Package ‘Radviz’

March 19, 2020

Type Package
Title Project Multidimensional Data in 2D Space
Version 0.9.0
Depends R (>= 3.0)
Imports ggplot2, dplyr, rlang, stats, utils, igraph, pracma
Suggests knitr, rmarkdown, bodenmiller, tidyr, RColorBrewer, cytofan, scales
LinkingTo Rcpp, RcppArmadillo
Description An implementation of the radviz projection in R. It enables the visualization of multidimensional data while maintaining the relation to the original dimensions. This package provides functions to create and plot radviz projections, and a number of summary plots that enable comparison and analysis. For reference see Ankerst *et al.* (1996) (<http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.68.1811>) for original implementation, see Di Caro *et al* (2012) (<http://link.springer.com/chapter/10.1007/978-3-642-13672-6_13>) for the original method for dimensional anchor arrangements, see Demsar *et al.* (2007) (<doi:10.1016/j.jbi.2007.03.010>) for the original Freeviz implementation.
License CC BY-NC-SA 4.0
URL http://github.com/yannabraham/Radviz
BugReports http://github.com/yannabraham/Radviz/issues
RoxygenNote 7.0.1
VignetteBuilder knitr
Encoding UTF-8
NeedsCompilation yes
Author Yann Abraham [aut, cre], Nicolas Sauwen [aut]
Maintainer Yann Abraham <yann.abraham@gmail.com>
Repository CRAN
Date/Publication 2020-03-19 12:10:22 UTC
**Description**

Plots the Dimensional Anchors and projected data points in a 2D space.

**Usage**

```r
bubbleRadviz(
  x,
  main = NULL,
  group = NULL,
  color = NULL,
  size = c(3, 16),
  label.color = NULL,
  label.size = NULL,
  bubble.color,
)```
bubbleRadviz

bubble.fg,
bubble.size,
scale,
decreasing,
add
)

Arguments

x          a radviz object as produced by do.radviz
main       [Optional] a title to the graph, displayed on top
group      the name of the grouping variable used to aggregate the data
color      [Optional] the name of the variable used to color the points
size       the size range for the plot
label.color the color of springs for visualization
label.size the size of labels
bubble.color deprecated, use geom_point instead
bubble.fg   deprecated, use geom_point instead
bubble.size deprecated, use geom_point instead
scale       deprecated, use geom_point instead
decreasing  deprecated, use geom_point instead
add         deprecated, use geom_point instead

Details

This function allows for the projection of clusters in Radviz (for example results of the SPADE algorithm), where the cluster size is derived from the number of events that fall into a specific cluster. If color is not specified the grouping variable is used.

Value

the internal ggplot2 object plus added layers, allowing for extra geoms to be added

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
bubbleRadviz(rv, group='Species')
contour.radviz

Plots the Dimensional Anchors and density lines for projected data points in a 2D space.

Description

Plots the Dimensional Anchors and density lines for projected data points in a 2D space.

