Number of minimum points in the eps region (for core points). In principle minimum number of points in the unit disk, if the unit disk is within the cluster (core) [Ester et al., 1996, p. 228]. if NULL, its 2.5 percent of points.
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
UpperLimitRadius	Limit for radius search, experimental
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

..

Value

List of	
Cls	[1:n] numerical vector defining the clustering; this classification is the main output of the algorithm. Points which cannot be assigned to a cluster will be reported as members of the noise cluster with 0.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Ertoz et al., 2003] Levent Ertoz, Michael Steinbach, Vipin Kumar: Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data, SIAM International Conference on Data Mining, 47-59, 2003.

See Also

[sNNclust](#)

Examples

```
data('Hepta')
out=SharedNearestNeighborClustering(
Hepta$Data, Knn=7,Radius=NULL,minPts=NULL,PlotIt = FALSE)
```

SOMclustering	<i>self-organizing maps based clustering implemented by [Whereas, Buydens, 2017].</i>
---------------	---

Description

Either the variant k-batch or k-online is possible in which every unit can be seen approximately as an cluster.

Usage

```
SOMclustering(Data,LC=c(1,2),ClusterNo=NULL,
Mode="online",PlotIt=FALSE,rlen=100,alpha = c(0.05, 0.01),...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
LC	Lines and Columns of a very small SOM, usually every unit is a cluster, will be ignored if ClusterNo is not NULL.
ClusterNo	Optional, A number k which defines k different clusters to be built by the algorithm. LC will then be set accordingly.
Mode	Either "batch" or "online"
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
rlen	Please see supersom
alpha	Please see supersom
...	Further arguments to be set for the clustering algorithm in somgrid , if not set, default arguments are used.

Details

This clustering algorithm is based on very small maps and, hence, not emergent (c.f. [Thrun, 2018, p.37]). A 3x3 map means 9 units leading to 9 clusters.

Batch is a deterministic clustering approach whereas online is a stochastic clustering approach and research indicates that online should be preferred (c.f. [Thrun, 2018, p.37]).

Value

List of	
Cls	[1:n] numerical vector defining the classification as the main output of the clustering algorithm
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Whereas, Buydens, 2017] R. Wehrens and L.M.C. Buydens, J. Stat. Softw. 21 (5), 2007; R. Wehrens and J. Kruisselbrink, submitted, 2017.

[Thrun, 2018] Thrun, M.C., Projection Based Clustering through Self-Organization and Swarm Intelligence. 2018, Heidelberg: Springer.

Examples

```
data('Hepta')
out=SOMclustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

SOTAclustering	<i>SOTA Clustering</i>
----------------	------------------------

Description

Self-organizing Tree Algorithm (SOTA) introduced by [Herrero et al., 2001].

Usage

SOTAclustering(Data, ClusterNo,PlotIt=FALSE,UnrestGrowth,...)

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
UnrestGrowth	TRUE: forces the ClusterNo option to uphold. FALSE: enables the algorithm to find its own number of clusters, in this cases ClusterNo should contain a high number because it is internally set as the number of iterations which is either reached or the max diversity criteria is satisfied priorly.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
sotaObject	Object defined by clustering algorithm as the other output of this algorithm

Note

*Luis Winckelman intergrated several function from clValid because it's ORPHANED.

Author(s)

Luis Winckelmann*, Vasyl Pihur, Guy Brock, Susmita Datta, Somnath Datta

References

[Herrero et al., 2001] Herrero, J., Valencia, A., & Dopazo, J.: A hierarchical unsupervised growing neural network for clustering gene expression patterns, Bioinformatics, Vol. 17(2), pp. 126-136. 2001.

Examples

```
#Does Work
data('Hepta')
out=SOTAClustering(Hepta$Data,ClusterNo=7)
table(Hepta$Cls,out$Cls)

#Does not work well
data('Lsun3D')
out=SOTAClustering(Lsun3D$Data,ClusterNo=100,PlotIt=FALSE,UnrestGrowth=FALSE)
```

SparseClustering	<i>Sparse Clustering</i>
------------------	--------------------------

Description

Implements the sparse clustering methods of [Witten/Tibshirani, 2010].

Usage

```
SparseClustering(DataOrDistances, ClusterNo = 0, Type = "Hierarchical",
                  ColorTreshold = 0, Silent = FALSE, NoPerms = 10,
                  Wbounds, PlotIt = FALSE, ...)
```

Arguments

DataOrDistances	Either a [1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features. or a [1:n,1:n] symmetric distance matrix.
ClusterNo	Numeric indicating number to cluster to find in Tree/ Dendrogramm in case of Type="Hierarchical" or numer of cluster to use in Type="kmeans"
Type	(optional) Char selecting methods Hierarchical or kmeans. Default: "Hierarchical"
ColorTreshold	If Type=="Hierarchical", Finite numeric scalar. If nonzero and ClusterNo = 0, the value is interpreted on the dendrogram height scale. For a monotone ultrametric portion of given dissimilarity, stats::rect.hclust() draws rectangles at this height and classification vector Cls is return based on the cut using the threshold. If these fusion heights are non-monotone, only a horizontal reference line is drawn and Cls=NULL
Silent	(optional) Boolean: print output or not (Default = FALSE = no output)
NoPerms	(optional), numeric scalar, Number of permutations.

