Introduction to Learning from Data Streams

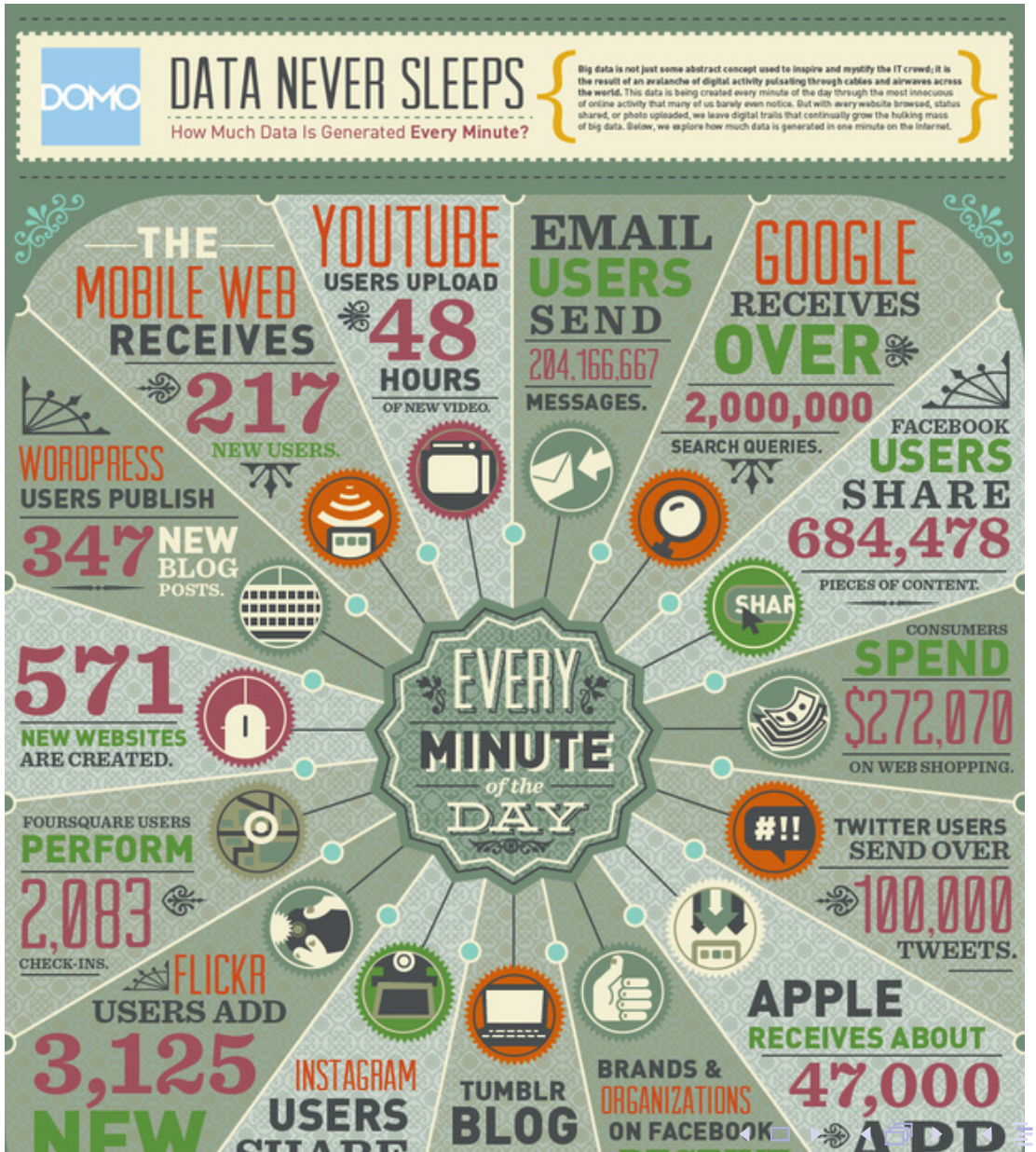
João Gama
LIAAD-INESC Porto,
University of Porto, Portugal
jgama@fep.up.pt

Summer School on Non-Linear Dynamics, Machine Learning and Climate Modeling
Ohrid, 5th September, Macedonia

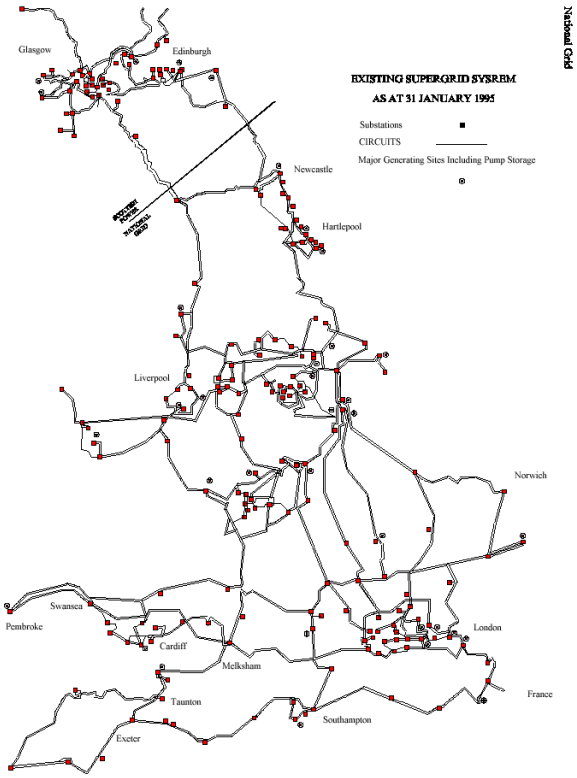
Tribute to Sir Ronald Fisher

Nowadays ...

Data Never Sleeps ...



Scenario



Electrical power Network: Sensors all around network monitor measurements of interest.

Scenario

- Sensors produce continuous flow of data at high speed:
  - Send information at different time scales;
  - Act in adversary conditions: they are prone to noise, weather conditions, battery conditions, etc;
- Huge number of Sensors, variable along time
- Geographic distribution:
  - The topology of the network and the position of the sensors are known.

Illustrative Learning Tasks:

- Cluster Analysis
  - Identification of Profiles: Urban, Rural, Industrial, etc.

Illustrative Learning Tasks:

- Cluster Analysis
  - Identification of Profiles: Urban, Rural, Industrial, etc.
- Predictive Analysis
  - Predict the value measured by each sensor for different time horizons.
  - Prediction of peaks on the demand.

Illustrative Learning Tasks:

- Cluster Analysis
  - Identification of Profiles: Urban, Rural, Industrial, etc.
- Predictive Analysis
  - Predict the value measured by each sensor for different time horizons.
  - Prediction of peaks on the demand.
- Monitoring Evolution
  - Change Detection
    - Detect changes in the behavior of sensors;
    - Detect Failures and Abnormal Activities;
  - Extreme Values, Anomalies and Outliers Detection
    - Identification of peaks on the demand;
    - Identification of **critical points** in load evolution;

Standard Approach:

This problem has been addressed time ago:

Strategy

- Select a finite sample
- Generate a static model (cluster structure, neural nets, Kalman filters, Wavelets, etc)
- Very good performance in next month!
- Six months later: Retrain everything!

Standard Approach:

This problem has been addressed time ago:

Strategy

- Select a finite sample
- Generate a static model (cluster structure, neural nets, Kalman filters, Wavelets, etc)
- Very good performance in next month!
- Six months later: Retrain everything!

What is the Problem?