Usage

```r
## S3 method for class 'radviz'
contour(
  x,
  ..., 
  main = NULL,
  color = NULL,
  size = 0.5,
  label.color = NULL,
  label.size = NULL,
  contour.color,
  contour.size,
  point.color,
  point.shape,
  point.size,
  n,
  drawlabels,
  drawpoints,
  add
)
```

Arguments

- `x`: a radviz object as produced by `do.radviz`
- `...`: further arguments to be passed to or from other methods (not implemented)
- `main`: [Optional] a title to the graph, displayed on top
- `color`: the variable in the Radviz projection used to color the contours
- `size`: The thickness of contour lines
- `label.color`: the color of springs for visualization
- `label.size`: the size of labels
- `contour.color`: deprecated, see `geom_density2d` instead
- `contour.size`: deprecated, see `geom_density2d` instead
- `point.color`: deprecated, see `geom_density2d` instead
- `point.shape`: deprecated, see `geom_density2d` instead
- `point.size`: deprecated, see `geom_density2d` instead

```r
```
cosine

Compute the Cosine Similarity between the Columns of a Data Set

Description
Given a dataset, compute the cosine similarity between to columns for use in optimization of Dimensional Anchors

Usage
cosine(mat)

Arguments
mat A matrix or data.frame

Details
implementation by David Ruau (see https://gist.github.com/bobthecat/2903031 for details)

Value
A symmetrical matrix with as many rows as there are columns in input
Author(s)
Yann Abraham
David Ruau

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
mat <- iris[,das]
sim.mat <- cosine(mat)
ncol(mat)
dim(sim.mat)

---

DB_weightedIdx  Computation of weighted version of the Davies-Bouldin index. This index serves as a measure of clustering quality of a 2D projection result with known class labels

Description

Computation of weighted version of the Davies-Bouldin index. This index serves as a measure of clustering quality of a 2D projection result with known class labels

Usage

DB_weightedIdx(x, className = NULL)

Arguments

x  an object of class Radviz, as returned by do.radviz
className  the name of the class column to use

Details

If className is left NULL (the default) the function expects a single extra column on top of the data columns (used to define springs) and the standard Radviz columns.

Value

weighted DB index value

Author(s)
Nicolas Sauwen
Perform L-Normalization on a Vector

Description

Standardizes all values in a vector to the unit vector ([0,1]) using local min and max

Usage

do.L(v, fun = range, na.rm = T)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
<td>a vector of values</td>
</tr>
<tr>
<td>fun</td>
<td>a function that will return the minimum and maximum values to use to scale v; defaults to range</td>
</tr>
<tr>
<td>na.rm</td>
<td>Logical: should NA be removed? defaults to TRUE</td>
</tr>
</tbody>
</table>

Details

This is an alternative to performing a L normalization over the full matrix. if the minimum and the maximum values returned after applying fun are the same, do.L will return 0.

Value

A vector of values of the same length as x, scaled to the unit vector.

Author(s)

Yann Abraham

Examples

data(iris)
mat <- iris[,c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')]
scaled <- apply(mat,2,do.L)
summary(scaled) # all values are between [0,1]
scaled2 <- apply(mat,2,do.L,fun=function(x) quantile(x,c(0.025,0.975)))
summary(scaled2) # all values are between [0,1]
plot(scaled,scaled2,
col=rep(seq(1,ncol(scaled)),each=nrow(scaled)),pch=16)
legend('topleft',legend=dimnames(scaled)[[2]],col=seq(1,ncol(scaled)),pch=16,bty='n')
do.optimFreeviz | Optimize the Dimensional Anchors Position using the Freeviz algorithm

**Description**

Allows to compute the best arrangement of Dimensional Anchors so that visualization efficiency (i.e. separation between classes) is maximized. The Freeviz algorithm is implemented in C++ for optimal computational efficiency.

**Usage**

```r
do.optimFreeviz(
  x, 
  classes, 
  attractG = 1, 
  repelG = 1, 
  law = 0, 
  steps = 10, 
  springs = NULL, 
  multilevel = FALSE, 
  nClusters = 5000, 
  minTreeLevels = 3, 
  subsetting = FALSE, 
  minSamples = 1000, 
  print = TRUE 
)
```

**Arguments**

- `x` | Dataframe or matrix, with observations as rows and attributes as columns
- `classes` | Vector with class labels of the observations
- `attractG` | Number specifying the weight of the attractive forces
- `repelG` | Number specifying the weight of the repulsive forces
- `law` | Integer, specifying how forces change with distance: 0 = (inverse) linear, 1 = (inverse) square
- `steps` | Number of iterations of the algorithm before re-considering convergence criterion
- `springs` | Numeric matrix with initial anchor coordinates. When NULL (=default), springs are initialized by `make.S`
- `multilevel` | Logical, indicating whether multi-level computation should be used. Setting it to TRUE can speed up computations
- `nClusters` | Number of clusters to be used at coarsest level of hierarchical tree (only used when multilevel is set to TRUE)
do.optimFreeviz

minTreeLevels  Minimum number of clustering levels to consider (only used when multilevel is set to TRUE). This parameter might over-rule nClusters.

