Clustering from Data Streams

João Gama
LIAAD-INESC Porto,
University of Porto, Portugal
jgama@fep.up.pt

Outline

Clustering

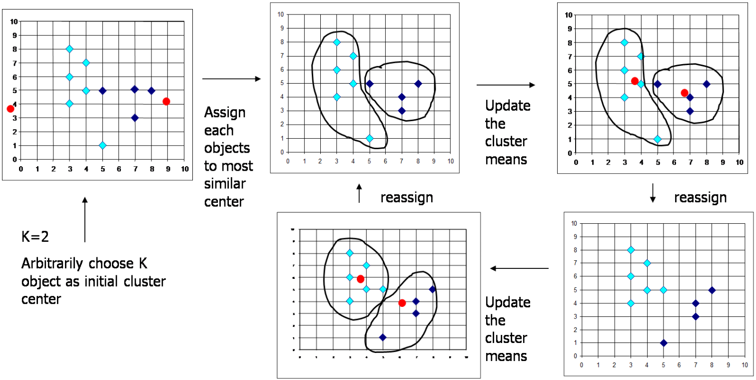
What is cluster analysis?

- Grouping a set of data objects into a set of clusters,
- the intra-cluster similarity is high and
- the inter-cluster similarity is low

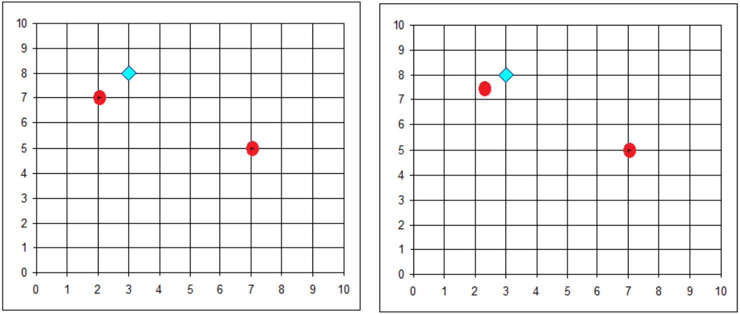
- The quality of a clustering result depends on both the similarity measure used
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

Illustrative Example: K-means

MacQueen 67: Each cluster is represented by the center of the cluster

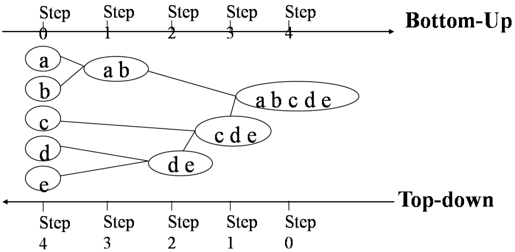


K-Means for Streaming Data



Illustrative Example: Hierarchical Clustering

- Bottom-Up
  - Initial State: Each object is a group.
  - Iteratively join two groups in a single one.
- Top-Down
  - Initial State: Single Group with all the objects.
  - Iteratively divide each group into two groups.



Major Clustering Approaches

- **Partitioning algorithms**: Construct various partitions and then evaluate them by some criterion
  - E.g., k-means, k-medoids, etc.
- **Hierarchy algorithms**: Create a hierarchical decomposition of the set of data (or objects) using some criterion.
  - Often needs to integrate with other clustering methods, e.g., BIRCH
- **Density-based**: based on connectivity and density functions
  - Finding clusters of arbitrary shapes, e.g., DBSCAN, OPTICS, etc.
- **Grid-based**: based on a multiple-level granularity structure
  - View space as grid structures, e.g., STING, CLIQUE
- **Model-based**: find the best fit of the model to all the clusters
  - Good for conceptual clustering, e.g., COBWEB, SOM

Learning Algorithms: Desirable Properties

- Processing each example:
  - Small constant time
  - Fixed amount of main memory
  - Single scan of the data
  - Without (or reduced) revisit old records.
- Processing examples at the speed they arrive
- Decision Models at anytime
- Ideally, produce a model equivalent to the one that would be obtained by a batch data-mining algorithm
- Ability to detect and react to concept drift

Clustering Data Streams

- New requirements in stream clustering
  - Generate high-quality clusters in one scan
  - High quality, efficient incremental clustering
  - Analysis should take care of multi-dimensional space
  - Analysis for different time granularity
  - Tracking the evolution of clusters
- Clustering: A stream data reduction technique

