Package 'ProjectionBasedClustering'

July 24, 2017

Type Package

Title Projection Based Clustering

Version 1.0.1

Date 2017-07-24

Description A clustering approach for every projection method based on the generalized U*-matrix visualization of a topographic map is made available here. The number of clusters and the cluster structure can be estimated by counting the valleys in a topographic map. If the number of clusters and the clustering method are chosen correctly, then the clusters will be well separated by mountains in the visualization. Most projection methods are wrappers for already available methods in R. However, the neighbor retrieval visualizer (NeRV) is based on C++ source code of the 'dredviz' software package and the Curvilinear Component Analysis (CCA) is translated from 'MATLAB' ('SOM Toolbox' 2.0) to R.

License GPL-3

Imports Rcpp, ggplot2, stats, graphics, vegan, deldir, geometry, GeneralizedUmatrix, RcppEigen

Suggests fastICA, tsne, FastKNN, MASS, pcaPP, spdep, methods, pracma, grid, mgcv

LinkingTo Rcpp

LazyData TRUE

Encoding UTF-8

NeedsCompilation yes

Author Michael Thrun [aut, cre],
Florian Lerch [aut],
Felix Pape [aut],
Kristian Nybo [cph],
Jarkko Venna [cph]

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

Repository CRAN

Date/Publication 2017-07-24 19:04:35 UTC

R topics documented:

`ProjectionBasedClustering-package`
Projection Based Clustering

Description

The package is based on a conference talk [Thrun/Ultsch, 2017]. The abstract follows:

Many data mining methods rely on some concept of the dissimilarity between pieces of information encoded in the data of interest. These methods can be used for cluster analysis. However, no generally accepted definition of clusters exists in the literature [Hennig et al., 2015]. Here, consistent with Bouveyron et al., it is assumed that a cluster is a group of similar objects [Bouveyron et al., 2012]. The clusters are called natural because they do not require a dissection; instead, they are clearly separated in the data [Duda et al., 2001, Theodoridis/Koutroumbas, 2009, pp. 579, 600]. These clusters can be identified by distance or density based high-dimensional structures. Dimensionality reduction techniques are able to reduce the dimensions of the input space to facilitate the exploration of structures in high-dimensional data. If they are used for visualization, they are called projection methods. The generalized U*-matrix technique is applicable for these and can be used to visualize both distance- and density-based structures [Thrun 2017; Ultsch/Thrun, 2017]. The idea that the abstract U*-matrix (AU-matrix) can be used for clustering [Ultsch et al., 2016]. The distances required for hierarchical clustering are defined by the AU-matrix [Lötsch/Ultsch, 2014]. Using this distance we propose a clustering approach for every projection method based on the U*-matrix visualization of a topographic map [Thrun 2017; Thrun/Ultsch, 2017]. The number of clusters and

the cluster structure can be estimated by counting the valleys in a topographic map [Thrun et al., 2016]. If the number of clusters and the clustering method are chosen correctly, then the clusters will be well separated by mountains in the visualization. Outliers are represented as volcanoes and can be interactively marked in the visualization after the automated clustering process.

Author(s)

Michael Thrun, Felix Pape, Florian Lerch

References

[Thrun/Ultsch, 2017] Thrun, M.C., Ultsch, A.: Projection based Clustering, accepted for publication at Conf. Int. Federation of Classification Societies, Tokyo, 2017.

[Bouveyron et al., 2012] Bouveyron, C., Hammer, B., & Villmann, T.: Recent developments in clustering algorithms, Proc. ESANN, Citeseer, 2012.

[Duda et al., 2001] Duda, R. O., Hart, P. E., & Stork, D. G.: Pattern classification, (Second Edition ed.), Ney York, USA, John Wiley & Sons, ISBN: 0-471-05669-3, 2001.

[Hennig et al., 2015] Hennig, C., Meila, M., Murtagh, F., & Rocci, R.: Handbook of cluster analysis, New York, USA, CRC Press, ISBN: 9781466551893, 2015.

