Evaluating Data Stream Mining Algorithms

João Gama
LIAAD-INESC Porto, University of Porto, Portugal

December 12, 2014

Outline

The State-of-the-art

How can we tell if one algorithm can learn better than another?

- Design an experiment to measure the accuracy of the two algorithms.
- Run multiple trials.
- Compare the samples not just their means. Do a statistically sound test of the two samples.
- Is any observed difference significant? Is it due to true difference between algorithms or natural variation in the measurements?

The State-of-the-art

J. Demsar, *Statistical Comparisons of Classifiers over Multiple Data Sets*, JMLR, 2006

In depth study of several statistical tests for comparing multiple classifiers in multiple datasets.

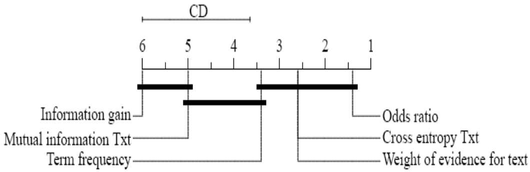


Figure 2: Comparison of recalls for various feature selection measures; analysis of the results from the paper by Mladenić and Grobelnik (1999).

The problem

Suppose we are given a large data set and a classifier. The classifier may have been constructed using part of this data, but there is enough data remaining for a separate test set. Hence we can measure the accuracy and construct a confidence interval.

T. Diettrich *Approximate Statistical Tests, 98*

In data streams scenario we are glutted of data!
Is the sample approach enough?

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

In 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- **forgetting** outdated information;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.
How to evaluate decision models that evolve over time?

Spatio-Temporal Data

- Data are made available through *unlimited streams* that continuously flow, eventually at high-speed, over time.
- The underlying *regularities may evolve over time* rather than be stationary.
- The data can no longer be considered as *independent and identically distributed*.
- The data is now often *spatially as well as time situated*.

Learning from Data Streams: 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
- Distributed processing distributed streams

Bounded Resources

Learning Algorithms are limited by:

- Limited computational power;
- Fixed amount of memory;
- Limited communications bandwidth;
- Limited battery power.

Data is characterized by:

- High-speed
- non-stationary distributions

Outline

Metrics for Evaluation in Data Streams

- **Loss**: measuring how appropriate is the current model to the actual status of the nature.
- **Memory used**: Learning algorithms run in fixed memory. We need to evaluate the memory usage over time, and the impact in accuracy when using the available memory.
- **Speed of Processing examples**: Algorithms must process the examples as fast if not faster than they arrive.

Environments - Memory constrains

R. Kirkby, *Improving Hoeffding Trees*, PhD Thesis, University of Waikato

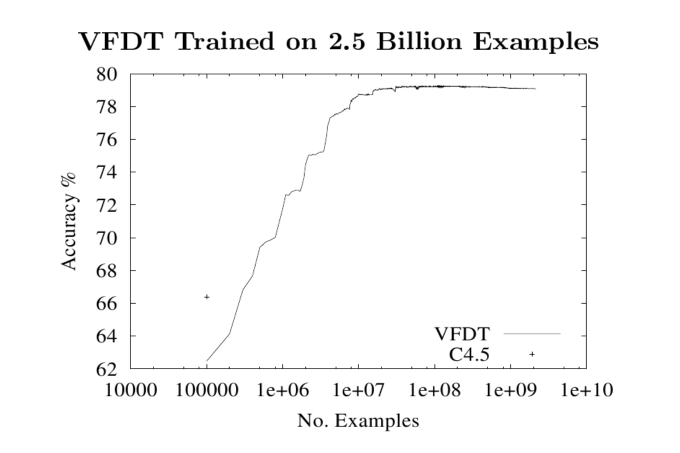
Evaluation in resource constrained environments:

- Sensor environment: memory hundreds of Kb
- Handheld computer: memory tens of Mb
- Server: several Gb

Do you need so many examples ?

