Online learning and Continual learning

A sequence of small transfers

Antoine Cornuéjols

AgroParisTech – INRAE MIA Paris-Saclay

EKINOCS research group

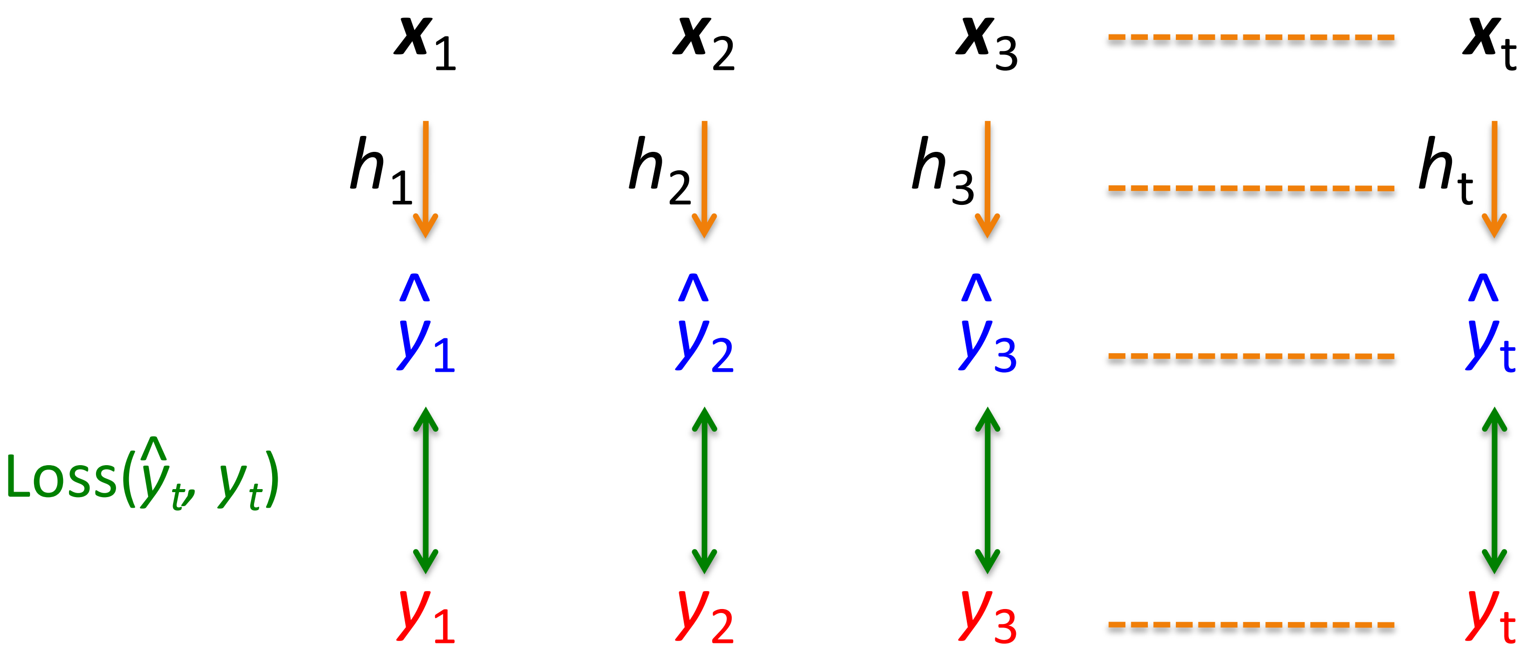
Can we say something **if** the teacher acts as an adversary?

Outline

1. Online learning: motivation, scenario, measure of performance
2. Theoretical framework: learning against any sequence
3. Heuristic approaches
4. Conclusion

The online learning scenario

A stream:



E.g. ***Choice of melons***. I see one, I make a prediction about its tastiness, then I eat it and know the answer.

Requirements

1. Process **an instance at a time**, and inspect it (at most) once
2. Use a **limited amount of time** to process each instance
   - Constant time for each instance
3. Use a **limited amount of memory**
   - Sublinear in the number of instances, and constant if possible
4. **Anytime** algorithm: be ready to provide an answer at any time
5. **Adapt** to temporal changes

Online learning **applications**

- **Sensor** data and the Internet of Things
  - Cities with sensors to monitor mobility of people, check the state of bridges and roads, ...
- **Telecommunication** data
  - Adapt the networks
- **Social** media
  - Topic and community discovery
  - Sentiment analysis
- **Marketing** and e-commerce
  - Detection of fraud in electronic transactions
  - Change of preferences of the consumers (fashion, prices changes, ...)

Online learning **applications**

- **Health** care
  - Monitoring patient vital signs
  - Telemedicine
- Epidemic and disasters
  - Following (and anticipating) the trends
- Computer **security**
  - Intrusion detection
- **Electricity** demand prediction

How to build a theory?

Non stationary

environment

Real risk: expected loss

$$R(h) \;=\; \mathbb{E}[\ell(h(\boldsymbol{x}), y)] \;=\; \int_{x \in \mathcal{X}, y \in \mathcal{Y}} \ell(h(\boldsymbol{x}), y)\; \mathbf{P}_{\mathcal{X}\mathcal{Y}}\; d(\boldsymbol{x}, y)$$

But $\mathbf{P}_{\mathcal{X}\mathcal{Y}}$ is unknown, then use: $\mathcal{S}_m = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m$

Empirical risk Minimization

$$\hat{h} \;=\; \underset{h \in \mathcal{H}}{\text{ArgMin}}\big[R_m(h)\big] + \text{Reg}\big] \;=\; \underset{h \in \mathcal{H}}{\text{ArgMin}}\left[\frac{1}{m}\sum_{i=1}^{m} \ell(h(\boldsymbol{x}_i), y_i)\right] + \lambda\, \text{Capacity}(\mathcal{H})\big]$$

1. All examples are equal: **no forgetting**
2. Commutative criterion: **no information from the sequence**

The **statistical** theory of learning

- What does allow « generalization » and induction??

> - Link between the past and the future:
>
>   distributions $P_{\mathcal{X}}$ et $P_{\mathcal{Y}|\mathcal{X}}$ are supposed stationnary
>
> - I.i.d. data

But … the world is constantly **evolving**

- New types of data

  - Data are made available through *unlimited streams* that continuously flow, possibly at high-speed
  - The underlying *regularities may evolve over time* rather than be stationary
  - The data is now often *spatially as well as time situated*

➡ Data can **no longer** be considered as **independently** and **identically distributed**

The question of the evaluation of learning

- Problems
  - Deciding is intermixed with learning
  - The environment may be changing

- Holdout evaluation (standard)

- **Prequential** evaluation (**pre**diction and se**quential**)
  - Aggregation of the **number of errors** of prediction **during learning**

In the **online** setting …

1. There cannot be any notion of **generalization**
   - Which implies a **future** that is like the **past**

2. There is **no distinction** between
   - A **training** phase
   - A **test** phase

“Online” learning

- Learning against **any** (!!!) sequence
  - **No** longer a **stationary** environment assumption
  - **Nor** any **temporal** regularity!!!

- In these conditions, *how can one measure if the learning algorithm is good?*
  - **No** possible **test set**
  - The performance can be arbitrarily bad (against an omniscient adversary)

- Idea of comparison with a committee of “experts” (N experts)

The notion of “regret”

The notion of “regret”

- At each time, **I had the choice** between several decisions

The notion of "regret"

- At each time, **I had the choice** between several decisions

- A posteriori, did I perform much worse **than the best decision** maker **known afterwards**?

The notion of "regret"

- At each time, **I had the choice** between several decisions
- A posteriori, did I perform much worse **than the best decision maker known afterwards**?
  - **Regret** with respect to **one** "expert" *E*

$$R_{E,T} = \sum_{t=1}^{T} \Big[\ell\big(h_t(\mathbf{x}_t), y_t\big) - \ell\big(f_t^E(\mathbf{X}_t, y_t)\big) \Big] = \widehat{L_T} - L_{E,T}$$

The notion of “regret”

- At each time, I **had the choice** between several decisions

- A posteriori, did I perform much worse **than the best decision** maker **known afterwards**?

