Package ‘streamMOA’

January 11, 2018

Version 1.1-3

Date 2018-01-07

Title Interface for MOA Stream Clustering Algorithms

Description Interface for data stream clustering algorithms implemented in the MOA (Massive Online Analysis) framework.

Depends stream (>= 1.1-2), rJava (>= 0.6-3)

Imports graphics, stats, methods

SystemRequirements Java (>= 5.0)

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

License GPL-3

NeedsCompilation no

Author Michael Hahsler [aut, cre, cph],
Matthew Bolanos [aut, cph],
John Forrest [aut, cph]

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

Repository CRAN

Date/Publication 2018-01-10 23:56:39 UTC

R topics documented:

DSC_CluStream *CluStream Data Stream Clusterer*

Description

Class implements the CluStream cluster algorithm for data streams.

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.

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

Class implements the ClusTree cluster algorithm for data streams.

Usage

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

Arguments

`horizon`	Range of the (time) window.
`maxHeight`	The maximum height of the tree.
`lambda`	number used to override computed lambda (decay).

Details

This is an interface to the MOA implementation of ClusTree.

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 http://dx.doi.org/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)

clustree <- DSC_ClusTree(maxHeight=3)
update(clustree, stream, 500)
clustree

# plot micro-clusters
plot(clustree, stream)

# recluster with k-means
kmeans <- DSC_Kmeans(k=3)
recluster(kmeans, clustree)
plot(kmeans, stream, type="both")

# create a two stage clusering using ClusTree and reachability reclustering
CTxReach <- DSC_TwoStage(
  micro=DSC_ClusTree(maxHeight=3),
  macro=DSC_Reachability(epsilon = .15)
)
CTxReach
update(CTxReach, stream, 500)
plot(CTxReach, stream, type="both")
```

DSC_DenStream *DenStream Data Stream Clusterer*

Description

Class implements the DenStream cluster algorithm for data streams.

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

Interface to the DenStream implementation in MOA.

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_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.

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

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

DSC

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)
```

Index