Concept Drift

Albert Bifet

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

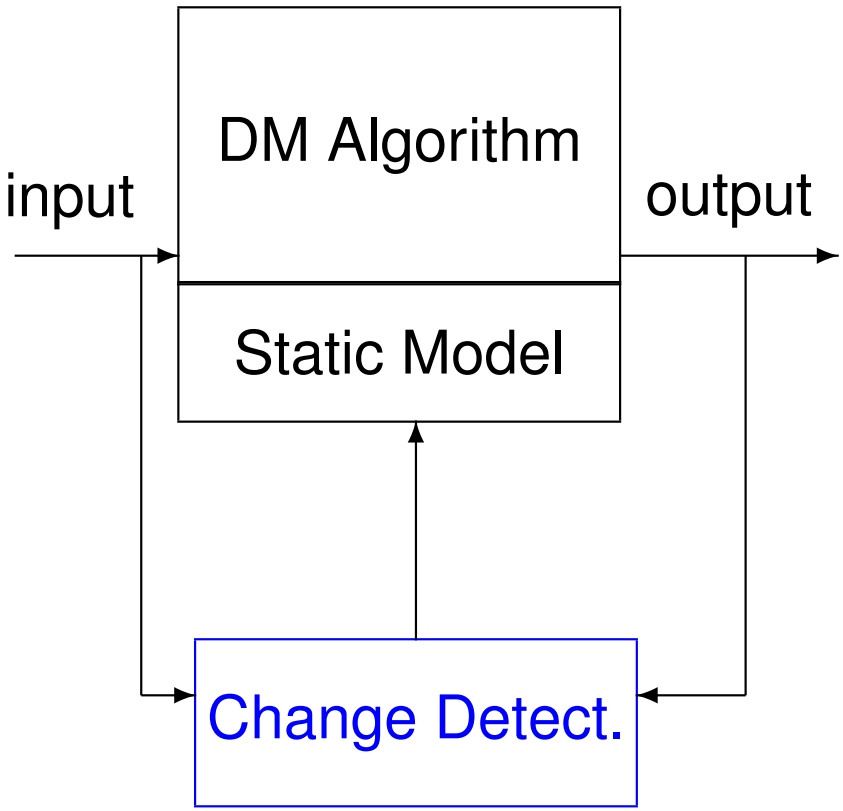
March 2012

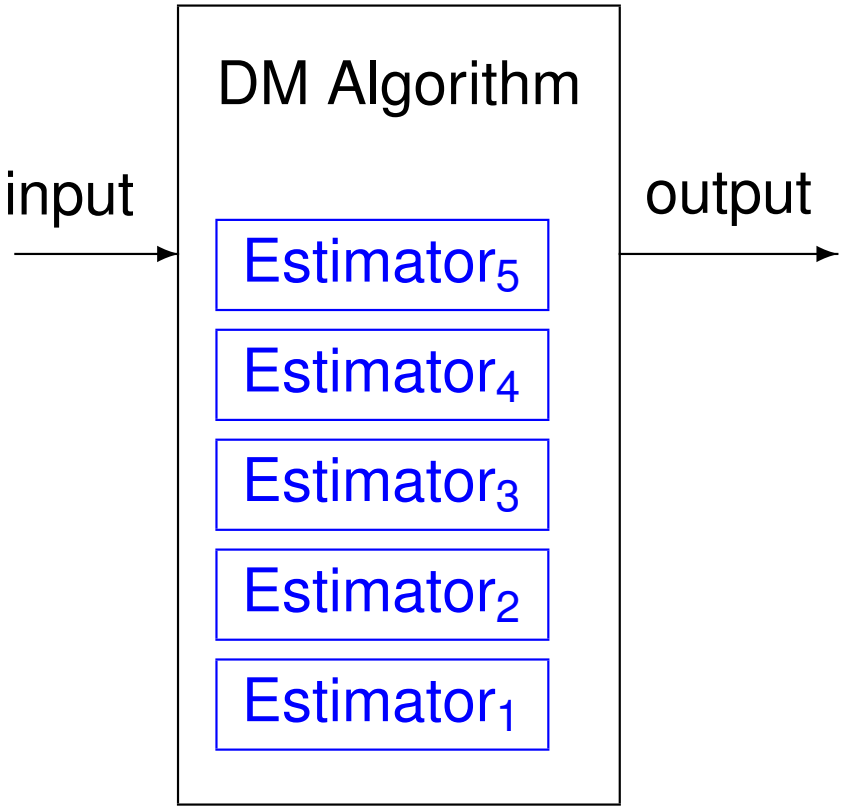
Outline

Data Streams

Big Data & Real Time

Data Mining Algorithms with Concept Drift.





