Tutorial on Learning Class Imbalanced Data Streams

Leandro L. Minku
University of Leicester

Shuo Wang
University of Birmingham

Giacomo Boracchi
Politecnico di Milano

Outline

- Background and motivation
- Problem formulation
- Challenges and brief overview of core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges

S. Wang, L. Minku, X. Yao. “A Systematic Study of Online Class Imbalance Learning with Concept Drift”, *IEEE Transactions on Neural Networks and Learning Systems*, 2017 (in press).

B. Krawczyk, L. Minku, J. Gama, J. Stefanowski, M. Wozniak. “Ensemble Learning for Data Stream Analysis: a survey”, *Information Fusion*, 37, 132-156, 2017.

G. Ditzler, M. Roveri, C. Alippi, R. Polikar. “Learning in Nonstationary Environments: A survey”, *IEEE Computational Intelligence Magazine,* 10 (4), 12-25, 2015.

J Gama, I Žliobaitė, A Bifet, M Pechenizkiy, A Bouchachia . “A Survey on Concept Drift Adaptation”, *ACM Computing Surveys*, 46 (4), 2014.

Outline

- Background and motivation
- Problem formulation
- Challenges and brief overview of core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges

Data Streams

- Organisations have been gathering large amounts of data.
- The amount of data frequently grows over time.

Labelled Data Streams

Labelled data stream: ordered and potentially infinite sequence of examples $S = <(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2), \ldots, (\mathbf{x}_t,y_t), \ldots>$ where $(\mathbf{x}_t, y_t) \in \mathcal{X} \times \mathcal{Y}$

Classification problems:

- $\mathcal{X}$ is a (typically high dimensional) space.
  - Real values.
  - Ordinal values.
  - Categorical values.
- $\mathcal{Y}$ is a set of categories.

This tutorial will concentrate on supervised learning for classification problems.

Example of Data Stream: Tweet Topic Classification

Labelled data stream: ordered and potentially infinite sequence of examples $S = <(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2), \ldots, (\mathbf{x}_t,y_t), \ldots>$ where $(\mathbf{x}_t, y_t) \in \mathcal{X} \times \mathcal{Y}$

Each training example corresponds to a tweet.

x = {td-idf word 1,
td-idf word 2,
etc}

y = {topic}

td-idf (term frequency - inverse document frequency)

Example of Data Stream: Fraud Detection

Labelled data stream: ordered and potentially infinite sequence of examples $S = <(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2), \ldots, (\mathbf{x}_t,y_t), \ldots>$ where $(\mathbf{x}_t, y_t) \in \mathcal{X} \times \mathcal{Y}$

Each training example corresponds to a credit or debit card transaction.

$\mathbf{x}$ = {merchant id,
purchase amount,
average expenditure,
average number of transactions
per day or on the same shop,
average transactions amount,
location of last purchase,
etc}

y = {genuine/ fraud}

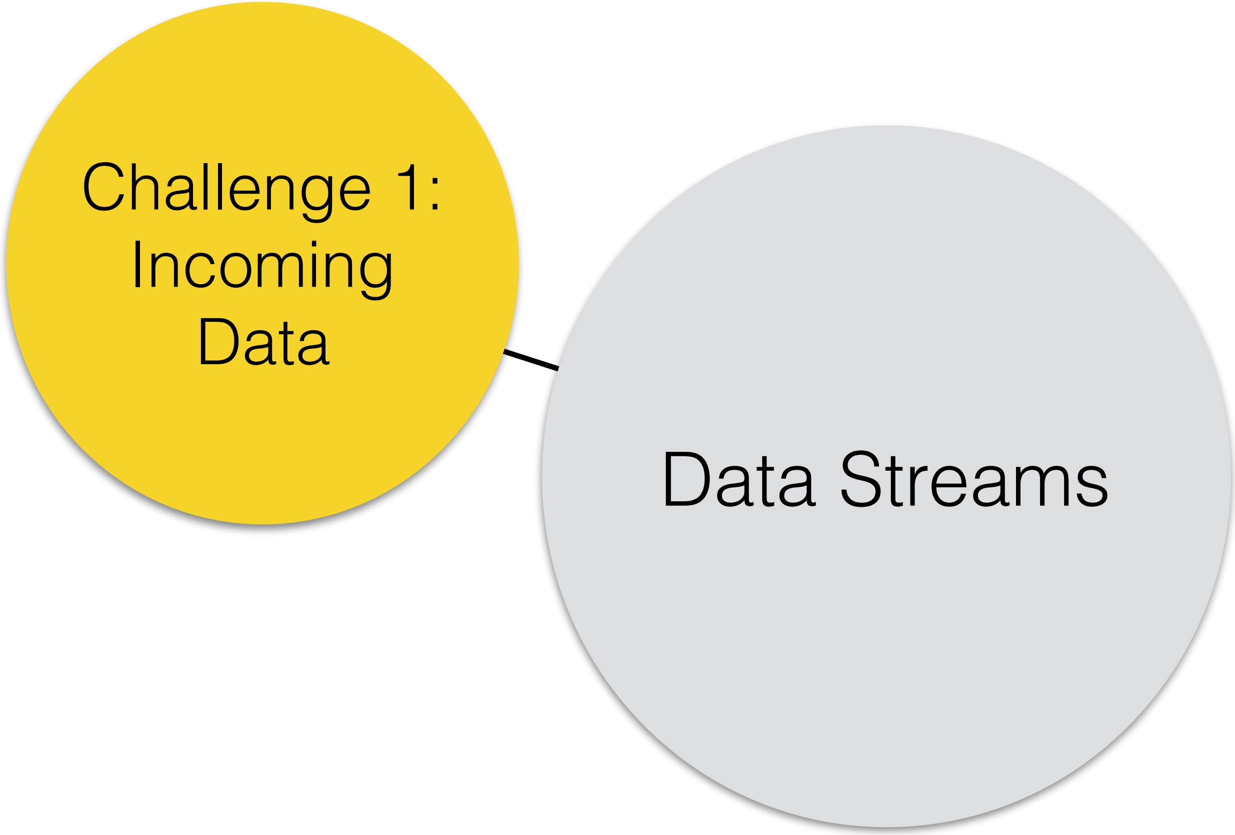
Example of Data Stream: Software Defect Prediction

Labelled data stream: ordered and potentially infinite sequence of examples $S = <(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2), \ldots, (\mathbf{x}_t,y_t), \ldots>$ where $(\mathbf{x}_t, y_t) \in \mathcal{X} \times \mathcal{Y}$

Each training example corresponds to a software module.

$\mathbf{x}$ = {lines of code,
branch count,
halstead complexity,
etc}

y = {buggy/not buggy}


Challenge 1: Incoming Data
Data Streams

Challenge 1: Incoming Data

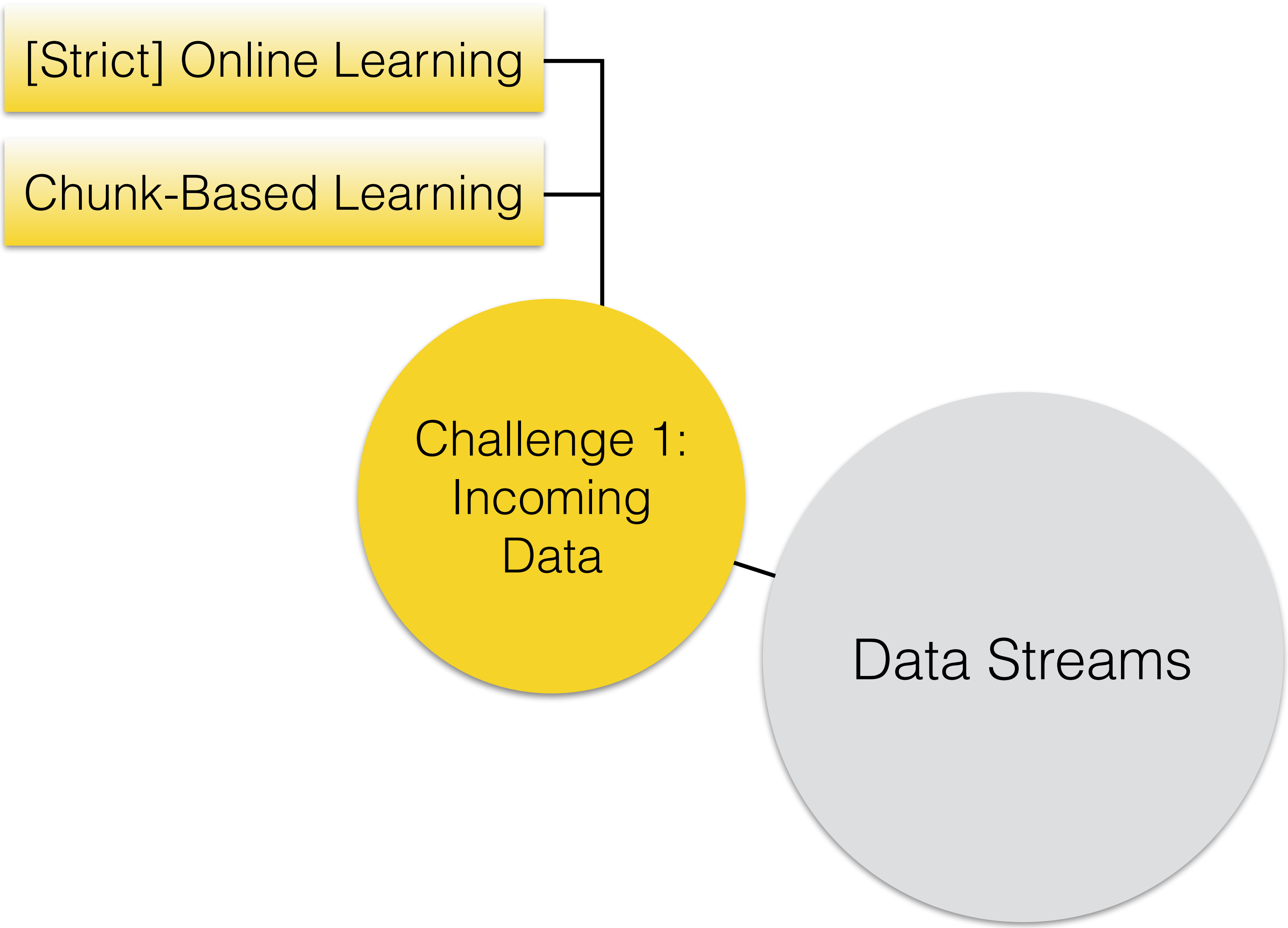
- Data streams are ordered and potentially infinite.
- In the beginning there may be few examples.
- Rate of incoming examples is typically high.

Everyday, on average, around 600,000 credit / debit card transactions are processed by Atos Worldline.

Challenge 1:
Incoming Data

- Data streams are ordered and potentially infinite.
- In the beginning there may be few examples.
- Rate of incoming examples is typically high.

Every second, on average, around 6,000 tweets are tweeted on Twitter (http://www.internetlivestats.com/twitter-statistics/)



[Strict] Online Learning

Chunk-Based Learning

Challenge 1: Incoming Data

Data Streams

Core Techniques: Online Supervised Learning

- Given a model $\hat{f}_{t-1}$: $\mathcal{X}$ —> $\mathcal{Y}$ and a new example ($\mathbf{x}_t$, y_t) drawn from a joint probability distribution $\mathcal{D}_t$.
- Learn a model $\hat{f}_t$: $\mathcal{X}$ —> $\mathcal{Y}$ able to generalise to unseen examples of $\mathcal{D}_t$.
- In *strict* online learning,($\mathbf{x}_t$, y_t) must be discarded soon after being learnt.

- Potential advantages: fast training, low memory requirements.
- Potential disadvantage: only one pass may result in lower predictive performance.

Core Techniques: Chunk-Based Supervised Learning

- 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 $\forall i$, $(\mathbf{x}_t^{(i)}, y_t^{(i)}) \sim_{\text{i.i.d.}} \mathcal{D}_t$.

- Learn a model $\hat{f}_t : \mathcal{X} \longrightarrow \mathcal{Y}$ able to generalise to unseen examples of $\mathcal{D}_t$.

- Typically, S_t is discarded soon after being learnt.

- Potential advantages: easy to fit in any offline learning technique, processing examples from each chunk several times may help to increase predictive performance.

- Potential disadvantages: higher training time; requires to store chunk; in reality, examples from S_t may not all come i.i.d. from $\mathcal{D}_t$.

Applying Online vs Chunk-Based Learning

Some applications lend themselves to a specific type of algorithm.

New training examples arrive separately.

New training examples arrive in chunks.

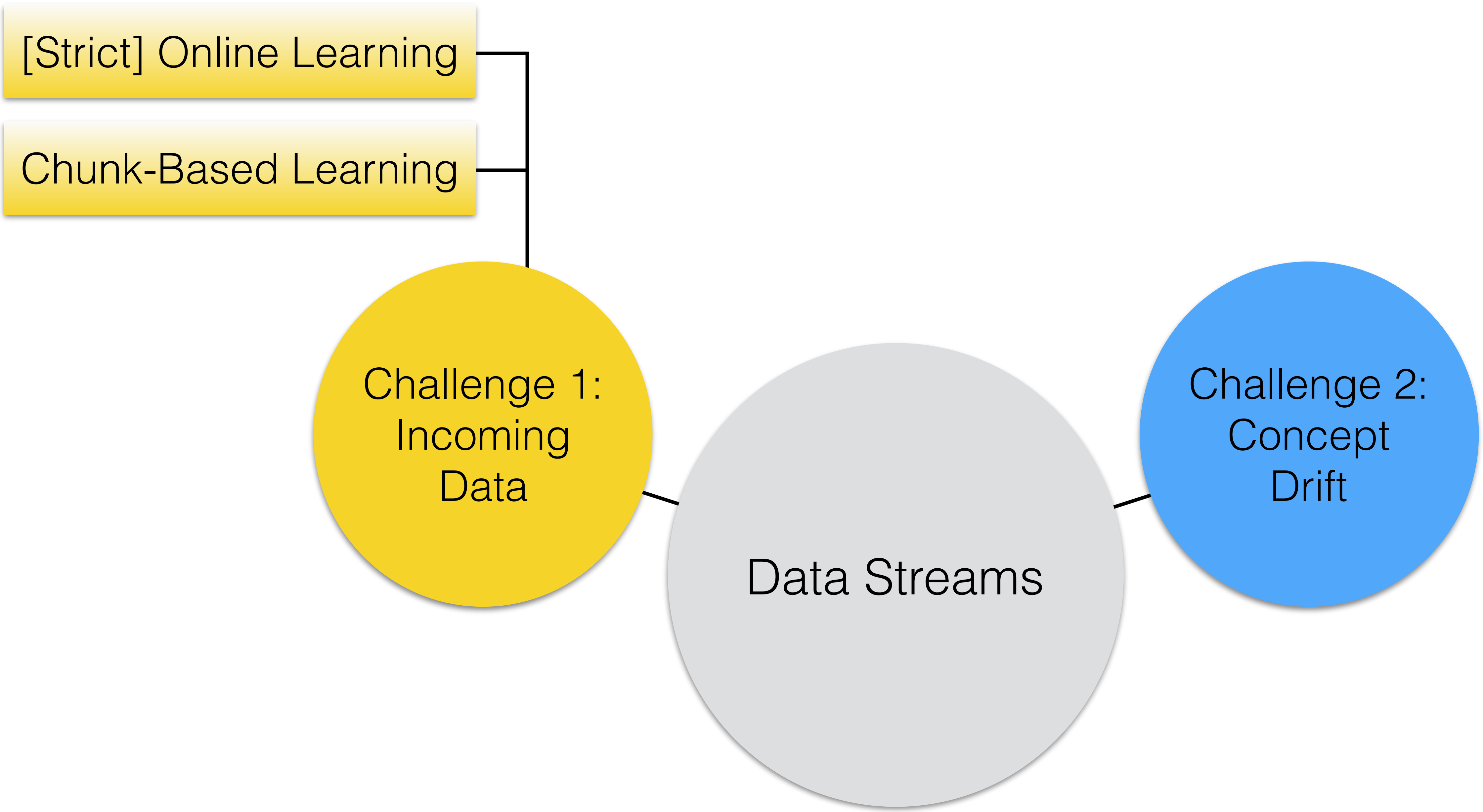
Applying Online vs Chunk-Based Learning

Online learning for applications where data arrive in chunks.

- Process each training example of the chunk separately.

Chunk-based learning for applications where data arrives separately.

- Wait to receive a whole chunk of data.



[Strict] Online Learning

Chunk-Based Learning

Challenge 1: Incoming Data

Data Streams

Challenge 2: Concept Drift

Challenge 2: Concept Drift

- The probability distributions $\mathcal{D}_t$ and $\mathcal{D}_{t+1}$ are not necessarily the same.

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

- In chunk-based learning, $\forall i$, $(\mathbf{x}_t^{(i)}, y_t^{(i)}) \in S_t$ are typically assumed to be drawn i.i.d. from the same $\mathcal{D}_t$. However, in reality, it may be difficult to guarantee that.

- Non-stationary environments: environments that may suffer concept drift.

Types of Concept Drift

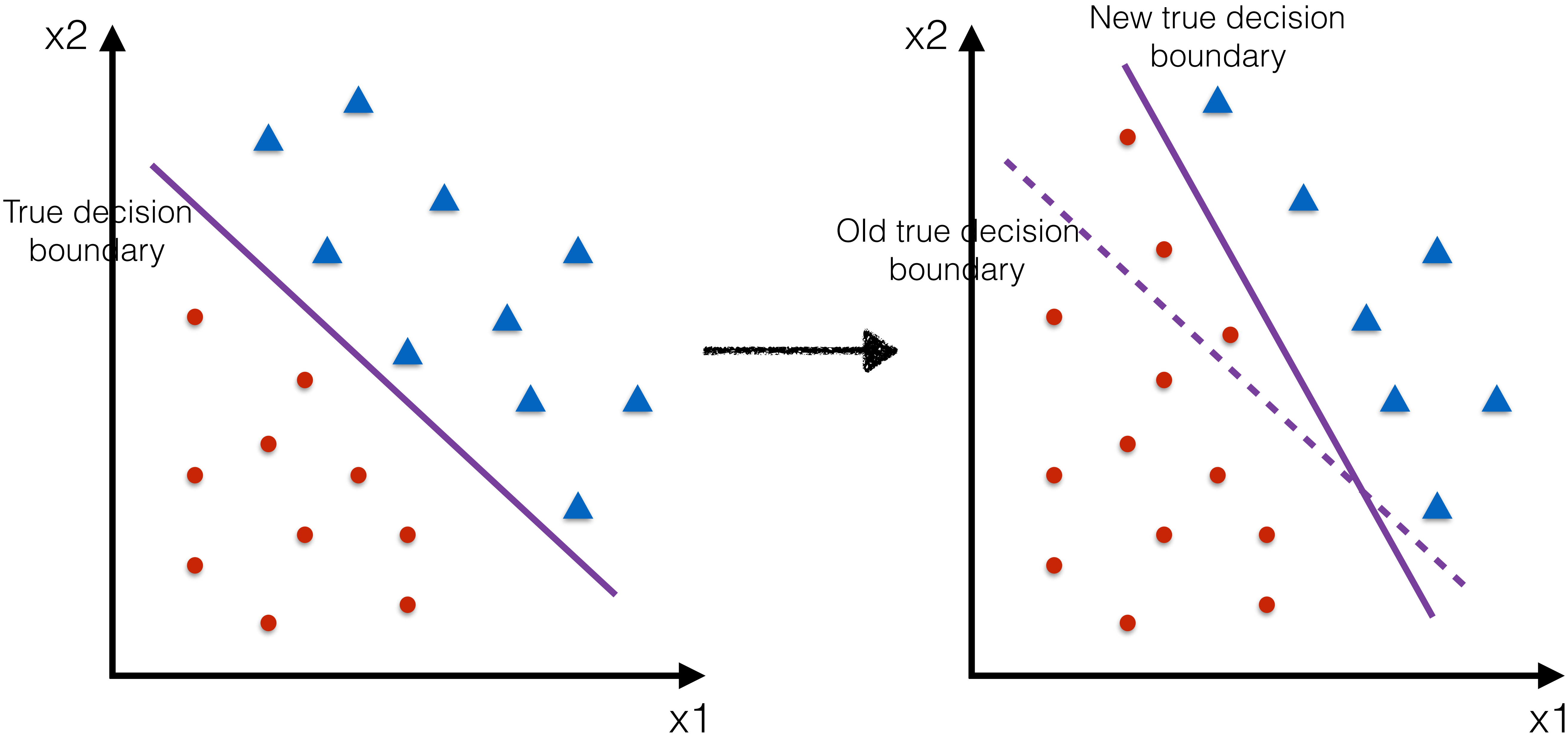
$$\mathcal{D}_t = p_t(\mathbf{x},y) = p_t(y|\mathbf{x})\ p_t(\mathbf{x})$$

$$\mathcal{D}_t = p_t(\mathbf{x},y) = p_t(\mathbf{x}|y)\ p_t(y)$$

Concept drift may affect different components of the joint probability distribution.

