Package 'clusterfly'

February 19, 2015

Type Package

Title Explore clustering interactively using R and GGobi

Version 0.4

Author Hadley Wickham <h.wickham@gmail.com>

Maintainer Hadley Wickham <h.wickham@gmail.com>

Description Visualise clustering algorithms with GGobi. Contains both general code for visualising clustering results and specific visualisations for model-based, hierarchical and SOM clustering.

URL http://had.co.nz/clusterfly

Depends rggobi

Imports e1071, reshape2, plyr, RGtk2

Suggests som, mclust, kohonen, ggplot2

License MIT + file LICENSE

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2014-04-24 07:43:09

R topics documented:

`addhull` *Add convex hulls Add conver hulls using the tool qconvex*

Description

To use this command you must have qconvex installed and available on your path. I'm not sure if this will work on windows (probably not) but it's not a big loss, because the technique isn't very useful anyway.

Usage

```
addhull(gd, g, by)
```

Arguments

`gd`	ggobi dataset
`g`	ggobi reference
`by`	grouping variable

`as.data.frame.clusterfly`
Convert clusterfly object to data.frame. Concatenates data and cluster assignments into one data.frame. Cluster assignments are prefixed with `cl_`.

Description

Convert clusterfly object to data.frame. Concatenates data and cluster assignments into one data.frame. Cluster assignments are prefixed with `cl_`.

Usage

```
## S3 method for class 'clusterfly'
as.data.frame(x, ...)
```

Arguments

x	clusterfly object
...	ignored

`cfly_animate` *Dynamic plot: Animate glyph colours*

Description

This function will animate until you manually break the loop using Ctrl-Break or Ctrl-C.

Usage

```
cfly_animate(cf, clusters = seq_along(cf$clusters), pause = 1,
  print = TRUE, max_iterations = 100)
```

Arguments

cf	list of cluster ids that you want to animate across
clusters	clusters to display
pause	clusters number of seconds to pause between each change
print	print current cluster to screen?
max_iterations	maximum number of interations

Examples

```
# Press Ctrl-Break or Ctrl-C to exit
if (interactive()) {
o <- olive_example()
cfly_animate(cfly_clarify(o), max = 5)
close(o)
}
```

`cfly_clarify` *Match all cluster indices to common reference.*

Description

It's a good idea to run this before running any animation sequences so that unnecessary colour changes are minimised.

Usage

```
cfly_clarify(cf, reference = 1, method = "rowmax")
```

Arguments

cf	clusterfly object
reference	index to reference clustering
method	method to use, see clarify

Examples

```
o <- olive_example()
o <- cfly_clarify(o, "Region")
```

---

cfly_cluster *Add clustering.*

Description

Clustering method needs to respond to clusters, if the default does not work, you will need to write your own to extract clusters.

Usage

```
cfly_cluster(cf, method, ..., name = deparse(substitute(method)))
```

Arguments

cf	clusterfly object
method	clusterfing method (function)
...	arguments passed to clustering method
name	name of clustering

Examples

```
o <- olive_example()
cfly_cluster(o, kmeans, 4)
cfly_cluster(o, kmeans, 4, name="blah")
```

cfly_dist *Static plot: Variable distribution. Draw a density plot for each continuous variable, facetted across clustering.*

Description

This allows you to quickly visualise how the cluster vary in a univariate manner. Currently, it is a bit of a hack, because ggplot does not support plots with different scales, so the variables are manually rescaled prior to plotting.

Usage

```
cfly_dist(cfly, index, scale = "range")
```

Arguments

cfly	clusterfly object
index	clustering to use
scale	scaling to use

Details

This plot is inspired by Gaguin http://www.rosuda.org/gaguin.

Examples

```
if (require("ggplot2")) {
o <- olive_example()
cfly_dist(o, "kmeans")
cfly_dist(o, "kmeans") + scale_y_continuous(limit=c(0, 2))
}
```

cfly_fluct *Static plot: Fluctuation diagram. Draw a fluctuation diagram comparing two clusterings.*

Description

Static plot: Fluctuation diagram. Draw a fluctuation diagram comparing two clusterings.