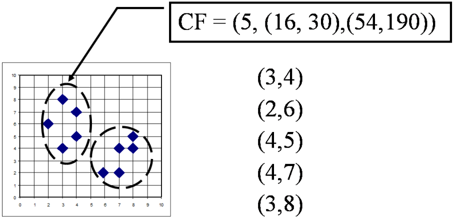
Outline

Cluster Feature Vector

Birch: Balanced Iterative Reducing and Clustering using Hierarchies, by Zhang, Ramakrishnan, Livny 1996

Cluster Feature Vector: CF $= (N, LS, SS)$

- N: Number of data points
- LS: $\sum_1^N \vec{x_i}$
- SS: $\sum_1^N (\vec{x_i})^2$



(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

Constant space irrespective to the number of examples!

Micro clusters

The sufficient statistics of a cluster A are $CF_A = (N, LS, SS)$.

- N, the number of data objects,
- LS, the linear sum of the data objects,
- SS, the sum of squared the data objects.

Properties:

- Centroid $= LS/N$
- Radius $= \sqrt{SS/N - (LS/N)^2}$
- Diameter $= \sqrt{\frac{2 \times N * SS - 2 \times LS^2}{N \times (N-1)}}$

Micro clusters

Given the sufficient statistics of a cluster A, $CF_A = (N_A, LS_A, SS_A)$. Updates are:

- Incremental: a point x is added to the cluster:
  $LS_A \leftarrow LS_A + x$; $SS_A \leftarrow SS_A + x^2$; $N_A \leftarrow N_A + 1$
- Additive: merging clusters A and B:
  $LS_C \leftarrow LS_A + LS_B$; $SS_C \leftarrow SS_A + SS_B$; $N_C \leftarrow N_A + N_B$
- Subtractive:
  $CF(C_1 - C_2) = CF(C_1) - FV(C_2)$

CluStream

CluStream: A Framework for Clustering Evolving Data Streams (VLDB03)

- Divide the clustering process into online and offline components
  - Online: periodically stores summary statistics about the stream data
    - Micro-clustering: better quality than k-means
    - Incremental, online processing and maintenance
  - Offline: answers various user queries based on the stored summary statistics
    - Tilted time frame work: register dynamic changes
- With limited overhead to achieve high efficiency, scalability, quality of results and power of evolution/change detection

CluStream: Online Phase

Inputs:

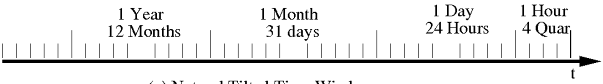
- Maximum micro-cluster diameter D_{max}

For each x in the stream:

- Find the nearest micro-cluster M_i
  - IF the diameter of $(M_i \cup x) < D_{max}$
  - THEN assign x to that micro-cluster $M_i \leftarrow M_i \cup x$
  - ELSE Start a new micro-cluster based on x

Pyramidal Time Frame

- The micro-clusters are stored at snapshots.
- When should we make the snapshot?
- The snapshots follow a pyramidal pattern:



(a) Natural Tilted Time Window

b) Logarithmic Tilted Time Window

Analysis

- find the cluster structure in the current window,
- find the cluster structure over time ranges with granularity confined by the specification of window size and boundary,
- put different weights on different windows to mine various kinds of weighted cluster structures,
- mine the evolution of cluster structures based on the changes of their occurrences in a sequence of windows

Any Time Stream Clustering

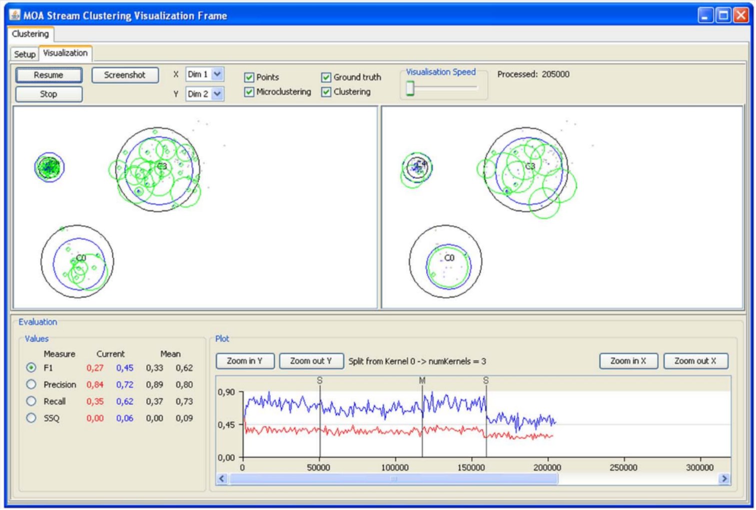
Properties of anytime algorithms

- Deliver a model at any time
- Improve the model if more time is available
  - Model adaptation whenever an instance arrives
  - Model refinement whenever time permits