Potentially, more than one component can be affected concurrently.

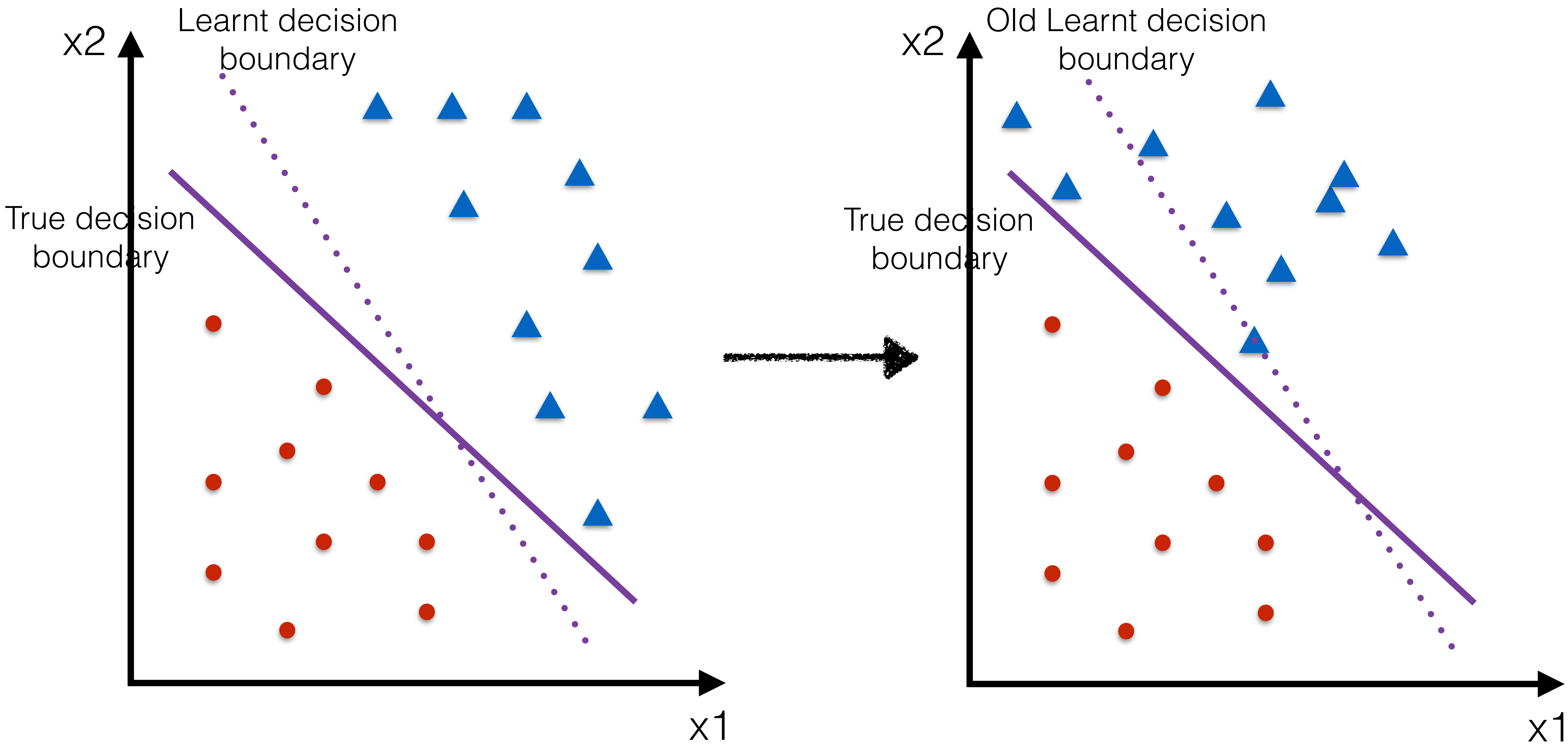
Change in $p(y|\mathbf{x})$



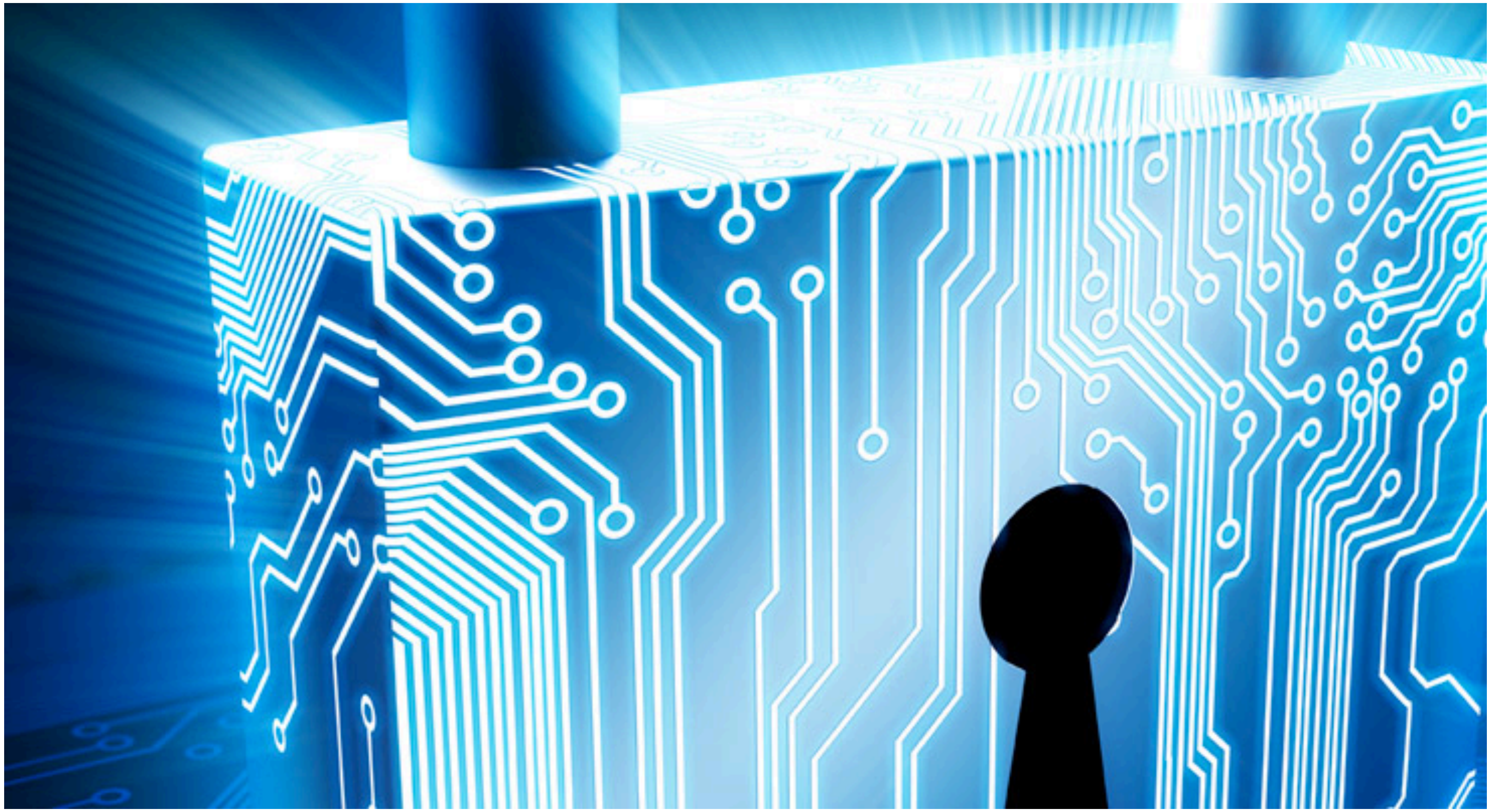
Change in p(y|**x**)

E.g.: a software module that was previously buggy may not be buggy in the new version of the software anymore, despite having similar input features.

Change in p(**x**)

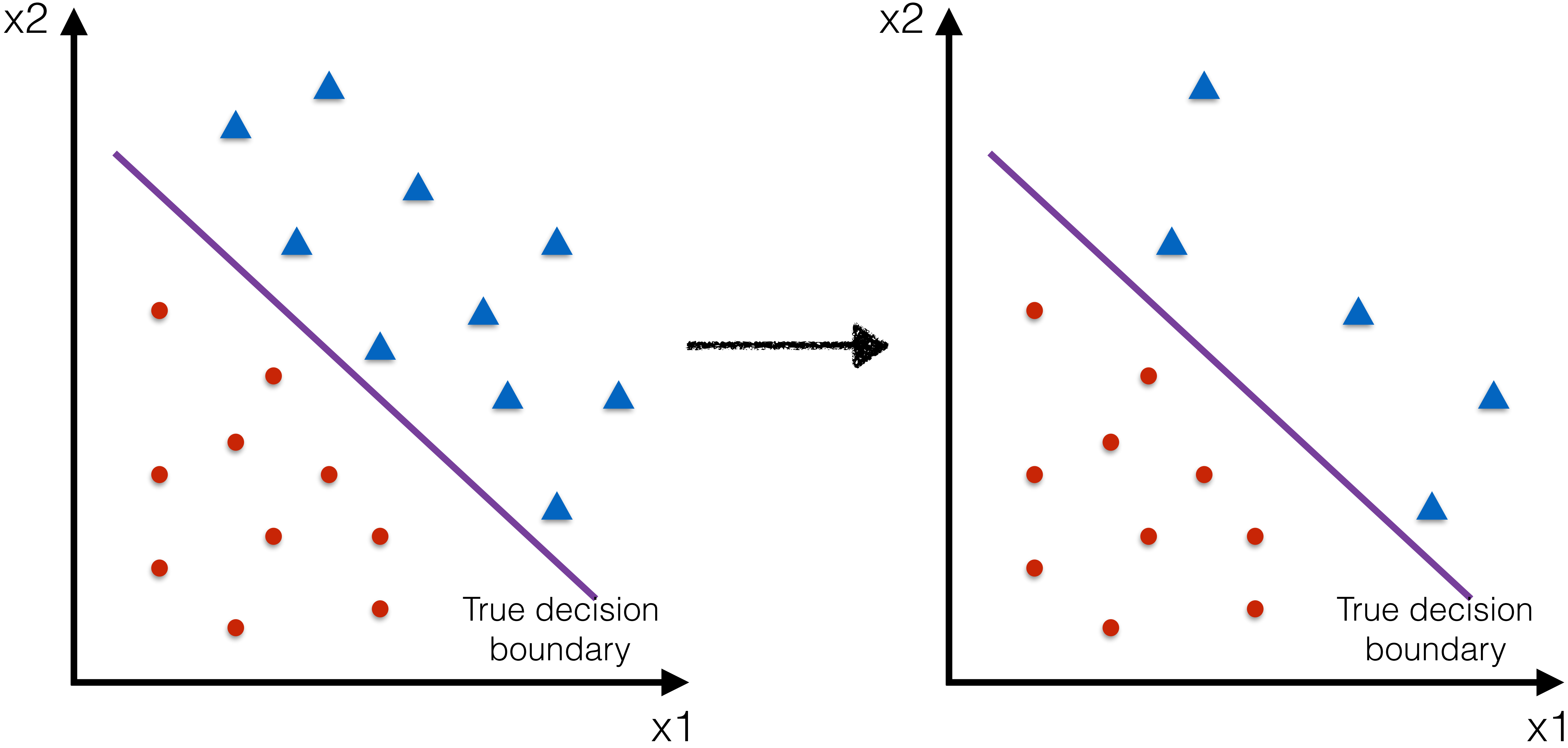

x2
Learnt decision boundary
True decision boundary
x1
x2
Old Learnt decision boundary
True decision boundary
x1

Example of Potential Change in p(**x**)



E.g.: fraud strategies and customer habits may change.

Change in p(y)



Example of Change in p(y)

FIFA Confederations Cup

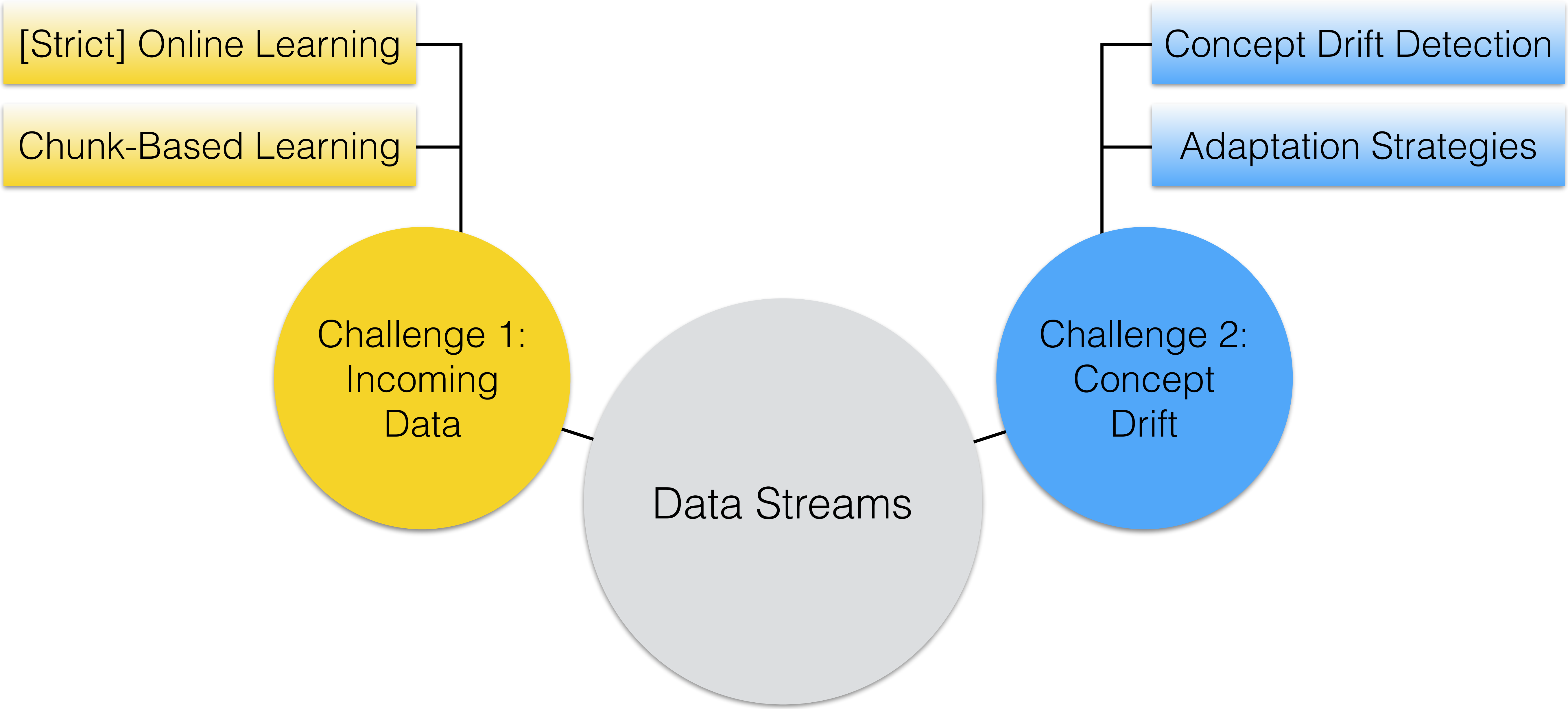
→

FIFA World Cup

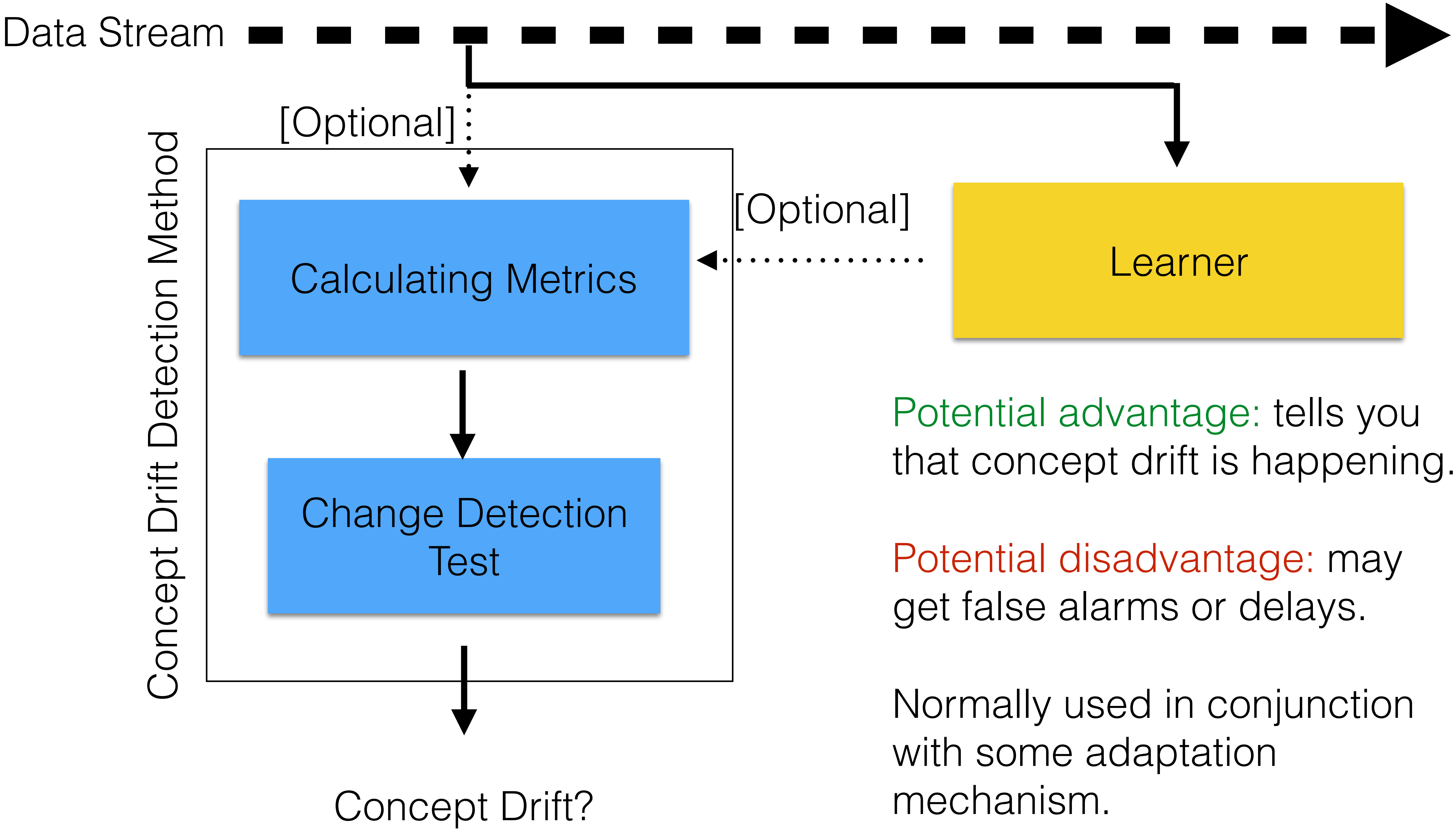
E.g., tweet topic becoming more or less popular.

