Package: ktaucenters (via r-universe)

September 17, 2024

Type Package

Title Robust Clustering Procedures

Version 1.0.0

Description A clustering algorithm similar to K-Means is implemented, it has two main advantages, namely (a) The estimator is resistant to outliers, that means that results of estimator are still correct when there are atypical values in the sample and (b) The estimator is efficient, roughly speaking, if there are no outliers in the sample, results will be similar to those obtained by a classic algorithm (K-Means). Clustering procedure is carried out by minimizing the overall robust scale so-called tau scale. (see Gonzalez, Yohai and Zamar (2019) <arxiv:1906.08198>).

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 2.10), MASS, stats, GSE

Language en-US

LazyData true

RoxygenNote 7.2.3

Suggests jpeg, tclust, knitr, rmarkdown, testthat (>= 3.1.0)

Imports Rcpp (>= 1.0.9)

LinkingTo Rcpp

Config/testthat/edition 3

NeedsCompilation yes

Repository https://jdgonzalezwork.r-universe.dev

RemoteUrl https://github.com/jdgonzalezwork/ktaucenters

RemoteRef HEAD

RemoteSha e275af4830cffbb9947dabf2d563a96627d329e6

Contents

`.distance` *Distance Matrix Computation*

Description

Computes and returns the distance matrix using euclidean distance measure to compute the distances between the rows of a data matrix.

Usage

```
.distance(x)
```

Arguments

`x`	a numeric matrix.

Value

A numeric matrix with the distances between the rows of a matrix.

`.flag_outliers` *Flag outliers*

Description

Flag outliers

Usage

```
.flag_outliers(cutoff, b, ktau)
```

Arguments

`cutoff`	quantile of chi-square to be used as a threshold for outliers detection.
`b`	break down point.
`ktau`	ktaucenters results.

Value

Numeric vector with the weight factor for each observation

`.ktaucenters_run` *Robust Clustering algorithm based on centers, a robust and efficient version of kmeans.*

Description

Robust Clustering algorithm based on centers, a robust and efficient version of kmeans.

Usage

```
.ktaucenters_run(x, centers, tolerance, max_iter)
```

Arguments

`x`	numeric matrix of size n x p with all observations.
`centers`	numeric matrix with initial cluster centers.
`tolerance`	maximum difference between current and new computed clusters. Parameter used for the algorithm stopping rule.
`max_iter`	a maximum number of iterations used for the algorithm stopping rule.

Value

A list with the following components:

`tau`	τ scale value.
`iter`	number of iterations until convergence is achieved or maximum number of iteration is reached.
`di`	distance of each observation to its nearest cluster center.
`centers`	numeric matrix of size K x p, with the estimated K centers.
`clusters`	integer vector of size n with the cluster location for each observation.

References

[1] Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

[2] Maronna, R. A. and Yohai, V. J. (2017). Robust and efficient estimation of multivariate scatter and location.Computational Statistics &Data Analysis, 109 : 64–75.

`derpsiOpt` *Second derivative of the quasi ρ function*

Description

Second derivative of the quasi ρ function

Usage

```
derpsiOpt(x, cc)
```

Arguments

`x`	numeric vector with positive values.
`cc`	tunning constant.

Value

Numeric vector with the second derivative of the quasi optimal ρ computation for each element of x.

`improvedktaucenters` *improvedktaucenters*

Description

Robust Clustering algorithm for non-spherical data. This function estimate clusters taking into account that clusters may have different size, volume or orientation.

Usage

```
improvedktaucenters(X, K, cutoff = 0.999, nstart = 5, INITcenters = NULL)
```

Arguments

`X`	numeric matrix of size n x p.
`K`	number of clusters.
`cutoff`	argument for outliers detection - quantiles of chi-square to be used as a threshold for outliers detection, defaults to 0.999.
`nstart`	number of trials that the base ktaucenters is run at the first stage. If it is greater than 1 and center is not set as NULL, a random set of (distinct) rows in x is chosen as the initial centres for each trial.
`INITcenters`	numeric matrix of size K x p indicating the initial centers for that clusters and robust covariance matrices will be computed, if it is set as NULL the algorithm will compute from ktaucenters routine. Set to NULL by default.

Value

A list with the following components:

`centers`	: Matrix of size K x p, with the estimated K centers.
`cluster`	: A vector of integer (from 1:k) indicating the cluster to which each point is allocated.
`sigmas`	: A list containing the k covariance matrices found by the procedure at its second step.
`outliers`	: indices observation that can be considered as outliers.