The world is not static!
Things change over time.

The Data Stream Phenomenon

- Highly detailed, automatic, rapid data feeds.
  - Radar: meteorological observations.
  - Satellite: geodetics, radiation,.
  - Astronomical surveys: optical, radio,.
  - Internet: traffic logs, user queries, email, financial,
  - Sensor networks: many more *observation points* ...
- Most of these data will never be seen by a human!
- Need for near-real time analysis of data feeds.
- Monitoring, intrusion, anomalous activity Classification, Prediction, Complex correlations, Detect outliers, extreme events, etc

Data Streams

Continuous flow of data generated at **high-speed** in **Dynamic**, **Time-changing** environments.
The usual approaches for *querying*, *clustering* and *prediction* use **batch procedures** cannot cope with this streaming setting.
Machine Learning algorithms assume:

- Instances are independent and generated at random according to some probability distribution $\mathcal{D}$.
- It is required that $\mathcal{D}$ is stationary

Practice: *finite* training sets, *static* models.

Data Streams

We need to maintain **Decision models** in **real time**.
Decision Models must be capable of:

- **incorporating** new information at the speed data arrives;
- **detecting** changes and **adapting** the decision models to the most recent information.
- **forgetting** outdated information;

Unbounded training sets, dynamic models.

Clustering Data Streams

Clustering

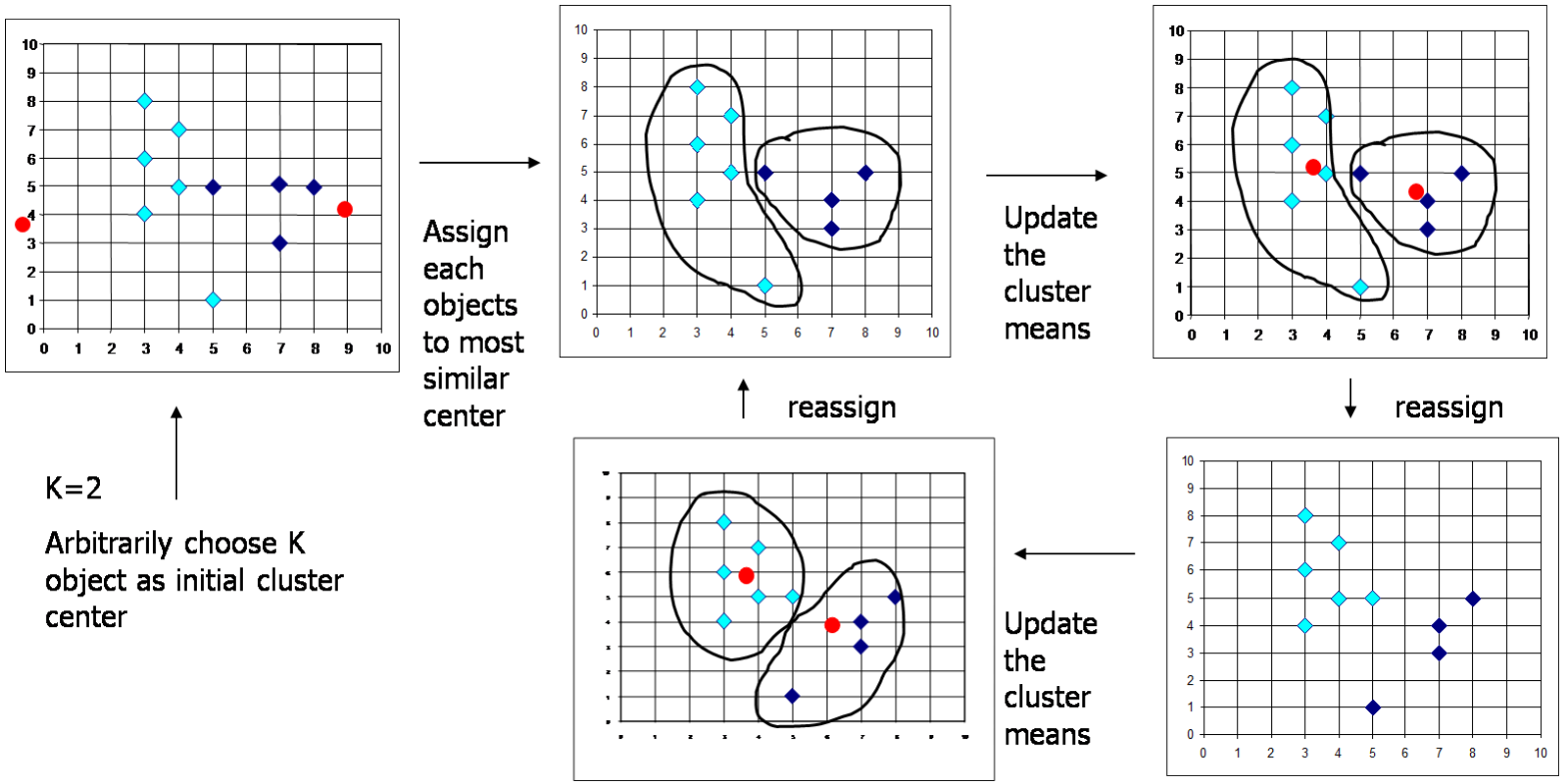
What is cluster analysis?

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

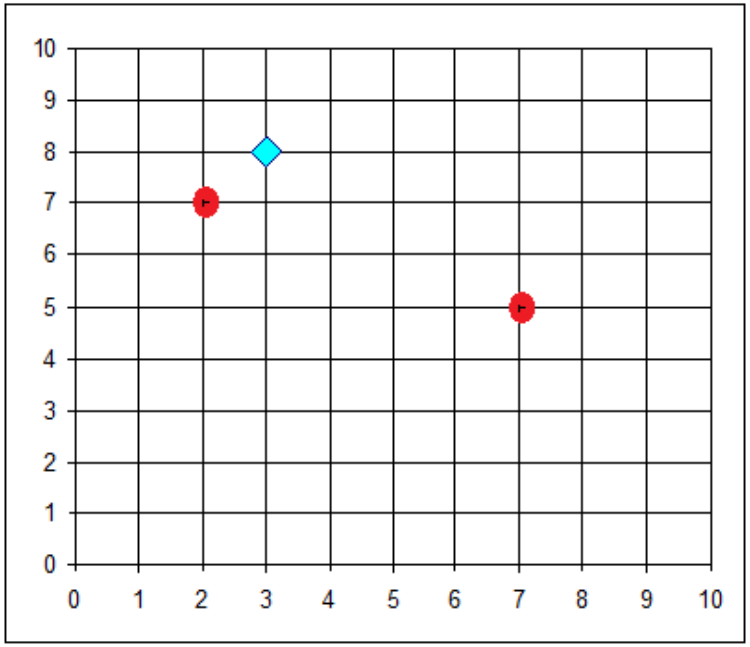
- The quality of a clustering result depends on both the similarity measure used
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