Concept Drift and the Need for Adaptation

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


[Strict] Online Learning
Chunk-Based Learning
Challenge 1: Incoming Data
Data Streams
Challenge 2: Concept Drift
Concept Drift Detection
Adaptation Strategies

Core Techniques:
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.

Core Techniques: The General Idea of Adaptation Mechanisms

- Adaptation mechanisms may or may not be used together with concept drift detection methods, depending on how they are designed.

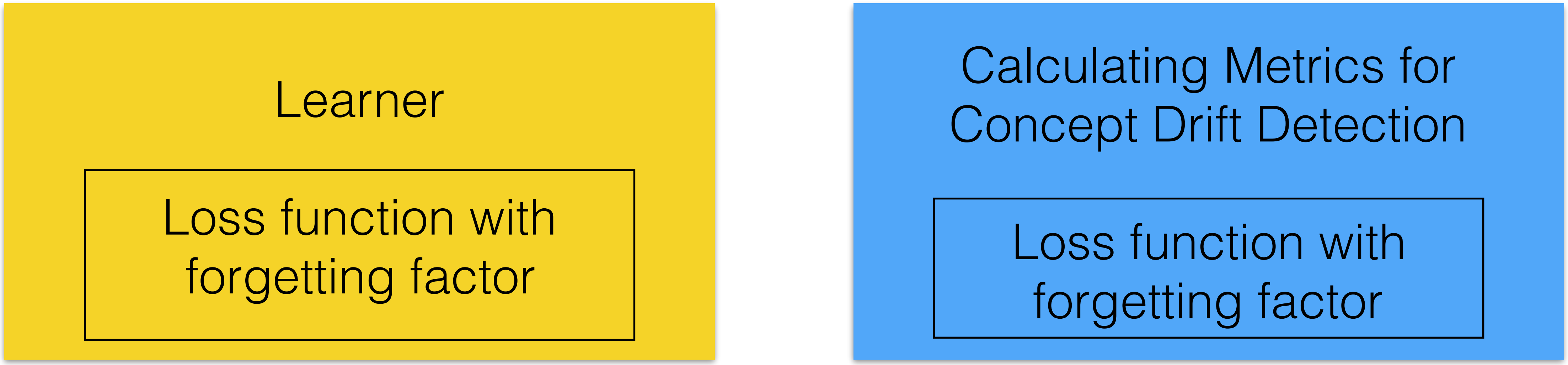
- Potential advantages of not using concept drift detection: no false alarms and delays, potentially more adequate for slow concept drifts.

- Potential disadvantage of not using concept drift detection: don’t inform users of whether concept drift is occurring.

- Several different adaptation mechanisms can be used together.

Core Techniques: The General Idea of Adaptation Mechanisms

Example of adaptation mechanism 1: forgetting factors



Learner

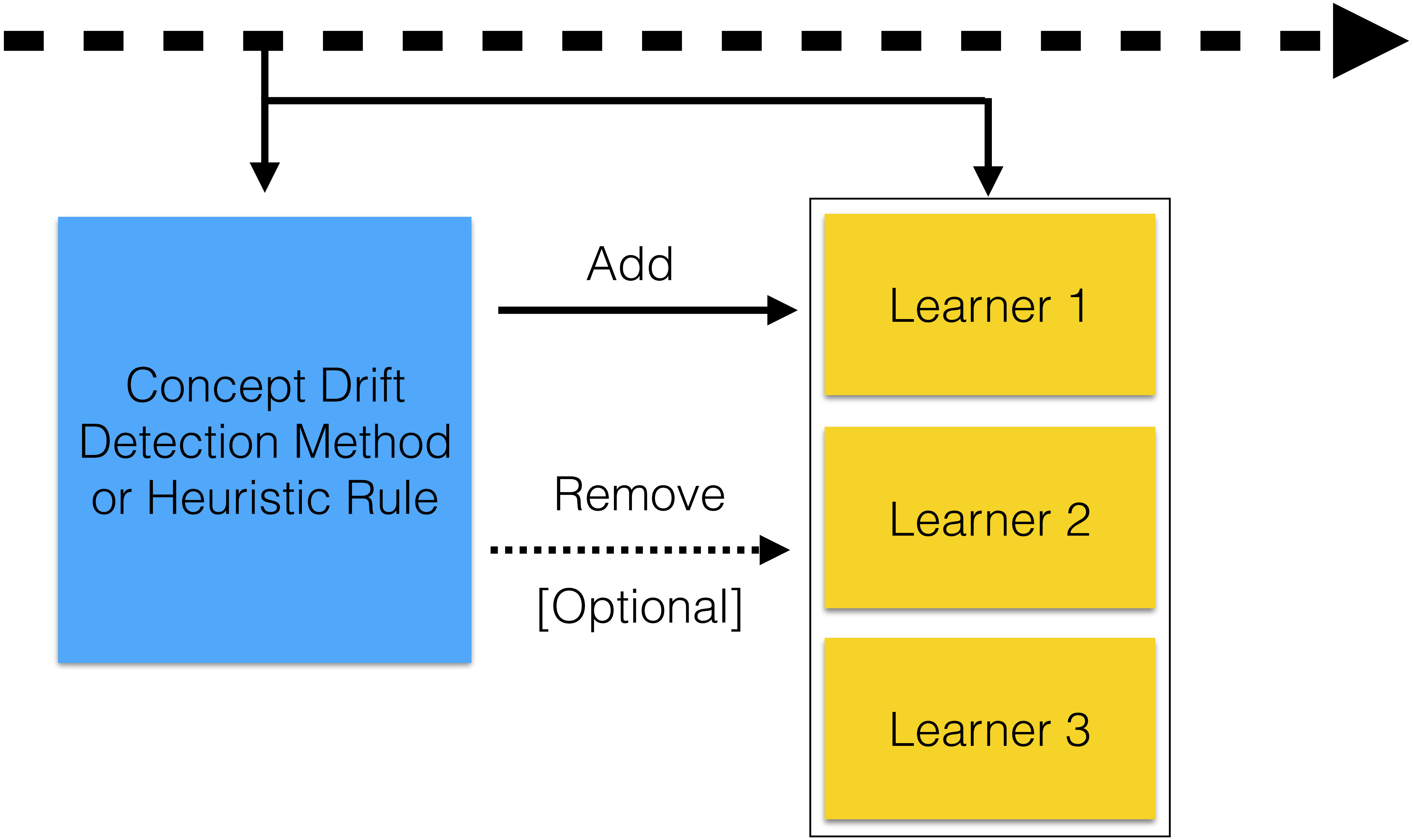
Loss function with forgetting factor

Calculating Metrics for Concept Drift Detection

Loss function with forgetting factor

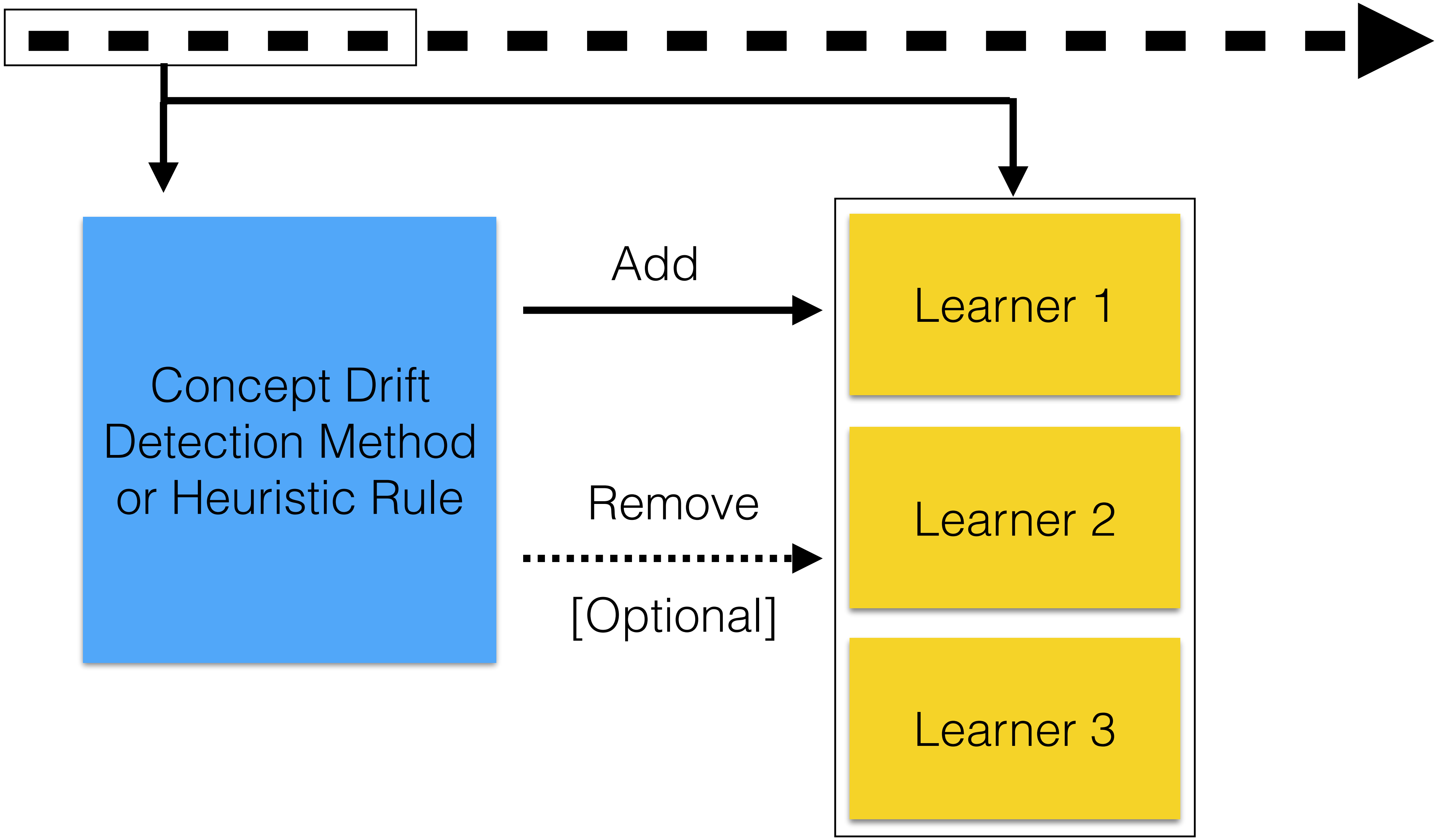
Core Techniques:
The General Idea of Adaptation Mechanisms

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



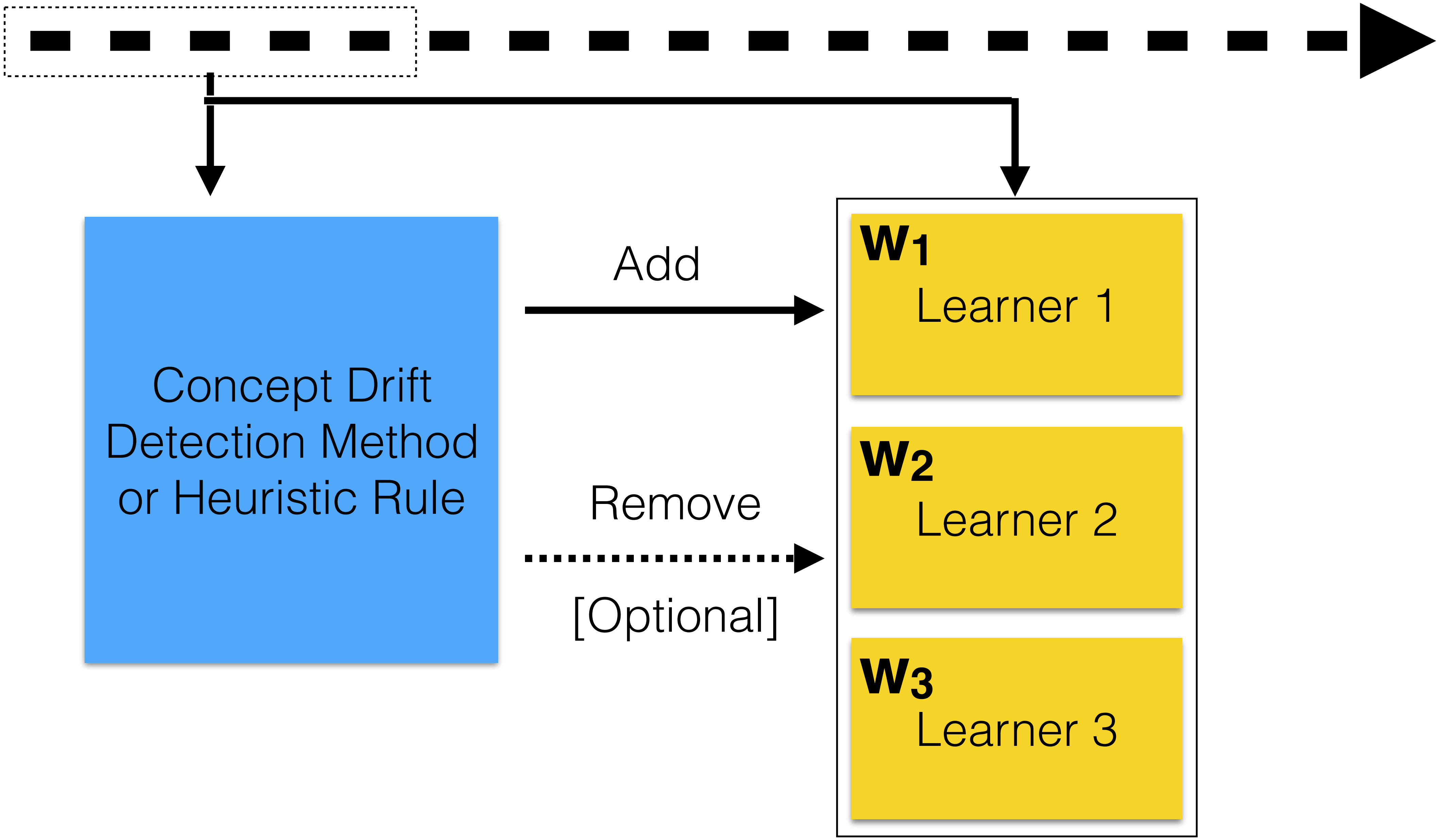
Core Techniques: The General Idea of Adaptation Mechanisms

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



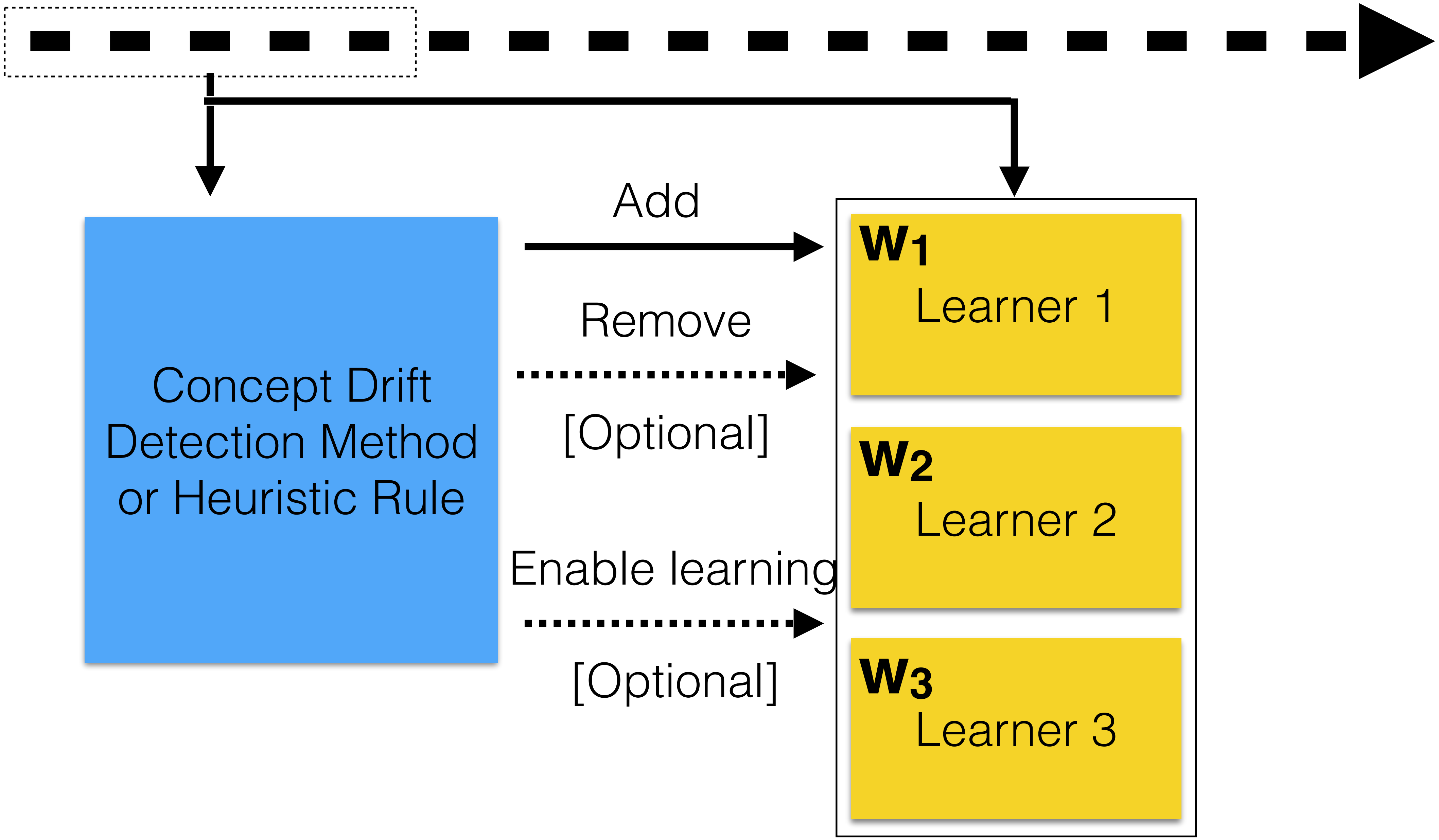
Core Techniques:
The General Idea of Adaptation Mechanisms

