

# Package ‘clst’

January 20, 2025

**Type** Package  
**Title** Classification by local similarity threshold  
**Version** 1.54.0  
**Depends** R (>= 2.10)  
**Imports** ROC, lattice  
**Suggests** RUnit  
**LazyLoad** yes  
**LazyData** yes  
**Author** Noah Hoffman  
**Maintainer** Noah Hoffman <ngh2@uw.edu>  
**Description** Package for modified nearest-neighbor classification based on calculation of a similarity threshold distinguishing within-group from between-group comparisons.  
**License** GPL-3  
**biocViews** Classification  
**git\_url** <https://git.bioconductor.org/packages/clst>  
**git\_branch** RELEASE\_3\_20  
**git\_last\_commit** 4cc5941  
**git\_last\_commit\_date** 2024-10-29  
**Repository** Bioconductor 3.20  
**Date/Publication** 2025-01-20

## Contents

|                         |           |
|-------------------------|-----------|
| clst-package . . . . .  | 2         |
| actino . . . . .        | 3         |
| bvseqs . . . . .        | 3         |
| classify . . . . .      | 4         |
| findThreshold . . . . . | 7         |
| plotDistances . . . . . | 9         |
| printClst . . . . .     | 10        |
| scaleDistPlot . . . . . | 10        |
| strep . . . . .         | 12        |
| <b>Index</b>            | <b>14</b> |

---

clst-package

*Classification by local similarity threshold*

---

## Description

Package for modified nearest-neighbor classification based on calculation of a similarity threshold distinguishing within-group from between-group comparisons.

## Details

Package: clst  
Type: Package  
License: GPL-3  
Author: Noah Hoffman <ngh2@uw.edu>

Index:

Further information is available in the following vignettes:

clstDemo clst (source, pdf)

TODO: write package overview.

## Author(s)

Noah Hoffman

Maintainer: <ngh2@uw.edu>

## See Also

[cmdscale](#)

## Examples

```
library(clst)
packageDescription("clst")
data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species
i <- 1
cc <- classify(dmat, groups, dvect=dmat[i,])
cat('query at i =',i,'is species',paste('I.', groups[i]),'\n')
printClst(cc)
i <- 125
cc <- classify(dmat, groups, dvect=dmat[i,])
cat('query at i =',i,'is species',paste('I.', groups[i]),'\n')
printClst(cc)
```

---

|        |                             |
|--------|-----------------------------|
| actino | <i>Actinomyces data set</i> |
|--------|-----------------------------|

---

**Description**

Square matrices describing pairwise distances among 16s rRNA sequences.

**Usage**

```
data(actino)
```

**Format**

```
List of 5
 $ dmat1 : num [1:146, 1:146] 0 0.763 1.25 10.345 12.771 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
 $ dmat2 : num [1:146, 1:146] 0 0.574 1.044 5.669 8.409 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
 $ dmat3 : num [1:146, 1:146] 0 0.763 1.25 8.571 11.233 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
  .. ..$ : chr [1:146] "200" "201" "202" "203" ...
 $ taxa : Factor w/ 33 levels "Actinomyces bowdenii",...: 12 12 12 23 20 20 8 22 12 20 ...
 $ abbrev: Factor w/ 33 levels "A bowdenii","A canis",...: 12 12 12 23 20 20 8 22 12 20 ...
```

**Details**

The matrices `$dmat1`, `dmat2`, and `dmat3` contain percent nucleotide difference with indels penalized heavily, little, and somewhat, respectively.

`$taxa` is a factor of species names; abbreviations of the same names are found in `$abbrev`.

**Examples**

```
data(actino)
```

---

|        |                          |
|--------|--------------------------|
| bvseqs | <i>BV reference set.</i> |
|--------|--------------------------|

---

**Description**

Tree-derived pairwise distances and taxonomic assignments among 16S rRNA sequences representing bacteria represented in the vaginal mucosa.

