POLITECNICO
MILANO 1863

# Hardware Architectures for Embedded and Edge AI

*Prof Manuel Roveri – manuel.roveri@polimi.it*

*Lecture 9 – Learning in presence of concept drift*

# Summary

Learning in nonstationary environments

What to adapt?

- Instance selection, Instance weighting, Model ensemble

When to adapt?

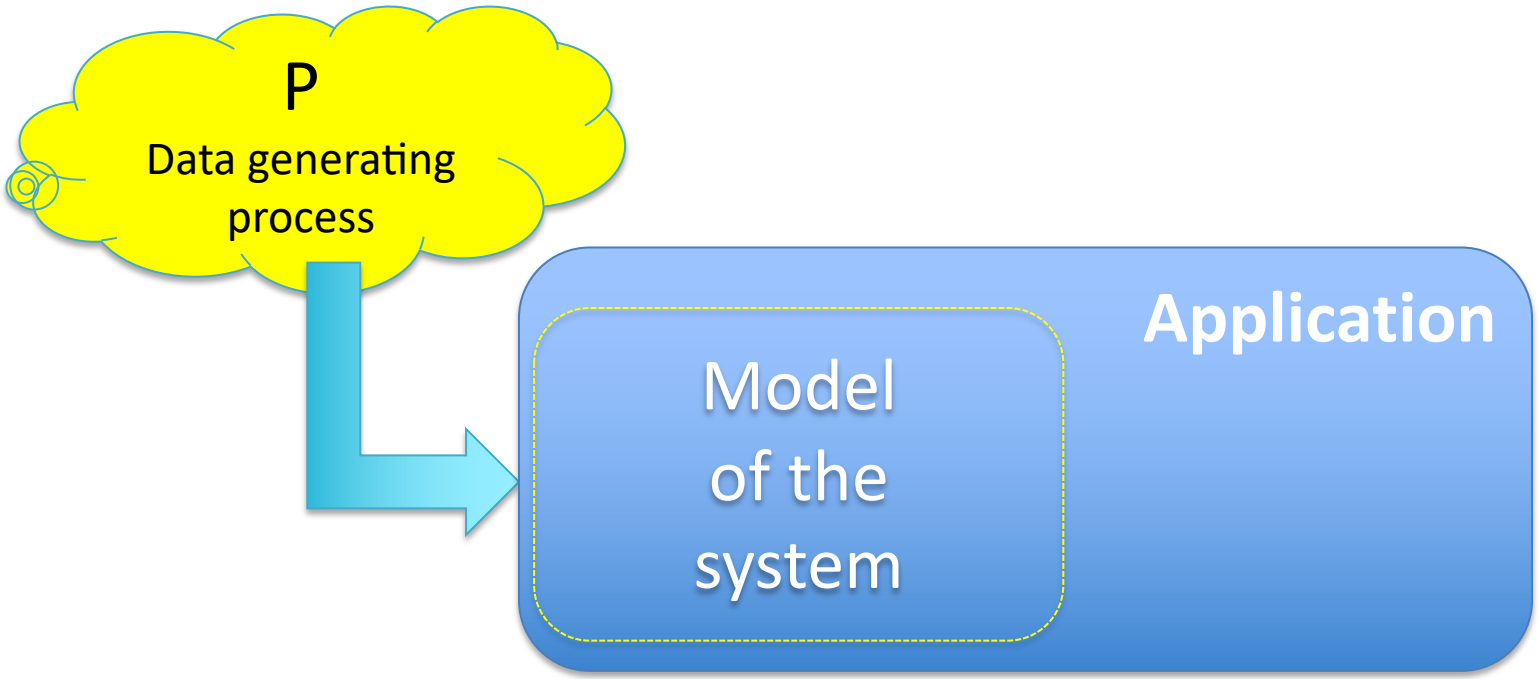
- Passive solutions, Active solutions

Concept drift detection

Just-in-Time Adaptive Classifiers

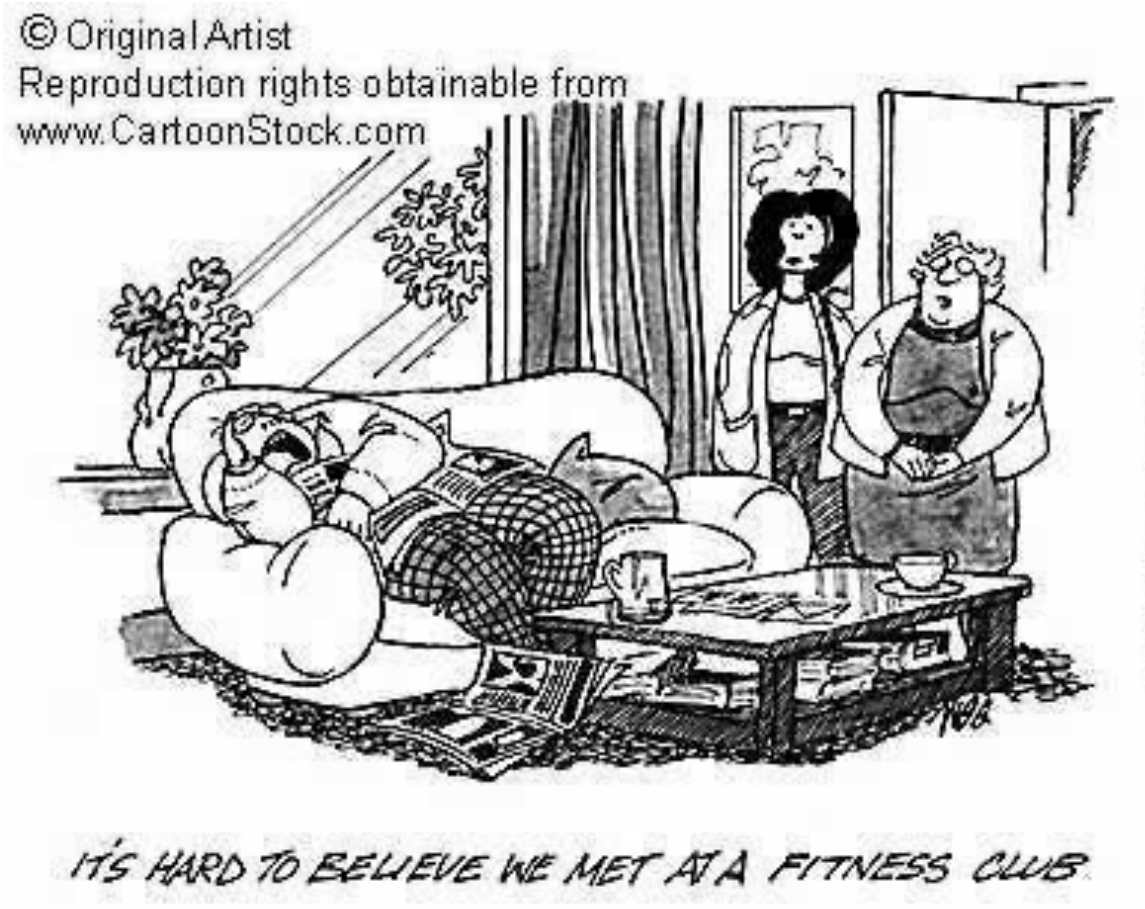
Recurrent concepts

# Data-processing and applications



- Up to now we assumed the system model to be time invariant…

# But everything and everybody changes over time …

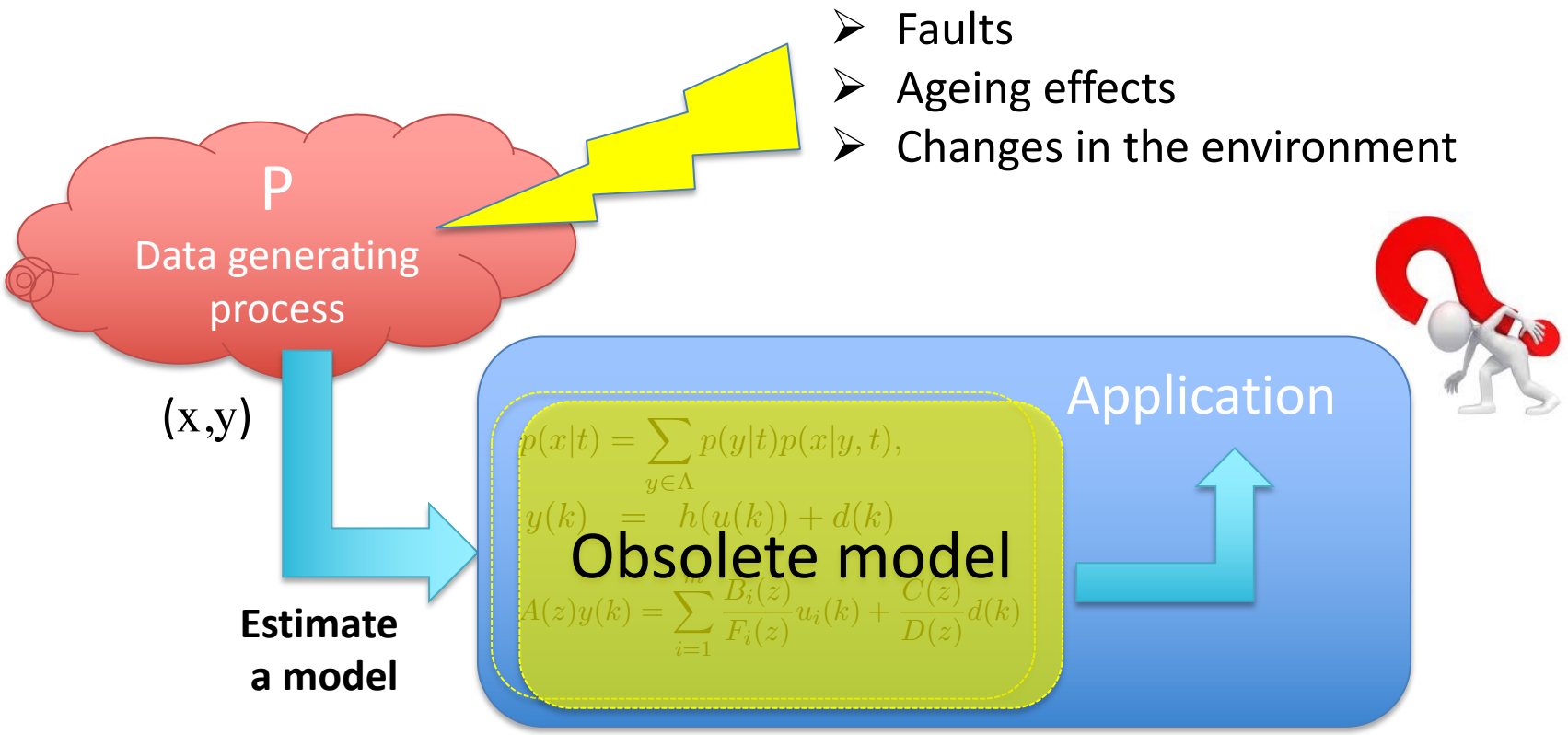



Be aware of *Concept drift*…

# Ageing effects …

# … changes in the system or the environment

# Learning in Nonstationary Environments: the effect of the non-stationarity




- Faults
- Ageing effects
- Changes in the environment

***Perturbed, incorrect and missing data*** *can hence heavily* ***affect the subsequent processing phase*** *so as to possibly induce wrong decisions or on-the-field reactions.*