[Lötsch/Ultsch, 2014] Lötsch, J., & Ultsch, A.: Exploiting the Structures of the U-Matrix, in Villmann, T., Schleif, F.-M., Kaden, M. & Lange, M. (eds.), Proc. Advances in Self-Organizing Maps and Learning Vector Quantization, pp. 249-257, Springer International Publishing, Mittweida, Germany, 2014.

[Theodoridis/Koutroumbas, 2009] Theodoridis, S., & Koutroumbas, K.: Pattern Recognition, (Fourth Edition ed.), Canada, Elsevier, ISBN: 978-1-59749-272-0, 2009.

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

[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, http://wscg.zcu.cz/wscg2016/short/A43-full.pdf, 2016.

[Ultsch et al., 2016] Ultsch, A., Behnisch, M., & Lötsch, J.: ESOM Visualizations for Quality Assessment in Clustering, In Merényi, E., Mendenhall, J. M. & O'Driscoll, P. (Eds.), Advances in Self-Organizing Maps and Learning Vector Quantization: Proceedings of the 11th International Workshop WSOM 2016, Houston, Texas, USA, January 6-8, 2016, (10.1007/978-3-319-28518-4_3pp. 39-48), Cham, Springer International Publishing, 2016.

[Ultsch/Thrun, 2017] Ultsch, A., & Thrun, M. C.: Credible Visualizations for Planar Projections, in Cottrell, M. (Ed.), 12th International Workshop on Self-Organizing Maps and Learning Vector Quantization, Clustering and Data Visualization (WSOM), IEEE Xplore, France, 2017.

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

Examples

```
data('Hepta')
#2d projection
```

```
## Not run: projectionpoints=NeRV(Hepta$Data)
#Computation of Generalized Umatrix
## Not run: visualization=GeneralizedUmatrix(Data = Hepta$Data,projectionpoints)
# Visualizuation of GenerelizedUmatrix
## Not run: plotTopographicMap(visualization$Umatrix,visualization$Bestmatches)
# Automatic Clustering
## Not run: LC=c(visualization$Lines,visualization$Columns)
# number of Cluster from dendrogram or visualization (PlotIt=T)
## Not run: #Cls=ProjectionBasedClustering(k=7, Hepta$Data,

visualization$Bestmatches, LC,PlotIt=T)
## End(Not run)
# Verification
## Not run: plotTopographicMap(visualization$Umatrix,visualization$Bestmatches,Cls)
```

CCA *Curvilinear Component Analysis*

Description

CCA Projects data vectors using Curvilinear Component Analysis.

Unknown values (NaN's) in the data: projections of vectors with unknown components tend to drift towards the center of the projection distribution. Projections of totally unknown vectors are set to unknown (NaN).

Usage

```
CCA(DataOrDists,Epochs,OutputDimension=2,method='euclidean',

alpha0 = 0.5, lambda0,PlotIt=FALSE,Cls)
```

Arguments

DataOrDists	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric epochs (scalar) training length
Epochs	(scalar) training length
OutputDimension	Number of dimensions in the Outputspace, default=2
method	method specified by distance string. One of: 'euclidean','cityblock=manhatten','cosine','chebychev','jacc
alpha0	(scalar) initial step size, 0.5 by default
lambda0	(scalar) initial radius of influence, 3*max(std(D)) by default
PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. OutputDimension>2: only the first two dimensions will be shown
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

A n by OutputDimension matrix containing coordinates of the projected points.

Note

Only Transfered from matlab to R. Matlabversion: Contributed to SOM Toolbox 2.0, February 2nd, 2000 by Juha Vesanto http://www.cis.hut.fi/projects/somtoolbox/

Author(s)

Florian Lerch

References

Demartines, P., Herault, J., "Curvilinear Component Analysis: a Self-Organizing Neural Network for Nonlinear Mapping of Data Sets", IEEE Transactions on Neural Networks, vol 8, no 1, 1997, pp. 148-154.