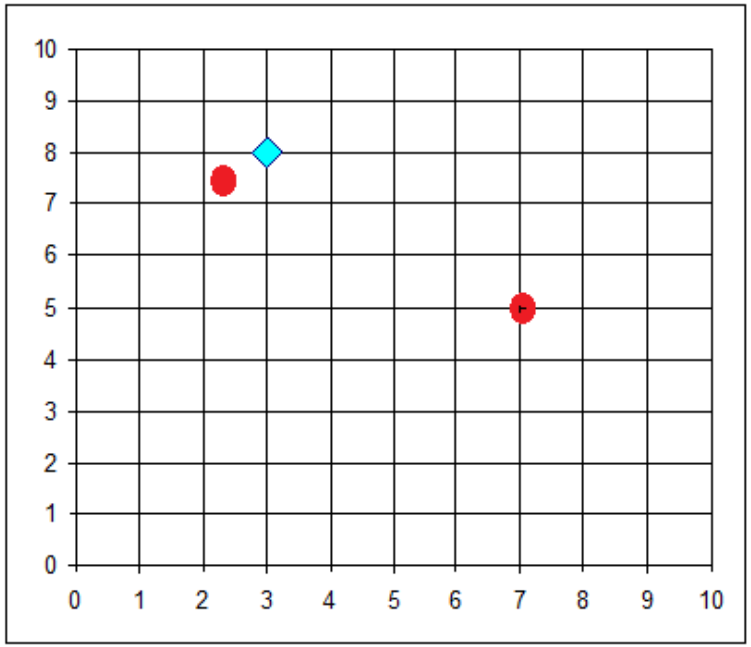
Illustrative Example: K-means

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



K-Means for Streaming Data



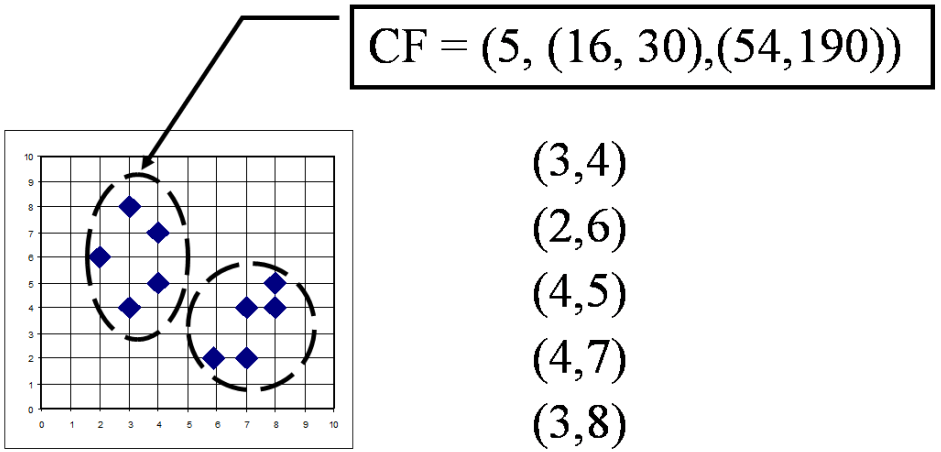


Cluster Feature Vector

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

Cluster Feature Vector: $\text{CF} = (N, LS, SS)$

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



Micro clusters

The sufficient statistics of a cluster A are (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 are (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:
  $FV(C_1 - C_2) = FV(C_1) - FV(C_2)$

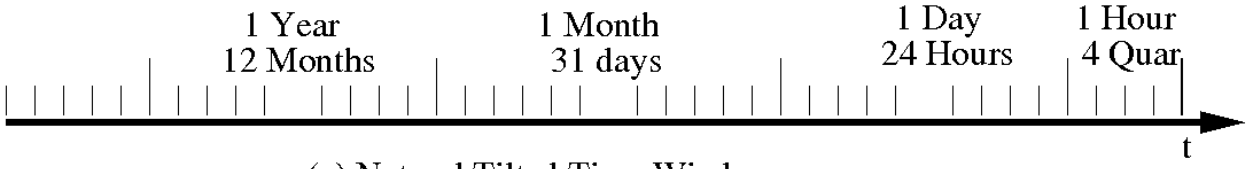
CluStream

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

Pyramidal Time Frame

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



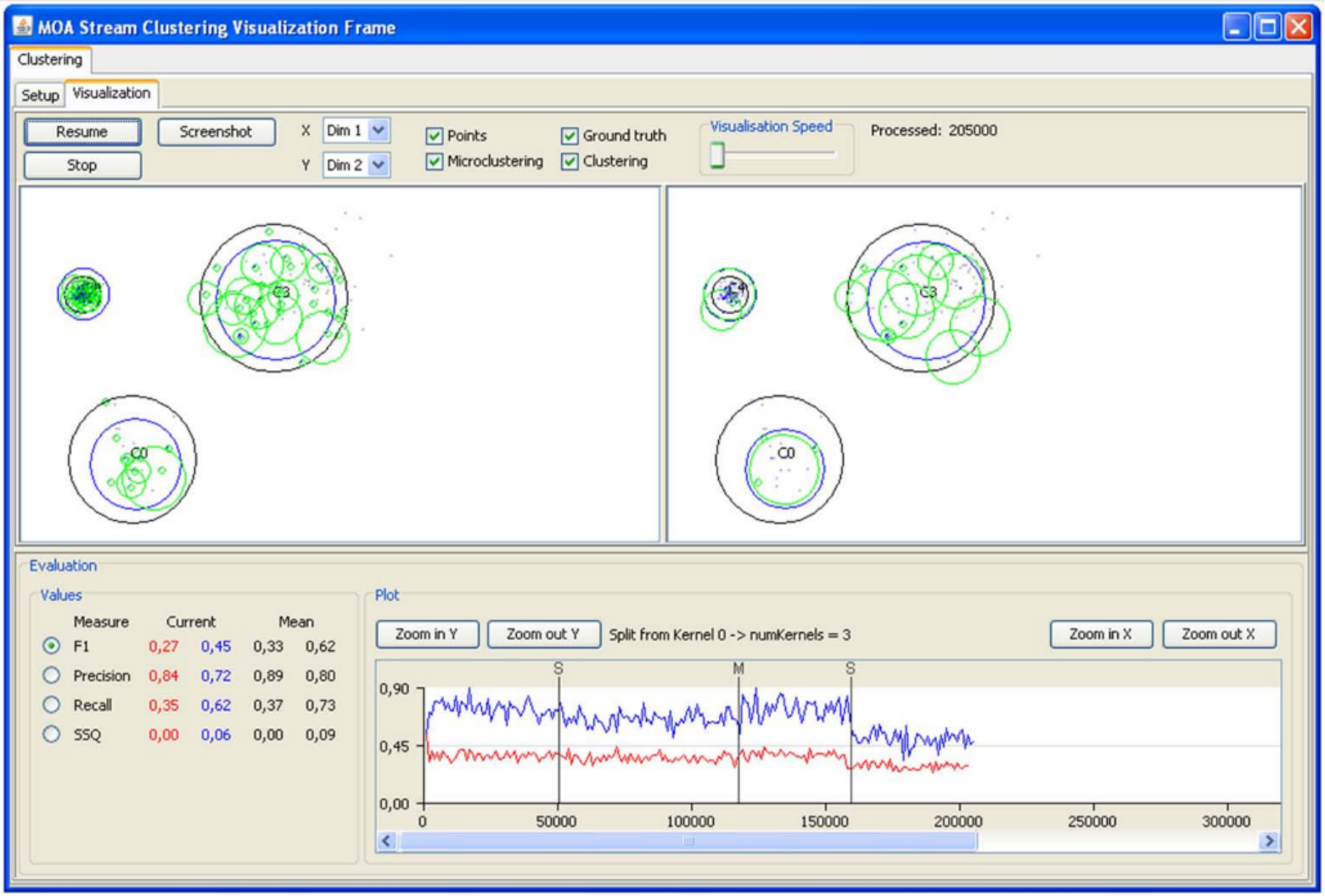
(a) Natural Tilted Time Window

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

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
C3
C0
C4
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
S
M
S
0,90
0,45
0,00
0
50000
100000
150000
200000
250000
300000