# Stationarity and time invariance

## Stationarity

- We say that a data generating process is stationary when generated data are i.i.d. realizations of a unique random variable whose distribution does not change with time

## Time invariance

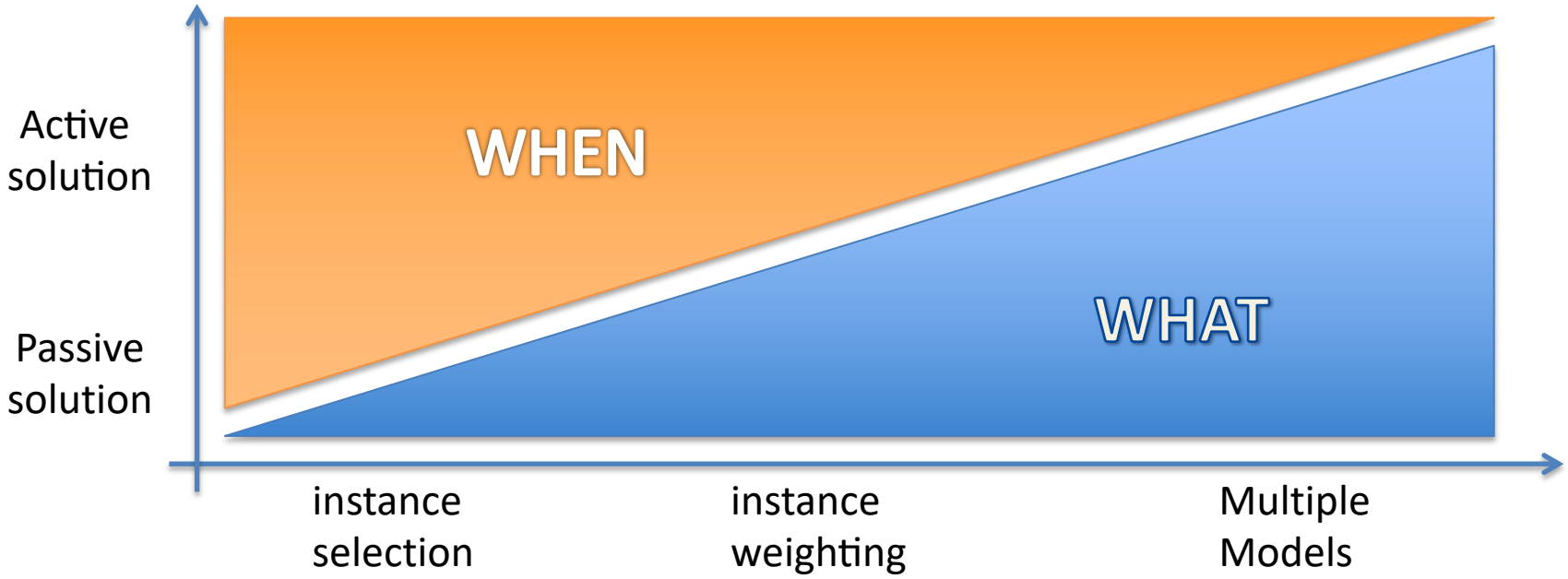
- We say that a process is time invariant when its outputs do not explicitely depend on time

$$y(t) = a_1(e^{t_0 - t})y(t-1) + a_2 y(t-2) + \eta, \eta = N(0, \sigma^2)$$

# Searching for adaptation

Traditional assumption: stationarity hypothesis

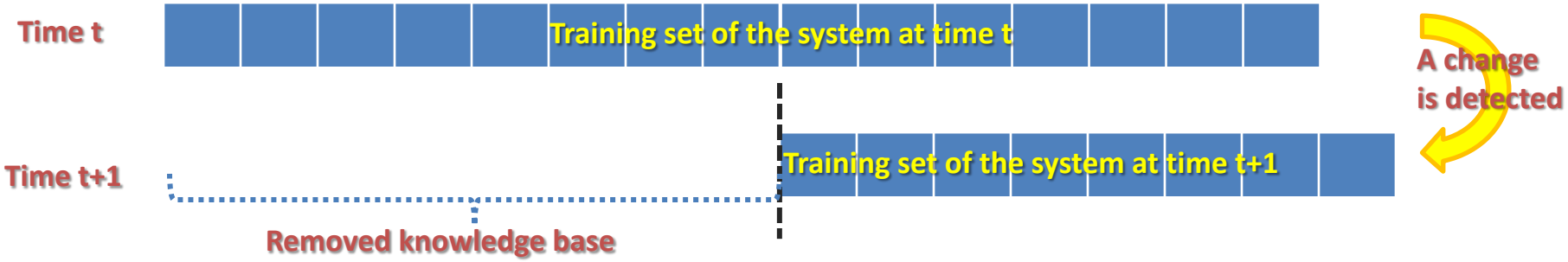
Adaptive solutions in a non-stationary framework:




A comprehensive methodology addressing this problem is not available

# WHAT: Instance Selection

**The idea**: *identifying the samples of the training set that are relevant to the current state of the process.*

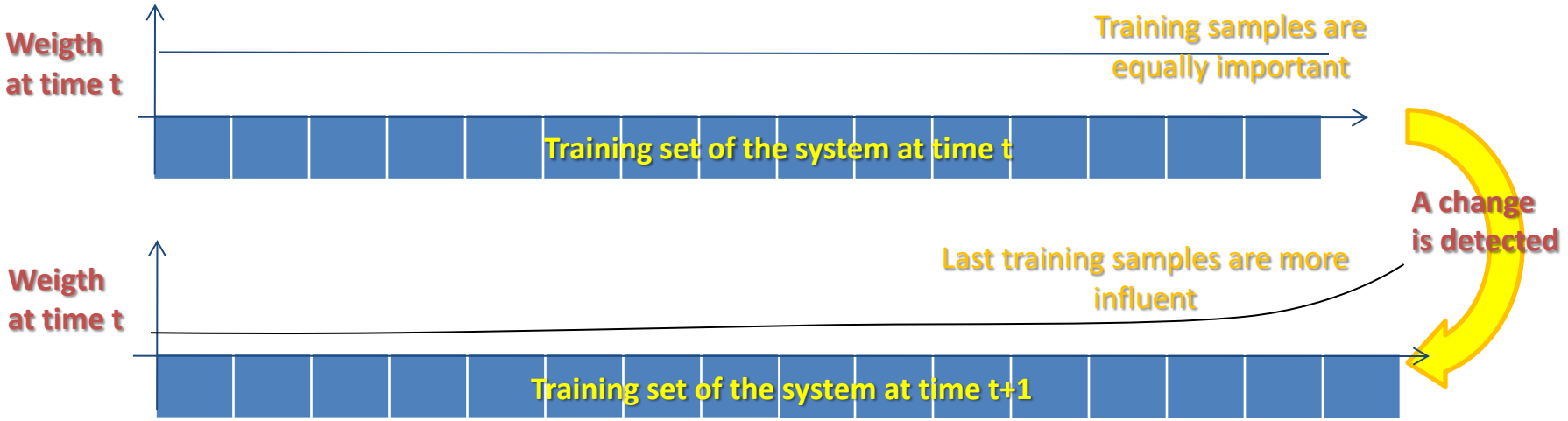



The adaptive systems generally rely on a window over the most recent training samples to process the upcoming data

- **fixed window approach:** the length of the window is fixed a-priori by the user
- **heuristic approaches:** adapt the window length over the latest samples to maximize the accuracy

# WHAT: Instance Weighting

**The idea**: *training samples are not removed from the training set of the system but all the training samples (suitably weighted) are considered.*

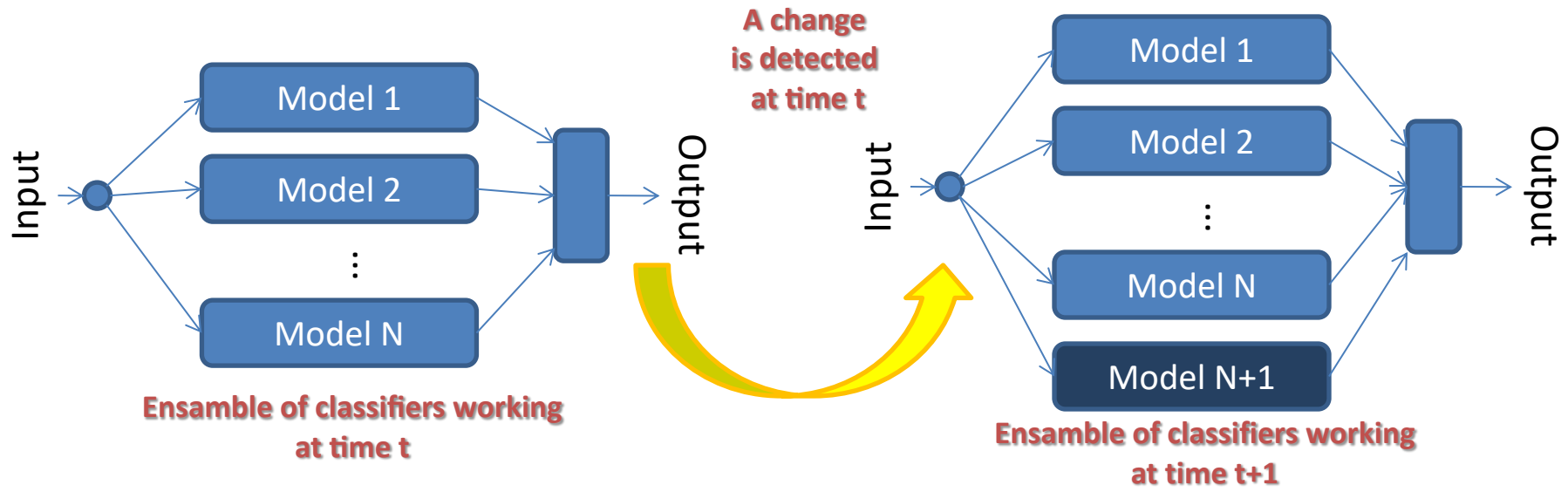



The training samples might be weighted according to

- the **age**
- the **relevancy to the current state** of the process in term of accuracy of the last batch of supervised data

# WHAT: Multiple Models

The idea: *the outputs of an ensemble of models are combined by means of voting or weighted mechanisms to form the final output*




All these systems includes **techniques for dynamically including new models in the system or deleting obsolete ones** (i.e., pruning techniques aiming at removing the oldest model or the one with the lowest accuracy).