**Usage**

```
data(bvseqs)
```

**Format**

```

The format is:
List of 3
$ dmat      : num [1:448, 1:448] 0 0.0494 0.0968 0.1002 0.1606 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:448] "S001098970" "S000859776" "S000539896" "S001352901" ...
  .. ..$ : chr [1:448] "S001098970" "S000859776" "S000539896" "S001352901" ...
$ groupTab:'data.frame': 448 obs. of  12 variables:
  ..$ superkingdom   : chr [1:448] "2" "2" "2" "2" ...
  ..$ superphylum  : chr [1:448] NA NA NA NA ...
  ..$ phylum       : chr [1:448] "1224" "1224" "1224" "1224" ...
  ..$ class          : chr [1:448] "1236" "1236" "1236" "1236" ...
  ..$ subclass       : chr [1:448] NA NA NA NA ...
  ..$ order          : chr [1:448] "72274" "72274" "72274" "72274" ...
  ..$ suborder       : chr [1:448] NA NA NA NA ...
  ..$ family         : chr [1:448] "468" "468" "468" "468" ...
  ..$ genus          : chr [1:448] "469" "469" "469" "469" ...
  ..$ species_group  : chr [1:448] NA NA NA NA ...
  ..$ species_subgroup: chr [1:448] NA NA NA NA ...
  ..$ species        : chr [1:448] "470" "470" "471" "470" ...
$ taxNames: Named chr [1:212] "Actinomyces urogenitalis" "Lactobacillus jensenii" "Proteobacteria"
  ..- attr(*, "names")= chr [1:212] "103621" "109790" "1224" "1236" ...

```

**Details**

(Describe creation of this data set)

**Source**

Sequences were assembled from both the RDP 16S rRNA database and from the laboratory of Dr. David Fredricks.

**References**

RDP url here.

**Examples**

```

data(bvseqs)
## maybe str(bvseqs) ; plot(bvseqs) ...

```

---

classify

*classify*

---

**Description**

Functions to perform classification by local similarity threshold.

**Usage**

```

classify(dmat, groups, dvect, method = "mutinfo", minScore = 0.45,
         doffset = 0.5, dStart = NA, maxDepth = 10, minGroupSize = 2,
         objNames = names(dvect), keep.data = TRUE, ..., verbose =
         FALSE)

classifyIter(dmat, groupTab, dvect, dStart = NA, multiple = FALSE,
            keep.data = TRUE, ..., verbose = FALSE)

classifier(dmat, groups, dvect, method = 'mutinfo', minScore = 0.45,
          doffset = 0.5, dStart = NA, minGroupSize = 2,
          objNames = names(dvect), keep.data = TRUE, ..., verbose = FALSE,
          depth = 1)

pull(dmat, groups, index)

pullTab(dmat, groupTab, index)

```

**Arguments**

|              |  |
|--------------|--|
| dmat         | Square matrix of pairwise distances.   |
| groups       | Object coercible to a factor identifying group membership of objects corresponding to either edge of dmat.   |
| groupTab     | a data.frame representing a taxonomy, with columns in increasing order of specificity from left to right (ie, Kingdom → Species). Column names are used to name taxonomic ranks. Rows correspond to margins of dmat. |
| dvect        | numeric vector of distance from query sequence to each reference corresponding to margins of dmat.   |
| method       | The method for calculating the threshold; only 'mutinfo' is currently implemented.   |
| minScore     | Threshold value for the match score to define a match.   |
| doffset      | Offset used in the denominator of the expression to calculate match score to penalize very small groups of reference objects.  |
| dStart       | start with this value of D.  |
| multiple     | if TRUE, stops at the rank that yields at least one match; if FALSE, continues to perform classification until exactly one match is identified.  |
| maxDepth     | Maximum number of iterations that will be attempted to perform classification.   |
| minGroupSize | The minimal number of members comprising at least one group required to attempt classification.  |
| objNames     | Optional character identifiers for objects corresponding to margin of dmat.  |
| keep.data    | Populates thresh\$differences (see <a href="#">findThreshold</a> ) if TRUE.  |
| verbose      | Terminal output is produced if TRUE.   |
| index        | an integer specifying an element in dmat   |
| ...          | see Details  |
| depth        | specifies iteration number (not meant to be user-defined)  |

## Details

`classify` performs iterative classification. See the vignette vignette for package **clst** for a description of the classification algorithm.

`classifier` performs non-iterative classification, and is typically not called directly by the user.

The functions `pull` and `pullTab` are used to remove a single element of `dmat` for the purpose of performing classification against the remaining elements. The value of these two functions (a list) can be passed directly to `classify` or `classifyIter` directly (see examples).