Usage

```
cfly_fluct(cfly, a, b, clarify = TRUE)
```

Arguments

cfly	clusterfly object
a	first clustering, will be reordered to match b if clarify=TRUE
b	second clustering
clarify	use clarify to rearranged cluster indices?

Examples

```
if (require("ggplot2")) {
o <- olive_example()
cfly_fluct(o, "kmeans", "Region")
cfly_fluct(o, "kmeans", "Region", clarify = FALSE)
}
```

---

cfly_pcp *Static plot: Parallel coordinates. Draw a parallel coordinates plot, facetted across clustering.*

Description

This really only a proof of concept, a truly useful PCP needs interaction, especially to move the variables around.

Usage

```
cfly_pcp(cfly, index, ...)
```

Arguments

cfly	clusterfly object
index	clustering to use
...	other arguments passed to geom_line

Examples

```
if (require("ggplot2")) {
o <- olive_example()
cfly_pcp(o, "kmeans")
cfly_pcp(o, "kmeans", alpha = 1/10)
cfly_pcp(o, "kmeans", alpha = 1/10) + coord_flip()
}
```

`cfly_show` *Show in ggobi. Opens an instance ggobi for this dataset (if not already open), and colours the points according the cluster assignment.*

Description

Show in ggobi. Opens an instance ggobi for this dataset (if not already open), and colours the points according the cluster assignment.

Usage

```
cfly_show(cf, idx = "true", hulls = FALSE)
```

Arguments

`cf`	clusterfly object
`idx`	clustering to display
`hulls`	add convex hull? see `addhull` for details

Examples

```
o <- olive_example()
cfly_show(o, 1)
cfly_show(o, "Region")
if (!interactive()) close(o)
```

`clarify` *Clarify matrix Clarify matrix ordering to minimize off diagonals*

Description

Clarify matrix Clarify matrix ordering to minimize off diagonals

Usage

```
clarify(a, b, method = "greedy")
```

Arguments

`a`	cluster assignments to reassign
`b`	matrix b
`method`	clarification method

Value

vector of reassigned cluster a

See Also

matchClasses

---

clusterfly — *Creates a convenient data structure for dealing with a dataset and a number of alternative clusterings.*

Description

Once you have created a clusterfly object, you can add clusterings to it with cfly_cluster, and visualise then in GGobi with cfly_show and cfly_animate. Static graphics are also available: cfly_pcp will produce a parallel coordinates plot, cfly_dist will show the distribution of each variable in each cluster, and cfly_fluct compares two clusterings with a fluctuation diagram.

Usage

```
clusterfly(df, extra = NULL, rescale = TRUE)
```

Arguments

df	data frame to be clustered
extra	extra variables to be included in output, but not clustered
rescale	rescale, if true each variable will be scaled to have mean 0 and variance 1.

Details

If you want to standardise the cluster labelling to one group, look at clarify and cfly_clarify

See Also

vignette("introduction")

Examples

```
ol <- olive_example()

if (interactive()) {
ggobi(ol)
cfly_show(ol, "k4-1")
cfly_animate(ol, max = 5)
close(ol)
}
```

cut.hierfly *Cut hierfly object into k clusters/colours.*

Description

Cut hierfly object into k clusters/colours.

Usage

```
## S3 method for class 'hierfly'
cut(x, k = 2, g = ggobi(x), ...)
```

Arguments

x	hierfly object to colour
k	number of clusters
g	GGobi instance displaying x, will create new if not specified
...	ignored

Examples

```
h <- hierfly(iris)
hfly <- ggobi(h)
cut(h, 2, hfly)
h <- hierfly(iris, method="ward")
g <- ggobi(h)
cut(h, 2, g)
```

ggobi.hierfly *Visualise hierarchical clustering with GGobi. Displays both data and dendrogram in original high-d space.*

Description

This adds four new variables to the original data set:

Usage

```
## S3 method for class 'hierfly'
ggobi(data, ...)
```

Arguments

data	hierfly object to visualise in GGobi
...	ignored

Details

- ORDER, the order in which the clusters are joined
- HEIGHT, the height of the branch, ie. the dissimilarity between the branches
- LEVEL, the level of the branch
- POINTS, the number of points in the branch

Make sure to select "attach edge set (edges)" in the in the edges menu on the plot window, when you create a new plot.