Clustering Time-series Data Streams

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

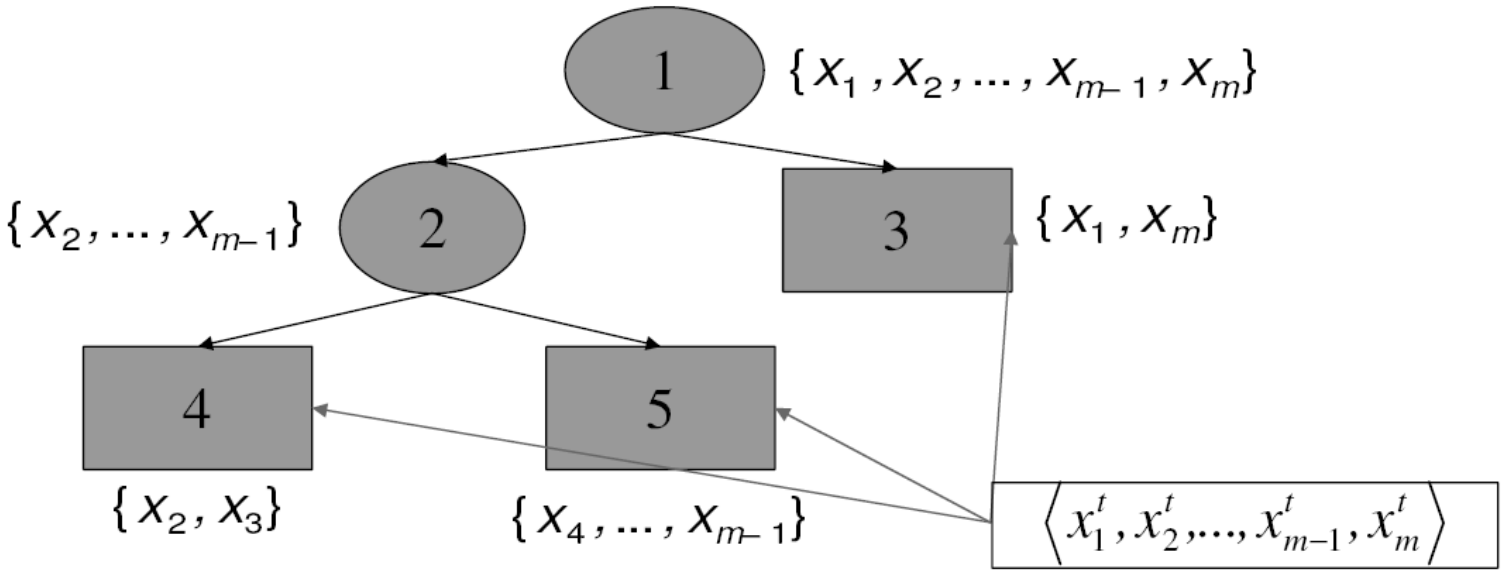
Feeding ODAC

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.



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

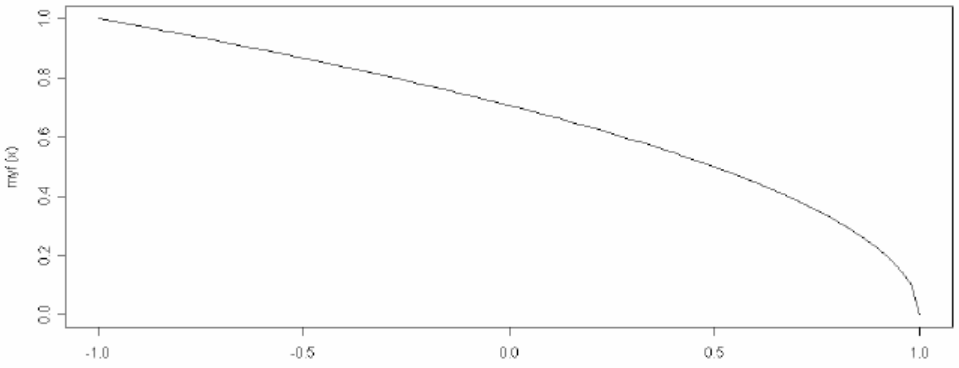
Similarity Distance

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

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

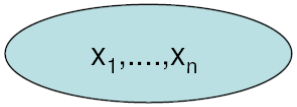
The *sufficient statistics* needed to compute the correlation are easily updated at each time step:

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

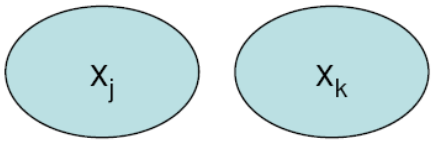


The Expand Operator: Expanding a Leaf

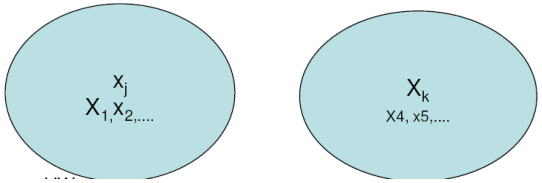
Step 1 Find Pivots:
$x_j, x_k : d(x_j, x_k) > d(a, b)$
$\forall a, b \neq j, k$



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:

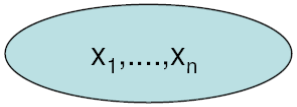
Split if $d_1 - d_2 > \epsilon$ with $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$
where R is the range of the random variable; δ is a user confidence level, and n is the number of observed data points.

Hoeffding bound

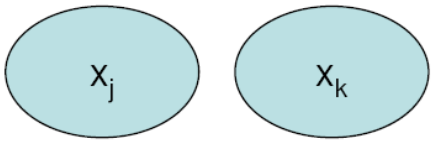
- Suppose we have made n independent observations of a random variable r whose range is R.
- The Hoeffding bound states that:
  - With probability $1 - \delta$
  - The true mean of r is in the range $\overline{r} \pm \epsilon$ where $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$
- Independent of the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

The Expand Operator: Expanding a Leaf

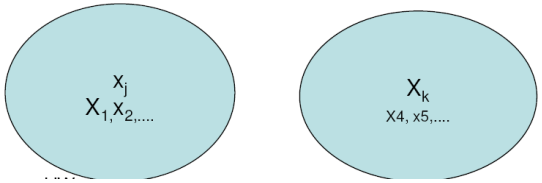
Step 1 Find Pivots:
$x_j, x_k : d(x_j, x_k) > d(a, b)$
$\forall a, b \neq j, k$



Step 2 If the Hoeffding bound applies:
Generate two new clusters.