subsetting     Logical, indicating whether a subsetting procedure should be used to compute the springs. The subset size is iteratively increased until the springs are found to be close enough to their true values, based on a confidence interval. For large datasets this option can considerably speed up computations.

minSamples     Minimum number of samples to be considered for subsetting (only used when subsetting is set to TRUE)

print          Logical, indicating whether information on the iterative procedure should be printed in the R console

Details

Freeviz is an optimization method that finds the linear projection that best separates instances of different classes, based on a physical metaphor. Observations are considered as physical particles, that exert forces onto each other. Attractive forces occur between observations of the same class, and repulsive forces between observations of different classes, with the force strength depending on the distance between observations. The goal of Freeviz is to find the projection with minimal potential energy. For more details, see the original Freeviz paper: http://dx.doi.org/10.1016/j.jbi.2007.03.010

Value

A matrix with 2 columns (x and y coordinates of dimensional anchors) and 1 line per dimensional anchor (so called springs).

Author(s)

Nicolas Sauwen

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
plot(rv,anchors.only=FALSE)
new.S <- do.optimFreeviz(x = iris[,das], classes = iris$Species)
new.rv <- do.radviz(iris,new.S)
plot(new.rv,anchors.only=FALSE)
do.optimGraphviz

Optimize the Dimensional Anchors Position using the Graphviz algorithm

Description

Allows to compute the best arrangement of Dimensional Anchors so that visualization efficiency (i.e. maintaining graph structure) is optimized. The Graphviz algorithm is implemented in C++ for optimal computational efficiency.

Usage

```r
do.optimGraphviz(
  x,
  graph,
  attractG = 1,
  repelG = 1,
  law = 0,
  steps = 10,
  springs = NULL
)
```

Arguments

- **x**: a data.frame or matrix to be projected, with column names matching row names in `springs`
- **graph**: igraph object
- **attractG**: Number specifying the weight of the attractive forces
- **repelG**: Number specifying the weight of the repulsive forces
- **law**: Integer, specifying how forces change with distance: 0 = (inverse) linear, 1 = (inverse) square
- **steps**: Number of iterations of the algorithm before re-considering convergence criterion
- **springs**: Numeric matrix with initial anchor coordinates. When `NULL` (=default), `springs` are initialized by `make.S`

Details

Graphviz is a variant of Freeviz (do.optimFreeviz, applicable to a dataset for which a graph structure (i.e. igraph object) is available. Attractive forces are defined between connected nodes in the graph, and repulsive forces between all non-connected nodes. To better maintain the original graph structure after projection, spring constants between connected nodes are proportional to their edge weights. Graphviz can be used as an alternative to Freeviz when class labels are not available.
Value

A matrix with 2 columns (x and y coordinates of dimensional anchors) and 1 line per dimensional anchor (so called springs).

Author(s)

Nicolas Sauwen

Examples

data(iris)
S <- make.S(das)
rv <- do.radviz(iris, S)

plot(rv, anchors.only=FALSE)

## compute distance matrix
d.iris <- dist(iris[, das])

## define a kNN matrix
n.iris <- as.matrix(d.iris)
n.iris <- apply(n.iris, 1, function(x, k=12) {
  x[order(x)>(k+1)] <- 0
  return(x)
})
diag(n.iris) <- 0

## compute weights for kNN matrix
w.iris <- n.iris
w.iris <- exp(-w.iris^2/(2*median(w.iris[w.iris!=0])^2))
w.iris[n.iris==0] <- 0

## create graph
library(igraph)
g.iris <- graph.adjacency(w.iris, mode='undirected', weight=TRUE, diag=FALSE)

V(g.iris)$Species <- as.character(iris[, 'Species'])
V(g.iris)$color <- as.numeric(iris[, 'Species'])

plot(g.iris, vertex.label=NA)

## project using Radviz
new.S <- do.optimGraphviz(iris[, das],
g.iris)

grv <- do.radviz(iris[, das],
  new.S, graph=g.iris)

library(ggplot2)
do.optimRadviz

Optimize the Dimensional Anchors Position for Radviz projection using a Genetic Algorithm

Description

Allows to compute the best arrangement of Dimensional Anchors so that visualization efficiency is maximized.

