# Data Stream Mining

School of Computer Science

# Outline

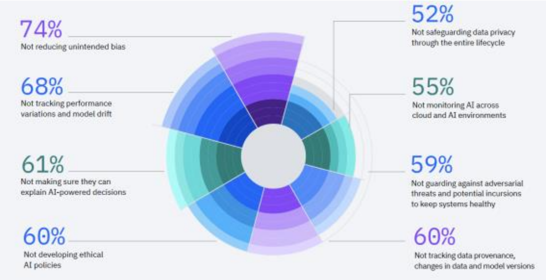
- An introduction to data stream mining
  - What is a data stream?
  - Data stream characteristics
  - Concept drift
- Learning algorithms for data streams
  - Main classifier types for drifting data streams
  - Ensemble learning from drifting data streams
- Limited access to ground truth in data streams
  - How to cope with sparse class labels?
  - Active learning under concept drift
  - Semi-supervised learning from drifting data streams
- Advanced problems
  - Recurrent Concepts
  - Open challenges and future directions

# New challenges for machine learning

Standard static and relatively small scenarios in machine learning and data mining do not reflect the current real-life problems we are facing.

We must deal with new data sources, generating high-speed, massive and heterogeneous data.

According to IDC Report in 2018 close to 5.8 zetabytes of data was generated.




IBM Global AI Adoption Index 2022

# What is a data stream?

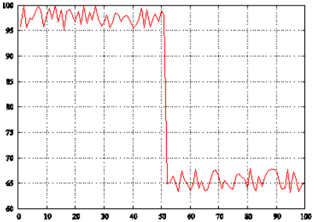
Data stream: an ordered, potentially unbounded sequence of instances which arrive continuously with time-varying intensity.

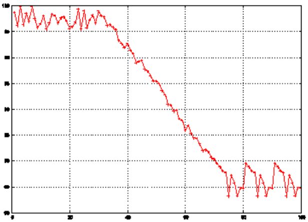
Given a model $\hat{f}_{t-1} : \mathcal{X} \longrightarrow \mathcal{Y}$ and a new chunk of examples $S_t = \{(\mathbf{x}_t^{(1)}, y_t^{(1)}), (\mathbf{x}_t^{(2)}, y_t^{(2)}), \ldots, (\mathbf{x}_t^{(n)}, y_t^{(n)})\} \subset S$, where $\forall i$, $(\mathbf{x}_t^{(i)}, y_t^{(i)}) \sim_{\text{i.i.d.}} \mathcal{D}_t$.

High-speed data streams: arising demands for fast-changing and continuously arriving data to be analyzed in real time.

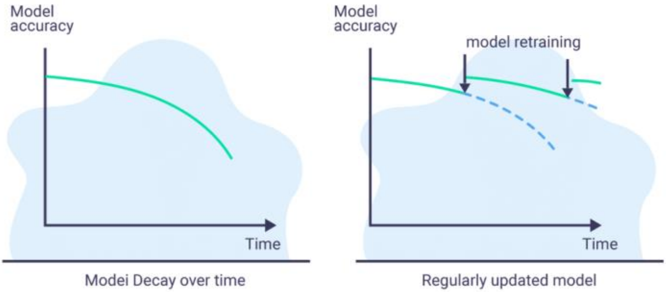
# Requirements for data stream algorithms

- Incremental processing
- Limited time:
  - Examples arrive rapidly
  - Each example can be processed only once
- Limited memory:
  - Streams are often too large to be processed as a whole
- Changes in data characteristics:
  - Data streams can evolve over time



# Model decay




Concept drift is one of the main reasons why we need to continue learning and adapting over time.

# Evaluating data stream algorithms

- Block / batch processing (data chunks)


- Online processing (instance after instance)

# Evaluating data stream algorithms

Standard metrics like accuracy, G-mean, Kappa etc. were designed for static problems.

One should use prequential metrics with forgetting, computed over most recent examples.

Prequential accuracy for standard problems and prequential AUC for binary and imbalanced streams.

$$acc^{(t)} = \begin{cases} acc_{ex}^{(t)}, & \text{if } t = 1 \\ \frac{(t-1)\cdot acc^{(t-1)} + acc_{ex}^{(t)}}{t}, & \text{otherwise} \end{cases}$$

where $acc_{ex}^{(t)}$ is the error (0 or 1) on the current training example $ex$ before its learning.

Additional metrics are crucial for evaluating streaming classifiers:

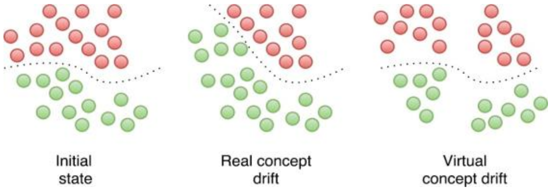
- Memory consumption
- Update time
- Classification

Interleaved Test-Then-Train or Prequential: Each individual example can be used to test the model before it is used for training, and from this the accuracy can be incrementally updated. When intentionally performed in this order, the model is always being tested on examples it has not seen.

# Concept drift

Concept drift can be defined as changes in distributions and definition of learned concepts over time.

