Uczenie się klasyfikatorów ze zmiennych strumieni danych

Jerzy Stefanowski

Instytut Informatyki
Politechnika Poznańska

Wersjaz 2020 − wykorzystująca materiały dla z PhD schools' talks

Inspiracje

Niektóre ze slajdów wykorzystjące pomysły z wykładów:

- Mining High Speed Data Streams, talk by P. Domingos, G. Hulten, SIGKDD 2000.
- State of the art in data streams mining, talk by M.Gaber and J.Gama, ECML 2007.
- J.Han slides for a lecture on Mining Data Streams – związanek z jego podręcznikiem Data Mining
- Myra Spiliopoulou, Frank Höppner, Mirko Böttcher - Knowledge Discovery from Evolving Data / tutorial at ECML 2008
- Indre Zliobaite − niektóre rysunki z jej publikacji

Inne pomysły współpracownicy (D.Brzeziński, M.Deckert) + mój cykl wykładów pt Ensemble Classifiers for Data Streams with Concept Drift → wykłady dla szkół doktoranckich

Powyższe → slajdy w języku angielskim

Motivation

- Real world data
  - In many applications available in a form of **data streams**
  - New computational requirement for processing them
- The task of supervised classification – more difficult
  - Data comes from complex environments that **evolve** over time
  - **Concept drift** = underlying distribution of data is changing
- Concept drifts – categorization and detection
- Algorithms need to **adapt** to changes quickly and accurately
- Survey learning algorithms

Outline of the talk – part 1

1. Introductory remarks
2. Previous incremental classifiers
3. General processing framework
4. Concept drifts
5. Data managements and forgetting mechanisms
6. Evaluation of streaming classifiers
7. Taxonomy of classifiers
   → Single classifiers
   → Decision trees and others

→ What will be in the part 2 − drift detectors and ensembles

Data Streams - definition

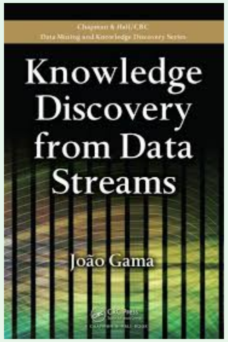
- *"A **data stream** is a potentially unbounded, ordered sequence of data items, which arrive continuously at high-speeds"*

  Springer Encyclopedia of Machine Leaning

- *"It is impossible to control the order in which items arrive, nor is it feasible to locally **store** a stream in its entirety"*

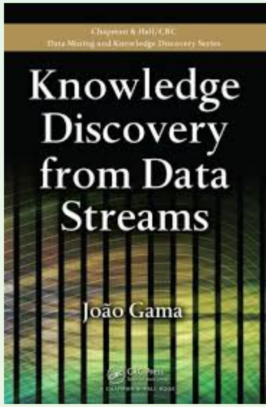
- Other definitions see →

+ Ph.D Thesis of D. Brzeziński (see his WWW)



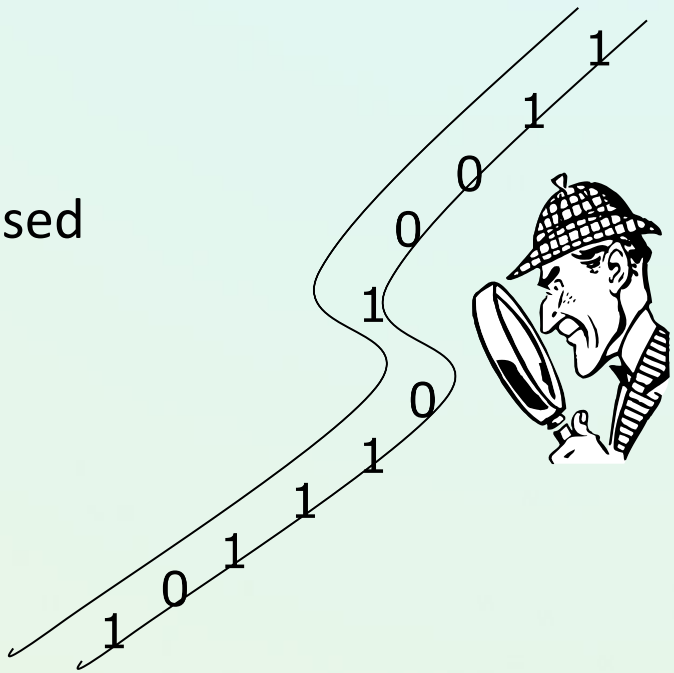
Data stream characteristic

- **Continuous flow** – the data elements arrive online one after another
  - Time intervals between element may vary
  - Each example can be processed only once (single scan)
  - The system has not control over the order of arriving elements
- **Huge volumes of data** (potentially unbounded in size)
- Data arrive at a rapid rate
  - With respect to the computational abilities of the processing system (time is costly)
- Data streams **may evolve** over time
  - Different types of concept drifts



New requirements for data stream algorithms

- Process incrementally an example
  - Inspect it usually only once
- Use a limited amount of memory
  - Streams are often too large to be processed as a whole
- Work in limited time
  - Examples arrive rapidly
- Be ready to predict at any time
- Deal with concept drift
  - When data streams evolve over time



New algorithms than ones known from static classification !

Data streams vs. time series

Both incremental and time dependent

However, there are strong differences

- Multi-dimensional attributes vs. focus on the main among them
- Different predictions
  - See the classification task / further elements of this lecture
- No typical auto-correlations and similar assumptions
- Other view of seasonal changes
- Non-stationary and concept-drifting characteristics
- Computational requirements
- and ⋯

Timestamp	Puis. A (kW)	Puis. R (kVAR)	U 1 (V)	I 1 (A)
…	…	…	…	…
16/12/2006-17:26	5,374	0,498	233,29	23
16/12/2006-17:27	5,388	0,502	233,74	23
16/12/2006-17:28	3,666	0,528	235,68	15,8
16/12/2006-17:29	3,52	0,522	235,02	15
…	…	…	…	…

Previous research efforts

- Incremental learning vs. batch
  - Neural networks (although repeated over epochs)
  - Generalizations of k-NN (Aha's IBL)
  - Incremental Naïve Bayes
- Incremental versions of symbolic knowledge reconstruction
  - Decision trees ID5 (Utgoff)
  - Rule learning (AQ15PM)
  - Clustering – COBWEB (D.Fisher)
- Specific sampling for larger data
  - Windowing for trees (Quinlan C4.5; Catlett)
  - Sampling for k-means or other clustering algorithms

Review of some incremental learners: V.Lemaire et al. A survey on supervised classification on data streams 2015

However, …

- Many of these solutions are just simple incremental learners
  - e.g., neural networks need several passes through data (epochs), time demanding tuning parameters, ..
- Not useful for processing streams:
  - Massive data streams
  - Computational demands (with limited memory, time,..)
  - Adapting to changes in data (non-stationary environments)

Table 3: Properties of incremental learning vs incremental learning on streams (yes=required, no=not required).

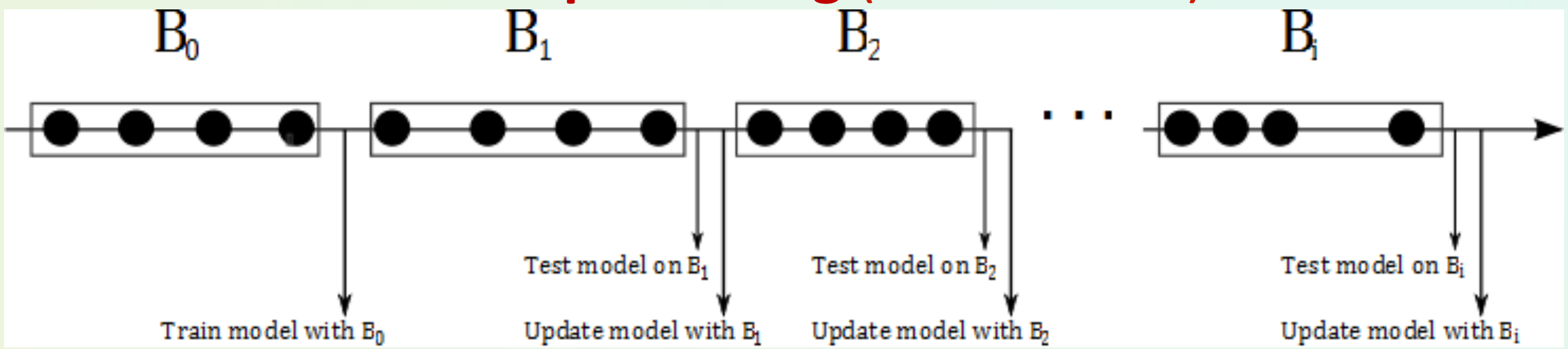
	Incremental	Incremental on streams
Tuning of the learner settings using cross-validation	No	No
Data read just once	Yes	Yes
Post-optimization after learning	Yes	No
Complexity for learning and prediction	Low	Very low
Memory management	Yes	Yes
Handling of the trade-off between accuracy and time to learn	No	Yes
Concept drift handling	No	Yes
Anytime	No	Recommended

V.Lemaire et al. A survey on supervised classification on data streams 2015

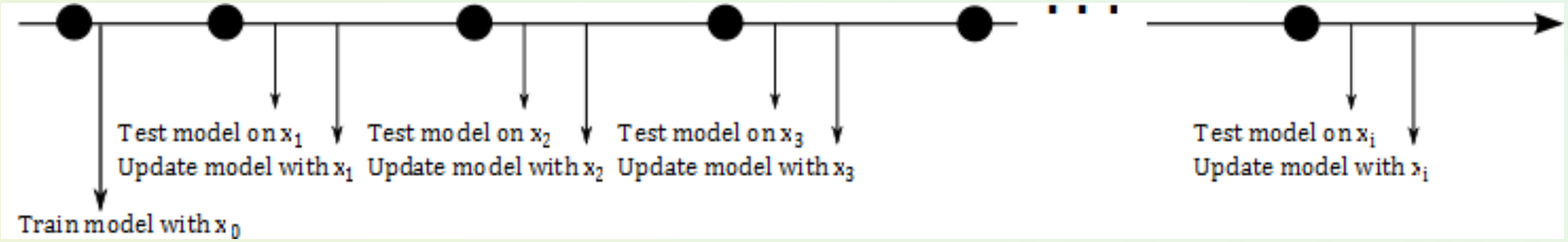
Different processing schemes

Data stream S is a sequence of labeled examples $\boldsymbol{z}_t=(\boldsymbol{x}_t,y_t)$ (t=1,2, …, T); may be considered in blocks

Block processing (data chunks)



Online processing (instances)



Completely labeled examples or partly …?

Labeling frameworks

- **Complete supervised**
  - Relatively immediate access to class labels for each incoming example
  - Labels could be used to evaluate and update the classifier
- Learning with delayed labeling
- Semi-supervised learning
- Unsupervised (initially labeled examples)

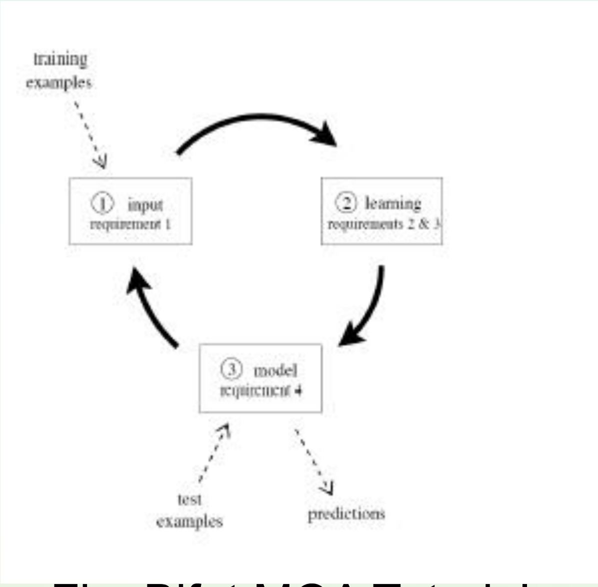


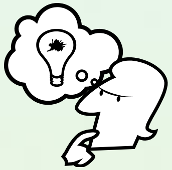
Fig: Bifet MOA Tutorial

Stationary vs. non-stationary data

ML/DM typical assumption:

- Instances are independent and coming from stationary distribution

Is it valid for data streams and changes?



