Package ‘protoclust’

February 22, 2015

Type Package

Title Hierarchical Clustering with Prototypes

Version 1.5

Date 2015-02-18

Author Jacob Bien and Rob Tibshirani

Maintainer Jacob Bien <jbien@cornell.edu>


License GPL-2

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-02-22 00:09:59

R topics documented:

protoclust-package ......................................................... 2
plotwithprototypes ....................................................... 3
protoclust ................................................................. 5
protocut ................................................................. 7

Index 10
Hierarchical Clustering with Prototypes: Minimax Linkage.

Description

Functions to perform minimax linkage hierarchical clustering and to cut such trees to return clusterings with prototypes.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>protoclust</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2011-06-21</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Jacob Bien and Rob Tibshirani

Maintainer: Jacob Bien <jbien@cornell.edu>

References


See Also

protoclust, protocut, plotwithprototypes

Examples

# generate some data:
set.seed(1)
n <- 100
p <- 2
x <- matrix(rnorm(n * p), n, p)
rownames(x) <- paste("A", 1:n, sep="")
d <- dist(x)

# perform minimax linkage clustering:
hc <- protoclust(d)
# cut the tree to yield a 10-cluster clustering:
k <- 10  # number of clusters
cut <- protocut(hc, k=k)
h <- hc$height[n - k]

# plot dendrogram (and show cut):
plotwithprototypes(hc, imerge=cut$imerge, col=2)
abline(h=h, lty=2)

# get the prototype assigned to each point:
pr <- cut$protos[cut$cl]

# find point farthest from its prototype:
dmat <- as.matrix(d)
ifar <- which.max(dmat[cbind(1:n, pr[1:n])])

# note that this distance is exactly h:
stopifnot(dmat[ifar, pr[ifar]] == h)

# since this is a 2d example, make 2d display:
plot(x, type="n")
points(x, pch=20, col="lightblue")
lines(rbind(x[ifar, ], x[pr[ifar], ]), col=3)
points(x[cut$protos, ], pch=20, col="red")
tt <- seq(0, 2 * pi, length=100)
for (i in cut$protos) {
  lines(x[i, 1] + h * cos(tt), x[i, 2] + h * sin(tt))
}

---

### plotwithprototypes

**Plot dendrogram with prototype labels added**

#### Description

Makes a plot of the dendrogram (using `plotNhclust`) and adds labels of prototypes on the interior nodes of a dendrogram.

#### Usage

```r
plotwithprototypes(hc, imerge = -seq(n), labels = NULL, bgcol = "white",
                   font = 1, col = 1, cex = 1, ...)  
```

#### Arguments

- **hc**: an object of class `protoclust` (as returned by the function `protoclust`)
- **imerge**: a vector of the nodes whose prototype labels should be added. Interior nodes are numbered from 1 (lowest merge) to n - 1 (highest merge, i.e. the root) and leaf-nodes are negative (so if element i is a prototype for a singleton cluster, then
-i is included in imerge). Example: `seq(1, n - 1)` means every interior node is labeled with a prototype. For larger trees, showing only the prototypes at a given cut may be easier (described more below). Default: `-seq(n)`, meaning all leaf labels and no interior-node labels are shown.

labels

an optional character vector of length n giving the labels of the elements clustered. If not provided, hc$labels is used (if not NULL) or else labels are taken to be `seq(n)`.

bgcol

background color for prototype labels

colLfont

color and font of prototype labels

cex

size of prototype label

...  

additional arguments to be passed to `plot.hclust`, such as hang

Details

This function lets one put prototype labels on a dendrogram. The argument imerge controls which interior nodes and leaves are labeled. A convenient choice for the argument imerge is the imerge-output of `protocut`. This allows one to label a dendrogram with the prototypes of a particular cut. See examples below. This function is called when one writes `plot(hc)`, where hc is an object of class protoclust.

Author(s)

Jacob Bien and Rob Tibshirani

References


See Also

`protoclust`, `protocut`

Examples

# generate some data:
set.seed(1)
n <- 100
p <- 2
x <- matrix(rnorm(n * p), n, p)
rownames(x) <- paste("x", 1:n, sep="")
d <- dist(x)

# perform minimax linkage clustering:
hc <- protoclust(d)

# cut the tree to yield a 10-cluster clustering:
k <- 10 # number of clusters
cut <- protocut(hc, k=k)
protoclust

Hierarchical clustering with prototypes: Minimax linkage.

Description

Performs minimax linkage hierarchical clustering given a set of dissimilarities. Returns an object that looks just like the output of \texttt{hclust} except that it has an additional element containing prototype indices.

Usage

\texttt{protoclust(d, verb = FALSE)}

Arguments

d dissimilarities object. Can be of class \texttt{dist} or \texttt{matrix}

verb see verbose output?

Details

This function provides an efficient implementation of minimax linkage hierarchical clustering. Consider two clusters G and H and their union U. The minimax linkage between G and H is defined to be the radius of the smallest ball that encloses all of U and that is centered at one of the points in U. If G and H are merged together, the prototype for the newly formed cluster U is that enclosing ball’s center. By construction, the prototype for a cluster will always be one of the objects being clustered. For more on minimax linkage and how one can use prototypes to help interpret a dendrogram, see Bien, J., and Tibshirani, R. (2011), "Hierarchical Clustering with Prototypes via Minimax Linkage," accepted for publication in \textit{The Journal of the American Statistical Association}, DOI: 10.1198/jasa.2011.tm10183.