DefaultColorSequence *Default color sequence for plots*

Description

Defines the default color sequence for plots made within the Projections package.

Usage

```
data("DefaultColorSequence")
```

Format

A vector with 562 different strings describing colors for plots.

Delaunay4Points *Adjacency matrix of the delaunay graph for BestMatches of Points*

Description

Calculates the adjacency matrix of the delaunay graph for BestMatches (BMs) in tiled form if BestMatches are located on a toroid grid

Usage

```
Delaunay4Points(Points, IsToroid = TRUE,Grid=NULL,PlotIt=FALSE)
```

Arguments

Points	[1:n,1:3] matrix containing the BMKey, X and Y coordinates of the n, Best-Matches NEED NOT BE UNIQUE, however, there is an edge in the Deaunay between duplicate points!
IsToroid	OPTIONAL, logical, indicating if BM's are on a toroid grid. Default is True
Grid	OPTIONAL, A vector of length 2, containing the number of lines and columns of the Grid
PlotIt	OPTIONAL, bool, Plots the graph

Details

as

Value

Delaunay[1:n,1:n] adjacency matrix of the Delaunay-Graph

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-Universität Marburg, Marburg, 2017.

DijkstraSSSP *Dijkstra SSSP*

Description

Dijkstra's SSSP (Single source shortest path) algorithm:

gets the shortest path (geodesic distance) from source vertice(point) to all other vertices(points) defined by the edges of the adjasency matrix

Usage

```
DijkstraSSSP(Adj, Costs, source)
```

Arguments

Adj	[1:n,1:n] 0/1 adjascency matrix, e.g. from delaunay graph or gabriel graph
Costs	[1:n,1:n] matrix, distances between n points (normally euclidean)
source	int, vertice(point) from which to calculate the geodesic distance to all other points

Details

Preallocating space for DataStructures accordingly to the maximum possible number of vertices which is fixed set at the number 10001.

Value

ShortestPaths[1:n] vector, shortest paths (geodesic) to all other vertices including the source vertice itself

Note

runs in O(E*Log(V))

Author(s)

Michael Thrun

References

uses a changed code which is inspired by Shreyans Sheth 28.05.2015, see http://ideone.com/qkmt31

Hepta *Hepta from FCPS*

Description

clearly defined clusters, different variances

Usage

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

Details

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

Classes 7, stored in `Hepta$Cls`

References

Ultsch, A.: U* C: Self-organized Clustering with Emergent Feature Maps, Lernen, Wissensentdeckung und Adaptivitaet (LWA), pp. 240-244, Saarbruecken, Germany, 2005.

Examples

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

ICA *Independent Component Analysis)*

Description

Independent Component Analysis:

Negentropie: difference of entropy to a corresponding normally-distributed random variable J(y)=|E(G(y)-E(G(v)))|^2

Usage

```
ICA(Data,OutputDimension=2,Contrastfunction="logcosh",

Alpha=1,Iterations=200,PlotIt=FALSE,Cls)
```

Arguments

Data	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric
OutputDimension	Number of dimensions in the Outputspace, default=2
Contrastfunction	Maximierung der Negentropie ueber geeignete geeignete Kontrastfunktion Default: 'logcosh' G(u)=1/a*log cosh(a*u) 'exp': G(u)=-exp(u^2/2)
Alpha	onstant with 1<=alpha<=2 used in approximation to neg-entropy when fun == "logcosh"
Iterations	maximum number of iterations to perform.
PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. OutputDimension>2: only the first two dimensions will be shown
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints	[1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projectio
Mixing	[1:OutputDimension,1:d] Mischungsmatrix s.d gilt Data=MixingMatrix*ProjectedPoints
Unmixing	Entmischungsmatrix with Data*Unmixing=ProjectedPoints
PCMatrix	pre-whitening matrix that projects data onto the first n.comp principal components.

