Package 'meanShiftR'

July 22, 2025

Type Package

Title A Computationally Efficient Mean Shift Implementation

Version 0.56

Date 2021-09-12

URL http://meanmean.me/meanshift/r/cran/2016/08/28/meanShiftR.html

BugReports https://github.com/jlisic/meanShiftR/issues

Maintainer Jonathan Lisic <jlisic@gmail.com>

Description Performs mean shift classification using linear and
k-d tree based nearest neighbor implementations for the Gaussian,
Epanechnikov, and biweight product kernels.

Depends R (>= 3.1.0)

License GPL (>= 2)

RoxygenNote 6.0.1

NeedsCompilation yes

Author Jonathan Lisic [aut, cre]

Repository CRAN

Date/Publication 2021-09-21 05:00:09 UTC

Contents

knn_meanShift *K-d tree based k nearest neighbor search*

Description

knn_meanShift performs a search for the k nearest neighbors of a single point, where nearest is determined by the Mahalanobis distance. This search is performed through a k-d tree.

Usage

```
knn_meanShift(points, trainData, k = min(5, NROW(trainData)), weight,
  leafSize = 40, maxDist = Inf)
```

Arguments

points	n vectors stored in an n by p matrix. k nearest neighbors are found for each vector.
trainData	A matrix or vector of potential nearest neighbors.
k	A scalar indicating the number neighbors to find.
weight	A scalar or vector of length equal to the number of columns of trainData. This value is used as the diagonal elements for the inverse covariance matrix of the Mahalanobis distance.
leafSize	A scalar used to specify the number of points to store in the leaf nodes.
maxDist	A vector specifying the maximum value of the Mahalanobis that will be considered.

Value

A list is returned containing two items: neighbors, an n by k matrix of k indexes for each of the n vectors in points, corresponding to the nearest neighbors in trainData. value, a matrix of scalars containing the k distances between the neighbors found in trainData and points.

Examples

```
x <- matrix(runif(20),10,2)
neighbors <- knn_meanShift(c(0,0),x)
```

meanShift *Mean shift classification*

Description

meanShift performs classification of a set of query points using steepest ascent to local maxima in a kernel density estimate.

Usage

```
meanShift(queryData, trainData = queryData, nNeighbors = NROW(trainData),
  algorithm = "LINEAR", kernelType = "NORMAL", bandwidth = rep(1,
  NCOL(trainData)), alpha = 0, iterations = 10, epsilon = 1e-08,
  epsilonCluster = 1e-04, parameters = NULL)
```

Arguments

queryData	A matrix or vector of points to be classified by the mean shift algorithm. Values must be finite and non-missing.
trainData	A matrix or vector of points used to form a kernel density estimate. The local maxima from this kernel density estimate will be used for steepest ascent classification. If missing, queryData is set to trainData.
nNeighbors	A scalar indicating the number neighbors to consider for the kernel density estimate. This is useful to speed up approximation by approximating the kernel density estimate. The default is all data.
algorithm	A string indicating the algorithm to use for nearest neighbor searches. Currently, only "LINEAR" and "KDTREE" methods are supported.
kernelType	A string indicating the kernel associated with the kernel density estimate that the mean shift is optimizing over. The possible kernels are NORMAL, EPANECHNIKOV, and BIWEIGHT; the default is NORMAL.
bandwidth	A vector of length equal to the number of columns in the queryData matrix, or length one when queryData is a vector. This value will be used in the kernel density estimate for steepest ascent classification. The default is one for each dimension.
alpha	A scalar tuning parameter for normal kernels. When this parameter is set to zero, the mean shift algorithm will operate as usual. When this parameter is set to one, the mean shift algorithm will be approximated through Newton's Method. When set to a value between zero and one, a generalization of Newton's Method and mean shift will be used instead providing a means to balance convergence speed with stability. The default is zero, mean shift.
iterations	The number of iterations to perform mean shift.
epsilon	A scalar used to determine when to terminate the iteration of a individual query point. If the distance between the query point at iteration i and i+1 is less than epsilon, then iteration ceases on this point.

`epsilonCluster` A scalar used to determine the minimum distance between distinct clusters. This distance is applied after all iterations have finished and in order of the rows of `queryData`.

`parameters` A scalar or vector of paramters used by the specific algorithm. There are no optional parameters for the "LINEAR" method, "KDTREE" supports optional parameters for the maximum number of points to store in a leaf node and the maximum value for the quadratic form in the normal kernel, ignoring the constant value -0.5.

Value

A list is returned containing two items: `assignment`, a vector of classifications. `value`, a vector or matrix containing the location of the classified local maxima in the support, each row is associated with the classified index in `assignment`.

References

Cheng, Y. (1995). *Mean shift, mode seeking, and clustering*. IEEE transactions on pattern analysis and machine intelligence, 17(8), 790-799.

Fukunaga, K., & Hostetler, L. (1975). *The estimation of the gradient of a density function, with applications in pattern recognition.* IEEE transactions on information theory, 21(1), 32-40.

Lisic, J. (2015). Parcel Level Agricultural Land Cover Prediction (Doctoral dissertation, George Mason University).

Examples

```
x <- matrix(runif(20),10,2)
classification <- meanShift(x,x)

x <- matrix(runif(20),10,2)
classification <- meanShift(x,
algorithm="KDTREE",
nNeighbor=8,
parameters=c(5,7.1) )
```

Index