# Critical analysis of the considered approaches

## Instance selection

↑: low computational-complexity
reduced training set

↓: fixed windows or heuristics to adapt the window size
forgetting mechanisms

## Instance weighting

↑: low computational-complexity
availability of all the training samples for recurrent models

↓: heuristics to define the sample weights
full training set

## Multiple models

↑: availability of a model for “each bunch of data”

↓: high computational-complexity

# WHEN: passive vs active approach

**Passive solutions** continuously adapt the model without the need to detect the change

- Ensembles of models with the adaptation phase consisting in a continuous update of the weights of the fusion/aggregation rule and creation/removal of models

**Active solutions** rely on triggering mechanisms to identify changes in the process and react by updating the model

- The most popular triggering mechanism is the change detection

# Passive approach: the general idea

The underlying data distributions may (or may not) change at any time with any rate of change.

A **continuous adaptation** of the model parameters every time new data arrive

**Maintain an up-to-date model** at all times

- **Avoid** the potential pitfall associated with **false alarms** in active solutions

# Passive learning

Online (incremental) learning

$$V_N(\theta, \{(x_i, y_i)\}) = L(y_i, f(\theta, x_i))$$

$$\theta_{i+1} = \theta_i - \eta \frac{\partial L(y_i, f(\theta, x_i))}{\partial \theta}|_{\theta_i}$$

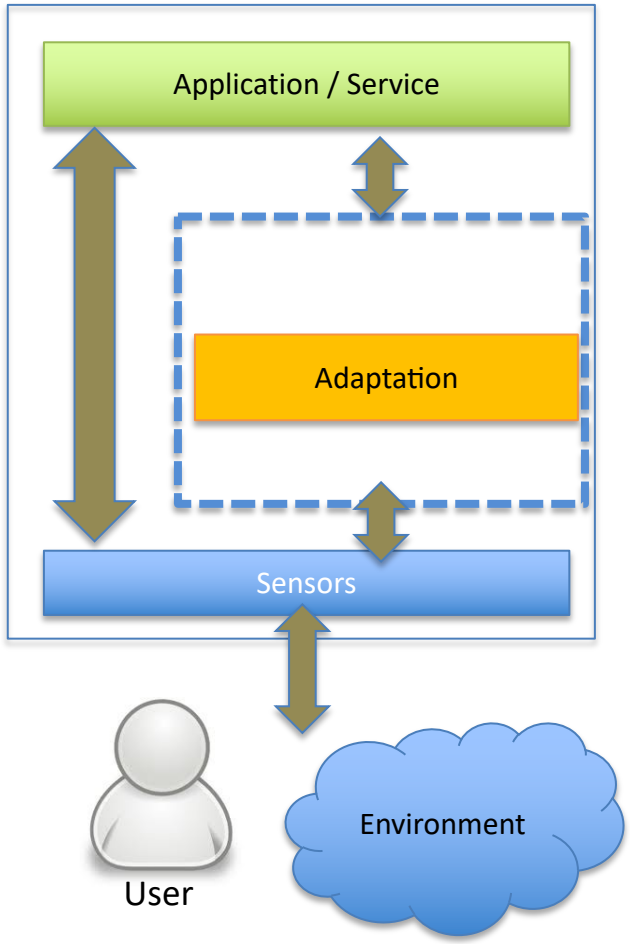
Batch learning

$$Z_{n,i} = \{(x_i, y_i), (x_{i-1}, y_{i-1}), \cdots (x_{i-n+1}, y_{i-n+1})\}$$

$$\theta_{i+1} = \theta_i - \eta \frac{\partial V_N(\theta, Z_{n,i})}{\partial \theta}|_{\theta_i}$$

Ensemble learning

$$y(x) = \sum_{i=1}^{k} w_i M_i(x)$$

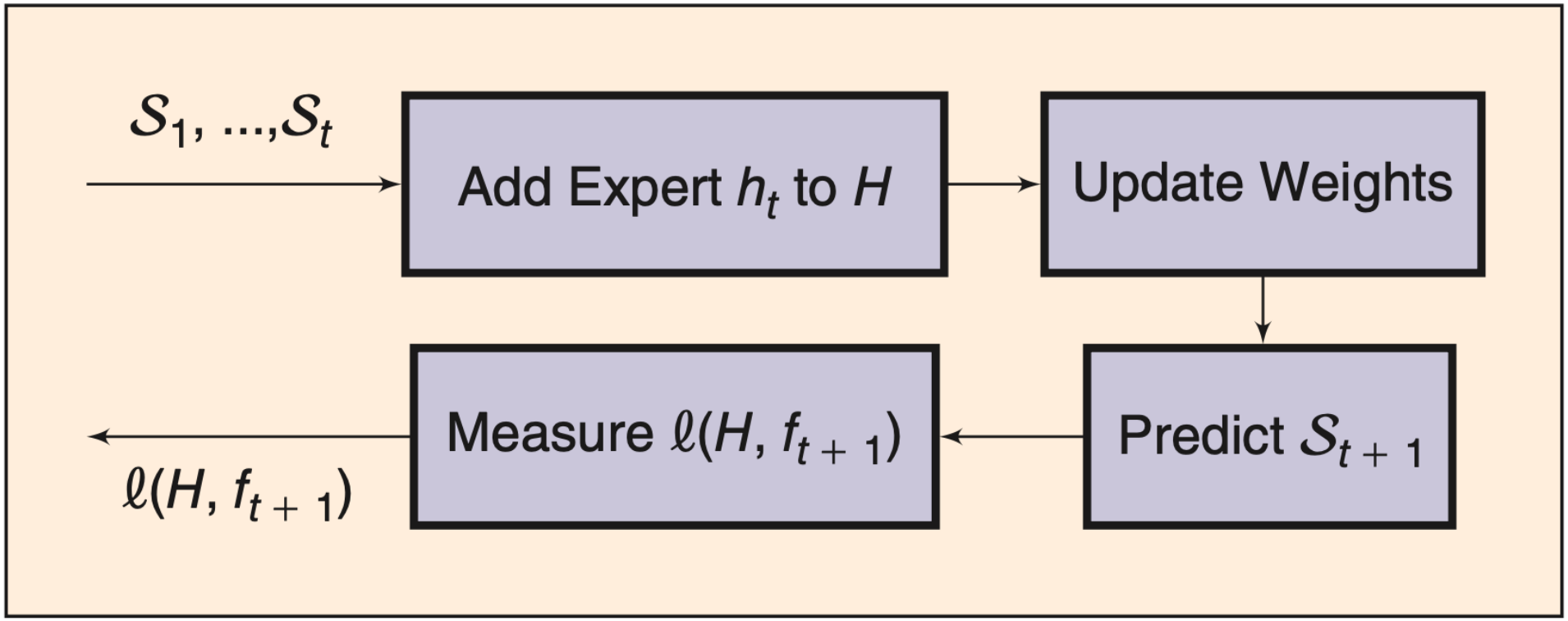


# Ensemble-based mechanisms

Ensemble-based approaches provide **a natural fit to the problem of learning in nonstationary environments**:

a) more accurate than single classifier-based systems
b) easily incorporate new data into a classification model (a new item into the ensemble)
c) provide a natural mechanism to forget irrelevant knowledge (removing an item from the ensemble)

# High-level block diagram used by incremental learning ensembles in nonstationary environments



# Incremental learning ensembles in nonstationary environments

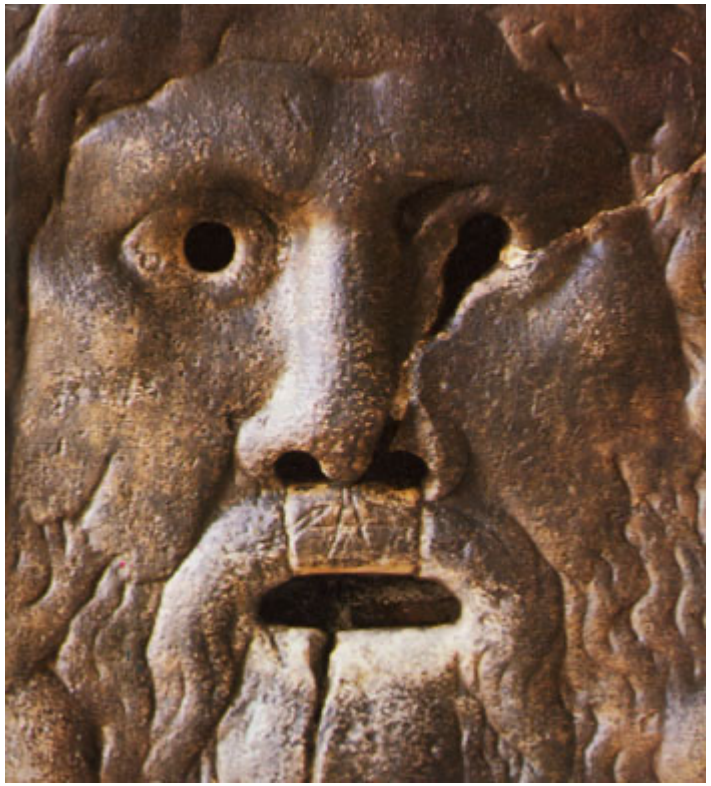
The **streaming ensemble algorithm** (**SEA**) was one of the earliest ensemble approaches:

- **New classifiers are added** as new batches of data arrive
- Classifiers are removed as the ensemble reaches a predetermined size
- **Which classifier must be removed**?
  - evaluation of classifier's predictions
  - age of the classifier
  - remove the least contributing member

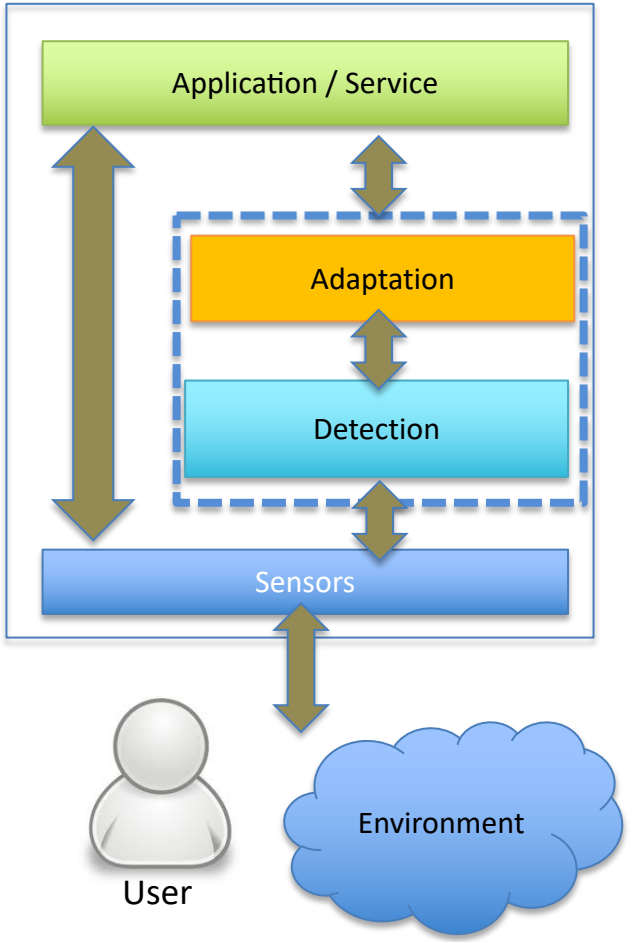
Other approaches:

- **Dynamic weighted majority** (DWM)
- **Online bagging/boosting** for nonstationary environments

# Active learning

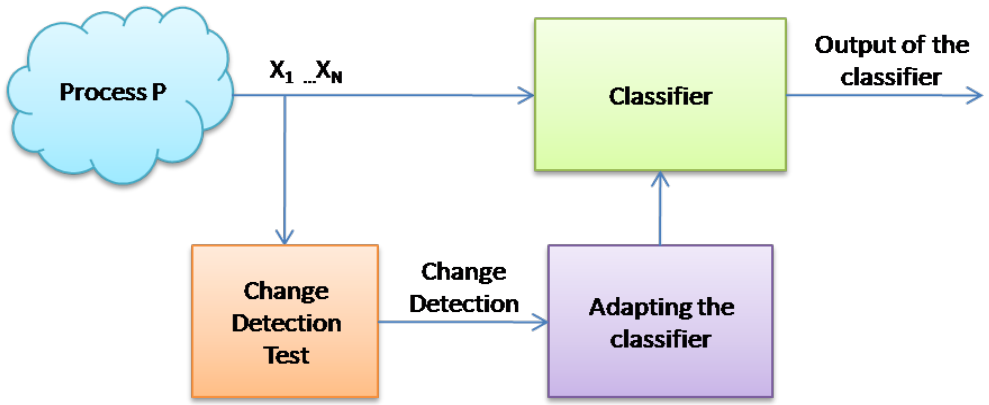


The Oracle provides information about an event, e.g., the occurrence of concept drift



# WHEN: Triggering mechanisms

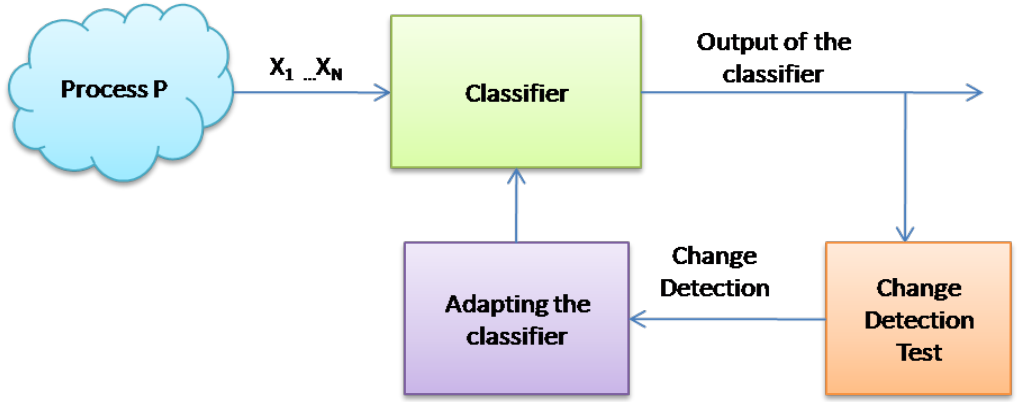
*Change detection on the pdf of the inputs:*




↑: monitoring the distribution of unlabeled observation

↓: this solution does not allow us for detecting changes that do not affect the distribution of observations

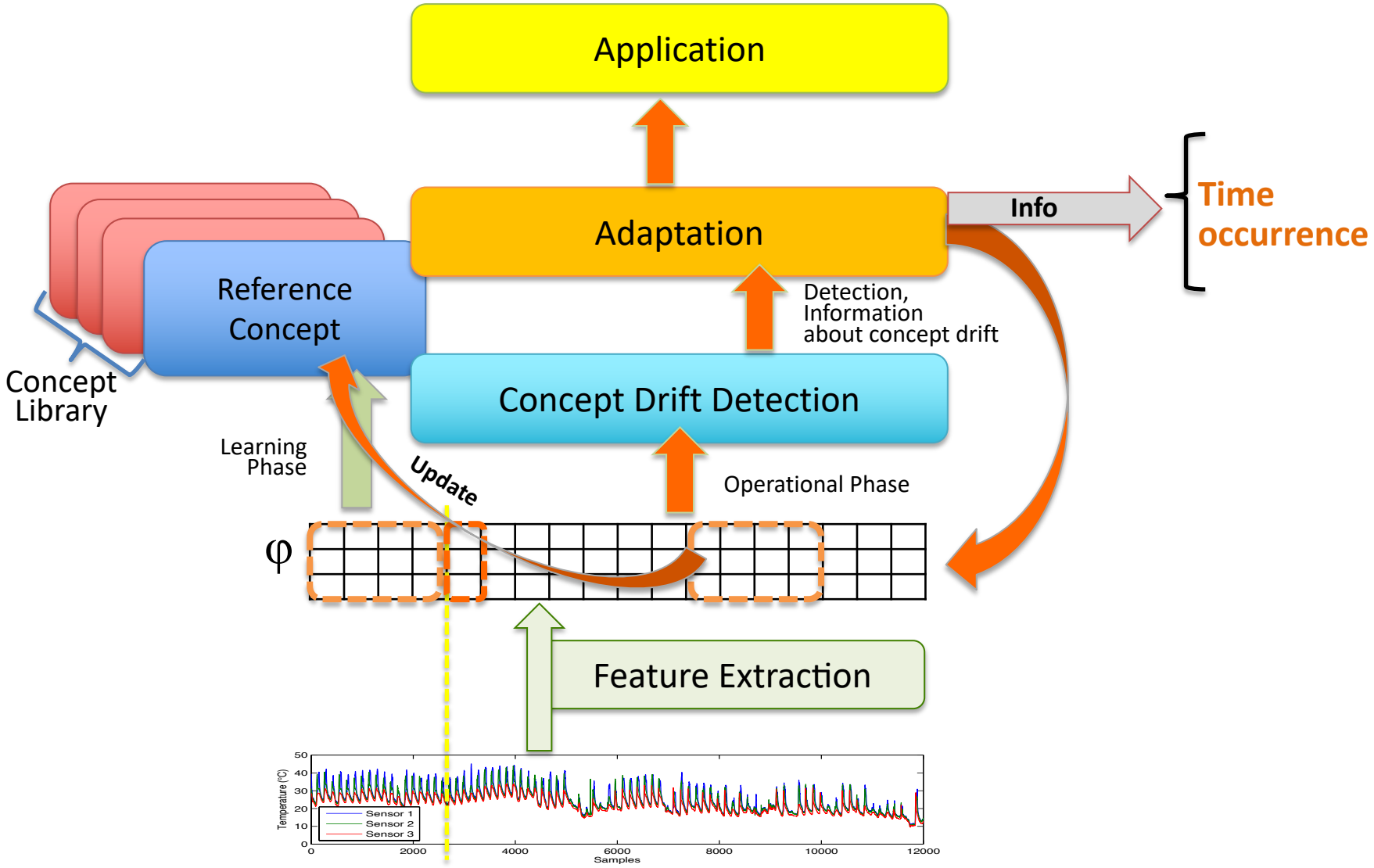
*Change detection on the classification error*




↑: reacting to changes when these directly influence its accuracy

↓: the need of supervised samples

# The active learning framework within an evolving environment

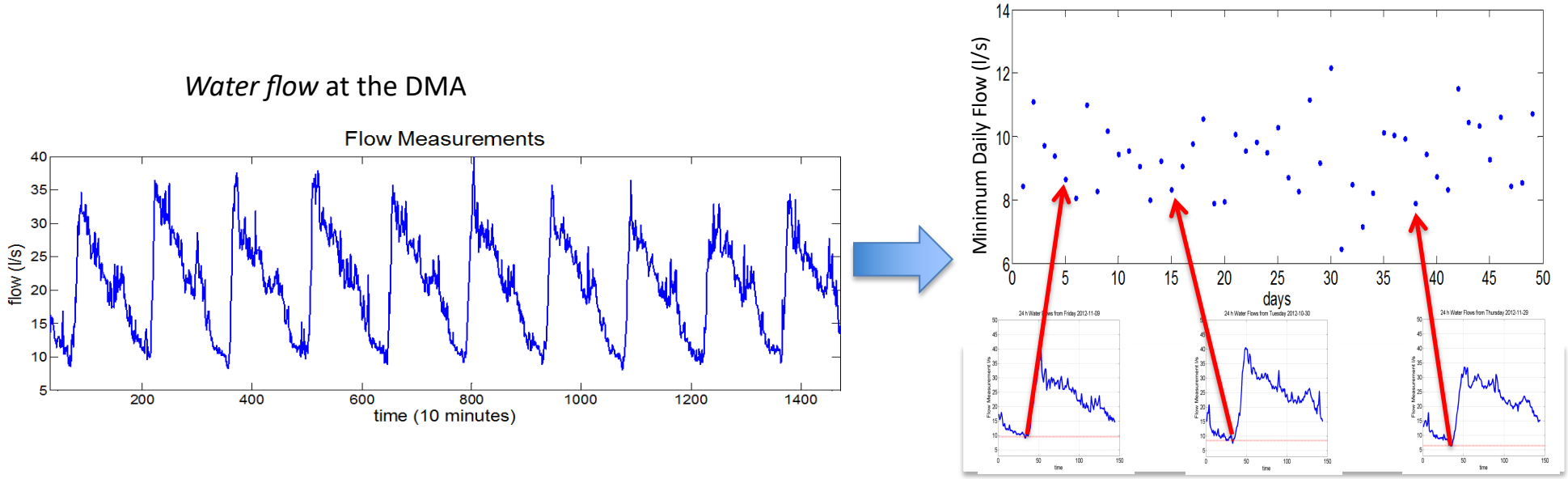


# Features (1)

Features must be i.i.d (but data are generally signals)