Note

A wrapper for fastICA

Author(s)

Michael Thrun

`Isomap` *Isomap projection method*

Description

Isomap procetion as introduced in 2000 by Tenenbaum, de Silva and Langford

Even with a manifold structure, the sampling must be even and dense so that dissimilarities along a manifold are shorter than across the folds. If data do not have such a manifold structure, the results are very sensitive to parameter values.

Usage

```
Isomap(Inputdistances,k,OutputDimension=2,PlotIt=FALSE,Cls)
```

Arguments

`Inputdistances`	Matrix containing the distances of the data
`k`	number of k nearest neighbors, if the data is fragmented choose an higher k
`OutputDimension`	Number of dimensions in the output space, default = 2
`PlotIt`	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. If OutputDimension > 2 only the first two dimensions will be shown.
`Cls`	Optional and only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints[1:n,OutputDimension] n by OutputDimension matrix containing coordinates of the Projection: A matrix of the fitted configuration..

Note

A wrapper for isomap of the package vegan

if Data fragmented choose an higher k

Author(s)

Michael Thrun

klrank *Computes rank-based smoothed precision/recall*

Description

Compares the projection in pData with the original data in Data and calculates the rank-based smoothed recall and precision.

Usage

```
klrank(Data, pData, NeighborhoodSize)
```

Arguments

Data	Matrix of original data
pData	Matrix of projected data
NeighborhoodSize	Sets the 'effective number of neighbors' used to control the width of the Gaussian

Value

Rank-based smoothed recall and smoothed precision. It's important to note that all values are the real result substracted from one. So precision_best is (1-precision_best).

Author(s)

Felix Pape

References

Jarkko Venna, Jaakko Peltonen, Kristian Nybo, Helena Aidos, and Samuel Kaski. Information Retrieval Perspective to Nonlinear Dimensionality Reduction for Data Visualization. Journal of Machine Learning Research, 11:451-490, 2010.

KruskalStress *Kruskal stress calculation*

Description

Calculates the stress as defined by Kruskal for 2 distance matrices

Usage

```
KruskalStress(InputDistances, OutputDistances)
```

Arguments

InputDistances	Distance matrix of the original Data
OutputDistances	Distance matrix of the projected Data

Value

A single numerical representing the Kruskal stress of the distance matrices.

Author(s)

Felix Pape

MDS *Classical multidimensional scaling (MDS)*

Description

Classical multidimensional scaling of a data matrix. Also known as principal coordinates analysis

Usage

```
MDS(DataOrDists,method='euclidean',OutputDimension=2,PlotIt=FALSE,Cls)
```

Arguments

DataOrDists	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric
method	method specified by distance string: 'euclidean','cityblock=manhatten','cosine','chebychev','jaccard','m
OutputDimension	Number of dimensions in the Outputspace, default=2
PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization.
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints	[1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projectio
Eigenvalues	the eigenvalues of MDSvalues*MDSvalues'
Stress	Shephard-Kruskal Stress

Note

A wrapper for `cmdscale`

Author(s)

Michael Thrun

`NeRV` *NeRV projection*

Description

Projection is done by the neighbor retrieval visualizer (NeRV)

Usage

```
NeRV(Data, lambda = 0.1, neighbors = 20, iterations = 10,

cg_steps = 2, cg_steps_final = 40, randominit = T, OutputDimension = 2,

PlotIt = FALSE, Cls)
```

Arguments

`Data`	Matrix of the Data to be projected
`lambda`	Optional: Controls the trustworthiness-continuity tradeoff. Default = 0.1
`neighbors`	Optional: Set the number of nearest neighbours that each point should have. Should be positive. Default = 20
`iterations`	Optional: The number of iterations to perform. Default = 10
`cg_steps`	Optional: The number of conjugate gradient steps to perform per iteration in NeRV's optimization scheme. Default = 2
`cg_steps_final`	Optional: The number of conjugate gradient steps to perform on the final iteration in NeRV's optimization scheme. Default = 40
`randominit`	Optional: TRUE: Random Initialization (default), FALSE: PCA initializiation
`OutputDimension`	Optional: Number of dimensions in the Outputspace, default=2
`PlotIt`	Optional: Should the projected points be plotted? Default: FALSE. Note: this is only usefull if OutputDimension = 2.
`Cls`	Optional: Vector containing the number of the class for each row in Data. This is only used to color the points according to their classes if PlotIt = T

Details

Uses the NeRV projection with matrix Data and lambda. Lambda controls the trustworthiness-continuity tradeoff.

