Package 'streamMOA'

May 10, 2022

Version 1.2-4

Date 2022-05-09

Encoding UTF-8

Title Interface for MOA Stream Clustering Algorithms

Description Interface for data stream clustering algorithms implemented in the MOA (Massive Online Analysis) framework (Albert Bifet, Geoff Holmes, Richard Kirkby, Bernhard Pfahringer (2010). MOA: Massive Online Analysis, Journal of Machine Learning Research 11: 1601-1604).

Depends stream (>= 1.5-1), rJava (>= 0.9-0)

Imports graphics, stats, methods

SystemRequirements Java (>= 8)

BugReports https://github.com/mhahsler/streamMOA

License GPL-3

Copyright MOA code in inst/java/moa.jar is Copyright (C) The University of Waikato and distributed under the Apache License, version 2.0. All other code is Copyright (C) Matthew Bolanos, John Forrest and Michael Hahsler

RoxygenNote 7.1.2

NeedsCompilation no

Author Michael Hahsler [aut, cre, cph],
John Forrest [aut, cph],
Matthew Bolanos [ctb],
Matthias Carnein [ctb],
Dalibor Krleža [ctb]

Maintainer Michael Hahsler <mhahsler@lyle.smu.edu>

Repository CRAN

Date/Publication 2022-05-10 17:30:02 UTC

R topics documented:

`DSC_BICO_MOA` *BICO - Fast computation of k-means coresets in a data stream*

Description

This is an interface to the MOA implementation of BICO. The original BICO implementation by Fichtenberger et al is also available as `DSC_BICO`.

Usage

```
DSC_BICO_MOA(
  Cluster = 5,
  Dimensions,
  MaxClusterFeatures = 1000,
  Projections = 10,
  k = NULL,
  space = NULL,
  p = NULL
)
```

Arguments

`Cluster, k`	Number of desired centers
`Dimensions`	The number of the dimensions of the input points (stream) need to be specified in advance
`MaxClusterFeatures, space`	Maximum size of the coreset
`Projections, p`	Number of random projections used for the nearest neighbour search

Details

BICO maintains a tree which is inspired by the clustering tree of BIRCH, a SIGMOD Test of Time award-winning clustering algorithm. Each node in the tree represents a subset of these points. Instead of storing all points as individual objects, only the number of points, the sum and the squared sum of the subset's points are stored as key features of each subset. Points are inserted into exactly one node.

Author(s)

Matthias Carnein

References

Hendrik Fichtenberger, Marc Gille, Melanie Schmidt, Chris Schwiegelshohn, Christian Sohler: BICO: BIRCH Meets Coresets for k-Means Clustering. ESA 2013: 481-492

Examples

```
# data with 3 clusters and 2 dimensions
stream <- DSD_Gaussians(k=3, d=2)

# cluster with BICO
bico <- DSC_BICO_MOA(Cluster=3, Dimensions=2)
update(bico, stream, 10000)
bico

# plot micro and macro-clusters
plot(bico, stream, type="both")
```

DSC_CluStream *CluStream Data Stream Clusterer*

Description

Class implements the CluStream cluster algorithm for data streams (Aggarwal et al, 2003).

Usage

```
DSC_CluStream(m = 100, horizon = 1000, t = 2, k = NULL)
```

Arguments

m	Defines the maximum number of micro-clusters used in CluStream
horizon	Defines the time window to be used in CluStream

t	Maximal boundary factor (=Kernel radius factor). When deciding to add a new data point to a micro-cluster, the maximum boundary is defined as a factor of t of the RMS deviation of the data points in the micro-cluster from the centroid.
k	Number of macro-clusters to produce using weighted k-means. NULL disables automatic reclustering.

Details

This is an interface to the MOA implementation of CluStream.

If k is specified, then CluStream applies a weighted k-means algorithm for reclustering (see Examples section below).

Value

An object of class DSC_CluStream (subclass of DSC_Micro, DSC_MOA and DSC), or, if k is not NULL then an object of DSC_TwoStage.

Author(s)

Michael Hahsler and John Forrest

References

Aggarwal CC, Han J, Wang J, Yu PS (2003). "A Framework for Clustering Evolving Data Streams." In "Proceedings of the International Conference on Very Large Data Bases (VLDB '03)," pp. 81-92.

Bifet A, Holmes G, Pfahringer B, Kranen P, Kremer H, Jansen T, Seidl T (2010). MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering. In Journal of Machine Learning Research (JMLR).