Usage

do.optimRadviz(
    springs,
    similarity,
    iter = 100,
    n = 1000,
    top = round(n * 0.1),
    lambda = 0.01,
    nlast = 5,
    optim = "in.da"
)

do.optim(
    springs,
    similarity,
    iter = 100,
    n = 1000,
    top = round(n * 0.1),
    lambda = 0.01,
    nlast = 5,
    optim = "in.da"
)

Arguments

springs A matrix of 2D dimensional anchor coordinates, as returned by make.S
similarity A similarity matrix measuring the correlation between Dimensional Anchors
iter The maximum number of iterations (defaults to 100)
n The number of permutations of Dimensional Anchors to be created at each generation
top The number of permutations to keep to create the next generation
lambda The threshold for the optimization process
nlast The number of generations to wait before lambda is applied
optim The optimization function (in or rv)
**Details**

The first generation is a random sampling of all Dimensional Anchors. For every generation afterwards, only the best solutions (as specified by `top`) are kept; the solutions are normalized around the unit circle (i.e., `c(1,2,3,4)` is equivalent to `c(4,1,2,3)` for Radviz projection) before the next generation is created. The next generation consists of

- all unique best solutions from the previous generation (after circular normalization)
- a permutation of all previous solutions.

Briefly, for every Dimensional Anchor position the previous generation is sampled to give a mixture of identical and slightly shifted (mutated) solutions. The algorithm will stop when the maximum number of iterations (as defined by `iter`) is reached, or when a number of generations (defined by `nlast`) as not improved over the best solution by more than a given threshold (specified by `lambda`).

**Value**

A list containing 3 sets of values:

- `perfs` the list of the best performances by generation
- `best` the best performing arrangement by generation
- `last` the top performing arrangements of the last generation

**Author(s)**

Yann Abraham

**Examples**

```r
data(iris)
S <- make.S(das)
rv <- do.radviz(iris, S)
plot(rv, anchors.only=FALSE)
sim.mat <- cosine(iris[, das])
in.da(S, sim.mat) # the starting value
new <- do.optimRadviz(S, sim.mat, iter=10, n=100)
new.S <- make.S(get.optim(new))
new.rv <- do.radviz(iris, new.S)
plot(new.rv, anchors.only=FALSE)
```

---

`do.optimRadviz` is being deprecated, please use `do.optimRadviz`. 
do.radviz

Description

do.radviz will return a projection of a multidimensional dataset onto a 2D space defined by dimensional anchors that have been projected on the unit circle using make.S

Usage

do.radviz(
  x,
  springs,
  trans = do.L,
  label.color = "orangered4",
  label.size = NA,
  type = NULL,
  graph = NULL
)

Arguments

x  a data.frame or matrix to be projected, with column names matching row names in springs
springs a matrix of 2D dimensional anchor coordinates, as returned by make.S
trans a transformation to be applied to the data before projection
label.color the color of springs for visualization
label.size the size of labels
type character string specifying the method used for obtaining the springs. Current methods are: Radviz, Freeviz and Graphviz. When not provided, type is derived from the other inputs
graph igraph object (only relevant for result obtained from do.optimGraphviz analysis)

Details

The function expects that at least some of the column names in df will be matched by row names in springs

Value

an object of class radviz with the following slots:

• proj: a ggplot2 object with a single geom_text layer corresponding to springs. the data slot of the ggplot2 corresponds to the input parameter x with the following extra columns:
get.optim

– rx and ry the X and Y coordinates of the radviz projection of x over springs
– rvalid an index of points corresponding to an invalid projection (any rx or ry is NA)
• type: character string specifying the method used for obtaining the springs.