Domingos, Hulten: *Mining High Speed Data Streams*, KDD00

VFDT: Illustrative Evaluation – Accuracy



Survey of Evaluation Methods

Work	Evaluation Method	Memory Management	Data Sources	Examples Train	Examples Test	Learning Curves	Drift
VFDT	holdout	Yes	Artif.	1M	50k	Yes	No
	holdout	Yes	real	4M	267k	Yes	No
CVFDT	holdout	Yes	Artif.	1M	Yes	Yes	Yes
VFDTc	holdout	No	Artif.	1M	250k	Yes	No
UFFT	holdout	No	Artif.	1.5M	250k	Yes	Yes
FACIL	holdout	Yes	Artif.	1M	100k	Yes	Yes
MOA	holdout	Yes	Artif.	1G		Yes	No
ANB	Prequential	No	Artif.			Yes	Yes

Outline

Evaluation Methods

You cannot touch the same water twice.

Cross Validation and variants does not apply.

Two alternatives:

- Holdout if data is stationary.
- Sequential Sampling

What if the distribution is non-stationary ?

- The *prequential* approach.
    - For each example:
        - First: make a prediction
        - Second: update the model, whenever the target is available.
- Evaluation over time-windows?

Prequential Evaluation

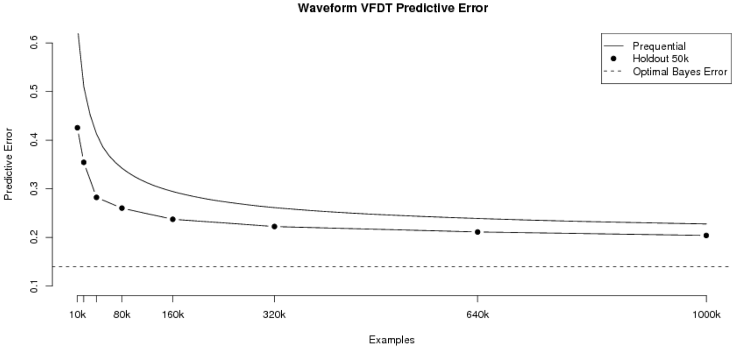
Definition: *The prequential error, computed at time i, is based on an accumulated sum of a loss function between the prediction and observed values:*

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

1. Provides a single number **at each time stamp**: a learning curve.
2. Pessimist estimator of accuracy.
3. Problematic to apply with algorithms with large testing time (k-NN).

Prequential versus Holdout

Prequential is a pessimistic estimator.



Definitions

Definition: The prequential error is computed, at time i, over a sliding window of size w ($\{e_j | j \in]i-w, i]\}$) as:

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

Definition: *The prequential error computed at time i, with fading factor α, can be written as:*

$$P_\alpha(i) = \frac{\sum_{k=1}^{i} \alpha^{i-k} L(y_k, \hat{y}_k)}{\sum_{k=1}^{i} \alpha^{i-k}} = \frac{\sum_{k=1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}}, \textit{ with } 0 \ll \alpha \leq 1.$$

Error Estimators Using Fading Factors.

The *fading sum* $S_{x,\alpha}(i)$ of observations from a stream x is computed at time i, as:

$$S_\alpha(i) = x_i + \alpha \times S_\alpha(i-1)$$

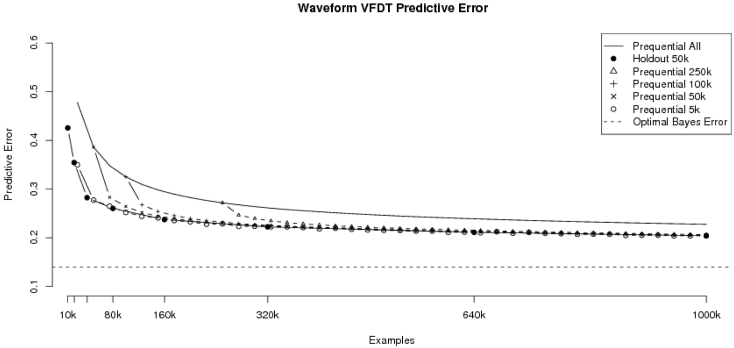
where $S_\alpha(1) = x_1$ and α $(0 \ll \alpha \leq 1)$ is a constant determining the forgetting factor of the sum, which should be close to 1 (for example 0.999).
The *fading average* at observation i is then computed as:

$$M_\alpha(i) = \frac{S_\alpha(i)}{N_\alpha(i)} \tag{1}$$

where $N_\alpha(i) = 1 + \alpha \times N_\alpha(i-1)$ is the corresponding *fading increment*, with $N_\alpha(1) = 1$.