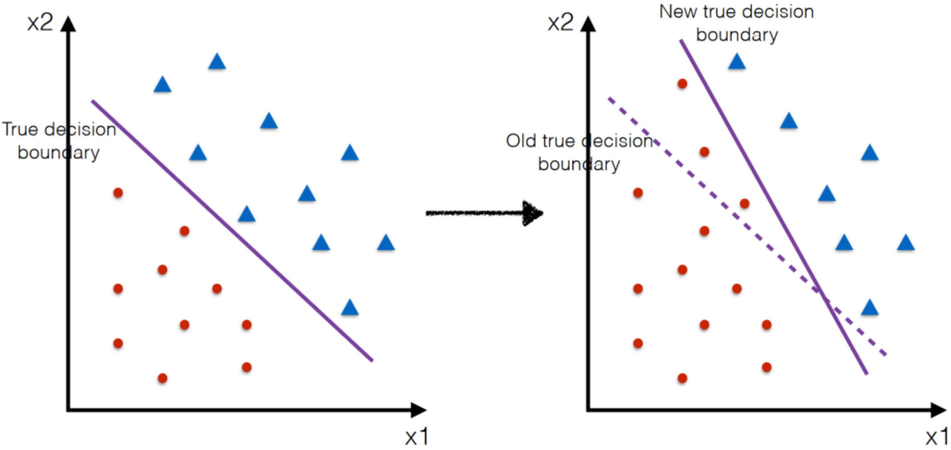
> Concept drift: change in the joint probability distribution of the problem. We say that there is concept drift in a data stream if, $\exists t, t+1 \mid \mathcal{D}_t \neq \mathcal{D}_{t+1}$.




Some real-life examples:

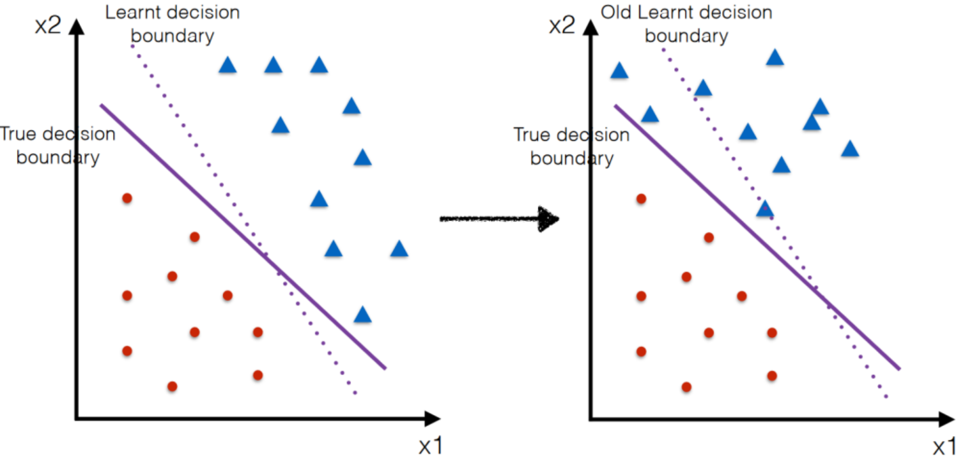
- changes of the user's interest in following news
- evolution of language used in text messages
- degradation or damage in networks of sensors

# Change in p(y|x)



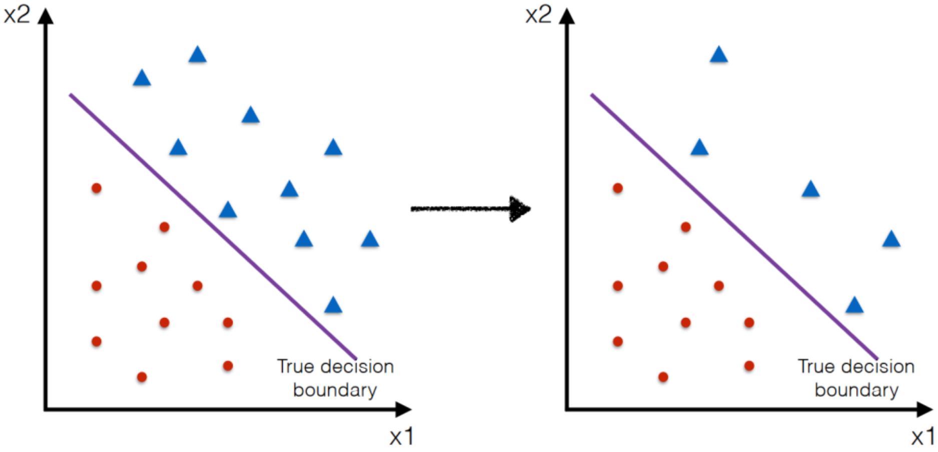

MINKU, L.; WANG, S.; BORACCHI, G. . "Learning Class Imbalanced Data Streams", *World Congress on Computational Intelligence (WCCI)*, July 2018

# Change in ?



x2
Learnt decision boundary
True decision boundary
x1
x2
Old Learnt decision boundary
True decision boundary
x1

# Change in ?



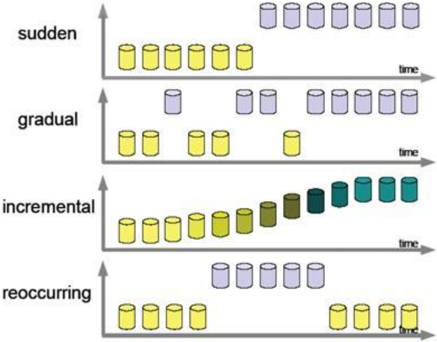
# Concept drift

Let us assume that our stream consist of a set of states $S = \{S_1, S_2, \cdot \cdot \cdot, S_n\}$, where $S_i$ is generated by a stationary distribution $D_i$.