This function has been designed to work like \texttt{hclust} in terms of inputs and outputs; however, unlike \texttt{hclust}, it outputs an additional element, namely a vector of length \(n - 1\) containing the indices of prototypes. It follows \texttt{hclust}’s convention for making the arbitrary choice of whether to put a subtree on the left or right side.
For cutting a minimax linkage hierarchical clustering, use `protocut`, which works like `cutree` except that it returns the set of prototypes in addition to the cluster assignments.

This function calls a C implementation of the algorithm detailed in Bien and Tibshirani (2011) that is based on an algorithm described in Murtagh (1983).

**Value**

An object of class `protoclust`, which is just like `hclust` but has an additional element:

- `merge`, `height`, `order`
  - identical to the values returned by `hclust`
- `protos` a vector of length `n - 1`. The `i`-th element is the index of the prototype corresponding to the cluster formed on the `i`-th merge.

**Author(s)**

Jacob Bien and Rob Tibshirani

**References**


**See Also**

`protocut`, `plotwithprototypes`, `hclust`

**Examples**

```r
# generate some data:
set.seed(1)
n <- 100
p <- 2
x <- matrix(rnorm(n * p), n, p)
rownames(x) <- paste("a", 1:n, sep="")
d <- dist(x)

# perform minimax linkage clustering:
hc <- protoclust(d)

# cut the tree to yield a 10-cluster clustering:
k <- 10 # number of clusters
cut <- protocut(hc, k=k)
h <- hc$height[n - k]

# plot dendrogram (and show cut):
plotwithprototypes(hc, imerge=cut$imerge, col=2)
abline(h=h, lty=2)
```
# get the prototype assigned to each point:
pr <- cut$protos[cut$cl]

# find point farthest from its prototype:
dmat <- as.matrix(d)
ifar <- which.max(dmat[cbind(1:n, pr[1:n])])

# note that this distance is exactly h:
stopifnot(dmat[ifar, pr[ifar]] == h)

# since this is a 2d example, make 2d display:
plot(x, type="n")
points(x, pch=20, col="lightblue")
lines(rbind(x[ifar, ], x[pr[ifar], ]), col=3)
points(x[cut$protos, ], pch=20, col="red")
tt <- seq(0, 2 * pi, length=100)
for (i in cut$protos) {
  lines(x[i, 1] + h * cos(tt), x[i, 2] + h * sin(tt))
}

protocut  
Cut a minimax linkage tree to get a clustering

Description

Cuts a minimax linkage tree to get one of n - 1 clusterings. Works like cutree except also returns the prototypes of the resulting clustering.

Usage

protocut(hc, k = NULL, h = NULL)

Arguments

hc  
an object returned by protoclust
k  
the number of clusters desired
h  
the height at which to cut the tree

Details

Given a minimax linkage hierarchical clustering, this function cuts the tree at a given height or so that a specified number of clusters is created. It returns both the indices of the prototypes and their locations. This latter information is useful for plotting a dendrogram with prototypes (see plotwithprototypes). As with cutree, if both k and h are given, h is ignored. Unlike cutree, in current version k and h cannot be vectors.
Value

A list corresponding to the clustering from cutting tree:

cl vector of cluster memberships
protos vector of prototype indices corresponding to the k clusters created. protos[i] gives the index of the prototype for all elements with cl==i
imerge vector describing the nodes where prototypes occur. We use the naming convention of the merge matrix in hclust: if imerge[i] is positive, it is the interior node (counting from the bottom) of the cluster with elements which(cl==i); if imerge[i] is negative, then this is a singleton cluster with a leaf as prototype.

Author(s)

Jacob Bien and Rob Tibshirani

References


See Also

protoclust, cutree, plotwithprototypes

Examples

# generate some data:
set.seed(1)
n <- 100
p <- 2
x <- matrix(rnorm(n * p), n, p)
rownames(x) <- paste("A", 1:n, sep="")
d <- dist(x)

# perform minimax linkage clustering:
hc <- protoclust(d)

# cut the tree to yield a 10-cluster clustering:
k <- 10 # number of clusters
cut <- protocut(hc, k=k)
h <- hc$height[n - k]

# plot dendrogram (and show cut):
plotwithprototypes(hc, imerge=cut$imerge, col=2)
abline(h=h, lty=2)

# get the prototype assigned to each point:
pr <- cut$protos[cut$cl]

# find point farthest from its prototype:
dmat <- as.matrix(d)
ifar <- which.max(dmat[cbind(1:n, pr[1:n])])

# note that this distance is exactly h:
stopifnot(dmat[ifar, pr[ifar]] == h)

# since this is a 2d example, make 2d display:
plot(x, type="n")
points(x, pch=20, col="lightblue")
lines(rbind(x[ifar, ], x[pr[ifar], ]), col=3)
points(x[cut$protos, ], pch=20, col="red")
tt <- seq(0, 2 * pi, length=100)
for (i in cut$protos) {
  lines(x[i, 1] + h * cos(tt), x[i, 2] + h * sin(tt))
}
Index

*Topic `cluster`
  - `plotwithprototypes`, 3
  - `protoclust`, 5
  - `protoclust-package`, 2
  - `protocut`, 7

`cutree`, 6–8

`hclust`, 6

`plotwithprototypes`, 2, 3, 6–8
`protoclust`, 2, 4, 5, 8
`protoclust-package`, 2
`protocut`, 2, 4, 6, 7