See Also

DSC, DSC_Micro, DSC_MOA

Examples

```
# data with 3 clusters and 5% noise
stream <- DSD_Gaussians(k=3, d=2, noise=.05)

# cluster with CluStream
clustream <- DSC_CluStream(m=50)
update(clustream, stream, 500)
clustream

# plot micro-clusters
plot(clustream, stream)

# plot assignment area (micro-cluster radius)
plot(clustream, stream, assignment=TRUE, weights=FALSE)
```

```
# reclustering. Use weighted k-means for CluStream
kmeans <- DSC_Kmeans(k=3, weighted=TRUE)
recluster(kmeans, clustream)
plot(kmeans, stream, type="both")

# use k-means reclustering automatically by specifying k
clustream <- DSC_CluStream(m=50, k=3)
update(clustream, stream, 500)
clustream

plot(clustream, stream, type="both")
```

DSC_ClusTree *ClusTree Data Stream Clusterer*

Description

Interface for the MOA implementation of the ClusTree data stream clustering algorithm (Kranen et al, 2009).

Usage

```
DSC_ClusTree(horizon = 1000, maxHeight = 8, lambda = NULL, k = NULL)
```

Arguments

horizon	Range of the (time) window.
maxHeight	The maximum height of the tree.
lambda	number used to override computed lambda (decay).
k	If specified, k-means with k clusters is used for reclustering.

Details

ClusTree uses a compact and self-adaptive index structure for maintaining stream summaries. Kranen et al (2009) suggest EM or k-means for reclustering.

Value

An object of class DSC_ClusTree (subclass of DSC, DSC_MOA, DSC_Micro).

Author(s)

Michael Hahsler and John Forrest

References

Philipp Kranen, Ira Assent, Corinna Baldauf, and Thomas Seidl. 2009. Self-Adaptive Anytime Stream Clustering. In Proceedings of the 2009 Ninth IEEE International Conference on Data Mining (ICDM '09). IEEE Computer Society, Washington, DC, USA, 249-258. doi:10.1109/ICDM.2009.47

Bifet A, Holmes G, Pfahringer B, Kranen P, Kremer H, Jansen T, Seidl T (2010). MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering. In Journal of Machine Learning Research (JMLR).

See Also

DSC, DSC_Micro, DSC_MOA

Examples

```
# data with 3 clusters and 5% noise
stream <- DSD_Gaussians(k=3, d=2, noise=0.05)

# Use automatically the k-means reclusterer with k=3 to create macro clusters
clustree <- DSC_ClusTree(maxHeight=3, k = 3)
update(clustree, stream, 500)
clustree

# plot micro-clusters
plot(clustree, stream, , type = "both")

# create a two stage clustering using ClusTree and reachability reclustering
CTxReach <- DSC_TwoStage(
  micro=DSC_ClusTree(maxHeight=3),
  macro=DSC_Reachability(epsilon = .15)
)
CTxReach
update(CTxReach, stream, 1000)
plot(CTxReach, stream, type = "both")
```

DSC_DenStream *DenStream Data Stream Clusterer*

Description

Interface for the DenStream cluster algorithm for data streams implemented in MOA.

Usage

```
DSC_DenStream(
  epsilon,
  mu = 1,
  beta = 0.2,
  lambda = 0.001,
  initPoints = 100,
  offline = 2,
  processingSpeed = 1,
  recluster = TRUE,
  k = NULL
)
```

Arguments

epsilon	defines the epsilon neighbourhood which is the maximal radius of micro-clusters (r<=epsilon). Range: 0 to 1.
mu	minpoints as the weight w a core-micro-clusters needs to be created (w>=mu). Range: 0 to max(int).
beta	multiplier for mu to detect outlier micro-clusters given their weight w (w<beta x mu). Range: 0 to 1
lambda	decay constant.
initPoints	number of points to use for initialization via DBSCAN.
offline	offline multiplier for epsilon. Range: between 2 and 20). Used for reachability reclustering
processingSpeed	Number of incoming points per time unit (important for decay). Range: between 1 and 1000.
recluster	logical; should the offline DBSCAN-based (i.e., reachability at a distance of epsilon) be performed?
k	integer; tries to automatically chooses offline to find k macro-clusters.

Details

DenStream applies reachbility (from DBSCAN) between micro-clusters for reclustering using `epsilon` x `offline` (defaults to 2) as the reachability threshold.

If `k` is specified it automatically chooses the reachability threshold to find k clusters. This is achieved using single-link hierarchical clustering.

Value

An object of class `DSC_DenStream` (subclass of `DSC`, `DSC_MOA`, `DSC_Micro`) or, for `recluster=TRUE`, an object of class `DSC_TwoStage`.