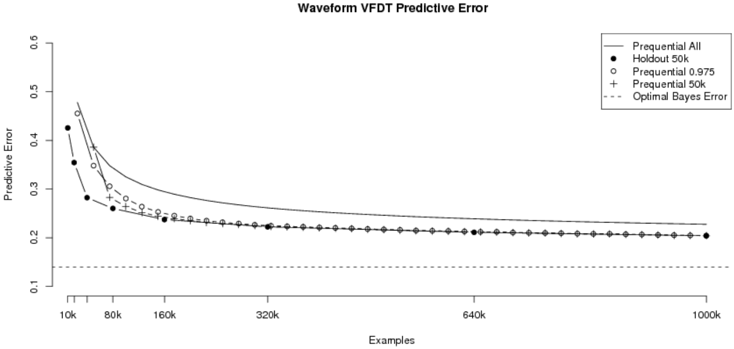
Prequential (sliding window) versus Holdout

Prequential over a sliding window converges to the holdout estimator.



Prequential (fading factor) versus Holdout

Prequential using fading factors converges to the holdout estimator.



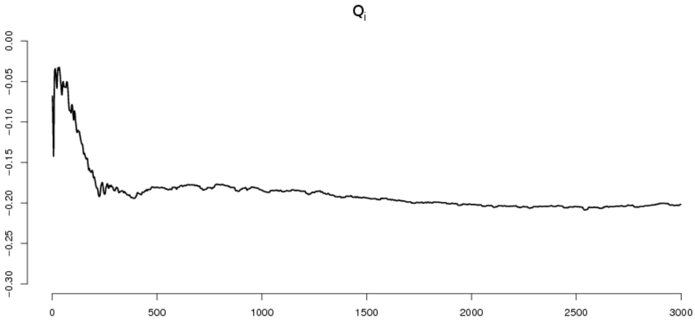
Outline

Accumulated Loss

- Let S_i^A and S_i^B be the sequences of the prequential accumulated loss for each algorithm.
- A useful statistic that can be used with almost any loss function, is: $Q_i(A, B) = log(\frac{S_i^A}{S_i^B})$.
- The signal of Q_i is informative about the relative performance of both models, while its value shows the strength of the differences.

Accumulated Loss

Q_i reflects the overall tendency but exhibit long term influences and is not able to fast capture when a model is in a recovering phase.



Accumulated Loss over sliding windows

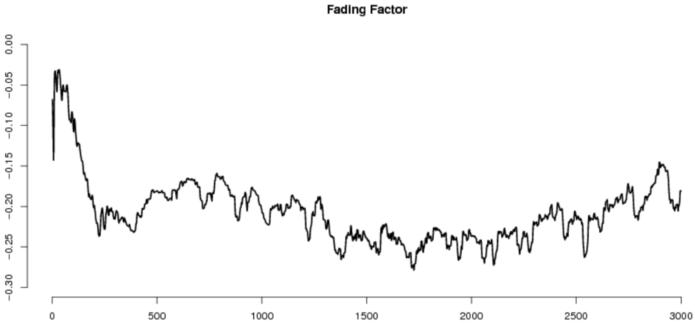
Q_i reflects the overall tendency but:

- exhibit long term influences and
- is not able to fast capture when a model is in a recovering phase.

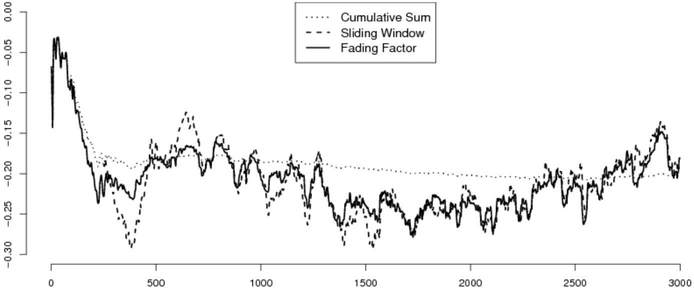
Sliding windows is an alternative, with the known problems of deciding the window-size,

Accumulated Loss using Fading Factors

$$Q_i^{\alpha}(A, B) = log(\frac{L_i(A) + \alpha \times S_{i-1}^{A}}{L_i(B) + \alpha \times S_{i-1}^{B}}).$$



Accumulated Loss using Fading Factors versus Sliding Window



Accumulated Loss using Fading Factors

- The fading factor is multiplicative, corresponding to an exponential forgetting.
- At time-stamp t the weight of example $t - k$ is α^k.
- Fading factors are fast and memoryless.