  – **Regret** with respect to **one** “expert”

$$R_{E,T} \;=\; \sum_{t=1}^{T} \Big[\ell\big(h_t(\mathbf{x}_t), y_t\big) \;-\; \ell\big(f_t^E(\mathbf{X}_t, y_t)\big) \Big] \;=\; \widehat{L_T} \;-\; L_{E,T}$$

  – **Regret** with respect to **a set** of “experts” (the best one among them)

$$R_{\mathcal{E},T} \;=\; \widehat{L_T} \;-\; \underset{E \in \mathcal{E}}{\mathrm{Min}} \; L_{E,t}$$

Outline

1. Online learning: motivation, scenario, measure of performance

2. Theoretical framework: learning against any sequence

3. Heuristic approaches

4. Conclusion

Online learning

Against **any** sequence

- Examples described using:

  Number (1 or 2); ***size*** (small or large); ***shape*** (circle or square); ***color*** (red or green)

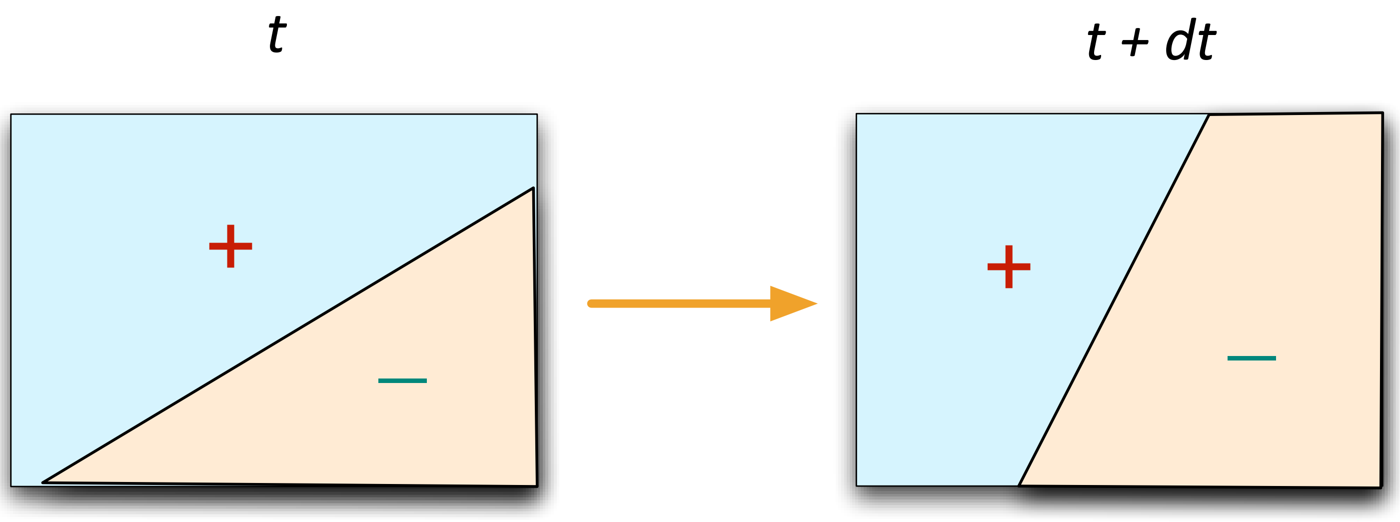
- They belong either to class '**+**' or to class '**-**'

Description	Your answer	True answer
1 large red square		-
1 large green square		

"Online" learning

- The scenario

$(X_{t-n}, Y_{t-n}), \ldots, (X_{t-2}, Y_{t-2}), (X_{t-1}, Y_{t-1}), (X_t, Y_t) \ldots (X_{t+1}, Y_{t+1})$?



- Performance criteria
  - Σ erreurs ; Mean error

Using a **committee of "experts"**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3	1	0	0	0	1	1
t_4	1	0	0	1	1	1
t_5	1	1	0	1	1	1
t_6	1	0	0	0	1	0

How to chose a decision at each time step *t*?

Algorithm to select one expert

- Choice of one **expert a priori, and then no change**
  - Properties?
    - Possibility of a infinite loss

Can we do better?

Using a **committee of "experts"**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	$\boxed{0}$	$\boxed{0}$	1	$\boxed{0}$	1
t_2						
t_3						
t_4						
t_5						
t_6						

Using a **committee of “experts”**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3						
t_4						
t_5						
t_6						

Using a **committee of "experts"**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3	1	0	0	0	1	1
t_4						
t_5						
t_6						

Greedy deterministic algorithm

Algorithm to select one expert

- **Greedy deterministic** algorithm
  - ***Properties***?
    - Can be very good
    - **Worst case?**

Greedy deterministic algorithm: **worst case**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
J_1	1	0	0	0	0	0
J_2	0	1	0	0	0	0
J_3	0	0	1	0	0	0
J_4	0	0	0	1	0	0
J_5	0	0	0	0	1	0
J_6	0	0	0	0	0	1

$$L \leq N(L^*) + N - 1$$

E.g. 6

Loss of the algo

Loss of the best expert

Algorithm to select experts

- **Greedy random** algorithm
  - ***Properties***?
    - Can be very good
    - **Worst case?**

$$L_{RG} \leq (\ln N + 1)\,(L^*) + \ln N$$

E.g. $N = 100$ & $L^* = 1$ => $L_{RG} \leq 11$!!

The "realizable" case

- Binary classification
- ∃an unknown expert which does not make error: $h_{i,t}(x_t) = y_t \quad \forall t$

The "realizable" case

- Binary classification
- ∃an unknown expert which does not make error: $h_{i,t}(x_t) = y_t \quad \forall t$
- Which strategy?

The "realizable" case

- Binary classification
- ∃an unknown expert which does not make error: $h_{i,t}(x_t) = y_t \quad \forall t$
- Which strategy?
  - We give **a weight** $w_t = 1$ to all experts
  - At each time step t
    - Take the **majority vote** as the decision: $H(x_t)$
    - **Compare** the predictions of each expert $h_{i,t}(x_t)$ with y_t
    - Set $\boldsymbol{w_t} = \mathbf{0}$ to all experts that **made an error**

$$L_{CR} \leq \lfloor \log_2 N \rfloor$$

The "realizable" case: proof

- Initially: $W_0 = N$
- At each time step: $$W_t \leq W_{t-1}/2$$

$$\boxed{L_{CR} \leq \lfloor \log_2 N \rfloor}$$

The "NON realizable" case

- At $t = 0$, $W_0 = N$ $\quad 0 < \beta < 1$

- For each t and expert i : $\quad w_i(t+1) = \begin{cases} w_i(t) & \text{if } y(t) \neq h_{i,t}(\mathbf{x}_t) \\ \beta w_i(t) & \text{if } y(t) = h_{i,t}(\mathbf{x}_t) \end{cases}$

$$W(t) \;\leq\; W(t-1)/2 \,+\, \beta\, W(t-1)/2$$

If, at t, majority was making a mistake

$$W(t) \;\leq\; W_0\, \frac{(1+\beta)^t}{2^t}$$

And the best expert so far has weight $\beta^{L^*(t)}$ thus: $\quad W(t) \;\geq\; \beta^{L^*(t)}$

Hence: $\quad \beta^{L^*(t)} \;\leq\; W_0\,(1+\beta)^t/2^m$ ← Nb of mistakes at t

$$L_{CR} \;\leq\; \left\lfloor \frac{\log_2 N + L^* \log_2(1/\beta)}{\log_2 \frac{2}{1+\beta}} \right\rfloor$$

Proof

$$
\begin{aligned}
2^m \, \beta^{L^*(t)} &\leq W_0 \, (1+\beta)^m \\
\beta^{L^*(t)} &\leq W_0 \left[\frac{1+\beta}{2}\right]^m \\
\log_2(\beta) \, L^*(t) &\leq \log_2(W_0) \, + \, m \, \log_2 \frac{1+\beta}{2} \\
m \, \log_2 \frac{2}{1+\beta} &\leq \log_2 W_0 \, + \, L^*(t) \, \log_2 \frac{1}{\beta}
\end{aligned}
$$

$$
m = L_{CR}(t) \leq \frac{\log_2 N \, + \, L^*(t) \, \log_2(\frac{1}{\beta})}{\log_2 \frac{2}{1+\beta}}
$$