By a stationary stream we can consider a transition $S_j \rightarrow S_{j+1}$, where $D_j = D_{j+1}$.

A non-stationary stream may have one or more of the following concept drift types:

- Sudden, where $S_j$ is suddenly replaced by $S_{j+1}$ and $D_j \neq D_{j+1}$
- Gradual, considered as a transition phase where examples in $S_{j+1}$ are generated by a mixture of $D_j$ and $D_{j+1}$
- Incremental, where rate of changes is much slower and $D_j \cap D_{j+1} \neq \emptyset$
- Reoccurring, where a concept from $k$-th previous iteration may reappear: $D_{j+1} = D_{j-k}$




One must not confuse concept drift with data noise.

We may also categorize concept drift according to its in uence on the probabilistic characteristics of the classi cation task:

- Virtual concept drift - changes do not impact the decision boundaries (posterior probabilities), but affect the conditional probability density functions
- Real concept drift - changes affect the decision boundaries (posterior probabilities) and may impact unconditional probability density function

# Handling concept drift

Three possible approaches to tackling drifting data streams:

- Rebuilding a classification model whenever new data becomes available (expensive, time-consuming, even impossible for rapidly evolving streams!)
- Detecting concept changes in new data (and rebuilding a classifier if these changes are sufficiently significant)
- Using an adaptive classifier (i.e. working in incremental or online mode)

We will discuss recurrent concept drift later!

# Handling concept drift

Algorithms for efficiently handling of concept drift presence can be categorized into four groups:

- Concept drift detectors
- Sliding window solutions
- Online learners
- Ensemble learners

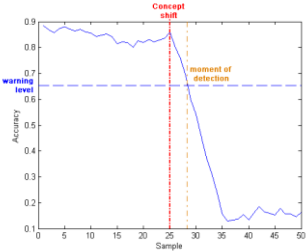
# Drift detectors

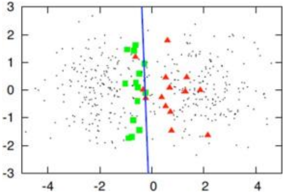
Algorithms that address the question of when drift occurs, being usually a separate tool from the actual classifier.

They aim at rising a signal when the change occurs. Some models also raise alarm when the chance of drift increases.

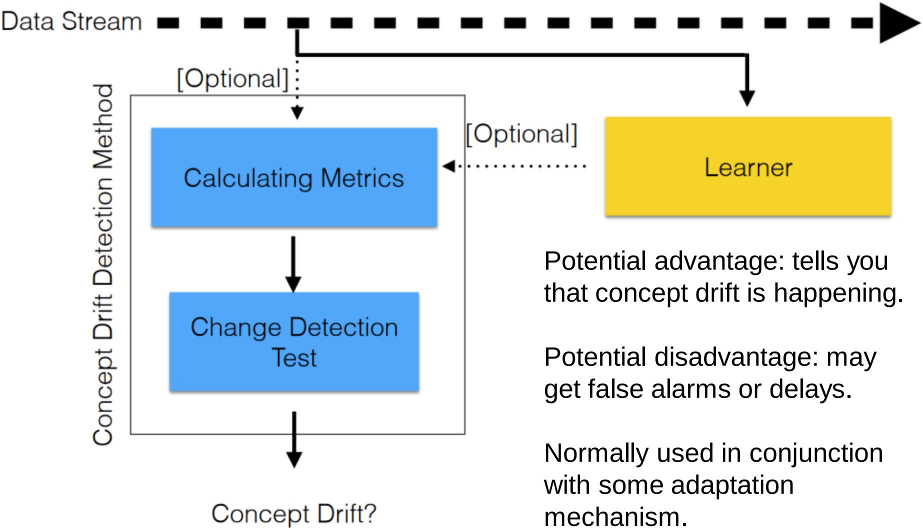
Three drift detector groups:

- Supervised.
  Use classification error or class distribution to detect changes - very expensive
- Semi-supervised.
  Use reduced number of important objects for detection - takes into account the cost of labeling
- Unsupervised.
  Based solely on properties of data - useful for detecting virtual drift, as real drift requires at least partial access to labels



# The General Idea of Concept Drift Detection



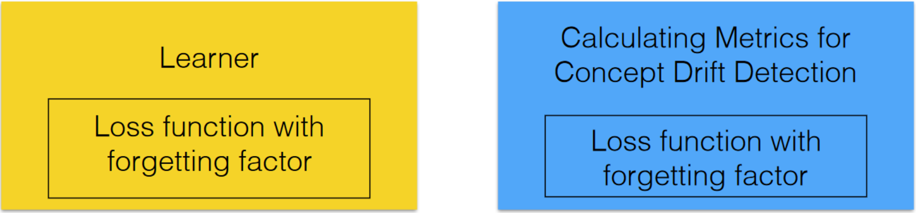

Potential advantage: tells you that concept drift is happening.

Potential disadvantage: may get false alarms or delays.

Normally used in conjunction with some adaptation mechanism.