Introduction.

Problem

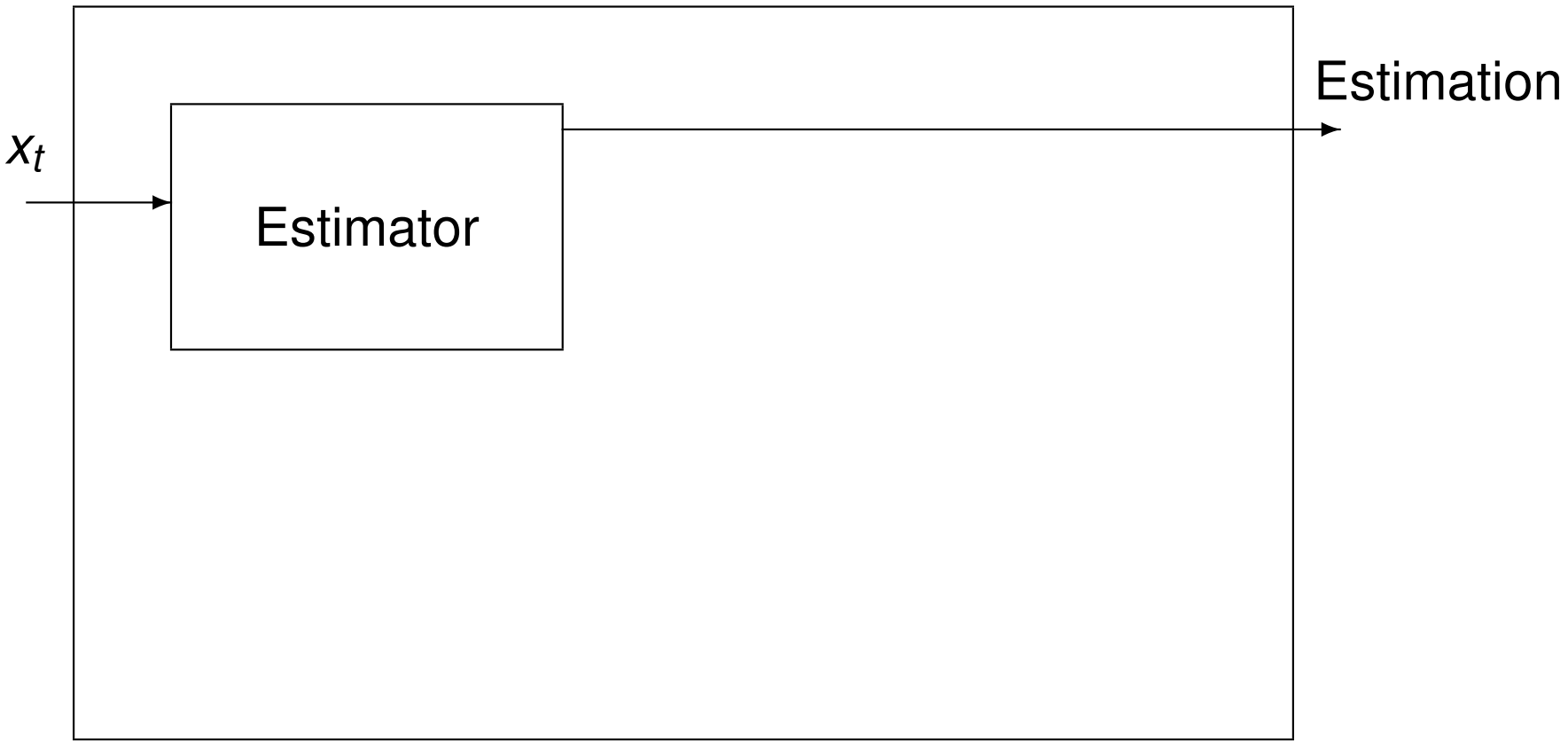
Given an input sequence $x_1, x_2, \cdots, x_t$ we want to output at instant t an alarm signal if there is a distribution change and also a prediction $\widehat{x}_{t+1}$ minimizing prediction error:

$$|\widehat{x}_{t+1} - x_{t+1}|$$

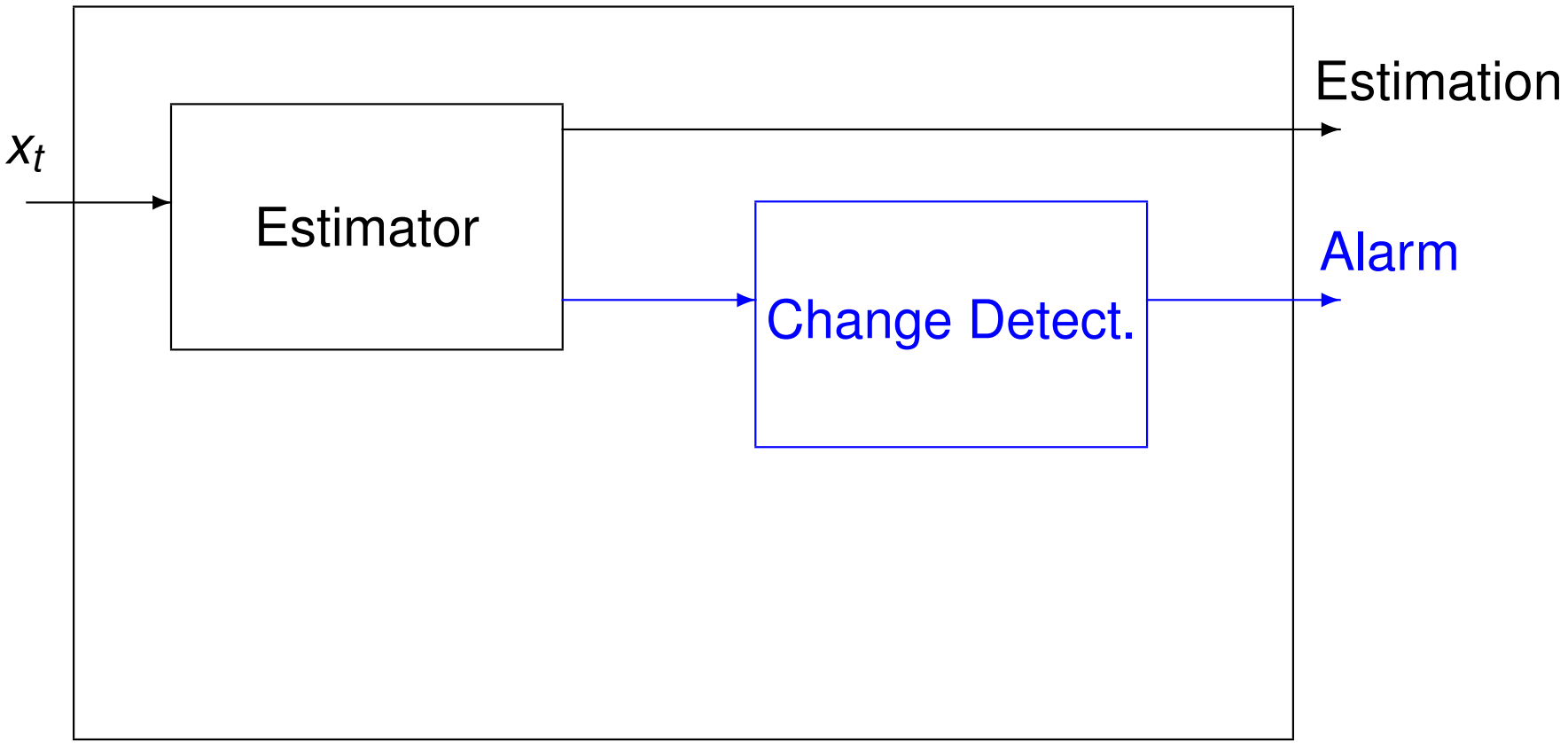
Outputs

- an estimation of some important parameters of the input distribution, and
- a signal alarm indicating that distribution change has recently occurred.