Wbounds	(optional) numeric vector, range of tuning parameters to consider. This is the L1 bound on w, the feature weights [Witten/Tibshirani, 2010].
PlotIt	If TRUE and ClusterNo>0 plots dendrogram using ClusterDendrogram If TRUE and ClusterNo==0 plots black dendrogram using base R plot FALSE: No plot
...	Further arguments passed on to <code>sparcl</code> HierarchicalSparseCluster or KMeansSparseCluster depending on Type.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm
Tree	Object Tree if Type="Hierarchical" is used.

Note

Quality of clustering results varies between sparse hierarchical if data is given in comparison to the case that distances are given.

Author(s)

Quirin Stier, Michael Thrun

References

[Witten/Tibshirani, 2010] Witten, D. and Tibshirani, R.: A Framework for Feature Selection in Clustering. Journal of the American Statistical Association, Vol. 105(490), pp. 713-726, 2010.

Examples

```
# Hepta
data("Hepta")
Data = Hepta$Data
V1 = SparseClustering(Data, ClusterNo=7, Type="kmeans")
Cls1 = V1$Cls

V2 = SparseClustering(Data, ClusterNo=7, Type="Hierarchical")
Cls2 = V2$Cls

InputDistances = parallelDist::parDist(Data, method="euclidean")
DistanceMatrix = as.matrix(InputDistances)
V3 = SparseClustering(DistanceMatrix, ClusterNo=7, Type="Hierarchical")
Cls3 = V3$Cls

## Not run:
set.seed(1)
```

```

Data = matrix(rnorm(100*50),ncol=50)
y     = c(rep(1,50),rep(2,50))
Data[y==1,1:25] = Data[y==1,1:25]+2

V1 = SparseClustering(Data, ClusterNo=2, Type="kmeans")
Cls1 = V1$Cls

## End(Not run)

```

SpectralClustering *Spectral Clustering*

Description

Clusters the Data into "ClusterNo" different clusters using the Spectral Clustering method

Usage

```
SpectralClustering(Data, ClusterNo,PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
PlotIt	default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used. e.g.: kernel : Kernelmethod, possible options: rbfdot Radial Basis kernel function "Gaussian" polydot Polynomial kernel function vanilladot Linear kernel function tanhdot Hyperbolic tangent kernel function laplacedot Laplacian kernel function besseldot Bessel kernel function anovadot ANOVA RBF kernel function splinedot Spline kernel stringdot String kernel kpar : Kernelparameter: a character string or the list of hyper-parameters (kernel parameters). The default character string "automatic" uses a heuristic to determine a suitable value for the width parameter of the RBF kernel. "local" (local scaling) uses a more advanced heuristic and sets a width parameter for every point in the data set. A list can also be used containing the parameters to be used with the kernel function.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[Ng et al., 2002] Ng, A. Y., Jordan, M. I., & Weiss, Y.: On spectral clustering: Analysis and an algorithm, Advances in neural information processing systems, Vol. 2, pp. 849-856. 2002.

Examples

```
data('Hepta')
out=SpectralClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

Spectrum	<i>Fast Adaptive Spectral Clustering [John et al, 2020]</i>
----------	---

Description

Spectrum is a self-tuning spectral clustering method for single or multi-view data. In this wrapper restricted to the standard use in other clustering algorithms.

Usage

```
Spectrum(Data, Type = 2, ClusterNo = NULL,

PlotIt = FALSE, Silent = TRUE,PlotResults = FALSE, ...)
```

Arguments

Data	1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
Type	Type=1: default eigengap method (Gaussian clusters) Type=2: multimodality gap method (Gaussian/ non-Gaussian clusters) Type=3: Allows to setClusterNo
ClusterNo	Optional, A number k which defines k different clusters to be built by the algorithm. For default ClusterNo=NULL please see details.

PlotIt	Default: FALSE, If TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
Silent	Silent progress of algorithm=TRUE
PlotResults	Plots result of spectrum with plot function
...	Method: numerical value: 1 = default eigengap method (Gaussian clusters), 2 = multimodality gap method (Gaussian/ non-Gaussian clusters), 3 = no automatic method (see fixk param)
	Other parameters defined in Spectrum packages

Details

Spectrum is a partitioning algorithm and either uses the eigengap or multimodality gap heuristics to determine the number of clusters, please see Spectrum package for details

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

[John et al, 2020] John, C. R., Watson, D., Barnes, M. R., Pitzalis, C., & Lewis, M. J.: Spectrum: Fast density-aware spectral clustering for single and multi-omic data. *Bioinformatics*, Vol. 36(4), pp. 1159-1166, 2020.

See Also

[Spectrum](#)

Examples

```
data('Hepta')

out=Spectrum(Hepta$Data,PlotIt=FALSE)
```

Description

Density estimation for ggplot with a clear model behind it.

Format

The format is: Classes 'StatPDEdensity', 'Stat', 'ggproto' <ggproto object: Class StatPDEdensity, Stat> aesthetics: function compute_group: function compute_layer: function compute_panel: function default_aes: uneval extra_params: na.rm finish_layer: function non_missing_aes: parameters: function required_aes: x y retransform: TRUE setup_data: function setup_params: function super: <ggproto object: Class Stat>

Details

PDE was published in [Ultsch, 2005], short explanation in [Thrun, Ultsch 2018] and the PDE optimized violin plot was published in [Thrun et al., 2018].

