Package 'leaderCluster'

May 8, 2026

Type Package

Title Leader Clustering Algorithm

Version 1.5

Date 2023-03-24

Author Taylor B. Arnold

Maintainer Taylor B. Arnold <tarnold2@richmond.edu>

Description The leader clustering algorithm provides
a means for clustering a set of data points. Unlike many other clustering
algorithms it does not require the user to specify the number of clusters,
but instead requires the approximate radius of a cluster as its primary
tuning parameter. The package provides a fast implementation of this
algorithm in n-dimensions using Lp-distances (with special cases for p=1,2,
and infinity) as well as for spatial data using the Haversine
formula, which takes latitude/longitude pairs as inputs and clusters
based on great circle distances.

License LGPL-2

LazyLoad yes

NeedsCompilation yes

RoxygenNote 7.2.3

Repository CRAN

Date/Publication 2023-03-24 18:30:02 UTC

Contents

`leaderCluster` *Calculate clusters using Hartigan's Leader Algorithm*

Description

Takes a matrix of coordinates and outputs cluster ids from running the leader algorithm. The coordinates can either be on points in the space R^n, or latitude/longitude pairs. A radius delta must be provided.

Usage

```
leaderCluster(
  points,
  radius,
  weights = rep(1, nrow(points)),
  max_iter = 10L,
  distance = c("Lp", "L1", "L2", "Linf", "haversine"),
  p = 2
)
```

Arguments

`points`	A matrix, or something which can be coerced into a matrix, of coordinates with rows representing points and columns representing dimensions. If using `haversine` distance, this must be a two column matrix with the first column containing latitudes in decimal degrees and the second containing longitudes in decimal degrees.
`radius`	A scalar value giving the radius of the resulting clusters; this is the main tuning parameter for the algorithm. When using the `haversine` distance this value should be in kilometres.
`weights`	An vector of weights, one per row of points, to apply to the clustering algorithm.
`max_iter`	Maximum number of times to iterate the algorithm; can safely set to 1 in many instances. See Details.
`distance`	The method to be used for calculating distances between points. If this is set to `haversine`, the `points` must be a two column matrix. If `Lp`, then the `p` input specifies which norm is being used.
`p`	When using `Lp` as the value for `distance`, this is a positive number specifing which Lp-norm to implement. For p equal to 1,2, or Inf, a special implementation will be used which is slightly more efficent than the more general application.

Details

The value for delta defines an approximate radius of each cluster. As the algorithm runs, a point within a distance delta from the centroid of a cluster will be labeled with the coorisponding cluster.

As centroid clusters move, it is possible for the final radius of each cluster to be slightly larger than delta.

Unlike many other iterative clustering algorithms, the leader algorithm typically provides reasonable clusters after just a single pass. When speed is of concern, the max_iter value may be safely set to 1. However, the algorithm typically fully converges in only a few cycles; also, a convergent solution will usually have a smaller number of clusters than a solution with only one pass.

The algorithm scales nicely, and can fit a model with 100s of columns and 100k's of rows in (on a relatively modest machine) under a minute. However, the processing time decays significantly if the radius is too small, since the number of clusters will be very high.

Value

A list containing a vector of cluster ids, a matrix of cluster centroids, the number of clusters, and the number iterations.

Author(s)

Taylor B. Arnold, <taylor.arnold@acm.org>

References

J. A. Hartigan. Clustering Algorithms. John Wiley & Sons, New York, 1975.

Examples

```
points <- 1:10
out <- leaderCluster(points, radius=2, distance="Lp", max_iter=1L)

par(mar = c(0,0,0,0), mfrow = c(1,3))
set.seed(1)
points <- matrix(runif(100*2), ncol=2)
for(r in c(0.1, 0.2, 0.4)) {
 out <- leaderCluster(points = points, radius = r, distance="L2")$cluster_id
 cols <- rainbow(length(unique(out)))[out]
 plot(points, pch = 19, cex = 0.7, col = cols, axes = FALSE)
 points(points[!duplicated(out),,drop=FALSE], cex = 2, col = unique(cols))
 box()
}
```

Index