Author(s)

Michael Hahsler and John Forrest

References

Cao F, Ester M, Qian W, Zhou A (2006). Density-Based Clustering over an Evolving Data Stream with Noise. In Proceedings of the 2006 SIAM International Conference on Data Mining, pp 326-337. SIAM.

Bifet A, Holmes G, Pfahringer B, Kranen P, Kremer H, Jansen T, Seidl T (2010). MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering. In Journal of Machine Learning Research (JMLR).

See Also

DSC, DSC_Micro, DSC_MOA

Examples

```
# data with 3 clusters and 5% noise
stream <- DSD_Gaussians(k = 3, d = 2, noise = 0.05)

# use Den-Stream with reachability reclustering
denstream <- DSC_DenStream(epsilon = .05)
update(denstream, stream, 500)
denstream

# plot macro-clusters
plot(denstream, stream)

# plot micro-cluster
plot(denstream, stream, type = "micro")

# show micro and macro-clusters
plot(denstream, stream, type = "both")

# reclustering. Choose reclustering reachability threshold automatically to find 3 clusters
denstream2 <- DSC_DenStream(epsilon = .05, k = 3)
update(denstream2, stream, 500)
plot(denstream2, stream, type = "both")
```

DSC_DStream_MOA *D-Stream Data Stream Clustering Algorithm*

Description

This is an interface to the MOA implementation of D-Stream. A C++ implementation (including reclustering with attraction) is available as DSC_DStream.

Usage

```
DSC_DStream_MOA(decayFactor = 0.998, Cm = 3, Cl = 0.8, Beta = 0.3)
```

Arguments

decayFactor	The decay factor
Cm	Controls the threshold for dense grids
Cl	Controls the threshold for sparse grids
Beta	Adjusts the window of protection for renaming previously deleted grids as sporadic

Details

D-Stream creates an equally spaced grid and estimates the density in each grid cell using the count of points falling in the cells. Grid cells are classified based on density into dense, transitional and sporadic cells. The density is faded after every new point by a decay factor.

Note: The MOA implementation of D-Stream currently does not return micro clusters.

Author(s)

Matthias Carnein

References

Yixin Chen and Li Tu. 2007. Density-based clustering for real-time stream data. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07). ACM, New York, NY, USA, 133-142.

Li Tu and Yixin Chen. 2009. Stream data clustering based on grid density and attraction. ACM Transactions on Knowledge Discovery from Data, 3(3), Article 12 (July 2009), 27 pages.

Examples

```
# data with 2 clusters in 2 dimensions
stream = DSD_Gaussians(2,2, mu = rbind(c(-10,-10), c(10,10)))

# cluster with D-Stream
dstream <- DSC_DStream_MOA(decayFactor=0.998)
update(dstream, stream, 10000)
dstream

# plot macro-clusters
plot(dstream, stream, type= "macro")
```

DSC_MOA *DSC_MOA Class*

Description

An abstract class that inherits from the base class DSC and provides the common functions needed to interface MOA clusterers.

Usage

```
DSC_MOA(...)
```

Arguments

...	further arguments.

Details

DSC_MOA classes operate in a different way in that the centers of the micro-clusters have to be extracted from the underlying Java object. This is done by using rJava to perform method calls directly in the JRI and converting the multi-dimensional Java array into a local R data type.

Author(s)

Michael Hahsler and John Forrest

References

Albert Bifet, Geoff Holmes, Richard Kirkby, Bernhard Pfahringer (2010). MOA: Massive Online Analysis, Journal of Machine Learning Research 11: 1601-1604

See Also

DSC

DSC_StreamKM *streamKM++*

Description

This is an interface to the MOA implementation of streamKM++.

Usage

```
DSC_StreamKM(sizeCoreset = 10000, numClusters = 5, length = 100000L)
```

Arguments

sizeCoreset	Size of the coreset
numClusters	Number of clusters to compute
length	Length of the data stream

Details

streamKM++ uses a tree-based sampling strategy to obtain a small weighted sample of the stream called coreset. Upon reclustering, the algorithm applies the k-means++ algorithm to find a given number of centres in the coreset.

