Package 'DataVisualizations'

October 12, 2022

Type Package

Title Visualizations of High-Dimensional Data

Version 1.2.3

Date 2022-05-21

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

Description Gives access to data visualisation methods that are relevant from the data scientist's point of view. The flagship idea of 'DataVisualizations' is the mirrored density plot (MD-plot) for either classified or non-classified multivariate data published in Thrun, M.C. et al.: ``Analyzing the Fine Structure of Distributions'' (2020), PLoS ONE, <DOI:10.1371/journal.pone.0238835>. The MD-plot outperforms the box-and-whisker diagram (box plot), violin plot and bean plot and geom_violin plot of ggplot2. Furthermore, a collection of various visualization methods for univariate data is provided. In the case of exploratory data analysis, 'DataVisualizations' makes it possible to inspect the distribution of each feature of a dataset visually through a combination of four methods. One of these methods is the Pareto density estimation (PDE) of the probability density function (pdf). Additionally, visualizations of the distribution of distances using PDE, the scatter-density plot using PDE for two variables as well as the Shepard density plot and the Bland-Altman plot are presented here. Pertaining to classified high-dimensional data, a number of visualizations are described, such as f.ex. the heat map and silhouette plot. A political map of the world or Germany can be visualized with the additional information defined by a classification of countries or regions. By extending the political map further, an uncomplicated function for a Choropleth map can be used which is useful for measurements across a geographic area. For categorical features, the Pie charts, slope charts and fan plots, improved by the ABC analysis, become usable. More detailed explanations are found in the book by Thrun, M.C.: ``Projection-Based Clustering through Self-Organization and Swarm Intelligence'' (2018) <DOI:10.1007/978-3-658-20540-9>.

License GPL-3

Imports Rcpp (>= 0.12.12), ggplot2, sp, pracma, reshape2

Suggests plyr, MBA, ggmap, plotrix, rworldmap, rgl, ABCanalysis, choroplethr, dplyr, R6, parallelDist, knitr (>= 1.12), rmarkdown (>= 0.9), vioplot, ggExtra, plotly, htmlwidgets, diptest, moments, signal, DatabionicSwarm, ggrepel, MASS

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.5)

SystemRequirements C++11

LazyLoad yes

LazyData TRUE

URL https://www.deepbionics.org/

VignetteBuilder knitr

BugReports https://github.com/Mthrun/DataVisualizations/issues

NeedsCompilation yes

Author Michael Thrun [aut, cre, cph] (<https://orcid.org/0000-0001-9542-5543>),
Felix Pape [aut, rev],
Onno Hansen-Goos [ctr, ctb],
Hamza Tayyab [ctr, ctb],
Dirk Eddelbuettel [ctr],
Craig Varrichio [ctr],
Alfred Ultsch [dtc, ctb, ctr]

Repository CRAN

Date/Publication 2022-05-21 10:50:02 UTC

R topics documented:

`DataVisualizations-package`

Visualizations of High-Dimensional Data

Description

Gives access to data visualisation methods that are relevant from the data scientist's point of view. The flagship idea of 'DataVisualizations' is the mirrored density plot (MD-plot) for either classified or non-classified multivariate data published in Thrun, M.C. et al.: "Analyzing the Fine Structure of Distributions" (2020), PLoS ONE, <DOI:10.1371/journal.pone.0238835>. The MD-plot outperforms the box-and-whisker diagram (box plot), violin plot and bean plot and geom_violin plot of ggplot2. Furthermore, a collection of various visualization methods for univariate data is provided. In the case of exploratory data analysis, 'DataVisualizations' makes it possible to inspect the distribution of each feature of a dataset visually through a combination of four methods. One of these methods is the Pareto density estimation (PDE) of the probability density function (pdf). Additionally, visualizations of the distribution of distances using PDE, the scatter-density plot using PDE for two variables as well as the Shepard density plot and the Bland-Altman plot are presented here. Pertaining to classified high-dimensional data, a number of visualizations are described, such as f.ex. the heat map and silhouette plot. A political map of the world or Germany can be visualized with the additional information defined by a classification of countries or regions. By extending the political map further, an uncomplicated function for a Choropleth map can be used which is useful for measurements across a geographic area. For categorical features, the Pie charts, slope charts and fan plots, improved by the ABC analysis, become usable. More detailed explanations are found in the book by Thrun, M.C.: "Projection-Based Clustering through Self-Organization and Swarm Intelligence" (2018) <DOI:10.1007/978-3-658-20540-9>.

Details

For a brief introduction to **DataVisualizations** please see the vignette A Quick Tour in Data Visualizations.

Please see https://www.deepbionics.org/. Depending on the context please cite either [Thrun, 2018] regarding visualizations in the context of clustering or [Thrun/Ultsch, 2018] for other visualizations.

For the Mirrored Density Plot (MD plot) please cite [Thrun et al., 2020] and see the extensive vignette in https://md-plot.readthedocs.io/en/latest/index.html. The MD plot is also available in Python https://pypi.org/project/md-plot/

Index of help topics:

```
ABCbarplot              Barplot with Sorted Data Colored by ABCanalysis
AccountingInformation_PrimeStandard_Q3_2019
                        Accounting Information in the Prime Standard in
                        Q3 in 2019 (AI_PS_Q3_2019)
BimodalityAmplitude     Bimodality Amplitude
ChoroplethPostalCodesAndAGS_Germany
                        Postal Codes and AGS of Germany for a
                        Choropleth Map
Choroplethmap           Plots the Choropleth Map
ClassBoxplot            Creates Boxplot plot for all classes
ClassMDplot             Class MDplot for Data w.r.t. all classes
ClassPDEplot            PDE Plot for all classes
ClassPDEplotMaxLikeli   Create PDE plot for all classes with maximum
                        likelihood
```

Classplot	Classplot
CombineCols	Combine vectors of various lengths
Crosstable	Crosstable plot
DataVisualizations-package	Visualizations of High-Dimensional Data
DefaultColorSequence	Default color sequence for plots
DensityScatter	Scatter Density Plot
DualaxisClassplot	Dualaxis Classplot
DualaxisLinechart	DualaxisLinechart
Fanplot	The fan plot
FundamentalData_Q1_2018	Fundamental Data of the 1st Quarter in 2018
GoogleMapsCoordinates	Google Maps with marked coordinates
Heatmap	Heatmap for Clustering
HeatmapColors	Default color sequence for plots
ITS	Income Tax Share
InspectBoxplots	Inspect Boxplots
InspectCorrelation	Inspect the Correlation
InspectDistances	Inspection of Distance-Distribution
InspectScatterplots	Pairwise scatterplots and optimal histograms
InspectStandardization	QQplot of Data versus Normalized Data
InspectVariable	Visualization of Distribution of one variable
JitterUniqueValues	Jitters Unique Values
Lsun3D	Lsun3D inspired by FCPS
MAplot	Minus versus Add plot
MDplot	Mirrored Density plot (MD-plot)
MDplot4multiplevectors	Mirrored Density plot (MD-plot)for Multiple Vectors
MTY	Muncipal Income Tax Yield
OptimalNoBins	Optimal Number Of Bins
PDEplot	PDE plot
PDEscatter	Scatter Density Plot
ParetoDensityEstimation	Pareto Density Estimation V3
ParetoRadius	ParetoRadius for distributions
Piechart	The pie chart
Pixelmatrix	Plot of a Pixel Matrix
Plot3D	3D plot of points
PlotMissingvalues	Plot of the Amount Of Missing Values
PlotProductratio	Product-Ratio Plot
PmatrixColormap	P-Matrix colors
QQplot	QQplot with a Linear Fit
ShepardDensityScatter	Shepard PDE scatter
Sheparddiagram	Draws a Shepard Diagram
SignedLog	Signed Log
Silhouetteplot	Silhouette plot of classified data.

```
Slopechart              Slope Chart
SmoothedDensitiesXY     Smoothed Densities X with Y
StatPDEdensity          Pareto Density Estimation
Worldmap                plots a world map by country codes
categoricalVariable     A categorical Feature.
inPSphere2D             2D data points in Pareto Sphere
stat_pde_density        Calculate Pareto density estimation for ggplot2
                        plots
world_country_polygons
                        world_country_polygons
zplot                   Plotting for 3 dimensional data
```

Author(s)

Michael Thrun, Felix Pape, Onno Hansen-Goos, Alfred Ultsch

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

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, 2018.

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

Examples

```
data("Lsun3D")
Data=Lsun3D$Data

Pixelmatrix(Data)



InspectDistances(as.matrix(dist(Data)))


data("ITS")
data("MTY")
Inds=which(ITS<900&MTY<8000)
plot(ITS[Inds],MTY[Inds],main='Bimodality is not visible in normal scatter plot')

PDEscatter(ITS[Inds],MTY[Inds],xlab = 'ITS in EUR',
```

```
ylab ='MTY in EUR' ,main='Pareto Density Estimation indicates Bimodality' )


MAlist=MAplot(ITS,MTY)

data("Lsun3D")
Cls=Lsun3D$Cls
Data=Lsun3D$Data
#clear cluster structure
plot(Data[,1:2],col=Cls)
#However, the silhouette plot does not indicate a very good clustering in cluster 1 and 2

Silhouetteplot(Data,Cls = Cls)


Heatmap(as.matrix(dist(Data)),Cls = Cls)
```

`ABCbarplot` *Barplot with Sorted Data Colored by ABCanalysis*

Description

This plot can be read like a scree plot for PCA. It allowed to select the most important values visually.

Usage

```
ABCbarplot(Data,

Colors=DataVisualizations::DefaultColorSequence[1:3],

main,xlab,ylab="Value")
```

Arguments

`Data`	[1:n] vector of Data, e.g. eigenvalues of PCA
`Colors`	three colors for A, B and C
`main`	title of plot
`xlab`	xlabel
`ylab`	ylabel

Details

ABC analysis is explained in **ABCanalysis**. The visualization is based on **ggplot2**.

Value

List V of

`ABCanalysis` output of **ABCanalysis**

`ggobject` object of **ggplot2** plotted

`DF` Data frame if another plot should be done manually

Author(s)

Michael Thrun

References

Ultsch. A ., Lotsch J.: Computed ABC Analysis for Rational Selection of Most Informative Variables in Multivariate Data, PloS one, Vol. 10(6), pp. e0129767. doi 10.1371/journal.pone.0129767, 2015.

See Also

`ABCanalysis`

Examples

```
data('FundamentalData_Q1_2018')
Data=as.matrix(FundamentalData_Q1_2018$Data)
Data[!is.finite(Data)]=0
results=prcomp(Data)
main="Scree plot with Class A of the Most-Important Eigenvalues"
plotlist = ABCbarplot(results$sdev,ylab='Eigenvalues',main=main)
plotlist$ggobject
```

`AccountingInformation_PrimeStandard_Q3_2019`
Accounting Information in the Prime Standard in Q3 in 2019 (AI_PS_Q3_2019)

Description

Accounting Information of 261 companies traded in the Frankfurt stock exchange in the German Prime standard.

Usage

```
data("AccountingInformation_PrimeStandard_Q3_2019")
```

Format

A list with of three objects

`Key` [1:n] Key of the 261 obeservations

`Data` [1:n,1:d] numeric matrix of 261 observations on the 45 variables describing the accounting information

`Cls` [1:n] a numeric vector of k clusters of the clustering performend in [Thrun/Ultsch, 2019]

Details

Detailed data description can be found in [Thrun/Ultsch, 2019].

Source

Yahoo Finance

References

[Thrun/Ultsch, 2019] Thrun, M. C., & Ultsch, A.: Stock Selection via Knowledge Discovery using Swarm Intelligence with Emergence, IEEE Intelligent Systems, Vol. under review, pp., 2019.

Examples

```
data(AccountingInformation_PrimeStandard_Q3_2019)

str(AI_PS_Q3_2019)
dim(AI_PS_Q3_2019$Data)
```

`BimodalityAmplitude` *Bimodality Amplitude*

Description

Computes the Bimodality Amplitude of [Zhang et al., 2003]

Usage

```
BimodalityAmplitude(x, PlotIt=FALSE)
```

Arguments

`x`	Data vector.
`PlotIt`	FALSE, TRUE if a figure with the antimodes and peaks is plotted

Details

This function calculates the Bimodality Ampltiude of a data vector. This is a measure of the proportion of bimodality and the existence of bimodality. The value lies between zero and one (that is: [0,1]) where the value of zero implies that the data is unimodal and the value of one implies the data is two point masses.

Note

function was rewritten after the flow of a function of Sathish Deevi because the original function was incorrect.

Author(s)

Michael Thrun

References

Zhang, C., Mapes, B., & Soden, B.: Bimodality in tropical water vapour, Quarterly Journal of the Royal Meteorological Society, Vol. 129(594), pp. 2847-2866, 2003.

Examples

```
#Example 1
data<-c(rnorm(299,0,1),rnorm(299,5,1))
BimodalityAmplitude(data,TRUE)

#Example 2
dist1<-rnorm(2100,5,2)
dist2<-dist1+11
data<-c(dist1,dist2)

BimodalityAmplitude(data,TRUE)

#Example 3
dist1<-rnorm(210,-15,1)
dist2<-rep(dist1,3)+30
data<-c(dist1,dist2)

BimodalityAmplitude(data,TRUE)

#Example 4
data<-runif(1000,-15,1)

BimodalityAmplitude(data,TRUE)
```

categoricalVariable *A categorical Feature.*

Description

Character vector of length 391029 with five different labels.

Usage

```
data("categoricalVariable")
```

Examples

```
data(categoricalVariable)
unique(categoricalVariable)
```

Choroplethmap *Plots the Choropleth Map*

Description

A thematic map with areas colored in proportion to the measurement of the statistical variable being displayed on the map. A political map geneated by this function was used in the conference talk of the publication [Thrun/Ultsch, 2018].

Usage

```
Choroplethmap(Counts, PostalCodes, NumberOfBins = 0,

 Breaks4Intervals, percentiles = c(0.5, 0.95),

 digits = 0, PostalCodesShapes, PlotIt = TRUE,

 DiscreteColors, HighColorContinuous = "red",

 LowColorContinuous = "deepskyblue1", NAcolor = "grey",

 ReferenceMap = FALSE, main = "Political Map of Germany",

 legend = "Range of values", Silent = TRUE)
```

Arguments

`Counts`	vector [1:m], statistical variable being displayed
`PostalCodes`	vector[1:n], currently german postal codes (zip codes), if `PostalCodesShapes` is not changed manually, does not need to be unique
`NumberOfBins`	Default: 1; 1 or below continously changes the color as defined by the package `choroplethr`. A Number between 2 and 9 sets equally sized bins. Higher numbers are not allowed
`Breaks4Intervals`	If NumberOfBins>1 you can set here the intervals of the bins manually
`percentiles`	If NumberOfBins>1 and Breaks4Intervals not set, then the percentiles of min and max bin can be set here. See also `quantile`.
`digits`	number of digits for `round`
`PostalCodesShapes`	Specially prepared shape file with postal codes and geographic boundaries. If you set this object, then you can use non german zip codes. You can see the required structure in map.df, github trulia choroplethr blob master r chloropleth. The German PostalCodesShapes can be downloaded from `https://github.com/Mthrun/DataVisualizations/blob/master/data/GermanPostalCodesShapes.rda`.
`PlotIt`	Either Plot the map directly or change the object manually before plotting it
`DiscreteColors`	Set the discrete colors manually if NumberOfBins>1, else it is ignored
`HighColorContinuous`	if NumberOfBins<=1: color of highest continuous value, else it is ignored
`LowColorContinuous`	if NumberOfBins<=1: color of lowest continuous value, else it is ignored
`NAcolor`	Color of NA values in the map (postal codes without any counts)
`ReferenceMap`	TRUE: With Google map, FALSE: without Google map
`main`	title of plot
`legend`	title of legend
`Silent`	TRUE: disable warnings of `choroplethr` package FALSE: enable warnings of `choroplethr` package

Details

This wrapper for the **choroplethr** enables to visualize a political map easily in the case of german zip codes based on given counts and postal codes. Other postal codes are in principle usable.