References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

Examples

```
# Generate synthetic data (three normal cluster in two dimensions)
# Clusters have different shapes and orientation.
# The data is contaminated uniformly (level 20%).

# Generates base clusters
set.seed(1)
Z1 <- c(rnorm(100, 0), rnorm(100, 0), rnorm(100, 0))
Z2 <- rnorm(300)
X <- matrix(0, ncol = 2, nrow = 300)
X[, 1] <- Z1
X[, 2] <- Z2
true.cluster <- c(rep(1, 100), rep(2, 100), rep(3, 100))

# Rotate, expand and translate base clusters
theta <- pi/3
aux1 <- matrix(c(cos(theta), -sin(theta), sin(theta), cos(theta)), nrow = 2)
aux2 <- sqrt(4) * diag(c(1, 1/4))
B <- aux1 %*% aux2 %*% t(aux1)
X[true.cluster == 3, ] <-
  X[true.cluster == 3, ] %*% aux2 %*% aux1 + matrix(c(5, 2),
                                                   byrow = TRUE,
                                                   nrow = 100,
                                                   ncol = 2)
X[true.cluster == 2, 2] <- X[true.cluster == 2, 2] * 5
X[true.cluster == 1, 2] <- X[true.cluster == 1, 2] * 0.1
X[true.cluster == 1, ] <- X[true.cluster == 1, ] + matrix(c(-5, -1),
                                                           byrow = TRUE,
                                                           nrow = 100,
                                                           ncol = 2)

# Generate 60 synthetic outliers (contamination level 20%)

outliers <- sample(1:300, 60)
X[outliers, ] <- matrix(runif( 40, 2 * min(X), 2 * max(X) ),
                                ncol = 2, nrow = 60)

# Applying the algorithm
robust <- improvedktaucenters(X, K = 3, cutoff = 0.999)

# Plotting results
oldpar <- par(mfrow = c(2, 1))
plot(X, main = "Actual clusters")
for (j in 1:3){
 points(X[true.cluster == j, ], pch = 19, col = j + 1)
}
points(X[outliers, ], pch = 19, col = 1)
plot(X, main = "Clusters estimation")
for (j in 1:3){
 points(X[robust$cluster == j,], pch = 19, col = j + 1)
}
points(X[robust$outliers, ], pch = 19)
```

```
par(oldpar)
```

`ktaucenters` *ktaucenters*

Description

Robust and efficient version of Kmeans algorithm for clustering based on centers.

Usage

```
ktaucenters(
  X,
  K,
  centers = NULL,
  tolmin = 1e-06,
  NiterMax = 100,
  nstart = 1,
  startWithKmeans = TRUE,
  startWithROBINPD = TRUE,
  cutoff = 0.999
)
```

Arguments

`X`	numeric matrix of size n x p.
`K`	number of clusters.
`centers`	a matrix of size K x p containing the K initial centers, one at each matrix-row. If centers is NULL a random set of (distinct) rows in `X` are chosen as the initial centers.
`tolmin`	a tolerance parameter used for the algorithm stopping rule.
`NiterMax`	a maximum number of iterations used for the algorithm stopping rule.
`nstart`	the number of trials that the base algorithm is run. If it is greater than 1 and centers is not set as NULL, a random set of (distinct) rows in `X` will be chosen as the initial centers.
`startWithKmeans`	if positive (or true) kmeans estimated centers are included as starting point.
`startWithROBINPD`	if positive (or true) ROBINDEN estimated centers are included as starting point.
`cutoff`	optional argument for outliers detection - quantiles of chi-square to be used as a threshold for outliers detection, defaults to 0.999.

Value

A list with the following components:

`centers`	: Matrix of size K x p with the estimated K centers.
`cluster`	: A vector of integer (from 1:K) indicating the cluster to which each point is allocated.
`iter`	: Number of iterations until convergence is achieved or maximum number of iterations reached.
`di`	: Distance of each observation to its assigned cluster-center.
`outliers`	: A vector of integers with indices for each observation considered as outlier.