Stationary vs. non-stationary streams

Generally two models of streams:

- Stationary – examples drawn from fixed (albeit unknown) probability distribution
- Non-stationary – data evolve over time

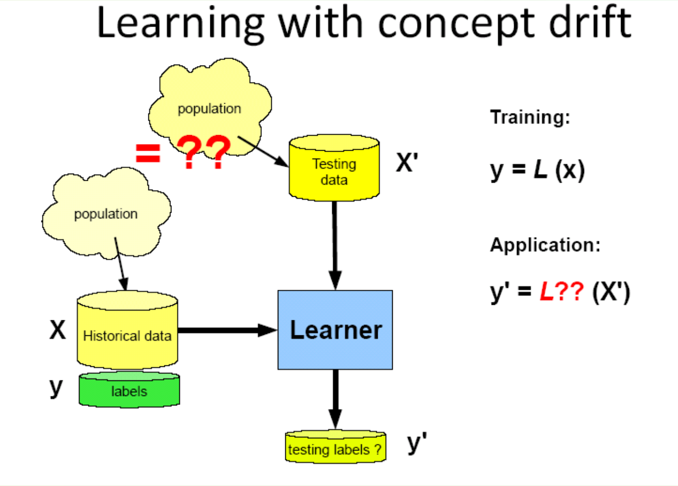
  Concept drift

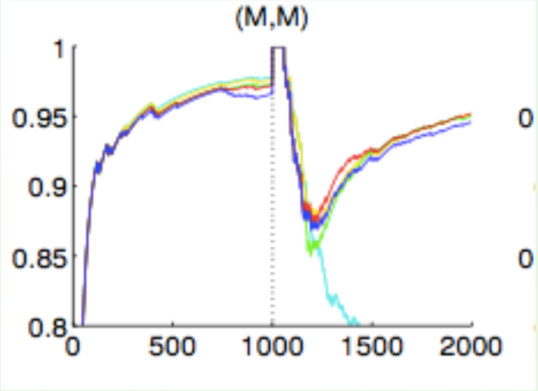
  - Recommendations “interesting literature“ -- from novice to expert
  - “spam email” – new versions arrive
  - Changes in controlling the manufacturing process

Classification in Changing Environments

Concept drift - means that the concept about which data is obtained may shift from time to time, each time after some minimum permanence (def. J.Gama).

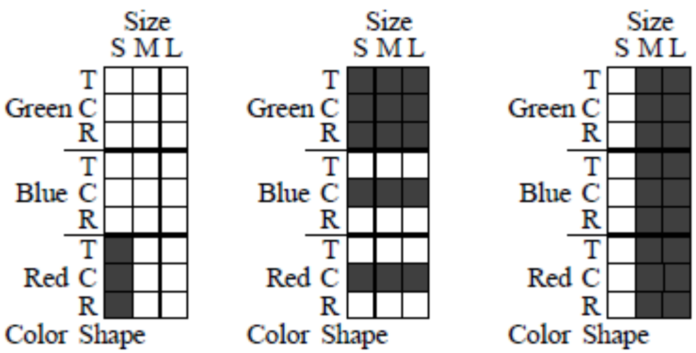
- Reasons – **hidden context** / not available for the learning algorithm in observed attributes (Widmer,Kubat)
- These are not seasonal changes as in time series
- Concept drifts are reflected in the incoming instances and deteriorate predictions of classifiers





Experiences with FLORA rule learning [Widmer,Kubat]

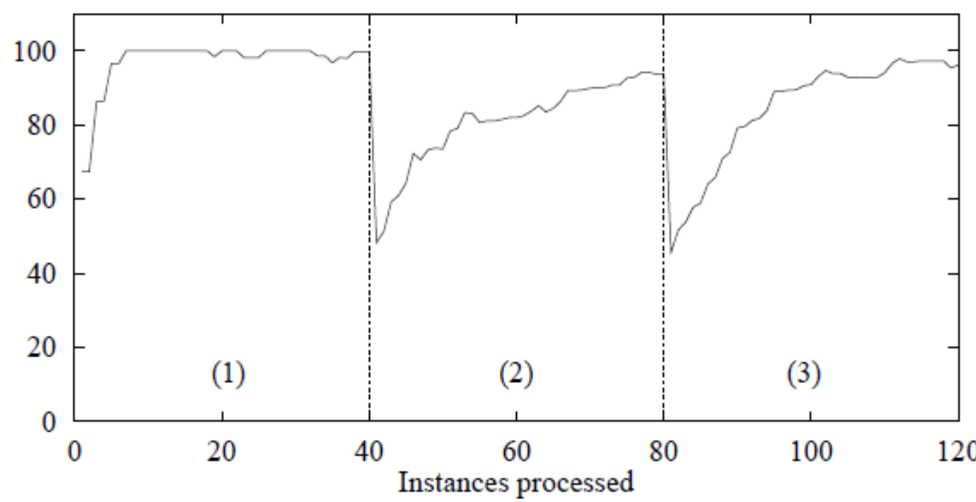
Sudden change → STAGGER problem (synthetic data)



Target concept $t = 1 \ldots 40$.

Target concept $t = 41 \ldots 80$.

Target concept $t = 81 \ldots 120$.



More real life examples

- Analysing customer preferences
- Approving bank loans
  - Financial market changes
- Filtering information
  - What is an interesting book, movie
- Medical decision aiding (disease progression changes in response to med. treatment)
- Predicting estate prices, or other goods

Concept drift applications

	Monitoring and control	Information management	Analytics and diagnostics
		Task	
task	detection, prediction	prediction ranking	prediction classification
input data	sequential	relational transactional	time series sequential relational
incoming	stream	batches	stream iterations
volume	high	moderate	moderate
multiple scans	no/yes	yes	yes
missing values	random	unlikely	systematic
		Environment	
change source	adversary complex	preferences contextual	population
change type	sudden	gradual incremental	incremental reoccurring
expectations	unpredictable	unpredictable predictable	identifiable unpredictable
		Operational settings	
label speed	fixed lag	on demand	real time
ground labels	objective	subjective	objective

.

See: Indre Zliobaite, Mykola Pechenizkiy, and Joao Gama: An overview of concept drift applications. Chapter 4 in N.Japkowicz and J.Stefanowski (Eds), Big Data Analysis: New Algorithms for a New Society, Springer (2016). -> see authors' web pages

What can change

- A data stream S – a sequence $x_t y_t$ (t=1,2, …, T)
  → Consider a **supervised classification.** A class label y_t of this example is available (after some time) and can be used for learning a classifier *C*
  - Joint probability distribution $p^t(x,y)$ at time *t*

- For two distinct points in time t and t+Δ, exist x such that $p^t(x,y) \neq p^{t+\Delta}(x,y)$
  - Component probabilities may change but which ones are the most important?

What can change

- Concept drift - for two distinct points in time t and t+Δ, exist x such that $p^t(x,y) \neq p^{t+\Delta}(x,y)$
- **Real drift** (supervised classification)
  - posterior probability of classes $p(y|x)$ changes
- Virtual drift → changes in incoming data, e.g. p(x), not affecting p(y|x), also drifting priori probabilities
  - May be used in novelty detection or semi-supervised settings

Real vs. virtual drift

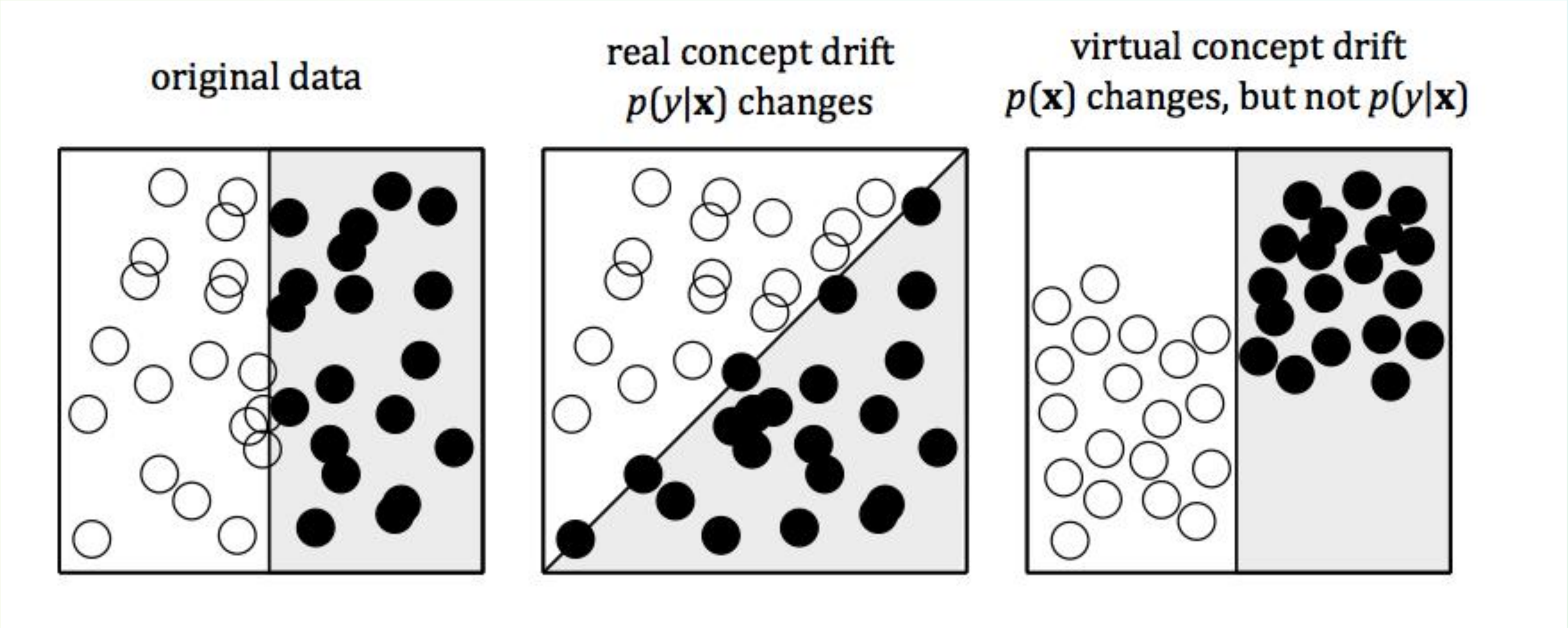
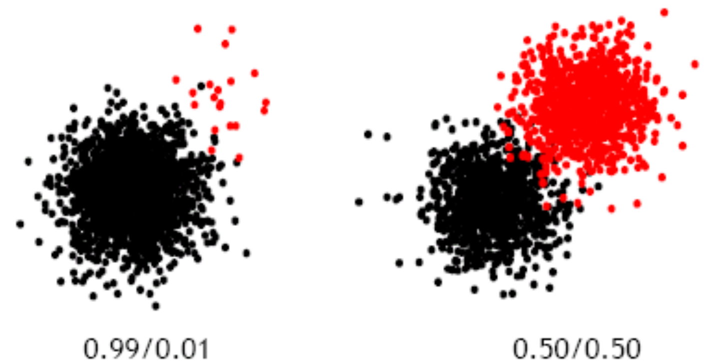


Fig: Dariusz Brzeziński: Block-based and online ensembles for concept drifting data streams. PhD Thesis, Poznań University of Technology, 2015.

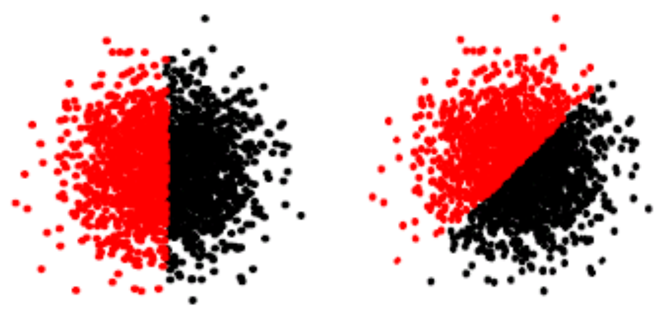
L. Kuncheva's examples (virtual vs. real)

- Changes of prior probabilities p(y) – is it concept drift or rare cases / data shift?