Example of adaptation mechanism 4: deciding how / which learners to use for predictions in online or chunk-based learning



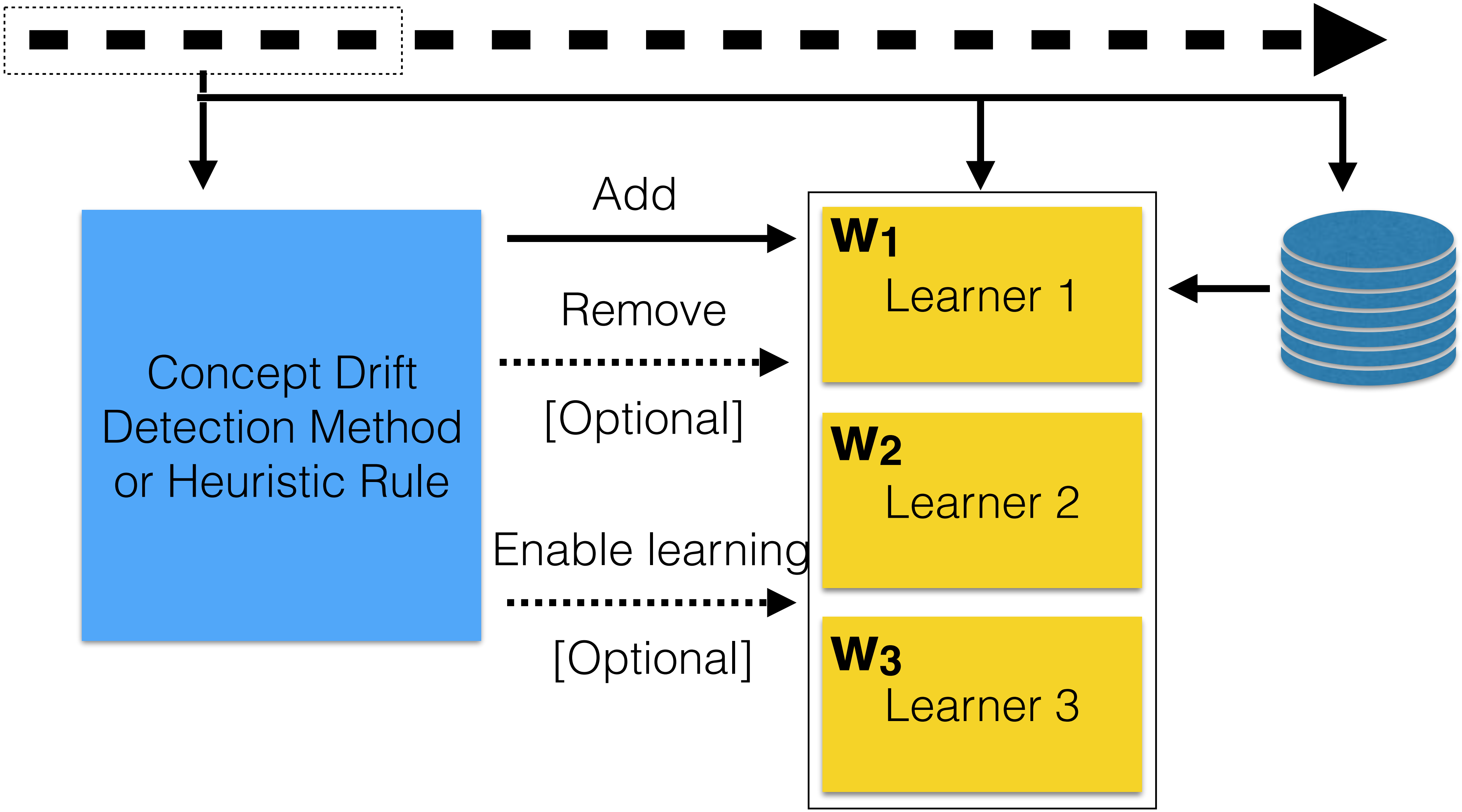
Core Techniques: The General Idea of Adaptation Mechanisms

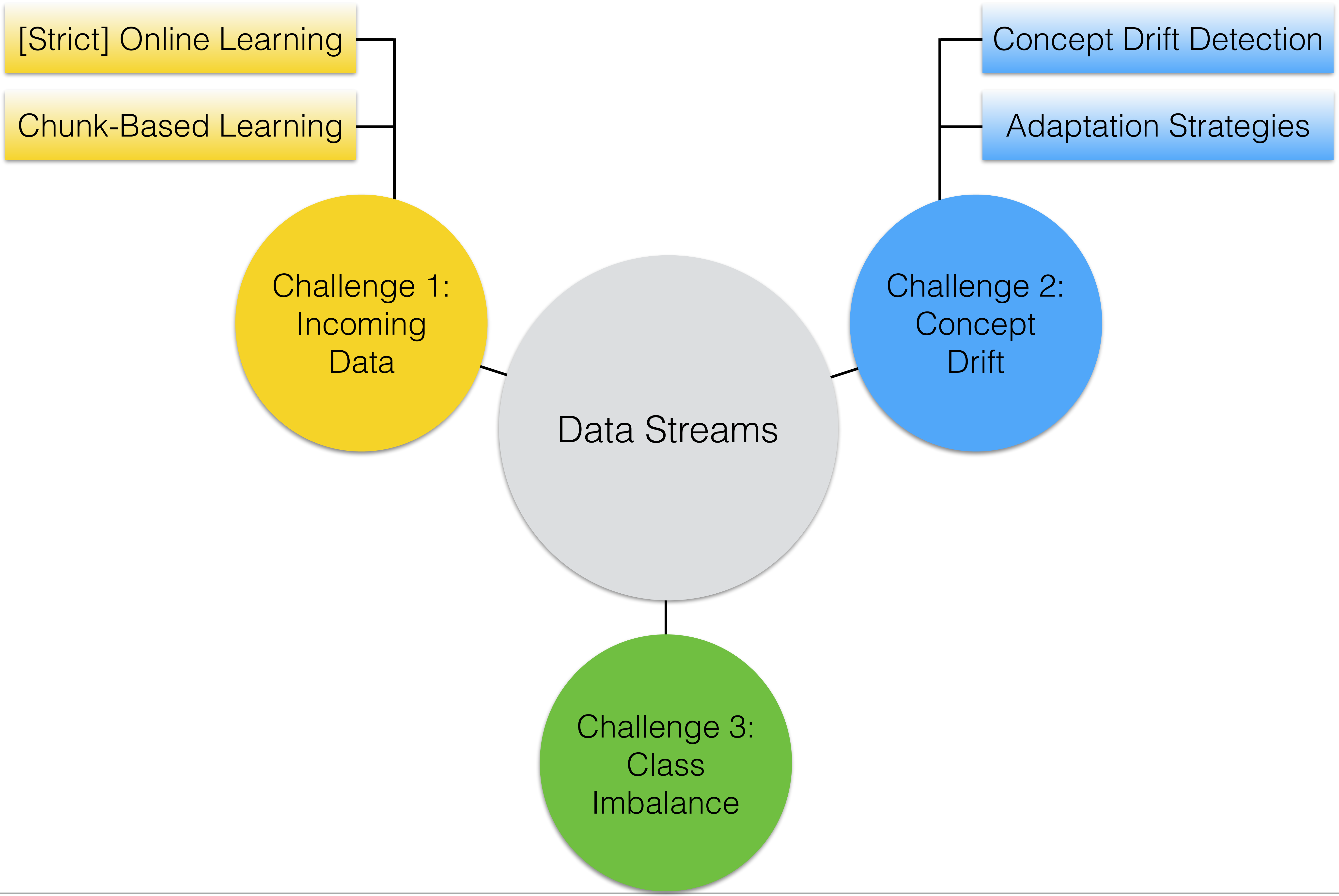
Example of adaptation mechanism 5: deciding which learners can learn current data in online or chunk-based learning



Core Techniques:
The General Idea of Adaptation Mechanisms

Other strategies / components are also possible





[Strict] Online Learning

Chunk-Based Learning

Concept Drift Detection

Adaptation Strategies

Challenge 1: Incoming Data

Data Streams

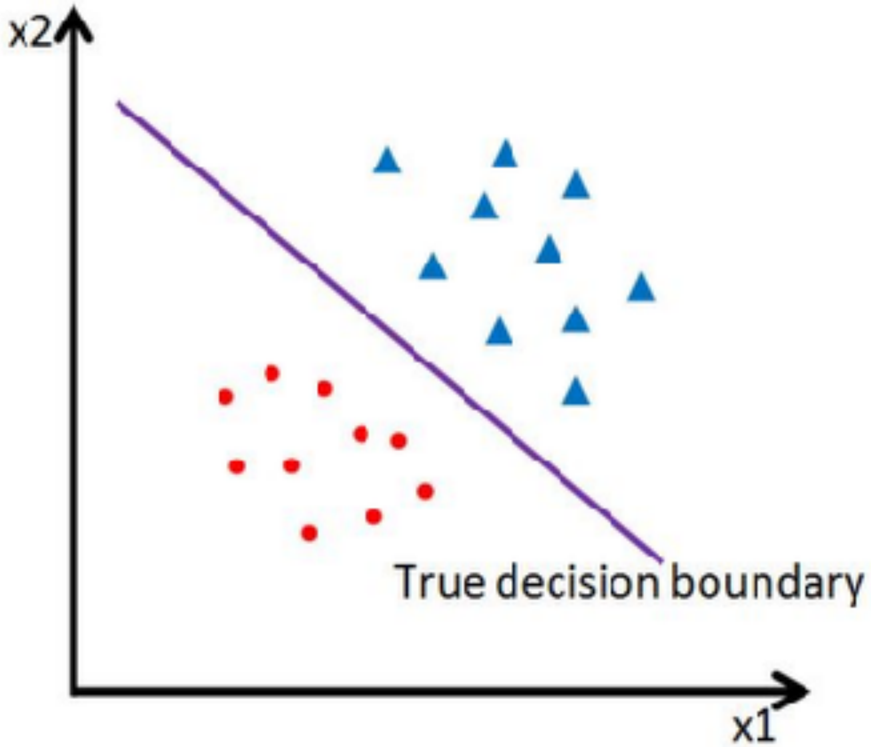
Challenge 2: Concept Drift

Challenge 3: Class Imbalance

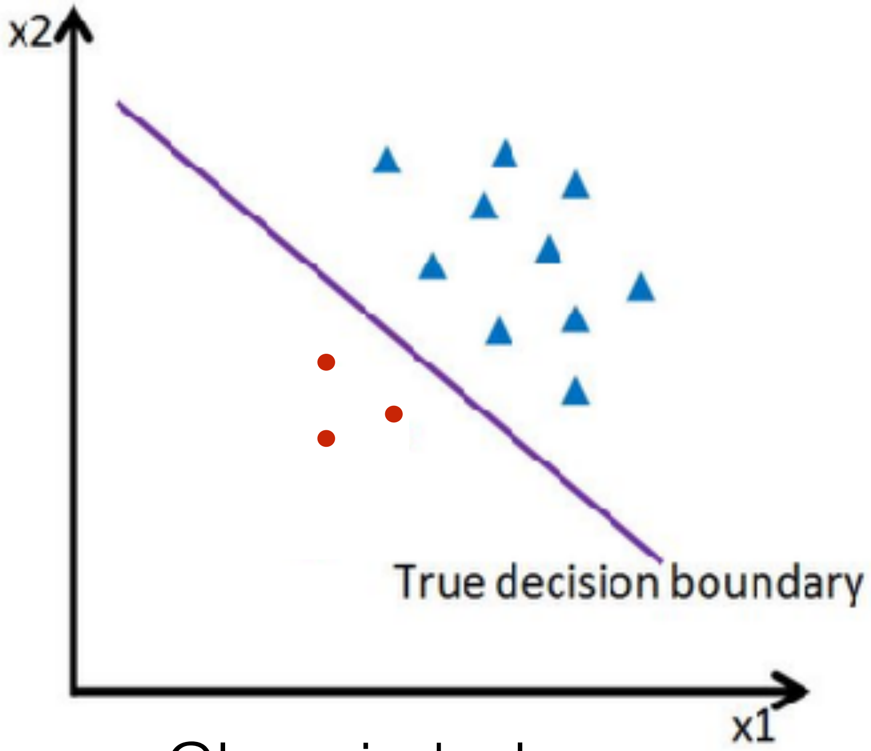
Challenge 3: Class Imbalance

Class imbalance occurs when $\exists c_i, c_j \in \mathcal{Y} \mid p_t(c_i) \leq \delta\, p_t(c_j)$, for a pre-defined $\delta \in (0,1)$.

- It is said that c_i is a minority class, and c_j is a majority class.



No class imbalance



Class imbalance ($\delta = 0.3$)

Challenge 3: Class Imbalance

Class imbalance occurs when $\exists c_i, c_j \in \mathcal{Y} \mid p_t(c_i) \leq \delta\, p_t(c_j)$, for a pre-defined $\delta \in (0,1)$.

- It is said that c_i is a minority class, and c_j is a majority class.

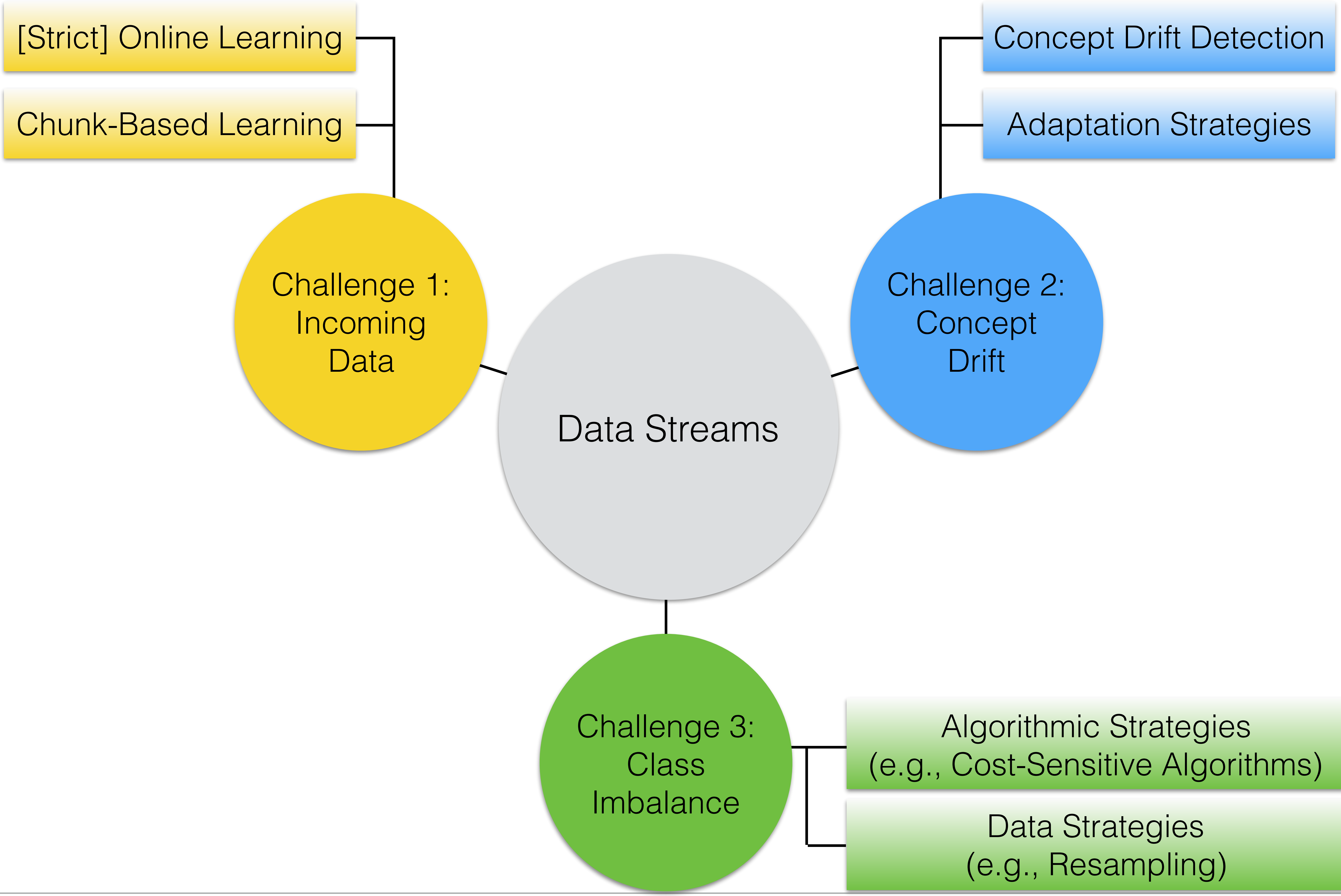
Only ~0.2% of transactions in Atas Worldline's data stream are fraud.

Typically ~20-30% of the software modules are buggy.

Challenge 3: Class Imbalance

Why is that a challenge?

- Machine learning algorithms typically give the same importance to each training example when minimising the average error on the training set.
    - If we have much more examples of a given class than the others, this class may be emphasized in detriment of the other classes.
- Depending on $\mathcal{D}_t$, a predictive model may perform poorly on the minority class.


[Strict] Online Learning
Chunk-Based Learning
Concept Drift Detection
Adaptation Strategies
Challenge 1: Incoming Data
Data Streams
Challenge 2: Concept Drift
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

Core Techniques: General Idea of Algorithmic Strategies

- Loss functions typically give the same importance to examples from different classes. E.g.: consider for illustration purposes:
  - *Accuracy = (TP + TN) / (P + N)*
- Consider the fraud detection problem where our training examples contain:
  - 99.8% of examples from class -1.
  - 0.2% of examples from class +1.
- Consider that our predictive model always predicts -1.
- What is its training accuracy?

Core Techniques:
General Idea of Algorithmic Strategies

- Consider again the following fraud detection problem:
  - 99.8% of examples from class -1.
  - 0.2% of examples from class +1.
- Consider a modification in the accuracy equation, where:
  - class -1 has weight 0.2%
  - class +1 has weight 99.8%
  - Accuracy = (0.998 TP + 0.002 TN) / (0.998 P + 0.002 N)
- What is the training accuracy of a model that always predicts -1?