References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

Examples

```
# Generate synthetic data (three clusters well separated)
Z <- rnorm(600)
mues <- rep(c(-3, 0, 3), 200)
X <- matrix(Z + mues, ncol = 2)

# Generate 60 synthetic outliers (contamination level 20%)
X[sample(1:300,60), ] <- matrix(runif( 40, 3 * min(X), 3 * max(X) ),
                                ncol = 2, nrow = 60)

robust <- ktaucenters(
     X, K = 3, centers = X[sample(1:300, 3), ],
     tolmin = 1e-3, NiterMax = 100)

oldpar <- par(mfrow = c(1, 2))

plot(X,type = "n", main = "ktaucenters (Robust) \n outliers: solid black dots")
points(X[robust$cluster == 1, ], col = 2)
points(X[robust$cluster == 2, ], col = 3)
points(X[robust$cluster == 3, ], col = 4)
points(X[robust$outliers, 1], X[robust$outliers, 2], pch = 19)

# Classical (non Robust) algorithm
non_robust <- kmeans(X, centers = 3, nstart = 100)

plot(X, type = "n", main = "kmeans (Classical)")
points(X[non_robust$cluster == 1, ], col = 2)
points(X[non_robust$cluster == 2, ], col = 3)
points(X[non_robust$cluster == 3, ], col = 4)

par(oldpar)
```

`ktaucentersfast` *ktaucentersfast*

Description

Robust and efficient version of Kmeans algorithm for clustering based on centers.

Usage

```
ktaucentersfast(
  x,
  centers,
  nstart = 1L,
  use_kmeans = TRUE,
  use_robin = TRUE,
  max_iter = 100L,
  max_tol = 1e-06,
  cutoff = 0.999
)
```

Arguments

`x`	numeric matrix of size n x p, or an object that can be coerced to a matrix (such as a numeric vector or a data frame with all numeric columns).
`centers`	either the number of clusters, say **k**, or a matrix of initial (distinct) cluster centers. If a number, a random set of distinct rows in `x` is chosen as the initial centers.
`nstart`	if centers is a number, how many random sets should be chosen?
`use_kmeans`	use kmeans centers as starting point?
`use_robin`	use robin algorithm centers as starting point?
`max_iter`	the maximum number of iterations allowed.
`max_tol`	maximum tolerance parameter used for the algorithm as stopping rule.
`cutoff`	quantile of chi-square distribution to be used as a threshold for outliers detection, defaults to 0.999.

Value

A list with the following components:

`centers`	: A matrix of cluster centers.
`cluster`	: A vector of integer (from 1:k) indicating the cluster to which each point is allocated.
`tau`	: τ scale value.
`iter`	: Number of iterations until convergence is achieved or maximum number of iteration reached.

`di`	: Distance of each observation to its assigned cluster-center
`outliers`	: A vector of integers with indices for each observation considered as outlier.

References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

Examples

```
# Generate synthetic data (three clusters well separated)
Z <- rnorm(600)
mues <- rep(c(-3, 0, 3), 200)
X <- matrix(Z + mues, ncol = 2)

# Generate 60 synthetic outliers (contamination level 20%)
X[sample(1:300,60), ] <- matrix(runif( 40, 3 * min(X), 3 * max(X) ),
                                ncol = 2, nrow = 60)

robust <- ktaucentersfast(
     X, centers = X[sample(1:300, 3), ],
     max_tol = 1e-3, max_iter = 100)

oldpar <- par(mfrow = c(1, 2))

plot(X,type = "n", main = "ktaucenters (Robust) \n outliers: solid black dots")
points(X[robust$cluster == 1, ], col = 2)
points(X[robust$cluster == 2, ], col = 3)
points(X[robust$cluster == 3, ], col = 4)
points(X[robust$outliers, 1], X[robust$outliers, 2], pch = 19)

# Classical (non Robust) algorithm
non_robust <- kmeans(X, centers = 3, nstart = 100)

plot(X, type = "n", main = "kmeans (Classical)")
points(X[non_robust$cluster == 1, ], col = 2)
points(X[non_robust$cluster == 2, ], col = 3)
points(X[non_robust$cluster == 3, ], col = 4)

par(oldpar)
```

`mars_screw` *Intensity and saturation values of a picture from mars.*

Description

A dataset containing the Intensity and Saturation values of a picture from Mars taken from Rover Curiosity.