## Value

`classify` and `classifyIter` return `x`, a list of lists, one for each iteration of the classifier. Each sub-list contains the following named elements:

|                      |   |
|----------------------|---|
| <code>depth</code>   | An integer indicating the number of the iteration (where <code>x[[i]]\$depth == i</code> )  |
| <code>tally</code>   | a <code>data.frame</code> with one row for each group or reference objects. Columns below and above contain counts of reference objects with distance values greater than or less than $D$ , respectively; <code>score</code> , containing match score $S$ ; <code>match</code> is 1 if $S \geq \text{minScore}$ , 0 otherwise; and the minimum, median, and maximum values of distances to all members of the indicated group. |
| <code>details</code> | a list of two matrices, named "below" and "above", itemizing each object with index $i$ in the reference set with distances below or above the distance threshold $D$ , respectively. Columns include <code>index</code> , the index $i$ ; <code>dist</code> , the distance between the object and the query; and <code>group</code> , indicating the classification of the object.   |
| <code>matches</code> | Character vector naming groups to which query object belongs.   |
| <code>thresh</code>  | object returned by <code>findThreshold</code>   |
| <code>params</code>  | a list of input arguments and their values  |
| <code>input</code>   | list containing copies of <code>dvect</code> and <code>groups</code>  |

## Author(s)

Noah Hoffman

## See Also

[findThreshold](#)

## Examples

```
## illustrate classification using the Iris data set
data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species

## remove one element from the data set and perform classification using
## the remaining elements as the reference set
ind <- 1
cat(paste('class of "unknown" sample is Iris',groups[ind]),fill=TRUE)
cc <- classify(dmat[-ind,-ind], groups[-ind], dvect=dmat[ind, -ind])
printClst(cc)

## this operation can be performed conveniently using the `pull` function
ind <- 51
```

```

cat(paste('class of "unknown" sample is Iris',groups[ind]),fill=TRUE)
cc <- do.call(classify, pull(dmat, groups, ind))
printClst(cc)
str(cc)

```

---

|               |                      |
|---------------|----------------------|
| findThreshold | <i>findThreshold</i> |
|---------------|----------------------|

---

## Description

Identify a distance threshold predicting whether a pairwise distance represents a comparison between objects in the same class (within-group comparison) or different classes (between-group comparison) given a matrix providing distances between objects and the group membership of each object.

## Usage

```

findThreshold(dmat, groups, distances, method = "mutinfo", prob = 0.5,
              na.rm = FALSE, keep.dists = TRUE, roundCuts = 2, minCuts =
              20, maxCuts = 300, targetCuts = 100, verbose = FALSE,
              depth = 1, ...)

```

```

partition(dmat, groups, include, verbose = FALSE)

```

## Arguments

|            |  |
|------------|--|
| dmat       | Square matrix of pairwise distances.   |
| groups     | Object coercible to a factor identifying group membership of objects corresponding to either edge of dmat.                                       |
| include    | vector (numeric or boolean) indicating which elements to retain in the output; comparisons including an excluded element will have a value of NA |
| distances  | Optional output of <a href="#">partition</a> provided in the place of dmat and groups  |
| method     | The method for calculating the threshold; only 'mutinfo' is currently implemented.   |
| prob       | Sets the upper and lower bounds of $D$ as some quantile of the within class distances and between-class differences, respectively.               |
| na.rm      | If TRUE, excludes NA elements in groups and corresponding rows and columns in dmat. Ignored if distances is provided.                            |
| keep.dists | If TRUE, the output will contain the distances element (output of <a href="#">partition</a> ).   |
| roundCuts  | Number of digits to round cutoff values (see <a href="#">Details</a> )   |
| minCuts    | Minimal length of vector of cutoffs (see <a href="#">Details</a> ).  |
| maxCuts    | Maximal length of vector of cutoffs (see <a href="#">Details</a> )   |
| targetCuts | Length of vector of cutoffs if conditions met by minCuts and maxCuts are not met (see <a href="#">Details</a> ).                                 |
| verbose    | Terminal output is produced if TRUE.   |
| depth      | Private argument used to track level of recursion.   |
| ...        | Extra arguments are ignored.   |

## Details

findThreshold is used internally in `classify`, but may also be used to calculate a starting value of `$D$`.

partition is used to transform a square (or lower triangular) distance matrix into a `data.frame` containing a column of distances (`$vals`) along with a factor (`$comparison`) defining each distance as a within- or between-group comparison. Columns `$row` and `$col` provide indices of corresponding rows and columns of `dmat`.