Core Techniques: General Idea of Algorithmic Strategies

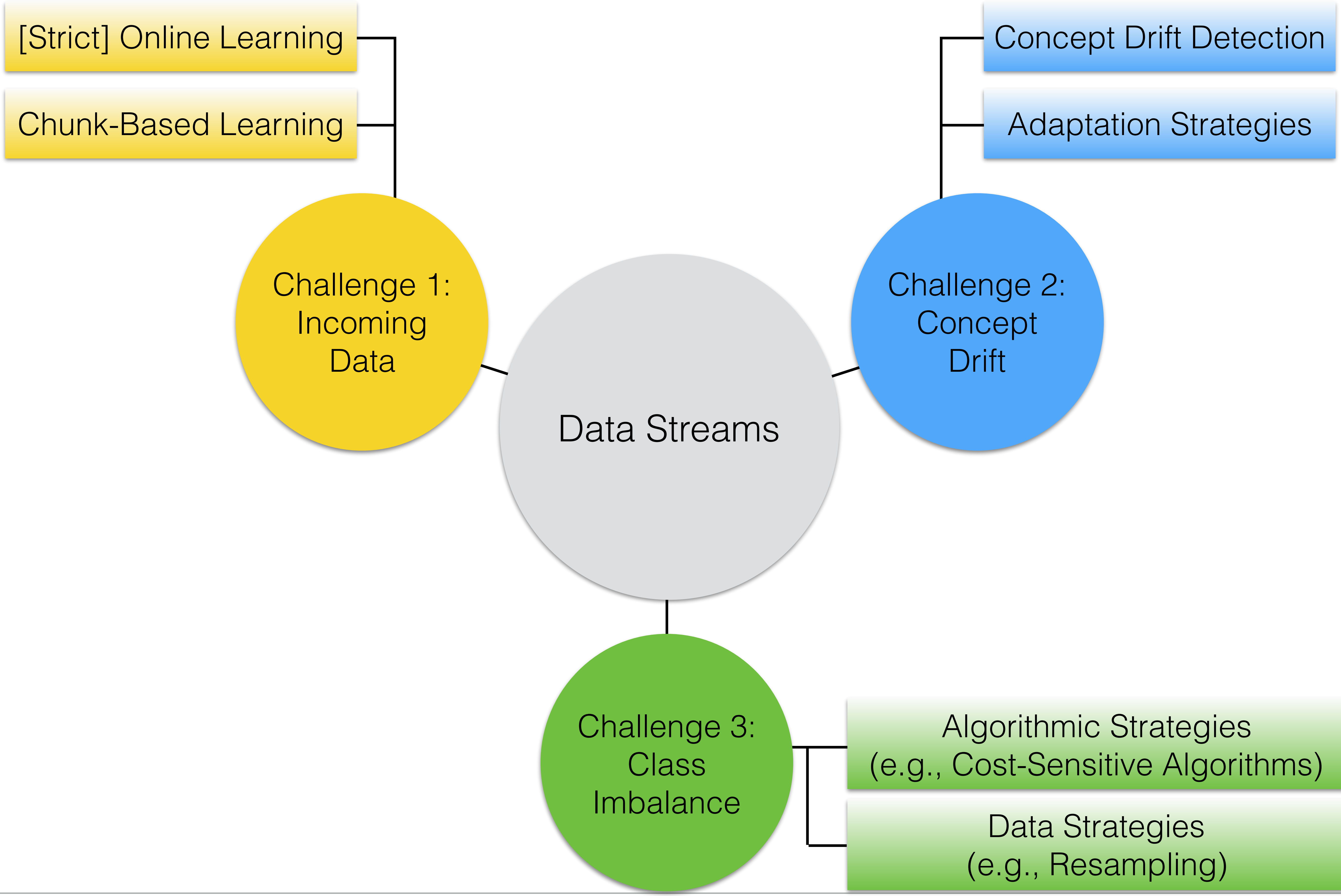
- Use loss functions that lead to a more balanced importance for the different classes.
    - E.g.: cost sensitive algorithms use loss functions that assign different costs (weights) to different classes.

Core Techniques: General Idea of Data Strategies

- Manipulate the data to give a more balanced importance for different classes.
  - E.g.: oversample the minority / undersample the majority class in the training set, so as to balance the number of examples of different classes.

- Potential advantages: applicable to any learning algorithm; could potentially provide extra information about the likely decision boundary.

- Potential disadvantages: increased training time in the case of oversampling; wasting potentially useful information in the case of undersampling.

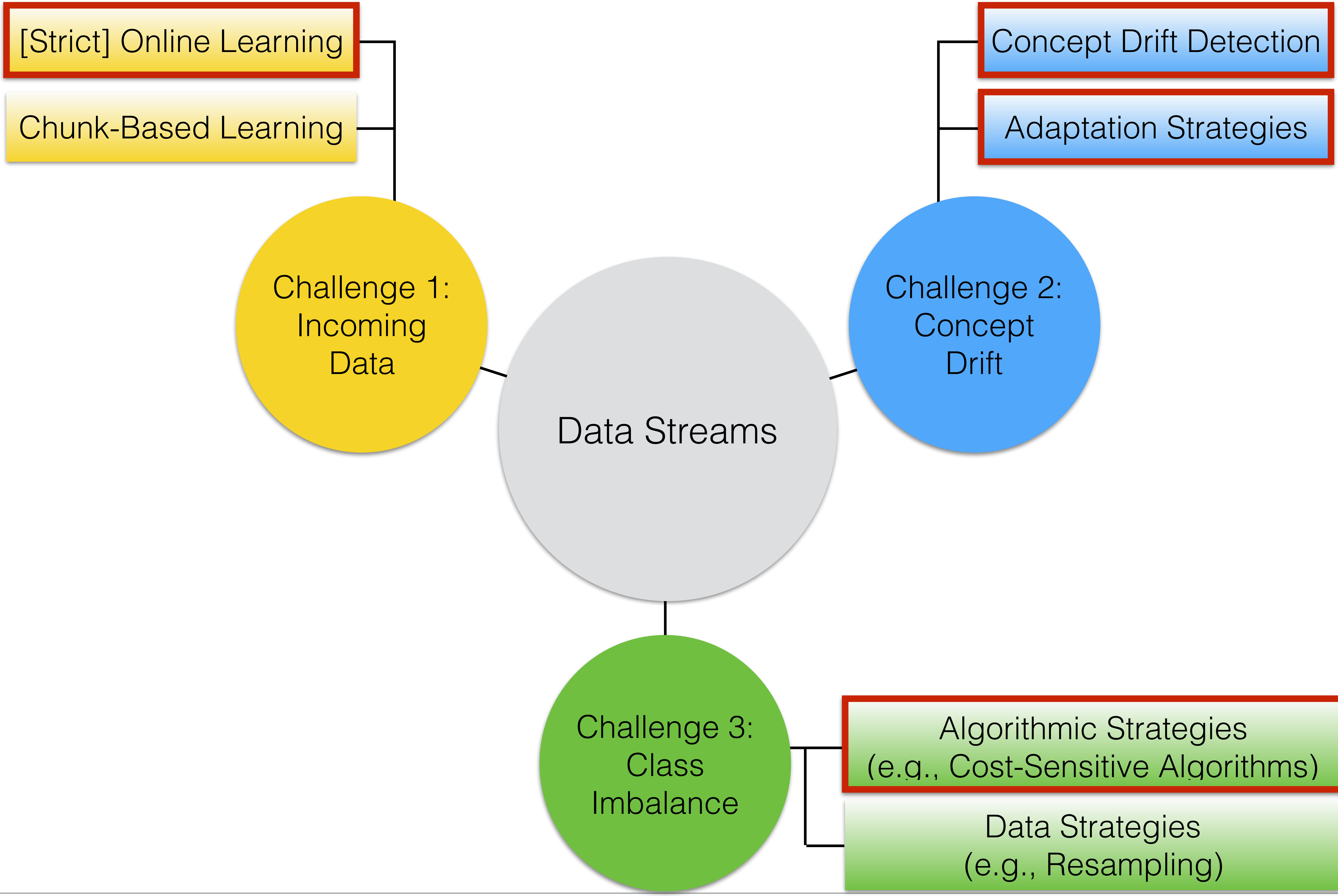

[Strict] Online Learning
Chunk-Based Learning
Concept Drift Detection
Adaptation Strategies
Challenge 1: Incoming Data
Data Streams
Challenge 2: Concept Drift
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

Challenge 4:

Dealing with the three challenges altogether

Outline

- Background and motivation
- Problem formulation
- Challenges and core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges


[Strict] Online Learning
Chunk-Based Learning
Challenge 1: Incoming Data
Concept Drift Detection
Adaptation Strategies
Challenge 2: Concept Drift
Data Streams
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

DDM-OCI: Drift Detection Method for Online Class Imbalance Learning

Detecting concept drift p(y|**x**) in an online manner with class imbalance.

- Metric monitored:
  - Recall of the minority class +1.
  - Whenever an example of class +1 is received, update recall on class +1 using the following time-decayed equation:

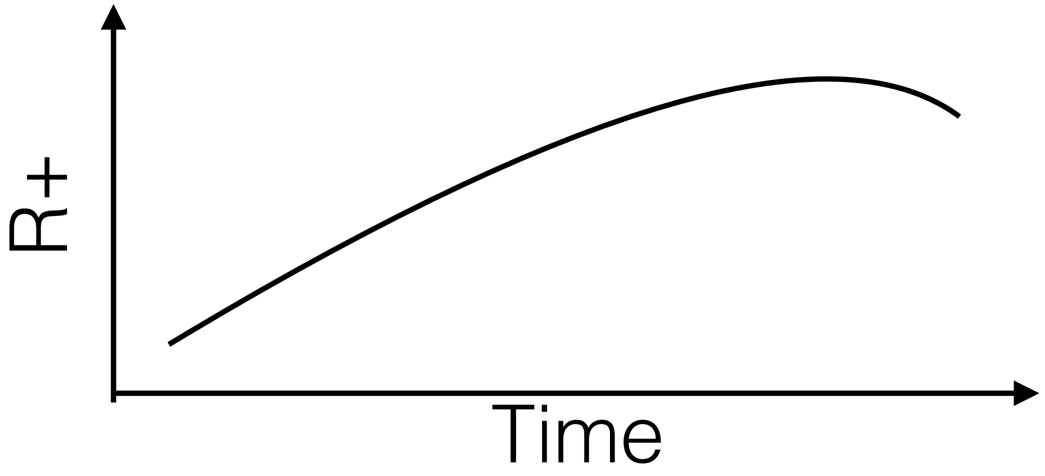
$$R_{+}^{(t)} = \begin{cases} 1_{[\hat{y}=+1]}, & \text{if (x,y) is the first example of class +1} \\ \eta R_{+}^{(t-1)} + (1-\eta) 1_{[\hat{y}=+1]}, & \text{otherwise} \end{cases}$$

where η is a forgetting factor.

S. Wang, L. Minku, D. Ghezzi, D. Caltabiano, P. Tino, X. Yao. "Concept Drift Detection for Online Class Imbalance Learning", in the *2013 International Joint Conference on Neural Networks (IJCNN)*, 10 pages, 2013.

DDM-OCI: Drift Detection Method for Online Class Imbalance Learning

- Change detection test:



Condition for concept drift detection:

$$R_{+}^{(t)} - \sigma_{+}^{(t)} \leq R_{+}^{min} - \alpha \cdot \sigma_{+}^{min}$$

Adapting from concept drift p(y|**x**):

- Resetting mechanism.

Learning class imbalanced data:

- Not achieved.

J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence (SBIA)*, vol. 3171, pp. 286–295, 2004.

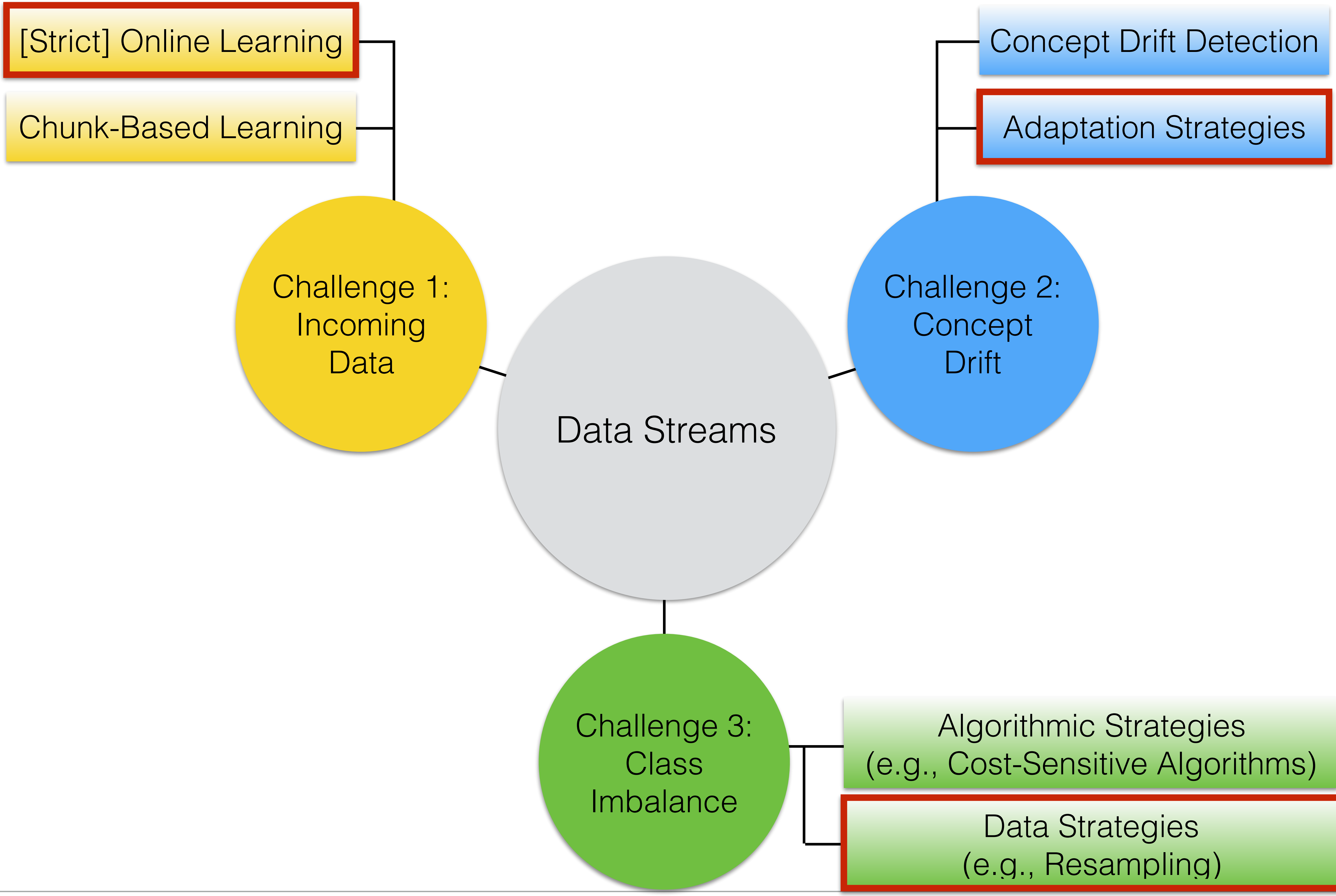
Other Examples of Concept Drift Detection Methods

- PAUC-PH: monitor the drop of Prequential AUC

D. Brzezinski and J. Stefanowski, “Prequential AUC for classifier evaluation and drift detection in evolving data streams,” in *New Frontiers in Mining Complex Patterns* (Lecture Notes in Computer Science), vol. 8983. 2015, pp. 87–101.

- Linear Four Rates: monitor 4 rates from the confusion matrix.

H. Wang and Z. Abraham, “Concept drift detection for streaming data,” in *the International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–9.


[Strict] Online Learning
Chunk-Based Learning
Concept Drift Detection
Adaptation Strategies
Challenge 1: Incoming Data
Data Streams
Challenge 2: Concept Drift
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

OOB and UOB: Oversampling and Undersampling Online Bagging

Dealing with concept drift affecting p(y):

- Time-decayed class size: automatically estimates imbalance status and decides the resampling rate.

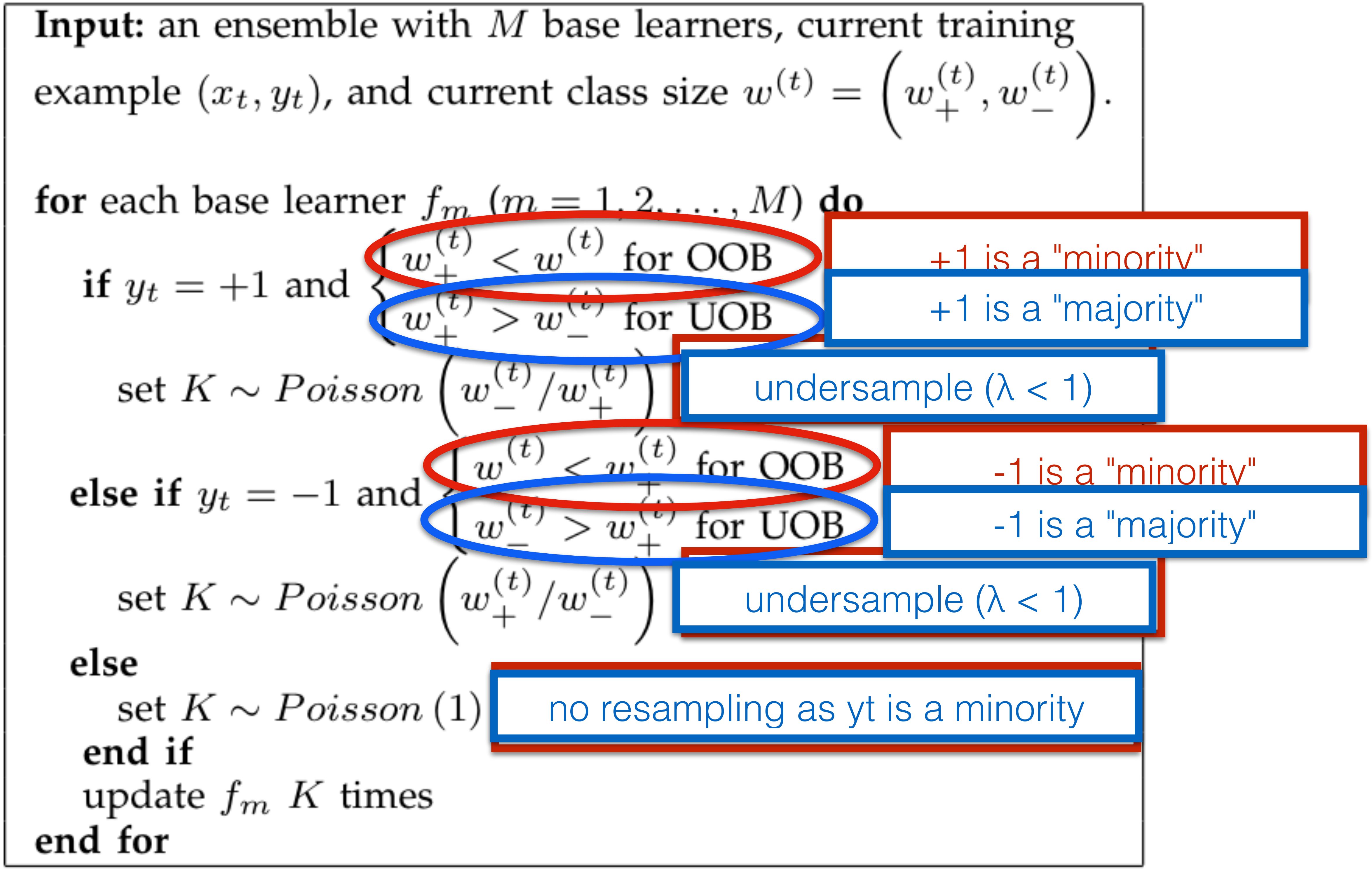
$$w_k^{(t)} = \eta w_k^{(t-1)} + (1-\eta)\ 1_{[(y^{(t)}=c_k)]}$$

where η is a forgetting factor.