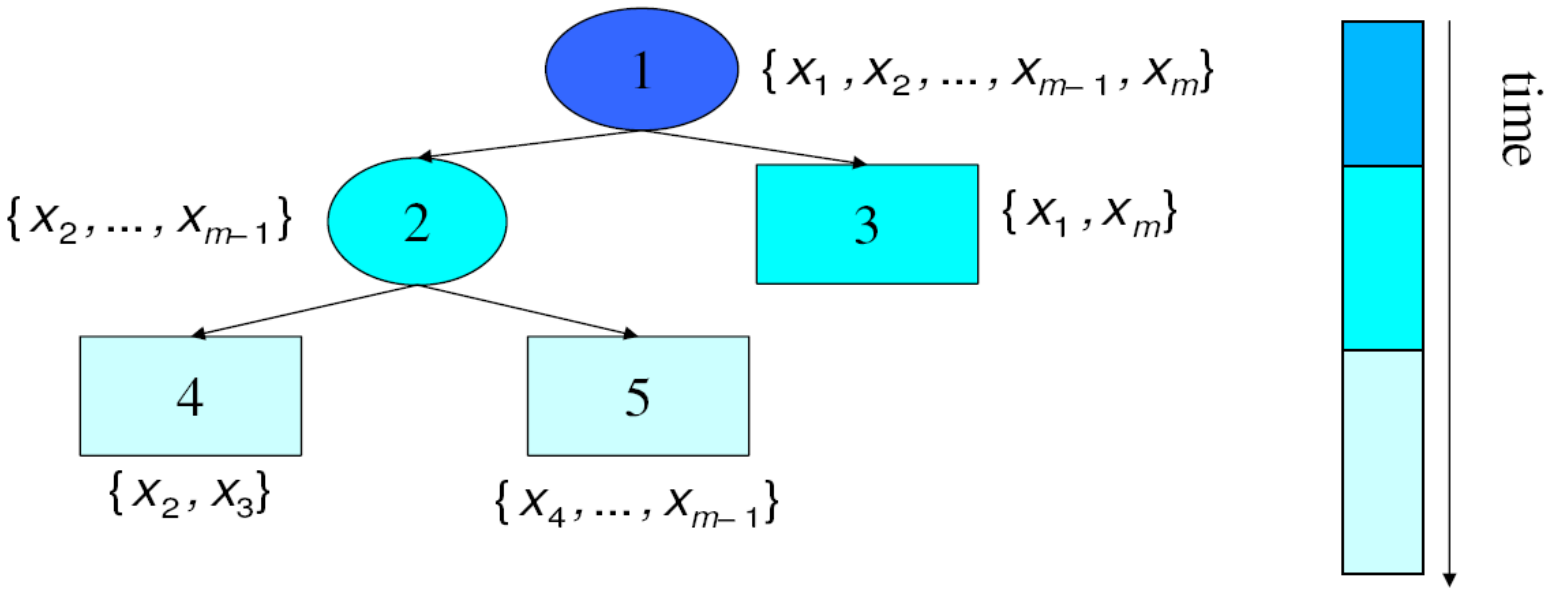


Step 3 Each new cluster attract nearest variables.



Multi-Time-Windows

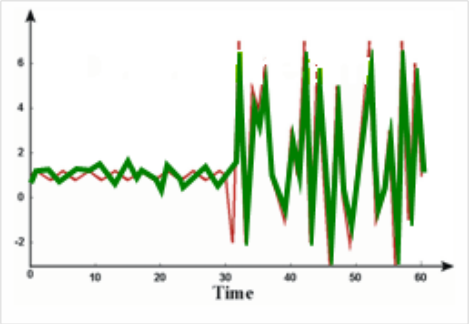
A multi-window system: each node (and leaves) receive examples from different time-windows.



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

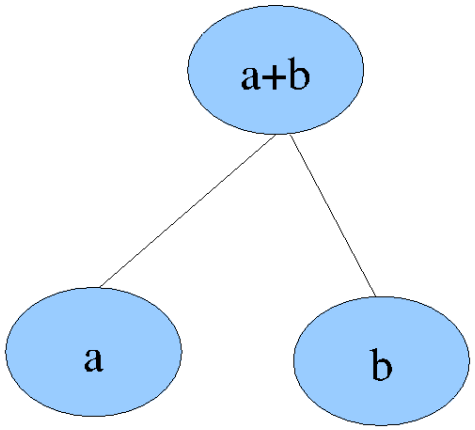
The Splitting Criteria guarantees that cluster's diameters monotonically decrease.

- 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.