Value

List of

`chorR6obj`	An R6 object of the package `choroplethr`
`DataFrame`	Transformed PostalCodes and Counts in a way that they can be used in the package `choroplethr`.

Note

You could read https://www.r-bloggers.com/2016/05/case-study-mapping-german-zip-codes-in-r/, if you want to change the map (PostalCodesShapes shape object).

Author(s)

Michael Thrun

References

[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.

See Also

Google choroplethr package.

Examples

```
#If you download the package from CRAN
## Not run:
# 1. Step: Downlaod the shape file from the website
# https://github.com/Mthrun/DataVisualizations/blob/master/data/GermanPostalCodesShapes.rda
# 2. Step: load it from the local path od the downloaded file with
load(file='GermanPostalCodesShapes.rda')

## End(Not run)

# If you download the package from GitHub, you can omit the two steps above.
# Then, do not use the 'PostalCodesShapes' input parameter

#Many postal codes are required to see a structure
#Exemplary two postal codes in the upper left corner of the map

## Not run:
out=Choroplethmap(c(4,8,5,4),

c('49838', '26817', '49838', '26817'),

NumberOfBins=2,PlotIt=FALSE,
 PostalCodesShapes=GermanPostalCodesShapes)

out$chorR6obj$render()

## End(Not run)
#bins are only presented in the map if the have values within
## Not run:
out=Choroplethmap(c(4,8,5,4),c('49838', '26817',
```

```
 '49838', '26817'),NumberOfBins=5,

 Breaks4Intervals=c(1,2,3,5,10),PlotIt=FALSE,
 PostalCodesShapes=GermanPostalCodesShapes)


out$chorR6obj$render()

## End(Not run)
# Result of [Thrun/Ultsch, 2018]
# Slightly misuse the function for visualizing a political map
# resulting out of a clustering

## Not run:
data('ChoroplethPostalCodesAndAGS_Germany')
res=Choroplethmap(as.numeric(ChoroplethPostalCodesAndAGS_Germany$Cls)+1,

ChoroplethPostalCodesAndAGS_Germany$PLZ,NumberOfBins = 2,

Breaks4Intervals = c(0,1,2,3,4,5,6),digits = 1,ReferenceMap = F,

DiscreteColors = c('white','green','blue','red','magenta'),

main = 'Classification of German Postal Codes based on Income Tax Share and Yield',

legend = 'ITS vs MTY Classification in 2010',NAcolor = 'black',PlotIt=FALSE,
 PostalCodesShapes=GermanPostalCodesShapes)


#takes time to process
res$chorR6obj$render()

## End(Not run)
```

ChoroplethPostalCodesAndAGS_Germany
Postal Codes and AGS of Germany for a Choropleth Map

Description

Zip Codes and Community Identification Number of Germany which can be used in a Choropleth Map.

Usage

```
data("ChoroplethPostalCodesAndAGS_Germany")
```

Format

A data frame with 8702 observations on the following 4 variables.

- `PLZ` German postal codes/zip codes
- `Cls` Clustering aggregated of germany postal codes by `MTY` and `ITS` features
- `AGS` It is the 'Amtlicher Gemeindeschluessel' (Community Identification Number) of German municipalities
- `Names` Names of municipalities

Details

CLS are the the labels of a MTS versus ITS Bayesian classification showing two main groups of low quota ('1') and high quota ('2') municipalities. Additionally, outliers are manually classified into two separated groups called sponsors ('3') and promoted ('4'). In the Bayesian Classification non classified data have the label '0'. If a 'AGS' code of a 'PLZ' was unclear than the label is 'NaN'.

Class	0	low quota	high quota	sponsors	promoted	non classified	unclear mapping
Labels	0	1	2	3	4	5	NaN
CountPerClass	31	1325	7239	10	95	5	2

Source

Generated for [Thrun/Ultsch, 2018] using the approach of [Ultsch/Behnisch, 2017].

References

[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.

[Ultsch/Behnisch, 2017] Ultsch, A., Behnisch, M.: Effects of the payout system of income taxes to municipalities in Germany, Applied Geography, Vol. 81, pp. 21-31, 2017.

Examples

```
data(ChoroplethPostalCodesAndAGS_Germany)
str(ChoroplethPostalCodesAndAGS_Germany)
```

`ClassBoxplot` *Creates Boxplot plot for all classes*

Description

Boxplot the data for all classes

Usage

```
ClassBoxplot(Data, Cls, ColorSequence = DataVisualizations::DefaultColorSequence,

 ClassNames = NULL,All=FALSE, PlotLegend = TRUE,

 main = 'Boxplot per Class', xlab = 'Classes', ylab = 'Range of Data')
```

Arguments

`Data`	Vector of the data to be plotted
`Cls`	Vector of class identifiers.
`ColorSequence`	Optional: The sequence of colors used, Default: DefaultColorSequence()
`ClassNames`	Optional: The names of the classes. Default: C1 - C(Number of Classes)
`All`	Optional: adds full data vector for comparison against classes
`PlotLegend`	Optional: Add a legent to plot. Default: TRUE)
`main`	Optional: Title of the plot. Default: "ClassBoxPlot""
`xlab`	Optional: Title of the x axis. Default: "Classes"
`ylab`	Optional: Title of the y axis. Default: "Data"

Value

A List of

`ClassData`	The DataFrame used to plot
`ggobject`	The ggplot2 plot object

in mode `invisible`

Author(s)

Michael Thrun, Felix Pape

Examples

```
data(ITS)
#please download package from cran
#model=AdaptGauss::AdaptGauss(ITS)
#Classification=AdaptGauss::ClassifyByDecisionBoundaries(ITS,

#DecisionBoundaries = AdaptGauss::BayesDecisionBoundaries(model$Means,model$SDs,model$Weights))

DataVisualizations::ClassBoxplot(ITS,Classification)$ggobject
```

ClassMDplot *Class MDplot for Data w.r.t. all classes*

Description

Creates a Mirrored-Density plot w.r.t. to each class of a numerical vector of data.

Usage

```
ClassMDplot(Data, Cls, ColorSequence = DataVisualizations::DefaultColorSequence,

                        ClassNames = NULL, PlotLegend = TRUE,Ordering = "Columnwise",

                            main = 'MDplot for each Class',

                            xlab = 'Classes', ylab = 'PDE of Data per Class',

                            MinimalAmoutOfData=40,

                            MinimalAmoutOfUniqueData=12,SampleSize=1e+05,...)
```

Arguments

`Data`	[1:n] Vector of the data to be plotted
`Cls`	[1:n] Vector of class identifiers of k clusters one number is the label of one cluster
`ColorSequence`	Optional: [1:k] vector, The sequence of colors used, Default: DataVisualizations::DefaultColorSequence
`ClassNames`	Optional: [1:k] named numerical vector, The names of the classes. Default: Class 1 - Class k with k beeing the number of classes
`PlotLegend`	Optional: Add a legent to plot. Default: TRUE)
`Ordering`	Optional: Ordering of Classes, please see `MDplot` for details)
`main`	Optional: Title of the plot. Default: MDplot for each Class
`xlab`	Optional: Title of the x axis. Default: "Classes"
`ylab`	Optional: Title of the y axis. Default: "Data"
`MinimalAmoutOfData`	Optional: numeric value defining a threshold. Below this threshold no density estimation is performed and a Jitter plot with a median line is drawn. Please see `MDplot` for details.
`MinimalAmoutOfUniqueData`	Optional: numeric value defining a threshold. Below this threshold no density estimation and statistical testing is performed and a Jitter plot is drawn. Only Data Science experts should change this value after they understand how the density is estimated (see [Ultsch, 2005]).

`SampleSize` Optional: numeric value defining a threshold. Above this thresholdclass-wise uniform sampling of finite cases is performed in order to shorten computation time. If required, `SampleSize=n` can be set to omit this procedure.

`...` Further arguments that are documented in `MDplot` except for `OnlyPlotOutput` which is always true.

Details

Further examples for the ClassMDplot can be found in https://md-plot.readthedocs.io/en/latest/application/example_application.html.

The `Cls` vector is reordered from lowest to highest number. The `ClassNames` vector and `ColorSequence` vectors are matched by this ordering of `Cls`, i.e. the lowest number gets the first color or class name.

Value

A List of

`ClassData` The matrix [1:m,1:NoOfClasses] used to plot with the reordered Cls, rows are filled partly with NaN, m is the length of the number of data in largest class.

`ggobject` The ggplot2 plot object

in mode `invisible`

Note

Function is still experimental because `ColorSequence` does not work yet, because we are unable to specify the colors in ggplot2. If someone knows a solution, please mail the maintainer of the package. Similar issue for `PlotLegend`.

Author(s)

Michael Thrun, Felix Pape

References

Thrun, M. C., Breuer, L., & Ultsch, A. : Knowledge discovery from low-frequency stream nitrate concentrations: hydrology and biology contributions, Proc. European Conference on Data Analysis (ECDA), Paderborn, Germany, 2018.

See Also

https://md-plot.readthedocs.io/en/latest/application/example_application.html

`MDplot` https://pypi.org/project/md-plot/

Examples

```
data(ITS)

#shortcut for example if AdaptGauss not installed
Classification = kmeans(ITS, centers = 2)$cluster

#better approach
#please download package from cran
#model=AdaptGauss::AdaptGauss(ITS)
#Classification=AdaptGauss::ClassifyByDecisionBoundaries(ITS,

#DecisionBoundaries = AdaptGauss::BayesDecisionBoundaries(model$Means,model$SDs,model$Weights))
ClassNames=c(1,2)
names(ClassNames)=c("Insert name \n of Class 1","Insert name \n  of Class 2")
ClassMDplot(ITS,Classification,ClassNames = ClassNames)
```

`ClassPDEplot` *PDE Plot for all classes*

Description

PDEplot the data for all classes, weights the pdf with priors

Usage

```
ClassPDEplot(Data, Cls, ColorSequence,

 ColorSymbSequence, PlotLegend = 1,

 SameKernelsAndRadius = 0, xlim, ylim, ...)
```

Arguments

`Data`	The Data to be plotted
`Cls`	Vector of class identifiers. Can be integers or NaN's, need not be consecutive nor positive
`ColorSequence`	Optional: the sequence of colors used, Default: DefaultColorSequence
`ColorSymbSequence`	Optional: the plot symbols used (theoretisch nicht notwendig, da erst wichtig, wenn mehr als 562 Cluster)
`PlotLegend`	Optional: add a legent to plot (default == 1)

SameKernelsAndRadius
: Optional: Use the same PDE kernels and radii for all distributions (default == 0)

xlim
: Optional: range of the x axis

ylim
: Optional: range of the y axis

...
: further arguments passed to plot

Value

Kernels of the Pareto density estimation in mode `invisible`

Author(s)

Michael Thrun

Examples

```
data(ITS)
#please download package from cran
#model=AdaptGauss::AdaptGauss(ITS)
#Classification=AdaptGauss::ClassifyByDecisionBoundaries(ITS,

#DecisionBoundaries = AdaptGauss::BayesDecisionBoundaries(model$Means,model$SDs,model$Weights))

DataVisualizations::ClassPDEplot(ITS,Classification)$ggobject
```

ClassPDEplotMaxLikeli *Create PDE plot for all classes with maximum likelihood*

Description

PDEplot the data for allclasses, weight the Plot with 1 (= maximum likelihood)

Usage

```
ClassPDEplotMaxLikeli(Data, Cls, ColorSequence = DataVisualizations::DefaultColorSequence,

 ClassNames, PlotLegend = TRUE, MinAnzKernels = 0,PlotNorm,

 main = "Pareto Density Estimation (PDE)",

 xlab = "Data", ylab = "ParetoDensity", xlim, ylim, lwd=1, ...)
```

Arguments

Data	The Data to be plotted
Cls	Vector of class identifiers. Can be integers or NaN's, need not be consecutive nor positive
ColorSequence	Optional: the sequence of colors used, Default: DefaultColorSequence
ClassNames	Optional: the names of the classes to be displayed in the legend
PlotLegend	Optional: add a legent to plot (default == 1)
MinAnzKernels	Optional: Minimum number of kernels
PlotNorm	Optional: ==1 => plot Normal distribuion on top , ==2 = plot robust normal distribution,; default: PlotNorm= 0
main	Optional: Title of the plot
xlab	Optional: title of the x axis
ylab	Optional: title of the y axis
xlim	Optional: area of the x-axis to be plotted
lwd	Optional: area of the y-axis to be plotted
ylim	numerical scalar defining the width of the lines
...	further arguments passed to plot

Value

Kernels	Kernels of the distributions
ClassParetoDensities	Pareto densities for classes
ggobject	ggplot2 plot object. This should be used to further modify the plot

Author(s)

Felix Pape

References

Aubert, A. H., Thrun, M. C., Breuer, L., & Ultsch, A. : Knowledge discovery from high-frequency stream nitrate concentrations: hydrology and biology contributions, Scientific reports, Nature, Vol. 6(31536), pp. doi 10.1038/srep31536, 2016.

Examples

```
data(ITS)
#model=AdaptGauss::AdaptGauss(ITS)
##please download package from cran
#Classification=AdaptGauss::ClassifyByDecisionBoundaries(ITS,
```

```
#DecisionBoundaries = AdaptGauss::BayesDecisionBoundaries(model$Means,model$SDs,model$Weights))

DataVisualizations::ClassPDEplotMaxLikeli(ITS,Classification)$ggobject
```

`Classplot` *Classplot*

Description

Allows to plot one time series or feauture with a classification as a labeled scatter plot with a line. The colors are the labels defined by the classification. Usefull to see if temporal clustering has time dependent variations and for Hidden Markov Models (see Mthrun/RHmm on GitHub).

Usage

```
Classplot(X, Y, Cls,Names=NULL,

na.rm=FALSE, xlab , ylab ,

main = "Class Plot", Colors,Size=8,

LineColor = NULL, LineWidth = 1, LineType = NULL,

Showgrid = TRUE, Plotter, pch,SaveIt = FALSE,...)
```

Arguments

`X`	[1:n] numeric vector or time
`Y`	[1:n] numeric vector of feature
`Cls`	[1:n] numeric vector of k classes, if not set per default every point is in first class
`Names`	[1:n] character vector of k classes, if not set perdefault Cls is used, if set, names the legend and the points
`na.rm`	Function may not work with non finite values. If these cases should be automatically removed, set parameter TRUE
`xlab`	Optional, string for xlabel
`ylab`	Optional, string for ylabel
`main`	Optional, string for title of plot
`Colors`	Optional, string defining the k colors, one per class
`Size`	Optional, size of points
`LineColor`	Optional, name of color, in plotly then all points are connected by a curve, in ggplot2 all points of one class ae connected by a curve of the color the class
`LineWidth`	Optional, number defining the width of the curve (plotly only)

LineType	Optional, string defining the type of the curve in plotly only, "dot", "dash", "-" for ggplot2: just set =1 here and then the curve is plotted
Showgrid	Optional, boolean (plotly only)
Plotter	Optional, either "ggplot", "plotly", or "native"
pch	scalar, symbol to use, for native Plotter only
SaveIt	Optional, boolean, if true saves plot as html (plotly) or png (ggplot2)
...	Further arguments for ggplot2::ggplot,or plotly::plot_ly, or plot (except "pch"" and "type") depending on Plotter

Details

Default is "plotly" if Names are NULL. However, ggplot2 is preferable in case that Names parameter is used because overlapping text labels are avoided. In that case the default is "ggplot". Note that ggplot2 options are currently slightly restricted.