Value

OutputDimension-dimensional matrix of projected points

Note

PCA initialization changes form the original C++ Sourcecode of http://research.cs.aalto.fi/pml/software/dredviz/ to the R version of the projections package. Other changes are made only regarding data types of Rcpp in comparison to the original C++ Source code.

Author(s)

Michael Thrun, Felix Pape

References

Jarkko Venna, Jaakko Peltonen, Kristian Nybo, Helena Aidos, and Samuel Kaski. Information Retrieval Perspective to Nonlinear Dimensionality Reduction for Data Visualization. Journal of Machine Learning Research, 11:451-490, 2010.

Jarkko Venna and Samuel Kaski. Nonlinear Dimensionality Reduction as Information Retrieval. In Marina Meila and Xiaotong Shen, editors, Proceedings of AISTATS 2007, the 11th International Conference on Artificial Intelligence and Statistics. Omnipress, 2007. JMLR Workshop and Conference Proceedings, Volume 2: AISTATS 2007.

PCA *Principal component analysis*

Description

Performs a principal components analysis on the given data matrix projection=SammonsMapping(Data)

Usage

```
PCA(Data,OutputDimension=2,Scale=FALSE,Center=FALSE,PlotIt=FALSE,Cls)
```

Arguments

Data	array of data: n cases in rows, d variables in columns
OutputDimension	Number of dimensions in the Outputspace, default=2
Scale	a logical value indicating whether the variables should be scaled to have unit variance before the analysis takes place. The default is FALSE for consistency with S, but in general scaling is advisable. Alternatively, a vector of length equal the number of columns of x can be supplied. The value is passed to scale.
Center	a logical value indicating whether the variables should be shifted to be zero centered. Alternately, a vector of length equal the number of columns of x can be supplied. The value is passed to scale

PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. OutputDimension>2: only the first two dimensions will be shown
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints	[1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projectio
Rotation	the matrix of variable loadings (i.e., a matrix whose columns contain the eigenvectors)
sDev	the standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix, though the calculation is actually done with the singular values of the data matrix)
TransformedData	matrix with PCA transformed Data
Center	the centering used, or FALSE
Scale	the scaling used, or FALSE

Note

A wrapper for prcomp

Author(s)

Michael Thrun

PlotProjectedPoints *Plot Projected Points*

Description

plots XY data colored by Cls with ggplot2

Usage

```
PlotProjectedPoints(Points,Cls,BMUorProjected=F,PlotLegend=FALSE,

xlab='X',ylab='Y',main="Projected Points",PointSize=2.5)
```

Arguments

Points	[1:n,1:2] xy cartesian coordinates of a projection
Cls	numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.
BMUorProjected	Default ==F, If TRUE assuming BestMatches of ESOM instead of Projected Points
PlotLegend	...
xlab	Optional: Label of the x axis
ylab	Optional: Label of the y axis
main	Optional: title
PointSize	Optional: size of points

Value

ggobject of ggplot2

Author(s)

Michael Thrun

ProjectionBasedClustering

automated Clustering approach of the Databonic swarm with abstact U distances

---

Description

automated Clustering approach of the Databonic swarm with abstact U distances, which are the geodesic distances based on high-dimensional distances combined with low dimensional graph paths by using ShortestGraphPathsC.