References

- [Ultsch,2005] Ultsch, A.: Pareto density estimation: A density estimation for knowledge discovery, in Baier, D.; Werrnecke, K. D., (Eds), Innovations in classification, data science, and information systems, Proc Gfkl 2003, pp 91-100, Springer, Berlin, 2005.
- [Thrun, Ultsch 2018] Thrun, M. C., & Ultsch, A. : Effects of the payout system of income taxes to municipalities in Germany, in Papiez, M. & Smiech,, S. (eds.), Proc. 12th Professor Aleksander Zelias International Conference on Modelling and Forecasting of Socio-Economic Phenomena, pp. 533-542, Cracow: Foundation of the Cracow University of Economics, Cracow, Poland, 2018.
- [Thrun et al, 2018] Thrun, M. C., Pape, F., & Ultsch, A. : Benchmarking Cluster Analysis Methods using PDE-Optimized Violin Plots, Proc. European Conference on Data Analysis (ECDA), accepted, Paderborn, Germany, 2018.

Description

Subspace (projected) clustering is a technique which finds clusters within different subspaces (a selection of one or more dimensions).

Usage

```
SubspaceClustering(Data,ClusterNo,DimSubspace,
Type='Orclus',PlotIt=FALSE,OrclusInitialClustersNo=ClusterNo+2,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases or d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the proclus or orclus algorithm.
DimSubspace	Numerical number defining the dimensionality in which clusters should be search in in the orclus algorithm, for proclus it is an optional parameter
Type	'Orclus', subspace clustering based on arbitrarily oriented projected cluster generation [Aggarwal and Yu, 2000] 'ProClus' ProClus algorithm for subspace clustering [Aggarwal/Wolf, 1999] 'Clique' ProClus algorithm finds subspaces of high-density clusters [Agrawal et al., 1999] and [Agrawal et al., 2005] 'SubClu' SubClu algorithm is a density-connected approach for subspace clustering [Kailing et al.,2004]
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in Cls
OrclusInitialClustersNo	Only for Orclus algorithm: Initial number of clusters (that are computed in the entire data space) must be greater than k. The number of clusters is iteratively decreased by a factor until the final number of k clusters is reached.
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used. For Subclue: "epsilon" and "minSupport", see DBSCAN For Clique: "xi" (number of intervals for each dimension) and "tau" (Density Threshold), see DBSCAN

Details

Subspace clustering algorithms have the goal to finde one or more subspaces with the assumption that sufficient dimensionality reduction is dimensionality reduction without loss of information. Hence subspace clustering aums at finding a linear subspace sucht that the subspace contains as much predictive information as the input space. The subspace is usually higher than two but lower than the input space. In contrast, projection-based clustering [AutomaticProjectionBasedClustering](#) projects the data (nonlinear) into two dimensions and tries only to preerve relevant neighborhoods.

Value

List of	
Cls	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering.
Object	Object defined by clustering algorithm as the other output of this algorithm

Note

subclustering of subclue package is not available on CRAN anymore. you can comment the code in and get the archive version, otherwise only orclus is available as an option.

JAVA_HOME has to be set for rJava to the ProClus algorithm (in windows set PATH env. variable to .../bin path of Java. The architecture of R and Java have to match. Java automatically downloads the Java version of the browser which may not be installed in the architecture in R. In such a case choose a Java version manually.

Author(s)

Michael Thrun

References

[Aggarwal/Wolf et al., 1999] Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., & Park, J. S.: Fast algorithms for projected clustering, Proc. ACM SIGMoD Record, Vol. 28, pp. 61-72, ACM, 1999.

[Aggarwal/Yu, 2000] Aggarwal, C. C., & Yu, P. S.: Finding generalized projected clusters in high dimensional spaces, (Vol. 29), ACM, ISBN: 1581132174, 2000.

[Agrawal et al., 1999]: Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan: Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications, In Proc. ACM SIGMOD, 1999.

[Agrawal et al., 2005] Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P.: Automatic subspace clustering of high dimensional data, Data Mining and Knowledge Discovery, Vol. 11(1), pp. 5-33. 2005.

[Kailing et al., 2004] Kailing, Karin, Hans-Peter Kriegel, and Peer Kroege: Density-connected subspace clustering for high-dimensional data, Proceedings of the 2004 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2004

Examples

```
data('Hepta')
out=SubspaceClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

TandemClustering

Tandem Clustering

Description

Summarizes clustering methods that combine k-means and pca

Usage

```
TandemClustering(Data,ClusterNo,Type="Reduced",PlotIt=FALSE,...)
```

Arguments

Data	[1:n,1:d] matrix of dataset to be clustered. It consists of n cases of d-dimensional data points. Every case has d attributes, variables or features.
ClusterNo	A number k which defines k different clusters to be built by the algorithm.
Type	Reduced: Reduced k-means (RKM) [De Soete/Carroll, 1994]. Factorial: Factorial k-mean (FKM) [Vichi/Kiers, 2001] KernelPCA: Kernel PCA with minimum normalised cut hyperplanes [Hofmeyr/Pavlidis, 2019]
PlotIt	Default: FALSE, if TRUE plots the first three dimensions of the dataset with colored three-dimensional data points defined by the clustering stored in C1s
...	Further arguments to be set for the clustering algorithm, if not set, default arguments are used.