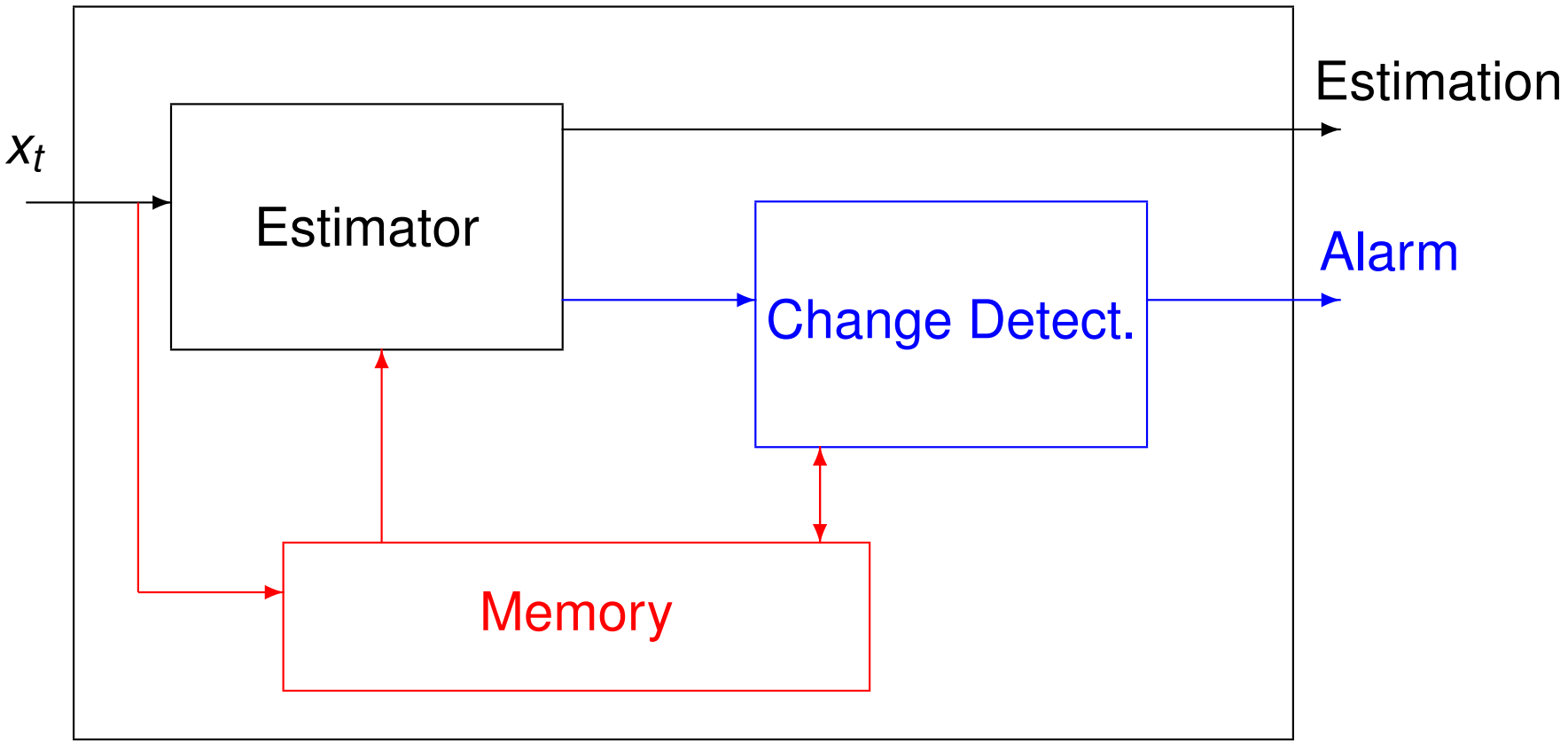
Change Detectors and Predictors



Change Detectors and Predictors



Change Detectors and Predictors



Concept Drift Evaluation

Mean Time between False Alarms (MTFA)

Mean Time to Detection (MTD)

Missed Detection Rate (MDR)

Average Run Length (ARL(θ))

The design of a change detector is a compromise between detecting true changes and avoiding false alarms.

Data Stream Algorithmics

- High accuracy in the prediction
- Low mean time to detection (MTD), false positive rate (FAR) and missed detection rate (MDR)
- Low computational cost: minimum space and time needed
- Theoretical guarantees
- No parameters needed

Main properties of an optimal change detector and predictor system.

The CUSUM Test

- The cumulative sum (CUSUM algorithm), gives an alarm when the mean of the input data is significantly different from zero.
- The CUSUM test is memoryless, and its accuracy depends on the choice of parameters υ and h.