Value

plotly object or ggplot2 objected depending on Plotter

Author(s)

Michael Thrun

See Also

DualaxisClassplot

Examples

```
data(Lsun3D)
Classplot(Lsun3D$Data[,1],Lsun3D$Data[,2],Lsun3D$Cls)

#plotly with line
data(Lsun3D)
Classplot(Lsun3D$Data[,1],Lsun3D$Data[,2],Lsun3D$Cls,
LineType="-",LineColor = "green")

#ggplot2 with line and labels
data(Lsun3D)
Classplot(Lsun3D$Data[,1],Lsun3D$Data[,2],Lsun3D$Cls,
Names = rownames(Lsun3D$Data),Size =2,LineType = 1)
```

`CombineCols` *Combine vectors of various lengths*

Description

Combine arbitrary vectors of data, filling in missing rows with NaN

Usage

```
CombineCols(...)
```

Arguments

`...`	d vectors of arbitrary lengths, see example

Details

Robust alternative to `cbind` that fills missing values with nan instead of extending length of vector by duplicating elements

Value

matrix of dimensionality of n x d with n beeing the length of the longest vector and d the number of vectors given as input

Note

special application by MCT of rowr cbind.fill which is now not on CRAN anymore

Author(s)

Craig Varrichio

Examples

```
CombineCols(c(1,2,3),c(1),c(2,3))
```

`Crosstable` *Crosstable plot*

Description

Presents a heatmap with values and a cross table of given Data matrix of two features and a bin width or percentualized values. In this approach the bin width is fixes. A more general way to approach this is the kernel density estimation plot of `PDEscatter`.

Usage

```
Crosstable(Data, xbins = seq(0, 100, 5), ybins = xbins,

NormalizationFactor = 1, PlotIt = TRUE, main='Cross Table',

PlotText=TRUE,TextDigits=0,TextProbs=c(0.05,0.95))
```

Arguments

`Data`	[1:n,1:2] matrix of two features from which the cross table should be generated from
`xbins`	[1:k] start of k bins as a vector generated with `seq` of the first feature of data. Default setting assumes percentiled values between zero and 100.
`ybins`	[1:k] start of k bins as a vector generated with `seq` of the second feature of data. Normally the same for both features, other settings are only possible if the length `k` is equal.
`NormalizationFactor`	Optional, Data feautures can be seen as regular time series, e.g. 1 measurement for a minute, in this case it is useful to normalize the output, e.g. to hours, then `NormalizationFactor=60`
`PlotIt`	Optional, Plots the heatmap if `TRUE`. The first feature is on the x-axis (left to right) and the second on y-axis (bottom to top).
`main`	In case of for `PlotIt=TRUE`: title of plot, see `title`
`PlotText`	In case of for `PlotIt=TRUE`: Default TRUE: plots text in heatmap with the values of the crosstable
`TextDigits`	In case of for `TextDigits=TRUE`: integer indicating the number of decimal places to use in `round`.
`TextProbs`	In case of for `TextDigits=TRUE`: [1:2] numeric vector of two probabilities defining the thresholds for white text to grey text and grey text to black text, e.g. below the first threshold (Default 0.05) all values (5% of values) will be printed in white because the lowest values of the heatmap are blue. The second value of 0.95 works well if cross table has many zeros; uses `quantile` internally.

Details

The interval in each bin is closed to the left and opened to the right. The cross table can be seen as a two-dimensional histogram. The idea to add histograms to the table is taken from [Charpentier. 2014].

Value

The cross table in `invisible` mode which depicts the number of values (frequency) in an specific range with regard to two features.

The first feature is on the x-axis (left to right), and the second on y-axis (top to bottom) contrary to the plot where it is bottom to top.

Note

For non percentiled values the `PlotText` part does not seem always to work, but I currently dont know why the text does not always overlap with the heatmap.

Author(s)

Michael Thrun

References

[Charpentier. 2014] Charpentier, Arthur, ed. Computational actuarial science with R. CRC Press, 2014.

See Also

`table`, `image`, `PDEscatter`

Examples

```
data(ITS)
data(MTY)
#simple but not a good transformation
Data=(cbind(ITS/max(ITS),MTY/max(MTY)))*100
#choice for bins could be better
Crosstable(Data)
```

`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.

`DensityScatter` *Scatter Density Plot*

Description

Density estimation (PDE) [Ultsch, 2005] or "SDH" [Eilers/Goeman, 2004] used for a scatter density plot.

Usage

```
DensityScatter(X,Y, DensityEstimation="SDH",

SampleSize, na.rm=FALSE,PlotIt=TRUE,

NrOfContourLines=20,Plotter='native', DrawTopView = TRUE,

xlab, ylab, main="DensityScatter",

xlim, ylim, Legendlab_ggplot="value",...)
```

Arguments

`X` Numeric vector [1:n], first feature (for x axis values)

`Y` Numeric vector [1:n], second feature (for y axis values)

`DensityEstimation`
"SDH" is very fast but maybe not correct, "PDE" is slow but proably more correct.

`SampleSize` Numeric, positiv scalar, maximum size of the sample used for calculation. High values increase runtime significantly. The default is that no sample is drawn

`na.rm` Function may not work with non finite values. If these cases should be automatically removed, set parameter TRUE

`PlotIt` `TRUE`: plots with function call

`FALSE`: Does not plot, plotting can be done using the list element `Handle`

`NrOfContourLines`
Numeric, number of contour lines to be drawn. 20 by default.

`Plotter` String, name of the plotting backend to use. Possible values are: `"native"`, `"ggplot"`, `"plotly"`

`DrawTopView`	Boolean, True means contur is drawn, otherwise a 3D plot is drawn. Default: TRUE
`xlab`	String, title of the x axis. Default: "X", see `plot()` function
`ylab`	String, title of the y axis. Default: "Y", see `plot()` function
`main`	string, the same as "main" in `plot()` function
`xlim`	see `plot()` function
`ylim`	see `plot()` function
`Legendlab_ggplot`	String, in case of `Plotter="ggplot"` label for the legend. Default: "value"
`...`	Density specifc parameters, for `PDEscatter()` or SDH (nbins,lambda,Xkernels,Ykernel))

Details

The `DensityScatter` function generates the density of the xy data as a z coordinate. Afterwards xyz will be plotted either as a contour plot or a 3d plot. It assumens that the cases of x and y are mapped to each other meaning that a `cbind(x,y)` operation is allowed. This function plots the Density on top of a scatterplot. Variances of x and y should not differ by extreme numbers, otherwise calculate the percentiles on both first. If `DrawTopView=FALSE` only the plotly option is currently available. If another option is chosen, the method switches automatically there.

`PlotIt=FALSE` is usefull if one likes to perform adjustements like axis scaling prior to plotting with **ggplot2** or **plotly**. In the case of `"native"`" the handle returns `NULL` because the basic R functon `plot()` is used

Value

List of:

`X`	Numeric vector [1:m],m<=n, first feature used in the plot or the kernels used
`Y`	Numeric vector [1:m],m<=n, second feature used in the plot or the kernels used
`Densities`	Number of points within the ParetoRadius of each point, i.e. density information
`Handle`	Handle of the plot object. Information-string if native R plot is used.

Note

MT contributed with several adjustments

Author(s)

Felix Pape

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, (Ultsch, A. & Huellermeier, E. Eds., 10.1007/978-3-658-20540-9), Doctoral dissertation, Heidelberg, Springer, ISBN: 978-3658205393, 2018.

[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.

[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.

[Eilers/Goeman, 2004] Eilers, P. H., & Goeman, J. J.: Enhancing scatterplots with smoothed densities, Bioinformatics, Vol. 20(5), pp. 623-628. 2004.

Examples

```
#taken from [Thrun/Ultsch, 2018]
data("ITS")
data("MTY")
Inds=which(ITS<900&MTY<8000)
plot(ITS[Inds],MTY[Inds],main='Bimodality is not visible in normal scatter plot')

DensityScatter(ITS[Inds],MTY[Inds],DensityEstimation="SDH",xlab = 'ITS in EUR',

ylab ='MTY in EUR' ,main='Smoothed Densities histogram indicates Bimodality' )

DensityScatter(ITS[Inds],MTY[Inds],DensityEstimation="PDE",xlab = 'ITS in EUR',

ylab ='MTY in EUR' ,main='PDE indicates Bimodality' )
```

`DualaxisClassplot` *Dualaxis Classplot*

Description

Allows to plot two time series or features with one or two classification(a) as labeled scatter plots. The colors are the labels defined by the classification. Usefull to see if temporal clustering has time dependent variations and for Hidden Markov Models (see Mthrun/RHmm on GitHub).

Usage

```
DualaxisClassplot(X, Y1, Y2, Cls1,

Cls2, xlab = "X", y1lab = "Y1", y2lab = "Y2",

main = "Dual Axis Class Plot", Colors, Showgrid = TRUE, SaveIt = FALSE)
```

Arguments

`X`	[1:n] numeric vector or time
`Y1`	[1:n] numeric vector of feauture

`Y2`	[1:n] numeric vector of feauture
`Cls1`	[1:n] numeric vector defining a classification of k1 classes
`Cls2`	Optional, [1:n] numeric vector defining a classification of k2 classes for `Y2`
`xlab`	Optional, string
`y1lab`	Optional, string
`y2lab`	Optional, string
`main`	Optional, string
`Colors`	[1:(k1+k2)] Colornames
`Showgrid`	Optional, boolean
`SaveIt`	Optional, boolean

Value

plotly object

Author(s)

Michael Thrun

See Also

`Classplot`

Examples

```
##ToDo
```

`DualaxisLinechart` *DualaxisLinechart*

Description

A line chart with dual axisSS

Usage

```
DualaxisLinechart(X, Y1, Y2, xlab = "X",

y1lab = "Y1", y2lab = "Y2", main = "Dual Axis Line Chart",

cols = c("black", "blue"),Overlaying="y", SaveIt = FALSE)
```

Arguments

`X`	[1:n] vector, both lines require the same xvalues, e.g. the time of the time series, `POSIXlt` or `POSIXct` are accepted
`Y1`	[1:n] vector of first line
`Y2`	[1:n] vector of second line
`xlab`	Optional, string for xlabel
`y1lab`	Optional, string for first ylabel
`y2lab`	Optional, string for second ylabel
`main`	Optional, title of plot
`cols`	Optional, color of two lines
`Overlaying`	Change only default in case of using `subplot`
`SaveIt`	Optional, default FALSE; TRUE if you want to save plot as html in `getwd()` directory

Details

enables to visualize to lines in one plot overlaying them using ploty (e.g. two time series with two ranges of values)

Value

`plotly` object

Author(s)

Michael Thrun

Examples

```
#subplot renames the numbering of subsequent plots
y1=runif(100,0,1)
y2=rnorm(100,m=5,s=1)
DualaxisLinechart(1:100, y1, y2,main="Random Time series")


y1=runif(100,0,1)
y2=(1:100*3+4)*runif(100,0,1)
p1=DualaxisLinechart(1:100, y1, y2,main="Random Time series",Overlaying="y2")

y3=1:100*(-2)+4
y4=rnorm(100,m=0,s=2)
p2=DualaxisLinechart(1:100, y3, y4,main="Random Time series",Overlaying="y4")
plotly::subplot(p1,p2)
```

`Fanplot` *The fan plot*

Description

The better alternative to the pie chart represents amount of values given in data.

Usage

```
Fanplot(Datavector,Names,Labels,MaxNumberOfSlices,main='',col,

MaxPercentage=FALSE,ShrinkPies=0.05,Rline=1.1)
```

Arguments

`Datavector` [1:n] a vector of n non unique values

`Names` Optional, [1:k] names to search for in Datavector, if not set `unique` of Datavector is calculated.

`Labels` Optional, [1:k] Labels if they are specially named, if not Names are used.

`MaxNumberOfSlices` Default is k, integer value defining how many labels will be shown. Everything else will be summed up to `Other`.

`main` Optional, title below the fan pie, see `plot`

`col` Optional, the default are the first [1:k] colors of the default color sequence used in this package, otherwise a character vector of [1:k] specifying the colors analog to `plot`

`MaxPercentage` default FALSE; if true the biggest slice is 100 percent instead of the biggest procentual count

`ShrinkPies` Optional, distance between biggest and smallest slice of the pie

`Rline` Optional, the distance between text and pie is defined here as the length of the line in numerical numbers

Details

A normal pie plot is dificult to interpret for a human observer, because humans are not trained well to observe angles [Gohil, 2015, p. 102]. Therefore, the fan plot is used. As proposed in [Gohil 2015] the `fan.plot`() of the `plotrix` package is used to solve this problem. If Number of Slices is higher than MaxNumberOfSlices then `ABCanalysis` is applied (see [Ultsch/Lotsch, 2015]) and group A chosen. If Number of Slices in group A is higher than MaxNumberOfSlices, then the most important ones out of group A are chosen. If MaxNumberOfSlices is higher than Slices in group A, additional slices are shown depending on the percentage (from high to low).

Color sequence is automatically shortened to the MaxNumberOfSlices used in the fan plot.

Value

silent output by calling `invisible` of a list with

`Percentages`	[1:k] percent values visualized in fanplot
`Labels`	[1:k] see input `Labels`, only relevant ones

Author(s)

Michael Thrun

References

[Gohil, 2015] Gohil, Atmajitsinh. R data Visualization cookbook. Packt Publishing Ltd, 2015.

[Ultsch/Lotsch, 2015] Ultsch. A ., Lotsch J.: Computed ABC Analysis for Rational Selection of Most Informative Variables in Multivariate Data, PloS one, Vol. 10(6), pp. e0129767. doi 10.1371/journal.pone.0129767, 2015.

Examples

```
data(categoricalVariable)
Fanplot(categoricalVariable)
```

`FundamentalData_Q1_2018` *Fundamental Data of the 1st Quarter in 2018*

Description

This dataset was extracted out of Yahoo finance and was investigated in [Thrun et al., 2019] and clustered in [Thrun, 2019].