Usage

```
mars_screw
```

Format

A list containing information about pixels of a picture form mars mainly containing red sand and metal form Rover itself. List include:

- SI_matrix: A matrix with 5063 rows and 128 columns. Elements 1 to 64 of each row indicate the Saturation values of pixels in a square cell 8 x 8 whereas elements 65 to 128 of each row indicate the cell's Intensity values.
- geographic_matrix: An integer matrix of dimension 5063 x 2, each row indicates each square cell's locations (x-axis y-axis) at the picture.
- screw_index: the index corresponding to the screw observation (screw_index=4180)

Source

https://www.nasa.gov/wp-content/uploads/2023/03/694811main_pia16225-43_full.jpg

---

`Mscale` *M scale*

Description

The M scale of an univariate sample.

Usage

```
Mscale(u, c, b)
```

Arguments

u	numeric vector with positive values.
c	a tuning constant. If consistency to standard normal distribution is desired use `normal_consistency_constants`.
b	the desired break down point.

Value

M scale value.

References

Maronna, R. A., Martin, R. D., Yohai, V. J., & Salibian-Barrera, M. (2018). Robust statistics: theory and methods (with R). Wiley.

Examples

```
Mscale(u = rnorm(100), c = 1, b = 0.5)
```

`normal_consistency_constants` *Normal Consistency Constants*

Description

M scale tuning constants so it is consistent with the standard normal distribution for the quasi optimal ρ function used in `rhoOpt`. These constants were computed for $1 \leq$ p ≤ 400.

Usage

```
normal_consistency_constants(p)
```

Arguments

`p`	dimension where observation lives.

Value

tuning constant.

References

[1] Maronna, R. A., Martin, R. D., Yohai, V. J., & Salibián-Barrera, M. (2018). 'Robust statistics: theory and methods (with ' R). Wiley.

[2] Salibian-Barrera, M., Willems, G., & Zamar, R. (2008). The fast-tau estimator for regression. 'Journal of Computational and Graphical Statistics, 17(3), 659-682.

`psiOpt` *Derivative of the quasi optimal ρ function*

Description

Derivative of the quasi optimal ρ function

Usage

```
psiOpt(x, cc)
```

Arguments

`x`	numeric vector with positive values.
`cc`	tunning constant.

Value

Numeric vector with the derivative of the quasi optimal ρ computation for each element of x.

`rhoOpt` *Quasi optimal ρ function*

Description

Quasi optimal ρ function

Usage

```
rhoOpt(x, cc)
```

Arguments

`x`	numeric vector with positive values.
`cc`	tunning constant.

Value

Numeric vector with quasi optimal ρ computation for each element of x.

References

[1] Salibian-Barrera, M., Willems, G., & Zamar, R. (2008). The fast-tau estimator for regression. Journal of Computational and GraphicalStatistics, 17(3), 659-682.

`robinden` *Robust Initialization based on Inverse Density estimator (ROBINDEN)*

Description

Searches for k initial cluster seeds for k-means based clustering methods.

Usage

```
robinden(D, n_clusters, mp)
```

Arguments

D	a distance matrix, which contains the distances between the rows of a matrix.
n_clusters	number of cluster centers to find.
mp	number of nearest neighbors to compute point density.

Details

The centers are the observations located in the most dense region and far away from each other at the same time. In order to find the observations in the highly dense region, this function uses point density estimation (instead of Local Outlier Factor, Breunig et al (2000)), see more details.

Value

A list with the following components:

centers	: A numeric vector with the initial cluster centers indices.
idpoints	: A real vector containing the inverse of point density estimation.

Note

This is a slightly modified version of ROBIN algorithm implementation done by Sarka Brodinova <sarka.brodinova@tuwien.ac.at>.

Author(s)

Juan Domingo Gonzalez <juanrst@hotmail.com>

References

Hasan AM, et al. Robust partitional clustering by outlier and density insensitive seeding. Pattern Recognition Letters, 30(11), 994-1002, 2009.

Examples

```
# Generate synthetic data (7 cluster well separated)
K <- 5
nk <- 100
Z <- rnorm(2 * K * nk)
mues <- rep(5 * -floor(K/2):floor(K/2), 2 * nk * K)
X <-  matrix(Z + mues, ncol = 2)

# Generate synthetic outliers (contamination level 20%)
X[sample(1:(nk * K), (nk * K) * 0.2), ] <-
  matrix(runif((nk * K) * 0.2 * 2, 3 * min(X), 3 * max(X)),
         ncol = 2,
         nrow = (nk * K)* 0.2)
res <- robinden(D = as.matrix(dist(X)), n_clusters = K, mp = 10);
# plot the Initial centers found
plot(X)
points(X[res$centers, ], pch = 19, col = 4, cex = 2)
```

Index