Data space

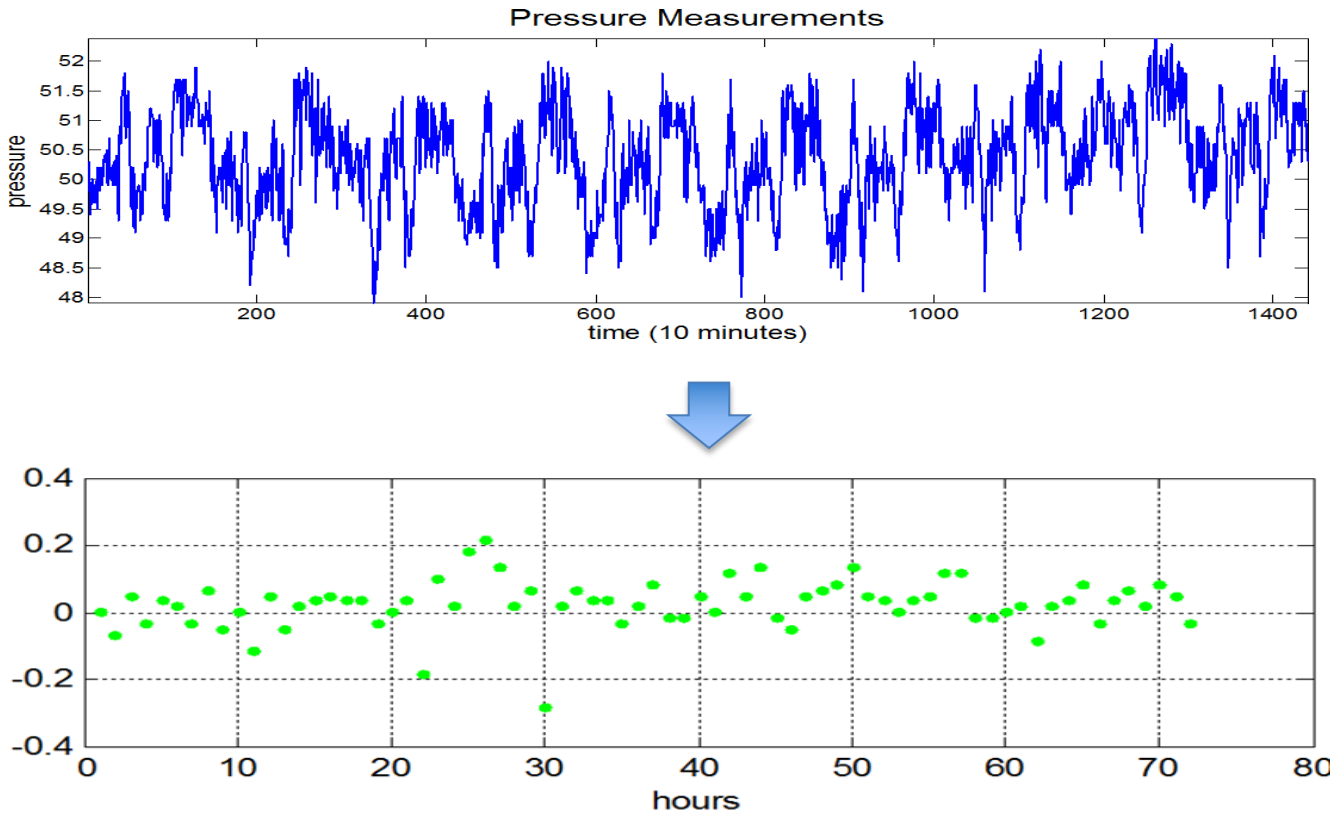
- ✓Raw data are used (e.g., the minimum of the water consumption of a day @ district metered area)



# Features (2)

Feature space

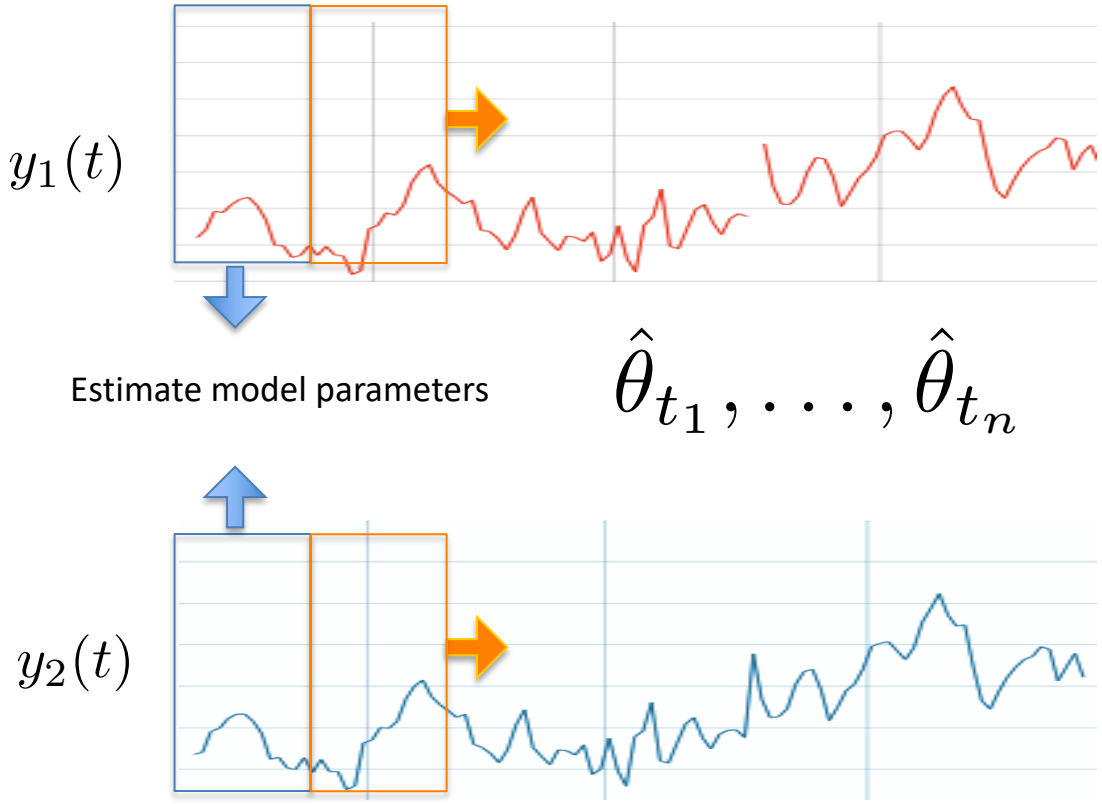
- ✓ Any i.i.d. feature (e.g., residual, measurements in a quality analysis applications)



# Features (3)

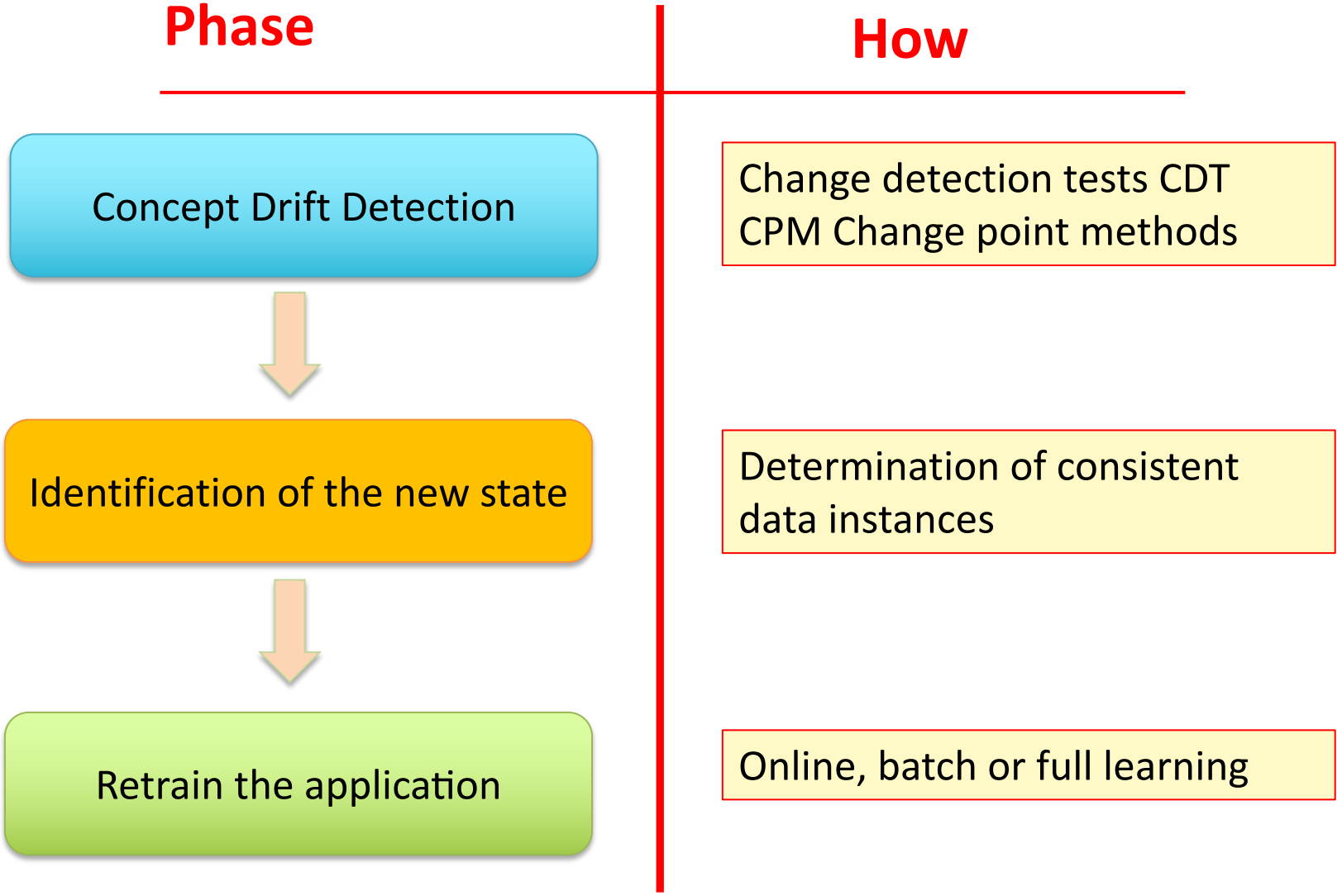
Model space

- ✓ LTI models are used to approximate the signal




$$y_1(t) \rightarrow \boxed{f_\theta^{1,2}} \longrightarrow y_2(t)$$

# Active learning



# Concept drift detection

Ad hoc triggers designed to detect changes by inspecting sequences of data or derived features

Data-based methods

- Limit checking
- Binary threshold

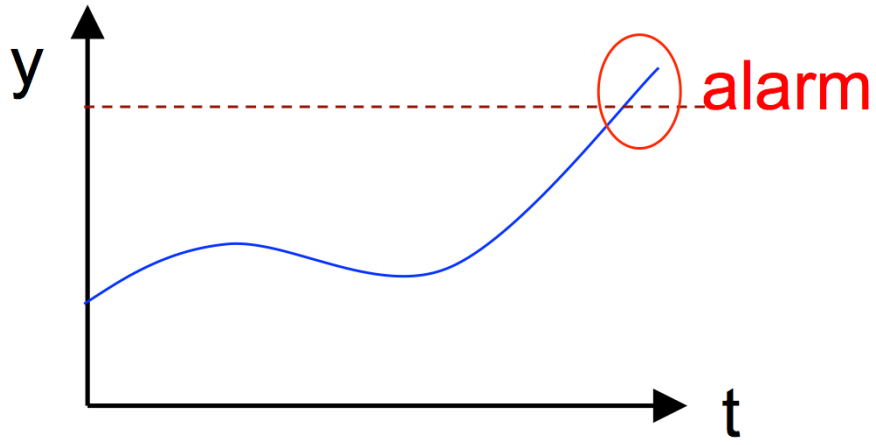
Statistical-based methods

- Statistical Hypothesis tests
- Change-Point Methods
- Change detection tests

# Limit checking

Testing if a given (measured) variable exceeds (indicating a change) or not a known absolute limit.

$$y(t) \leq Y_{\text{lim}} \rightarrow F(t) = 0$$
$$y(t) > Y_{\text{lim}} \rightarrow F(t) = 1$$




Variants:

- Two limits, associated to different levels of safety.
- Use of superior and inferior limits.