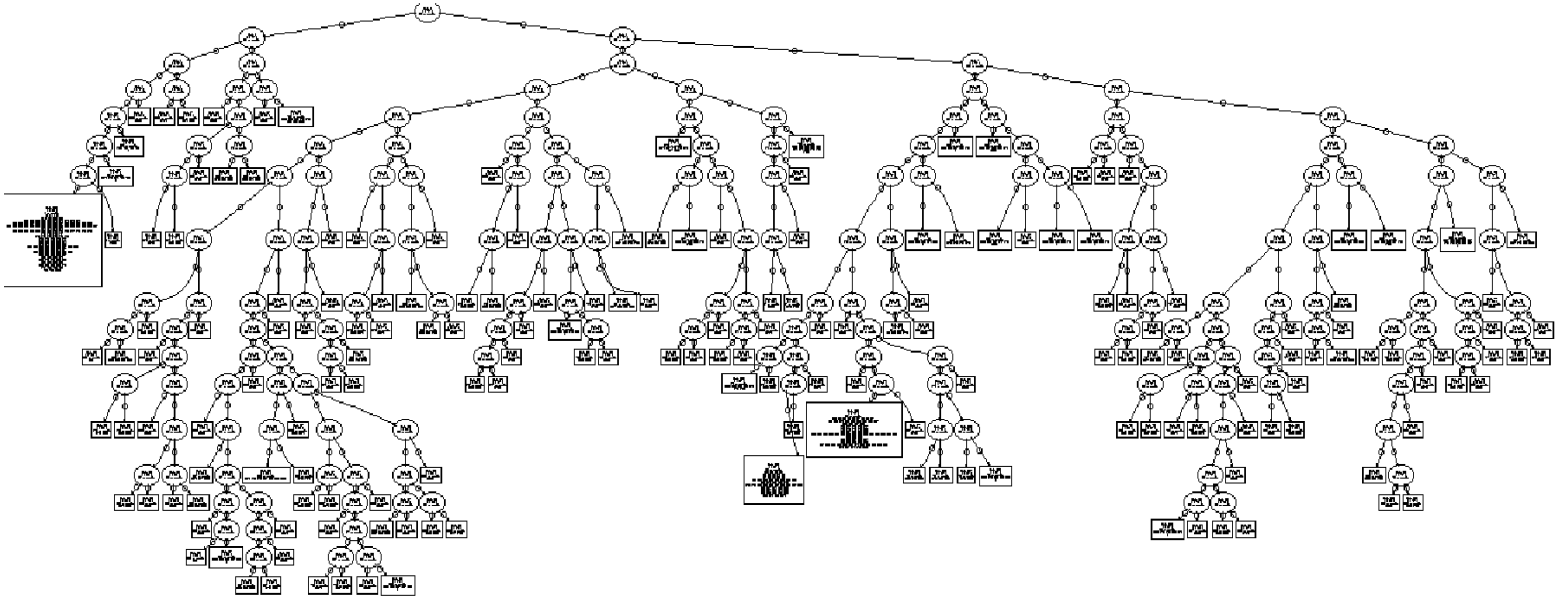
Properties of ODAC

- 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



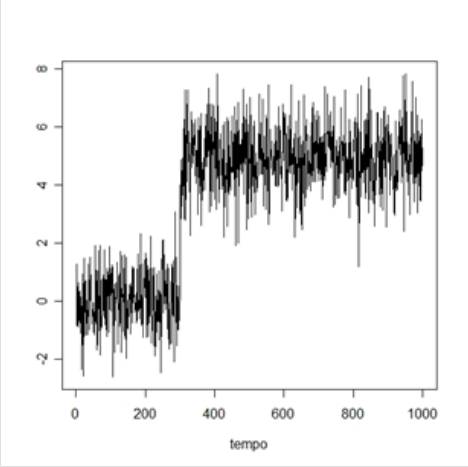
Concept Drift

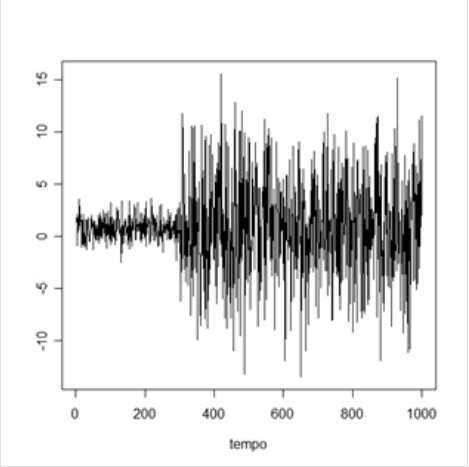
Introduction

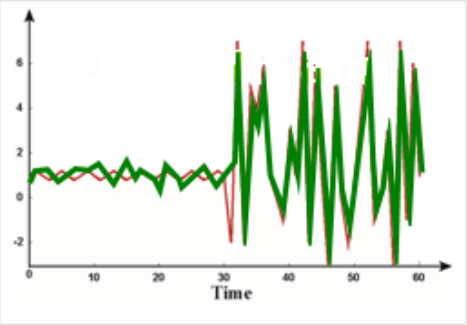
Concept drift means that the concept about which data is obtained may shift from time to time, each time after some minimum permanence.

Any change in the distribution underlying the data

Context: a set of examples from the data stream where the underlying distribution is stationary.



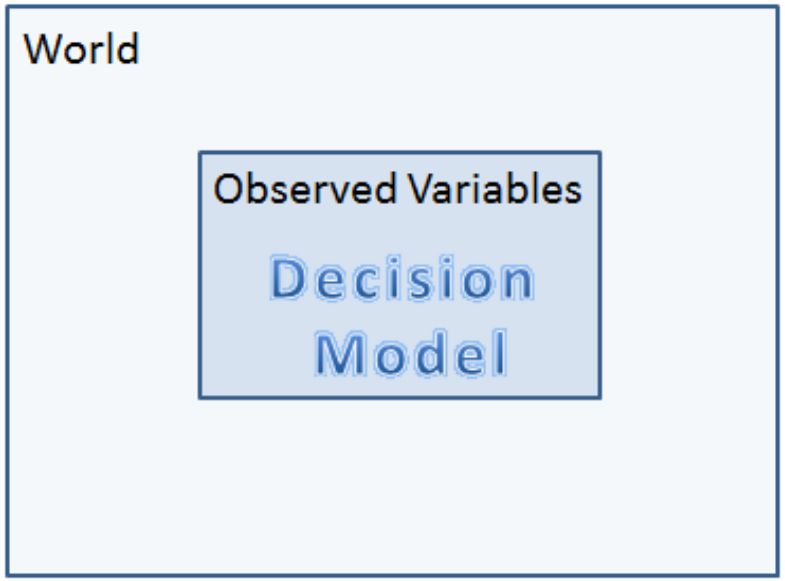




The Nature of Change

The causes of change:

- Changes due to modifications in the context of learning due to changes in **hidden variables**.
- Changes in the characteristic properties of the observed variables.



The Nature of Change

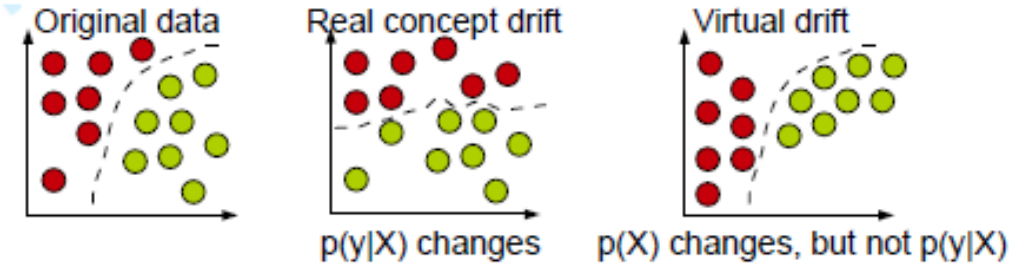
The *rate* of change.

- Concept Drift
  - Abrupt changes in the target concept
- Concept Shift
  - Refers to gradual changes

Patterns of changes over time (outlier is not concept drift).

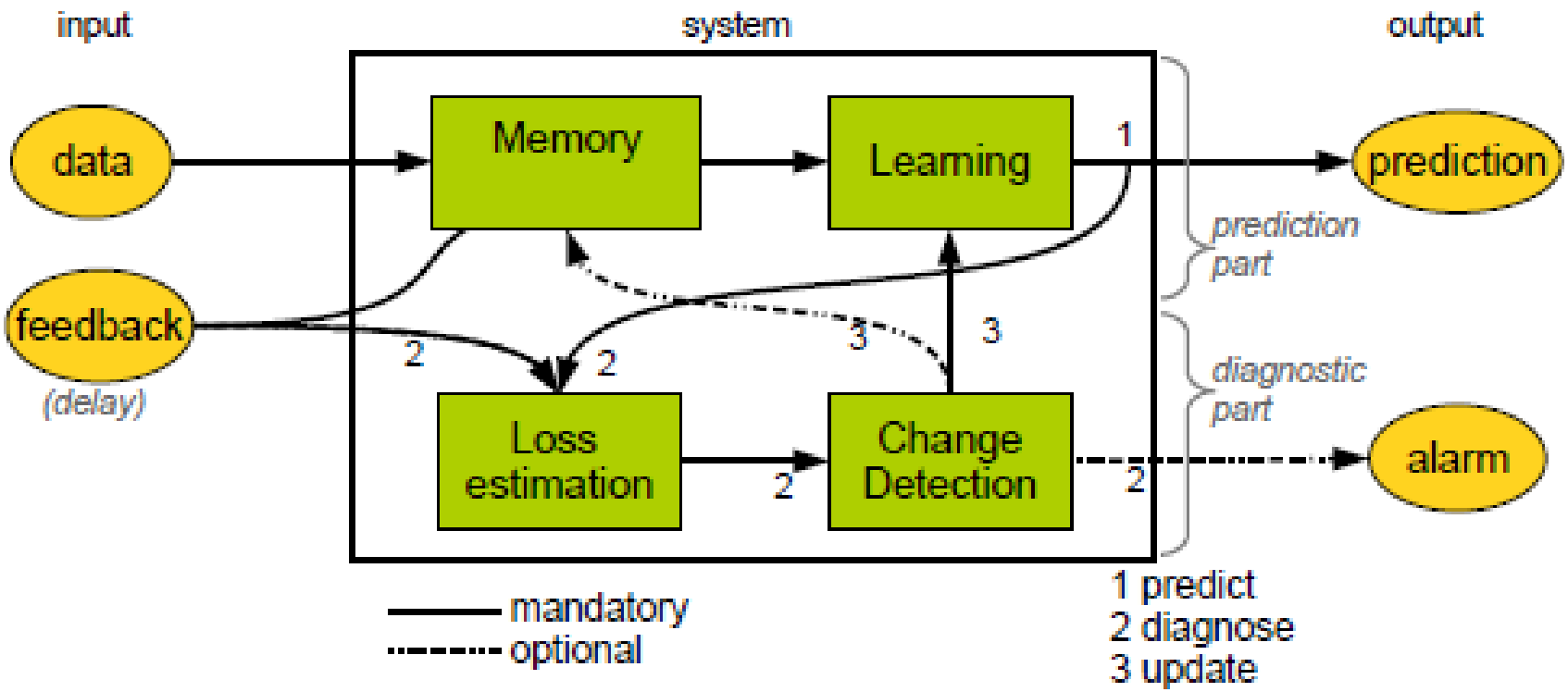
The Nature of Change

- Whenever a change in the underlying concept generating data occurs, the class-distribution of examples changes,
  - At least in some regions of the instance space.
- It is possible to observe changes in the class-distribution without concept drift.
- This is usually referred as *virtual drift*



Types of drifts.

