Package 'greenclust'

January 10, 2020

Type Package

Title Combine Categories Using Greenacre's Method

Version 1.1.0

Description Implements a method of iteratively collapsing the rows of a contingency table, two at a time, by selecting the pair of categories whose combination yields a new table with the smallest loss of chi-squared, as described by Greenacre, M.J. (1988) <doi:10.1007/BF01901670>. The result is compatible with the class of object returned by the 'stats' package's hclust() function and can be used similarly (plotted as a dendrogram, cut, etc.). Additional functions are provided for automatic cutting and diagnostic plotting.

License MIT + file LICENSE

Encoding UTF-8

URL https://github.com/JeffJetton/greenclust

BugReports https://github.com/JeffJetton/greenclust/issues

LazyData true

RoxygenNote 7.0.2

Suggests testthat, knitr, rmarkdown

NeedsCompilation no

Author Jeff Jetton [aut, cre]

Maintainer Jeff Jetton <jeff.jetton@gmail.com>

Repository CRAN

Date/Publication 2020-01-10 16:50:08 UTC

R topics documented:

`assign.cluster` *Assign clusters to a new vector of categories*

Description

Maps a vector of cluster numbers to another categorical vector, yielding a new vector of matching cluster numbers. Useful for distributing cluster numbers back out to the original observations in cases where the clustering was performed on a table of unique levels rather than directly on the observations (such as with `greenclust`).

Usage

```
assign.cluster(x, clusters, impute = FALSE)
```

Arguments

`x`	a factor or character vector representing a categorical variable
`clusters`	a named numeric vector of cluster numbers, such as an object returned by `greencut` or `cutree`
`impute`	a boolean controlling the behavior when a value in `x` is not found in `names(clusters)` (see Details).

Details

Any categories in `x` that do not exist in `names(clusters)` are given a cluster of `NA`, or (if `impute` is `TRUE`) assigned the cluster number that is most-frequently used for the other existing categories, with ties going to the lowest cluster number. If there are no matching clusters for any of the categories in `x`, imputation will simply use the first cluster number in `clusters`.

If there are duplicate names in `clusters`, the first occurrence takes precedence.

Value

A factor vector of the same length as `x`, representing assigned cluster numbers.

See Also

`greenclust`, `greencut`, `greenplot`

Examples

```
# Cluster feed types based on number of "underweight" chicks
grc <- greenclust(table(chickwts$feed,
                        ifelse(chickwts$weight < 200, "Y", "N")))
# Assign clusters to each original observation
feed.clustered <- assign.cluster(chickwts$feed, greencut(grc))
table(chickwts$feed, feed.clustered)
```

greenclust *Row Clustering Using Greenacre's Method*

Description

Iteratively collapses the rows of a table (typically a contingency table) by selecting the pair of rows each time whose combination creates the smalled loss of chi-squared.

Usage

```
greenclust(x, correct = FALSE, verbose = FALSE)
```

Arguments

x	a numeric matrix or data frame
correct	a logical indicating whether to apply a continuity correction if and when the clustered table reaches a 2x2 dimension.
verbose	if TRUE, prints the clustered table along with r-squared and p-value at each step

Value

An object of class `greenclust` which is compatible with most `hclust` object functions, such as `plot()` and `rect.hclust()`. The height vector represents the proportion of chi-squared, relative to the original table, seen at each clustering step. The greenclust object also includes a vector for the chi-squared test p-value at each step and a boolean vector indicating whether the step had a tie for "winner".

References

Greenacre, M.J. (1988) "Clustering the Rows and Columns of a Contingency Table," *Journal of Classification 5*, 39-51. https://doi.org/10.1007/BF01901670

See Also

`greencut`, `greenplot`, `assign.cluster`

Examples

```
# Combine Titanic passenger attributes into a single category
tab <- t(as.data.frame(apply(Titanic, 4:1, FUN=sum)))
# Remove rows with all zeros
tab <- tab[apply(tab, 1, sum) > 0, ]

# Perform clustering on contingency table
grc <- greenclust(tab)

# Plot r-squared and p-values for each potential cut point
greenplot(grc)
```

```
# Get clusters at suggested cut point
clusters <- greencut(grc)

# Plot dendrogram with clusters marked
plot(grc)
rect.hclust(grc, max(clusters))
```

---

greencut *Cut a Greenclust Tree into Optimal Groups*

Description

Cuts a greenclust tree at an automatically-determined number of groups.