## Value

In the case of `findThreshold`, output is a list with elements described below. In the case of `partition`, output is the `data.frame` returned as the element named `$distances` in the output of `findThreshold`.

|                        |   |
|------------------------|---|
| <code>D</code>         | The distance threshold (distance cutoff corresponding to the PMMI).   |
| <code>pmmi</code>      | Value of the point of maximal mutual information (PMMI)   |
| <code>interval</code>  | A vector of length 2 indicating the upper and lower bounds over which values for the threshold are evaluated.   |
| <code>breaks</code>    | A <code>data.frame</code> with columns <code>x</code> and <code>y</code> providing candidate breakpoints and corresponding mutual information values, respectively. |
| <code>distances</code> | If <code>keep.distances</code> is <code>TRUE</code> , a <code>data.frame</code> containing pairwise distances identified as within- or between classes.             |
| <code>method</code>    | Character corresponding to input argument <code>method</code> .   |
| <code>params</code>    | Additional input parameters.  |

## Author(s)

Noah Hoffman

## See Also

[plotDistances](#), [plotMutinfo](#)

## Examples

```
data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species
thresh <- findThreshold(dmat, groups, type="mutinfo")
str(thresh)
```



---

plotDistances                      *Visualize results of link{findThreshold}*

---

### Description

The functions `plotDistances` and `plotMutinfo` are used to visualize the distance threshold calculated by `findThreshold` in the context of pairwise distances among objects in the reference set.

### Usage

```
plotDistances(distances, D = NA, interval = NA,
              ylab = "distances", ...)

plotMutinfo(breaks, D = NA, interval = NA,
            xlab = "distance", ylab = "mutual information", ...)
```

### Arguments

|                        |   |
|------------------------|---|
| <code>distances</code> | The <code>\$distances</code> element of the output value of <code>findThreshold</code>  |
| <code>breaks</code>    | The <code>\$breaks</code> element of the output value of <code>findThreshold</code>   |
| <code>D</code>         | The distance threshold  |
| <code>interval</code>  | The range of values over which candidate values of PMMI are evaluated.  |
| <code>xlab</code>      | Label the x axis of the plot.   |
| <code>ylab</code>      | Label the y axis of the plot.   |
| <code>...</code>       | Additional arguments are passed to <code>bwplot</code> ( <code>plotDistances</code> ) or <code>xyplot</code> ( <code>plotMutinfo</code> ) |

### Details

`plotDistances` produces a box-and-whisker plot contrasting within- and between-group distances. `plotMutinfo` produces a plot of cutpoints vs mutual information scores.

### Value

Returns a lattice grid object.

### Author(s)

Noah Hoffman

### See Also

[findThreshold](#)

### Examples

```
data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species
thresh <- findThreshold(dmat, groups)
do.call(plotDistances, thresh)
do.call(plotMutinfo, thresh)
```

---

|           |  |
|-----------|--|
| printClst | <i>Print a summary of the classifier output.</i> |
|-----------|--|

---

**Description**

Prints a description of the output of `classify`.

**Usage**

```
printClst(cc, rows = 8, nameWidth = 30, groupNames)
```

**Arguments**

|                         |  |
|-------------------------|--|
| <code>cc</code>         | Output of <code>classify</code>  |
| <code>rows</code>       | Number of rows corresponding to groups of reference objects to show.   |
| <code>nameWidth</code>  | Character width of group names.  |
| <code>groupNames</code> | a named vector containing replacement names for groups keyed by categories in groups ( <code>classify</code> ) or <code>groupTab</code> ( <code>classifyIter</code> ). |

**Value**

Output value is NULL; output is to stdout.

**Author(s)**

Noah Hoffman

**See Also**

[classify](#), [classifyIter](#)

**Examples**

```
data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species
```

---

|               |  |
|---------------|--|
| scaleDistPlot | <i>Annotated multidimensional scaling plots.</i> |
|---------------|--|

---

**Description**

Produces annotated representations of two-dimensional multidimensional scaling plots using `cmdscale`.