A Generic Model for Adaptive Learning Algorithms



A generic schema for an online adaptive learning algorithm.

The Page-Hinckley Test

- The PH test is a sequential adaptation of the detection of an abrupt change in the average of a Gaussian signal.
- It considers a cumulative variable m_T, defined as the cumulated difference between the observed values and their mean till the current moment:

$$m_{t+1} = \sum_{1}^{t} (x_t - \bar{x}_t + \alpha)$$

  - where $\bar{x} = 1/t \sum_{l=1}^{t} x_l$ and
  - α corresponds to the magnitude of changes that are allowed.

The Page-Hinckley Test

$$m_{t+1} = \sum_{1}^{t} (x_t - \bar{x}_t + \alpha)$$

- The minimum value of this variable is also computed with the following formula: $M_T = min(m_t, t = 1 \ldots T)$.
- The test monitors the difference between M_T and m_T: $PH_T = m_T - M_T$.
- When this difference is greater than a given threshold (λ) we alarm a change in the distribution.

Analysis

The threshold λ depends on the admissible false alarm rate. Increasing λ will entail fewer false alarms, but might miss some changes.

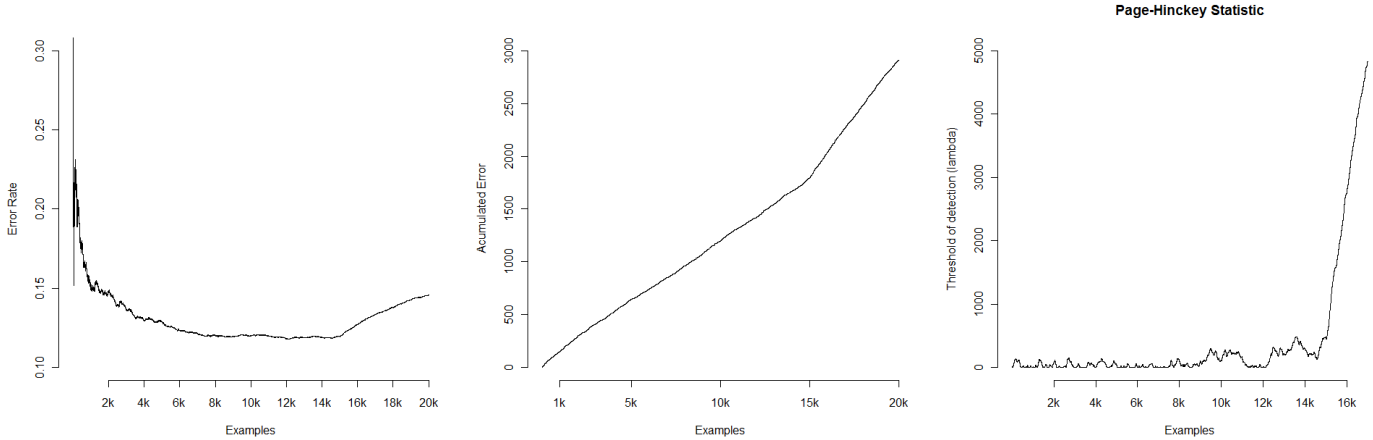


Figure : Illustrative example of the Page-Hinckey test. The left figure plots the on-line error rate of a learning algorithm. The center plot is the accumulated on-line error. The slope increases after the occurrence of a change. The right plot presents the evolution of the *PH* statistic.

Illustrative Example

Illustrative Problem I

A (real) data warehouse problem

- Suppose you have a retail data warehouse
- 3 TB of data
- 100s GB new sales records updated daily
- Millions of different items

Problem: hot-list

Identify *hot items*: the top-20 items in popularity
Restricted memory: Can have a memory of 100s-1000s bytes only

Illustrative Examples

- We see a large number of individual transactions.
  - What are the top sellers today?
- We are monitoring network traffic.
  - Which hosts/subnets are responsible for most of the traffic?
- We have a network of satellites monitoring events over large areas.
  - Which areas are experiencing the most activity over a week / day /hour?

The Top-k Elements Problem

Count the top-K most frequent elements in a stream.

First Approach

Maintain a count for each element of the alphabet.
Return the k first elements in the sorted list of counts.

Problems

Exact and Efficient solution for small alphabets.
Large alphabets: Space inefficient – large number of zero counts.

The Space Saving Algorithm

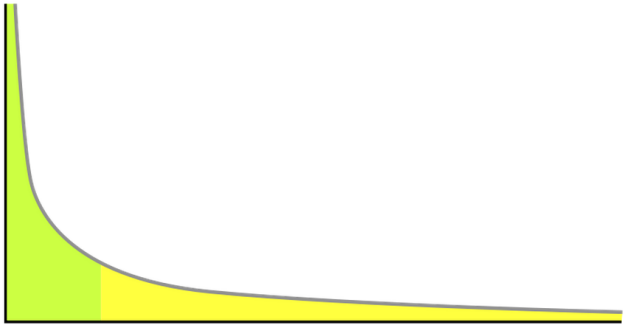
Metwally, D. Agrawal, A. Abbadi, *Efficient Computation of Frequent and Top-k Elements in Data Streams*, ICDT 2005

Maintain partial information of interest; monitor only a subset m of elements.

- For each element e in the stream
  - If e is monitored: Increment $Count_e$
  - Else
    - Let e_m be the element with least hits min.
    - Replace e_m with e with $count_e = min + 1$

The Space Saving Algorithm: Properties

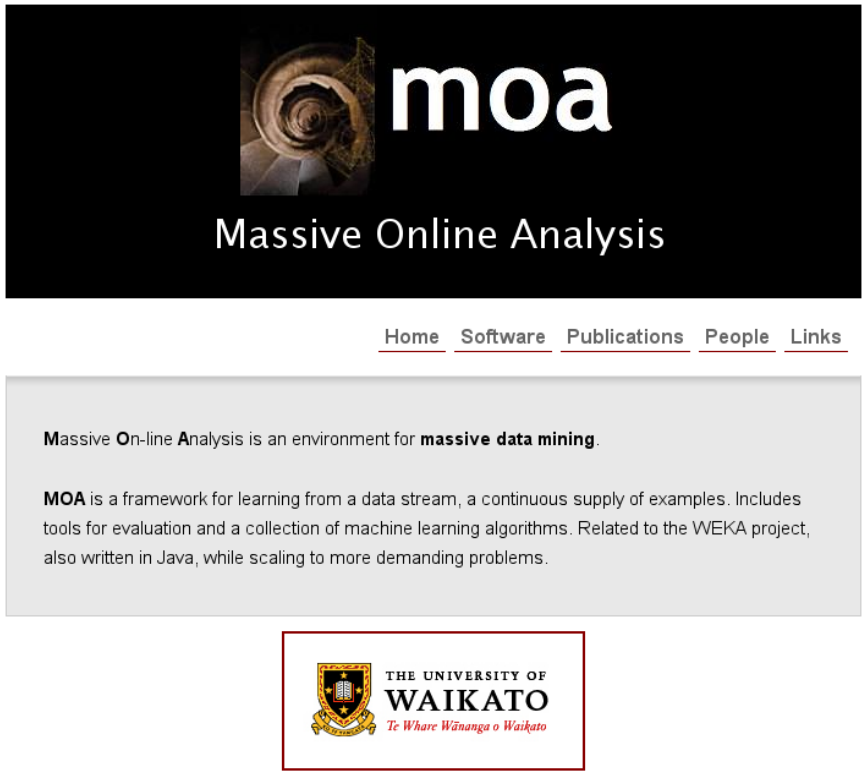
- Efficient for skewed data!
- Ensures no false negatives are kept in the top-k list:
  no non frequent item is in the top-k list.
- It allows false positive in the list:
  some non frequent items appear in the list.
- If the popular elements evolve over time, the elements that are growing more popular will gradually be pushed to the top of the list.