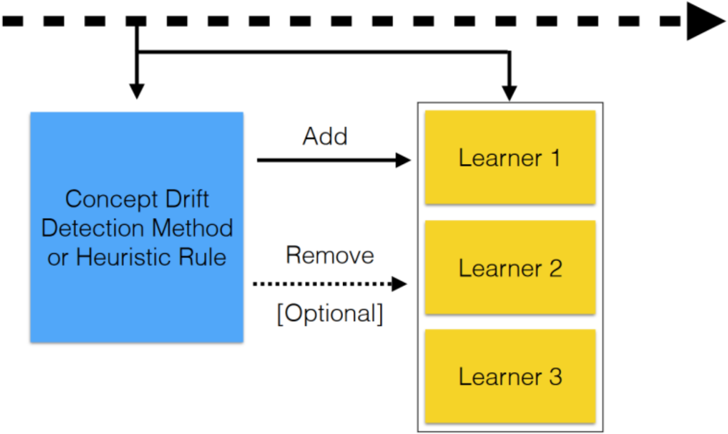
# The General Idea of Adaptation Mechanisms

Example of adaptation mechanism 1: forgetting factors



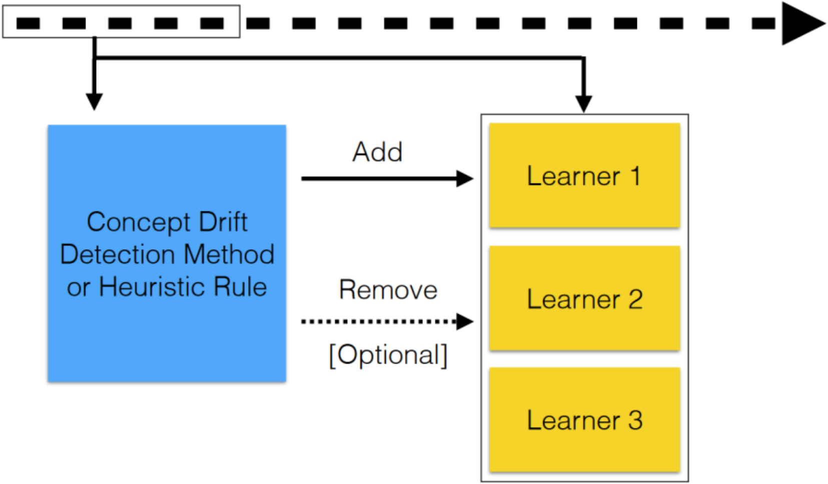
# The General Idea of Adaptation Mechanisms

Example of adaptation mechanism 2: adding / removing learners in online learning



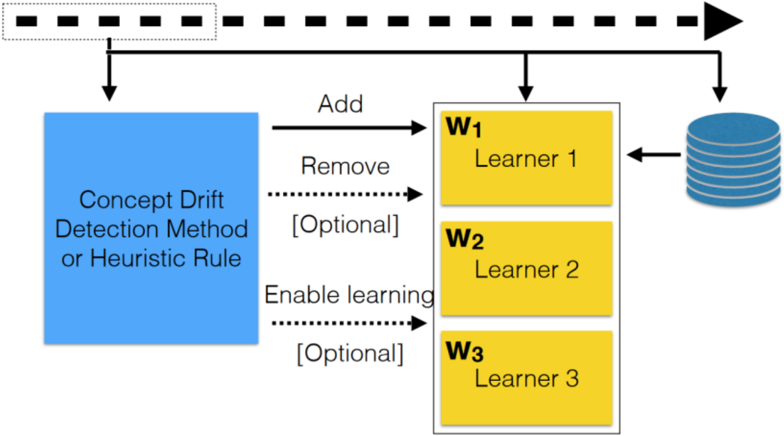
# The General Idea of Adaptation Mechanisms

Example of adaptation mechanism 3: adding / removing learners in chunk-based learning



# The General Idea of Adaptation Mechanisms

Other strategies / components are also possible



# Limited access to true class labels

Most of the works done in data streams assume that true class labels are available for each example or batch of objects immediately after processing.

This would however require extremely high labeling costs - which is far from being a realistic assumption.

We should assume either that we deal with labeling delay or we have a limited labeling budget.

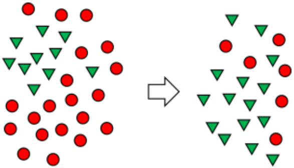
Active learning allows us to select samples to be labeled according to their value to drift detector and / or learner.

Active learning is especially challenging in the presence of concept drift, in order to rapidly adapt to changes.

# Learning from imbalanced data streams

The issue of class imbalance is becomes much more difficult in non-stationary streaming scenarios[1,2]:

- Imbalance ratio, as well as role of classes may evolve
- Class separation may change, as well as class structures
- We work with limited computational resources under time constraints
- Batch cases easier to handle, as one may handle chunks independently
- Online cases highly difficult due necessity of adapting to local changes



# Outline

- An introduction to data stream mining
  - What is a data stream?
  - Data stream characteristics
  - Concept drift
- Learning algorithms for data streams
  - Main classifier types for drifting data streams
  - Ensemble learning from drifting data streams
- Limited access to ground truth in data streams
  - How to cope with sparse class labels?
  - Active learning under concept drift
  - Semi-supervised learning from drifting data streams
- Advanced problems
  - Recurrent Concepts
  - Open challenges and future directions

# Three main families of classifiers for data streams

All classifiers for data stream mining can be categorized into three groups:

- Sliding window solutions
- Online learners
- Ensemble learners

# Sliding window

Assumption: recently arrived data are the most relevant - contain characteristics of the current context. However, their relevance diminishes with the passage of time.