S. Wang, L. L. Minku, and X. Yao, “A learning framework for online class imbalance learning,” in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2013, pp. 36–45.

S. Wang, L.L.Minku and X. Yao, "Resampling-Based Ensemble Methods for Online Class Imbalance Learning", *IEEE Transactions on Knowledge and Data Engineering*, 27(5):1356-1368, 2015.

Learning class imbalanced data in online manner with concept drift affecting p(y):



```
Input: an ensemble with $M$ base learners, current training example $(x_t, y_t)$, and current class size $w^{(t)} = \left(w_+^{(t)}, w_-^{(t)}\right)$.

for each base learner $f_m$ $(m = 1, 2, \ldots, M)$ do
    if $y_t = +1$ and $\begin{cases} w_+^{(t)} < w_-^{(t)} & \text{for OOB} \\ w_+^{(t)} > w_-^{(t)} & \text{for UOB} \end{cases}$
        set $K \sim Poisson\left(w_-^{(t)}/w_+^{(t)}\right)$
    else if $y_t = -1$ and $\begin{cases} w_-^{(t)} < w_+^{(t)} & \text{for OOB} \\ w_-^{(t)} > w_+^{(t)} & \text{for UOB} \end{cases}$
        set $K \sim Poisson\left(w_+^{(t)}/w_-^{(t)}\right)$
    else
        set $K \sim Poisson\,(1)$
    end if
    update $f_m$ $K$ times
end for
```

+1 is a "minority"

+1 is a "majority"

undersample (λ < 1)

-1 is a "minority"

-1 is a "majority"

undersample (λ < 1)

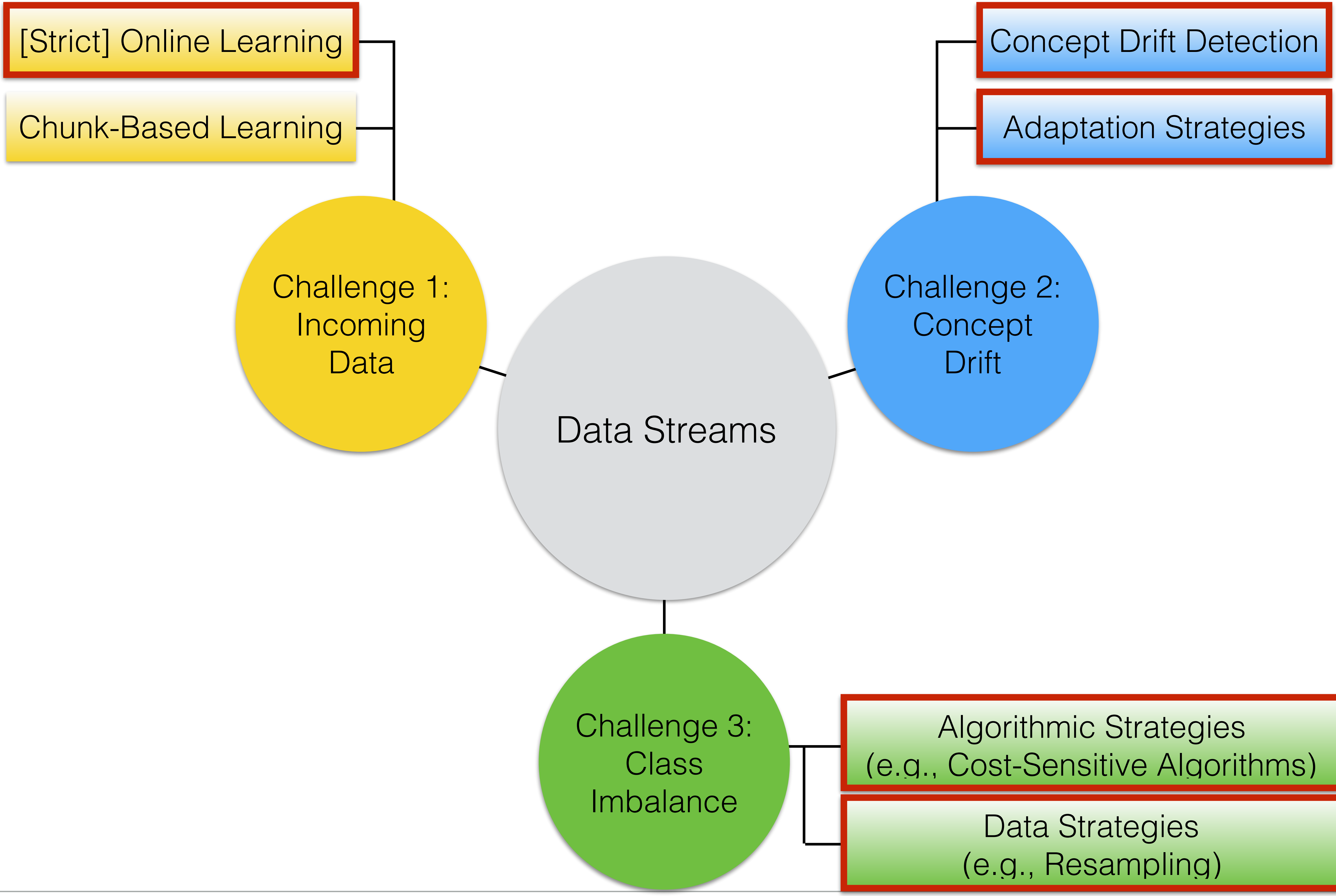
no resampling as yt is a minority

Problem: can't handle multi-class problems, and concept drifts other than p(y).

Other Examples of Algorithms

MOOB and MUOB: extensions of OOB and UOB for multi-class problems.

S.Wang, L.L.Minku, and X.Yao. “Dealing with Multiple Classes in Online Class Imbalance Learning”, in t*he 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. Pages 2118-2124, 2016.


[Strict] Online Learning
Chunk-Based Learning
Challenge 1: Incoming Data
Concept Drift Detection
Adaptation Strategies
Challenge 2: Concept Drift
Data Streams
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

DDM-OCI + Resampling

Detecting concept drift p(y|**x**) in an online manner with class imbalance and adapting from it:

- DDM-OCI.

Learning class imbalanced data in an online manner with concept drift p(y):

- OOB or UOB.

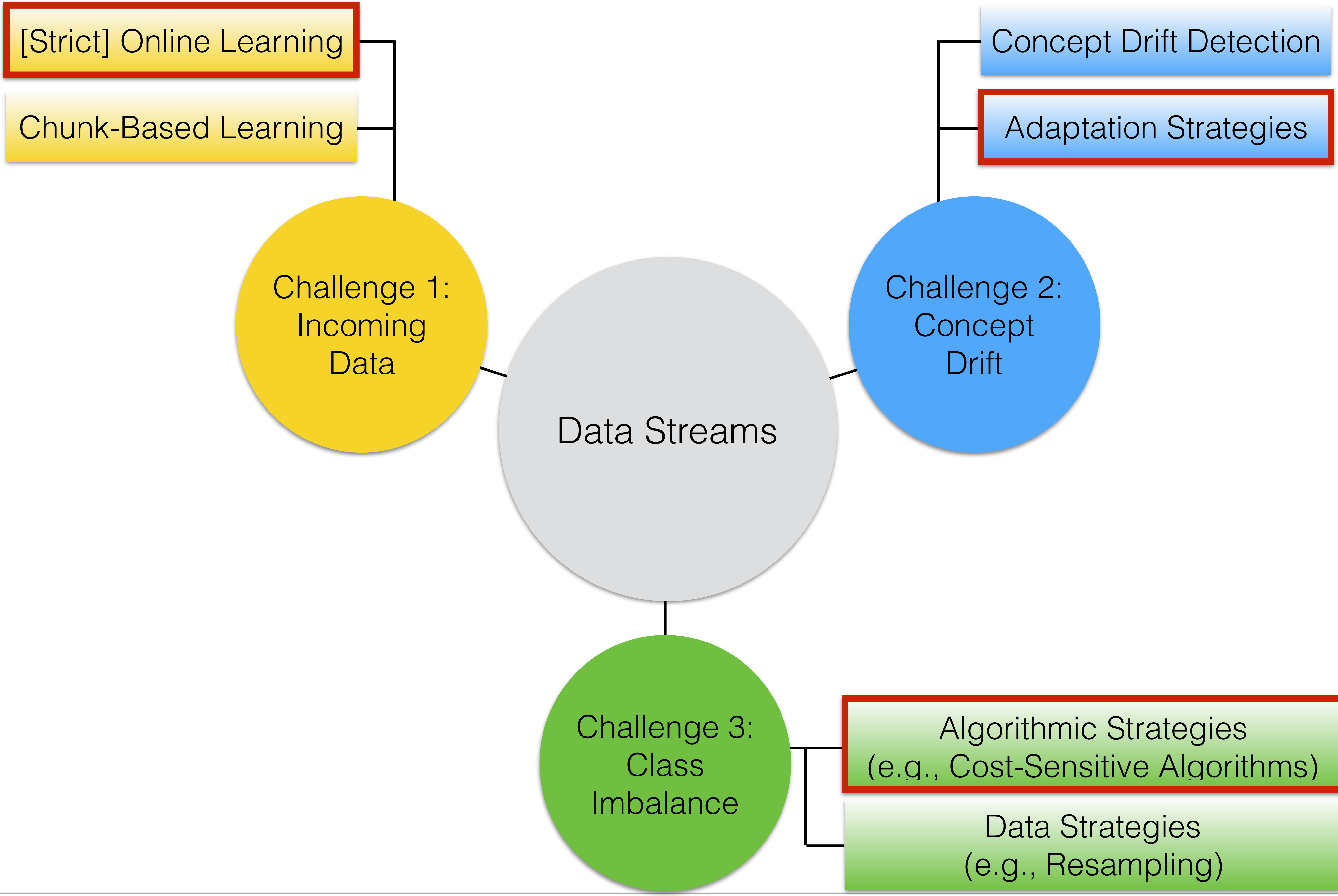
S. Wang, L. Minku, X. Yao. "A Systematic Study of Online Class Imbalance Learning with Concept Drift", IEEE Transactions on Neural Networks and Learning Systems, 2017 (in press).

Other Examples of Algorithms

ESOS-ELM: Ensemble of Subset Online Sequential Extreme Learning Machine

- Also uses algorithmic class imbalance strategy for concept drift detection and online resampling strategy for learning, but
- it preserves a whole ensemble of models representing potentially different concepts, weighted based on G-mean.

B. Mirza, Z. Lin, and N. Liu, “Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift,” *Neurocomputing*, vol. 149, pp. 316–329, Feb. 2015.



[Strict] Online Learning

Chunk-Based Learning

Challenge 1: Incoming Data

Data Streams

Challenge 2: Concept Drift

Concept Drift Detection

Adaptation Strategies

Challenge 3: Class Imbalance

Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)

Data Strategies (e.g., Resampling)

RLSACP: Recursive Least Square Adaptive Cost Perceptron

Loss function:

$$E_t(\beta) = \sum_{i=1}^{t} w_i(y_i) \cdot \lambda^{t-i} \cdot e_i(\beta) \qquad e_t(\beta) = \frac{1}{2}(y_t - \phi(\beta_t^T x_t))^2$$

(x_t, y_t) is the training example received at time step *t*; ϕ is the activation function of the neuron, β_t are the neuron parameters at time *t*;

$\lambda \in [0,1]$ is a forgetting factor to deal with concept drift p(y|**x**);

$w_t(y_t)$ is the weight associated to class y_t at time *t*, to deal with class imbalance.

A. Ghazikhani, R. Monsefi, and H. S. Yazdi, "Recursive least square perceptron model for non-stationary and imbalanced data stream classification", Evolving Systems, 4(2):119–131, 2013.

RLSACP: Recursive Least Square Adaptive Cost Perceptron

Learning class imbalanced data in an online manner with concept drift affecting p(y|**x**):

$$E_t(\beta) = w_i(y_i) \cdot e_i(\beta) \; + \; \lambda \cdot E_{t-1}(\beta)$$

β are the neuron parameters;

$\lambda \in [0,1]$ is a forgetting factor to deal with concept drift;

$w_t(y_t)$ is the weight associated to class y_t at time *t*, to deal with class imbalance.

A. Ghazikhani, R. Monsefi, and H. S. Yazdi, "Recursive least square perceptron model for non-stationary and imbalanced data stream classification", Evolving Systems, 4(2):119–131, 2013.

RLSACP: Recursive Least Square Adaptive Cost Perceptron

Dealing with concept drift affecting p(y):

- Update $w_t(y_t)$ based on:
  - Imbalance ratio based on a fixed number of recent examples.
  - Current recalls on the minority and majority class.

Problem: single perceptron.

A. Ghazikhani, R. Monsefi, and H. S. Yazdi, “Recursive least square perceptron model for non-stationary and imbalanced data stream classification”, Evolving Systems, 4(2):119–131, 2013.

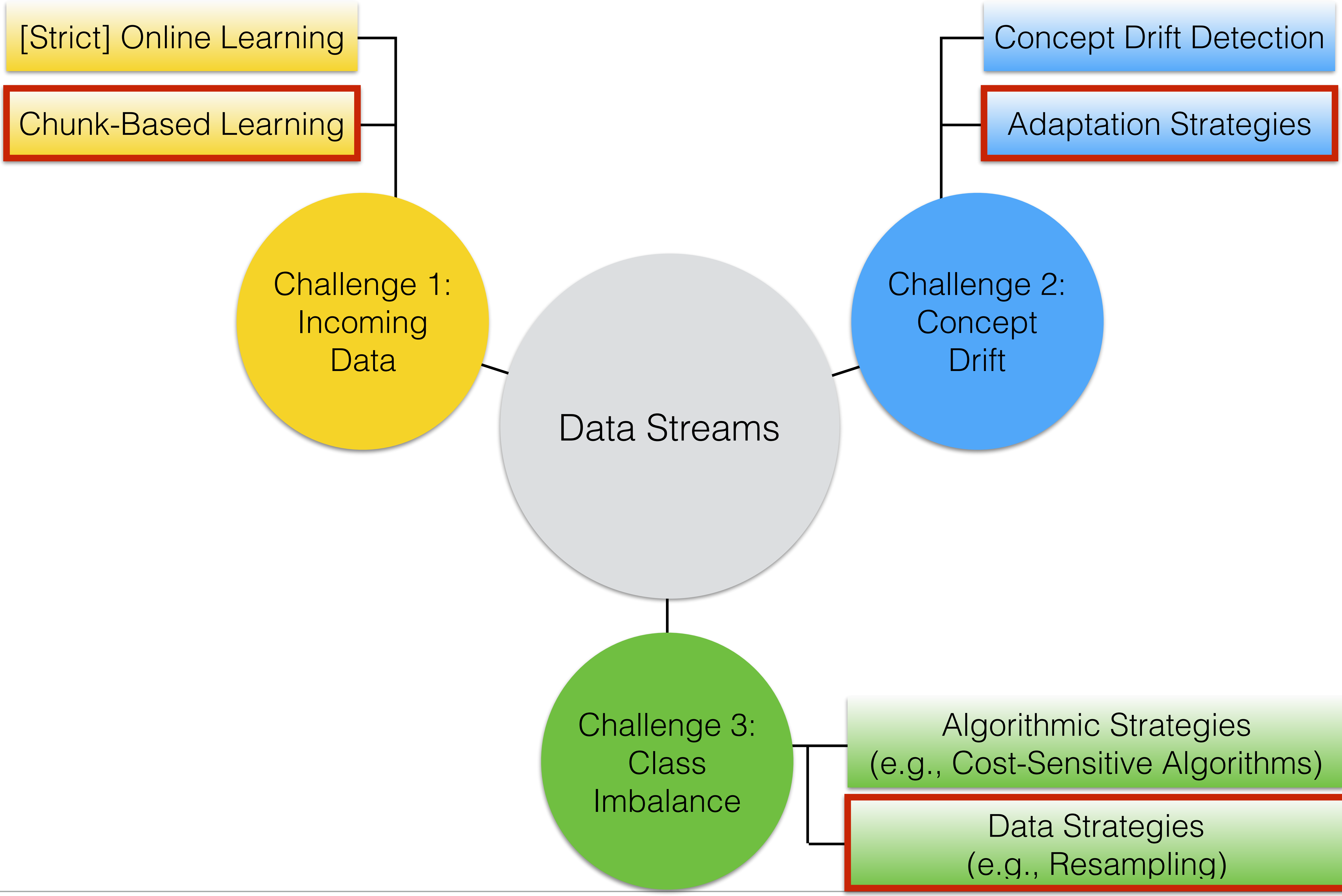
Other Examples of Algorithms

ONN: Online Multi-Layer Perceptron NN model.

A. Ghazikhani, R. Monsefi, and H. S. Yazdi, “Online neural network model for non-stationary and imbalanced data stream classification,” *International Journal of Machine Learning and Cybernetics*, 5(1):51–62, 2014.