$$g_0 = 0, \qquad g_t = \max\left(0, g_{t-1} + \epsilon_t - \upsilon\right)$$

$$\text{if } g_t > h \text{ then alarm and } g_t = 0$$

Cumulative sum algorithm (CUSUM).

Page Hinckley Test

- The CUSUM test

$$g_0 = 0, \qquad g_t = \max\left(0, g_{t-1} + \epsilon_t - \upsilon\right)$$

$$\text{if } g_t > h \text{ then alarm and } g_t = 0$$

- The Page Hinckley Test

$$g_0 = 0, \qquad g_t = g_{t-1} + (\epsilon_t - \upsilon)$$

$$G_t = \min(g_t)$$

$$\text{if } g_t - G_t > h \text{ then alarm and } g_t = 0$$

Geometric Moving Average Test

- The CUSUM test

$$g_0 = 0, \qquad g_t = \max\left(0, g_{t-1} + \epsilon_t - \upsilon\right)$$

$$\text{if } g_t > h \text{ then alarm and } g_t = 0$$

- The Geometric Moving Average Test

$$g_0 = 0, \qquad g_t = \lambda g_{t-1} + (1-\lambda)\epsilon_t$$

$$\text{if } g_t > h \text{ then alarm and } g_t = 0$$

The forgetting factor λ is used to give more or less weight to the last data arrived.

Statistical test

$$\hat{\mu}_0 - \hat{\mu}_1 \in N(0, \sigma_0^2 + \sigma_1^2), \text{ under } H_0$$