ClusTree [Kranen et al., 2011]

- an online component to learn micro-clusters
- Any variety of online components can be utilized
- Micro-clusters are subject to exponential aging

MOA


MOA Stream Clustering Visualization Frame
Clustering
Setup
Visualization
Resume
Stop
Screenshot
X Dim 1
Y Dim 2
Points
Microclustering
Ground truth
Clustering
Visualisation Speed
Processed: 205000
Evaluation
Values
Measure Current Mean
F1 0,27 0,45 0,33 0,62
Precision 0,84 0,72 0,89 0,80
Recall 0,35 0,62 0,37 0,73
SSQ 0,00 0,06 0,00 0,09
Plot
Zoom in Y
Zoom out Y
Split from Kernel 0 -> numKernels = 3
Zoom in X
Zoom out X
0,90
0,45
0,00
0 50000 100000 150000 200000 250000 300000

Outline

Clustering Time Series Data Streams

Goal: Continuously maintain a clustering structure from evolving time series data streams.

- Ability to Incorporate new Information;
- Process new Information at the rate it is available.
- Ability to Detect and React to *changes* in the Cluster's Structure.

Clustering of *variables* (sensors) not examples!
The standard technique of transposing the working-matrix does not work: transpose is a blocking operator!

Online Divisive-Agglomerative Clustering

Online Divisive-Agglomerative Clustering, Rodrigues & Gama, 2008

Goal: Continuously maintain a hierarchical cluster's structure from evolving time series data streams.

- Performs hierarchical clustering
- Continuously monitor the evolution of **clusters' diameters**
- Two Operators:
  - Splitting: expand the structure
    more data, more detailed clusters
  - Merge: contract the structure
    reacting to changes.
- Splitting and agglomerative criteria are supported by a confidence level given by the **Hoeffding bounds**.

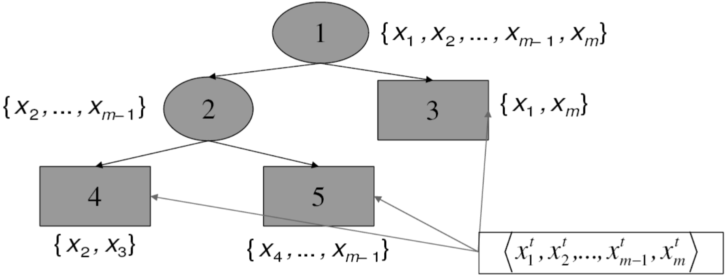
Main Algorithm

- ForEver
    - Read Next Example
    - For all the clusters
        - Update the sufficient statistics
    - Time to Time
        - Verify Merge Clusters
        - Verify Expand Cluster

Feeding ODAC

Each example is processed once.
Only sufficient statistics **at leaves** are updated.
Sufficient Statistics: a triangular matrix of the correlations between variables in a leaf.

Released when a leaf expands to a node.



$$C_1 = \{x_2, x_3\}, C_2 = \{x_4, \ldots, x_{m-1}\}, C_3 = \{x_1, x_m\}$$

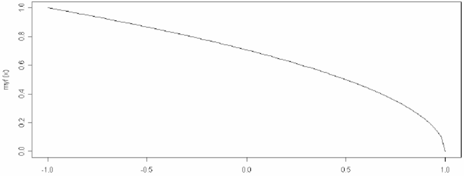
Similarity Distance

Distance between time Series: $rnomc(a, b) = \sqrt{\frac{1-corr(a,b)}{2}}$
where $corr(a, b)$ is the Pearson Correlation coefficient:

$$corr(a, b) = \frac{P - \frac{AB}{n}}{\sqrt{A_2 - \frac{A^2}{n}}\sqrt{B_2 - \frac{B^2}{n}}}$$

The *sufficient statistics* needed to compute the correlation are easily updated at each time step:

$$A = \sum a_i, \; B = \sum b_i, \; A_2 = \sum a_i^2, \; B_2 = \sum b_i^2, \; P = \sum a_i b_i$$

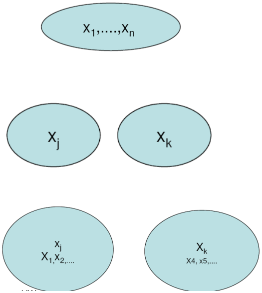


The Expand Operator: Expanding a Leaf

Step 1 Find Pivots:
$x_j, x_k : d(x_j, x_k) > d(a, b)$
$\forall a, b \neq j, k$

Step 2 If Splitting Criteria applies:
Generate two new clusters.