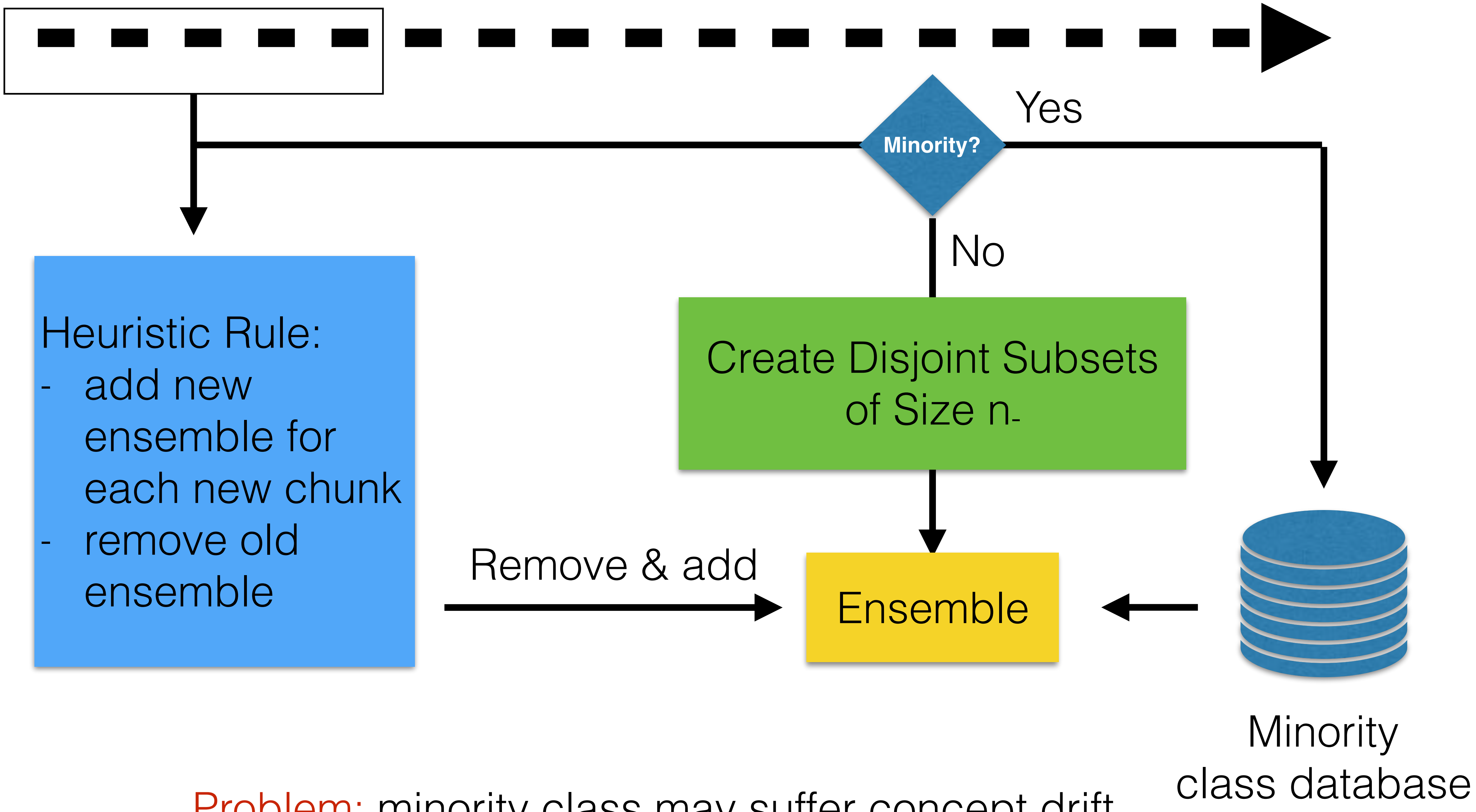
Outline

- Background and motivation
- Problem formulation
- Challenges and core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges



[Strict] Online Learning
Chunk-Based Learning
Challenge 1: Incoming Data
Data Streams
Concept Drift Detection
Adaptation Strategies
Challenge 2: Concept Drift
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

Uncorrelated “Bagging”



Problem: minority class may suffer concept drift.

J. Gao, W. Fan, J. Han, P. S. Yu. “A General Framework for Mining Concept-Drifting Data Streams with Skewed Distributions”, in the *International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 226-235, 2003.

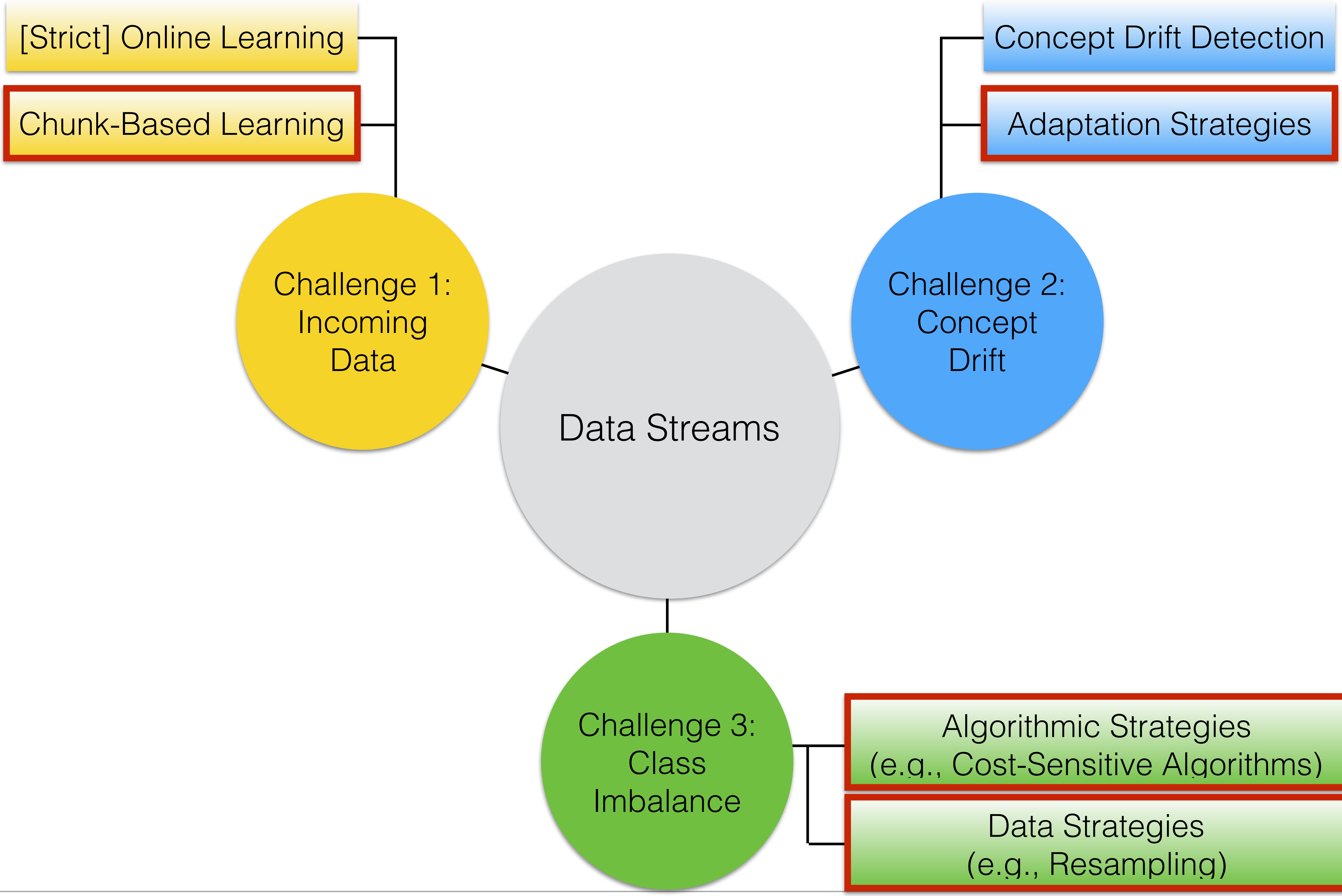
Other Examples of Algorithms

- SERA — uses the N old examples of the minority class with the smallest distance to the new examples of the minority class.

S. Chen and H. He. "SERA: Selectively Recursive Approach towards Nonstationary Imbalanced Stream Data Mining", in the *International Joint Conference on Neural Networks*, 2009.

- REA — uses the N old examples of the minority class that have the largest number of nearest neighbours of the minority class in the new chunk.

S. Chen and H. He. "Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach", *Evolving Systems* 2:35–50, 2011.



Data Streams

Challenge 1: Incoming Data

[Strict] Online Learning

Chunk-Based Learning

Challenge 2: Concept Drift

Concept Drift Detection

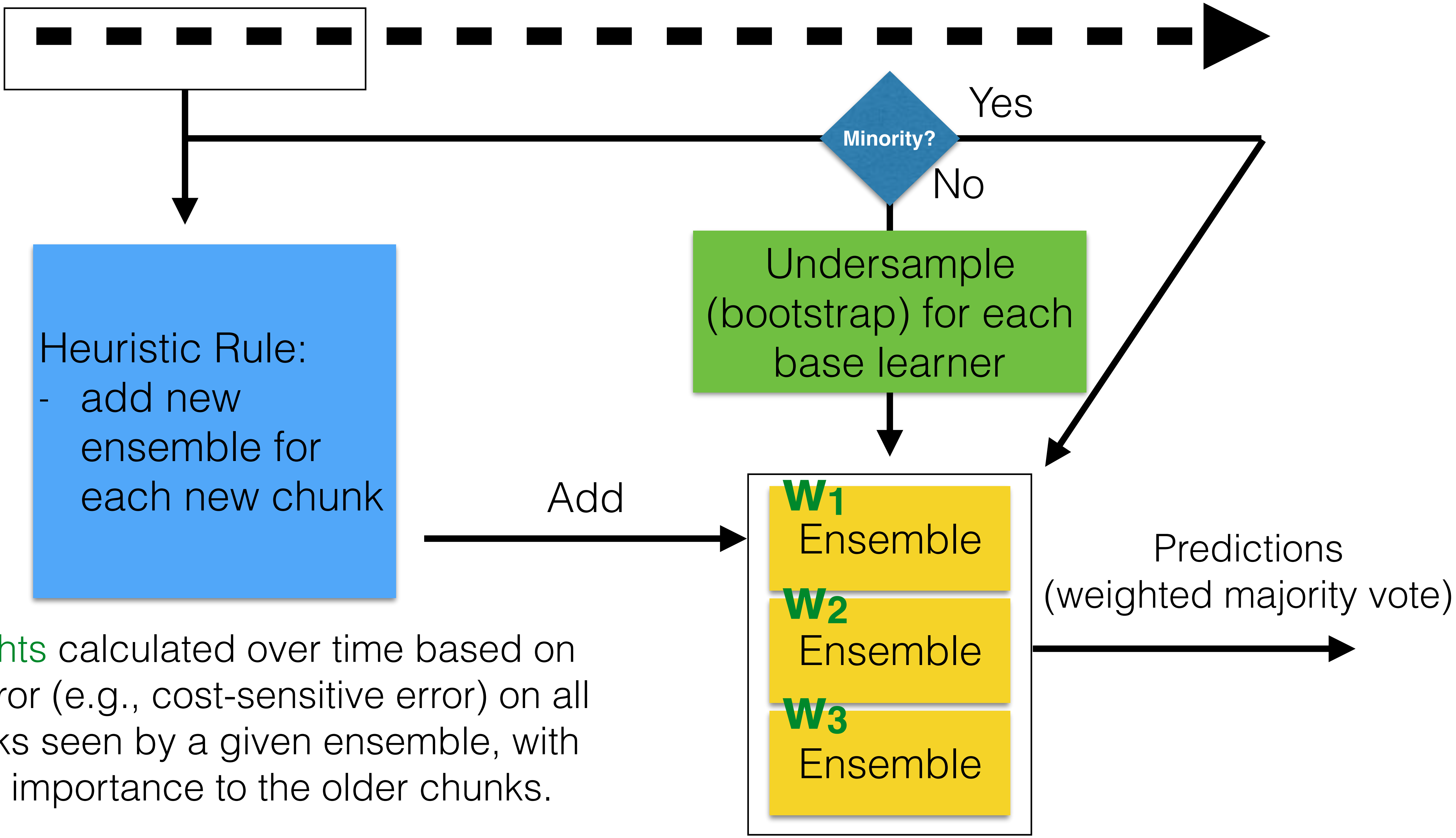
Adaptation Strategies

Challenge 3: Class Imbalance

Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)

Data Strategies (e.g., Resampling)

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


Minority?
Yes
No
Undersample (bootstrap) for each base learner
Heuristic Rule:
- add new ensemble for each new chunk
Add
W_1 Ensemble
W_2 Ensemble
W_3 Ensemble
Predictions (weighted majority vote)

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.

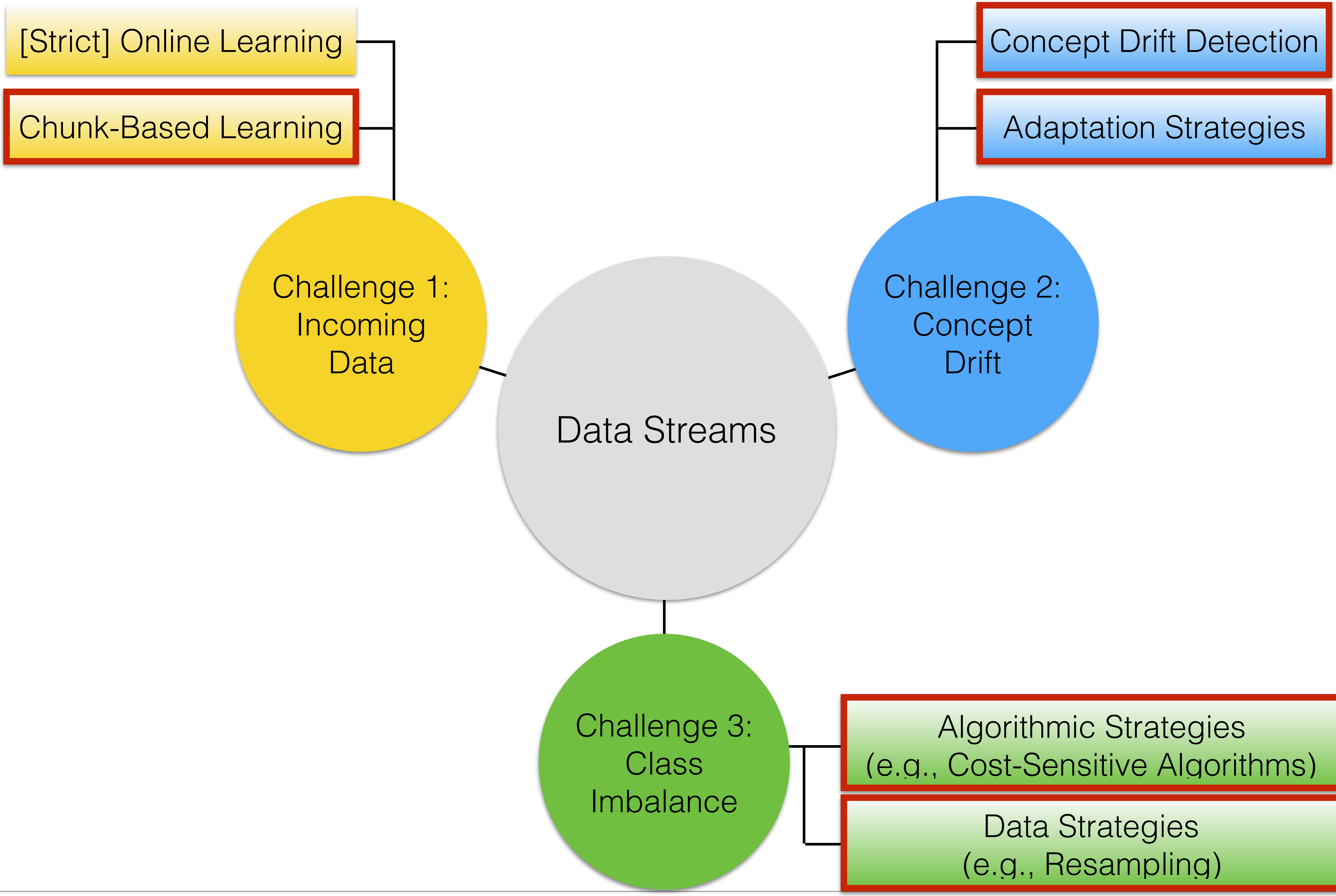
G. Ditzler and R. Polikar. "Incremental Learning of Concept Drift from Streaming Imbalanced Data", *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2283-2301, 2013.

Other Examples of Algorithms

Learn++.CDS: Learn++ for Concept Drift with SMOTE

- Also creates new classifiers for new chunks and combine them into an ensemble.
- Uses SMOTE-like resampling and boosting-like weights for ensemble classifiers.

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.


[Strict] Online Learning
Chunk-Based Learning
Challenge 1: Incoming Data
Data Streams
Challenge 2: Concept Drift
Concept Drift Detection
Adaptation Strategies
Challenge 3: Class Imbalance
Algorithmic Strategies (e.g., Cost-Sensitive Algorithms)
Data Strategies (e.g., Resampling)

Other Examples of Algorithms

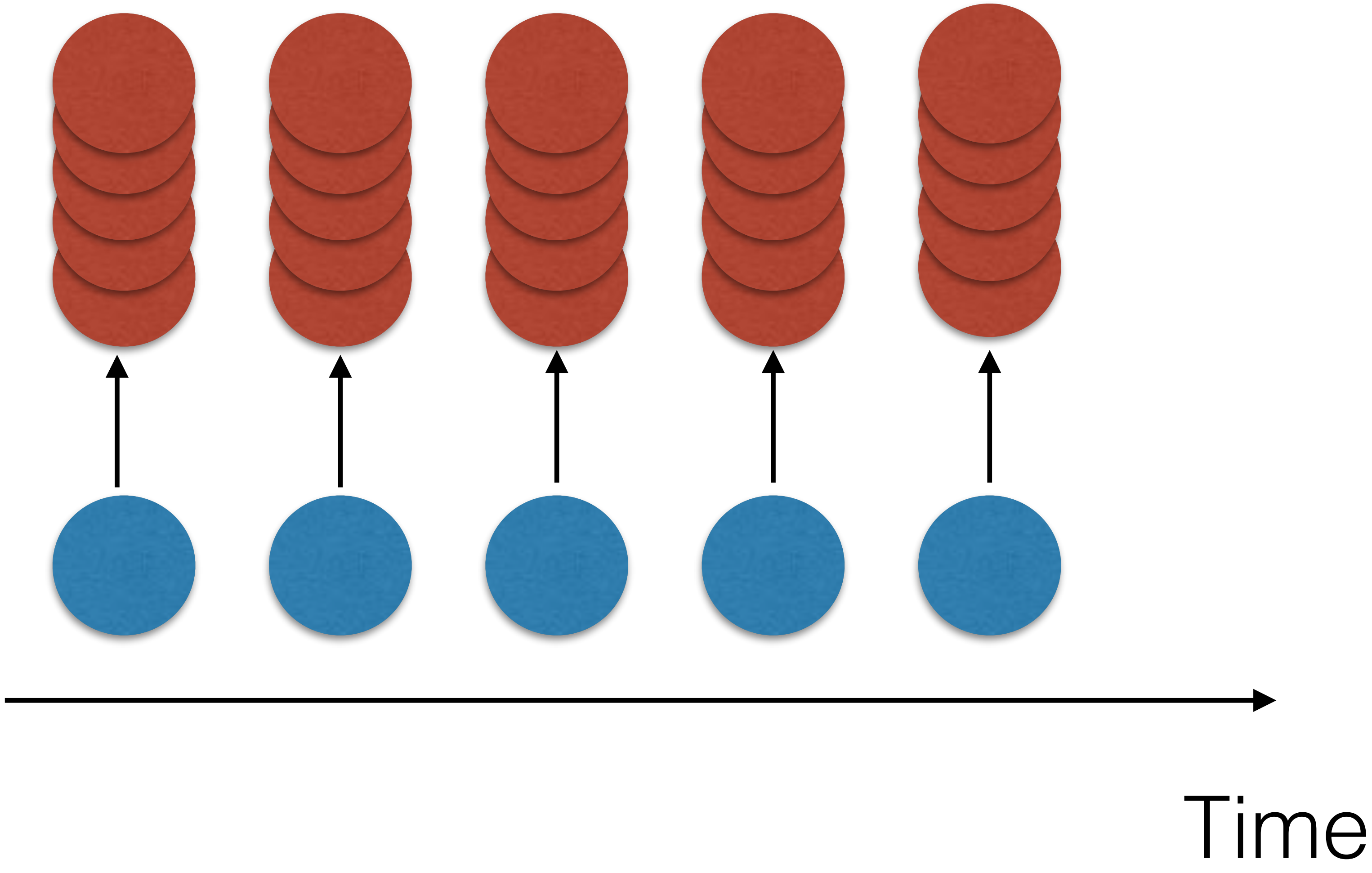
- HUWRS.IP: Heuristic Updatable Weighted Random Subspaces-Instance Propagation
  - Trains new learners on new chunks, based on resampling.
  - Uses cost-sensitive distribution distance function to decide weights of ensemble members.
  - Cost-sensitive distance function could be argued to be a concept drift detector.