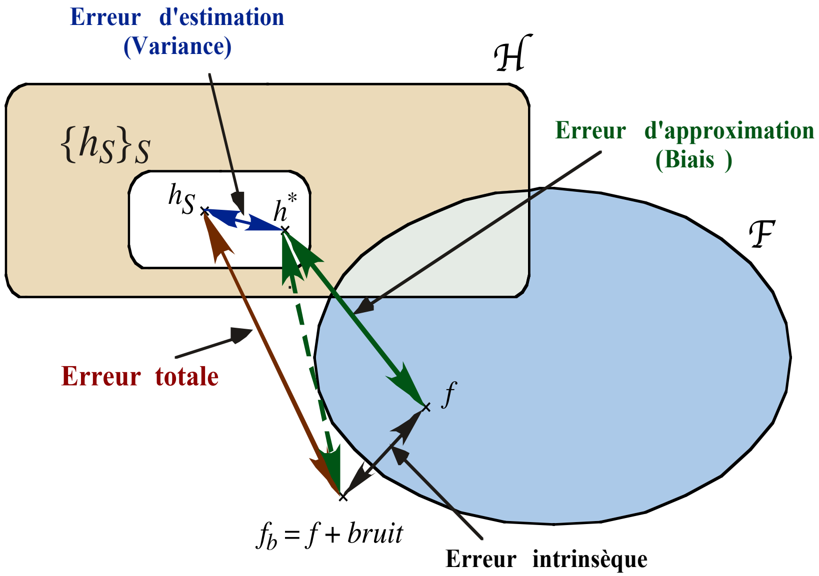
- Interpretation?

Interpretation

Capacity of the hypothesis space *H*

Error of the **best hypothesis** in *H*

$$m = L_{CR}(t) \leq \frac{\log_2 N + L^*(t) \log_2(\frac{1}{\beta})}{\log_2 \frac{2}{1+\beta}}$$



...

Another perspective on the problem

- At each time step, there exists a distribution $\mathbf{P}_t$ over the space $\mathcal{H}$ of hypotheses
- At each round of learning:
  - **Receive** instance $x_t \in \mathcal{X}$
  - **Choose** h_t randomly according to the current distribution $\mathbf{P}_t$ over $\mathcal{H}$
  - **Predict** $\hat{y}_t = h_t(x_t)$
  - **Receive** the true label y_t
  - **Computes** the new distribution $\mathbf{P}_{t+1}$ using the **Multiplicative Weight algorithm**

$$\mathbf{P}_{t+1} = \frac{\mathbf{P}_t(h)}{Z_t} \times \begin{cases} e^{(-\eta)} & \text{if } h(\mathbf{x}_t) \neq y_t \\ 1 & \text{otherwise} \end{cases}$$

The multiplicative weight technique

- Provides theorems of the form:

bound on the **learner's** cumulative loss in terms of:

the **cumulative loss** of the **best** strategy in hindsight

+ an **additional term** which can be shown to be relatively insignificant for large T

Assessment on this type of analysis

- Allows one to get **theorems**!!
- But **too demanding** and not realistic
- Interesting idea: **committee of experts**

  and **multiplicative weights**

Can you say what is the **difference** between:

1. Online learning with **expert advices**

2. **Bandit** problems

?

Outline

1. Online learning: motivation, scenario, measure of performance

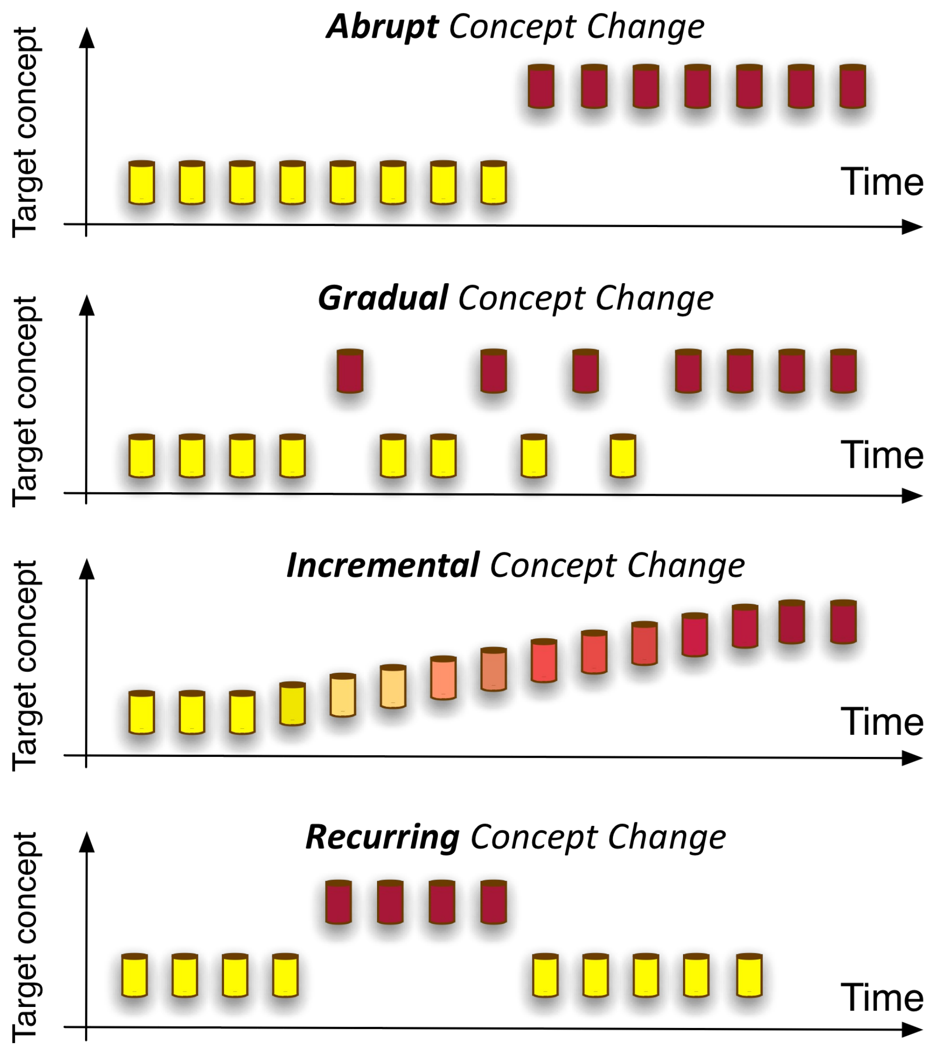
2. Theoretical framework: learning against any sequence

3. Heuristic approaches

4. Conclusion

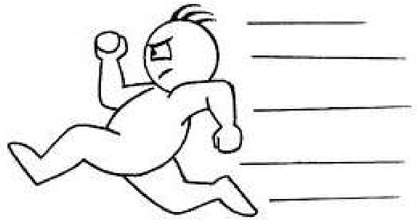
Concept changes

Types of concept changes

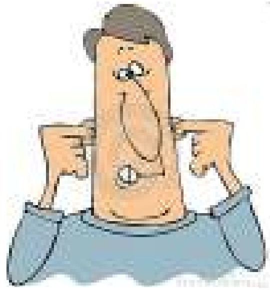


Desirable properties of a system that handle concept drift

- **Adapt** to concept drift **as soon as possible**



- Distinguish **noise** from **true changes**
  - Robust to noise but adaptive to changes



- Recognize and react to **recurring contexts**

- Adapt with **limited resources** (time and memory)