There are two most popular strategies employed:

- Instance selection with a sliding window that cut offs older examples
- Instance weighting that assigns relevance level to each example present in the window

Size of the window has crucial impact. Shorter window - focus on the current concept, prone to local overfitting. Wider window- global outlook on the stream, may consist of instances from mixed concepts.

There is a number of proposals on applying windows with dynamic size or multiple windows at the same time.

# Online learners

Online learners for data streams must fulfill the following requirements:

- Each object must be processed only once in the course of training
- The system should consume only limited memory and processing time
- The training process can be paused at any time, and its accuracy should not be lower than that of a classi er trained on batch data collected up to the given time

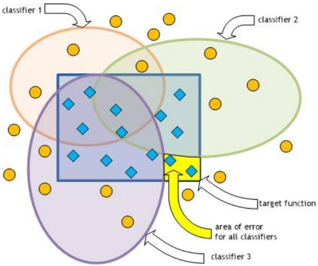
Some of standard classifiers like Naïve Bayes or Random Forest can work in online mode.

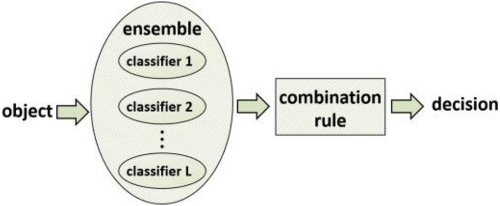
More sophisticated: Concept-adapting Very Fast Decision Trees, online Support Vector Machines, Mondrian Forests or weighted learners.

# Advantages of ensembles

Ensemble learning is a well-established area in static machine learning due to the following reasons:

- Classifiers combination can improve the performance of the best individual ones and it can exploit unique classifier strengths
- Avoiding the selection of the worst classifier
- Usually they offer more flexible decision boundary and at the same time they do not suffer from overfitting
- Can be simply and efficiently applied to distributed environments



# Ensembles for stream mining - taxonomy

Ensembles according to processing modes:

- Block ensembles
- Online ensembles

Ensembles according to their method for adapting to drifting streams:

- Dynamic combiners: base classifiers learned in advance, combination rule adapts to changes
- Ensemble updating: all / some base classifiers updated with incoming examples
- Dynamic ensemble line-up: new classifiers added for incoming data, weakest ones removed from the committee

# Dynamic combiners

Based on assumption that concept drift can be modeled as varying classifier combination scheme, e.g., with weights assigned to each classifier.

In order to work we require an efficient pool of initial classifiers with high diversity to capture different properties of the analyzed stream.

Classifier combination block is subject to identical limitations as standard classifiers in regard to time and memory consumption.

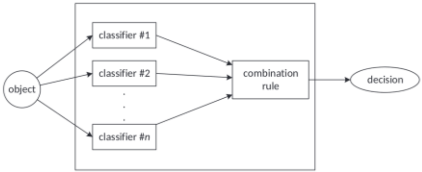



**Fig. 1.** A diagram of the classifier ensemble.

# Ensemble updating

This approach assumes that our ensemble consist of classifiers that can be updated in batch or online modes.

At the beginning we train a set of classifiers that will be continually adapted to the current state of the data stream.

This requires a diversity assurance method, usually realized as initial training on different examples (online Bagging) or different features (online Random Subspaces or adaptive Random Forest).

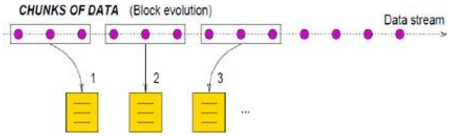
Additional diversity may be assured by using incoming examples to update only some of the classifiers in a random or guided manner.

# Dynamic ensemble line-up

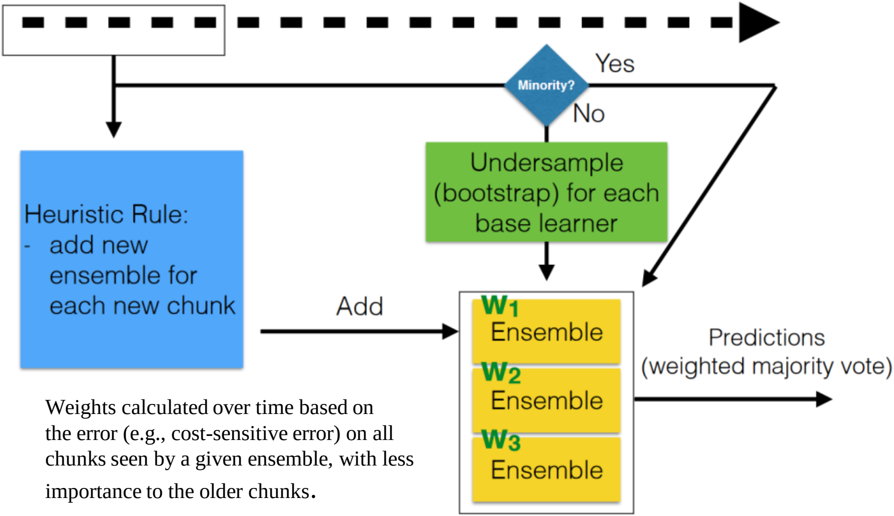
This approach assumes that we have a flexible ensemble set-up and add new classifiers for each incoming chunk of data.