Details

If the ClusterNo exceeds the number of dimensions, than the function is called recursively with ClusterNo=2. In each iteration the cluster with the highest number of overall points is clustered again, until the number of clusters is met.

"KernelPCA" uses additionally the package kernlab and is implemented as given in the fifth example on page 18, section "extension" of [Hofmeyr/Pavlidis, 2019]

The first idea of using non-PCA projections for clustering was published by [Bock, 1987] as an definition. However, to the knowledge of the author it was not applied to any data. The first systematic comparison to Projection-Pursuit Methods [ProjectionPursuitClustering](#) and [AutomaticProjectionBasedClustering](#) can be found in [Thrun/Ultsch, 2018].

Value

List of	
C1s	[1:n] numerical vector with n numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers representing the arbitrary labels of the clustering. Points which cannot be assigned to a cluster will be reported with 0.
Object	Object defined by clustering algorithm as the other output of this algorithm

Author(s)

Michael Thrun

References

- [De Soete/Carroll, 1994] De Soete, G., & Carroll, J. D.: K-means clustering in a low-dimensional Euclidean space, New approaches in classification and data analysis, (pp. 212-219), Springer, 1994.
- [Hofmeyr/Pavlidis, 2019] Hofmeyr, D., & Pavlidis, N.: PPCI: an R Package for Cluster Identification using Projection Pursuit, The R Journal, 2019.

[Vichi/Kiers, 2001] Vichi, M., & Kiers, H. A.: Factorial k-means analysis for two-way data, Computational Statistics & Data Analysis, Vol. 37(1), pp. 49-64. 2001.

[Thrun/Ultsch, 2018] Thrun, M. C., & Ultsch, A.: Using Projection based Clustering to Find Distance and Density based Clusters in High-Dimensional Data, Journal of Classification, Vol. in revision, 2018.

[Bock, 1987] Bock, H.: On the interface between cluster analysis, principal component analysis, and multidimensional scaling, Multivariate statistical modeling and data analysis, (pp. 17-34), Springer, 1987.

Examples

```
data('Hepta')
out=TandemClustering(Hepta$Data,ClusterNo=7,PlotIt=FALSE)
```

Target	<i>Target introduced in [Ultsch, 2005].</i>
--------	---

Description

Detailed description of dataset and its clustering challenge of outliers is provided in [Thrun/Ultsch, 2020]

Usage

```
data("Target")
```

Details

Size 770, Dimensions 2, stored in Target\$Data

Classes 6, stored in Target\$Cls

References

[Ultsch, 2005] Ultsch, A.: U* C: Self-organized Clustering with Emergent Feature Maps, Proc. Lernen, Wissensentdeckung und Adaptivitaet (LWA/FGML), pp. 240-244, Saarbruecken, Germany, 2005.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:[10.1016/j.dib.2020.105501](https://doi.org/10.1016/j.dib.2020.105501), 2020.

Examples

```
data(Target)
str(Target)
```

Tetra	<i>Tetra introduced in [Ultsch, 1993]</i>
-------	---

Description

Almost touching clusters. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("Tetra")
```

Details

Size 400, Dimensions 3, stored in Tetra\$Data

Classes 4, stored in Tetra\$Cls

References

[Ultsch, 1993] Ultsch, A.: Self-organizing neural networks for visualisation and classification, Information and classification, (pp. 307-313), Springer, 1993.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

Examples

```
data(Tetra)
str(Tetra)
```

TwoDiamonds	<i>TwoDiamonds introduced in [Ultsch, 2003a, 2003b]</i>
-------------	---

Description

Cluster border defined by density. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("TwoDiamonds")
```

Details

Size 800, Dimensions 2, stored in TwoDiamonds\$Data

Classes 2, stored in TwoDiamonds\$Cls

References

- [Ultsch, 2003a] Ultsch, A. Optimal density estimation in data containing clusters of unknown structure, technical report, Vol. 34, University of Marburg, Department of Mathematics and Computer Science, 2003.
- [Ultsch, 2003b] Ultsch, A.: U*-matrix: a tool to visualize clusters in high dimensional data, Fachbereich Mathematik und Informatik, 2003.
- [Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

Examples

```
data(TwoDiamonds)
str(TwoDiamonds)
```

WingNut

WingNut introduced in [Ultsch, 2005]

Description

Density vs. distance. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

Usage

```
data("WingNut")
```

Details

Size 1016, Dimensions 2, stored in WingNut\$Data

Classes 2, stored in WingNut\$Cls

References

- [Ultsch, 2005] Ultsch, A.: Clustering with SOM: U* C, Proc. Proceedings of the 5th Workshop on Self-Organizing Maps, Vol. 2, pp. 75-82, 2005.
- [Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