Example: epidemic spread of a disease - symptoms do not change, only the prevalence (priors) do.

- Changes of posterior prob. p(y|**x**)



Example: User preferences in document retrieval. The documents come from the same general pool (same $p(\mathbf{x})$), only their labels (relevant/irrelevant) change.

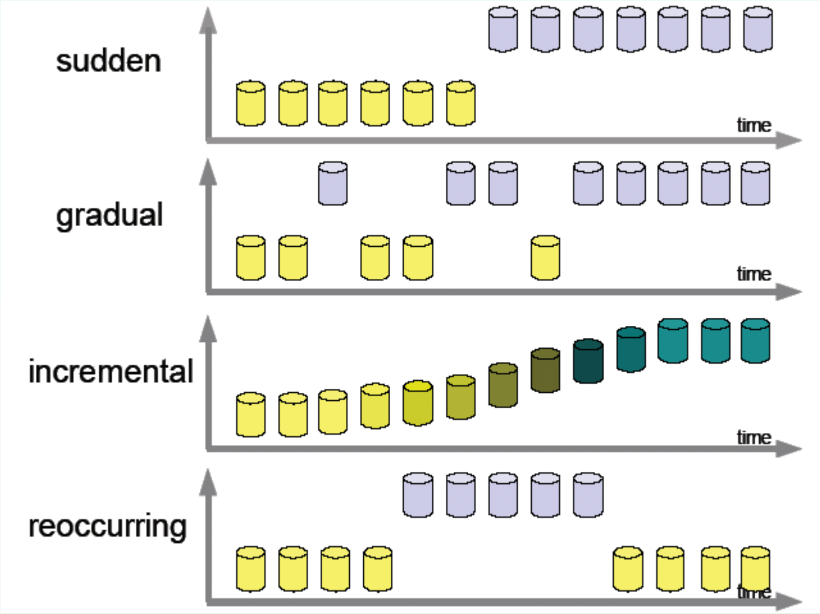
Types of drifts

Stream $S = <S_1,S_2,S_3,\ldots,S_n>$, where subset S_i
 generated by a stationary distribution D_i
 $\rightarrow$ transition between S_j and S_{j+1}

Hidden context of changes

Different types of drifts

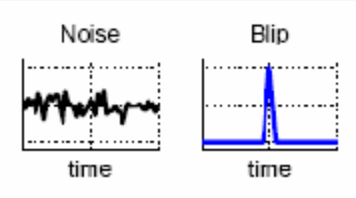
- A **sudden** (abrupt) drift - S_j is suddenly replaced by a different distribution in S_{j+1} ($D_j \neq D_{j+1}$)
- **Gradual** drifts - a slower rate of changes
  - Transition phase where examples from two different distributions are mixed
  - Incremental – many smaller changes
- **Reoccurring** concepts



Not react to blips

Robustness against noise

 Distinguish noise from slowly changing context



Figures - I.Zliobaite

Four basic drifts

Sudden drift

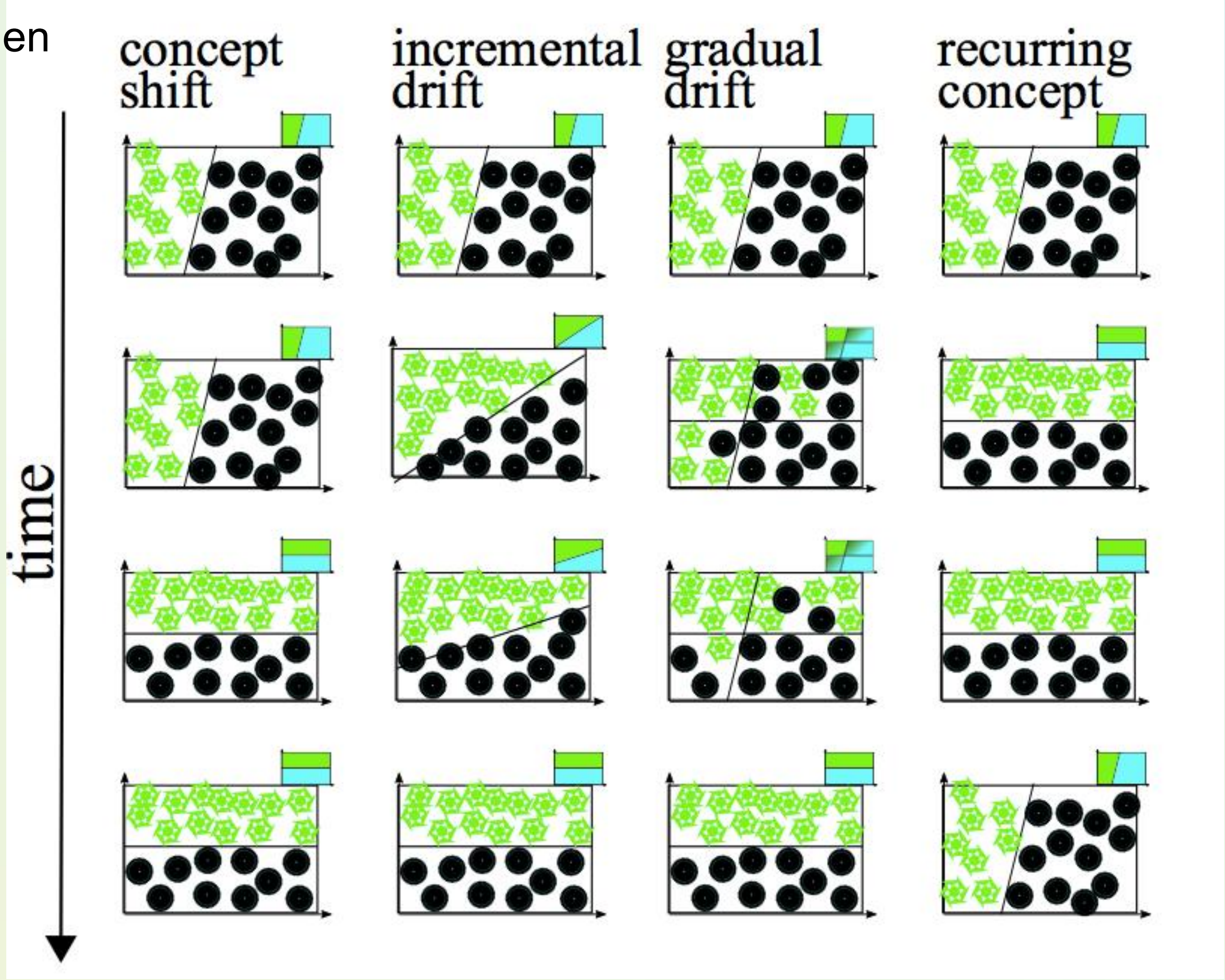


Fig: Ammar Shaker: Novel methods for mining and learning from data streams. PhD Thesis, Paderborn University, 2016.

More on Types of Concept Drifts

Better distinguish types of drifts

- Real concept drift
  - Complete or sub-concept drifts
  - Drift severity (magnitude) – drift change between some time points
- Drift reoccurrence
  - Cyclical drifts – concepts reoccur in a specific order [Tsymbal 2004]; → fixed or varying periodicity
  - Non-cyclical drifts
- Covariance drift (a part of virtual drift)

  $$p^t(x) \neq p^{t+\Delta}(x)$$

- Novel class appearance [Masud et al. 2011]

  $p^t(y=C_l)=0$ for t and $p^{t+\Delta}(y=C_l)>0$

Is the previously learned classifier still valid?

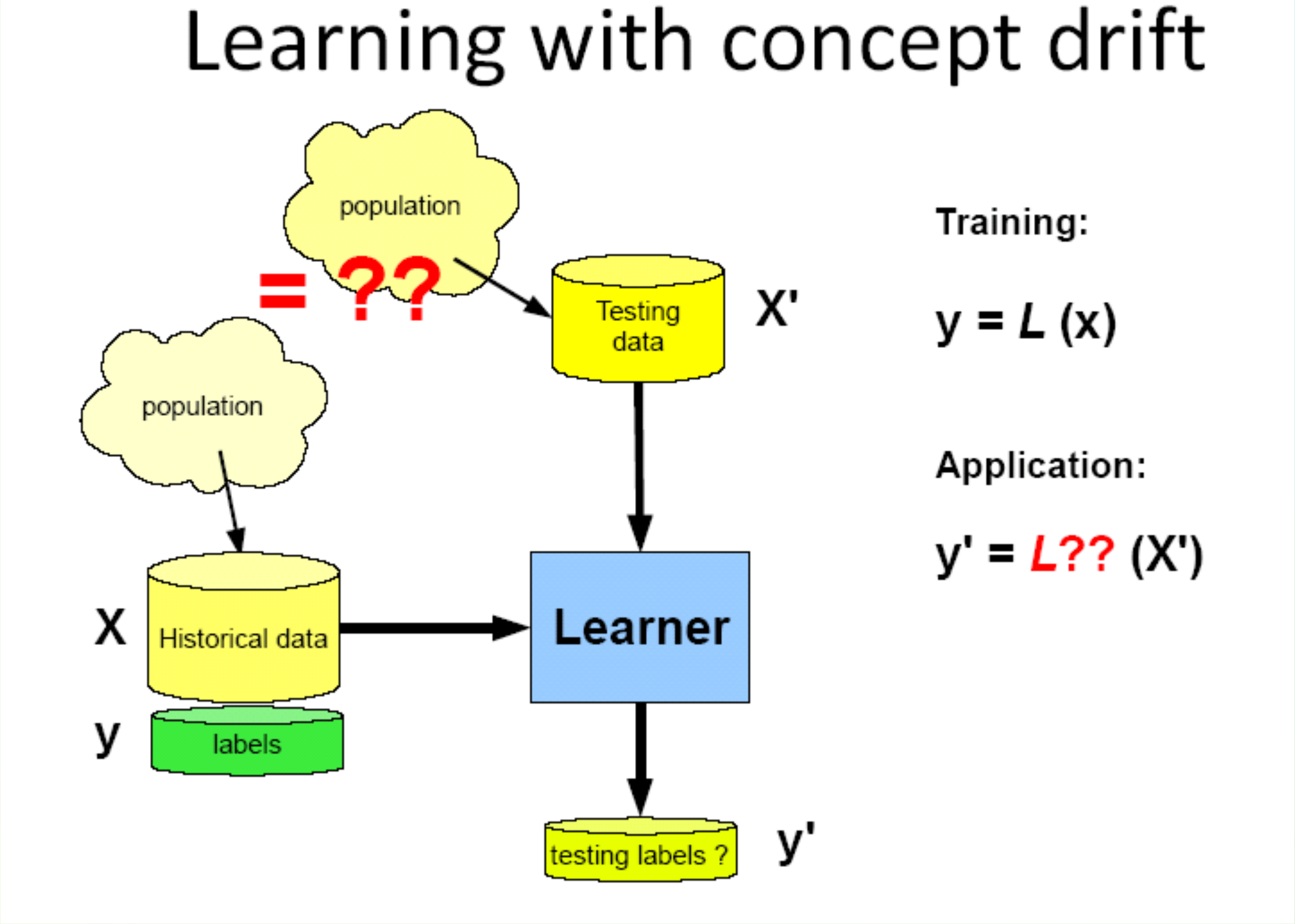


Figure by I.Zliobaite

Data management and forgetting mechanisms

- Necessary to meet time and memory requirements
- Support reaction to changes by eliminating examples from an old concept (forgetting)

Characterize the information stored in memory to maintain a classifier consistent with the actual state of the nature

Different strategies:

- Full vs. partial memory
- No memory – include information about examples in the learned model