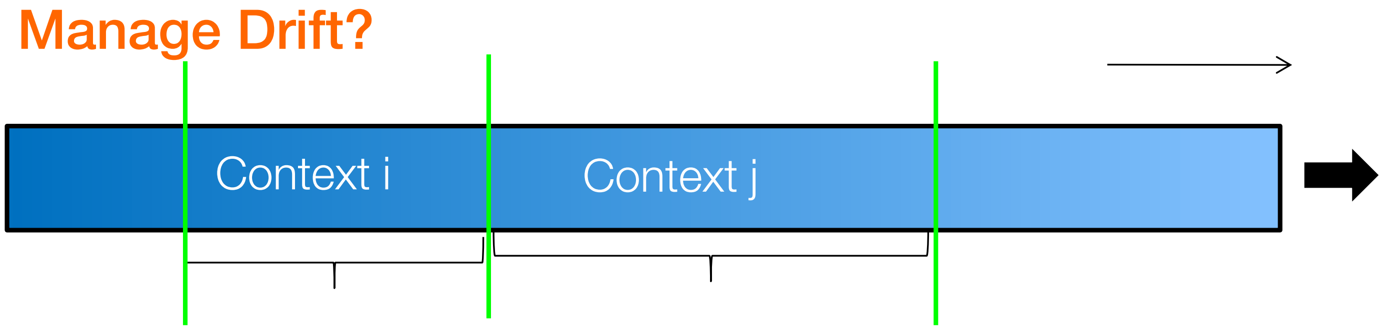
On-line adaptation

- *Assumption*: the current hypothesis h_t is **somewhat relevant** to label x_{t+1}.
  - A kind of **transfer** between successive "tasks"

➢ How one should **control** and tune this **transfer**?

  - What should be the **weight of the past**?
  - The **plasticity** vs. **stability** dilemma

Two types of approaches



1. Either **detect first**
   - Adapt statistics (summaries) and retrain the model
   - Or adapt the current model

2. **Adapt** the model **continuously**
   - A single model
   - An ensemble of models

Outline

1. Online learning: motivation, scenario, measure of performance
2. Theoretical framework: learning against any sequence
3. Heuristic approaches
4. Conclusion

Outline

1. Online learning: motivation, scenario, measure of performance

2. Theoretical framework: learning against any sequence

3. **Heuristic** approaches

    1. **Detection**-based methods

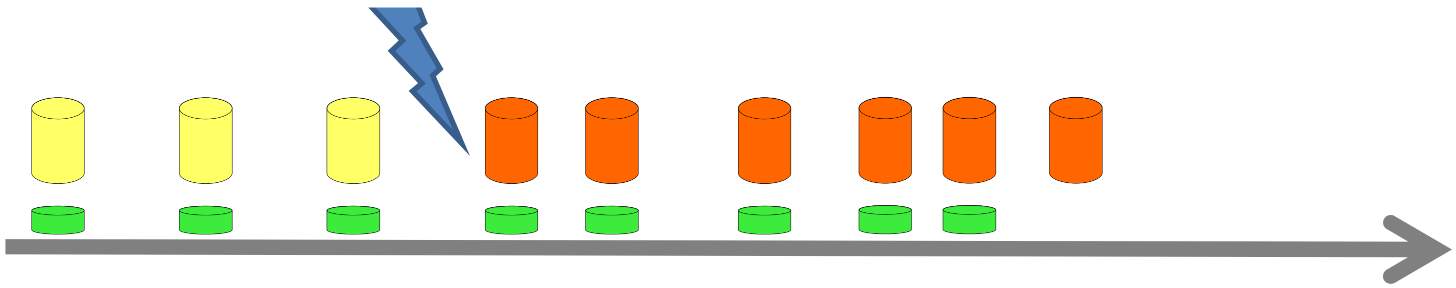
    2. Adaptation-based methods

4. Conclusion

Heuristic approaches:

Detection-based methods

Variable training windows

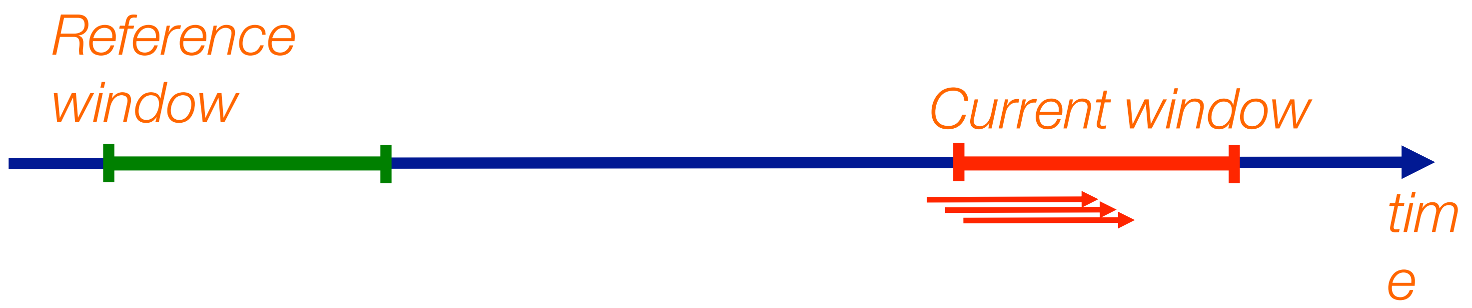


detect a change

- Problem: how to detect a “**true**” change?

How to detect a drift

- Main idea : if the distributions of the "*current window*" and the "*reference window*" are significantly different, that means a drift is occurring ….



…

Definition of a change in a data stream?

- Statistical properties of the data **change more** than what can be attributed to **chance fluctuations**

The CUSUM test

Designed to give an alarm when the mean of the input data significantly deviates from its previous value

- Given a sequence of observations $\{x_t\}_t$, define $z_t = (x_t - \mu)/\sigma$, where $\mu$is the expected value of and σ is their standard deviation in “normal” conditions
- If μ and σ are not known a priori, they are estimated from the sequence itself.
- The CUSUM computes the indices and alarm:

$$\begin{cases} g_0 = 0 \\ g_t = \max(0, g_{t-1} + z_t - k) \\ \text{If } g_t > h, \text{ declare change and reset } g_t = 0, \text{ and } \mu \text{ and } \sigma \end{cases}$$

k and *h* are parameters to be given

Adapting to the Change

- **ADWIN** (average value in windows of training data)

[A. Bifet. Adaptive learning and mining for data streams and frequent patterns. ACM SIGKDD Explorations Newsletter, 11(1):55–56, 2009.]

- **DDM** (monitor the number of errors)

[J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In Advances in Artificial Intelligence–SBIA 2004, pages 286–295. Springer, 2004.]

- **EDDM** (monitor the distance between errors)

[M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno. Early drift detection method. Fourth International Workshop on Knowledge Discovery from Data Streams, 2006.]