Generic scheme:

- Train single initial classifier or $K$ initial classifiers (subject to training data availability)
- For each incoming chunk of data:
  - Train a new component classifier
  - Test other classifiers against the recent chunk
  - Assign weight to each classifier
  - Select top L classifiers (remove the weaker classifiers)



# Learn++.NIE: Learn++ for Nonstationary and Imbalanced Environments




Weights calculated over time based on the error (e.g., cost-sensitive error) on all chunks seen by a given ensemble, with less importance to the older chunks.

G. Ditzler and R. Polikar. "Incremental Learning of Concept Drift from Streaming Imbalanced Data", IEEE Transactions on Knowledge and Data Engineering, 25(10):2283-2301, 2013.

# Outline

- An introduction to data stream mining
  - What is a data stream?
  - Data stream characteristics
  - Concept drift
- Learning algorithms for data streams
  - Main classifier types for drifting data streams
  - Ensemble learning from drifting data streams
- Limited access to ground truth in data streams
  - How to cope with sparse class labels?
  - Active learning under concept drift
  - Semi-supervised learning from drifting data streams
- Advanced problems
  - Recurrent Concepts
  - Open challenges and future directions

# Access to true class labels

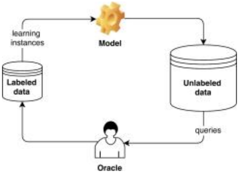
Most of the works done in data streams assume that true class labels are available for each example or batch of objects immediately after processing.

This would however require extremely high labeling costs - which is far from being a realistic assumption.

We should assume either that we deal with labeling delay or we have a limited labeling budget.

Active learning allows us to select samples to be labeled according to their value to drift detector and / or learner.

Access to labels is especially valuable when changes occur and thus active learning should be conducted in a more guided manner.



# Active learning for drifting data stream

Active learning assume that we have a realistic labeling budget at our disposal (e.g., 1%, 5%, 10% of instances etc.)

Uniform budget usage is not a good decision, as we should conserve it for the change moment.

Additionally, there are no techniques that allow for saving budget for novel class appearance - yet obtaining labeled instanced from new class is of crucial importance.

Furthermore, in imbalanced data streams we should be interested in getting as much labeled minority instances as possible - but how to predetermine if new instance is in fact a minority one?

# Semi-supervised learning for static and streaming data

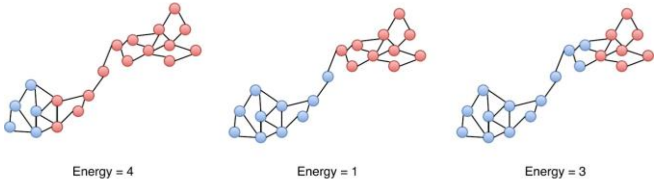
Semi-supervised learning assume that we have a small initial subset of labeled instances and large subset of unlabeled ones.

Labeled instances are used to guide the semi-supervised procedure in order to exploit efficiently the decision space.

Main characteristics of semi-supervised solutions are:

- confidence measure
- addition mechanism
- stopping criteria
- single or multiple learning models

Main approaches based on self-labeling, graph-based solutions and clustering.



# Semi-supervised learning for static and streaming data

Two types of methods dedicated to semi-supervised learning:

- transductive - do not generate a model for unseen data, aims at labeling instances
- inductive - train a classifier using unlabeled instances

Semi-supervised learning algorithms usually try to satisfy one of these three assumptions:

- smoothness assumption - if samples are close to each other in high density region, then they may share the same label
- cluster assumption - if samples can be grouped into separated clusters, then points in the same cluster are likely to be in the same class
- manifold assumption - high-dimensionality data can be effectively analyzed in lower dimensions

# Active learning framework

Our active learning is guided by a underlying classifier that selects most useful instances for labeling from unlabeled set $\mathscr{U}$:

$$q = \arg\max_{x \in \mathscr{U}} \Psi(h, x). \tag{1}$$

As we work with data streams, we formulate an incremental update of the underlying classification hypothesis under selected training algorithm $A$ and $i$-th iteration:

$$h_{i+1} = A\left(\{q_k, \mathrm{o}(q_k)\}_{k=1}^{i}\right), \tag{2}$$

where

$$q_i = \arg\max_{x \in \mathscr{U}_i} \Psi(h_i, x), \tag{3}$$

$$\mathscr{U}_{i+1} = \mathscr{U}_i \setminus \{q_i\}. \tag{4}$$

Thus classifier in our active learning scenario adapts over time based on previous experience:

$$q_i = \arg\max_{x \in \mathscr{U}_i} \Psi_i(h_i, x), \tag{5}$$

# Ensemble active learning

Conduct an active learning process using an ensemble of $L$ classifiers[1]:

$$\Pi = \{\Psi_1, \cdots, \Psi_L\}, \tag{6}$$

This allows for a more robust instance selection for label query.

Instead of pooling their decision using voting strategies (like in Query by Committee), we propose to select a classifier responsible for a given instance query.

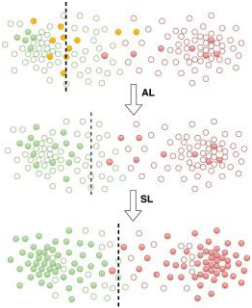
This allows to better utilize a pool of diverse classifiers and select one that can anticipate the direction of changes better than remaining ones.