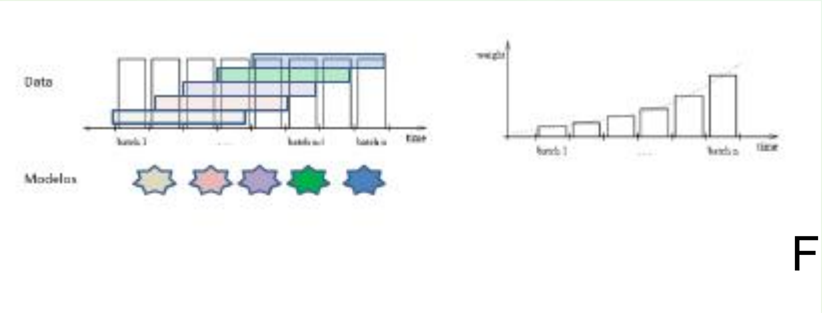


Fig Gama

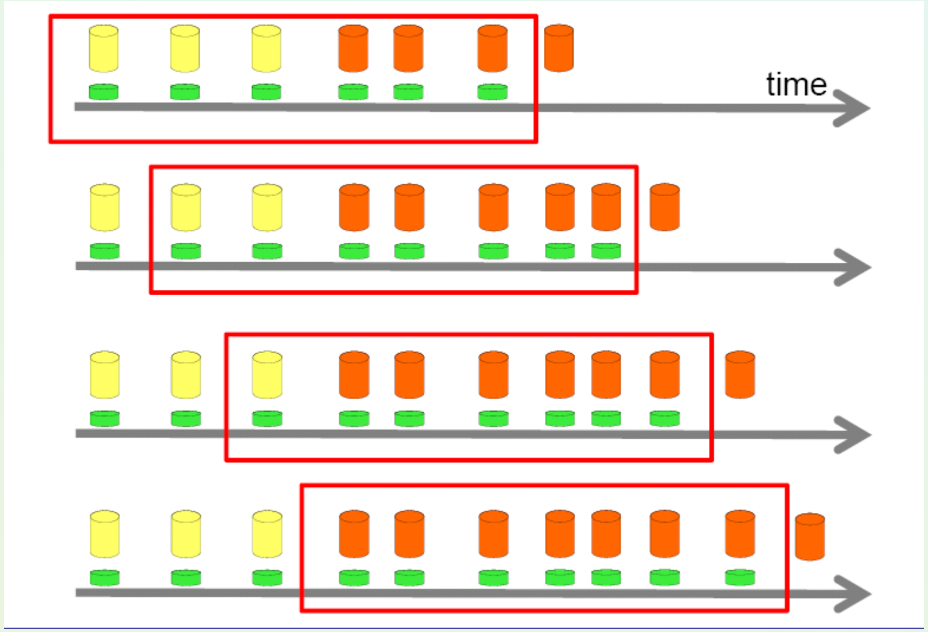
Partial memory - windows

- Store in memory only some examples
- Sliding windows – limit training examples to the most recent ones and consequently discard the oldest ones (FIFO)
- At each time step the learning model induces / updates a classifiers using only the examples that are included in the window

Algorithm 2.2 Basic windowing algorithm

Input: $\mathcal{S}$: data stream of examples
W: window of examples
Output: C: a classifier built on examples in window W

```
initialize window W;
for all examples x^t ∈ S do
    W ← W ∪ {x^t};
    if necessary remove outdated examples from W;
    rebuild/update C using W;
end for
```

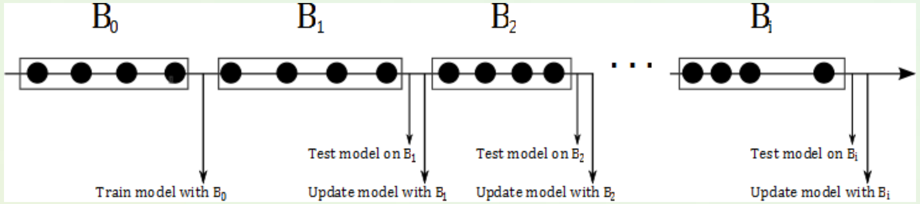


Sliding windows – Forgetting old concepts

Retrain examples with selected examples (but computational costly)

Sliding windows typical for online instance processing

Block (chunk) based mode – more natural dividing a stream in portions. The algorithm may be focused on the recent or latest blocks!



Sliding windows

- How to select the appropriate window size:
  - Too small window
    - A fast adaptation in moments of concept changes
    - Affecting too much computational aspects, in more stable periods
  - Large windows
    - Can not react sufficiently quickly to changes
    - Work well in stable periods
- Fixed vs. adaptive size windows
  - Fixed – simple and may be a baseline
  - Varying the size – usually used with a classifier or a special drift detector
    - Attempt to decrease when changes and increasing in stable periods.

ADWIN adjusting a window size

Bifet – adapting sliding window algorithm, also a drift detector

- The main idea: to compare basic statistics (averages) over two sliding sub windows in main window *W*
  - Whenever two "large enough" sub windows W_0 and W_1 show distinct enough averages, there is difference and the older part of the windows is dropped
- How to compare averages
  - Either a statistical test or a special bound for $|\mu_{W0}-\mu_{W1}| < \varepsilon_{cut}$
- A specific variant of Hoeffding bound

$$\varepsilon_{cut} = \sqrt{\frac{1}{2m}\ln\frac{4|W|}{\delta}} \qquad m = \frac{1}{\frac{1}{|W_0|}+\frac{1}{|W_1|}}$$

- The number of cut splatting points should be reduced – extended ADWIN2; also block growing W too much in stable periods

Bifet A., Avalda R., Kalman filters and adaptive windows for learning in data streams. In Proc. of the 9th int. conf. on Discovery Science, DS. 29–40, 2006.

Selecting examples in a different way

- FISH algorithms [Zliobaite] – consider similarities both in time and attribute space to create a window.
- The distance between x_i and x_j is aggregated in space d^s and in time d^t (with weight coefficients a)

$$D_{ij} = a_1 \cdot d_{ij}^s + a_2 \cdot d_{ij}^t$$

FISH 1 – fixed sized window

For a new example x_{t+1} is builds a window (past examples sorted with D_{it+1})

FISH 2 – using internal leave-one-out classification tune s - the size the window

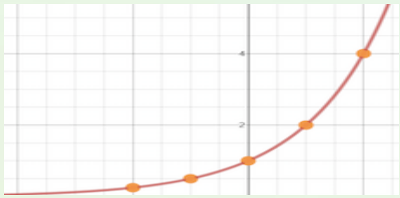
FISH 3 – also search for coefficients in an aggregated

Simple sizes of windows vs. costly optimization

Indre Zliobaite: Combining time and space similarity for small size learning under concept drift. ISMIS Conf. 2009; More in her PhD Thesis on Adaptive training set formation 2010.

Weighting Examples

- Time forgetting
  - Weighting examples (full vs. partial memory)
    - **Full** – store in memory sufficient statistics over all examples
    - Partial – weighted windows
  - Weighting the examples accordingly to their age
  - Oldest examples are less important
- Basic schema [Gama]
  - Assume, the observed statistics S and aggregation function $G(X;S)$
  - At step t, available new example X_t
  - the new value of $S_t = G(X_t; w(t)S(X_{t-1}))$ where $w[0,1]$ is the fading (decay) function



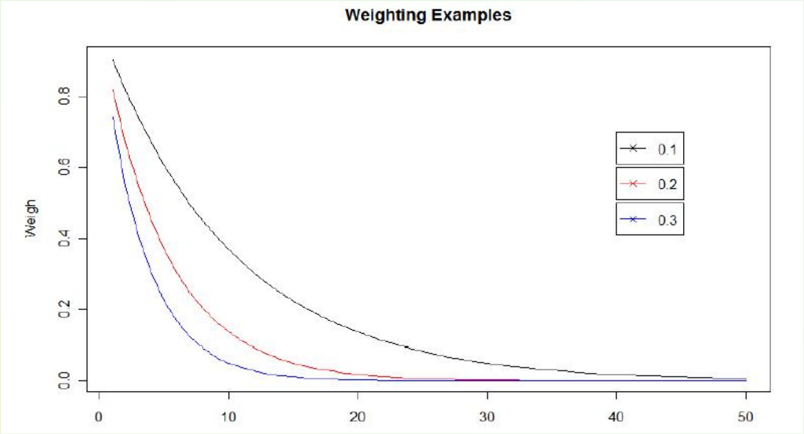
Exponential fading function

Weighting Examples

The role of the oldest examples is decreased by a decay function assigning a weight to each example, e.g.

$$w_1(t) = e^{-\lambda t}, \quad \lambda > 0$$
$$w_2(t) = t^{-\alpha}, \quad a > 0$$
$$w_3(t) = 1 - t/|W|$$

where t refers to the “age” of the example



Fading may concerns blocks

How to adapt = more

- Window forgetting = rather blind adaptation
  - There is no direct change detection
  - Do not provide information about the moment of change and dynamics of the process
- Alternative option – to apply a change detection method and trigger re-training the classifier based on the recent examples

- Wait till drift detectors

Evaluation of streaming classifiers

Main issues

- Evaluation measures
- Estimation techniques
- Comparing classifiers
- Others – synthetic data for adaptability to drifts

Evaluation measures

Basic error / classification accuracy

Easy to calculate also in an online way

Original	Predicted	
	+	-
+	*TP*	*FN*
-	*FP*	*TN*

Specialized (imbalanced classes)

- Sensitivity (Recall) – minority class
- G-mean
- Kappa statistic
- Generalized Kappa (M)
- Prequential AUC (ROC)

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$\kappa = \frac{p_0 - p_C}{1 - p_C}$$

p_0 - accuracy of the classifier
p_C –probability that a chance classifier makes a correct prediction

Combined computational costs (**memory-time**)

How to estimate measures

- Standard ML/DM:
  - independent train and test sets (the same distribution)

    Hold-out and cross validation variants
- Cross validation does not apply in streams

Two alternatives:

- Special hold-out if data is stationary
- Sequential test-and-train

  For each example

  1: make a prediction

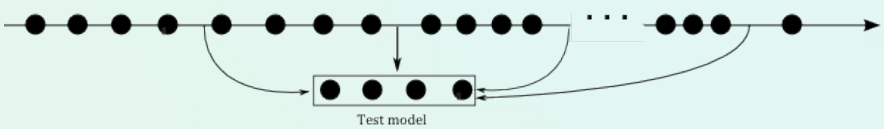
  2: update the classifier, whenever target value is available

The prequential approach over time windows or fading

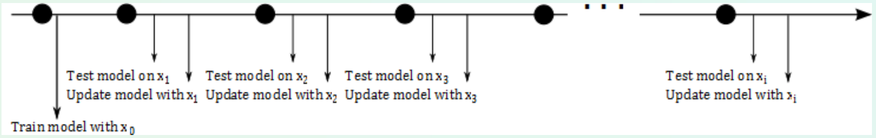
Main estimation techniques

- Holdout

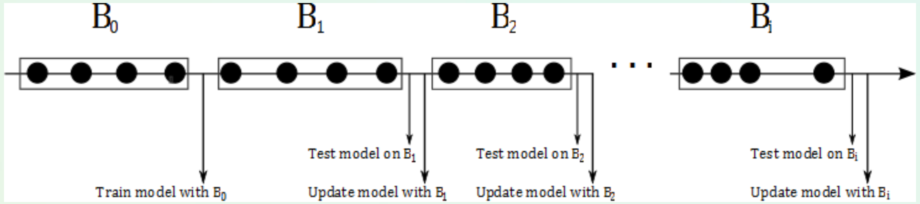
  [Kirkby 2007]



- Interleaved Test-then-train



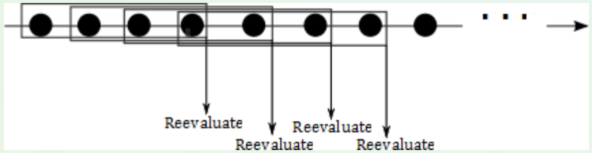
- Block-based evaluation



  [Brzezinski & Stefanowski 2010]

- Prequential with forgetting

  [Gama et al. 2013]



Prequential Evaluation

Basic cumulative (all examples) variant:

- The prequential error, computed at time *i*, is an accumulated sum of a loss between the prediction and target)

$$P_e(i) = \frac{1}{i}\sum_{k=1}^{i} L(y_k, \hat{y}_k) = \frac{1}{i}\sum_{k=1}^{i} e_k$$

- Provides a single number at each time stamp – easy for a learning curve
- The pessimistic estimator of accuracy
- Problematic – influenced by first examples used to train examples / and for evolving data