Usage

```
data("FundamentalData_Q1_2018")
```

Format

The format is: List of 3 $ Data :'data.frame': 269 obs. of 45 variables: ..$ TotalRevenue : num [1:269] 3779000 78225 48220 63726 3084 ... ..$ CostofRevenue : num [1:269] 2348000 60835 26174 35203 882 ... ..$ GrossProfit : num [1:269] 1431000 17390 22046 28523 2202 ... ..$ SellingGeneralandAdministrative : num [1:269] 459000 NaN 15162 17072 2005 ... ..$ Others : num [1:269] -3000 10272 -52 3131 1784 ... ..$ TotalOperatingExpenses : num [1:269] 2872000 73833 41284 56787 5081 ... ..$ OperatingIncomeorLoss : num [1:269] 907000 4392 6936 6939 -1997 ... ..$ TotalOtherIncomeDIVxpensesNet : num [1:269] -28000 -344 1 -210 -240 ... ..$ EarningsBeforeInterestandTaxes : num [1:269] 907000 4392 6936 6939 -1997 ... ..$ InterestExpense : num [1:269] -20000 -415 NaN -243 -238 ... ..$ IncomeBeforeTax : num [1:269] 879000 4048 6937 6729

-2237 ... ..$ IncomeTaxExpense : num [1:269] 233000 1365 2188 1896 7 ... ..$ NetIncomeFromContinuingOps : num [1:269] 646000 2683 4749 4833 -2244 ... ..$ NetIncome_x : num [1:269] 644000 2817 4645 4833 -2244 ... ..$ NetIncome : num [1:269] 644000 2817 4645 4833 -2244 ... ..$ CashAndCashEquivalents : num [1:269] 926000 29047 45911 94859 11217 ... ..$ NetReceivables : num [1:269] 2527000 46171 20774 151952 2774 ... ..$ Inventory : num [1:269] 2011000 471 NaN 10572 8924 ... ..$ TotalCurrentAssets : num [1:269] 5674000 80224 68061 267187 25989 ... ..$ LongTermInvestments : num [1:269] 234000 450 NaN 4155 872 ... ..$ PropertyPlantandEquipment : num [1:269] 4216000 14561 3093 32247 7073 ... ..$ IntangibleAssets : num [1:269] 78000 40706 3975 6169 125 ... ..$ OtherAssets : num [1:269] 810000 8224 1091 2978 13310 ... ..$ DeferredLongTermAssetCharges : num [1:269] 759000 684 1091 784 1405 ... ..$ TotalAssets : num [1:269] 11262000 167807 83155 351220 47369 ... ..$ AccountsPayable : num [1:269] 1442000 10567 1698 17316 1386 ... ..$ ShortDIVurrentLongTermDebt : num [1:269] 1275000 30192 NaN 26668 917 ... ..$ OtherCurrentLiabilities : num [1:269] 1064000 36942 22781 92297 2659 ... ..$ TotalCurrentLiabilities : num [1:269] 2577000 54430 24479 114210 4299 ... ..$ OtherLiabilities : num [1:269] 1795000 19435 6876 29347 2018 ... ..$ TotalLiabilities : num [1:269] 5576000 97136 31355 165628 6980 ... ..$ CommonStock : num [1:269] 198000 14946 5198 15250 28644 ... ..$ RetainedEarnings : num [1:269] NaN 44030 34767 40374 -8965 ... ..$ TreasuryStock : num [1:269] 5455000 11686 NaN 129968 20710 ... ..$ OtherStockholderEquity : num [1:269] 5455000 11686 NaN 129968 20710 ... ..$ TotalStockholderEquity : num [1:269] 5653000 70662 51212 185592 40389 ... ..$ NetTangibleAssets : num [1:269] 5325000 6314 40302 140939 40264 ... ..$ Depreciation : num [1:269] 156000 2728 331 1381 410 ... ..$ AdjustmentsToNetIncome : num [1:269] 216000 1911 116 2912 39 ... ..$ ChangesInOtherOperatingActivities : num [1:269] -20000 -2174 -829 NaN 428 ... ..$ TotalCashFlowFromOperatingActivities : num [1:269] 452000 7349 4274 -8241 -1367 ... ..$ CapitalExpenditures : num [1:269] -88000 -966 -1778 -2067 -155 ... ..$ TotalCashFlowsFromInvestingActivities: num [1:269] 30000 -879 -1766 -2746 -484 ... ..$ TotalCashFlowsFromFinancingActivities: num [1:269] -789000 -6660 -21867 -961 -204 ... ..$ ChangeInCashandCashEquivalents : num [1:269] -306000 -215 2508 -11842 -2062 ... $ Names: chr [1:269, 1:6] "1COV" "A1OS" "AAD" "AAG" ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:6] "Key" "ISIN" "Company" "Sector" ... $ Cls : num [1:269] 1 1 1 1 2 1 1 1 3 1 ...

Details

Stocks are selected by the German Prime standard accoridingly to the "Names" data frame. Fundamental Data with missing values is stored in "Data". The rownames of "Data" have the same Key as the first row of "Names" which is the trading symbol. "Cls" provides the clustering as a numerical vector of 1:k classes performed by Databionic Swarm in [Thrun, 2019].

Source

Yahoo finance

References

Thrun, M. C., : Knowledge Discovery in Quarterly Financial Data of Stocks Based on the Prime Standard using a Hybrid of a Swarm with SOM, in Verleysen, M. (Ed.), European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Vol. 27, pp. 397-402, Ciaco, ISBN: 978-287-587-065-0, Bruges, Belgium, 2019.

[Thrun et al., 2019] Thrun, M. C., Gehlert, Tino, & Ultsch, A. : Analyzing the Fine Structure of Distributions, arXiv:1908.06081, 2019.

Examples

```
data(FundamentalData_Q1_2018)
## maybe str(FundamentalData_Q1_2018) ; plot(FundamentalData_Q1_2018) ...
```

`GoogleMapsCoordinates` *Google Maps with marked coordinates*

Description

Google Maps with marked coordinates.

Usage

```
GoogleMapsCoordinates(Longitude,Latitude,Cls=rep(1,length(Longitude)),
zoom=3,location= c(mean(Longitude),mean(Latitude)),stroke=1.7,size=6,sequence)
```

Arguments

`Longitude`	sphaerischer winkel der Kugeloberflaeche, coord 1
`Latitude`	sphaerischer winkel der Kugeloberflaeche, coord 2
`Cls`	Vorklassification/Clusterung
`zoom`	map zoom, an integer from 3 (continent) to 21 (building), default value 10 (city). openstreetmaps limits a zoom of 18, and the limit on stamen maps depends on the maptype. "auto" automatically determines the zoom for bounding box specifications, and is defaulted to 10 with center/zoom specifications. maps of the whole world currently not supported
`location`	Optional, default: c(mean(Longitude),mean(Latitude); an address, longitude/latitude pair (in that order), or left/bottom/right/top bounding box
`stroke`	Optional, plotting parameter, dicke der linien der coordiantensymbole
`size`	Optional, plotting parameter, groesse der koordinatensymbole
`sequence`	Optional, vector of length of number of clusers with numbers indicating the plotting symbols and colors to use

Details

This plot was used in [Thrun, 2018, p. 135].

Value

ggobject()

Note

requires an Internet connection, requires an API key of Google. See `?ggmap::register_google` for details.

Author(s)

Michael Thrun

References

[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.

`Heatmap` *Heatmap for Clustering*

Description

Heatmap of Distances of Data sorted by Cls. Clustering algorithms provide a Classifcation of data, where the labels are defined as a numeric vector `Cls`. Then, a typical cluster-respectively group structure is displayed by the `Heatmap` function. At the margin of the heatmap a dendrogram can be shown, if hierarchical cluster algorithms are used [Wilkinson,2009]. Here the dendrogram has to be shown separately and only the heatmap itself is displayed

Usage

```
Heatmap(DataOrDistances,Cls,method='euclidean',

LowLim=0,HiLim,LineWidth=0.5,Clabel="Cluster No.")
```

Arguments

`DataOrDistances`
: if not symmetric, then the function assumes a [1:n,1:d] numeric matrix of n data cases in rows amd d variables in columns. In this case, the distance metric specifed in `method` will be used.

  Otherwise,

  [1:n,1:n] distance matrix that is symmetric

`Cls`
: [1:n] numerical vector of numbers defining the classification as the main output of the clustering algorithm. It has k unique numbers for k clusters that represent the arbitrary labels of the clustering, assuming a descending order of 1 to k. If not ordered please use `ClusterRenameDescendingSize`. Otherwise x and y label will be incorrect.

`method`
: Optional, if `DataOrDistances` is a [1:n,1:d] not symmetric numerical matrix, please see `parDist` for accessible distance methods, default is Euclidean

`LowLim`
: Optional: limits for the color axis

`HiLim`
: Optional: limits for the color axis

`LineWidth`
: Width of lines seperating the clusters in the heatmap

`Clabel`
: Default "`Cluster No.`", for large number of clusters abbrevations can be used like "`Cls No.`" or "`C`" in order to fit as the x and y axis labels

Details

"Cluster heatmaps are commonly used in biology and related fields to reveal hierarchical clusters in data matrices. Heatmaps visualize a data matrix by drawing a rectangular grid corresponding to rows and columns in the matrix and coloring the cells by their values in the data matrix. In their most basic form, heatmaps have been used for over a century [Wilkinson, 2012]. In addition to coloring cells, cluster heatmaps reorder the rows and/or columns of the matrix based on the results of hierarchical clustering. (...) . Cluster heatmaps have high data density, allowing them to compact large amounts of information into a small space [Weinstein, 2008]", [Engle, 2017].

The procedure can be adapted to distance matrices [Thrun, 2018]. Then, the color scale is chosen such that pixels of low distances have blue and teal colors, pixels of middle distances yellow colors, and pixels of high distances have orange and red colors [Thrun, 2018]. The distances are ordered by the clustering and the clusters are divided by black lines. A clustering is valid if the intra-cluster distances are distinctively smaller that inter-cluster distances in the heatmap [Thrun, 2018]. For another example, please see [Thrun, 2018] (Fig. 3.7, p. 31).

Value

object of ggplot2

Author(s)

Michael Thrun

References

[Wilkinson,2009] Wilkinson, L., & Friendly, M.: The history of the cluster heat map, The American Statistician, Vol. 63(2), pp. 179-184. 2009.

[Engle et al., 2017] Engle, S., Whalen, S., Joshi, A., & Pollard, K. S.: Unboxing cluster heatmaps, BMC bioinformatics, Vol. 18(2), pp. 63. 2017.

[Weinstein, 2008] Weinstein, J. N.: A postgenomic visual icon, Science, Vol. 319(5871), pp. 1772-1773. 2008.

[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, 2018.

See Also

`Pixelmatrix`

Examples

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

#Data
Heatmap(Data,Cls = Cls)
```

```
#Distances
Heatmap(as.matrix(dist(Data)),Cls = Cls)
```

HeatmapColors — *Default color sequence for plots*

Description

Defines the default color sequence for plots made with PixelMatrixPlot

Usage

```
data("HeatmapColors")
```

Format

A vector with different strings describing colors for this plot.

inPSphere2D — *2D data points in Pareto Sphere*

Description

This function determines the 2D data points inside a ParetoSphere with ParetoRadius.

Usage

```
inPSphere2D(data, paretoRadius=NULL)
```

Arguments

`data`	numeric matrix of data.
`paretoRadius`	numeric value. radius of P-spheres. If not given, calculate by the function 'paretoRad'

Value

numeric vector with the number of data points inside a P-sphere with ParetoRadius.

Author(s)

Felix Pape

InspectBoxplots *Inspect Boxplots*

Description

Enables to inspect the boxplots for multiple variables in ggplot2 syntax. Each boxplot also has a point for the mean of the variable.

Usage

```
InspectBoxplots(Data, Names,Means=TRUE)
```

Arguments

`Data`	Matrix containing the data. Each column is one variable.
`Names`	Optional: Names of the variables. If missing the columnnames of data are used.
`Means`	Optional: TRUE: with mean, FALSE: Only median.

Value

The ggplot object of the boxplots

Author(s)

Felix Pape

Examples

```
x <- cbind(A = rnorm(200, 1, 3), B = rnorm(100, -2, 5))
InspectBoxplots(x)
```

InspectCorrelation *Inspect the Correlation*

Description

Inspects the correlation between two given features using density scatter plots.

Usage

```
InspectCorrelation(X, Y, DensityEstimation = "SDH",

CorMethod = "spearman", na.rm = TRUE,

SampleSize = round(sqrt(5e+08), -3),

NrOfContourLines = 20, Plotter = "native",

DrawTopView = T, xlab, ylab,

main = "Spearman correlation coef.:", xlim, ylim,

Legendlab_ggplot = "value", ...)
```

Arguments

`X`	Numeric vector [1:n], first feature (for x axis values)
`Y`	Numeric vector [1:n], second feature (for y axis values)
`DensityEstimation`	"SDH" is very fast but maybe not correct, "PDE" is slow but proably more correct.
`CorMethod`	method of correlation of the cor function, One of "pearson" (default), "kendall", or "spearman
`SampleSize`	Numeric, positiv scalar, maximum size of the sample used for calculation. High values increase runtime significantly. The default is that no sample is drawn
`na.rm`	Function may not work with non finite values. If these cases should be automatically removed, set parameter TRUE
`NrOfContourLines`	Numeric, number of contour lines to be drawn. 20 by default.
`Plotter`	String, name of the plotting backend to use. Possible values are: `"native"`, `"ggplot"`, `"plotly"`
`DrawTopView`	Boolean, True means contur is drawn, otherwise a 3D plot is drawn. Default: TRUE
`xlab`	String, title of the x axis. Default: "X", see `plot()` function
`ylab`	String, title of the y axis. Default: "Y", see `plot()` function
`main`	string, the same as "main" in `plot()` function
`xlim`	see `plot()` function
`ylim`	see `plot()` function
`Legendlab_ggplot`	String, in case of `Plotter="ggplot"` label for the legend. Default: "value"
`...`	Density specifc parameters, for `PDEscatter()` or SDH (nbins,lambda,Xkernels,Ykernel))

Details

Example shows that features with high correlation coefficient do not correlate because of bimodality.

Value

plotting handler

Author(s)

Michael Thrun

References

[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.

See Also

`DensityScatter`

Examples

```
data(ITS)
data(MTY)
Inds=which(ITS<900&MTY<8000)

InspectCorrelation(ITS[Inds],MTY[Inds])
```

`InspectDistances` *Inspection of Distance-Distribution*

Description

Visualizes the distances between objects in the data matrix

Usage

```
InspectDistances(DataOrDistances,method= "euclidean",sampleSize = 50000,...)
```

Arguments

`DataOrDistances`
: [1:n,1:d] data cases in rows, variables in columns, if not symmetric
  or
  [1:n,1:n] distance matrix, if symmetric

`method`
: Optional, if Data[1:n,1:d] see `parallelDist::parDist` for distance method

`sampleSize`
: double value defining the size of the sample for large distance matrizes, see `InspectVariable`

`...`
: further arguments passed on to `InspectVariable`

Details

For an interpretation of the distribution analysis of the distance please read [Thrun, 2018, p. 27, 185].

Note

uses `InspectVariable`

Author(s)

Michael Thrun

References

[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.

Examples

```
data("Lsun3D")
Data=Lsun3D$Data

InspectDistances(as.matrix(dist(Data)))
```

`InspectScatterplots` *Pairwise scatterplots and optimal histograms*

Description

Pairwise scatterplots and optimal histograms of all features stored as columns of data are plotted

Usage

```
InspectScatterplots(Data,Names=colnames(Data))
```

Arguments

`Data`	[1:n,1:d] Data cases in rows (n), variables in columns (d)
`Names`	Optional: Names of the variables. If missing the columnnames of data are used.

Details

For two features, `PDEscatter` function should be used to isnpect modalities [Thrun/Ultsch, 2018]. For many features the function takes too lang. In such a case this function can be used. See [Thrun/Ultsch, 2018] for optimal histogram description.

Author(s)

Michael Thrun

References

[Thrun/Ultsch, 2018] Thrun, M. C., & Ultsch, A.: Effects of the payout system of income taxes to municipalities in Germany, 12th Professor Aleksander Zelias International Conference on Modelling and Forecasting of Socio-Economic Phenomena, Vol. accepted, Foundation of the Cracow University of Economics, Zakopane, Poland, 2018.

Examples