Step 3 Each new cluster attract nearest variables.



Splitting Criteria

When should we expand a leaf?
Let

- $d_1 = d(a, b)$ the farthest distance
- d_2 the second farthest distance

Question:

Is d_1 a stable option?
what if we observe more examples?

Hoeffding bound:

Split if $d_1 - d_2 > \epsilon$ with $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$
where R is the range of the random variable; δ is a user confidence level, and n is the number of observed data points.

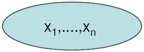
Hoeffding bound

- Suppose we have made n independent observations of a random variable r whose range is R.
- The Hoeffding bound states that:
  - With probability $1 - \delta$
  - The true mean of r is in the range $\bar{r} \pm \epsilon$ where $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$
- Independent of the probability distribution generating the examples.

The Expand Operator: Expanding a Leaf

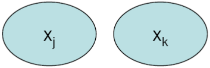
Step 1

Find Pivots:
$x_j, x_k : d(x_j, x_k) > d(a, b)$
$\forall a, b \neq j, k$



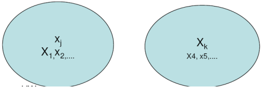
Step 2

If the Hoeffding bound applies:
Generate two new clusters.



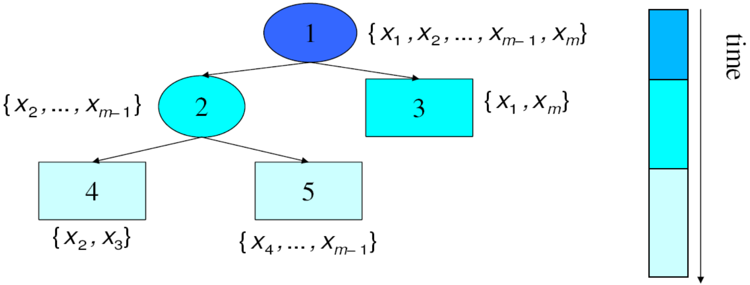
Step 3

Each new cluster attract nearest variables.



Multi-Time-Windows

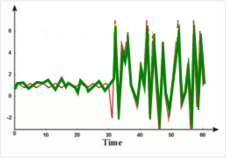
A multi-window system: each node (and leaves) receive examples from different time-windows.



The Merge Operator: Change Detection

Time Series Concept Drift:

- Changes in the distribution generating the observations.
- Clustering Concept Drift
  - Changing in the way time series correlate with each other
  - Change in the cluster Structure.



The Merge Operator: Change Detection

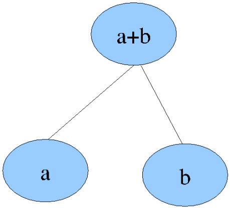
The Splitting Criteria guarantees that cluster's diameters monotonically decrease.

- Assume Clusters: c_j with descendants c_k and c_s.
- If $diameter(c_k) - diameter(c_j) > \epsilon$ OR $diameter(c_s) - diameter(c_j) > \epsilon$
  - Change in the correlation structure!
  - Merge clusters c_k and c_s into c_j.

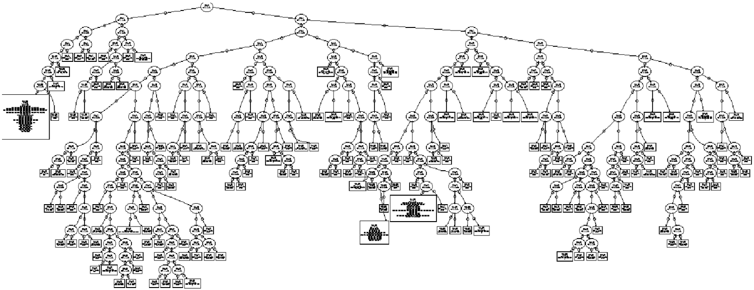
Properties of ODAC

- For stationary data the cluster's diameters monotonically decrease.
- **Constant update time/memory consumption** with respect to the number of examples!
- Every time a **split** is reported
  - the **time** to process the next example **decreases**, and
  - the **space** used by the new leaves is **less than** that used by the parent.