Concept change

- … always the problem of controlling what to memorize

The dilemma ***plasticity-stability***

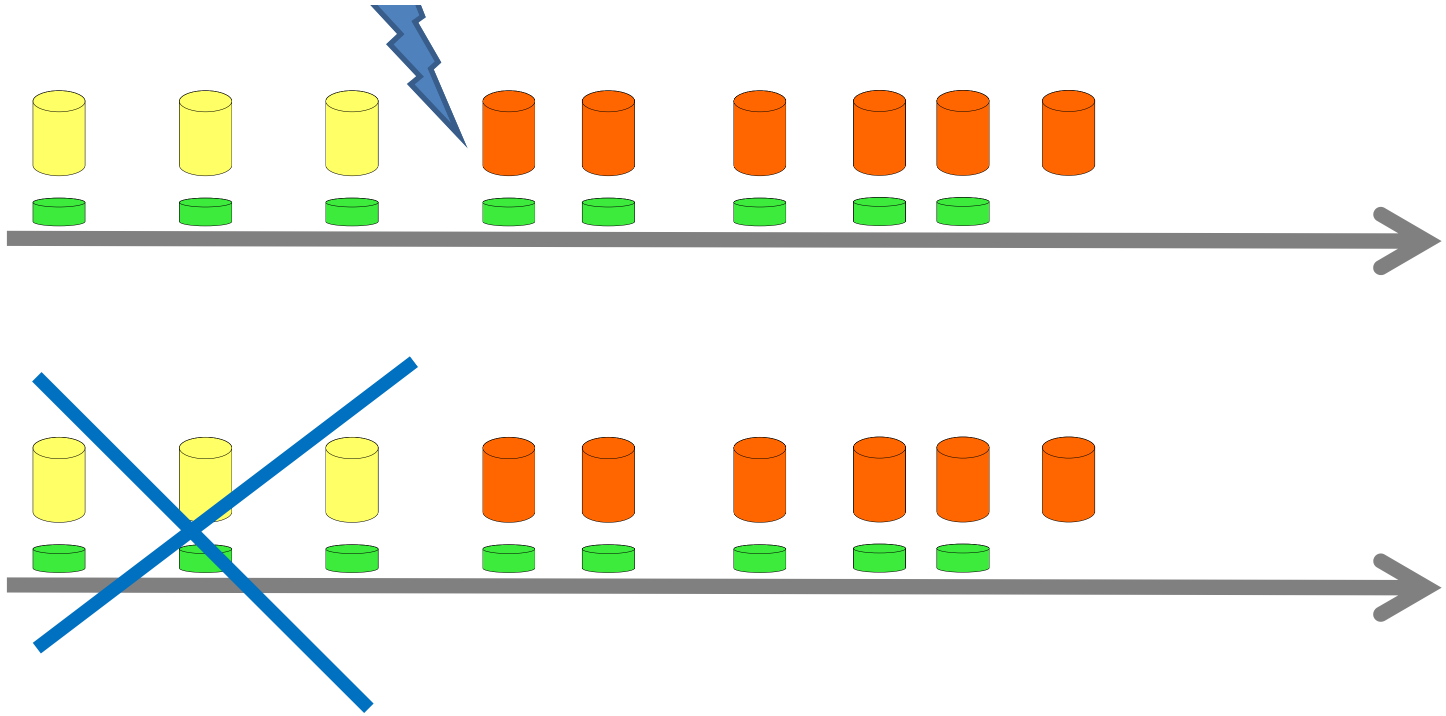
Fixed sliding windows

How to choose the size?

- **Small** window size
  - Fast **adaptability**
  - **Less** precision
- **Large** window size
  - **Good and stable learning results** if the environment is stationary
  - Does **not react quickly** to concept changes

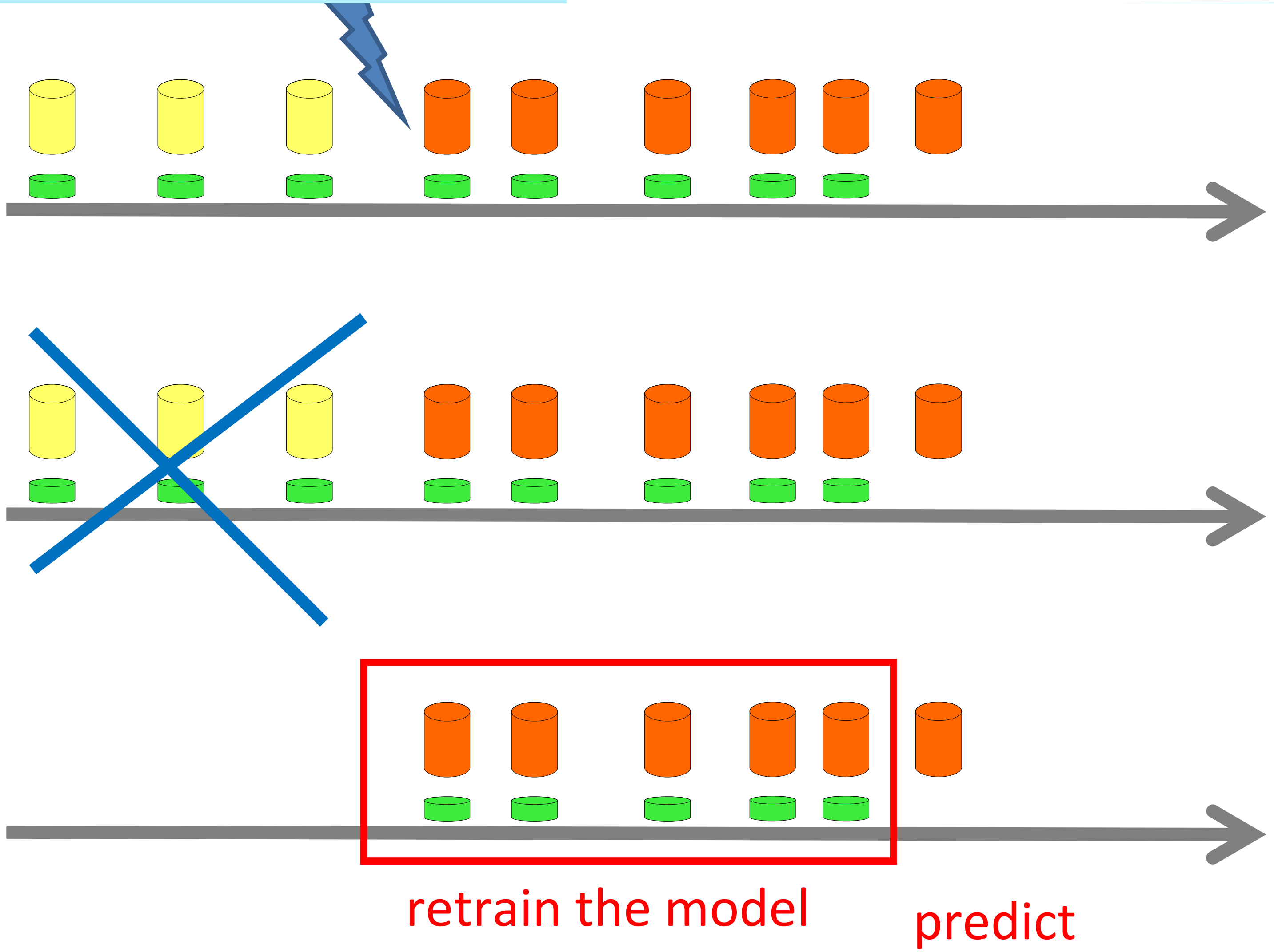
Fixed sliding windows

Variable training windows



drop old data

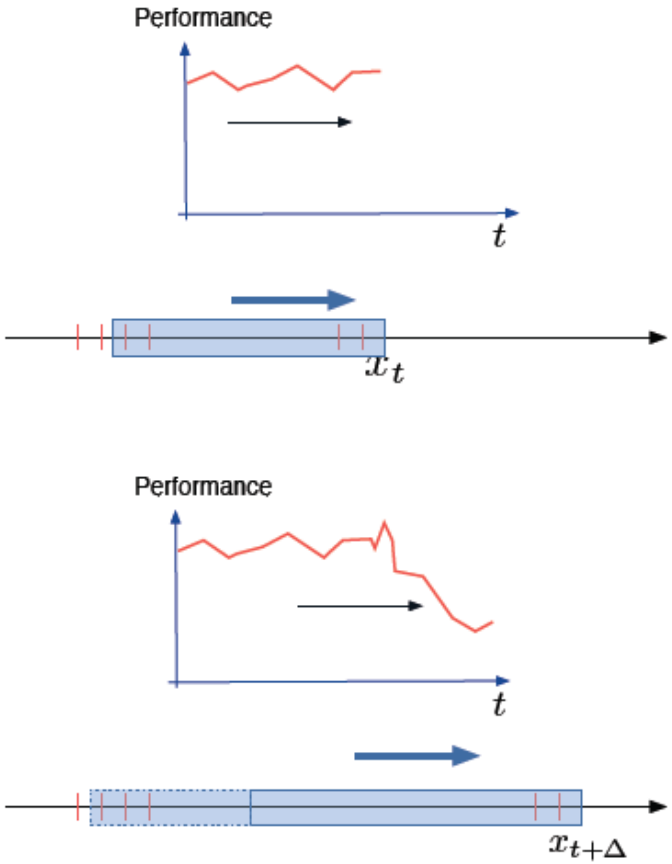
Variable training windows



- Pb: how to select the right window size?