Usage

```
greencut(g, k = NULL, h = NULL)
```

Arguments

g	a tree as producted by greenclust
k	an integer scalar with the desired number of groups
h	numeric scalar with the desired height where the tree should be cut

Details

The cut point is calculated by finding the number of groups/clusters that results in a collapsed contingency table with the most-significant (lowest p-value) chi-squared test. If there are ties, the smallest number of groups wins.

If a certain number of groups is required or a specific r-squared (1 - height) threshold is targeted, values for either k or h may be provided. (While the regular cutree function could also be used in this circumstance, it may still be useful to have the additional attributes that greencut() provides.)

As with cutree(), k overrides h if both are given.

Value

greencut returns a vector of group memberships, with the resulting r-squared value and p-value as object attributes, accessable via attr.

References

Greenacre, M.J. (1988) "Clustering the Rows and Columns of a Contingency Table," *Journal of Classification 5*, 39-51. https://doi.org/10.1007/BF01901670

See Also

greenclust, greenplot, assign.cluster

Examples

```
# Combine Titanic passenger attributes into a single category
# and create a contingency table for the non-zero levels
tab <- t(as.data.frame(apply(Titanic, 4:1, FUN=sum)))
tab <- tab[apply(tab, 1, sum) > 0, ]

grc <- greenclust(tab)
greencut(grc)

plot(grc)
rect.hclust(grc, max(greencut(grc)),
            border=unique(greencut(grc))+1)
```

---

greenplot *Plot Statistics for a Greenclust Object*

Description

Displays a connected scatterplot showing the r-squared values (x-axis) and p-values (y-axis) at each clustering step of a greenclust object. Points are labeled with their cutpoints, i.e., the number of groups/clusters found at each step. The point with the lowest p-value (typically the optimal cutpoint) is highlighted.

Usage

```
greenplot(
  g,
  type = "b",
  bg = "gray75",
  pch = 21,
  cex = 1,
  optim.col = "red",
  pos = 2,
  main = "P-Value vs. R-Squared for Num. Clusters",
  xlab = "r-squared",
  ylab = NULL,
  ...
)
```

Arguments

g	an object of the type produced by greenclust
type	1-character string giving the type of plot desired: "p" for points, "l" for lines, and "b" (the default) for both points and lines.

bg	a vector of background colors for open plot symbols. Also used for the line color if type is "b".
pch	a vector of plotting characters or symbols: see points
cex	a numerical vector giving the amount by which plotting characters and symbols should be scaled relative to the default. For this plot, the numeric labels on each point are always scaled to 0.80 of this value.
optim.col	color to use for highlighting the "optimal" cutpoint.
pos	specifies the position of labels relative to their points: 1 = below, 2 = left, 3 = above, and 4 = right.
main	an overall title for the plot.
xlab	a title for the x axis.
ylab	a title for the y axis.
...	additional arguments to be passed to the plotting methods.

References

Greenacre, M.J. (1988) "Clustering the Rows and Columns of a Contingency Table," *Journal of Classification 5*, 39-51. https://doi.org/10.1007/BF01901670

See Also

greenclust, greencut, assign.cluster

Examples

```
# Combine Titanic passenger attributes into a single category
# and create a contingency table for the non-zero levels
tab <- t(as.data.frame(apply(Titanic, 4:1, FUN=sum)))
tab <- tab[apply(tab, 1, sum) > 0, ]

grc <- greenclust(tab)
greenplot(grc)


# Plot using custom graphical parameters
greenplot(grc, type="p", bg="lightblue", optim.col="darkorange",
          pos=3, bty="n", cex.main=2, col.main="blue")
```

Index