Prequential with forgetting

Exploit latest examples to estimate

- The prequential error, computed at time i, over a sliding window of size *w* is:

$$P_W(i) = \frac{1}{W}\sum_{k=i-W+1}^{i} L(y_k, \hat{y}_k) = \frac{1}{W}\sum_{k=1 i-W+1}^{i} e_k$$

- A version with fading function *a* (0<< *a* <1) is:

$$P_\alpha(i) = \frac{\sum_{k=1}^{i} \alpha^{i-k} L(y_k \hat{y}_k)}{\sum_{k=1}^{i} \alpha^{i-k}} = \frac{\sum_{k=1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}}$$

Prequential with fading factors

Similar (but easier) forgetting mechanism:

The prequential error, could be $e_i = S_i / n$ where $S_1 = L_1$ and $S_i = L_i + \alpha * S_{i-1}$ but use correction for larger n

$$E_i = \frac{S_i}{N_i} = \frac{L_1 + \alpha \cdot S_{i-1}}{1 + \alpha \cdot N_{i-1}}$$

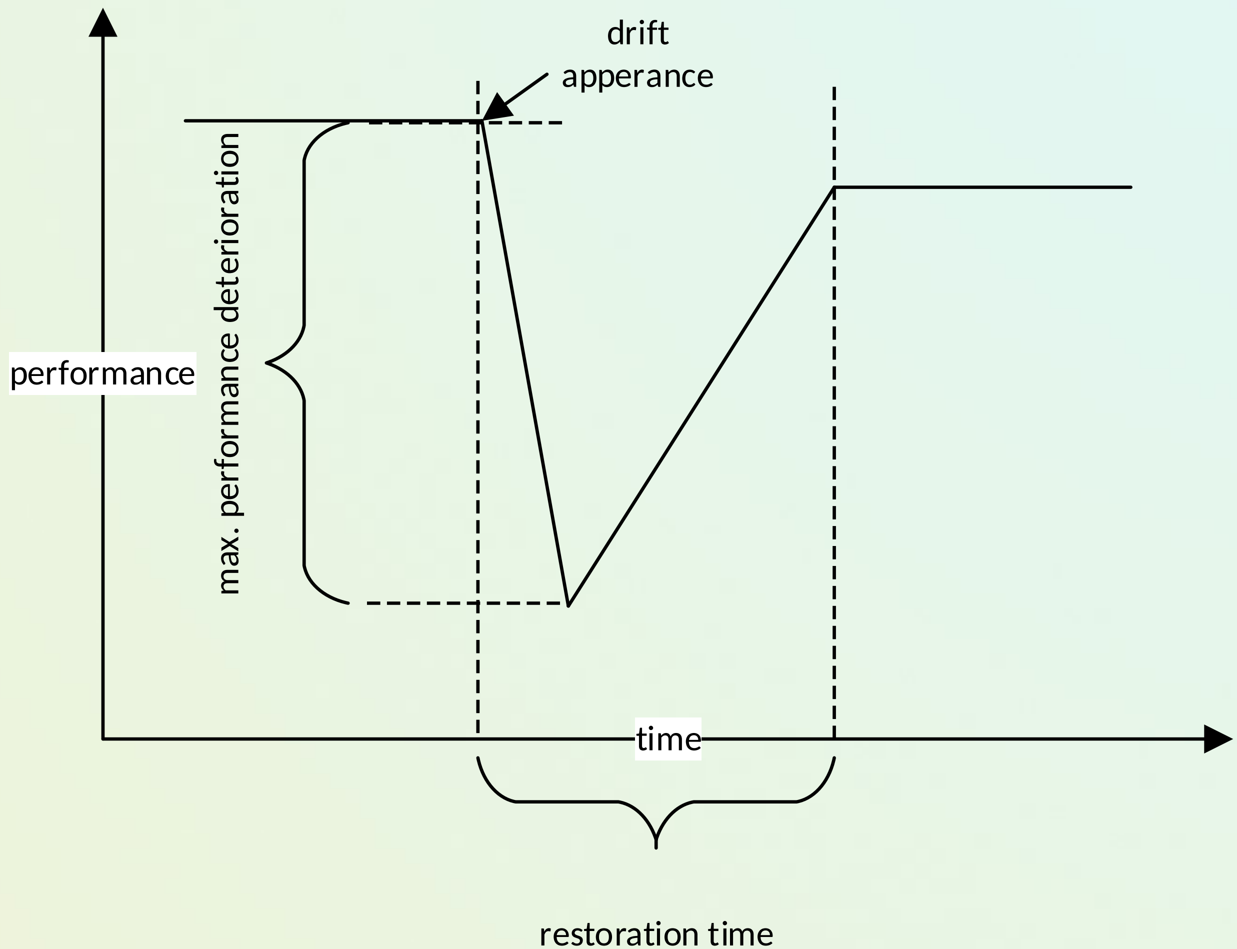
where $n_1=1$ and a is close to 1 (for example 0.9)

Algorithm 3 Update rule for Prequential error estimator using fading factors.

Require: Fading factor α $(0 \ll \alpha \leq 1)$
Require: e_i {/* Loss at example i */}
Ensure: Fading error estimator $P_\alpha(i)$

$S_\alpha(0) \leftarrow 0; N_\alpha(0) \leftarrow 0$ {/* Initialize the error estimate */}
...
{/* Update the error estimate */}
$S_\alpha(i) \leftarrow e_i + \alpha * S_\alpha(i-1)$
$N_\alpha(i) \leftarrow 1 + \alpha * N_\alpha(i-1)$
$P_\alpha(i) = \frac{S_\alpha(i)}{N_\alpha(i)}$
...

Evaluation of dealing with concept drifts



Drift detection and dynamics / classifier deterioration + recovery

Evaluating reactions to drifts

Limited access to real-world data sets with know drifts (and their localizations), e.g. SPAM datasets

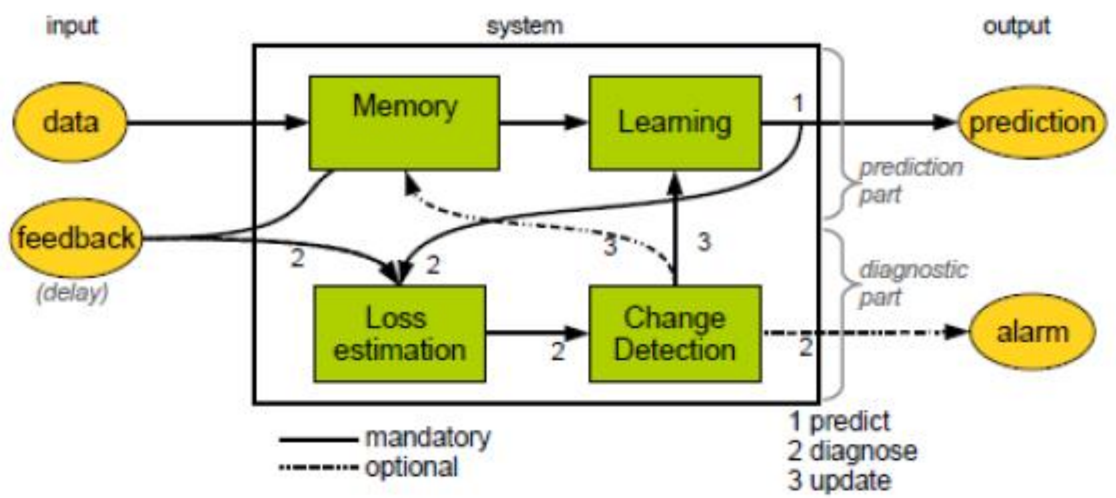
- Synthetic data generators (see MOA), e.g.
  - SEA – sudden concept
  - STAGGER
  - Rotating hyperplane (gradual)
  - RBF
  - Minku’s problems (not in MOA)

- Advanced scenarios, e.g.
  - Recovery analysis [Shaker and Hüllermeier] join 2 real stationary streams into a new with controlled drifts
  - Controlled permutations [Zliobaite]

Generic scheme of online adaptive learning

Recall general requirements

- Detect or adapt to drifts asap.,
- while distinguishing between drift and noise,
- doing so in less time than the arrival of the next instance
- without requiring more than a fixed amount of (memory for) storage.



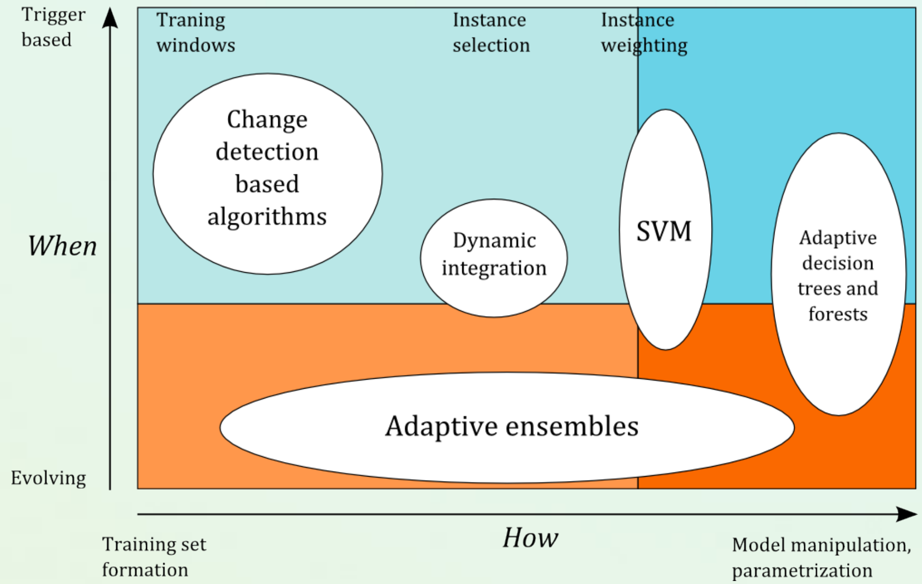
A generic schema for an online adaptive learning algorithm.

Fig. Gama

Categorization of algorithms

- Solutions for stationary or non-stationary streams

- In case of evolving data / non-stationary streams
  - Active (drift triggers) vs. passive (no drift detection but adaptive one)



- Single vs. multiple classifiers (ensembles)

- Different modes of processing examples
  - Online instance by instances vs. block / chunk bases ones

Categorization of learning algorithms

	Triggering	Evolving
Single classifiers	**Detectors**	**Forgetting**
Multiple classifiers	**Contextual**	**Adaptive ensemble**

Bifet A., Gama. J., Pechenizky M., Zliobaite I.: Handling concept drift. Importance, challenges and solutions. PAKDD Tutorial (2011)