```
Data=cbind(rnorm(100, mean = 2, sd = 3 ),rnorm(100,mean = 0, sd = 1),rnorm(100,mean = 6, sd = 0.5))
#InspectScatterplots(Data)
```

`InspectStandardization`

QQplot of Data versus Normalized Data

Description

Allows to inspect if standardization of data makes sense

Usage

```
InspectStandardization(Data, TransData, xug = -3, xog = 3, xlab = "Normal", yDataLab =

"Data", yTransDataLab = "Trasformated Data", Symbol4Gerade = "red", main = "", ...)
```

Arguments

`Data`	...
`TransData`	...
`xug`	...
`xog`	...

`xlab`	...
`yDataLab`	...
`yTransDataLab`	...
`Symbol4Gerade`	...
`main`	...
`...`	...

Details

...

Value

plot

Author(s)

Michael Thrun

References

Michael, J. R.: The stabilized probability plot, Biometrika, Vol. 70(1), pp. 11-17, 1983.

`InspectVariable` *Visualization of Distribution of one variable*

Description

Enables distribution inspection by visualization as described in [Thrun, 2018] and for example used in

Usage

```
InspectVariable(Feature, Name, i = 1, xlim, ylim,

 sampleSize =1e+05, main)
```

Arguments

`Feature`	[1:n] Variable/Vector of Data to be plotted
`Name`	Optional, string, for x label
`i`	Optional, No. of variable/feature, an integer of the for lope
`xlim`	[2] Optional, range of x-axis for PDEplot
`ylim`	[2] Optional, range of y-axis for PDEplot
`sampleSize`	Optional, default(100000), sample size, if datavector is to big
`main`	string for the title if other than what is desribed in `N`

Author(s)

Michael Thrun

References

[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.

[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.

Examples

```
data("ITS")
InspectVariable(ITS,Name='Income in EUR',main='ITS')
```

ITS *Income Tax Share*

Description

Numerical vector of length 11194. details in [Ultsch/Behnisch, 2017; Thrun/Ultsch, 2018].

Usage

```
data("ITS")
```

References

[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.

[Ultsch/Behnisch, 2017] Ultsch, A., Behnisch, M.: Effects of the payout system of income taxes to municipalities in Germany, Applied Geography, Vol. 81, pp. 21-31, 2017.

Examples

```
data(ITS)
str(ITS)
```

`JitterUniqueValues` *Jitters Unique Values*

Description

Jitters Unique Values for Visualizations

Usage

```
JitterUniqueValues(Data, Npoints = 20,

min = 0.99999, max = 1.00001)
```

Arguments

`Data`	[1:n] vector of data
`Npoints`	number of jittered points generated from the m unique values of the datavector Data
`min`	minimum value of jittering
`max`	maximum value of jittering

Details

min and max are either multiplied or added to data depending on the range of values. If Npoints==2, then only two values per unique of Data is jittered otherwise additional values are generated.Npoints==1 does not jitter the values but gives the unique values back.

Value

vector of DataJitter[1:(m+Npoints-1)] jittered values

Author(s)

Michael Thrun

See Also

used for example in `MDplot`

Examples

```
data=c(rep(1,10),rep(0,10),rep(100,10))

JitterUniqueValues(data,Npoints=1)

JitterUniqueValues(data,Npoints=2)

DataJitter=JitterUniqueValues(data,Npoints=20)
```

Lsun3D *Lsun3D inspired by FCPS*

Description

clearly defined clusters, different variances

Usage

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

Details

Size n=404, Dimensions d=3

Dataset defined discontinuites, where the clusters have different variances. Three main Clusters, and four Outliers (in Cluster 4), see [Thrun, 2018]

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, ISBN: 978-3-658-20540-9, Heidelberg, 2018.

Examples

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

MAplot *Minus versus Add plot*

Description

Bland-Altman plot [Altman/Bland, 1983].

Usage

```
MAplot(X,Y,islog=TRUE,LoA=FALSE,CI=FALSE,

densityplot=FALSE,main,xlab,ylab,

Cls,lwd=2,ylim=NULL,...)
```

Arguments

`X`	[1:n] numerical vector of a feature/variable
`Y`	[1:n] another numerical vector of a feature/variable
`islog`	Optional, TRUE: MAplot, FALSE: M=x-y versus a=0.5(x+y)
`LoA`	Optional, if TRUE: limits of agreement are plottet as lines if densityplot=FALSE
`CI`	Optional, if TRUE: confidence intervals for LoA, see [Stockl et al., 2004], if densityplot=FALSE
`densityplot`	Optional, FALSE: Scatterplot using `Classplot`, TRUE: density scatter plot with `DensityScatter`
`main`	Optional, see `plot`
`xlab`	Optional, see `plot`
`ylab`	Optional, see `plot`
`Cls`	Optional, prior Classification as a numeric vector.
`lwd`	Optional, if `LoA=TRUE` or `CI=TRUE` the width of the lines, otherwise input argument is ignored
`ylim`	Optional, default `=NULL` sets this parameter automatically, otherwise see `Classplot`.
`...`	for example, `ylim`, Please see either `Classplot` in the mode `Plotter="native"`, or `DensityScatter` for further arguments depending on `densityplot`, see also details

Details

Bland-Altman plot [Altman/Bland, 1983] for visual representation of genomic data or in order to decorrelate data.

"The limits of agreement (LoA) are defined as the mean difference +- 1.96 SD of differences. If these limits do not exceed the maximum allowed difference between methods (the differences within mean +- 1.96 SD are not clinically important), the two methods are considered to be in agreement and may be used interchangeably." cited as in URL. Please note, that the underyling assumption is the normal distribution of the differences. Input argument `LoA=TRUE` shows the mean of the difference in blue and +- 1.96 SD in green. Input argument `CI=TRUE` shows the mean of the difference in blue and the confidence intervall as red dashed lines similar to the cited URL.

In case of `densityplot=FALSE`, the function `Classplot` is always called with `Plotter="native"`. Then, the input argument `"Colors""` of points can only be set in `Classplot` if `"Cls""` is given in this function, otherwise the points are always black. The input argument `"Size""` sets the size of points in `Classplot`.

Value

`MA`	[1:n,2] Matrix of Minus component of two features and Add component of two features
`Handle`	see `DensityScatter` for output options, if densityplot=TRUE, otherwise NULL
`Statistics`	Named list of four element, each consisting of one value depending on input parameters `LoA` and `CI`, of this function. If not specifically set each list element is `NULL`. The elements are `Mean_value` - mean allowed difference, `SD_value` - standard deviation of difference, `LoA_value` - Limits of agreement=1.96*SD, `CI_value` - confidence intervall, i.e., maximum allowed difference

Author(s)

Michael Thrun

References

[Altman/Bland, 1983] Altman D.G., Bland J.M.: Measurement in medicine: the analysis of method comparison studies, The Statistician, Vol. 32, p. 307-317, doi:10.2307/2987937, 1983.

https://www.medcalc.org/manual/bland-altman-plot.php

[Stockl et al., 2004] Stockl, D., Rodriguez Cabaleiro, D., Van Uytfanghe, K., & Thienpont, L. M.: Interpreting method comparison studies by use of the Bland-Altman plot: reflecting the importance of sample size by incorporating confidence limits and predefined error limits in the graphic, Clinical chemistry, Vol. 50(11), pp. 2216-2218. 2004.

Examples

```
data("ITS")
data("MTY")
MAlist=MAplot(ITS,MTY)
```

MDplot *Mirrored Density plot (MD-plot)*

Description

This function creates a MD-plot for each variable of the data matrix. The MD-plot is a visualization for a boxplot-like shape of the PDF published in [Thrun et al., 2020] with the default ordering by shape. It is an improvement of violin or so-called bean plots and posses advantages in comparison to the conventional well-known box plot [Thrun et al., 2020].

A complete guide about the MDplot can be found in https://md-plot.readthedocs.io/en/latest/index.html.

Usage

```
MDplot(Data, Names, Ordering='Default', Scaling="None",

Fill='darkblue', RobustGaussian=TRUE, GaussianColor='magenta',

Gaussian_lwd=1.5, BoxPlot=FALSE,BoxColor='darkred',

MDscaling='width', LineColor='black', LineSize=0.01,

QuantityThreshold=50, UniqueValuesThreshold=12,

SampleSize=5e+05,SizeOfJitteredPoints=1,OnlyPlotOutput=TRUE,

main="MD-plot",ylab="Range of values in which PDE is estimated",

BW=FALSE,ForceNames=FALSE)
```

Arguments

- `Data` — [1:n,1:d] Numerical Matrix containing the n cases of d variables. Each column is one variable. A data.frame is automatically transformed to a numerical matrix.
- `Names` — Optional: [1:d] Names of the variables. If missing, the columnnames of data are used. If not missing, than the names can be cleaned or not (see `ForceNames`).
- `Ordering` — Optional: string, either `Default`, `Columnwise` or `AsIs`, `Alphabetical`, `Average`, `Bimodal`, `Variance` or `Statistics`. Please see details for explanation.
- `Scaling` — Optional, Default is `None`, `Percentalize`, `CompleteRobust`, `Robust` or `Log`, Please see details for explanation.
- `Fill` — Optional: string, color with which MDs are to be filled with.
- `RobustGaussian` — Optional: If TRUE: each MDplot of a variable is overlayed with a roubustly estimated unimodal Gaussian distribution in the range of this variable, if statistical testing does not yield a significant p.value. In this case the packages **moments**, **diptest** and **signal** are required.
- `GaussianColor` — Optional: string, color of robustly estimated gaussian, only for `RobustGaussian=TRUE`.
- `Gaussian_lwd` — Optional: numerical, line width of robustly estimated gaussian, only for `RobustGaussian=TRUE`.
- `BoxPlot` — Optional: If TRUE: each MDplot is overlayed with a Box-Whisker Diagram.
- `BoxColor` — Optional: string, color of Boxplot, only for `BoxPlot=TRUE`.
- `MDscaling` — Optional: if "area", all violins have the same area (before trimming the tails). If "count", areas are scaled proportionally to the number of observations. If "width" (default), all MDs have the same maximum width.
- `LineColor` — Optional: string, color of line around the mirrored densities. `NA` disables this features which is usefull if ones wants to avoid vertical lines leading to outliers.
- `LineSize` — Optional: numerical, linewidth of line around the mirrored densities.
- `QuantityThreshold` — Optional: numeric value defining the threshold of the minimal amount of values in data. Below this threshold no density estimation is performed and a 1D scatter plot with jittered points is drawn. Only Data Science experts should change this value after they understand how the density is estimated (see [Ultsch, 2005]).
- `UniqueValuesThreshold` — Optional: numeric value defining the threshold of the minimal amount of unique values in data. Below this threshold no density estimation and statistical testing is performed and a 1D scatter plot with jittered points drawn. Only Data Science experts should change this value after they understand how the density is estimated (see [Ultsch, 2005]).
- `SampleSize` — Optional: numeric value defining a threshold. Above this threshold uniform sampling of finite cases is performed in order to shorten computation time.If **rowr** is not installed, uniform sampling of all cases is performed. If required, `SampleSize=n` can be set to omit this procedure.
- `SizeOfJitteredPoints` — Optional: scalar. If not enough unique values for density estimation are given, data points are jittered. This parameter defines the size of the points.
- `OnlyPlotOutput` — Optional: Default TRUE only a ggplot object is given back, if FALSE: Additinally, scaled data and ordering are the output of this function in a `list`.

`main`	string defining the (centered) title of the plot
`ylab`	string defining the y label, PDE= pareto density estimation (see [Ultsch, 2005])
`BW`	FALSE: usual ggplot2 background and style which is good for screen visualizations TRUE: theme_bw() is used which is more appropriate for publications
`ForceNames`	FALSE: Per Default column names are cleaned for propper plotting TRUE: forces to set the column names as given. Beware, this can result in plotting errors.

Details

In short, the MD-plot can be described as a PDE optimized violin plot. The Pareto Density Estimation (PDE) is an approach to estimate the probability density function (pdf) [Ultsch, 2005].

The MD-plot is in the process of beeing peer-reviewed [Thrun/Ultsch, 2019].

Statistical testing is performed with `dip.test` and `agostino.test`.

For the paramter `Ordering` the following options are possible:

`Default` Ordering of plots by convex/concav/unimodal/nonunimodal shapes using statistical criteria. In this case the **signal** is required.

`Columnwise` Ordering of plots by the order of columns of `Data`.

`AsIs` Synonym of Columnwise: Ordering of plots by the order of columns of `Data`.

`Alphabetical` Ordering of plots by the order of columns of `Data` sorted in alphabetical order by column names.

`Average` Ordering of plots by the order of columns of `Data` sorted in order of increasing columnwise average

`Bimodal` Ordering of plots by the order of columns of `Data` sorted in order of decreasing bimodality amplitude[Zhang et al., 2003]

`Variance` Ordering of plots by the order of columns of `Data` sorted in order of increasing interquartile range

`Statistics` Ordering of plots depending on the logarithm of the p-vlaues of statistical testing. In this case the packages **moments**, **diptest** and **signal** are required.

For the paramter `Scaling` the following options are possible:

`None` No Scaling of data is done.

`Percentalize` Data is scaled between zero and 100.

`CompleteRobust` Data is first robustly scaled between zero and 1, then centered to zero and outliers are capped by a robustly formula described in the **DatabionicSwarm** package.

`Robust` Data is robustly scaled between zero and 1 by a formula described in the **DatabionicSwarm** package.

`Log` Data is transformed with a sgined log allowing for negative values to be transformed with a logarithm of base 10, please see `SignedLog` for details.

Value

In the default case of `OnlyPlotOutput==TRUE`: The ggplot object of the MD-plot.

Otherwise for `OnlyPlotOutput==FALSE`: A list of

`ggplotObj`	The ggplot object of the MD-plot.
`Ordering`	The ordering of columns of data defined by `Ordering`.
`DataOrdered`	[1:n,1:d] matrix of ordered and scaled data defined by `Ordering` and `Scaling`.

Note that the package **ggExtra** is not necessarily required but if given the feature names are automatically rotated.

Note

1.) One would assume that in the first of the two following cases ggplot2 only adjusts the plotting region but:

`MDplot(MTY)+ylim(c(0,7000))` is equal to `MDplot(MTY[MTY<7000])`.

This means in both cases the data is clipped and AFTERWARDS the density estimation is performed.

2.) Because of a (sometimes) strange behavior of either ggplot2 or reshape2, numerical column names are changed to character by adding 'C_' which can disabled using `ForceNames=TRUE`.

3.) Columnnames will be automatically deblanked and cleaned. To force specific columnnames the input `Names` can be used in combination with `ForceNames=TRUE`. However, this can result in plotting errors or other strange behavior.

4.) Overlaying MD-plots with robustly estimated gaussians seldomly will yield magenta (or other `GaussianColor`) lines overlaying more than the violin plot they should overlay, because the width of the two plots is not the same (but I am unable to set it strictly in ggplot). In such a case just call the function again.

Author(s)

Michael Thrun, Felix Pape contributed with the idea to use ggplot2 as the basic framework.

References

[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 10.1371/journal.pone.0238835, 2020.

[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.

[Zhang et al., 2003] Zhang, C., Mapes, B., & Soden, B.: Bimodality in tropical water vapour, Quarterly Journalof the Royal Meteorological Society, 129(594), 2847-2866, 2003.

See Also

https://md-plot.readthedocs.io/en/latest/index.html

`ClassMDplot`

https://pypi.org/project/md-plot/

Examples