**Usage**

```
scaleDistPlot(dmat, groups, fill, X, O, indices = "no",
              include, display, labels,
              shuffleGlyphs = NA, key = "top",
              keyCols = 4, glyphs,
              xflip = FALSE, yflip = FALSE, ...)
```

**Arguments**

|               |  |
|---------------|--|
| dmat          | Square matrix of pairwise distances.   |
| groups        | Object coercible to a factor identifying group membership of objects corresponding to either edge of dmat. |
| fill          | vector (logical or indices) of points to fill  |
| X             | vector of points to mark with an X   |
| O             | vector of points to mark with a circle   |
| indices       | label points with indices (all points if 'yes', or a subset indicated by a vector)                         |
| include       | boolean or numeric vector of elements to include in call to cmdscale                                       |
| display       | boolean or numeric vector of elements to include in call to display  |
| labels        | list or data frame with parameters \$i indicating indices and \$text containing labels.                    |
| shuffleGlyphs | modify permutation of shapes and colors given an integer to serve as a random seed.                        |
| key           | 'right' (single column), 'top' (variable number of columns), or NULL for no key                            |
| keyCols       | number of columns in key   |
| glyphs        | a data.frame with columns named col and pch corresponding to elements of unique(groups)                    |
| xflip         | if TRUE, flip orientation of x-axis  |
| yflip         | if TRUE, flip orientation of y-axis  |
| ...           | additional arguments are passed to <a href="#">xyplot</a>  |

**Value**

Returns a lattice grid object.

**Author(s)**

Noah Hoffman

**See Also**

[cmdscale](#), [xyplot](#)

**Examples**

```

data(iris)
dmat <- as.matrix(dist(iris[,1:4], method="euclidean"))
groups <- iris$Species

## visualize pairwise euclidean distances among items in the Iris data set
fig <- scaleDistPlot(dmat, groups)
plot(fig)

## leave-one-out analysis of the classifier
loo <- lapply(seq_along(groups), function(i){
  do.call(classify, pull(dmat, groups, i))
})
matches <- lapply(loo, function(x) rev(x)[[1]]$matches)
result <- sapply(matches, paste, collapse='-')
confusion <- sapply(matches, length) > 1
no_match <- sapply(matches, length) < 1
plot(scaleDistPlot(dmat, groups, fill=confusion, O=confusion, X=no_match))

```

---

strep

*Streptococcus data set.*


---

**Description**

Square matrices describing pairwise distances among 16s rRNA sequences.

**Usage**

```
data(strep)
```

**Format**

```

List of 5
 $ dmat1 : num [1:150, 1:150] 0 5.81 8.38 10.28 10.64 ...
   ..- attr(*, "dimnames")=List of 2
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
 $ dmat2 : num [1:150, 1:150] 0 5.09 3.82 7.21 7.59 ...
   ..- attr(*, "dimnames")=List of 2
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
 $ dmat3 : num [1:150, 1:150] 0 5.63 5.81 8.77 9.14 ...
   ..- attr(*, "dimnames")=List of 2
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
   .. ..$ : chr [1:150] "197" "199" "207" "208" ...
 $ taxa : Factor w/ 50 levels "Streptococcus acidominimus",...: 31 44 26 4 4 31 32 39 42 31 ...
 $ abbrev: Factor w/ 50 levels "S acidominimus",...: 31 44 26 4 4 31 32 39 42 31 ...

```

**Details**

The matrices \$dmat1, dmat2, and dmat3 contain percent nucleotide difference with indels penalized heavily, little, and somewhat, respectively.

\$taxa is a factor of species names; abbreviations of the same names are found in \$abbrev.

*strep*

13

**Examples**

```
data(strep)
```

# Index

## \* **classif**

- classify, 4
- clst-package, 2
- findThreshold, 7
- plotDistances, 9

## \* **datasets**

- actino, 3
- bvseqs, 3
- strep, 12

## \* **package**

- clst-package, 2

actino, 3

bvseqs, 3

bwplot, 9

classifier (classify), 4

classify, 4, 8, 10

classifyIter, 10

classifyIter (classify), 4

clst (clst-package), 2

clst-package, 2

cmdscales, 2, 10, 11

findThreshold, 5, 6, 7, 9

partition, 7

partition (findThreshold), 7

plotDistances, 8, 9

plotMutinfo, 8

plotMutinfo (plotDistances), 9

printClst, 10

pull (classify), 4

pullTab (classify), 4

scaleDistPlot, 10

strep, 12

xyplot, 9, 11