Learning algorithms with respect to changes

Triggering

Evolving

Change detect
Re-train

Adapt in
some points

Single
classifiers

Detectors

Forgetting

Windowing
Fading

No or partial
memory

Multiple
classifiers

Contextual

Dynamic ensemble

Maintain
Some memory

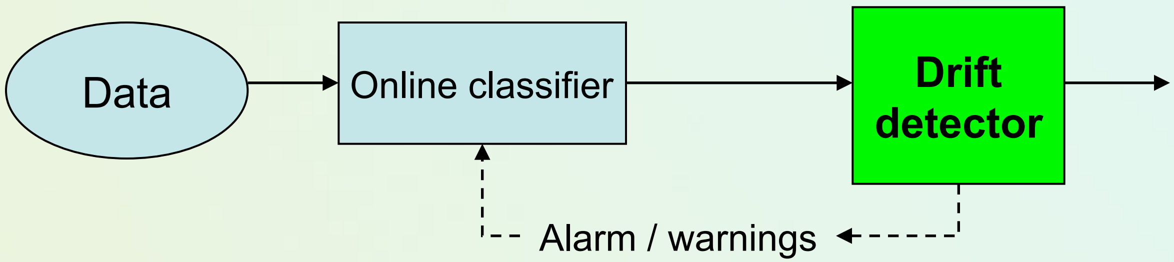
Swap between
models

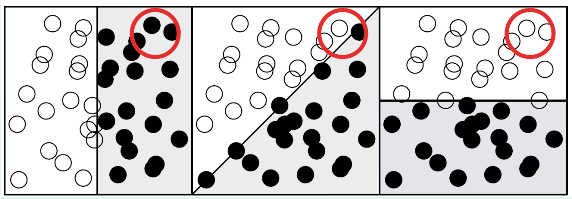
Exploit many models

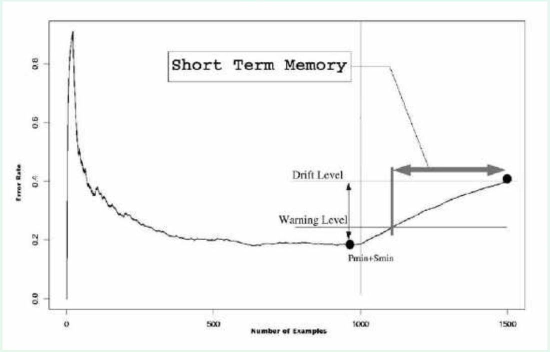
Adaptive
combination

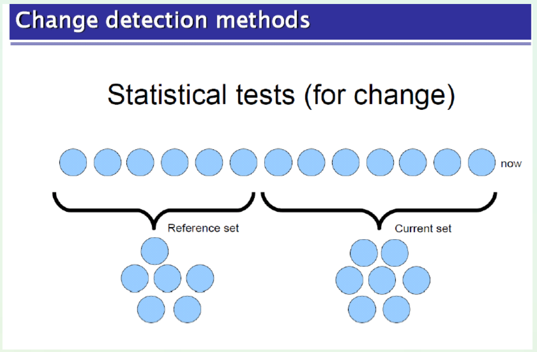
Bifet A., Gama. J., Pechenizky M., Zliobaite I.: Handling concept drift. Importance, challenges and solutions. PAKDD Tutorial (2011)

Triggers – the use of drift detectors









Statistical Process Control

- DDM, EWMA,…

Sequential Analysis

- Cumulative Sum Test, Page-Hinkley test

Monitoring distributions over windows

- ADWIN

Context approaches

More: J.Gama, I.Zliobaite, M.Pechenizkiy, A. Bouchachia: A Survey on Concept Drift Adaptation. ACM Compt. 2013

Wait to the next lecture – a brief review of popular methods

Single classifiers

[Lemaire et al. 2015]

- Decision tress (Hoeffding bounds → VFDT)
- Decision rules (VFDR,FACIL, RILL)
- Naive Bayes
- Lazy learning with K-NN, e.g. IBLStreams
- Incremental SVM
- Online ANN perceptrons

Main versions – for stationary streams

Incremental tress - challenges

- Classic decision tree algorithms assume all training data can be simultaneously stored in main memory
- Disk-based algorithms repeatedly read training data from disk sequentially
  - Prohibitively expensive when learning complex trees
- Goal: design decision tree learners that read incrementally each example <u>at most once</u>, and use a small constant time to process it

Stream context → Hoeffding trees

P. Domingos and G. Hulten: “Mining high-speed data streams” KDD’ 2000

Very influential paper!

- Very Fast induction of Decision Trees, a.k.a. Hoeffding trees (extended)
- Algorithm for efficient inducing trees from massive data streams
  - Reasonable time and memory costs

- With high probability will incrementally construct a tree as good as one generated by static (greedy) algorithms from all examples
- Does not store examples - memory independent of data size
- Does not deal with time change!

Stream context of inducing trees

P. Domingos and G. Hulten: “Mining high-speed data streams” KDD’ 2000

Decision trees for streams:

When to make a split with an attribute?

Basic idea: A small sample of the stream can often be enough to choose the optimal splitting attribute

- Collect sufficient statistics from a small set of examples
- Estimate the merit of each attribute
  - Use **Hoeffding bound** to guarantee that the best attribute is really the best
  - Statistical evidence that it is better than the second best

Hoeffding bound (inequality)

- A result in probability theory that gives an upper bound on the probability for the sum of random variables to deviate from its expected value [V.Hoeffding 1963]
- Hoeffding Bound (Additive Chernoff Bound)

Given: *n* independent observations of a real valued random variable *r*, with range of *R* (must be bounded)

It states with probability 1 - δ (where δ is user-specified) that the true mean of r (μ_r) will not differ from the estimated value by more than ε, i.e.

$$P\left(\bar{r} - \mu_r \leq \varepsilon\right) \geq 1 - \delta$$

$$\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

Hoeffding inequality in trees over streams

- Let G() be the measure for choosing the split attribute in a tree node, e.g. information gain
- Assume G is to be maximized, and let A_1 be the first attribute with highest observed G after seeing n examples, and A_2 be the second-best attribute. Let

$$\Delta G = G(A_1) - G(A_2) \geq 0$$

  be the difference between their observed heuristic values. Then, given a desired δ, the Hoeffding bound guarantees that A_1 is the correct choice with probability $1-\delta$ if *n* examples have been seen at this node and **D(G) > ε.**
- The current sample size is enough to decide on attributes, otherwise the sample size its not enough to make a stable decision
- With *R* and δ fixed, the only variable to change ε is *n*

Hoeffding tree basic algorithm

δ - desired probability level

T := Root leaf with empty statistics- counts n_{ijk};

For i = 1,2; … **do** HTGrow(T,x_i) / for each example in a stream

HTGrow(T,x_i)

Propagate example through tree T till a leaf L

Update statistics n_{ijk} at leaf L

if examples seen so far at L are not all of the same class

then Compute G for each attribute

if G(Best Attr.)- G(2ndBest) > ε then

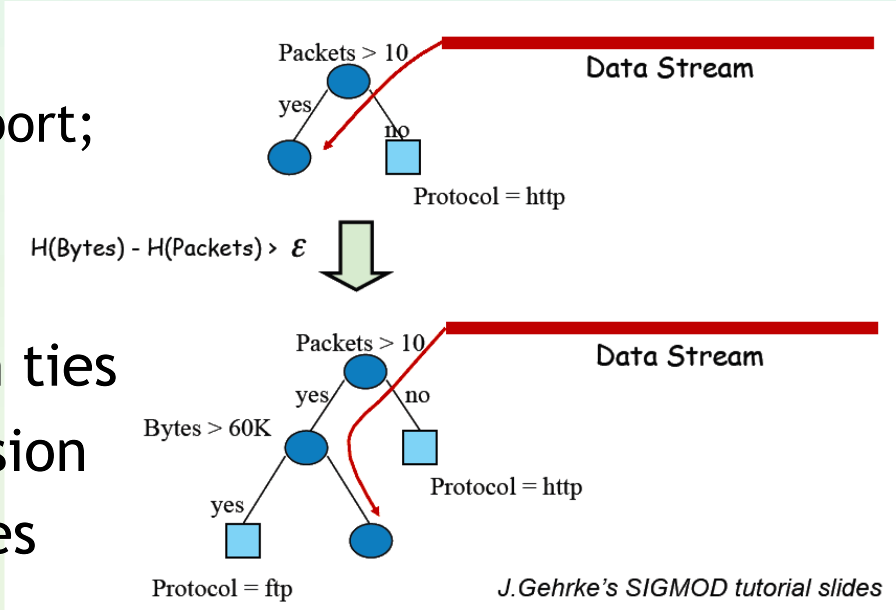
Split leaf L on the best attribute A1

For each expanded branch start a new leaf and statistics

$$\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

Hoeffding Tree: Strengths and Weaknesses

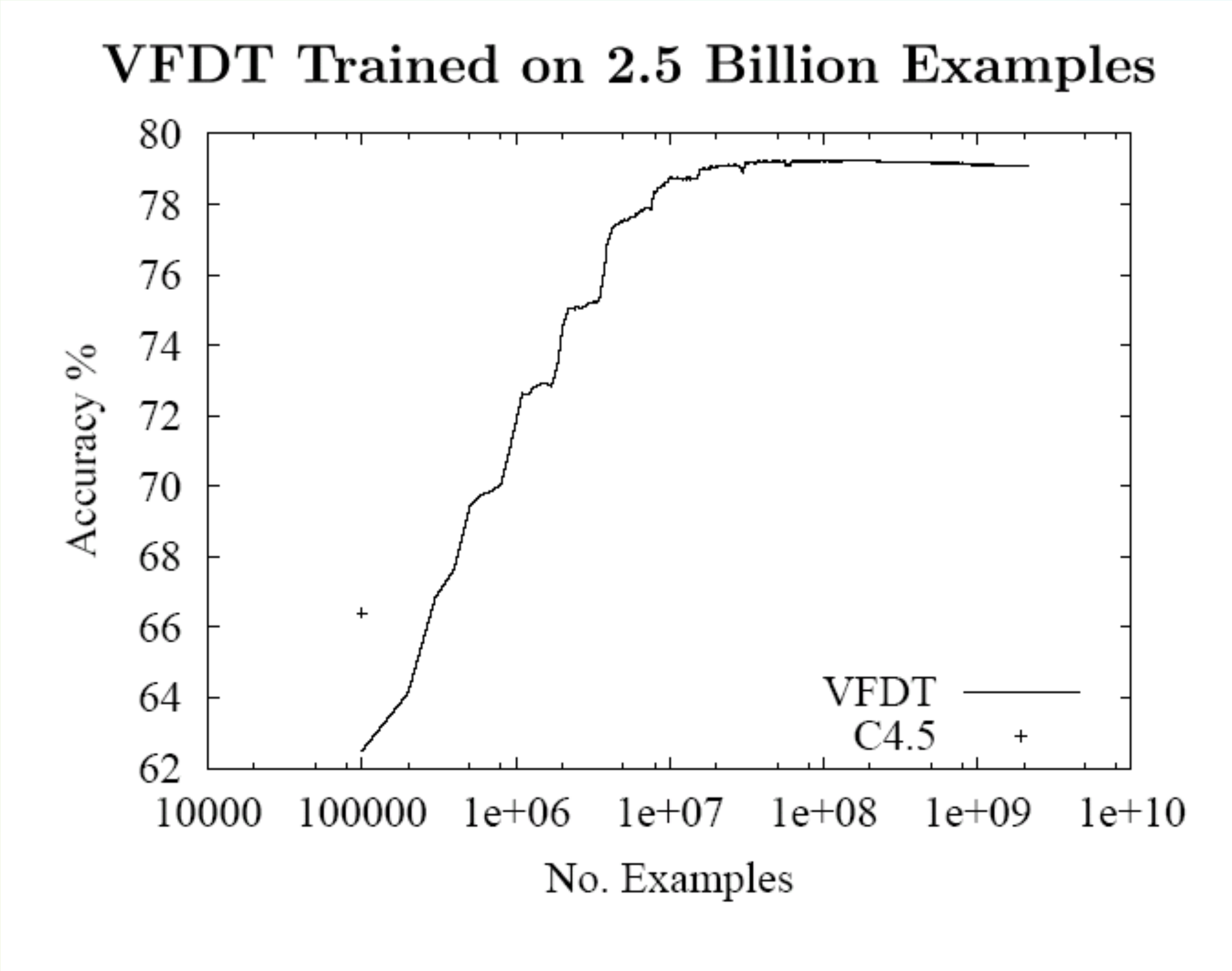
- Strengths
  - Scales better than traditional methods
    - Sublinear with sampling
    - Very small memory utilization
  - Incremental
    - Make class predictions, if necessary
    - New examples are added as they come
  - No need for pruning;
    - Decisions with statistical support;
  - Low overfitting:
- Weakness
  - Could spend a lot of time with ties
  - Memory used with tree expansion
  - Number of candidate attributes