Usage

```
ProjectionBasedClustering(k, Data, BestMatches, LC, StructureType = TRUE, PlotIt = FALSE,
                  method = "euclidean")
```

Arguments

k	number of clusters, how many to you see in the 3d landscape?
Data	[1:n,1:d] Matrix of Data (n cases, d dimensions) that will be used. One Data-Point per row
BestMatches	[1:n,1:2] Matrix with positions of Bestmatches=ProjectedPoints, one matrix line per data point
LC	grid size c(Lines,Columns)

StructureType	Optional, bool; =TRUE: compact structure of clusters assumed, =FALSE: connected structure of clusters assumed. For the two options vor Clusters, see [Thrun, 2017] or Handl et al. 2006
PlotIt	Optional, bool, Plots Dendrogramm
method	Optional, distance method, do not change

Details

ProjectionBasedClustering is a flexible and robust clustering framework based on a chose projection method and projection method a parameter-free high-dimensional data visualization technique, which generates projected points on a topographic map with hypsometric colors, see package GeneralizedUmatrix function GeneralizedUmatrix, called the generalized U-matrix. The clustering method with no sensitive parameters is done by this function. The clustering can be verified by the visualization and vice versa.

Value

Cls [1:n] vector with selected classes of the bestmatches. You can use `plotTopographicMap(Umatrix,Bestmatches,Cls)` for verification.

Note

If you used pswarm with distance matrix instead of a data matrix you can mds transform your distances into data. The correct dimension can be found through the Sheppard diagram or kruskals stress.

Often it is better to mark the outliers manually after the prozess of clustering; use in this case the visualization `plotTopographicMap` of the package GeneralizedUmatrix. If you would like to mark the outliers interactivly in the visualization use the Umatrix package in https://www.uni-marburg.de/fb12/datenbionik/software-en

Author(s)

Michael Thrun

References

[Thrun/Ultsch, 2017] Thrun, M.C., Ultsch, A.: Projection based Clustering, accepted for publication at Conf. Int. Federation of Classification Societies, Tokyo, 2017.

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

Examples

```
data('Hepta')
#2d projection
## Not run: projectionpoints=NeRV(Hepta$Data)
#Computation of Generalized Umatrix
## Not run: visualization=GeneralizedUmatrix(Data = Hepta$Data,projectionpoints)
# Visualizuation of GenerelizedUmatrix
```

```
## Not run: plotTopographicMap(visualization$Umatrix,visualization$Bestmatches)
# Automatic Clustering
## Not run: LC=c(visualization$Lines,visualization$Columns)
# number of Cluster from dendrogram or visualization (PlotIt=T)
## Not run: #Cls=ProjectionBasedClustering(k=7, Hepta$Data,

visualization$Bestmatches, LC,PlotIt=T)
## End(Not run)
# Verification
## Not run: plotTopographicMap(visualization$Umatrix,visualization$Bestmatches,Cls)
```

`ProjectionPursuit` *Projection Pursuit*

Description

In the absence of a generative model for the data the algorithm can be used to find the projection pursuit directions. Projection pursuit is a technique for finding 'interesting' directions in multidimensional datasets

Usage

```
ProjectionPursuit(Data,OutputDimension=2,Indexfunction="logcosh",

Alpha=1,Iterations=200,PlotIt=FALSE,Cls)
```

Arguments

`Data`	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric
`OutputDimension`	Number of dimensions in the Outputspace, default=2
`Indexfunction`	Criterium for Minimization: default: 'logcosh' G(u)=1/a*log cosh(a*u) (ICA) 'exp': G(u)=-exp(u^2/2) 'kernel' 1/(1* pi)*exp(r/2)
`Alpha`	constant with 1<=alpha<=2 used in approximation to neg-entropy when fun == "logcosh"
`Iterations`	maximum number of iterations to perform.
`PlotIt`	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. OutputDimension>2: only the first two dimensions will be shown
`Cls`	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints
: [1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projectio

Author(s)

MT 06/2015

SammonsMapping *Sammons Mapping*

Description

Improved MDS thorugh a normalization of the Input space

Usage

```
SammonsMapping(DataOrDists,method='euclidean',OutputDimension=2,PlotIt=FALSE,Cls)
```