A tour over the original variables will show how the clusters agglomerate in space. Plotting order vs height, level or points will give various types of dendograms. A correlation tour with height/level/points on the y axis and the original variables on the x axis will show a mobile blowing in the wind.

See Also

cut.hierfly

Examples

```
h <- hierfly(iris)
ggobi(h)
h <- hierfly(iris, method="single")
```

---

ggobi.som *Visualise Kohonen self organising maps with GGobi Displays both data, and map in original high-d space.*

Description

Map variables added as map1 and map2. Plot these to get traditional SOM plot. Tour over all other variables to see how well the map fits the original data.

Usage

```
## S3 method for class 'som'
ggobi(data, ...)
```

Arguments

data	SOM object
...	ignored

Examples

```
## Not run:
d.music <- read.csv("http://www.ggobi.org/book/data/music-all.csv")

music <- rescaler(d.music)[complete.cases(d.music), 1:10]
music.som <- som::som(music[,-(1:3)], 6, 6, neigh="bubble", rlen=1000)
ggobi(music.som)

## End(Not run)
## Not run:
d.music <- read.csv("http://www.ggobi.org/book/data/music-all.csv")

music <- rescaler(d.music)[complete.cases(d.music), 1:10]
music.hex <- kohonen::som(music[,-(1:3)], grid = somgrid(3, 3, "hexagonal"), rlen=1000)
music.rect <- kohonen::som(music[,-(1:3)], grid = somgrid(6, 6, "rectangular"), rlen=1000)
ggobi(music.rect)

## End(Not run)
```

`hierarchical` *Hierachical clustering Convenient methods for hierachical clustering*

Description

Hierachical clustering Convenient methods for hierachical clustering

Usage

```
hierarchical(df, method = "complete", metric = "euclidean", n = 5)
```

Arguments

`df`	data frame
`method`	method to use, see `hclust`
`metric`	distance metric to use, see `dist`
`n`	number of clusters to retrieve, see `cut`

`hierfly` *Visualisig hierarchical clustering. This method supplements a data set with information needed to draw a dendrogram*

Description

Intermediate cluster nodes are added as needed, and positioned at the centroid of the combined clusters.

Usage

```
hierfly(data, metric = "euclidean", method = "average")
```

Arguments

`data`	data set
`metric`	distance metric to use, see `dist` for list of possibilities
`method`	cluster distance measure to use, see `hclust` for details

Value

object of type, hierfly

See Also

`cut.hierfly`, `ggobi.hierfly`

Examples

```
h <- hierfly(iris)
ggobi(h)
h <- hierfly(iris, method="single")
```

`mefly` *Display model based clustering with mvn ellipses. Displays the results of model based clustering with an ellipse drawn from the multivariate normal model for each group.*

Description

Display model based clustering with mvn ellipses. Displays the results of model based clustering with an ellipse drawn from the multivariate normal model for each group.

Usage

```
mefly(model, data)
```

Arguments

model	output from me function
data	input data frame to me

Examples

```
if(require("mclust")) {
eei <- me(modelName = "EEI", data = iris[,-5], z = unmap(iris[,5]))
vvv <- me(modelName = "VVV", data = iris[,-5], z = unmap(iris[,5]))
vvi <- me(modelName = "VVI", data = iris[,-5], z = unmap(iris[,5]))
mefly(eei, iris[,-5])
mefly(vvi, iris[,-5])
mefly(vvv, iris[,-5])
}
```

---

olive_example *Example clusterfly object created with olives data*

Description

Example clusterfly object created with olives data

Usage

```
olive_example()
```

Index