```
x = cbind(
    A = runif(20000, 1, 5),
    B = c(rnorm(10000, 0, 1), rnorm(10000, 2.6, 1)),
    C = c(rnorm(20000, 2.5, 1)),
    D = rpois(20000, 5)
  )
MDplot(x)
```

`MDplot4multiplevectors`

Mirrored Density plot (MD-plot)for Multiple Vectors

Description

This function creates a MD-plot for multiple numerical vectors of various lenghts. The MD-plot is a visualization for a boxplot-like Shape of the PDF published in [Thrun et al., 2020]. It is an improvement of violin or so-called bean plots and posses advantages in comparison to the conventional well-known box plot [Thrun et al., 2020].

Usage

```
MDplot4multiplevectors(..., Names, Ordering = 'Default',
Scaling = "None", Fill = 'darkblue', RobustGaussian = TRUE,

GaussianColor = 'magenta', Gaussian_lwd = 1.5, BoxPlot = FALSE,

BoxColor = 'darkred', MDscaling = 'width', LineSize = 0.01,

LineColor = 'black', QuantityThreshold = 40, UniqueValuesThreshold = 12,

SampleSize = 5e+05, SizeOfJitteredPoints = 1, OnlyPlotOutput = TRUE)
```

Arguments

`...`	Either d numerical vectors of different lengths or a list of length d where each element of the list is an vector of arbitrary length
`Names`	Optional: [1:d] Names of the variables. If missing, the columnnames of data are used.
`Ordering`	Optional: string, either `Default`, `Columnwise`, `Alphabetical` or `Statistics`. Please see details for explanation.
`Scaling`	Optional, Default is `None`, `Percentalize`, `CompleteRobust`, `Robust` or `Log`, Please see details for explanation.

`Fill`	Optional: string, color with which MDs are to be filled with.
`RobustGaussian`	Optional: If TRUE: each MDplot of a variable is overlayed with a roubustly estimated unimodal Gaussian distribution in the range of this variable, if statistical testing does not yield a significant p.value. In this case the packages **moments**, **diptest** and **signal** are required.
`GaussianColor`	Optional: string, color of robustly estimated gaussian, only for `RobustGaussian=TRUE`.
`Gaussian_lwd`	Optional: numerical, line width of robustly estimated gaussian, only for `RobustGaussian=TRUE`.
`BoxPlot`	Optional: If TRUE: each MDplot is overlayed with a Box-Whisker Diagram.
`BoxColor`	Optional: string, color of Boxplot, only for `BoxPlot=TRUE`.
`MDscaling`	Optional: if "area", all violins have the same area (before trimming the tails). If "count", areas are scaled proportionally to the number of observations. If "width" (default), all MDs have the same maximum width.
`LineSize`	Optional: numerical, linewidth of line around the mirrored densities.
`LineColor`	Optional: string, color of line around the mirrored densities. `NA` disables this features which is usefull if ones wants to avoid vertical lines leading to outliers.
`QuantityThreshold`	Optional: numeric value defining a threshold. Below this threshold no density estimation is performed and a jitter plot with a median line is drawn. Only Data Science experts should change this value after they understand how the density is estimated (see [Ultsch, 2005]).
`UniqueValuesThreshold`	Optional: numeric value defining a threshold. Below this threshold no density estimation and statistical testing is performed and a Jitter plot is drawn. Only Data Science experts should change this value after they understand how the density is estimated (see [Ultsch, 2005]).
`SampleSize`	Optional: numeric value defining a threshold. Above this threshold uniform sampling of finite cases is performed in order to shorten computation time.If **rowr** is not installed, uniform sampling of all cases is performed. If required, `SampleSize=n` can be set to omit this procedure.
`SizeOfJitteredPoints`	Optional: scalar. If Not enough unique values for density estimation are given, data points are jittered. This parameter defines the size of the points.
`OnlyPlotOutput`	Optional: Default TRUE only a ggplot object is given back, if FALSE: Additinally Scaled Data and ordering are the output of this function in a `list`.

Details

Please see `MDplot` for details.

Value

In the default case of `OnlyPlotOutput==TRUE`: The ggplot object of the MD-plot.

Otherwise for `OnlyPlotOutput==FALSE`: A list of

`ggplotObj`	The ggplot object of the MD-plot.

`Ordering` The ordering of columns of data defined by `Ordering`.

`DataOrdered` [1:n,1:d] matrix of ordered and scaled data defined by `Ordering` and `Scaling`.

Note that the package **ggExtra** is not necessarily required but if given the feauture names are automatically rotated.

Note

cbind.fill is internally used from the depricated R package rowr of Craig Varrichio.

Author(s)

Michael Thrun.

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 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 10.1371/journal.pone.0238835, 2020.

See Also

`ClassMDplot` `MDplot` `https://pypi.org/project/md-plot/`

Examples

```
MDplot4multiplevectors(runif(20000, 1, 5),c(rnorm(20000,0,1),

rnorm(20000,2.6,1)),c(rnorm(2000,2.5,1)),rpois(25000,5),

Names=c('A','B','C','D'))

V=list(runif(20000, 1, 5),c(rnorm(20000,0,1),

rnorm(20000,2.6,1)),c(rnorm(2000,2.5,1)),rpois(25000,5))


MDplot4multiplevectors(V,Names=c('A','B','C','D'))
```

MTY *Muncipal Income Tax Yield*

Description

Numerical vector of length 11194. details in [Ultsch/Behnisch, 2017; Thrun/Ultsch, 2018].

Usage

```
data("MTY")
```

References

[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.

[Ultsch/Behnisch, 2017] Ultsch, A., Behnisch, M.: Effects of the payout system of income taxes to municipalities in Germany, Applied Geography, Vol. 81, pp. 21-31, 2017.

Examples

```
data(MTY)
str(MTY)
```

OptimalNoBins *Optimal Number Of Bins*

Description

Optimal Number Of Bins is a kernel density estimation for fixed intervals.

Calculation of the optimal number of bins for a histogram.

Usage

```
OptimalNoBins(Data)
```

Arguments

`Data`	Data

Details

The bin width ist defined with bw=3.49*stdrobust(1/(n)^1/3)

Value

optNrOfBins The best possible number of bins. Not less than 10 though

Note

This the second version of the function prior available in **AdaptGauss**

Author(s)

Alfred Ultsch, Michael Thrun

References

David W. Scott Jerome P. Keating: A Primer on Density Estimation for the Great Home Run Race of 98, STATS 25, 1999, pp 16-22.

See Also

ParetoRadius

Examples

```
Data = c(rnorm(1000),rnorm(2000)+2,rnorm(1000)*2-1)

optNrOfBins = OptimalNoBins(Data)

minData = min(Data,na.rm = TRUE)

maxData = max(Data,na.rm = TRUE)

i = maxData-minData

optBreaks = seq(minData, maxData, i/optNrOfBins) # bins in fixed intervals

hist(Data, breaks=optBreaks)
```

`ParetoDensityEstimation` *Pareto Density Estimation V3*

Description

This function estimates the Pareto Density for the distribution of one variable. In the default setting the functions estimates internally the appropriate number and position of kernels to estimate the density properly. However, the user can set the `kernels` manually. In this case density will only be estimated only around these values even if data exists outside the range of kernels or the internally estimated `paretoRadius` does not contain all datapoints between each kernel. See example for details.

Usage

```
ParetoDensityEstimation(Data, paretoRadius, kernels = NULL,
  MinAnzKernels = 100,PlotIt=FALSE,Silent=FALSE)
```

Arguments

`Data`	[1:n] numeric vector of data.
`paretoRadius`	Optional scalar, numeric value, see `ParetoRadius`.If not given it is estimated internally. Please do not set manually
`kernels`	Optional,[1:m] numeric vector data values where pareto density is measured at. If 0 (by default) kernels will be computed.
`MinAnzKernels`	Optional, minimal number of kernels, default MinAnzKernels==100
`PlotIt`	Optional, if TRUE: raw basic r plot of density estimation of debugging purposes. Usually please use **ggplot2** interface via `PDEplot` or `MDplot`
`Silent`	Optional, if TRUE: disables all warnings

Details

Pareto Density Estimation (PDE) is a method for the estimation of probability density functions using hyperspheres. The Pareto-radius of the hyperspheres is derived from the optimization of information for minimal set size. It is shown, that Pareto Density is the best estimate for clusters of Gaussian structure. The method is shown to be robust when cluster overlap and when the variances differ across clusters. This is the best density estimation to judge Gaussian Mixtures of the data see [Ultsch 2003].

If input argument `kernels` is set manually the output arguments `paretoDensity_internal` and `kernels_internal` provide the internally estimated density and kernels. Otherwise these arguments are NULL. The function provides a message if range of kernels and range of data does not overlap completly.

Typically it is not advisable to set `paretoRadius` manually. However in specific cases, the function `ParetoRadius` is used prior to calling this function. In such cases the input argument can use a priorly estimated `paretoRadius`.

Value

List With

kernels [1:m] numeric vector. data values at with Pareto Density is measured.

paretoDensity [1:m] numeric vector containing the determined density by `paretoRadius`.

paretoRadius numeric value of defining the radius

kernels_internal Either NULL or internally estimated [1:p] numeric vector of kernels if input argument `kernels` was set by the user

paretoDensity_internal Either NULL or internally estimated density if input argument `kernels` was set by the user

Note

This the second version of the function prior available in **AdaptGauss**

Author(s)

Michael Thrun

References

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.

See Also

ParetoRadius

PDEplot

MDplot

Examples

```
#kernels are estimated internally
data = c(rnorm(1000),rnorm(2000)+2,rnorm(1000)*2-1)
pdeVal        <- ParetoDensityEstimation(data)
plot(pdeVal$kernels,pdeVal$paretoDensity,type='l',xaxs='i',
yaxs='i',xlab='Data',ylab='PDE')

##data exist outside of the range kernels
kernels=seq(from=-3,to=3,by=0.01)
pdeVal        <- ParetoDensityEstimation(data,  kernels=kernels)
plot(pdeVal$kernels,pdeVal$paretoDensity,type='l',xaxs='i',
yaxs='i',xlab='Data',ylab='PDE')

#data exists in-between kernels that is not measured
pdeVal$paretoRadius#0.42
kernels=seq(from=-8,to=8,by=1)
pdeVal        <- ParetoDensityEstimation(data,  kernels=kernels)
plot(pdeVal$kernels,pdeVal$paretoDensity,type='l',xaxs='i',
yaxs='i',xlab='Data',ylab='PDE')
```

ParetoRadius *ParetoRadius for distributions*

Description

Calculation of the ParetoRadius i.e. the 18 percentiles of all mutual Euclidian distances in data.

Usage

```
ParetoRadius(Data, maximumNrSamples = 10000,
  plotDistancePercentiles = FALSE)
```

Arguments

`Data` numeric data vector

`maximumNrSamples`
Optional, numeric. Maximum number for which the distance calculation can be done. 1000 by default.

`plotDistancePercentiles`
Optional, logical. If TRUE, a plot of the percentiles of distances is produced. FALSE by default.

Details

The Pareto-radius of the hyperspheres is derived from the optimization of information for minimal set size. ParetoRadius() is a kernel density estimation for variable intervals. It works only on Data without missing values (NA) or NaN. In other cases, please use ParetoDensityEstimation directly.

Value

numeric value, the Pareto radius.

Note

This the second version of the function prior available in **AdaptGauss**.

For larger datasets the quantile_c() function is used instead of quantile in R which was programmed by Dirk Eddelbuettel on Jun 6 and taken by the author from `https://github.com/RcppCore/Rcpp/issues/967`.

Author(s)

Michael Thrun

References

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.

See Also

ParetoDensityEstimation, OptimalNoBins

PDEplot *PDE plot*

Description

This function plots the Pareto probability density estimation (PDE), uses PDEstimationForGauss and ParetoRadius.

Usage

```
PDEplot(Data, paretoRadius = 0, weight = 1, kernels = NULL,

                 LogPlot = F, PlotIt = TRUE, title =
                 "ParetoDensityEstimation(PDE)", color = "blue",

                 xpoints = FALSE, xlim, ylim, xlab, ylab =
                 "PDE", ggPlot = ggplot(), sampleSize = 2e+05, lwd = 2)
```

Arguments

`Data`	[1:n] numeric vector of data to be plotted.
`paretoRadius`	numeric, the Pareto Radius. If omitted, calculate by paretoRad.
`weight`	numeric, Weight*ParetoDensity is plotted. 1 by default.
`kernels`	numeric vector of kernels. Optional
`LogPlot`	LogLog PDEplot if TRUE, xpoints has to be FALSE. Optional
`PlotIt`	logical, if plot. TRUE by default.
`title`	character vector, title of plot.
`color`	character vector, color of plot.
`xpoints`	logical, if TRUE only points are plotted. FALSE by default.
`xlim`	Arguments to be passed to the plot method.
`ylim`	Arguments to be passed to the plot method.
`xlab`	Arguments to be passed to the plot method.
`ylab`	Arguments to be passed to the plot method.
`ggPlot`	ggplot2 object to be plotted upon. Insert an exisiting plot to add a new PDEPlot to it. Default: empty plot
`sampleSize`	default(200000), sample size, if datavector is to big
`lwd`	linewidth, see `plot`

Value

`kernels`	numeric vector. The x points of the PDE function.
`paretoDensity`	numeric vector, the PDE(x).
`paretoRadius`	numeric value, the Pareto Radius used for the plot.
`ggPlot`	ggplot2 object. Can be used to further modify the plot or add other plots.

Author(s)

Michael Thrun

References

Ultsch, A.: Pareto Density Estimation: A Density Estimation for Knowledge Discovery, Baier D., Wernecke K.D. (Eds), In Innovations in Classification, Data Science, and Information Systems - Proceedings 27th Annual Conference of the German Classification Society (GfKL) 2003, Berlin, Heidelberg, Springer, pp, 91-100, 2005.

Examples

```
x <- rnorm(1000, mean = 0.5, sd = 0.5)
y <- rnorm(750, mean = -0.5, sd = 0.75)
plt <- PDEplot(x, color = "red")$ggPlot
plt <- PDEplot(y, color = "blue", ggPlot = plt)$ggPlot

# Second Example
#  ggplotObj=ggplot()
#  for(i in 1:length(Variables))
#     ggplotObj=PDEplot(Data[,i],ggPlot = ggplotObj)$ggPlot
```

`PDEscatter` *Scatter Density Plot*

Description

Pareto density estimation (PDE) [Ultsch, 2005] used for a scatter density plot.

Usage