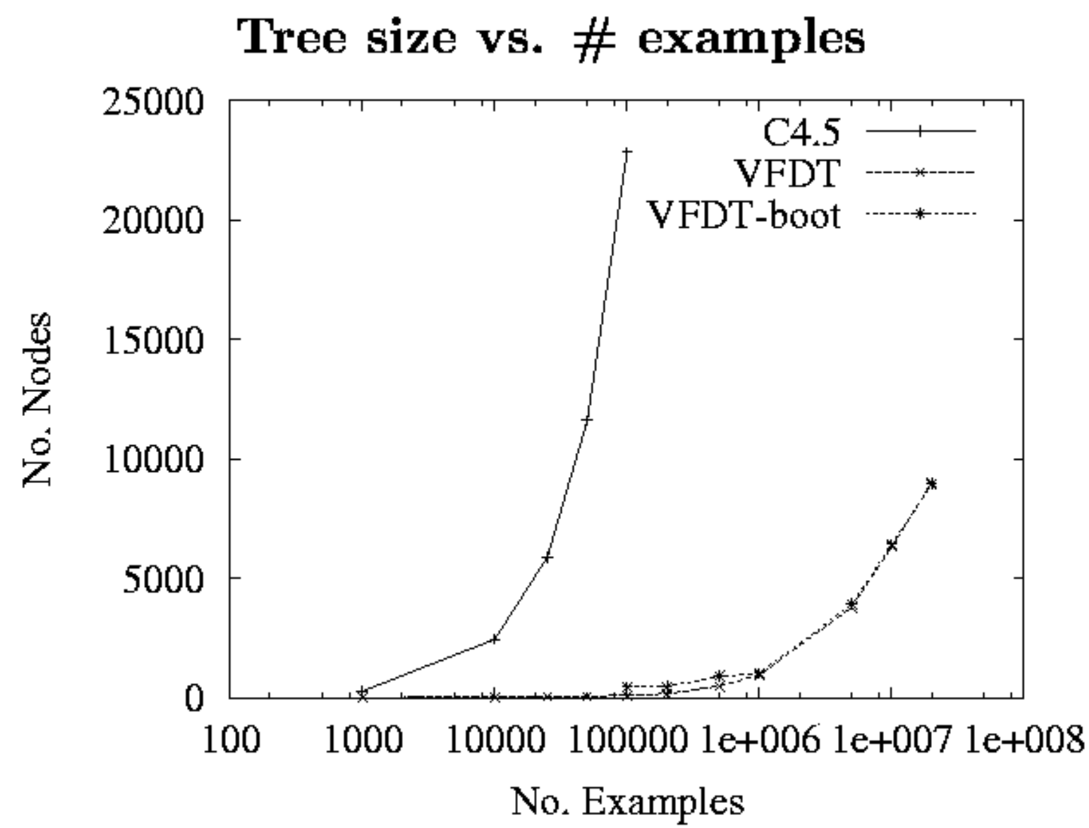
VFDT (Very Fast Decision Tree)

- Modifications to Hoeffding Tree
  - Near-ties, when two best attributes have similar high evaluation G, broken more aggressively G(Best Attr.)- G(2ndBest) < τ
  - G computed every n_{min}
  - Deactivates least promising nodes to save memory
  - Poor attributes dropped
  - New initialization (helps learning curve)
- Compare to Hoeffding Tree: Better time and memory
- Compare to traditional decision tree
  - Similar accuracy
  - Better runtime with 1.61 million examples
    - 21 minutes for VFDT
    - 24 hours for C4.5
- Still does not handle concept drift

Prediction accuracy vs. # examples



Experimental analysis of a tree size



Different versions of VFDT (2 special initilization)

Extensions of VFDT

IADEM [G. Ramos, J. del Campo, R. Morales-Bueno 2006]

- Better splitting and expanding criteria

VFDTc [J. Gama, R. Fernandes, R. Rocha 2006], UFFT [J. Gama, P. Medas 2005]

- Dealing with continuous attributes by special Btrees or Univariate Quadratic Discriminant (UFFT)
- Naive Bayes at inner nodes and leaves
- Short term memory window for detecting concept drift
- Different splitting and expanding criteria

CVFDT [G. Hulten, L. Spencer, P. Domingos 2001]

The adaptation of Hoeffding bound – some criticism, see L. Rutkowski et al.

CVFDT

- Concept-adapting VFDT
  - Mining Time-Changing Data Streams. Hulten, Spencer, Domingos, KDD 2001
- Goal
  - Classifying concept-drifting data streams
- Approach
  - Incorporate "windowing"
  - Monitor changes of information gain for attributes.
  - If change reaches threshold, generate alternate subtree with new "best" attribute, but keep on background.
  - Replace if new subtree becomes more accurate.

- There are alternative approaches
  - Adaptive Trees [Bifet]
    - Replace frequency statistics counters by estimators don't need a window
    - change the way of checking the substitution of alternate subtrees, using a change detector (ADWIN)
  - Gama et al. – comparing distributions

Next inspirations

Regression (Model Trees):

- E. Ikonomovska, J. Gama, S. Dzeroski: Learning model trees from evolving data streams. Data Min. Knowl. Discov. 2011

Rules (VFDR):

- J. Gama, P. Kosina: Learning Decision Rules from Data Streams, IJCAI 2011

Multiple classifiers:

- A. Bifet, E. Frank, G. Holmes, B. Pfahringer: Ensembles of Restricted Hoeffding Trees. ACM TIST; 2012

Other …

Very Fast Decision Rules [Kosina, Gama]

- Rules potentially more comprehensive than larger trees
- Generic scheme of specialization rules + Hoeffding bound

Algorithm 1: VFDR: Rule Learning Algorithm.

input : S: Stream of examples
N_{min}: Minimum number of examples
$ordered\_set$: boolean flag
output: RS: Set of Decision Rules
begin
- Let $RS \leftarrow \{\}$
- Let $default\,rule\ \mathcal{L} \leftarrow \emptyset$
- **foreach** *example* $(x, y_k) \in S$ **do**
  - **foreach** *Rule* $r \in RS$ **do**
    - **if** *r covers the example* **then**
      - Update sufficient statistics of Rule r
      - **if** *Number of examples in* $\mathcal{L}_r > N_{min}$ **then**
        - $r \leftarrow ExpandRule(r)$
      - **if** *ordered_set* **then**
        - BREAK
  - **if** *none of the rules in RS trigger* **then**
    - Update sufficient statistics of the empty rule
    - **if** *Number of examples in* $\mathcal{L} > N_{min}$ **then**
      - $RS \leftarrow RS \cup$ ExpandRule($default\ rule$)

Algorithm 2: ExpandRule: Expanding one Rule.

input : r: One Rule
H: Split evaluation function;
δ: is one minus the desired probability of choosing the correct attribute;
output: $r\prime$: Expanded Rule
begin
- Let h_0 the entropy of the class distribution at $\mathcal{L}_r$
- Compute $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$ (Hoeffding bound)
- **if** $(h_0 > \epsilon)$ **then**
  - **foreach** *attribute* X_i **do**
    - Let h_{ij} be the $H()$ of the best split based on attribute X_i and value v_j
    - **if** $h_{ij} < h_{best}$ *and* $n_{ij} > 0.1 * n$ **then**
      - Let $h_{best} \leftarrow h_{ij}$
  - **if** $(h_0 - h_{best} > \epsilon)$ **then**
    - Extend r with a new condition based on the best attribute $X_a = v_j$
    - Release sufficient statistics of $\mathcal{L}_r$
    - $r \leftarrow r \cup \{X_a = v_j\}$
- **return** r

- AVFDR – extra pruning of rules with drift detectors

Other rule learning algorithms

FLORA Family

- Gerhard Widmer, M.Kubat,Learning in the presence of concept drift and hidden contexts, Machine Learning, 1996.

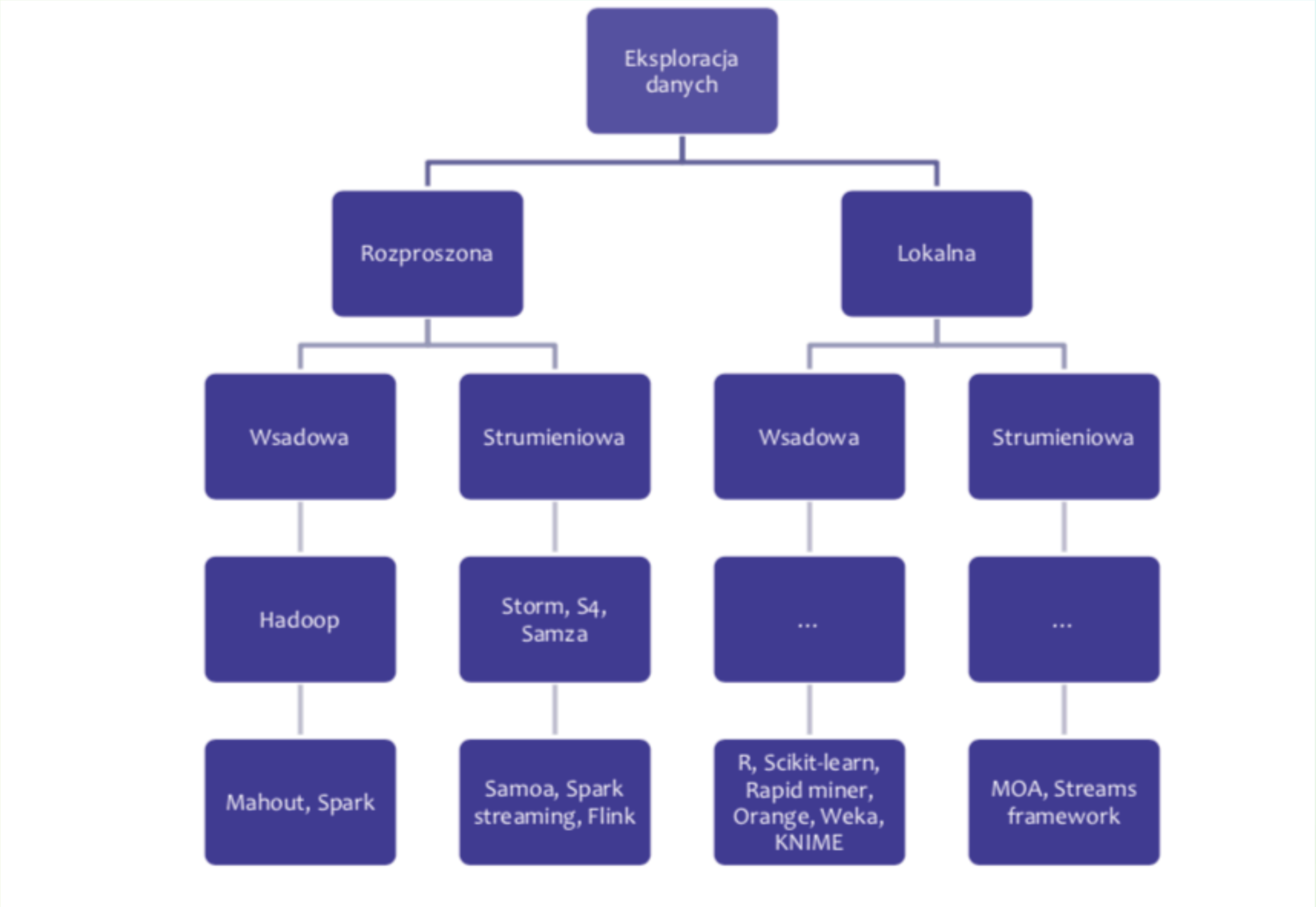
FACIL

- Francisco J. Ferrer-Troyano,Jesús S. Aguilar-Ruiz,José Cristóbal Riquelme Santos, Discovering decision rules from numerical data streams, Proc. ACM Symp. Applied Computing, 2004.

RILL

- Magdalena Deckert, Jerzy Stefanowski, RILL: algorithm for learning rules from streaming data with concept drift, Proc. ISMIS 2014.

Looking for software support



Not real streams

Fig. D.Brzezinski

Do we have software support?