Examples

```
data(WingNut)
str(WingNut)
```

Index

- * **AMI**
 - ClusterAMI, 17
- * **ARI**
 - ClusterARI, 20
- * **Accuracy**
 - ClusteringAccuracy, 31
- * **Affinity Propagation**
 - APclustering, 7
- * **Agglomerative Nesting**
 - AgglomerativeNestingClustering, 5
- * **Agglomerative**
 - GenieClustering, 64
 - HierarchicalClusterData, 70
 - MinimaxLinkageClustering, 93
- * **Atom**
 - Atom, 9
- * **AutomaticProjectionBasedClustering**
 - AutomaticProjectionBasedClustering, 9
- * **Bouldin**
 - ClusterDaviesBouldinIndex, 25
- * **CA**
 - parApplyClusterAnalysis, 106
 - parApplyDataBasedCA, 109
 - parApplyDistanceBasedCA, 111
- * **Chainlink**
 - Chainlink, 12
- * **Cluster Challenge**
 - ClusterChallenge, 22
- * **Cluster Count**
 - ClusterCount, 23
- * **Cluster Dendrogram**
 - ClusterDendrogram, 26
- * **Cluster Normalize**
 - ClusterNormalize, 39
- * **Cluster Sampling**
 - ClusterEqualWeighting, 30
- * **ClusterApply**
 - ClusterApply, 18
- * **ClusterCount**
 - ClusterCount, 23
- * **ClusterCreateClassification**
 - ClusterCreateClassification, 24
- * **ClusterDendrogram**
 - ClusterDendrogram, 26
- * **ClusterEqualWeighting**
 - ClusterEqualWeighting, 30
- * **ClusterNoEstimation**
 - ClusterNoEstimation, 37
- * **ClusterNormalize**
 - ClusterNormalize, 39
- * **ClusterPlotMDS**
 - ClusterPlotMDS, 40
- * **ClusterRenameDescendingSize**
 - ClusterRenameDescendingSize, 44
- * **ClusterRename**
 - ClusterRedefine, 42
 - ClusterRename, 43
- * **Clusterability**
 - ClusterabilityMDplot, 13
- * **Clustering**
 - ClusterChallenge, 22
 - ClusterDaviesBouldinIndex, 25
 - ClusterDunnIndex, 29
 - ClusteringAccuracy, 31
 - DBSCAN, 51
 - DBScusteringAndVisualization, 53
 - EstimateRadiusByDistance, 61
 - GenieClustering, 64
 - HierarchicalClusterData, 70
 - HierarchicalClusterDists, 71
 - HierarchicalClustering, 75
 - HierarchicalDBSCAN, 79
 - kmeansClustering, 82
 - kmeansDist, 85
 - LargeApplicationClustering, 86
 - MeanShiftClustering, 91
 - MinimalEnergyClustering, 92

- MinimaxLinkageClustering, 93
- OPTICSclustering, 103
- pdfClustering, 113
- * **Consecutive Clustering**
 - ClusterNormalize, 39
- * **ConsensusClustering**
 - ConsensusClustering, 48
- * **Create Cluster Classification**
 - ClusterCreateClassification, 24
- * **Cross-Entropy Clustering**
 - CrossEntropyClustering, 49
- * **Cross-Entropy**
 - CrossEntropyClustering, 49
- * **CrossEntropyClustering**
 - CrossEntropyClustering, 49
- * **DBSCAN**
 - DBSCAN, 51
 - HierarchicalDBSCAN, 79
- * **DBS**
 - DBSclusteringAndVisualization, 53
- * **Databionic swarm**
 - DBSclusteringAndVisualization, 53
- * **DatabionicSwarm**
 - DBSclusteringAndVisualization, 53
- * **Davies Bouldin Index**
 - ClusterDaviesBouldinIndex, 25
- * **DaviesBouldinIndex**
 - ClusterDaviesBouldinIndex, 25
- * **Davies**
 - ClusterDaviesBouldinIndex, 25
- * **Dendrogram**
 - ClusterDendrogram, 26
- * **Density Peak Clustering**
 - DensityPeakClustering, 56
- * **Density Peak**
 - DensityPeakClustering, 56
- * **DensityPeakClustering**
 - DensityPeakClustering, 56
- * **Descending Clustering**
 - ClusterRenameDescendingSize, 44
- * **Distances**
 - HierarchicalClusterDists, 71
- * **Divisive Analysis Clustering**
 - DivisiveAnalysisClustering, 58
- * **DivisiveAnalysisClustering**
 - DivisiveAnalysisClustering, 58
- * **Dunn Index**
 - ClusterDunnIndex, 29
- * **DunnIndex**
 - ClusterDunnIndex, 29
- * **EM clustering**
 - MoGclustering, 98
- * **EngyTime**
 - EngyTime, 59
- * **Equal Weighting**
 - ClusterEqualWeighting, 30
- * **Estimation of Number of Clusters**
 - ClusterNoEstimation, 37
- * **Expectation Maximization**
 - MoGclustering, 98
- * **FCPS**
 - Atom, 9
 - Chainlink, 12
 - ClusterChallenge, 22
 - EngyTime, 59
 - FCPS-package, 4
 - GolfBall, 66
 - Hepta, 69
 - Leukemia, 88
 - Lsun3D, 89
 - Spectrum, 127
 - Target, 133
 - Tetra, 134
 - TwoDiamonds, 134
 - WingNut, 135
- * **Fundamental Clustering Problems Suite**
 - FCPS-package, 4
- * **Gap Statistic**
 - GapStatistic, 63
- * **Gap**
 - GapStatistic, 63
- * **Generate Fundamental Clustering Problem**
 - ClusterChallenge, 22
- * **GolfBall**
 - GolfBall, 66
- * **HCLclustering**
 - HCLclustering, 66
- * **HDDC**
 - HDDCclustering, 67
- * **Hard Competitive learning clustering**
 - HCLclustering, 66
- * **Hepta**
 - Hepta, 69
- * **Hierarchical DBSCAN**
 - HierarchicalDBSCAN, 79