Easy to implement.

Too conservative (low change sensitivity).

# Change-detection tests

*Change detection tests are methods designed to detect variations in the pdf of the process generating the data*

Parametric approach: knowledge of the pdf before and after the change

- CUSUM test
- Shiryaev-Robert test

Nonparametric approach:

- CI-CUSUM test, NPCUSUM test
- ICI-based change detection test

Semi-parametric approach:

- Semiparametric log-likelihood criterion (SPLL)

# The CUSUM test

$$X = \{x_1, x_2, .., x_N\} \sim p_\theta(x)$$

The change at $t_0$ modeled as a transition from $\theta_0$ to $\theta_1$ (Hp: we keep the pdf structure)

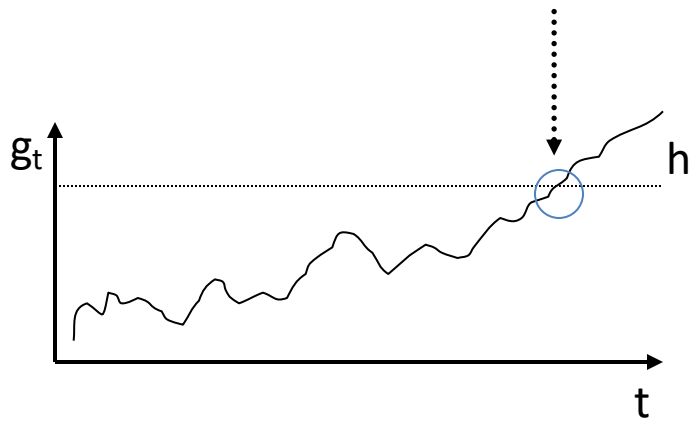
Measure a discrepancy at time time t: $s_t = \ln \frac{p_{\theta_1}(x_t)}{p_{\theta_0}(x_t)}$ Kulback-Leibler

Evaluate the cumulative sum $S_t = \sum_{i=1}^{t} s_t$

CUSUM identifies a change at time $\bar{t}$

when $g_t = S_t - m_t \geq h\big|_{\bar{t}}$

with $m_t = \min_{1 \leq i \leq t}(S_t)$



*The answer to the question "what happened?" is not enough ...*

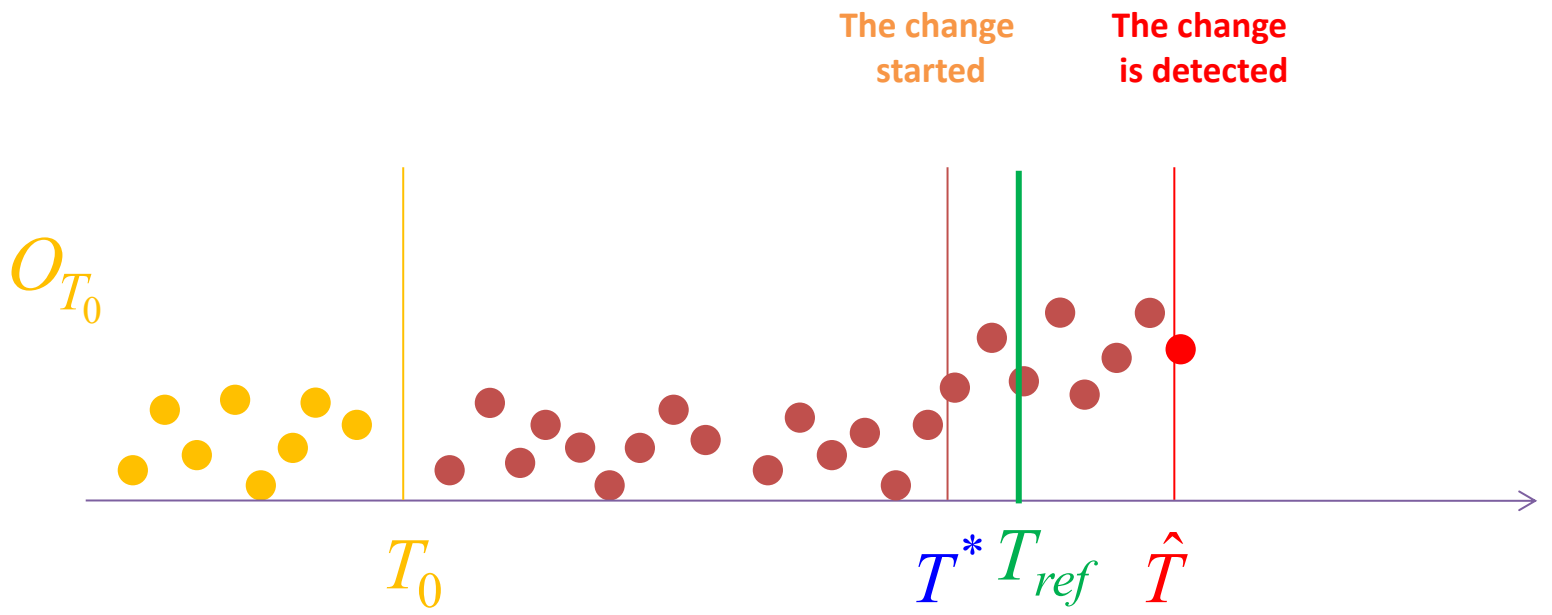
"Apparently you collapsed when told the price of these ..."

*... Tell me: "when did it happen?"*

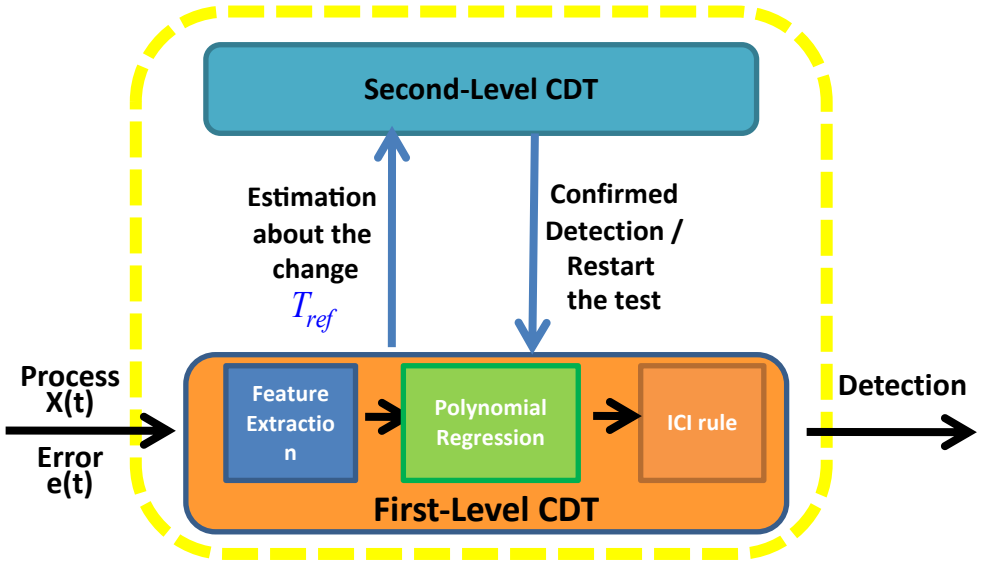
# What and when …

- Not only detection of the change, but also **estimation of the time instant** the process becomes non stationary

After the detection, we want an estimate $T_{ref}$ of $T^*$
by means of a refinement procedure



# Hierarchical CDT




**Second level** change-detection test aiming at confirming (or not) the change hypothesis:

- Multivariate hypothesis test
- Change-point methods

A **multivariate hypothesis** test based on the Hotelling T-square statistics

$$S = (\bar{F}^0 - \bar{F}^1)' \left( \left( \frac{1}{n_0} + \frac{1}{n_1} \right) \Sigma \right)^{-1} (\bar{F}^0 - \bar{F}^1),$$

$$\left( \frac{n_0+n_1-2}{n_0+n_1-N-1} \right) \mathcal{F}(N, n_0 + n_1 - N - 1),$$

**Change-point methods:** statistical tests able to assess whether a given data-sequence contains (or not) a change point

$$\mathcal{X} = \{x(t), t = 1, \ldots, n\}, \begin{cases} \mathcal{A}_\tau = \{x(t),\ t = 1, \ldots, \tau\}, \\ \mathcal{B}_\tau = \{x(t),\ t = \tau+1, \ldots, n\}, \end{cases}$$