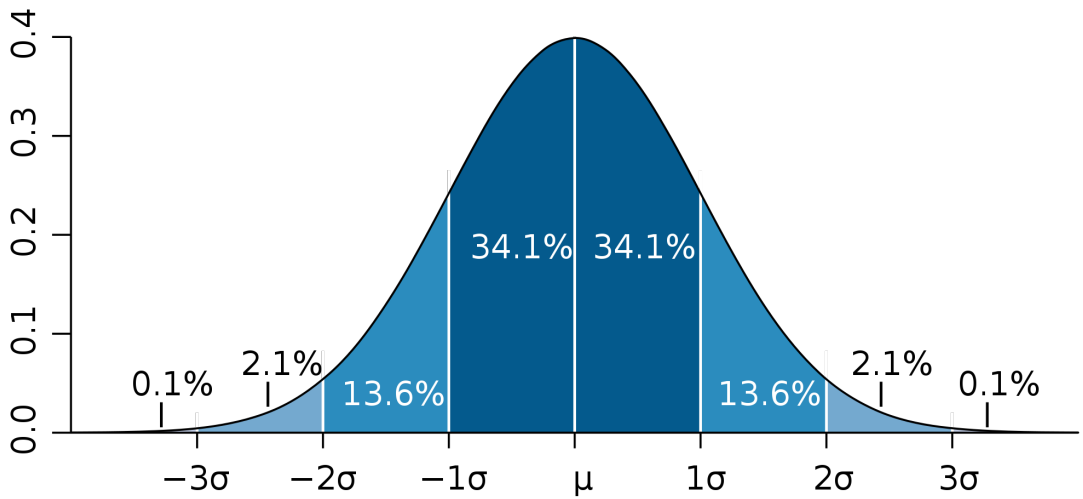
Example: Probability of false alarm of 5%

$$\Pr\left(\frac{|\hat{\mu}_0 - \hat{\mu}_1|}{\sqrt{\sigma_0^2 + \sigma_1^2}} > h\right) = 0.05$$

As $P(X < 1.96) = 0.975$ the test becomes

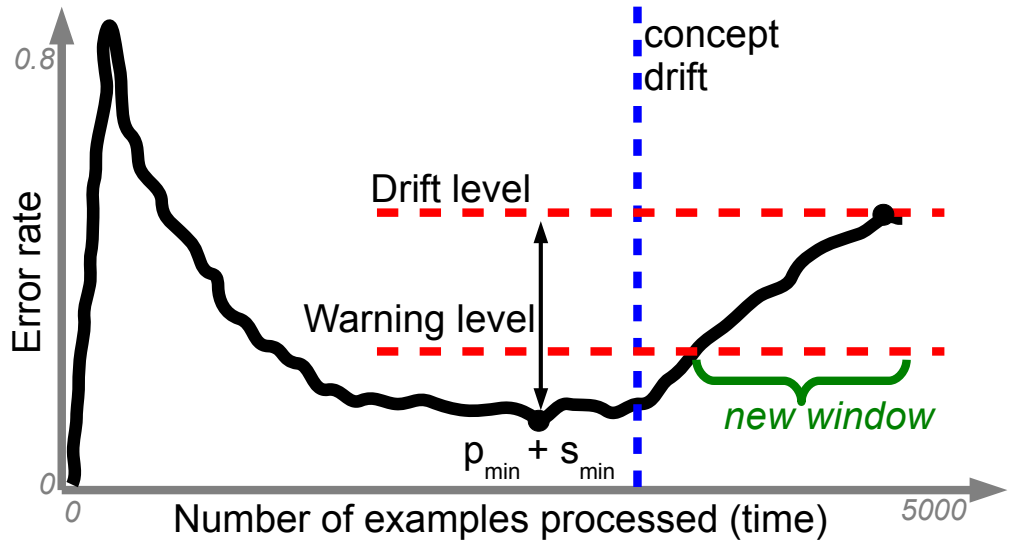
$$\frac{(\hat{\mu}_0 - \hat{\mu}_1)^2}{\sigma_0^2 + \sigma_1^2} > 1.96^2$$

Concept Drift



6 sigma

Concept Drift



Statistical Drift Detection Method
(Joao Gama et al. 2004)

ADWIN: Adaptive Data Stream Sliding Window

Let $W =$ 101010110111111

- Equal & fixed size subwindows: 1010 1011011 1111
- Equal size adjacent subwindows: 1010101 1011 1111
- Total window against subwindow: 10101011011 1111
- ADWIN: All adjacent subwindows:

1	01010110111111
1010	10110111111
1010101	10111111
1010101101	11111
10101011011111	1