MOA framework/software

For traditional supervised learning WEKA, RapidMiner, R and other open-source DM libraries are popular

For streaming settings MOA: http://moa.cs.waikato.ac.nz/

- **Massive Online Analysis** software environment for implementing algorithms and running experiments for online learning from evolving data streams.;
- includes a collection of offline and online methods for online learning (boosting, bagging, Hoeffding Trees) with or without explicit change detection;
- tools for evaluation;
- bi-directional interaction with WEKA

WEKA
The University of Waikato

Look at MOA Web page

moa

BOOK NEWS DOWNLOADS COMMUNITY DOCUMENTATION

Classifiers

The classifiers implemented in MOA are the following:

- Bayesian classifiers
  - Naive Bayes
  - Naive Bayes Multinomial
- Decision trees classifiers
  - Decision Stump
  - Hoeffding Tree
  - Hoeffding Option Tree
  - Hoeffding Adaptive Tree
- Meta classifiers
  - Bagging
  - Boosting
  - Bagging using `ADWIN`

News

How to use MOA in Docker

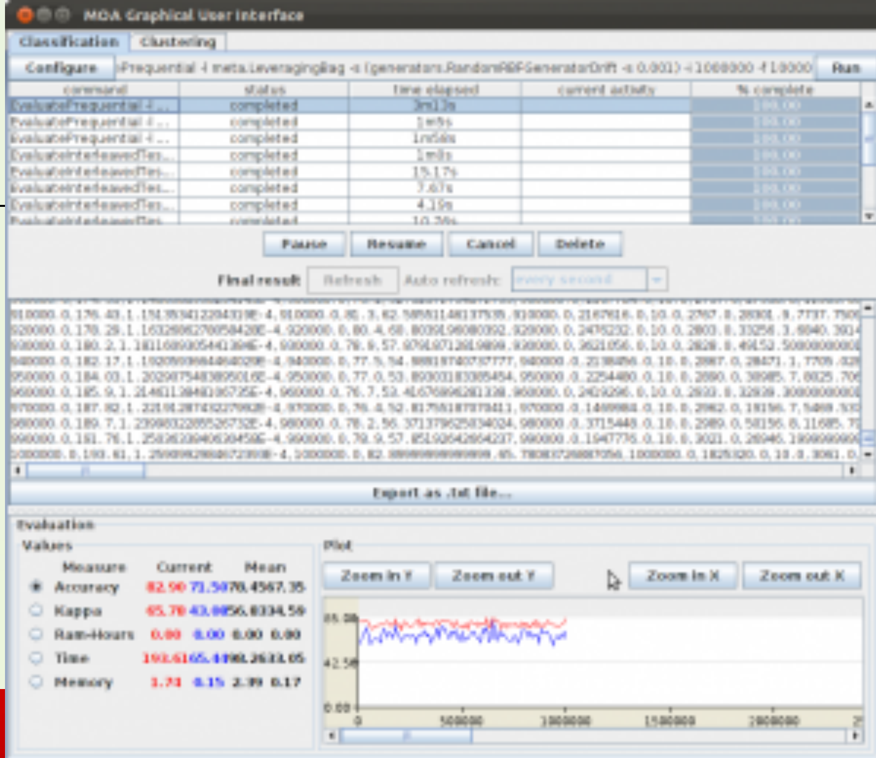
New “LITE” Mode for Beginners

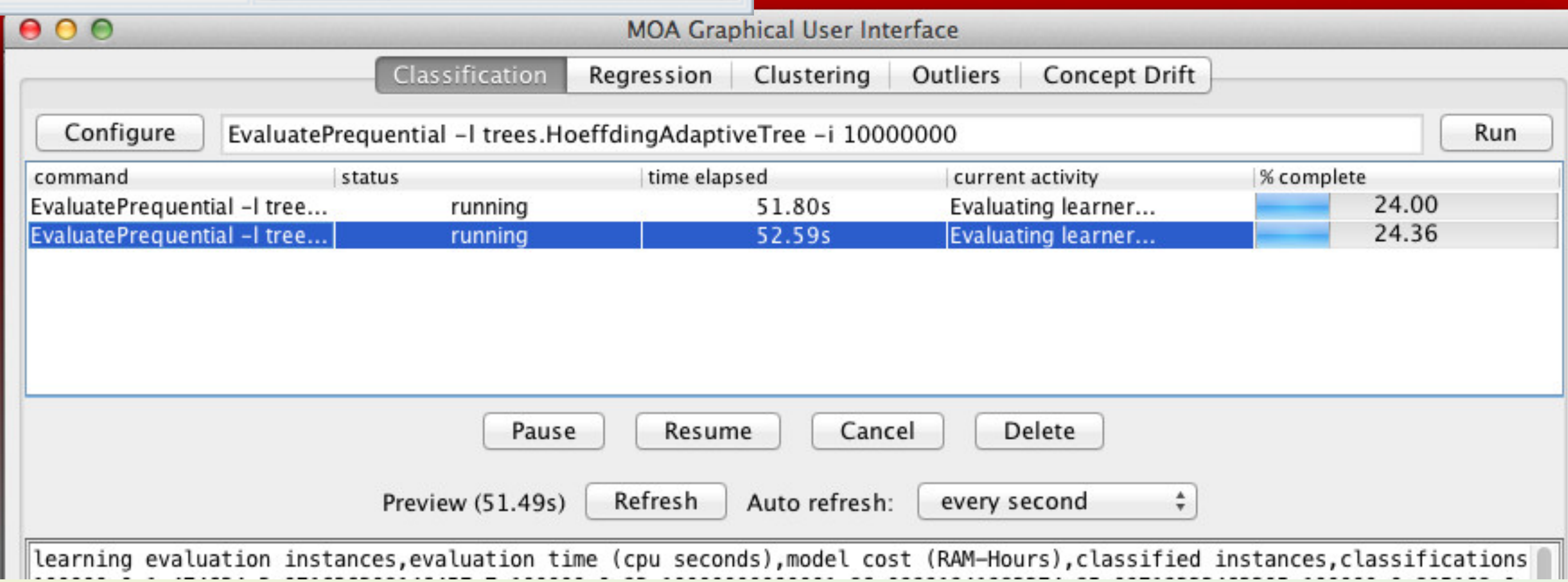
New Release of MOA 19.04

How to use Jupyter Notebooks with MOA

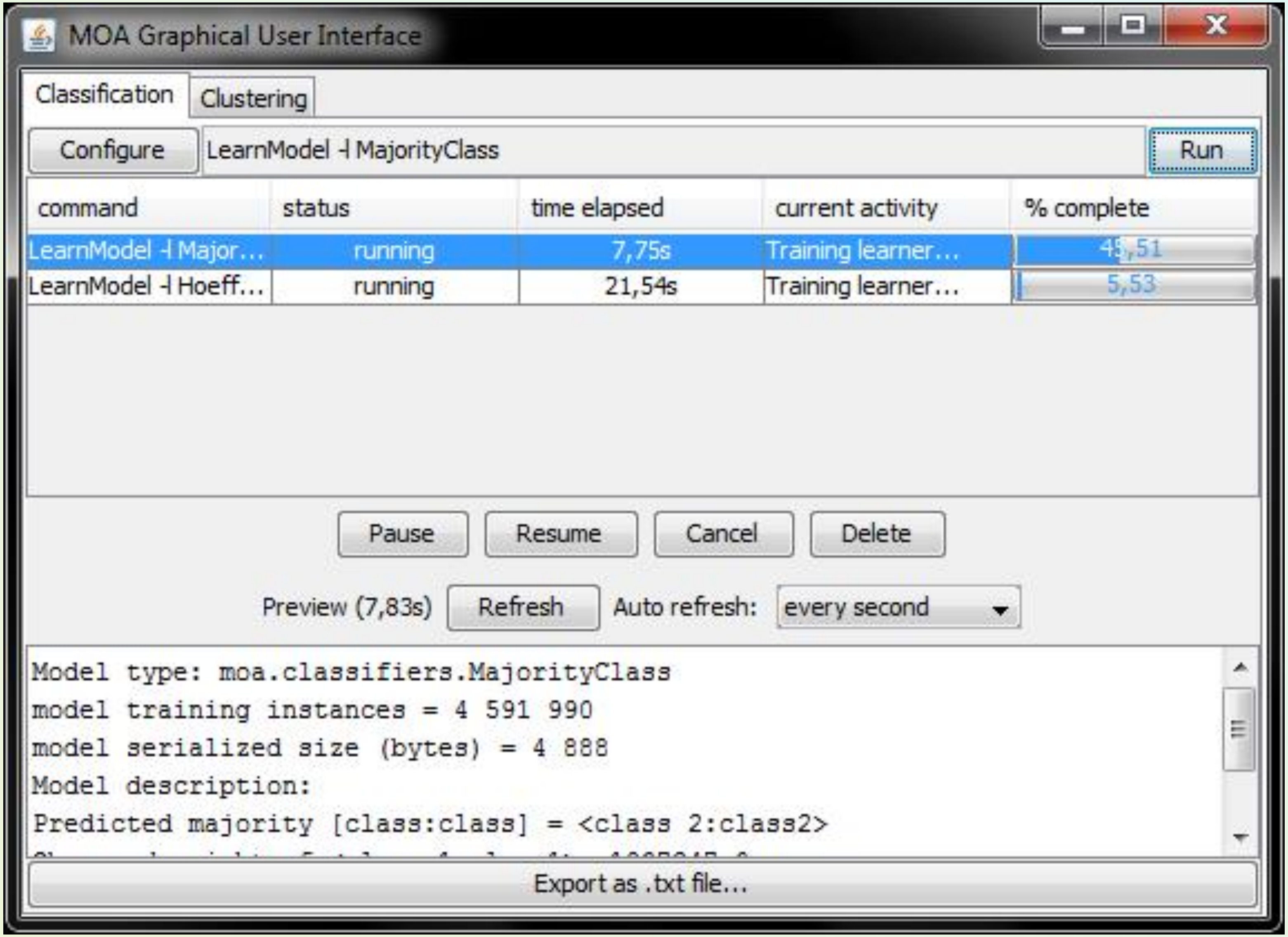
Online Comments Available for the MOA Book

moa


MOA Graphical User Interface
Classification
Clustering
Configure
Run
command
status
time elapsed
current activity
% complete
Pause
Resume
Cancel
Delete
Final result
Refresh
Auto refresh:
Export as .txt file...
Evaluation
Values
Measure
Current
Mean
Accuracy
Kappa
Ram-Hours
Time
Memory
Plot
Zoom in Y
Zoom out Y
Zoom in X
Zoom out X


MOA Graphical User Interface
Classification
Regression
Clustering
Outliers
Concept Drift
Configure
EvaluatePrequential -l trees.HoeffdingAdaptiveTree -i 10000000
Run
command
status
time elapsed
current activity
% complete
EvaluatePrequential -l tree...
running
51.80s
Evaluating learner...
24.00
EvaluatePrequential -l tree...
running
52.59s
Evaluating learner...
24.36
Pause
Resume
Cancel
Delete
Preview (51.49s)
Refresh
Auto refresh:
every second
learning evaluation instances,evaluation time (cpu seconds),model cost (RAM-Hours),classified instances,classifications

MOA – an open source framework for massive data and data streams



See more at Waikato Univeristy web page

Some references

Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. Moa: Massive online analysis. Journal of Machine Learning Research, 11:1601–1604 (2010).

Bifet A., Gama. J., Pechenizky M., Zliobaite I.: Handling concept drift. Importance, challenges and solutions. PAKDD Tutorial (2011)

Bifet A., Avalda R., Kalman filters and adaptive windows for learning in data streams. In Proc. of the 9th int. conf. on Discovery science, DS. 29–40, 2006.

Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. Learning in nonstationary environments: A survey. IEEE Computational Intelligence Magazine, 10(4):12–25. (2015).

Gama, J. Knowledge Discovery from Data Streams. Chapman and Hall/CRC. (2010).

João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, Learning with drift detection, In AnaL.C. Bazzan and Sofiane Labidi, editors, Advances in Artificial Intelligence – SBIA 2004.

João Gama, Raquel Sebastião, Pedro P. Rodrigues, On evaluating stream learning algorithms, Machine Learning, vol. 90, no. 3, pp. 317–346, 2013.

Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the 7th KDD (2001) 97-106;

V. Lemaire, Ch. Slaperwyck, A. Bondu, A Survey on Supervised Classification on Data Streams,

Lecture Notes in Business Information Processing, vol. 205, pp. 88-125, Srpinger, 2015

Petr Kosina, João Gama, Very fast decision rules for classification in data streams, Data Min. Knowl. Discov., vol. 29, no. 1, pp. 168–202, 2015.

Leszek Rutkowski, Lena Pietruczuk, Piotr Duda, Maciej Jaworski, Decision trees for mining data streams based on the McDiarmid's bound, IEEE TFKE, vol. 25, no. 6, pp. 1272-1279, 2013.

Gerhard Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, In Machine Learning, pp. 69-101, 1996.

End of part 1, …

What you will hear – mainly streaming adaptive ensembles