Compute $\mathcal{T}_\tau = \mathcal{T}(\mathcal{A}_\tau, \mathcal{B}_\tau),$

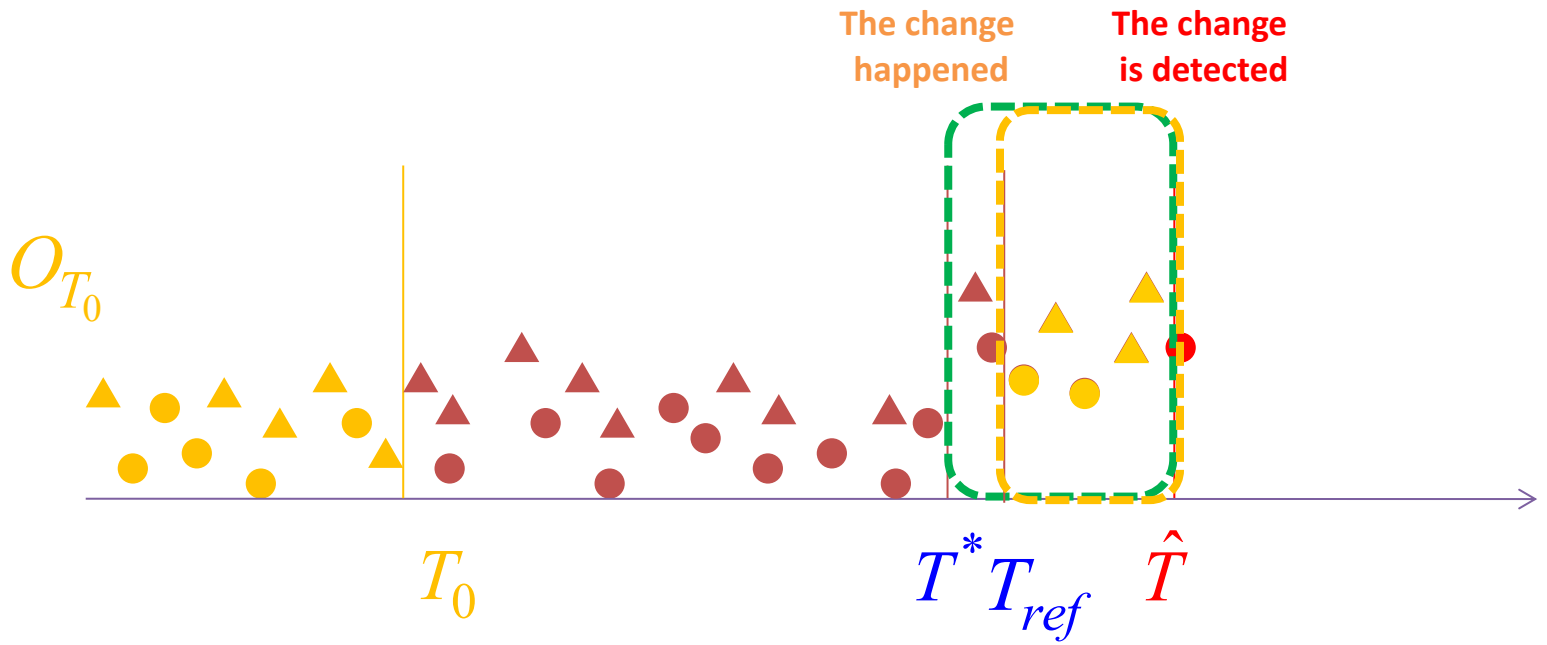
$$\mathcal{T}_M = \max_{s=1,\ldots,n} (\mathcal{T}_\tau)$$

$$\begin{cases} \text{The estimated change-point in } \mathcal{X} \text{ is } M_{\mathcal{X}} & \text{if } \mathcal{T}_M \geq h_{n,\alpha} \\ \text{No change-point identified in } \mathcal{X}, & \text{if } \mathcal{T}_M < h_{n,\alpha} \end{cases}$$