The idea behind this is similar to dynamic classifier selection - exploiting individual classifier's competencies.

[1] Bartosz Krawczyk, Alberto Cano: Adaptive Ensemble Active Learning for Drifting Data Stream Mining. IJCAI 2019: 2763-2771

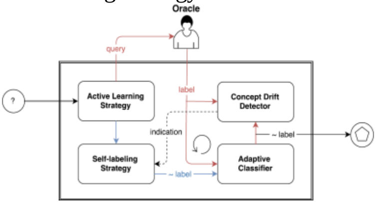
# Motivation: Limited Access to Ground Truth

- Active learning allows for an informative selection of instances that will be most useful for adjusting the classifier to the current state of the stream. However, each such query reduces the available budget.
- Self-labeling allows to exploit discovered data structures and improve the competency of a classifier at no cost, yet offers no quality validation.
- These procedures are complimentary - active learning can be interpreted as an exploration step and semi-supervised learning as an exploitation step.



# Hybrid framework for drifting data stream mining on a budget

- We developed a hybrid framework that uses active learning for creating a meaningful input for self-labeling strategy[2].



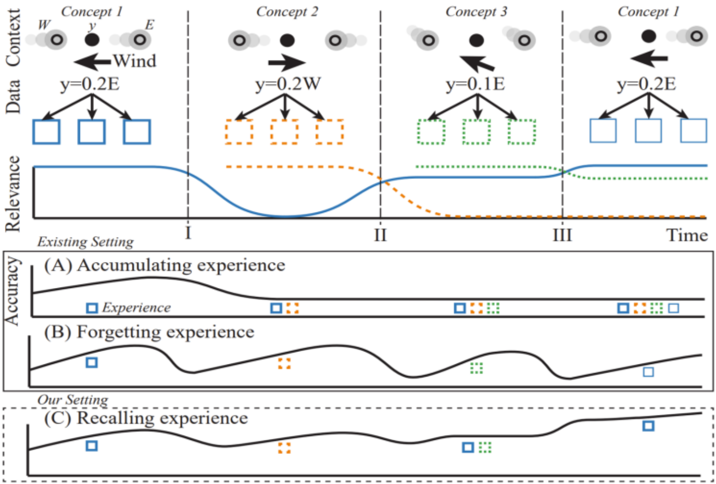

- Drifting data self-labeling were proposed, divided into two groups:
  - blind self-labeling strategies relied on adaptation of uncertainty threshold in a similar manner to their active learning counterparts.
  - informed self-labeling strategies utilized input from the drift detector to adapt their actions depending on the current state of the stream.

[2]Lukasz Korycki, Bartosz Krawczyk: Combining Active Learning and Self-Labeling for Data Stream Mining. CORES 2017: 481-490

# Outline

- An introduction to data stream mining
  - What is a data stream?
  - Data stream characteristics
  - Concept drift
- Learning algorithms for data streams
  - Main classifier types for drifting data streams
  - Ensemble learning from drifting data streams
- Limited access to ground truth in data streams
  - How to cope with sparse class labels?
  - Active learning under concept drift
  - Semi-supervised learning from drifting data streams
- Advanced problems
  - Recurrent Concepts
  - Open challenges and future directions

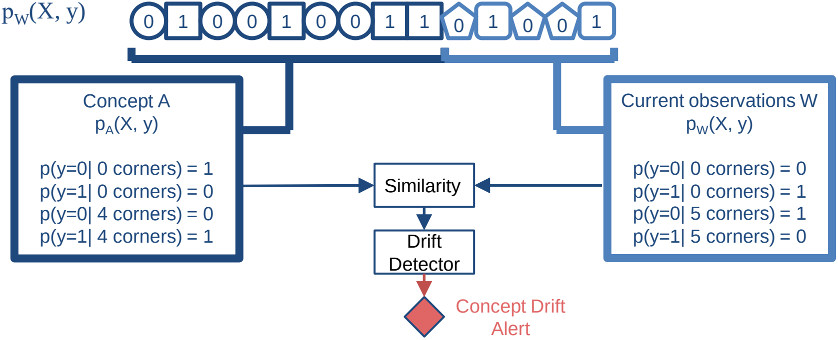
# Relevance Experience




B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

# Concept Drift Detection

Methods for detecting a difference in the distribution of current observations: $p_A(X, y) \neq p_W(X, y)$



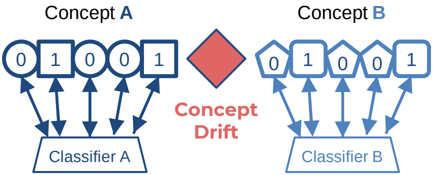

B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

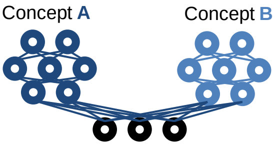
# Forgetting experience

Previous observations describe the old distribution and may conflict with the new concept.

Conflict: $p_A(y|X) \neq p_W(y|X)$

The forgetting function modifies the active classifier so it does not use experience irrelevant to the new concept.