Concept drift: **adaptive sliding windows**

Principle:



WK96 G. Widmer and M. Kubat (1996) *"Learning in the presence of concept drift ans hidden contexts"* Machine Learning 23: 69–101, 1996.

ADWIN: Adaptive Sliding WINdow

- Tries to optimize the trade-off between **reacting quickly** to changes and **having few false alarms**
  - Have **long windows** to have robust estimates
  - Have **short windows** to detect a change as soon as it happens

- ADWIN keeps a **variable-length window** of recently seen items, with the property that the window has the maximal length statistically consistent with the hypothesis "there has been no change in the average value inside the window"
  - An old fragment of the window is **dropped if and only if** there is enough evidence that its average value differs from that of the rest of the window
  - Two consequences:
    1. **Change is** reliably **detected** whenever the window shrinks
    2. At any time, **the average** over the current window can be used as a **reliable** estimate of the current average in the stream

ADWIN: Adaptive Sliding WINdow

The algorithm is parameterized by a test $T(W_0, W_1, \delta)$ (δ is a parameter of the algorithm) that compares the average of two windows W_0 and W_1 and decides whether they are likely to come from the same distribution. A good test should satisfy the following criteria:

- If W_0 and W_1 are generated from the same distribution (no change), then with probability at least $1 - \delta$ the test says "no change"
- If W_0 and W_1 were generated from two different distributions whose average differs by more than somme quantity $\epsilon(W_0, W_1, \delta)$, then with probability at least $1 - \delta$ the test says "no change".

Let assume that the current stream of items x_t is stored as a sequence of b subsequences. For i in $1 \ldots b-1$, let W_0 be formed by the i oldest subsequences, and W_1 be formed by the $b-i$ most recent ones, then perform the test $T(W_0, W_1, \delta)$.

- If some test returns "change", it is assumed that change has occurred somewhere and **the oldest subsequence is dropped** ; the window has shrunk by the size of the dropped subsequence.
- If no test returns "change", then **no subsequence is dropped**, so the window increases by 1.

...

ADWIN: Adaptive Sliding WINdow

W is the size of the longest window preceding the current item on which the test T is unable to detect any change.

The memory used by ADWIN is $O(\log W)$ and its update time is $O(\log W)$.

Often, the Hoeffding-based test T is used.

$$\forall \ \varepsilon \ > \ 0$$

$$Pr\left[\left|\frac{\sum_{i=1}^{n} X_i}{n} - \mathbb{E}[X]\right| > \varepsilon\right] \ \leq \ 2\exp(-2\varepsilon^2 n)$$

Window W_1 Window W_0

...

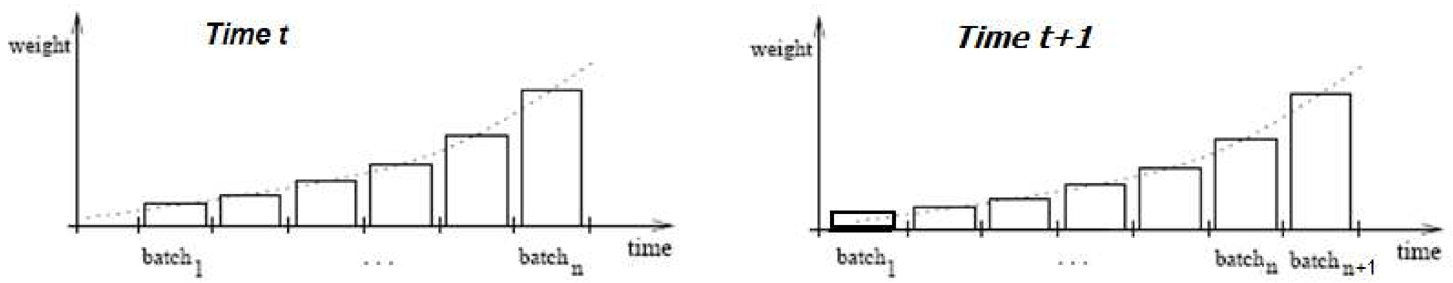
Instance weighting methods

- Examples are weighted depending on their age or relevance regarding the current concept
  - Store in memory sufficient statistics over all examples

- Recent examples are given more weight than past ones
  - Often an exponential weighting mechanism is used

    => decide on a decay factor λ

$$w(\mathbf{x}) = e^{-\lambda\, t_{\mathbf{x}}}$$



Outline

1. Online learning: motivation, scenario, measure of performance
2. Theoretical framework: learning against any sequence
3. **Heuristic** approaches
    1. Detection-based methods
    2. Adaptation-based methods
4. Conclusion

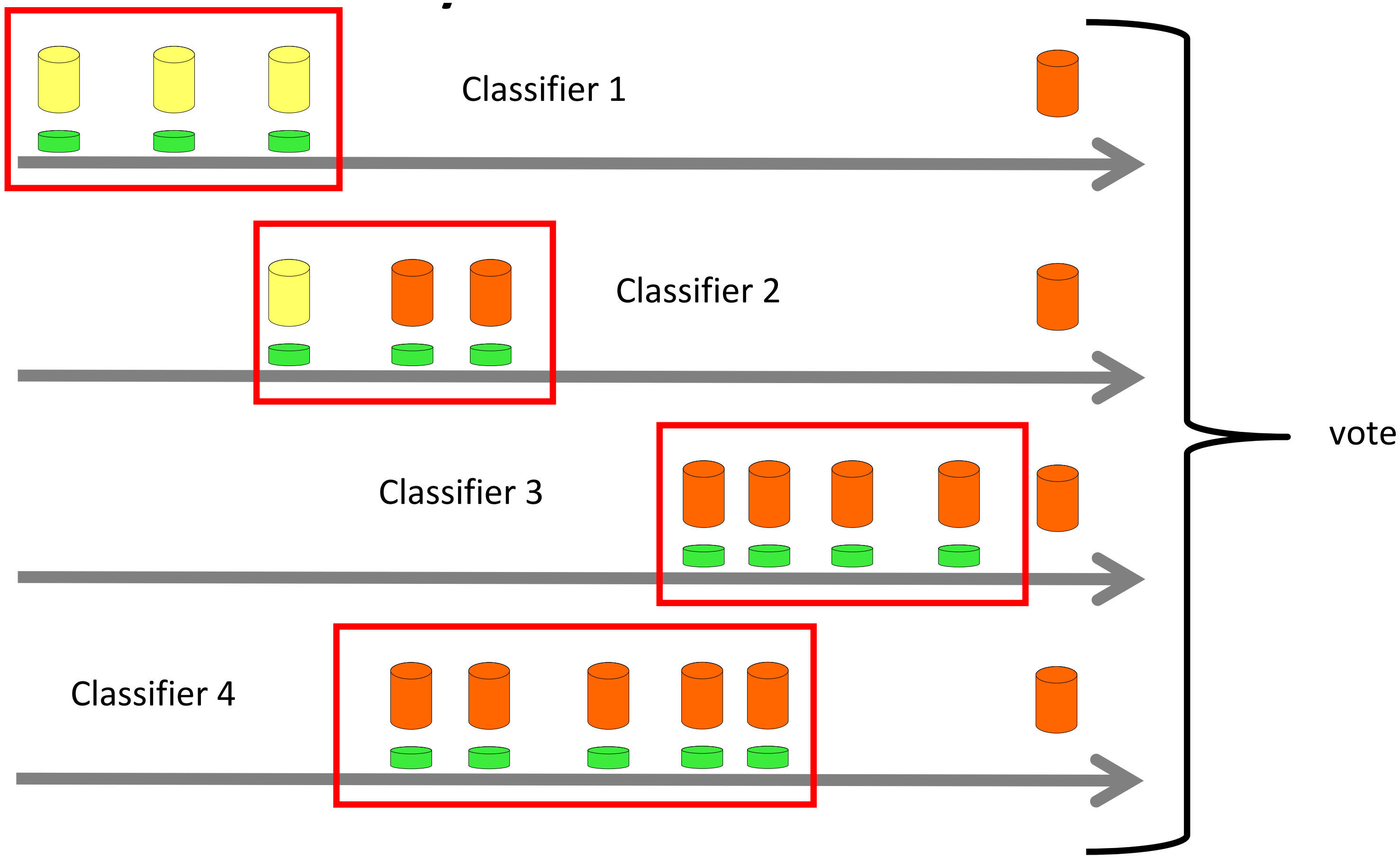
Heuristic approaches:

Adaptation-based methods

Concept drift: **ensemble methods**

- Learn **experts** on **various windows**
- **Weight** the experts depending on their (recent) performance
- **Replace** the worst experts

Ensemble methods



Concept drift: **ensemble methods**

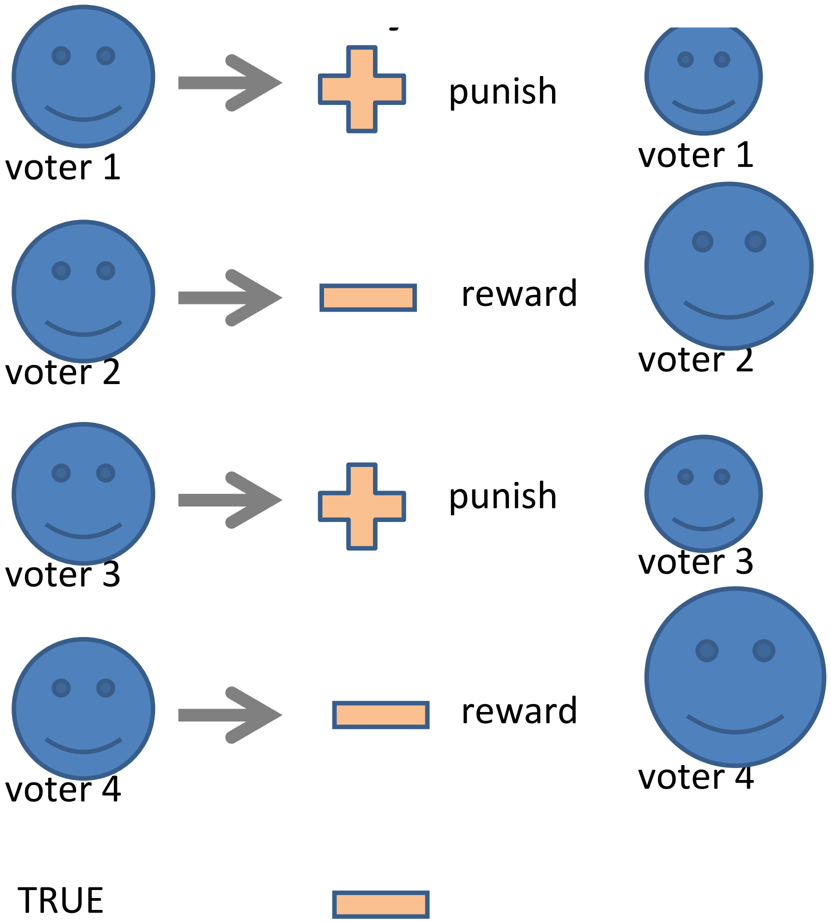
Dynamic weighted majority

- Classifiers in ensemble have initially a weight of 1
- For each new instance:
  - If a ***classifier predicts incorrectly***, reduce its weight
  - If ***weight drops below threshold***, remove classifier
  - If ***ensemble*** then ***predicts incorrectly***, install new classifier
  - Finally, all classifiers are (incrementally) updated by considering new instance

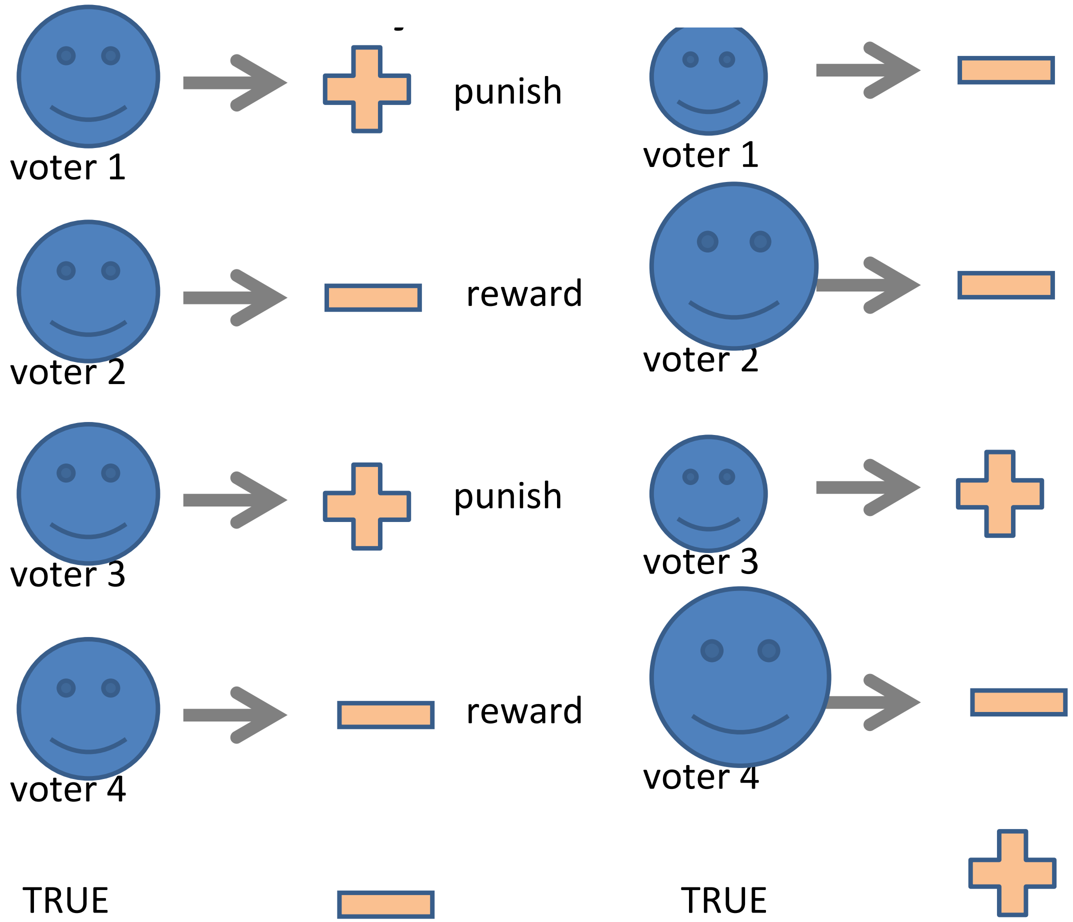
KM03 Kolter, Maloof (2003) *"Dynamic weighted majority: a new ensemble method for tracking concept drift"* ICDM 2003, 123-130.