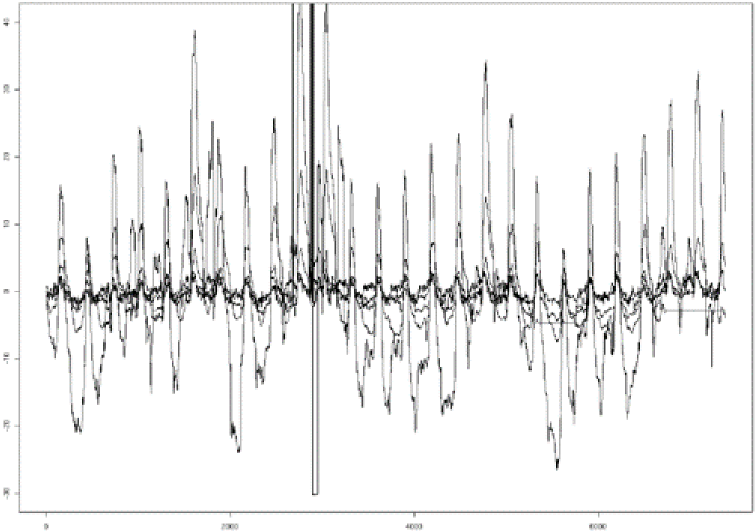
$$(a+b)^2 > a^2 + b^2$$



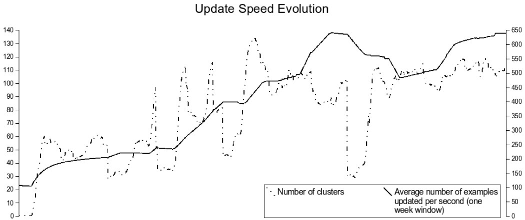
The Electrical Load Demand Problem



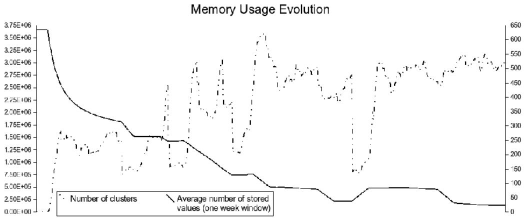
The Electrical Load Demand Problem



Evolution of Processing Speed



Evolution of Memory Usage



Outline

Master References

- J. Gama, *Knowledge Discovery from Data Streams*, CRC Press, 2010.
- S. Muthukrishnan *Data Streams: Algorithms and Applications*, Foundations & Trends in Theoretical Computer Science, 2005.
- C. Aggarwal (Ed) *Data Streams: Models and Algorithms*, Springer, 2007
- B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. *Models and Issues in Data Stream Systems*, Proceedings of PODS, 2002.
- Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., *Mining Data Streams: A Review*, in ACM SIGMOD Record, Vol. 34, No. 1, 2005.
- J. Silva, E. Faria, R. Barros, E. Hruschka, A. Carvalho, J. Gama: Data stream clustering: A survey. ACM Comput. Surv. 46(1): 13 (2013)

Bibliography on Clustering

- P. Rodrigues, J. Gama and J. P. Pedroso. *Hierarchical Clustering of Time Series Data Streams*; TKDE, 2008.
- Zhang, Ramakrishnan, Livny; *Birch: Balanced Iterative Reducing and Clustering using Hierarchies*, SIGMOD 1996.
- Charu Aggarwal, Jiawei Han, J. Wang, P. S. Yu, *A Framework for Clustering Evolving Data Streams*, by VLDB 2003.
- G. Cormode, S. Muthukrishnan, and W. Zhuang, *Conquering the divide: Continuous clustering of distributed data streams*. ICDE 2007.
- Kranen, Assent, Baldauf, and Seidl; *The ClusTree: indexing micro-clusters for anytime stream mining*, Knowl. Inf. Syst. 29, 2 2011
- Cormode, Muthu, Zhuang; *Conquering the Divide: Continuous Clustering of Distributed Data Streams*. ICDE 2007
- P. Rodrigues, J. Gama: Clustering Distributed Sensor Data Streams. ECML/PKDD 2008
- P. Rodrigues, J. Gama: L2GClust: local-to-global clustering of stream sources. SAC 2011