This is a strong advantage over sliding-windows that require to maintain in memory all the observations inside the window.

Outline

Statistical Hypothesis

Statistical Hypothesis: A statement about the parameters of one or more populations

- Hypothesis Testing: A procedure for deciding to accept or reject the hypothesis
  - Identify the parameter of interest
  - State a null hypothesis, $H0$;
  - Specify an alternate hypothesis, $H1$;
  - Choose a significance level α
  - State an appropriate test statistic

Error in Hypothesis Testing

- **Type I** error occurs when $H0$ is rejected but it is in fact true P(Type I error)$= \alpha$ or significance level
- **Type II** error occurs when we fail to reject $H0$ but it is in fact false P(Type II error)$= \beta$

Power $= 1 - \beta$: Probability of correctly rejecting $H0$, e.g., ability to distinguish between the two populations

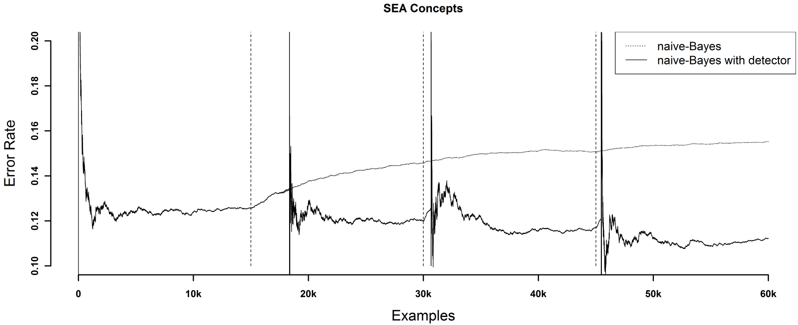
Signed McNemar Test for Comparative Assessment

- The McNemar test is one of most used tests for the 0-1 loss function;
- We need to compute two numbers:
  - $n_{0,1}$ denotes the number of examples misclassified by A and not by B;
  - $n_{1,0}$ denotes the number of examples misclassified by B and not by A;
- Both can be updated on the fly,
- The statistic $\frac{(n_{0,1}-n_{1,0})^2}{n_{0,1}+n_{1,0}}$ has a χ^2 distribution with 1 degree of freedom.

For a confidence level of 0.99, the null hypothesis is rejected if the statistic is greater than 7.