```
PDEscatter(x,y,SampleSize,

na.rm=FALSE,PlotIt=TRUE,ParetoRadius,sampleParetoRadius,

NrOfContourLines=20,Plotter='native', DrawTopView = TRUE,

xlab="X", ylab="Y", main="PDEscatter",

xlim, ylim, Legendlab_ggplot="value")
```

Arguments

x	Numeric vector [1:n], first feature (for x axis values)
y	Numeric vector [1:n], second feature (for y axis values)
SampleSize	Numeric m, positiv scalar, maximum size of the sample used for calculation. High values increase runtime significantly. The default is that no sample is drawn
na.rm	Function may not work with non finite values. If these cases should be automatically removed, set parameter TRUE
ParetoRadius	Numeric, positiv scalar, the Pareto Radius. If omitted (or 0), calculate by paretoRad.
sampleParetoRadius	Numeric, positiv scalar, maximum size of the sample used for estimation of "kernel", should be significantly lower than SampleSize because requires distance computations which is memory expensive

PlotIt	`TRUE`: plots with function call
	`FALSE`: Does not plot, plotting can be done using the list element `Handle`
	`-1`: Computes density only, does not perfom any preperation for plotting meaning that `Handle=NULL`

NrOfContourLines	Numeric, number of contour lines to be drawn. 20 by default.
Plotter	String, name of the plotting backend to use. Possible values are: `"native"`, `"ggplot"`, `"plotly"`
DrawTopView	Boolean, True means contur is drawn, otherwise a 3D plot is drawn. Default: TRUE
xlab	String, title of the x axis. Default: "X", see `plot()` function
ylab	String, title of the y axis. Default: "Y", see `plot()` function
main	string, the same as "main" in `plot()` function
xlim	see `plot()` function
ylim	see `plot()` function
Legendlab_ggplot	String, in case of `Plotter="ggplot"` label for the legend. Default: "value"

Details

The `PDEscatter` function generates the density of the xy data as a z coordinate. Afterwards xyz will be plotted either as a contour plot or a 3d plot. It assumens that the cases of x and y are mapped to each other meaning that a `cbind(x,y)` operation is allowed. This function plots the PDE on top of a scatterplot. Variances of x and y should not differ by extreme numbers, otherwise calculate the percentiles on both first. If `DrawTopView=FALSE` only the plotly option is currently available. If another option is chosen, the method switches automatically there.

The method was succesfully used in [Thrun, 2018; Thrun/Ultsch 2018].

`PlotIt=FALSE` is usefull if one likes to perform adjustements like axis scaling prior to plotting with **ggplot2** or **plotly**. In the case of `"native"`" the handle returns `NULL` because the basic R functon `plot()` is used

Value

List of:

X	Numeric vector [1:m],m<=n, first feature used in the plot or the kernels used
Y	Numeric vector [1:m],m<=n, second feature used in the plot or the kernels used
Densities	Numeric vector [1:m],m<=n, Number of points within the ParetoRadius of each point, i.e. density information
Matrix3D	1:n,1:3] marix of x,y and density information
ParetoRadius	ParetoRadius used for PDEscatter
Handle	Handle of the plot object. Information-string if native R plot is used.

Note

MT contributed with several adjustments

Author(s)

Felix Pape

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, (Ultsch, A. & Huellermeier, E. Eds., 10.1007/978-3-658-20540-9), Doctoral dissertation, Heidelberg, Springer, ISBN: 978-3658205393, 2018.

[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.

[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

```
#taken from [Thrun/Ultsch, 2018]
data("ITS")
data("MTY")
Inds=which(ITS<900&MTY<8000)
plot(ITS[Inds],MTY[Inds],main='Bimodality is not visible in normal scatter plot')

PDEscatter(ITS[Inds],MTY[Inds],xlab = 'ITS in EUR',

ylab ='MTY in EUR' ,main='Pareto Density Estimation indicates Bimodality' )
```

Piechart *The pie chart*

Description

the pie chart represents amount of values given in data.

Usage

```
Piechart(Datavector,Names,Labels,MaxNumberOfSlices,
main='',col,Rline=1,...)
```

Arguments

`Datavector`	[1:n] a vector of n non unique values
`Names`	Optional, [1:k] names to search for in Datavector, if not set `unique` of Datavector is calculated.
`Labels`	Optional, [1:k] Labels if they are specially named, if not Names are used.
`MaxNumberOfSlices`	Default is k, integer value defining how many labels will be shown. Everything else will be summed up to `Other`.
`main`	Optional, title below the fan pie, see `plot`
`col`	Optional, the default are the first [1:k] colors of the default color sequence used in this package, otherwise a character vector of [1:k] specifying the colors analog to `plot`
`Rline`	Optional, the radius of the pie in numerical numbers
`...`	Optional, further arguments passed on to `plot`

Details

If Number of Slices is higher than MaxNumberOfSlices then `ABCanalysis` is applied (see [Ultsch/Lotsch, 2015]) and group A chosen. If Number of Slices in group A is higher than MaxNumberOfSlices, then the most important ones out of group A are chosen. If MaxNumberOfSlices is higher than Slices in group A, additional slices are shown depending on the percentage (from high to low). Parameters of visualization a set as in [Schwabish, 2014] defined.

Color sequence is automatically shortened to the MaxNumberOfSlices used in the pie chart.

Value

silent output by calling `invisible` of a list with

`Percentages`	[1:k] percent values visualized in fanplot
`Labels`	[1:k] see input `Labels`, only relevant ones

Note

You see in the example below that a pie chart does not visualize such data well contrary to the `fanPlot`.

Author(s)

Michael Thrun

References

[Schwabish, 2014] Schwabish, Jonathan A. An Economist's Guide to Visualizing Data. Journal of Economic Perspectives, 28 (1): 209-34. DOI: 10.1257/jep.28.1.209, 2014.

[Ultsch/Lotsch, 2015] Ultsch. A ., Lotsch J.: Computed ABC Analysis for Rational Selection of Most Informative Variables in Multivariate Data, PloS one, Vol. 10(6), pp. e0129767. doi 10.1371/journal.pone.0129767, 2015.

Examples

```
data(categoricalVariable)
Piechart(categoricalVariable)
```

`Pixelmatrix` *Plot of a Pixel Matrix*

Description

Plots Data matrix as a pixel coulour image.

Usage

```
Pixelmatrix(Data, XNames, LowLim, HiLim,

YNames, main,FillNotFiniteWithHighestValue=FALSE)
```

Arguments

`Data`	[1:n,1:d] Data cases in rows (n), variables in columns (d)
`LowLim`	Optional: limits for the color axis
`HiLim`	Optional: limits for the color axis
`XNames`	Optional: Vector - names for the X-ticks, NULL: no ticks at all
`YNames`	Optional: Vector - names for the Y-ticks, NULL: no ticks at all
`main`	Optinal: String - Title of the plot
`FillNotFiniteWithHighestValue`	Optinal, Default FALSE = Non finite values are shown in black, TRUE=non finite values are transformed to a value higher than the highest value and shown in this color

Details

Low values are shown in blue and green, middle values in yellow and high values in orange and red.

Author(s)

Michael Thrun, Felix Pape

Examples

```
data("Lsun3D")
Data=Lsun3D$Data

Pixelmatrix(Data)
```

Plot3D *3D plot of points*

Description

A wrapper for Data with systematic clustering colors for either a 2D (x,y) or 3D (x,y,z) plot combined with a classification

Usage

```
Plot3D(Data,Cls,UniqueColors,

size=2,na.rm=FALSE,Plotter3D="rgl",...)
```

Arguments

`Data`	[1:n,1:d] matrix with either `d=2` or `d=3`, if `d>3` only the first 3 dimensions are taken
`Cls`	[1:n] numeric vector of the classification of data with `k` classes
`UniqueColors`	[1:k] character vector of colors, if not given DataVisualizations::DefaultColorSequence is used
`size`	size of points, for plotly additional a vector [1:n] of a mapping of sizes to Cls has to be given in the (...) argument with `sizes=`
`na.rm`	if `na.rm=TRUE`, then missing values are removed
`Plotter3D`	in case of 3 dimensions, choose either "plotly" or "rgl", if one of this packages is not given, the other one is selected as a fallback method
`...`	further arguments to be processed by `plot3d` or `geom_point` or `plot_ly` of type "scatter3d"

Details

For geom_point only size and na.rm is available as further arguments.

Note

Uses either geom_point for 2D or plot3d for 3D or plot_ly

Author(s)

Michael Thrun

References

RGL vignette in https://cran.r-project.org/package=rgl

Examples

```
#Spin3D similar output

data(Lsun3D)
Plot3D(Lsun3D$Data,Lsun3D$Cls,type='s',radius=0.1,box=FALSE,aspect=TRUE)
rgl::grid3d(c("x", "y", "z"))


#Projected Points with Classification
Data=cbind(runif(500,min=-3,max=3),rnorm(500))

# Classification
Cls=ifelse(Data[,1]>0,1,2)
Plot3D(Data,Cls,UniqueColors = DataVisualizations::DefaultColorSequence[c(1,3)],size=2)

## Not run:
#Points with Non-Overlapping Labels
#require(ggrepel)
Data=cbind(runif(30,min=-1,max=1),rnorm(30,0,0.5))
Names=paste0('VeryLongName',1:30)
ggobj=Plot3D(Data)
ggobj +  geom_text_repel(aes(label=Names), size=3)

## End(Not run)
```

PlotMissingvalues *Plot of the Amount Of Missing Values*

Description

Percentage of missing values per feature are visualized as a bar plot.

Usage

```
PlotMissingvalues(Data,Names,

WhichDefineMissing=c('NA','NaN','DUMMY','.',' '),

PlotIt=TRUE,

xlab='Amount Of Missing Values in Percent',

xlim=c(0,100),...)
```

Arguments

`Data`	[1:n,1:d] data cases in rows, variables/features in columns
`Names`	[1:d] optional vector of string describing the names of the features
`WhichDefineMissing`	[1:d] optional vector of string describing missing values, usefull for character features. Currently up to five different options are possible.
`PlotIt`	If FALES: Does not plot
`xlab`	x label of bar plot
`xlim`	x axis limits in percent
`...`	Further arguments passed on to `barplot`, such as `main` for title

Value

plots not finite and missing values as a bar plot for each feature d and returns with `invisible` the amount of missing values as a vector. Works even with character variables, but WhichDefineMissing cannot be changed at the current version. Please make a suggestion on GitHub how to improve this.

Note

Does not work with the tibble format, in such a case please call `as.data.frame(as.matrix(Data))`

Author(s)

Michael Thrun

Examples

```
data("ITS")
data("MTY")

PlotMissingvalues(cbind(ITS,MTY),Names=c('ITS','MTY'))
```

PlotProductratio *Product-Ratio Plot*

Description

The product-ratio plot as defined in [Tukey, 1977, p. 594].

Usage

```
PlotProductratio(X, Y, na.rm = FALSE,

main='Product Ratio Analysis',xlab = "Log of Ratio",ylab = "Root of Product", ...)
```

Arguments

`X`	[1:n] positive numerical vector, negativ values are removed automatically
`Y`	[1:n] positive numerical vector, negativ values are removed automatically
`na.rm`	Function may not work with non finite values. If these cases should be automatically removed, set parameter TRUE
`main`	see `plot`
`ylab`	see `plot`
`xlab`	see `plot`
`...`	further arguments passed on to `plot`

Details

In the case where there are many instances of very small values, but a small number of very large ones, this plot is usefull [Tukey, 1977, p. 615].

Value

matrix[1:n,2] with sqrt(x*y) and log(x/y) as the two columns

Author(s)

Michael Thrun

References

[Tukey, 1977] Tukey, J. W.: Exploratory data analysis, United States Addison-Wesley Publishing Company, ISBN: 0-201-07616-0, 1977.

Examples

```
#Beware: The data does no fit ne requirements for this approach
data('ITS')
data(MTY)
PlotProductratio(ITS,MTY)
```

`PmatrixColormap` *P-Matrix colors*

Description

Defines the default color sequence for plots made with PDEscatter

Usage

```
data("PmatrixColormap")
```

Format

Returns the vectors for a (heat) colormap.

`QQplot` *QQplot with a Linear Fit*

Description

Qantile-quantile plot with a linear fit

Usage

```
QQplot(X,Y,xlab, ylab,col="red",main='',...)
```

Arguments

`X`	[1:n] numerical vector, First Feature
`Y`	1:n] numerical vector, Second Feature to compare first feature with
`xlab`	x label, see `plot` ...
`ylab`	y label, see `plot`
`col`	color of line, see `plot`
`main`	title of plot, see `plot`
`...`	other parameters for `qqplot`

Details

Output is the evaluation of a linear fit of `lm` called 'line' and a quantile quantile plot (QQplot).

Value

List with

Residuals	Output of`residuals.lm(line)`
Summary	Output of`summaryline)`
Anova	Output of`anova(line)`

Author(s)

Michael Thrun

References

Michael, J. R.: The stabilized probability plot, Biometrika, Vol. 70(1), pp. 11-17, 1983.

`ShepardDensityscatter` *Shepard PDE scatter*

Description

Draws ein Shepard Diagram (scatterplot of distances) with an two-dimensional PDE density estimation .

Usage

```
ShepardDensityScatter(InputDists,OutputDists,

Plotter='native',xlab='Input Distances',

ylab='Output Distances',main='ProjectionMethod',

 sampleSize=500000)
```

Arguments

InputDists	[1:n,1:n] with n cases of data in d variables/features: Matrix containing the distances of the inputspace.
OutputDists	[1:n,1:n] with n cases of data in d dimensionalites of the projection method variables/features: Matrix containing the distances of the outputspace.
xlab	Label of the x axis in the resulting Plot.
ylab	Label of the y axis in the resulting Plot.
Plotter	see `PDEscatter` for details
main	Title of the Shepard diagram
sampleSize	Optional, default(500000), reduces a.ount of data for density estimation, if too many distances given

Details

Introduced and described in [Thrun, 2018, p. 63] with examples in [Thrun, 2018, p. 71-72]

Author(s)

Michael Thrun

References

[Thrun, 2018] Thrun, M. C.: Projection Based Clustering through Self-Organization and Swarm Intelligence, doctoral dissertation 2017, Springer, ISBN: 978-3-658-20540-9, Heidelberg, 2018.

Examples

```
data("Lsun3D")
Cls=Lsun3D$Cls
Data=Lsun3D$Data
InputDist=as.matrix(dist(Data))
res = stats::cmdscale(d = InputDist, k = 2, eig = TRUE,
        add = FALSE, x.ret = FALSE)

ProjectedPoints = as.matrix(res$points)
ShepardDensityScatter(InputDist,as.matrix(dist(ProjectedPoints)),main = 'MDS')
ShepardDensityScatter(InputDist[1:100,1:100],

as.matrix(dist(ProjectedPoints))[1:100,1:100],main = 'MDS')
```

Sheparddiagram *Draws a Shepard Diagram*

Description

This function plots a Shepard diagram which is a scatter plot of InputDist and OutputDist

Usage

```
Sheparddiagram(InputDists, OutputDists, xlab = "Input Distances",

                   ylab= "Output Distances", fancy = F,

 main = "ProjectionMethod", gPlot = ggplot())
```

Arguments

`InputDists`	[1:n,1:n] with n cases of data in d variables/features: Matrix containing the distances of the inputspace.
`OutputDists`	[1:n,1:n] with n cases of data in d dimensionalites of the projection method variables/features: Matrix containing the distances of the outputspace.
`xlab`	Label of the x axis in the resulting Plot.
`ylab`	Label of the y axis in the resulting Plot.
`fancy`	Set FALSE for PC and TRUE for publication
`main`	Title of the Shepard diagram
`gPlot`	ggplot2 object to plot upon.

Value

ggplot2 object containing the plot.

Author(s)

Michael Thrun

Examples

```
data("Lsun3D")
Cls=Lsun3D$Cls
Data=Lsun3D$Data
InputDist=as.matrix(dist(Data))
res = stats::cmdscale(d = InputDist, k = 2, eig = TRUE,
        add = FALSE, x.ret = FALSE)
ProjectedPoints = as.matrix(res$points)