• graphEdges: when the input graph is provided (for a Graphviz analysis), this slot will contain a dataframe with the graph edges

Author(s)
Yann Abraham

Examples

data(iris)
S <- make.S(das)
rv <- do.radviz(iris,S)
summary(rv)
data(iris)
iris0 <- rbind(iris,c(rep(0,length(das)),NA))
S <- make.S(das)
rv0 <- do.radviz(iris0,S)

sum(!is.valid(rv0)) # should be 1

# to find which points where invalid in the data
which(!is.valid(rv0))

# to review the original data points
rv1 <- subset(rv0,is.valid(rv0))

summary(rv1)

get.optim

Get the Result of the Optimization Operation

Description
Once the order of anchors has been optimized using do.optimRadviz this function can be used to recover the optimized anchors or any intermediate step

Usage
get.optim(opt, n = NULL)

Arguments
opt the result of the optimization operation performed by do.optimRadviz
n the optimized order of anchors to return; defaults to NULL, which returns the best identified combination
hexplot

Value

a character vector of the anchor names, ordered as in the n\textsuperscript{th} step of the optimization

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
sim.mat <- cosine(iris[,das])
in.da(S,sim.mat) # the starting value
new <- do.optimRadviz(S,sim.mat,iter=10,n=100)
get.optim(new) # the optimal order
get.optim(new,2) # the second step of the optimization

hexplot

A hexplot function for Radviz objects

Description

Plots the Dimensional Anchors and a hexplot-based density representation of projected data points in a 2D space.

Usage

hexplot(
x, 
main = NULL, 
nbins = 30, 
color = NULL, 
label.color = NULL, 
label.size = NULL, 
mincnt, 
style
)

Arguments

x a radviz object as produced by do.radviz
main [Optional] a title to the graph, displayed on top
nbins the number of equally spaced bins for the binning computation (see geom_hex for details)
**in.da**

**Description**

Visual efficiency of Radviz plots depends heavily on the correct arrangement of Dimensional Anchors. These functions implement the optimization strategies described in Di Caro *et al.* 2012.

**Usage**

```r
in.da(springs, similarity)
rv.da(springs, similarity)
```

**Arguments**

- `springs` A matrix of 2D dimensional anchor coordinates, as returned by `make.S`
- `similarity` A similarity matrix measuring the correlation between Dimensional Anchors

**Details**

Following the recommendation of Di Caro *et al.* we used a cosine function to calculate the similarity between Dimensional Anchors (see `cosine` for details). The in.da function implements the independent similarity measure, where the value increases as the Radviz projection improves. The rv.da function implements the radviz-dependent similarity measure, where the value decreases as the Radviz projection improves.
Value

A measure of the efficiency of the Radviz projection of the similarity matrix onto a set of springs

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
mat <- iris[,das]
sim.mat <- cosine(mat)
in.da(S,sim.mat)
rv.da(S,sim.mat)

is.radviz # Test if the object is a Radviz object

Description

The function will return TRUE if the object is a Radviz object

Usage

is.radviz(x)

Arguments

x an object of class Radviz, as returned by do.radviz

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
is.radviz(rv) # should be true
is.valid

Identify the valid projections from a Radviz object

Description

The function will return a vector as long as the data in x where points that could not be projected are TRUE.

Usage

is.valid(x)

Arguments

x 
an object of class Radviz, as returned by do.radviz

Author(s)

Yann Abraham

Examples

data(iris)
das <- c("Sepal.Length","Sepal.Width","Petal.Length","Petal.Width")
iris0 <- rbind(iris,c(rep(0,length(das)),NA))
S <- make.S(das)
rv0 <- do.radviz(iris0,S)

sum(!is.valid(rv0)) # should be 1

# to find which points where invalid in the data
which(!is.valid(rv0))

# to review the original data points
rv1 <- subset(rv0,is.valid(rv0))

summary(rv1)

make.S

Define Dimensional Anchors around the Unit Circle

Description

make.S will return [x,y] coordinates for n dimensional anchors equally spaced around the unit circle.