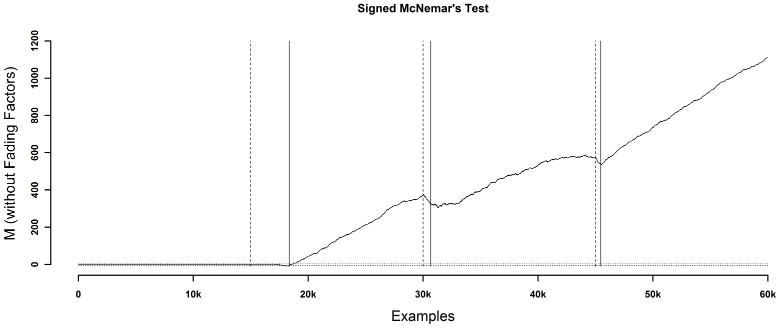
Signed McNemar Test

Illustrative Problem



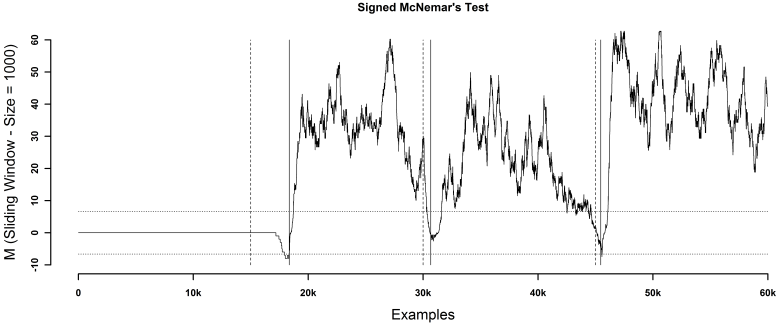
Signed McNemar Test

Evolution of McNemar Test



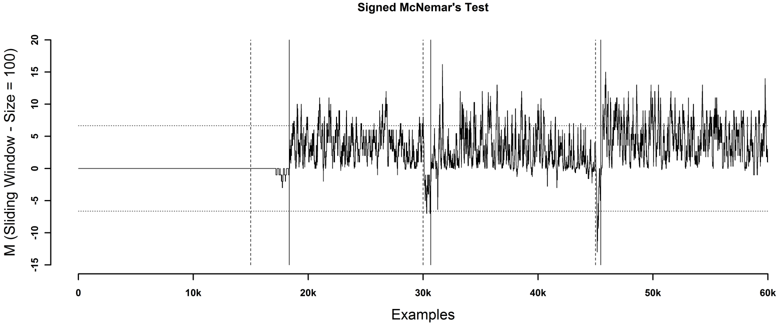
Signed McNemar Test

Evolution of McNemar Test using sliding windows (w=1000)



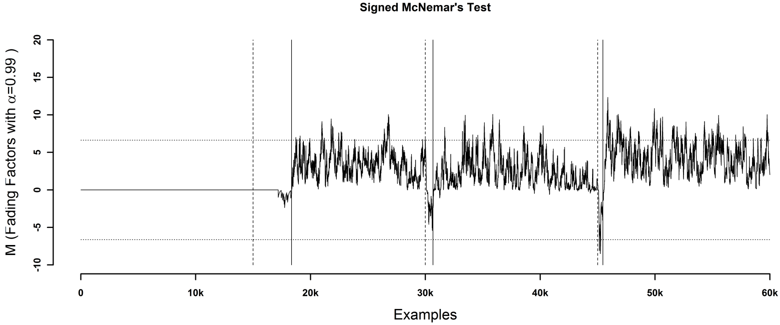
Signed McNemar Test

Evolution of McNemar Test using sliding windows (w=100)



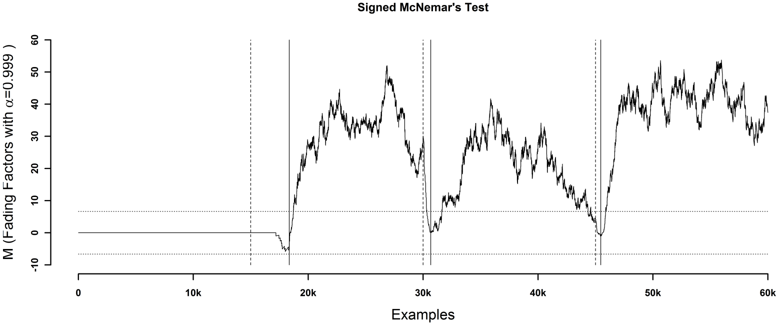
Signed McNemar Test

Evolution of McNemar Test using fading factors ($\alpha = 0.99$)



Signed McNemar Test

Evolution of McNemar Test using fading factors ($\alpha = 0.999$)



Outline

Concept drift

Any change in the distribution underlying the data

- **Concept drift** means that the concept about which data is obtained may shift from time to time, each time after some minimum permanence.
- **Context**: a set of examples from the data stream where the underlying distribution is stationary

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.

Metrics for Evaluation in Dynamic Environments

- Evolution of loss over time
  - All methods including *blind methods* (learn from a time window, weight examples).
- Methods for explicit change detection: informative about the dynamics of the process.
  - Probability of False Alarms;
  - Probability of True Alarms;
  - Delay in detection.

Evaluation under drift conditions

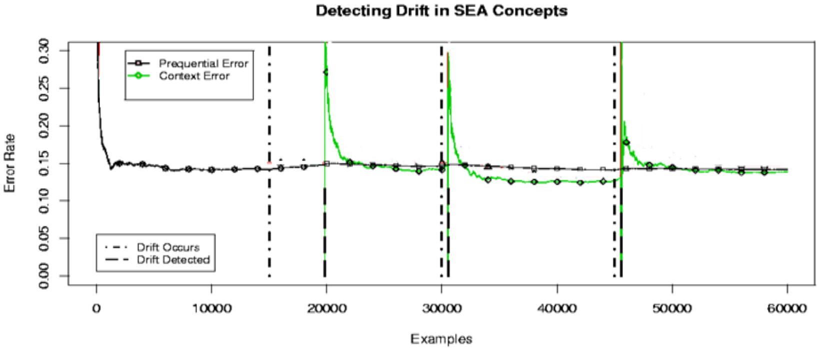
N. Street, Y. Kim: *A Streaming Ensemble Algorithm (SEA) for LargeScale Classification*, KDD01