Author(s)

Matthias Carnein

References

Marcel R. Ackermann, Christiane Lammersen, Marcus Maertens, Christoph Raupach, Christian Sohler, Kamil Swierkot. "StreamKM++: A Clustering Algorithm for Data Streams." In: Proceedings of the 12th Workshop on Algorithm Engineering and Experiments (ALENEX '10), 2010

Examples

```
# data with 3 clusters
stream <- DSD_Gaussians(k=3, d=2)

# cluster with streamKM++
streamkm <- DSC_StreamKM(sizeCoreset=10000, numClusters=3, length=10000)
update(streamkm, stream, 10000)
streamkm

# plot macro-clusters
plot(streamkm, stream, type="macro")
```

DSD_RandomRBFGeneratorEvents

Random RBF Generator Events Data Stream Generator

Description

A class that generates random data based on RandomRBFGeneratorEvents implemented in MOA.

Usage

```
DSD_RandomRBFGeneratorEvents(
  k = 3,
  d = 2,
  numClusterRange = 3L,
  kernelRadius = 0.07,
  kernelRadiusRange = 0,
  densityRange = 0,
  speed = 100L,
  speedRange = 0L,
  noiseLevel = 0.1,
  noiseInCluster = FALSE,
  eventFrequency = 30000L,
  eventMergeSplitOption = FALSE,
  eventDeleteCreate = FALSE,
  modelSeed = NULL,
  instanceSeed = NULL
)
```

Arguments

k	The average number of centroids in the model.
d	The dimensionality of the data.
numClusterRange	Range for numner of clusters.
kernelRadius	The average radius of the micro-clusters.
kernelRadiusRange	Deviation of the number of centroids in the model.
densityRange	Density range.
speed	Kernels move a predefined distance of 0.01 every X points.
speedRange	Speed/Velocity point offset.
noiseLevel	Noise level.
noiseInCluster	Allow noise to be placed within a cluster.
eventFrequency	Frequency of events.
eventMergeSplitOption	Merge and split?
eventDeleteCreate	Delete and create?
modelSeed	Random seed for the model.
instanceSeed	Random seed for the instances.

Details

There are an assortment of parameters available for the underlying MOA data structure, however, we have currently limited the available parameters to the arguments above. Currently the modelSeed and instanceSeed are set to default values every time a DSD_MOA is created, therefore the generated data will be the same. Because of this, it is important to set the seed manually when different data is needed.

The default behavior is to create a data stream with 3 clusters and concept drift. The locations of the clusters will change slightly, and they will merge with one another as time progresses.

Value

An object of class DSD_RandomRBFGeneratorEvent (subclass of DSD_MOA, DSD).

Author(s)

Michael Hahsler and John Forrest

References

MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, Thomas Seidl. Journal of Machine Learning Research (JMLR).

See Also

DSD

Examples

```
stream <- DSD_RandomRBFGeneratorEvents()
get_points(stream, 10, class=TRUE)

## Not run:
animate_data(stream, n=5000, pointInterval=100, xlim=c(0,1), ylim=c(0,1))

## End(Not run)
```

DSOutlier_MCOD *Micro-cluster Continuous Outlier Detector (MCOD)*

Description

Class interfaces the MOA implementation of the MCOD algorithm for distance-based data stream outlier detection.

Usage

```
DSOutlier_MCOD(r = 0.1, t = 50, w = 1000, recheck_outliers = TRUE)
```

Arguments

r	Defines the micro-cluster radius
t	Defines the number of neighbors (k in the article)
w	Defines the window width in data points
recheck_outliers	Defines that the MCOD algorithm allows re-checking of detected outliers.

Details

The algorithm detects density-based outliers. An object x is defined to be an outlier if there are less than t objects lying at distance at most r from x.

Value

An object of class DSC_MCOD (subclass of DSOutlier, DSC_Micro, DSC_MOA and DSC).

Author(s)

Dalibor Krleža

References

Kontaki M, Gounaris A, Papadopoulos AN, Tsichlas K, and Manolopoulos Y (2016). Efficient and flexible algorithms for monitoring distance-based outliers over data streams. Information systems, vol. 55, pp. 37-53. doi:10.1109/ICDE.2011.5767923

See Also

DSOutlier, DSC, DSC_Micro, DSC_MOA

Examples

```
set.seed(1234)

# two-stage example
stream <- DSD_Gaussians(k = 3, d = 2,
            separation_type = "Mahalanobis", separation = 2,
            space_limit = c(0, 30), variance_limit = 0.8,
            outliers = 10,
            outlier_options = list(outlier_horizon = 200))

mic_c <- DSC_MCOD(r = 2, t = 10, w = 200)
mac_c <- DSC_Kmeans(3)
c <- DSC_TwoStage(mic_c, mac_c)

evaluate(c, stream, n = 200, type = "macro",
```

```
  measure = c("crand","outlierjaccard"))

reset_stream(stream)
plot(c, stream, n = 200, type = "all")
```

Index