Usage

make.S(x)
Arguments

x  
a vector of dimensional anchors, or a list of dimensional anchors for Class Discrimination Layout, or the number of anchors to put on the circle

Details

If x is a vector or a list, values will be used to set the row names of the matrix.

Value

A matrix with 2 columns (x and y coordinates of dimensional anchors) and 1 line per dimensional anchor (so called springs). If x is a vector, the row names of the matrix will be set to the syntactically correct version of values in the vector (through a call to `make.names`). Please note that some functions expect to match column names of data to row names of the spring matrix.

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
make.S(length(das)) # without row names
make.S(das) # with row names
make.S(list(c('Sepal.Length','Sepal.Width'),c('Petal.Length','Petal.Width')))
plot.radviz

... )

Arguments

x a radviz object as produced by do.radviz
main [Optional] a title to the graph, displayed on top
anchors.only by default only plot the anchors so that other methods can easily be chained
label.color the color of springs for visualization
label.size the size of labels
point.color deprecated, use geom_point instead
point.shape deprecated, use geom_point instead
point.size deprecated, use geom_point instead
add deprecated, use geom_point instead
... further arguments to be passed to or from other methods (not implemented)

Details

by default the plot function only shows the anchors. Extra geoms are required to display the data

Value

the internal ggplot2 object, allowing for extra geoms to be added

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
plot(rv)
plot(rv,anchors.only=FALSE)

library(ggplot2)
## should look the same as before
plot(rv)+geom_point()
plot(rv)+geom_point(aes(color=Species))
Radviz

Radviz Projection of Multidimensional Data

Description

Radviz uses Dimensional Anchors and the spring paradigm to project a multidimensional space in 2D. This allows for the quick visualization of large and complex datasets.

Examples

data(iris)
das <- c("Sepal.Length","Sepal.Width","Petal.Length","Petal.Width")
S <- make.S(das)
rv <- do.radviz(iris,S)
plot(rv,anchors.only=FALSE)

---

recenter

Rotate Dimensional Anchors around the Unit Circle

Description

recenter will rotate the order of the dimensional anchors around the circle, to put a channel of reference to the top of the display.

Usage

recenter(springs, newc)

Arguments

springs a spring object as created by make.S
newc a string specifying which dimensional anchor should be placed on top of the unit circle

Value

a spring object with rotated labels

Author(s)

Yann Abraham
**Examples**

data(iris)
iris.S <- make.S(das)
iris.S
recenter(iris.S, 'Petal.Length')

---

**Description**

Rescaling of projected data for plotting

**Usage**

rescalePlot(x, fraction = 0.9)

**Arguments**

- `x` : a radviz object as produced by `do.radviz`
- `fraction` : numeric value, indicating which fraction of the unit circle should be used for the rescaled plot

**Details**

A different rescaling is used here for plotting the projected data as compared to `do.radviz`. Only feature-wise rescaling is applied to the original data (through `do.L`), in accordance with the rescaling used in `do.optimFreeviz` and `do.optimGraphviz`. The projected data is then rescaled based on amplitude, to cover a pre-specified fraction of the unit circle.

**Value**

a radviz object as produced by `do.radviz`

**Author(s)**

Nicolas Sauwen

**Examples**

data(iris)
S <- make.S(das)
rv <- do.radviz(iris, S)
library(ggplot2)
plot(rv)+geom_point(aes(color=Species))
smoothRadviz <- rescalePlot(rv)
plot(new.rv)+geom_point(aes(color=Species))

smoothRadviz

A smoothScatter function for Radviz objects

Description
Plots the Dimensional Anchors and a smoothed color density representation of projected data points in a 2D space.