Data Stream Sliding Window

101100011110101 0111010

Sliding Window

We can maintain simple statistics over sliding windows, using $O(\frac{1}{\epsilon}\log^2 N)$ space, where

- N is the length of the sliding window
- ϵ is the accuracy parameter

M. Datar, A. Gionis, P. Indyk, and R. Motwani.
Maintaining stream statistics over sliding windows. 2002

Exponential Histograms

$M = 2$

1010101	101	11	1	1	1
Content: 4	2	2	1	1	1
Capacity: 7	3	2	1	1	1

1010101	101	11	11	1
Content: 4	2	2	2	1
Capacity: 7	3	2	2	1

1010101	10111	11	1
Content: 4	4	2	1
Capacity: 7	5	2	1

Exponential Histograms

1010101	101	11	1	1
Content: 4	2	2	1	1
Capacity: 7	3	2	1	1

Error $<$ content of the last bucket W/M

$\epsilon = 1/(2M)$ and $M = 1/(2\epsilon)$

$M \cdot \log(W/M)$ buckets to maintain the data stream sliding window

Exponential Histograms

1010101	101	11	1	1
Content: 4	2	2	1	1
Capacity: 7	3	2	1	1

To give answers in $O(1)$ time,
it maintain three counters LAST, TOTAL and VARIANCE.

$M \cdot \log(W/M)$ buckets to maintain the data stream sliding window

Algorithm ADaptive Sliding WINdow

ADWIN: ADAPTIVE WINDOWING ALGORITHM

1 Initialize W as an empty list of buckets
2 Initialize WIDTH, VARIANCE and TOTAL
3 **for** each $t > 0$
4 **do** SETINPUT(x_t, W)
5 output $\hat{\mu}_W$ as TOTAL/WIDTH and ChangeAlarm

SETINPUT(item e, List W)

1 INSERTELEMENT(e, W)
2 **repeat** DELETEELEMENT(W)
3 **until** $|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| < \epsilon_{cut}$ holds
4 for every split of W into $W = W_0 \cdot W_1$

Algorithm ADaptive Sliding WINdow

INSERTELEMENT(item e, List W)

1 create a new bucket b with content e and capacity 1
2 $W \leftarrow W \cup \{b\}$ (i.e., add e to the head of W)
3 update WIDTH, VARIANCE and TOTAL
4 COMPRESSBUCKETS(W)

DELETEELEMENT(List W)

1 remove a bucket from tail of List W
2 update WIDTH, VARIANCE and TOTAL
3 ChangeAlarm $\leftarrow$ **true**

Algorithm ADaptive Sliding WINdow

COMPRESSBUCKETS(List W)

```
1  Traverse the list of buckets in increasing order
2      do If there are more than M buckets of the same capacity
3          do merge buckets
4              COMPRESSBUCKETS(sublist of W not traversed)
```

Algorithm ADaptive Sliding WINdow

Theorem

At every time step we have:

1. (False positive rate bound). *If μ_t remains constant within W, the probability that* ADWIN *shrinks the window at this step is at most δ.*
2. (False negative rate bound). *Suppose that for* some *partition of W in two parts $W_0 W_1$ (where W_1 contains the most recent items) we have $|\mu_{W_0} - \mu_{W_1}| > 2\epsilon_{cut}$. Then with probability $1 - \delta$* ADWIN *shrinks W to W_1, or shorter.*

ADWIN tunes itself to the data stream at hand, with no need for the user to hardwire or precompute parameters.

Algorithm ADaptive Sliding WINdow

ADWIN using a Data Stream Sliding Window Model,

- can provide the exact counts of 1's in $O(1)$ time per point.
- tries $O(\log W)$ cutpoints
- uses $O(\frac{1}{\epsilon} \log W)$ memory words
- the processing time per example is $O(\log W)$ (amortized and worst-case).

Sliding Window Model

	1010101	101	11	1	1
Content:	4	2	2	1	1
Capacity:	7	3	2	1	1