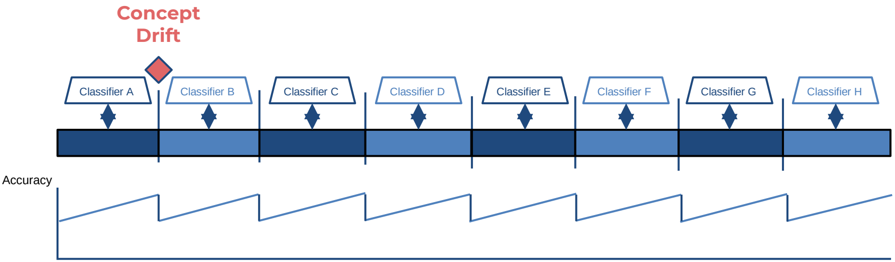



B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

# Adaptive Learning

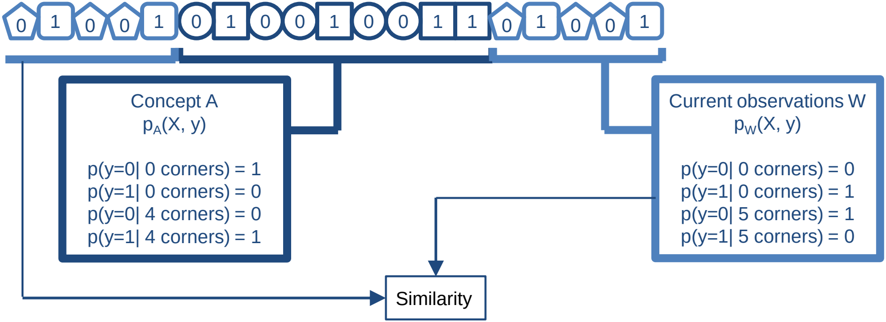
Forgetting avoids experience conflict, but may lead to catastrophic forgetting




B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

# Concept Re-identification

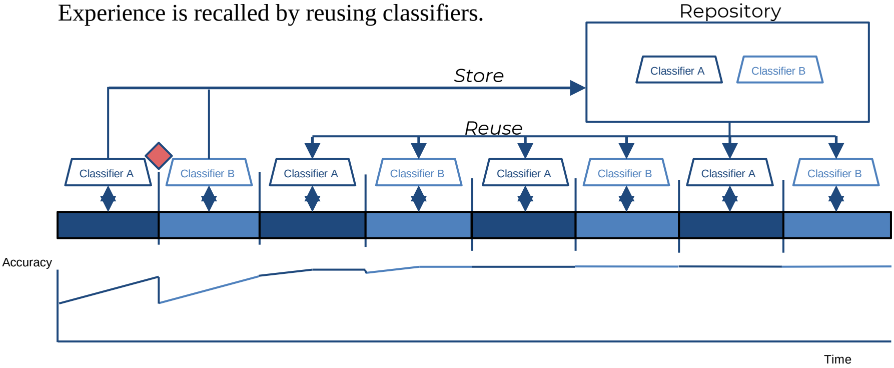
Methods for detecting a similarity in the distribution of previous observations: $p_i(X, y) \sim p_W(X, y)$




B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

# Recalling experience - reuse

A repository is used to store and query trained classifiers.
Experience is recalled by reusing classifiers.



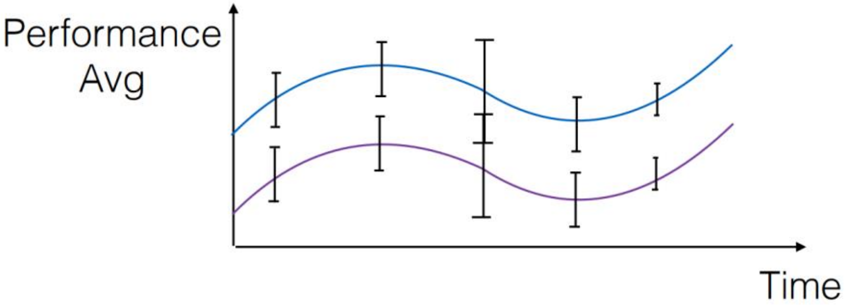

B. Halstead, Y. S. Koh, P. Riddle, M. Pechenizkiy, A. Bifet and R. Pears, "Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information," 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 2021, pp. 1056-1067, doi: 10.1109/ICDE51399.2021.00096.

# Open challenges and future directions

- Interpretability vs accuracy of drifting data streams: explaining the concept drift
  - Explainable artificial intelligence (XAI) for non-stationary data
  - Understanding what and why changed and how can we use this knowledge to improve adaptation
- Learning for extremely sparsely labeled data streams
  - Learning from data streams without any access to class labels
  - Merging unsupervised methods with supervised predictors
- Multi-view asynchronous data streams
  - Transferring useful information among multiple data streams
  - Using different views on data streams to extract more information-rich representation and better detect drifts
- Robustness to adversarial attacks
  - "Fake" and malicious concept drifts
  - Appearance of artificial classes to increase the class imbalance and learning difficulty

# Are the Performances of Different Approaches Really Different?

Specially when our predictors are stochastic...
Standard deviations throughout the learning.




J. Demsar. Statistical comparisons of classifiers over multiple data sets. JMLR, 7:1–30, 2006.

# Resources

Tutorial
https://riverml.xyz/
https://streamlearningtutorial2020.netlify.app/

Part of these are from:
MINKU, L.; WANG, S.; BORACCHI, G. . "Learning Class Imbalanced Data Streams", *World Congress on Computational Intelligence (WCCI)*, July 2018

http://bigdataieee.org/BigData2020/files/IEEE_BigData_2020_Tutorial7.pdf