- * **Hierarchical**
 - GenieClustering, 64
 - HierarchicalClusterData, 70
 - HierarchicalClusterDists, 71
 - HierarchicalClustering, 75
 - HierarchicalDBSCAN, 79
 - MinimalEnergyClustering, 92
 - MinimaxLinkageClustering, 93
- * **IsDissimilarity**
 - IsDissimilarity, 81
- * **Large Application Clusteringg**
 - LargeApplicationClustering, 86
 - MeanShiftClustering, 91
- * **LargeApplicationClustering**
 - LargeApplicationClustering, 86
- * **Lsun3D**
 - Leukemia, 88
 - Lsun3D, 89
- * **MCC**
 - ClusterMCC, 34
- * **MDS**
 - ClusterPlotMDS, 40
- * **MDplot**
 - ClusterabilityMDplot, 13
- * **MSTclustering**
 - MSTclustering, 100
- * **Markov Clustering**
 - MarkovClustering, 89
- * **Markov**
 - MarkovClustering, 89
- * **Matthews Correlation Coefficient**
 - ClusterMCC, 34
- * **Matthews Correlation**
 - ClusterMCC, 34
- * **Matthews**
 - ClusterMCC, 34
- * **MeanShiftClustering**
 - MeanShiftClustering, 91
- * **Minimal Energy**
 - MinimalEnergyClustering, 92
- * **MinimalEnergy**
 - MinimalEnergyClustering, 92
- * **Minimax Linkage**
 - MinimaxLinkageClustering, 93
- * **Minimax**
 - MinimaxLinkageClustering, 93
- * **Mixture Of Gaussians**
 - ModelBasedClustering, 95
- * **MixtureOfGaussians**
 - ModelBasedClustering, 95
 - MoGclustering, 98
- * **MoG**
 - ModelBasedClustering, 95
 - MoGclustering, 98
- * **Model based clustering**
 - ModelBasedClustering, 95
- * **Model-based clustering**
 - ModelBasedVarSelClustering, 96
- * **Multidimensional scaling**
 - ClusterPlotMDS, 40
- * **NMI**
 - ClusterNMI, 35
- * **Network Clustering**
 - NetworkClustering, 101
- * **Neural Gas**
 - NeuralGasClustering, 102
- * **NeuralGasClustering**
 - NeuralGasClustering, 102
- * **PAM**
 - PAMclustering, 105
- * **PDE**
 - StatPDEdensity, 129
- * **PPCI**
 - ProjectionPursuitClustering, 115
- * **Pareto Density Estimation**
 - StatPDEdensity, 129
- * **Partitioning Around Medoids**
 - PAMclustering, 105
- * **Projection Based Clustering**
 - AutomaticProjectionBasedClustering, 9
- * **Projection Method**
 - ClusterPlotMDS, 40
- * **ProjectionPursuitClustering**
 - ProjectionPursuitClustering, 115
- * **Projection**
 - ClusterPlotMDS, 40
- * **QTClustering**
 - QTclustering, 117
- * **Radius**
 - EstimateRadiusByDistance, 61
- * **Rand**
 - ClusterARI, 20
- * **Rename Descending Cluster Size**
 - ClusterRenameDescendingSize, 44
- * **Rk statistic**