Arguments

DataOrDists	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric
method	method specified by distance string: 'euclidean','cityblock=manhatten','cosine','chebychev','jaccard','m
OutputDimension	Number of dimensions in the Outputspace, default=2
PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization.
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints
: [1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projectio

Stress
: Shephard-Kruskal Stress

Note

A wrapper for sammon

Author(s)

Michael Thrun

ShepardDiagram *Draw a Shepard diagram*

Description

This function plots a Shepard diagram of InputDist and OutputDist

Usage

```
ShepardDiagram(InputDist, OutputDist, xlabel = "Input Distances",
ylabel = "Output Distances", fancy = F, label = "ProjectionMethod", gPlot = ggplot())
```

Arguments

InputDist	Matrix containing the distances of the inputspace.
OutputDist	Matrix containing the distances of the outputspace.
xlabel	Label of the x axis in the resulting Plot.
ylabel	Label of the y axis in the resulting Plot.
fancy	Set FALSE for PC and TRUE for publication
label	Title of the Shepard diagram
gPlot	Ggplot2 object to plot upon.

Value

ggplot2 object containing the plot.

Author(s)

Michael Thrun

ShortestGraphPathsC *Shortest GraphPaths = geodesic distances*

Description

Dijkstra's SSSP (Single source shortest path) algorithm, from all points to all points

Usage

```
ShortestGraphPathsC(Adj, Cost)
```

Arguments

Adj	[1:n,1:n] 0/1 adjascency matrix, e.g. from delaunay graph or gabriel graph
Cost	[1:n,1:n] matrix, distances between n points (normally euclidean)

Details

Vertices are the points, edges have the costs defined by weights (normally a distance). The algorithm runs in runs in O(n*E*Log(V)), see also [Jungnickel, 2013, p. 87]. Further details can be foubd in [Jungnickel, 2013, p. 83-87].

Value

ShortestPaths[1:n,1:n] vector, shortest paths (geodesic) to all other vertices including the source vertice itself from al vertices to all vertices, stored as a matrix

Note

require C++11 standard (set flag in Compiler, if not set automatically)

Author(s)

Michael Thrun

References

Dijkstra, E. W.: A note on two problems in connexion with graphs, Numerische mathematik, Vol. 1(1), pp. 269-271. 1959.

Jungnickel, D.: Graphs, networks and algorithms, (4th ed ed. Vol. 5), Berlin, Heidelberg, Germany, Springer, ISBN: 978-3-642-32278-5, 2013.

See Also

DijkstraSSSP

---

tSNE *T-distributed Stochastic Neighbor Embedding*

Description

T-distributed Stochastic Neighbor Embedding res = tSNE(Data, KNN=30,OutputDimension=2)

Usage

```
tSNE(DataOrDists,k,OutputDimension=2,method="euclidean",Whitening=TRUE,

InitialDimensions=NULL, Iterations=1000,PlotIt=FALSE,Cls)
```

Arguments

DataOrDists	array of data: n cases in rows, d variables in columns, matrix is not symmetric or distance matrix, in this case matrix has to be symmetric
k	number of k nearest neighbors=number of effective nearest neighbors("perplexity") Important parameter, if not given Settings of package t-SNE will be used
OutputDimension	Number of dimensions in the Outputspace, default=2
method	method specified by distance string: 'euclidean','cityblock=manhatten','cosine','chebychev','jaccard','mi
Whitening	A boolean value indicating whether the matrix data should be whitened
InitialDimensions	The number of dimensions to use in reduction method.
Iterations	maximum number of iterations to perform.
PlotIt	Default: FALSE, If TRUE: Plots the projection as a 2d visualization. OutputDimension>2: only the first two dimensions will be shown
Cls	[1:n,1] Optional,: only relevant if PlotIt=TRUE. Numeric vector, given Classification in numbers: every element is the cluster number of a certain corresponding element of data.

Value

ProjectedPoints[1:n,OutputDimension], n by OutputDimension matrix containing coordinates of the Projection

Note

A wrapper for tsne

Author(s)

Michael Thrun

Index