Usage
smoothRadviz(
  x,
  main = NULL,
  color = "dodgerblue4",
  nbin = 200,
  label.color = NULL,
  label.size = NULL,
  smooth.color,
  max.dens,
  transformation,
  nrpoints,
  ncols,
  bandwidth
)

Arguments
x a radviz object as produced by do.radviz
main [Optional] a title to the graph, displayed on top
color the gradient will be generated from white to color
nbin the number of equally spaced grid points for the density estimation (see geom_density_2d for details)
label.color the color of springs for visualization
label.size the size of labels
smooth.color deprecated, see stat_density2d instead
max.dens deprecated, see stat_density2d instead
transformation deprecated, see stat_density2d instead
nrpoints deprecated, see stat_density2d instead
ncols deprecated, see stat_density2d instead
bandwidth deprecated, see stat_density2d instead
subset.radviz

Value

the internal ggplot2 object plus added layers, allowing for extra geoms to be added

Author(s)

Yann Abraham

Examples

data(iris)
S <- make.S(das)
rv <- do.radviz(iris, S)
smoothRadviz(rv)
Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
# subset rv
srv <- subset(rv,iris$Species=='setosa')
summary(srv)
sum(iris$Species=='setosa')  # 50 objects in srv corresponding to setosa values

summary.radviz

Radviz Object Summary, head, print, dim and springs Methods

Description

Provides a summary for Radviz objects

Usage

## S3 method for class 'radviz'
summary(object, ..., n = 6)

## S3 method for class 'radviz'
head(x, n = 6, ...)

## S3 method for class 'radviz'
dim(x)

## S3 method for class 'radviz'
print(x, ...)

springs(x)

Arguments

object an object of class Radviz, as returned by do.radviz

... further arguments to be passed to or from other methods (not implemented)

n the number of lines from each slots in the Radviz object to display (defaults to 6)

x an object of class Radviz, as returned by do.radviz

Details

dim returns the number of points and the number of dimensions used for the projection
Examples

data(iris)
S <- make.S(das)
rv <- do.radviz(iris, S)

summary(rv)
head(rv)
dim(rv)
print(rv)

Description

Text draws the strings given in the vector labels at the coordinates given by the radviz projection

Usage

## S3 method for class 'radviz'
text(
x, 
..., 
main = NULL,
labels = NULL,
size = FALSE,
label.color,
label.size,
adj,
pos,
offset,
vfont,
cex,
col,
font,
add
)
theme_radviz

Arguments

- **x**: a radviz object as produced by do.radviz
- **main**: [Optional] a title to the graph, displayed on top if add is TRUE
- **labels**: the name of the variable used for labeling (see details)
- **size**: [Logical] if TRUE labels are sized after the number of points they correspond to
- **label.color**: deprecated, see do.radviz
- **label.size**: deprecated, see do.radviz
- **adj**: deprecated, see geom_text instead
- **pos**: deprecated, see geom_text instead
- **offset**: deprecated, see geom_text instead
- **vfont**: deprecated, see geom_text instead
- **cex**: deprecated, see geom_text instead
- **col**: deprecated, see geom_text instead
- **font**: deprecated, see geom_text instead
- **add**: deprecated, see geom_text instead

Author(s)

Yann Abraham

Examples

data(iris)
das <- c("Sepal.Length","Sepal.Width","Petal.Length","Petal.Width")
S <- make.S(das)
rv <- do.radviz(iris,S)
text(rv,labels="Species")

Description

A complete Radviz theme based on ‘ggplot2::theme_light’

Usage

```r
theme_radviz(
  base_size = 11,
  base_family = "",
  base_line_size = base_size/22,
  base_rect_size = base_size/22
)
```
tuneForceRatio

Arguments

- `base_size`: base font size
- `base_family`: base font family
- `base_line_size`: base size for line elements
- `base_rect_size`: base size for rect elements

Details

On top of `ggplot2::theme_light` this theme removes axis title, text and ticks, as well as the reference grid. See `theme` for details.