Sheparddiagram(InputDist,as.matrix(dist(ProjectedPoints)),main = 'MDS')
```

`SignedLog` *Signed Log*

Description

Computes the Signed Log if Data

Usage

```
SignedLog(Data,Base="Ten")
```

Arguments

`Data`	[1:n,1:d] Data matrix with n cases and d variables
`Base`	Either "Ten", "Two", "Zero", or any number.

Details

A neat transformation for data, it it has a better representation on the log scale.

Value

Transformed Data

Note

Number Selections for `Base` for 2,10, "Two" or "Ten" add 1 to every datapoint as defined in the lectures.

Author(s)

Michael Thrun

References

Prof. Dr. habil. A. Ultsch, Lectures in Knowledge Discovery, 2014.

See Also

`log`

Examples

```
# sampling is done
# because otherwise the example takes too long
# in the CRAN check
data('ITS')
ind=sample(length(ITS),1000)

MDplot(SignedLog(cbind(ITS[ind],MTY[ind])*(-1),Base = "Ten"))
```

Silhouetteplot *Silhouette plot of classified data.*

Description

Silhouette plot of cluster silhouettes for the n-by-d data matrix Data or distance matrix where the clusters are defined in the vector Cls.

Usage

```
Silhouetteplot(DataOrDistances, Cls, method='euclidean',

PlotIt=TRUE,...)
```

Arguments

`DataOrDistances`	[1:n,1:d] data cases in rows, variables in columns, if not symmetric or [1:n,1:n] distance matrix, if symmetric
`Cls`	numeric vector, [1:n,1] classified data
`method`	Optional if Datamatrix is used, one of "euclidean", "maximum", "manhattan", "canberra", "binary" or "minkowski". Any unambiguous substring can be given, see `dist`
`PlotIt`	Optional, Default:TRUE, FALSE to supress the plot
`...`	If `PlotIt=TRUE`: Further arguements to `barplot`

Details

"The Silhouette plot is a common unsupervised index for visual evaluation of a clustering [L. R. Kaufman/Rousseeuw, 2005] [introduced in [Rousseeuw, 1987]]. A reasonable clustering is characterized by a silhouette width of greater than 0.5, and an average width below 0.2 should be interpreted as indicating a lack of any substantial cluster structure [Everitt et al., 2001, p. 105]. However, it is evident that silhouette scores assume clusters that are spherical or Gaussian in shape [Herrmann, 2011, pp. 91-92]" [Thrun, 2018, p. 29].

Value

`silh`	Silhouette values in a N-by-1 vector

Author(s)

Onno Hansen-Goos, Michael Thrun

References

[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.

[Rousseeuw, 1987] Rousseeuw, Peter J.: Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis, Computational and Applied Mathematics, 20, p.53-65, 1987.

Examples

```
data("Lsun3D")
Cls=Lsun3D$Cls
Data=Lsun3D$Data
#clear cluster structure
plot(Data[,1:2],col=Cls)
#However, the silhouette plot does not indicate a very good clustering in cluster 1 and 2
Silhouetteplot(Data,Cls = Cls,main='Silhouetteplot')
```

`Slopechart` *Slope Chart*

Description

ABC analysis improved slope chart

Usage

```
Slopechart(FirstDatavector,

SecondDatavector,

Names,

Labels,

MaxNumberOfSlices,

TopLabels=c('FirstDatavector','SecondDatavector'),

main='Comparision of Descending Frequency')
```

Arguments

`FirstDatavector`
: [1:n] a vector of n non unique values - a features

`SecondDatavector`
: [1:m] a vector of n non unique values - a second feature

`Labels`
: Optional, [1:k] Labels if they are specially named, if not Names are used.

`Names`	[1:k] names to search for in Datavector, if not set `unique` of Datavector is calculated.
`MaxNumberOfSlices`	Default is k, integer value defining how many labels will be shown. Everything else will be summed up to `Other`.
`TopLabels`	Labels of of feature names
`main`	title of the plot

Details

still experimental.

Value

silent output by calling `invisible` of a list with

`Percentages`	[1:k] percent values visualized in fanplot
`Labels`	[1:k] see input `Labels`, only relevant ones

Author(s)

Michael Thrun

References

[Gohil, 2015] Gohil, Atmajitsinh. R data Visualization cookbook. Packt Publishing Ltd, 2015.

See Also

`Piechart`, `Fanplot`

Examples

```
## will follow
```

`SmoothedDensitiesXY` *Smoothed Densities X with Y*

Description

Density is the smothed histogram density at [X,Y] of [Eilers/Goeman, 2004]

Usage

```
SmoothedDensitiesXY(X, Y, nbins, lambda, Xkernels, Ykernels, PlotIt = FALSE)
```

Arguments

`X`	Numeric vector [1:n], first feature (for x axis values)
`Y`	Numeric vector [1:n], second feature (for y axis values), nbins= nxy => the nr of bins in x and y is nxy nbins = c(nx,ny) => the nr of bins in x is nx and for y is ny
`nbins`	number of bins, nbins =200 (default)
`lambda`	smoothing factor used by the density estimator or c() default: lambda = 20 which roughly means that the smoothing is over 20 bins around a given point.
`Xkernels`	bin kernels in x direction are given
`Ykernels`	bin kernels y direction are given
`PlotIt`	FALSE: no plotting, TRUE: simple plot

Details

lambda has to chosen by the user and is a sensitive parameter.

Value

List of:

`Densities`	numeric vector [1:n] is the smothed density in 3D
`Xkernels`	numeric vector [1:nx], nx defined by `nbins`, such that mesh(Xkernels,Ykernels,F) form the (not NaN) smothed densisties
`Ykernels`	numeric vector [1:ny], nx defined by `nbins`, such that mesh(Xkernels,Ykernels,F) form the (not NaN) smothed densisties
`hist_F_2D`	matrix [1:nx,1:ny] beeing the smoothed 2D histogram
`ind`	an index such that `Densities` = `hist_F_2D[ind]`

Author(s)

Michael Thrun, reimplemented from Matlab (Alfred Ultsch)

References

[Eilers/Goeman, 2004] Eilers, P. H., & Goeman, J. J.: Enhancing scatterplots with smoothed densities, Bioinformatics, Vol. 20(5), pp. 623-628. 2004.

See Also

`DensityScatter`

Examples

```
data("ITS")
data("MTY")
Inds=which(ITS<900&MTY<8000)
V=SmoothedDensitiesXY(ITS[Inds],MTY[Inds])
```

StatPDEdensity *Pareto Density Estimation*

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.

stat_pde_density *Calculate Pareto density estimation for ggplot2 plots*

Description

This function enables to replace the default density estimation for ggplot2 plots with the Pareto density estimation [Ultsch, 2005]. It is used for the PDE-Optimized violin plot published in [Thrun et al, 2018].

Usage

```
stat_pde_density(mapping = NULL,
                 data = NULL,
                 geom = "violin",
                 position = "dodge",
                 ...,
                 trim = TRUE,
                 scale = "area",
                 na.rm = FALSE,
                 show.legend = NA,
                 inherit.aes = TRUE)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame.`, and will be used as the layer data.
`geom`	The geometric object to use display the data
`position`	Position adjustment, either as a string, or the result of a call to a position adjustment function.
`...`	Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `color = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.
`trim`	This parameter only matters if you are displaying multiple densities in one plot. If ‘FALSE‘, the default, each density is computed on the full range of the data. If ‘TRUE‘, each density is computed over the range of that group: this typically means the estimated x values will not line-up, and hence you won’t be able to stack density values.
`scale`	When used with geom_violin: if "area" (default), all violins have the same area (before trimming the tails). If "count", areas are scaled proportionally to the number of observations. If "width", all violins have the same maximum width.
`na.rm`	If `FALSE` (the default), removes missing values with a warning. If `TRUE` silently removes missing values.
`show.legend`	logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

inherit.aes — If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders().

Details

Pareto Density Estimation (PDE) is a method for the estimation of probability density functions using hyperspheres. The Pareto-radius of the hyperspheres is derived from the optimization of information for minimal set size. It is shown, that Pareto Density is the best estimate for clusters of Gaussian structure. The method is shown to be robust when cluster overlap and when the variances differ across clusters.

Author(s)

Felix Pape

References

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 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.

See Also

[ggplot2]stat_density

Examples

```
miris <- reshape2::melt(iris)

ggplot2::ggplot(miris,

mapping = ggplot2::aes_string(y = 'value', x = 'variable')) +

ggplot2::geom_violin(stat = "PDEdensity")
```

Worldmap — *plots a world map by country codes*

Description

The Worldmap function is used in [Thrun, 2018].

Usage

```
Worldmap(CountryCodes, Cls, Colors,

MissingCountryColor = grDevices::gray(0.8), ...)
```

Arguments

- `CountryCodes` — [1:n] vector of characters identifying countries by ISO 3166 codes (2 or 3 letters)
- `Cls` — [1:n] numerical vector of classification
- `Colors` — optional, vector of charcters specifying the used colors
- `MissingCountryColor` — if not all countries are specified in `CountryCodes` then the color of non relevant countries can be changed here
- `...` — Further arguments passed on to `plot`, see also `sp::SpatialPolygons-class`

Value

List of

- `Colors` — [1:m] colors used in map, m<=n
- `CountryCodeList` — [1:m] countries found, m<=n
- `world_country_polygons` — `SpatialPolygonsDataFrame` of `maptools`

Author(s)

Michae Thrun

References

Used in

[Thrun, 2018] Thrun, M. C. : Cluster Analysis of the World Gross-Domestic Product Based on Emergent Self-Organization of a Swarm, 12th Professor Aleksander Zelias International Conference on Modelling and Forecasting of Socio-Economic Phenomena, Foundation of the Cracow University of Economics, Zakopane, Poland, accepted, 2018.

Source for shapefile: - package maptoops and

Originally 'mappinghacks.com/data/TM_WORLD_BORDERS_SIMPL-0.2.zip', now available from https://github.com/nasa/World-Wind-Java/tree/master/WorldWind/testData/shapefiles

Examples

```
# data from [Thrun, 2018]
Cls=c(1L, 1L, 2L, 2L, 2L, 2L, 2L, 1L, 2L, 1L, 1L, 1L, 2L, 2L, 2L,
2L, 2L, 1L, 2L, 2L, 2L, 1L, 2L, 1L, 2L, 1L, 2L, 2L, 1L, 1L, 1L,
1L, 2L, 1L, 1L, 2L, 2L, 2L, 1L, 2L, 2L, 2L, 2L, 2L, 1L, 2L, 1L,
2L, 2L, 2L, 1L, 2L, 2L, 2L, 1L, 1L, 1L, 1L, 3L, 2L, 2L, 2L, 1L,
```

```
2L, 1L, 1L, 2L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 1L,
1L, 2L, 2L, 2L, 1L, 2L, 1L, 2L, 1L, 1L, 2L, 2L, 1L, 1L, 1L, 2L,
2L, 1L, 2L, 1L, 1L, 1L, 2L, 1L, 2L, 2L, 1L, 1L, 1L, 2L, 2L, 1L,
2L, 2L, 1L, 2L, 2L, 1L, 2L, 1L, 2L, 2L, 2L, 1L, 2L, 1L, 1L, 1L,
2L, 1L, 1L, 2L, 1L, 1L, 2L, 2L, 1L, 2L, 1L, 1L, 1L, 2L, 2L, 2L,
2L, 2L, 2L, 1L, 1L, 2L, 2L, 2L, 2L, 1L, 2L, 2L, 2L, 1L, 1L, 1L
)
Codes=c("AFG", "AGO", "ALB", "ARG", "ATG", "AUS", "AUT", "BDI", "BEL",
"BEN", "BFA", "BGD", "BGR", "BHR", "BHS", "BLZ", "BMU", "BOL",
"BRA", "BRB", "BRN", "BTN", "BWA", "CAF", "CAN", "CH2", "CHE",
"CHL", "CHN", "CIV", "CMR", "COG", "COL", "COM", "CPV", "CRI",
"CUB", "CYP", "DJI", "DMA", "DNK", "DOM", "DZA", "ECU", "EGY",
"ESP", "ETH", "FIN", "FJI", "FRA", "FSM", "GAB", "GBR", "GER",
"GHA", "GIN", "GMB", "GNB", "GNQ", "GRC", "GRD", "GTM", "GUY",
"HKG", "HND", "HTI", "HUN", "IDN", "IND", "IRL", "IRN", "IRQ",
"ISL", "ISR", "ITA", "JAM", "JOR", "JPN", "KEN", "KHM", "KIR",
"KNA", "KOR", "LAO", "LBN", "LBR", "LCA", "LKA", "LSO", "LUX",
"MAC", "MAR", "MDG", "MDV", "MEX", "MHL", "MLI", "MLT", "MNG",
"MOZ", "MRT", "MUS", "MWI", "MYS", "NAM", "NER", "NGA", "NIC",
"NLD", "NOR", "NPL", "NZL", "OMN", "PAK", "PAN", "PER", "PHL",
"PLW", "PNG", "POL", "PRI", "PRT", "PRY", "ROM", "RWA", "SDN",
"SEN", "SGP", "SLB", "SLE", "SLV", "SOM", "STP", "SUR", "SWE",
"SWZ", "SYC", "SYR", "TCD", "TGO", "THA", "TON", "TTO", "TUN",
"TUR", "TWN", "TZA", "UGA", "URY", "USA", "VCT", "VEN", "VNM",
"VUT", "WSM", "ZAF", "ZAR", "ZMB", "ZWE")
Worldmap(Codes,Cls)
```

`world_country_polygons` *world_country_polygons*

Description

world_country_polygons shapefile

Usage

```
data("world_country_polygons")
```

Format

world_country_polygons stores data objects using classes defined in the sp package or inheriting from those classes updated to sp Y= 1.4 and rgdal >= 1.5.

Since DataVisualization Version 1.2.1 it stores now a CRS objects with a comment containing an WKT2 CRS representation, thanks to a suggestion of Roger Bivand.

Details

Note that the rebuilt CRS object contains a revised version of the input Proj4 string as well as the WKT2 string, and may be used with both older and newer versions of sp. See maptools package for further details.

Author(s)

Hamza Tayyab, Michael Thrun

Source

maptools package

References

maptools package

Examples

```
data(world_country_polygons)
str(world_country_polygons)
```

---

zplot *Plotting for 3 dimensional data*

Description

Plots z above xy plane as 3D mountain or 2D contourlines

Usage

```
zplot(x, y, z, DrawTopView = TRUE, NrOfContourLines = 20,

                   TwoDplotter = "native", xlim, ylim)
```

Arguments

`x`	Vector of x-coordinates of the data. If y and z are missing: Matrix containing 3 rows, one for each coordinate
`y`	Vector of y-coordinates of the data.
`z`	Vector of z-coordinates of the data.
`DrawTopView`	Optional: Boolean, if true plot contours otherwise a 3D plot. Default: True
`NrOfContourLines`	Optional: Numeric. Only used when DrawTopView == True. Number of lines to be drawn in 2D contour plots. Default: 20
`TwoDplotter`	Optional: String indicating which backend to use for plotting. Possible Values: 'ggplot', 'native', 'plotly'
`xlim`	
`ylim`	

Value

If the plotting backend does support it, this will return a handle for the generated plot.

Author(s)

Felix pape

Examples

Index