Ensemble methods

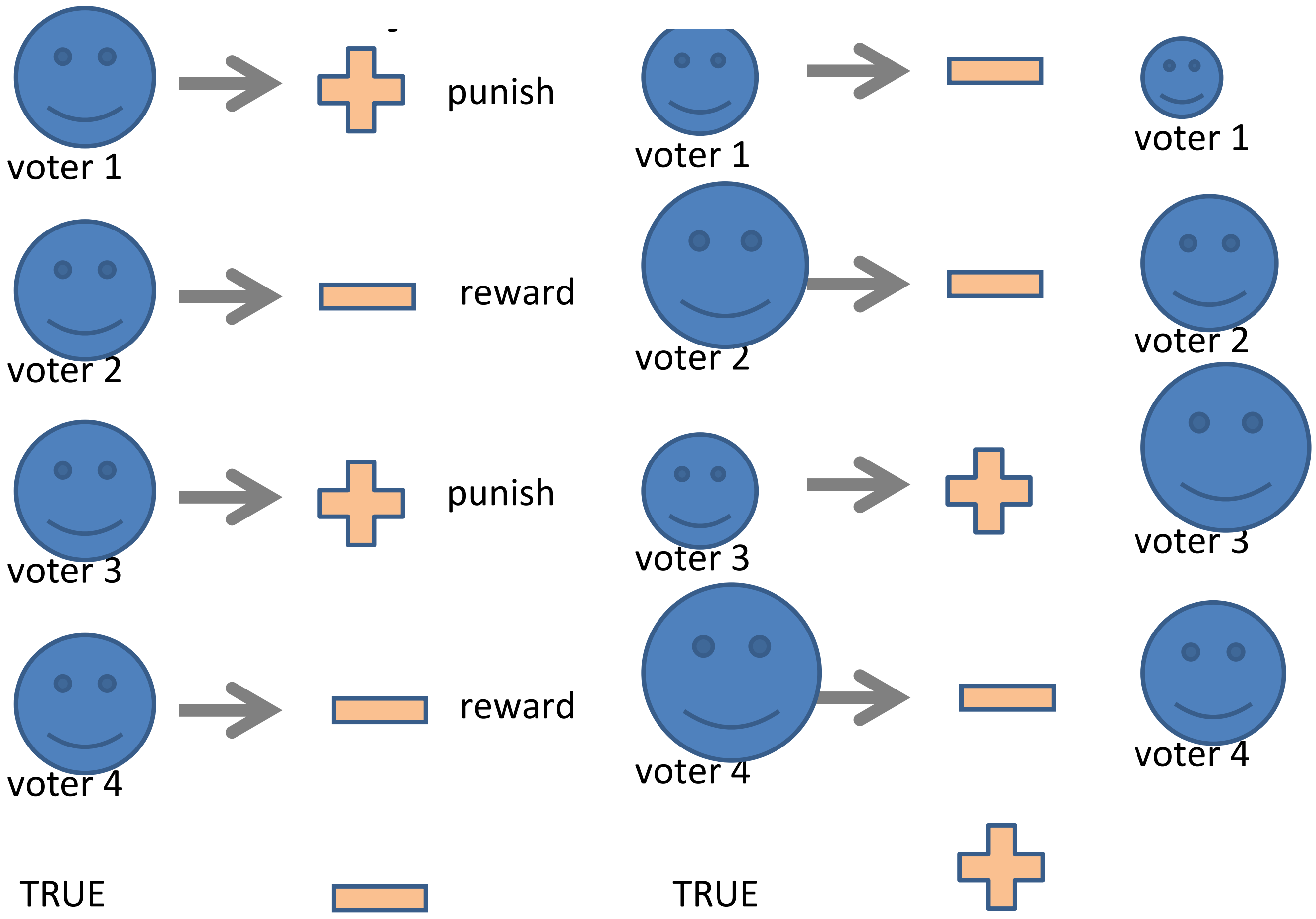
Ensemble methods



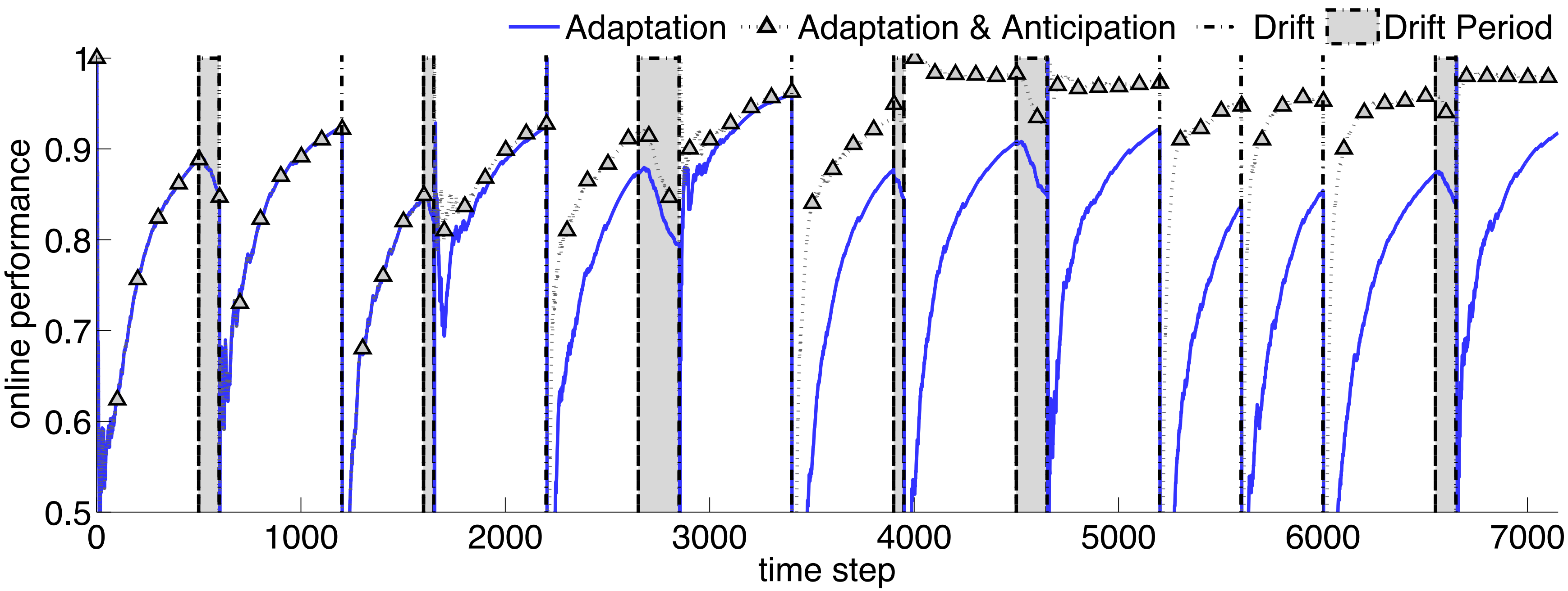
Ensemble methods



Ensemble methods



From adaptation to anticipation



The drifting concept is a 10D linear separator.

G. Jaber, A. Cornuéjols & Ph. Tarroux (2013) “*Anticipative and adaptive adaptation to concept changes*”. Submitted to IJCAI-2013.

Heuristical approaches to online learning: **assessment**

- Effective in situations with some kind of regularities in the change of environment
- Need to set various parameters
  - Window size
  - Nb of experts
  - …
- Rising interest

But a lack of solid theoretical foundations

A problem when **studying a new problem**

- Lack of agreed benchmark data bases
- Real data often difficult to get due to privacy or proprietary reasons

- Therefore forced to rely on controlled “artificial” data
  - Often cause for bad reviews in papers

The MOA plateform

- Massive Online Analysis (MOA)
  - http://moa.cms.waikato.ac.nz/
  - Set of implemented algorithms
    - Classification
    - Outlier detection
    - Online clustering
    - Frequent pattern mining
    - ...
  - MOA also provides:
    - **data generators** (e.g., AGRAWAL, Random Tree Generator, and SEA);
    - **evaluation methods** (e.g., periodic holdout, test-then-train, prequential);
    - and **statistics** (CPU time, RAM-hours, Kappa).
  - MOA can be used through a GUI (**Graphical User Interface**) or via command line, which facilitates running batches of tests.
    The implementation is in Java

Outline

1. Online learning: motivation, scenario, measure of performance
2. Theoretical framework: learning against any sequence
3. Heuristic approaches
4. Conclusion

Online learning: **conclusions**

- Against **any sequence**, *can we still say something?*
  - YES!!!
  - Guarantees with **similarities** with the in-distribution learning

- But a **too demanding** scenario
  - Several **types** of “realistic” concept shifts
  - The stability-plasticity **tradeoff**
  - And several type of **approaches**
    - **Detect** then relearn
    - **Adapt** continuously

General conclusions

- **Growing importance** of O.O.D. learning
  - Need for continual learning
  - Not retraining large models form scratch
  - Transfer learning when scarce target training data

- Many **heuristic** works
  - But lots of phenomena **not** completely **mastered** or **understood**
    - **Catastrophic** forgetting
    - Roles of the **source** and of the “**distance**” between tasks

- **Lack** of **theoretical** formalization and guarantees

- **Not much known** when the **source** and **target domains** are **different**