# Which data are consistent with the current status?

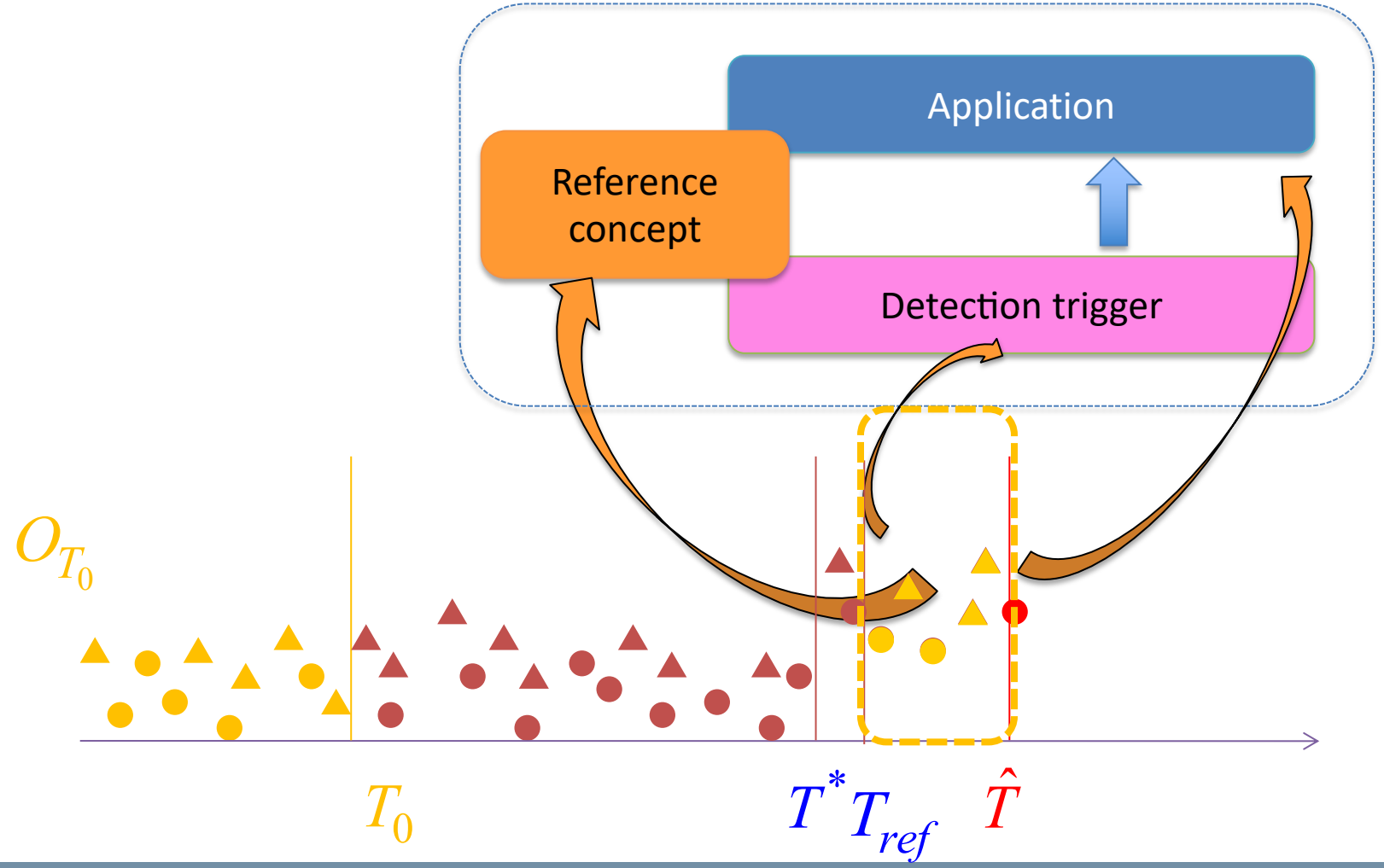
Instances: between $T^*$ and $\hat{T}$

T* is unknown: use estimates $T_{ref}$ and $\hat{T}$



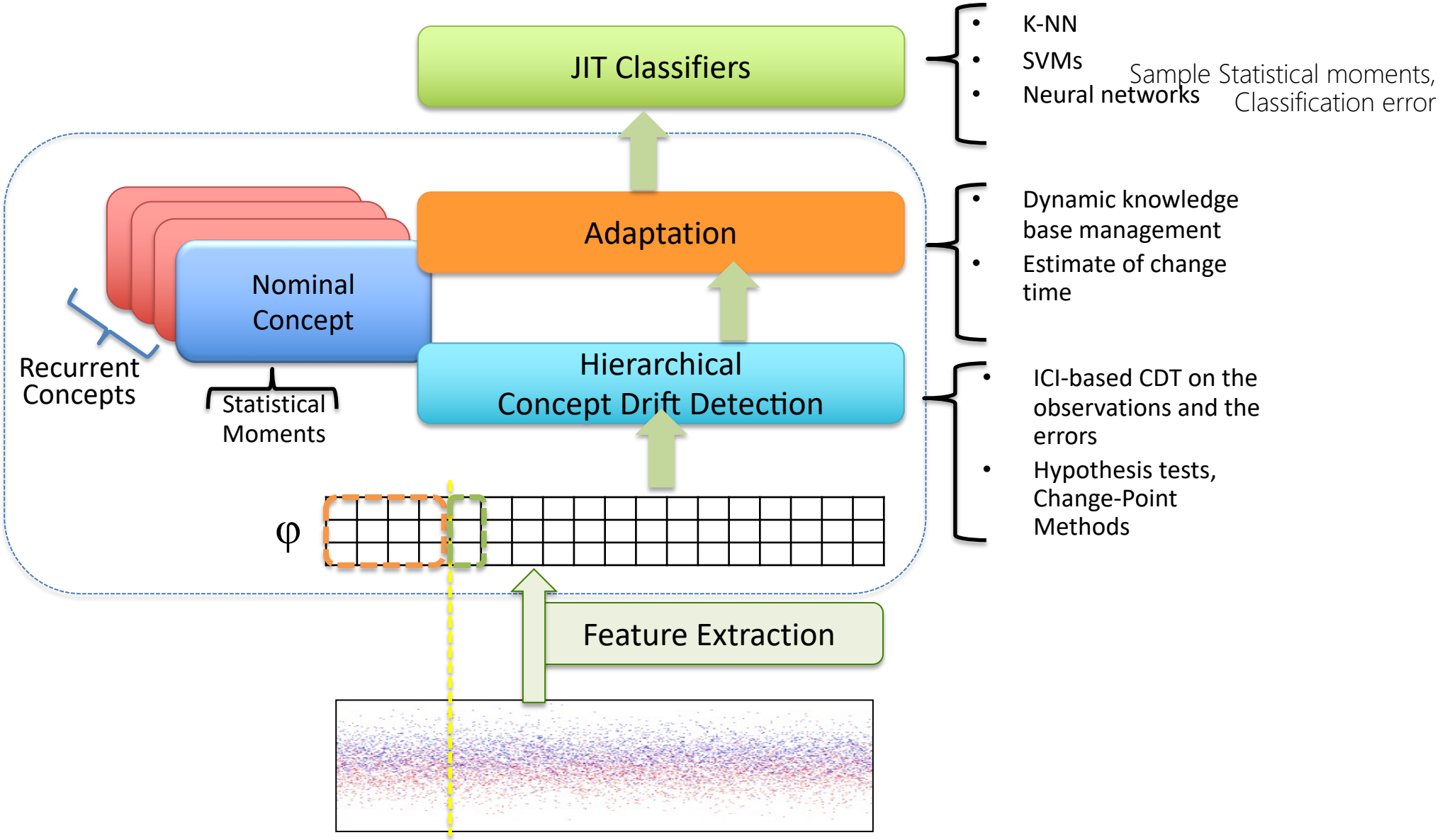
# Retrain the application

- If concet drift is detected the whole framework is retrained



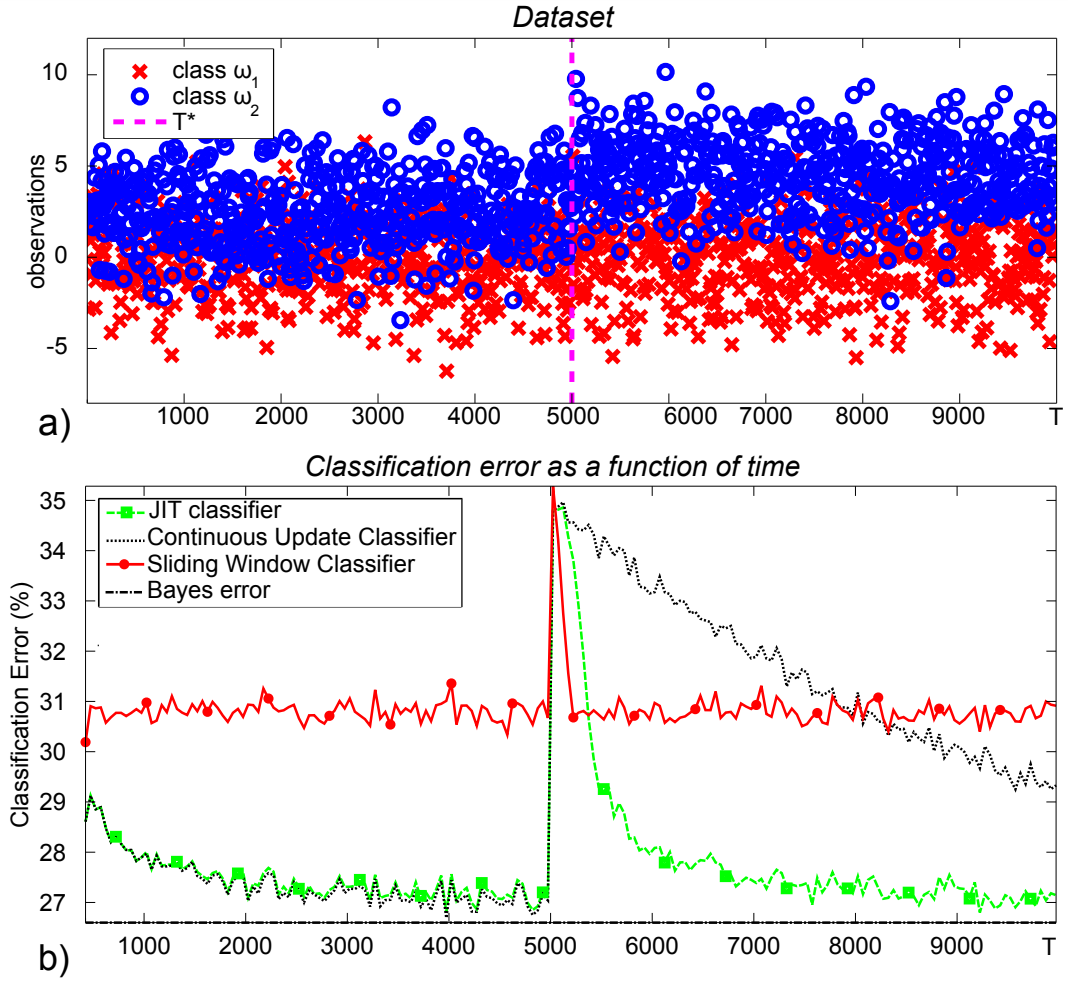
# An example:
# Just-in-Time Adaptive Classifiers

# Just-in-Time Adaptive Classifiers



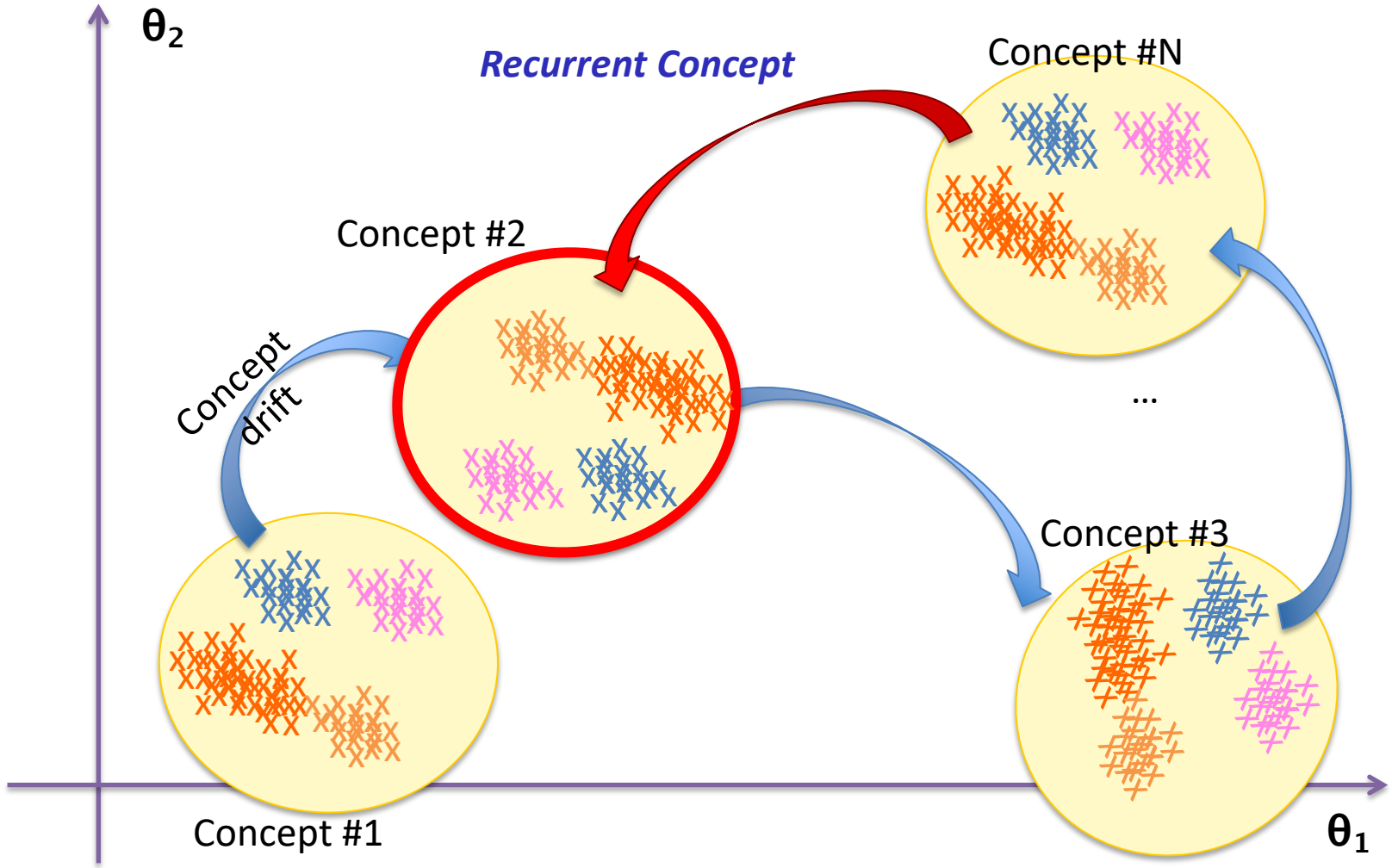
# Asymptotic optimality with JIT classifiers

JIT adaptive classifiers grant asymptotic optimality when the process generating the data is affected by a sequence of abrupt concept drift




Gaussian classes

# Dealing with concept drift …




$$p(x|t) = p(\omega_1|t)p(x|\omega_1, t) + p(\omega_2|t)p(x|\omega_2, t)$$

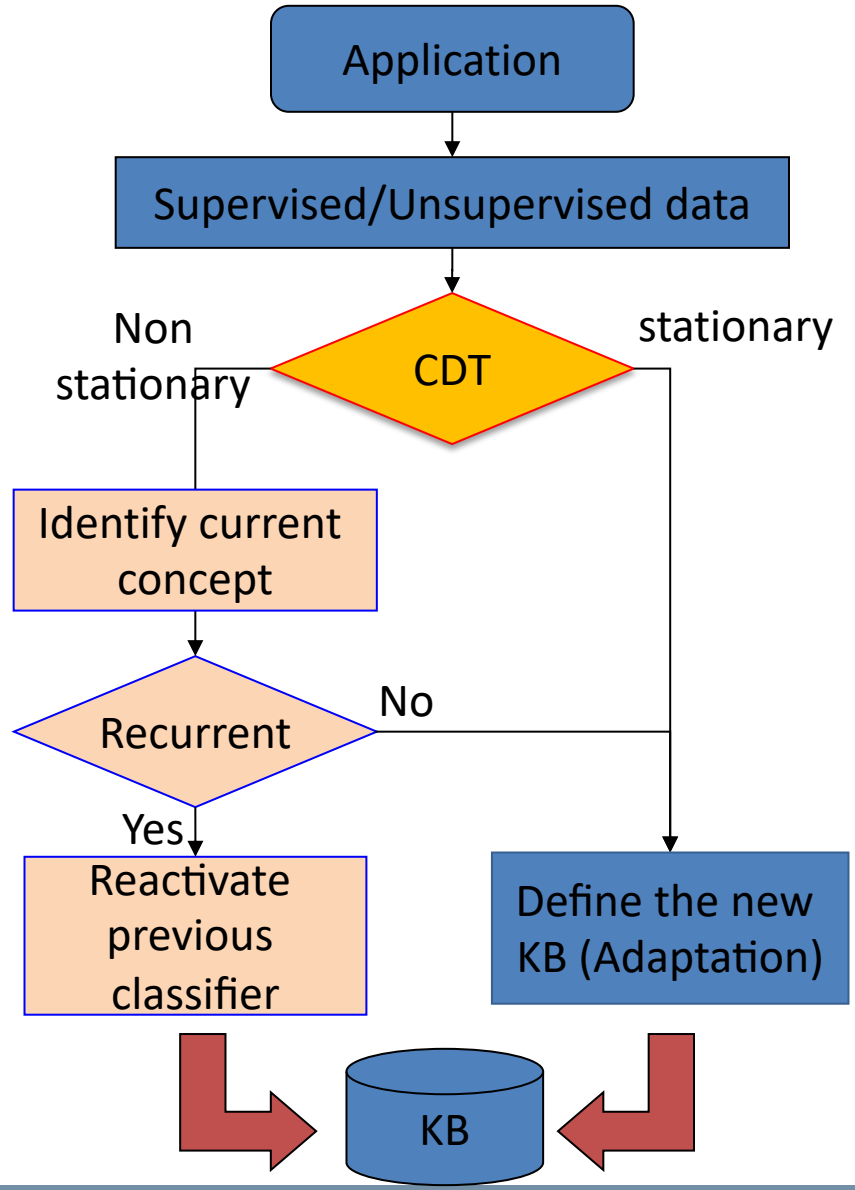
# The novel idea: extending the JIT classifier

**Two** CDTs are to asses if:

- The **pdf** of the **input** is **stationary**
- the **classification error** is **stationary**

Adaptation phase consists in:

- **Isolation of the current concept**
- Identification of **recurrent concepts**
- Training the classifier by exploiting all the **available supervised information**



# Some relevant remarks …

- ✓Being acquainted with learning techniques is a plus in everybody's background
- ✓Most of the time we can assume that the process generating the data is time invariant. When it is not we need to pay attention…
- ✓Learning in a changing environment must be considered and represents a key property intelligent systems should possess