Master References

Software

http://moa.cms.waikato.ac.nz/

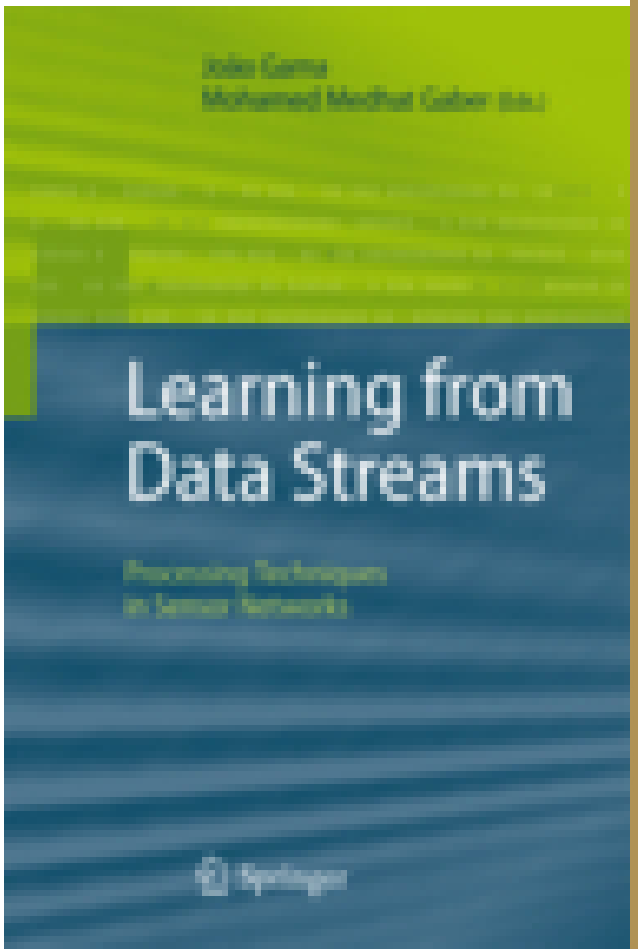


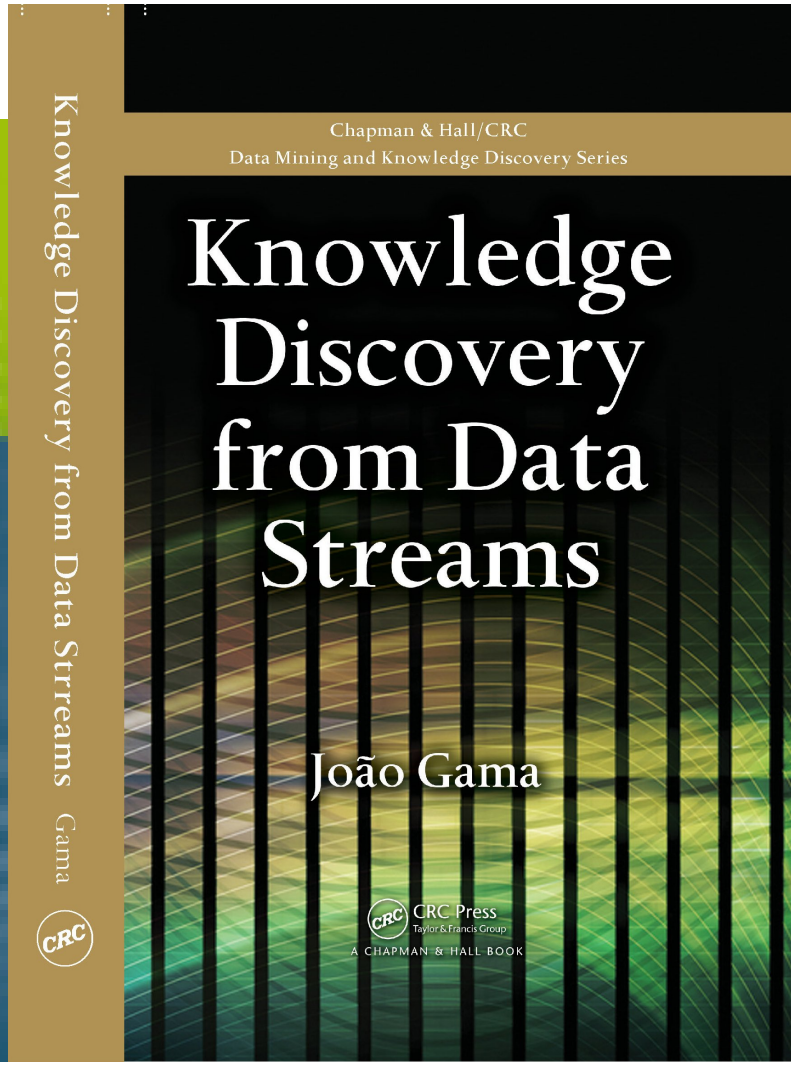
Resources

- Massive Data Analysis
  http://dimacs.rutgers.edu/~graham/
- Distributed Data Mining
  Maintained by Hillol Kargupta
- UCR Time-Series Data Sets
  Maintained by Eamonn Keogh, UCR, US
  http://www.cs.ucr.edu/~eamonn/time_series_data
- Mining Data Streams Bibliography
  Maintained by Mohamed Gaber
  http://www.csse.monash.edu.au/~mgaber/WResources.html

Data Stream Management Systems

- Niagara (OGI/Wisconsin) – Internet XML databases
- Aurora (Brown/MIT) – sensor monitoring, dataflow
  `http://www-db.stanford.edu/sdt`
- Stream (Stanford) – general-purpose DSMS
  `http://www-db.stanford.edu/stream/index.html`
- COUGAR (Cornell)
- GigaScope and Hancock (At&T)
- Medusa (Brown University)


Learning from
Data Streams


Knowledge Discovery from Data Strreams
Gama
CRC
Chapman & Hall/CRC
Data Mining and Knowledge Discovery Series
Knowledge
Discovery
from Data
Streams
João Gama
CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

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.
- B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems, in Proc. PODS, 2002.
- Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., Mining Data Streams: A Review, in ACM SIGMOD Record, Vol. 34, No. 1, 2005.
- C. Aggarwal, *Data Streams: Models and Algorithms*, Ed. Charu Aggarwal, Springer, 2007
- J. Gama, M. Gaber (Eds), *Learning from Data Streams – Processing Techniques in Sensor Networks*, Springer, 2007.