- ClusterMCC, 34
- * **Robust Trimmed Clustering**
 - RobustTrimmedClustering, 118
- * **RobustTrimmedClustering**
 - RobustTrimmedClustering, 118
- * **SMOTE**
 - ClusterUpsamplingMinority, 47
- * **SOM**
 - SOMclustering, 121
- * **SOTAclustering**
 - SOTAclustering, 123
- * **Self-organizing Tree Algorithm**
 - SOTAclustering, 123
- * **Shannon information**
 - ClusterShannonInfo, 45
- * **Shannon**
 - ClusterShannonInfo, 45
- * **SharedNearest Neighbor Clustering**
 - SharedNearestNeighborClustering, 120
- * **Sparse Clustering**
 - SparseClustering, 124
- * **Spectral Clustering**
 - SpectralClustering, 126
 - Spectrum, 127
- * **SpectralClustering**
 - SpectralClustering, 126
- * **Spectrum**
 - Spectrum, 127
- * **Subspace Clustering**
 - SubspaceClustering, 129
- * **SubspaceClustering**
 - SubspaceClustering, 129
- * **Tandem Clustering**
 - TandemClustering, 131
- * **TandemClustering**
 - TandemClustering, 131
- * **Target**
 - Target, 133
- * **Tetra**
 - Tetra, 134
- * **TwoDiamonds**
 - TwoDiamonds, 134
- * **Variable Selection**
 - ModelBasedVarSelClustering, 96
- * **WingNut**
 - WingNut, 135
- * **adjusted mutual information**
 - ClusterAMI, 17
- * **adjusted rand index**
 - ClusterARI, 20
- * **agnes**
 - AgglomerativeNestingClustering, 5
- * **apcluster**
 - APclustering, 7
- * **benchmarking**
 - FCPS-package, 4
 - parApplyClusterAnalysis, 106
 - parApplyDataBasedCA, 109
 - parApplyDistanceBasedCA, 111
- * **clara**
 - LargeApplicationClustering, 86
 - MeanShiftClustering, 91
- * **classification**
 - ClusterAlignLabels, 15
- * **cluster analysis**
 - AgglomerativeNestingClustering, 5
 - DBScusteringAndVisualization, 53
- * **clustering**
 - AgglomerativeNestingClustering, 5
 - ConsensusClustering, 48
 - FCPS-package, 4
 - PAMclustering, 105
 - parApplyClusterAnalysis, 106
 - parApplyDataBasedCA, 109
 - parApplyDistanceBasedCA, 111
- * **cluster**
 - ClusterAlignLabels, 15
 - FCPS-package, 4
- * **data entropy**
 - EntropyOfDataField, 60
- * **data field**
 - EntropyOfDataField, 60
- * **data set**
 - FCPS-package, 4
- * **databionic**
 - DBScusteringAndVisualization, 53
- * **datasets**
 - Atom, 9
 - Chainlink, 12
 - EngyTime, 59
 - GolfBall, 66
 - Hepta, 69
 - Leukemia, 88
 - Lsun3D, 89
 - Target, 133

- Tetra, 134
- TwoDiamonds, 134
- WingNut, 135
- * **density estimation**
 - StatPDEdensity, 129
- * **diana**
 - DivisiveAnalysisClustering, 58
- * **distances**
 - ClusterDistances, 28
 - ClusterInterDistances, 33
 - kmeansDist, 85
- * **distance**
 - IsDissimilarity, 81
- * **dunn**
 - ClusterDunnIndex, 29
- * **entropy**
 - EntropyOfDataField, 60
- * **factor**
 - ClusterCreateClassification, 24
- * **fanny**
 - FannyClustering, 62
- * **fuzzy clustering**
 - FannyClustering, 62
- * **generalized Umatrix**
 - DBSclusteringAndVisualization, 53
- * **ggproto density estimation**
 - StatPDEdensity, 129
- * **information**
 - ClusterAMI, 17
 - ClusterNMI, 35
 - ClusterShannonInfo, 45
- * **inter cluster**
 - ClusterInterDistances, 33
- * **intercluster**
 - ClusterInterDistances, 33
- * **intra cluster**
 - ClusterDistances, 28
- * **intracluster**
 - ClusterDistances, 28
- * **k-batch clustering**
 - SOMclustering, 121
- * **k-batch**
 - SOMclustering, 121
- * **kmeans Clustering**
 - kmeansClustering, 82
 - kmeansDist, 85
- * **kmeansClustering**
 - kmeansClustering, 82
- kmeansDist, 85
- * **leiden**
 - NetworkClustering, 101
- * **louvain**
 - NetworkClustering, 101
- * **model-based clustering**
 - HDDClustering, 67
- * **mst**
 - MSTclustering, 100
- * **mutual information**
 - ClusterAMI, 17
 - ClusterNMI, 35
- * **normalized mutual information**
 - ClusterNMI, 35
- * **optics**
 - OPTICSclustering, 103
- * **over sampling**
 - ClusterUpsamplingMinority, 47
- * **over-sampling**
 - ClusterUpsamplingMinority, 47
- * **parApply**
 - parApplyClusterAnalysis, 106
 - parApplyDataBasedCA, 109
 - parApplyDistanceBasedCA, 111
- * **parallel cluster analysis**
 - parApplyClusterAnalysis, 106
 - parApplyDataBasedCA, 109
 - parApplyDistanceBasedCA, 111
- * **pdfClustering**
 - pdfClustering, 113
- * **snn**
 - SharedNearestNeighborClustering, 120
- * **som clustering**
 - SOMclustering, 121
- * **subspace**
 - HDDClustering, 67
- * **swarm**
 - DBSclusteringAndVisualization, 53
- * **up sampling**
 - ClusterUpsamplingMinority, 47
- * **upsampling**
 - ClusterUpsamplingMinority, 47
- * **utilities**
 - IsDissimilarity, 81
- adjustedRandIndex, 21
- AgglomerativeNestingClustering, 5
- agnes, 6