Value

A complete `ggplot2` theme

Author(s)

Yann Abraham

Examples

data(iris)
das <- c('Sepal.Length','Sepal.Width','Petal.Length','Petal.Width')
S <- make.S(das)
rv <- do.radviz(iris,S)
plot(rv, main='Iris projection')
plot(rv, main='Iris projection') +
  theme_radviz(base_size=16)

---

tuneForceRatio Method to compute optimal ratio between repulsive and attractive forces for Freeviz.

Description

Method to compute optimal ratio between repulsive and attractive forces for Freeviz.

Usage

tuneForceRatio(
  x,
  classes,
  law = 0,
  steps = 10,
  springs = NULL,
  multilevel = TRUE,
  print = TRUE
)
Arguments

- `x` Dataframe or matrix, with observations as rows and attributes as columns
- `classes` Vector with class labels of the observations
- `law` Integer, specifying how forces change with distance: 0 = (inverse) linear, 1 = (inverse) square
- `steps` Number of iterations of the algorithm before re-considering convergence criterion
- `springs` Numeric matrix with initial anchor coordinates. When NULL (=default), springs are initialized by `make.S`
- `multilevel` Logical, indicating whether multi-level computation should be used. Setting it to TRUE can speed up computations
- `print` Logical, indicating whether information on the iterative procedure should be printed in the R console

Details

Running Freeviz, it is hard to know what weights to specify for the attractive and repulsive forces to optimize the projection result. This function runs an iterative procedure to find the optimal force ratio. First, a logarithmic grid search is performed, followed by 1D optimization on the refined interval. This approach is less prone to getting stuck in a suboptimal local optimum, and requires less Freeviz evaluations than direct 1D optimization

Value

Value of the optimal force ratio (attractive force in the nominator)

Author(s)

Nicolas Sauwen

Examples

data(iris)
das <- c(’Sepal.Length’,’Sepal.Width’,’Petal.Length’,’Petal.Width’)S <- make.S(das)rv <- do.radviz(iris,S)plot(rv,anchors.only=FALSE)forceRatio <- tuneForceRatio(x = iris[,das], classes = iris$Species)new.S <- do.optimFreeviz(x = iris[,das], classes = iris$Species, attractG = forceRatio, repelG = 1)new.rv <- do.radviz(iris,new.S)plot(new.rv,anchors.only=FALSE)
Index

*Topic hplot
  contour.radviz, 4
*Topic multivariate
  contour.radviz, 4

bubbleRadviz, 2

contour.radviz, 4
cosine, 5, 17

DB_weightedIdx, 6
dim.radviz (summary.radviz), 26
do.L, 7, 23
do.optim (do.optimRadviz), 12
do.optimFreeviz, 8, 10, 23
do.optimGraphviz, 10, 14, 23
do.optimRadviz, 12, 13, 15
do.radviz, 6, 14, 18, 19, 21, 23, 24, 26, 28
gem_density2d, 4, 5
gem_density_2d, 24
gem_hex, 16
gem_point, 3, 21
gem_text, 28
get.optim, 15

head.radviz (summary.radviz), 26
hexplot, 16

in.da, 17
is.radviz, 18
is.valid, 19

make.names, 20
make.S, 8, 10, 12, 14, 17, 19, 22, 30

plot.radviz, 20
print.radviz (summary.radviz), 26

Radviz, 22
range, 7

recenter, 22
rescalePlot, 23
rv.da (in.da), 17
smoothRadviz, 24
springs (summary.radviz), 26
stat_density2d, 24
stat_summary_hex, 17
subset.radviz, 25
summary.radviz, 26
text.radviz, 27
theme, 29
theme_radviz, 28
tuneForceRatio, 29