- Randomly generate sets of examples for each concept
- Training sets are composed by sequences of concepts
- Evaluation of the resulting models:
- In a test set using the last concept

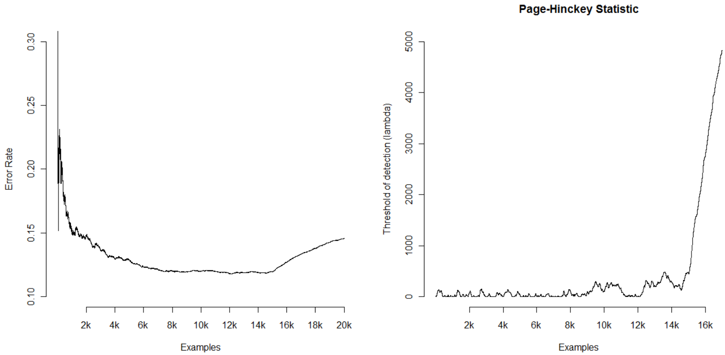
Is this process reasonable?

Illustrative Evaluation – Drift

Castillo, Gama; *An Adaptive Prequential Learning Framework for Bayesian Network Classifiers*, PKDD06

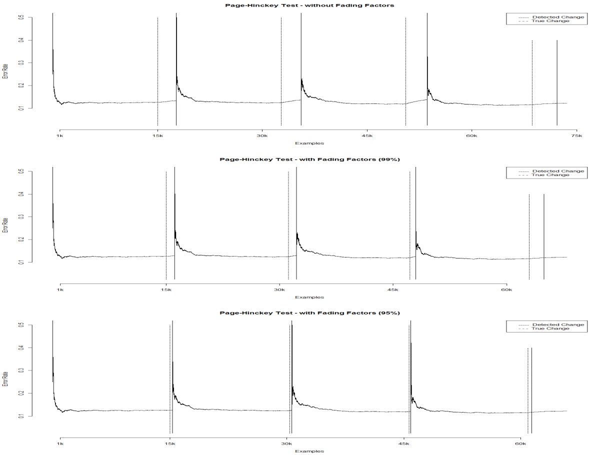


Illustrative Evaluation – Drift



The top figure shows the accumulated error of a classifier with a change in the context at point 15000. The bottom figure represents the evolution of the Page-Hinckley test statistic and the detection threshold λ.

Fading Factors and Delay Time



The evolution of the error rate and the delay times in drift detection using the Page-Hinckley test and different *fading-factors*.

Fading Factors and Delay Time

	Fading Factors				
Drifts	50%	80%	95%	99%	without
1st drift	164	323	346	1127	2707
2nd drift	249	283	318	1073	2825
3rd drift	172	213	234	759	3054
4th drift	238	455	476	1970	3581

Table: Delay times in drift scenarios using different *fading factors*.

Outline

Lessons Learned I

The main goal in the evaluation methods when learning from dynamic, non-stationary, data streams:

- Assess the performance of learning algorithms in dynamic environments
- Compare algorithms and variants

Lessons Learned II

- The prequential error computed over a sliding window converges for the holdout error;
- Fading factors are a faster and memory less approach, that do not require to store in memory all the errors in the window.
- The Q statistic is a fast and incremental statistic to continuously compare the performance of two classifiers.
- The use of fading factors in drift detection achieve faster detection rates, maintaining the capacity of being resilient to false alarms when there are no drifts.

One additional advantage: Monitor the evolution of the learning process itself.

References

- R. Bouckaert. *Choosing between two learning algorithms based on calibrated tests*. ICML, 20003.
- T. Dietterich. *Approximate statistical tests for comparing supervised classification learning algorithms*. Neural Computation, 1998.
- Demsar, *Statistical Comparisons of Classifiers over Multiple Data Sets*, JMLR, 2006
- J.Gama, P.Rodrigues, R.Sebastiao, *On Evaluating Stream Learning Algorithms*, Machine Learning (to appear)