- APclustering, [7](#)
- as.dendrogram, [70](#), [72](#), [92](#), [94](#)
- Atom, [9](#)
- AutomaticProjectionBasedClustering, [9](#), [116](#), [130](#), [132](#)
- Chainlink, [12](#)
- clara, [87](#)
- ClusterabilityMDplot, [13](#)
- ClusterAccuracy, [34](#), [35](#)
- ClusterAccuracy (ClusteringAccuracy), [31](#)
- ClusterAlignLabels, [15](#), [32](#)
- ClusterAMI, [17](#)
- ClusterApply, [18](#)
- clusterApply, [108](#)
- ClusterARI, [20](#)
- ClusterChallenge, [22](#)
- ClusterCount, [23](#)
- ClusterCreateClassification, [24](#)
- ClusterDaviesBouldinIndex, [25](#)
- ClusterDendrogram, [26](#), [74](#)
- ClusterDistances, [28](#)
- ClusterDunnIndex, [29](#)
- ClusterEqualWeighting, [30](#)
- ClusteringAccuracy, [31](#)
- ClusteringAlgorithms (FCPS-package), [4](#)
- ClusterInterDistances, [29](#), [33](#)
- ClusterIntraDistances (ClusterDistances), [28](#)
- ClusterMCC, [32](#), [34](#)
- ClusterNMI, [35](#)
- ClusterNoEstimation, [37](#)
- ClusterNormalize, [39](#), [45](#)
- ClusterPlotMDS, [22](#), [23](#), [40](#), [47](#)
- ClusterRedefine, [42](#)
- ClusterRename, [43](#)
- ClusterRenameDescendingSize, [40](#), [44](#)
- ClusterShannonInfo, [45](#)
- ClusterUpsamplingMinority, [47](#)
- clustvarsel, [96](#)
- ConsensusClustering, [48](#)
- CrossEntropyClustering, [49](#)
- cutree, [27](#), [74](#)
- DatabionicSwarmClustering (DBScusteringAndVisualization), [53](#)
- DBSCAN, [51](#), [61](#), [130](#)
- DBscan (DBSCAN), [51](#)
- DBScustering, [55](#)
- DBScusteringAndVisualization, [53](#)
- densityClust, [57](#)
- DensityPeakClustering, [56](#)
- dist, [7](#), [82](#)
- DivisiveAnalysisClustering, [58](#)
- EngyTime, [59](#)
- EntropyOfDataField, [60](#)
- EstimateRadiusByDistance, [61](#)
- FannyClustering, [62](#)
- FCPS-package, [4](#)
- GapStatistic, [63](#)
- GeneratePmatrix, [62](#)
- GeneratePswarmVisualization, [55](#)
- GenieClustering, [64](#)
- GolfBall, [66](#)
- HCLclustering, [66](#)
- hclust, [27](#), [74](#)
- hddc, [68](#)
- HDDClustering, [67](#)
- Hepta, [69](#)
- Hierarchical_DBSCAN (HierarchicalDBSCAN), [79](#)
- Hierarchical_DBscan (HierarchicalDBSCAN), [79](#)
- HierarchicalCluster (HierarchicalClusterData), [70](#)
- HierarchicalClusterData, [70](#), [71](#), [74](#), [78](#)
- HierarchicalClusterDists, [71](#), [71](#), [78](#)
- HierarchicalClustering, [38](#), [65](#), [70](#), [71](#), [74](#), [75](#), [93](#), [94](#)
- HierarchicalDBSCAN, [78](#), [79](#)
- HierarchicalSparseCluster, [125](#)
- index.DB, [25](#), [26](#)
- InterClusterDistances (ClusterInterDistances), [33](#)
- IntraClusterDistances (ClusterDistances), [28](#)
- IsDissimilarity, [81](#)
- isSymmetric, [82](#)
- kmeansClustering, [82](#)
- kmeansDist, [85](#)
- KMeansSparseCluster, [83](#), [125](#)

- LargeApplicationClustering, 86
- Leukemia, 88
- Lsun3D, 89

- makeCluster, 110
- MarkovClustering, 89
- MDplot, 15, 29, 34
- MeanShiftClustering, 91
- MinimalEnergyClustering, 92
- MinimaxLinkageClustering, 93
- ModelBasedClustering, 95, 95, 98, 99
- ModelBasedVarSelClustering, 96
- MoGclustering, 96, 98
- mst.knn, 100, 101
- MSTclustering, 100

- NetworkClustering, 101
- NeuralGasClustering, 102

- optics, 104
- OPTICSclustering, 103

- PAMClustering (PAMclustering), 105
- PAMclustering, 105
- parApplyClusterAnalysis, 106, 110, 112
- parApplyDataBasedCA, 109, 112
- parApplyDistanceBasedCA, 110, 111
- parDist, 57, 64, 70, 90, 92, 94, 100
- parLapply, 112
- pdfClustering, 113
- PFclustering, 114
- Plot3D, 41, 42
- plot_ly, 57
- plotTopographicMap, 11, 54
- ProjectionPursuitClustering, 11, 115, 116, 132
- Pswarm, 55

- QTClustering (QTclustering), 117
- QTclustering, 117

- RobustTrimmedClustering, 118

- SharedNearestNeighborClustering, 120
- similarities, 7, 8
- sNNclust, 121
- solve_LSAP, 17
- SOMclustering, 121
- somgrid, 122
- SOTAclustering, 123

- sotaClustering (SOTAclustering), 123
- SparseClustering, 124
- SpectralClustering, 126
- Spectrum, 127, 128
- StatPDEdensity, 129
- SubspaceClustering, 11, 129
- supersom, 122

- TandemClustering, 11, 116, 131
- Target, 133
- tclust, 119
- Tetra, 134
- TwoDiamonds, 134

- VarSelCluster, 96

- WingNut, 135