T. Ryan Hoens and N. Chawla.. "Learning in Non-stationary Environments with Class Imbalance", in the *International Conference on Pattern Recognition*, 2010.

Outline

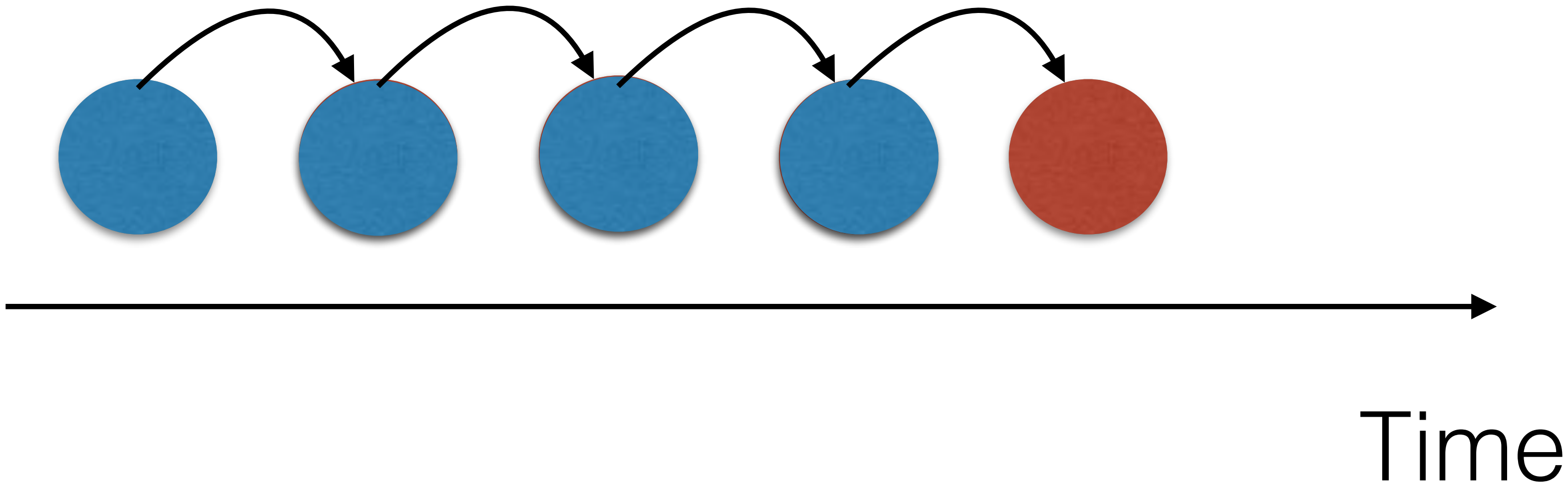
- Background and motivation
- Problem formulation
- Challenges and core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges

Performance on a Separate Test Set



Problem: typically infeasible for real world problems.

Prequential Performance



$$perf^{(t)} = \begin{cases} perf_{ex}^{(t)}, \text{ if t=1} \\ \dfrac{(t-1)perf^{(t-1)} + perf_{ex}^{(t)}}{t}, \text{ otherwise} \end{cases}$$

Problem: does not reflect the *current* performance.

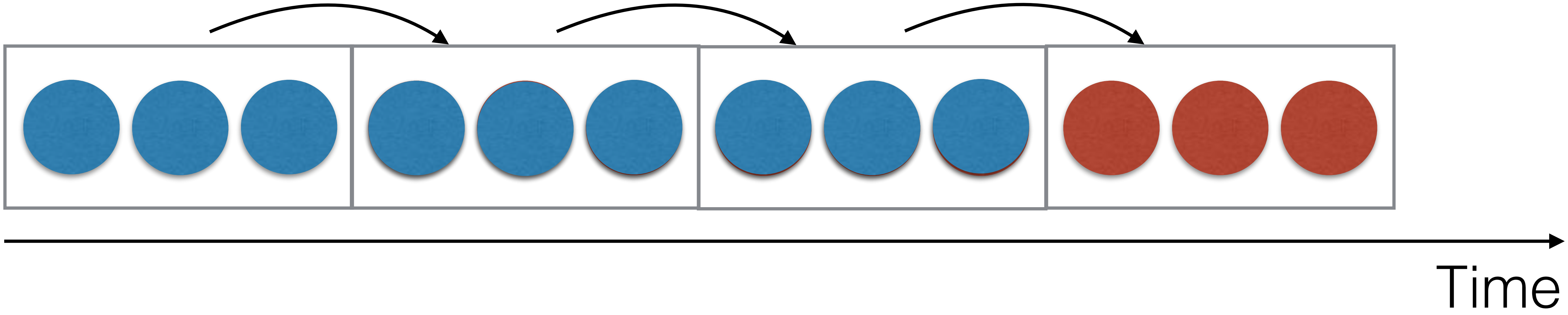
Exponentially Decayed Prequential Performance

$$perf^{(t)} = \begin{cases} perf_{ex}^{(t)}, \text{ if t=1} \\ \eta \cdot perf^{(t-1)} + (1-\eta) \cdot perf_{ex}^{(t)}, \text{ otherwise} \end{cases}$$

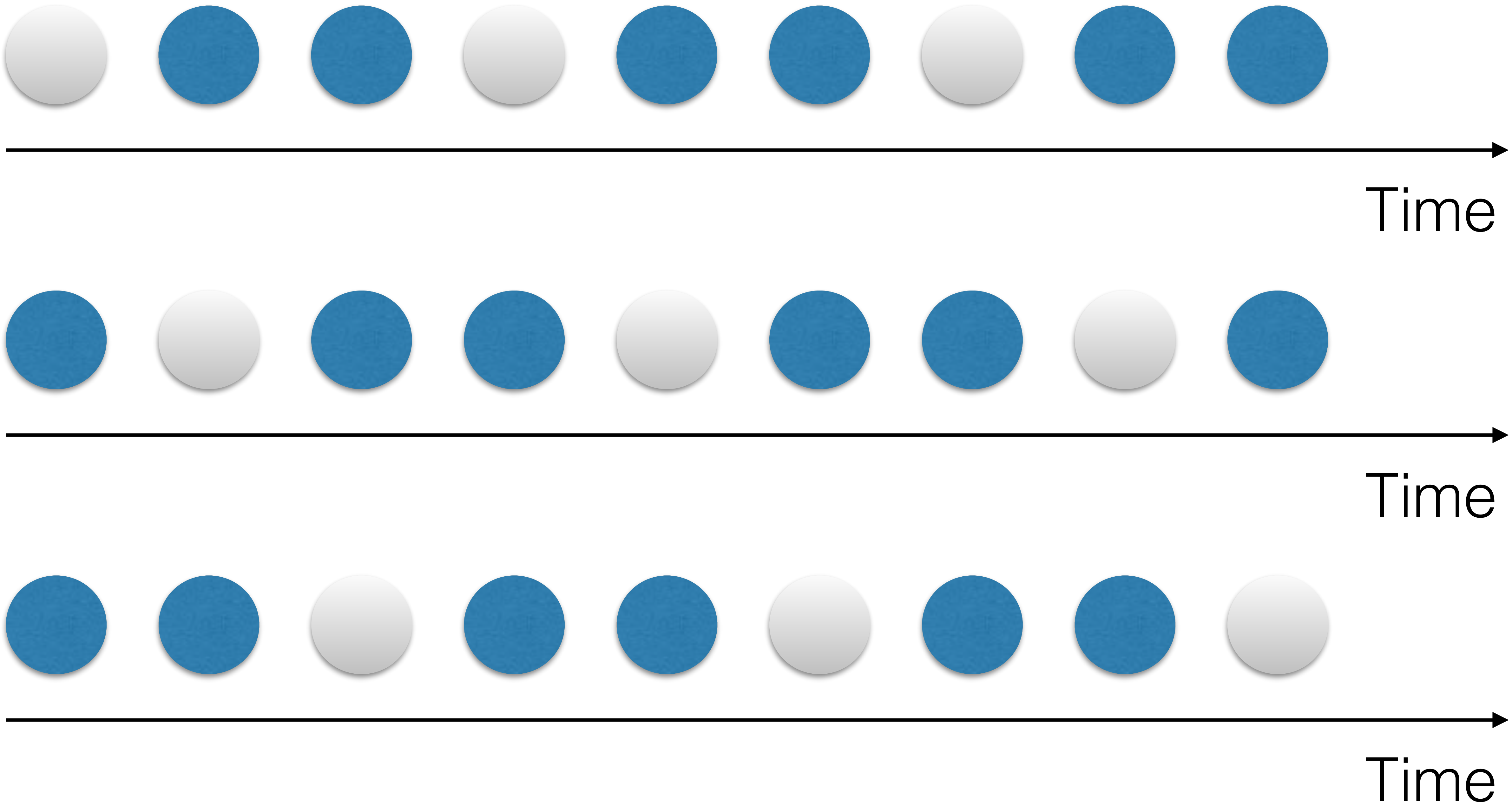
- Alternative for artificial datasets: reset prequential performance upon known concept drifts.

J.Gama, R.Sebastiao, P.P.Rodrigues. “Issues in Evaluation of Stream Learning Algorithms”, in the ACM SIGKDD international conference on knowledge discovery and data mining, 329338, 2009.

Chunk-Based Performance



Variations of Cross-Validation

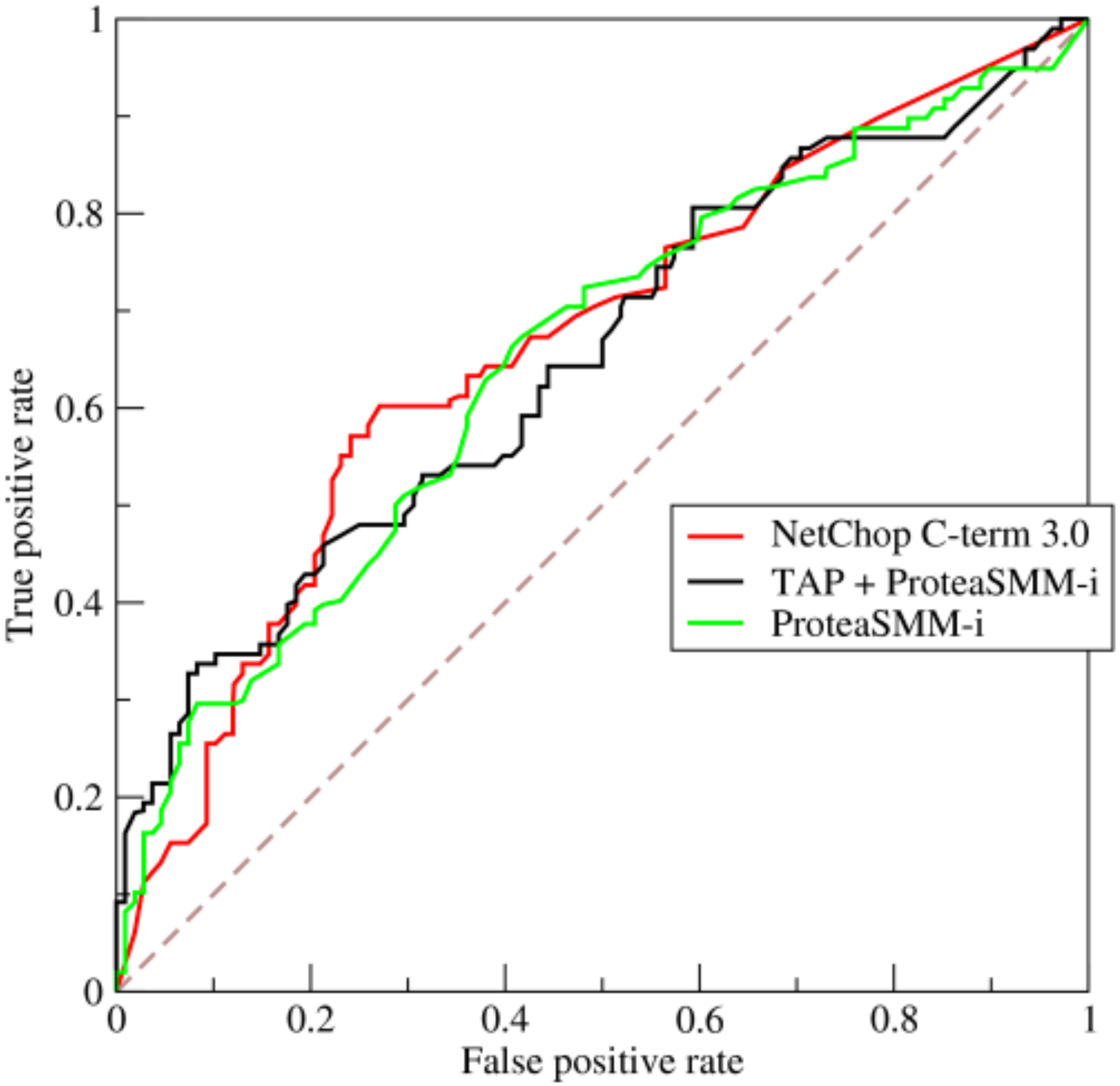


Y. Sun, K. Tang, L. Minku, S. Wang and X. Yao. Online Ensemble Learning of Data Streams with Gradually Evolved Classes, *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1532-1545, 2016.

Performance Metrics for Class Imbalanced Data

- Accuracy is inadequate.
  - *(TP + TN) / (P + N)*
- Precision is inadequate.
  - *TP / (TP + FP)*
- Recall on each class separately is more adequate.
  - *TP / P* and *TN / N*.
- F-measure: not very adequate.
  - Harmonic mean of precision and recall.
- G-mean is more adequate.
  - $\sqrt{TP/P * TN/N}$
- ROC Curve is more adequate.
  - Recall on positive class *(TP / P)* vs False Alarms *(FP / N)*

Prequential AUC



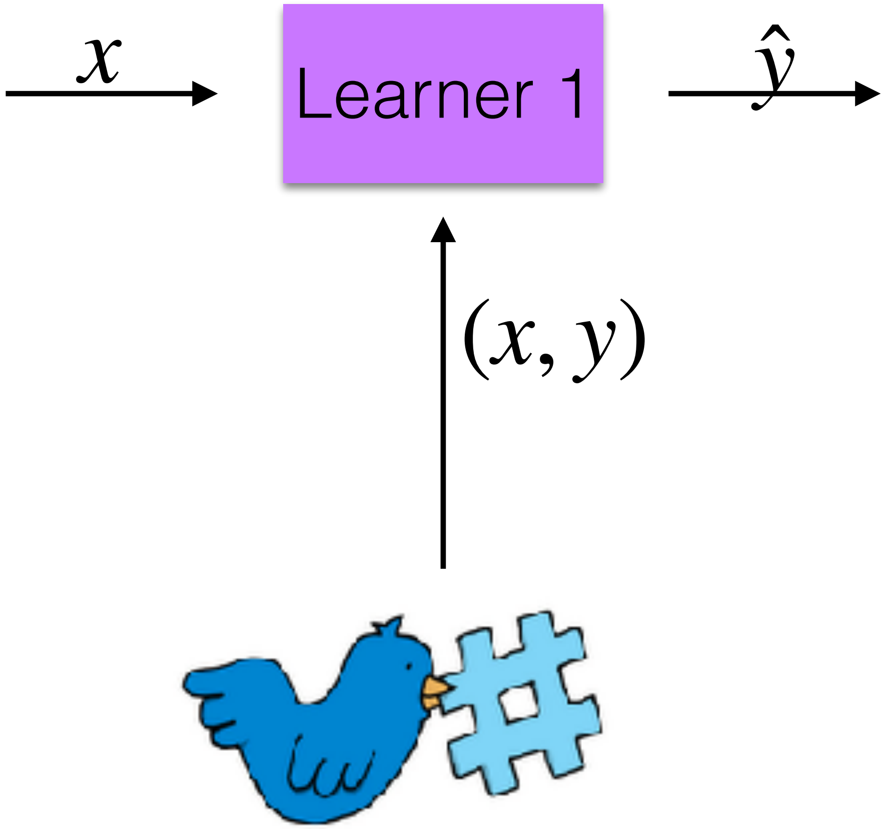
- We need to sort the scores given by the classifiers to compute AUC.
- A sorted sliding window of scores can be maintained in a red-black tree.
- Scores can be added and removed from the sorted tree in O(2log d), where d is the size of the window.
- Sorted scores can be retrieved in O(d).
- For each new example, AUC can be computed in O(d+2log d).
- If size of the window is considered a constant, AUC can be computed in O(1).

D. Brzezinski and J. Stefanowski. “Prequential AUC for classifier evaluation and drift detection in evolving data streams”, in the *3rd International Conference on New Frontiers in Mining Complex Patterns*, pp. 87-101, 2014.

Outline

- Background and motivation
- Problem formulation
- Challenges and core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges

Tweet Topic Classification



x

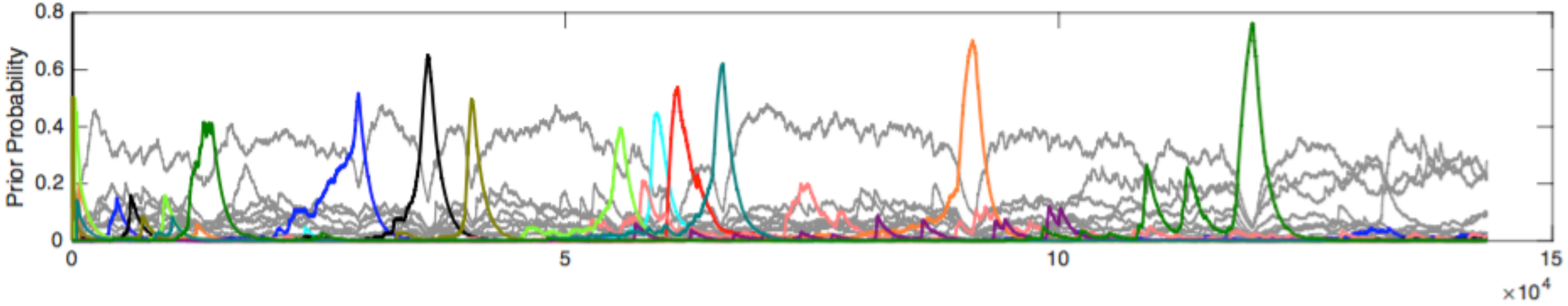
Learner 1

$\hat{y}$

(x, y)

Characteristics of Tweet Topic Classification

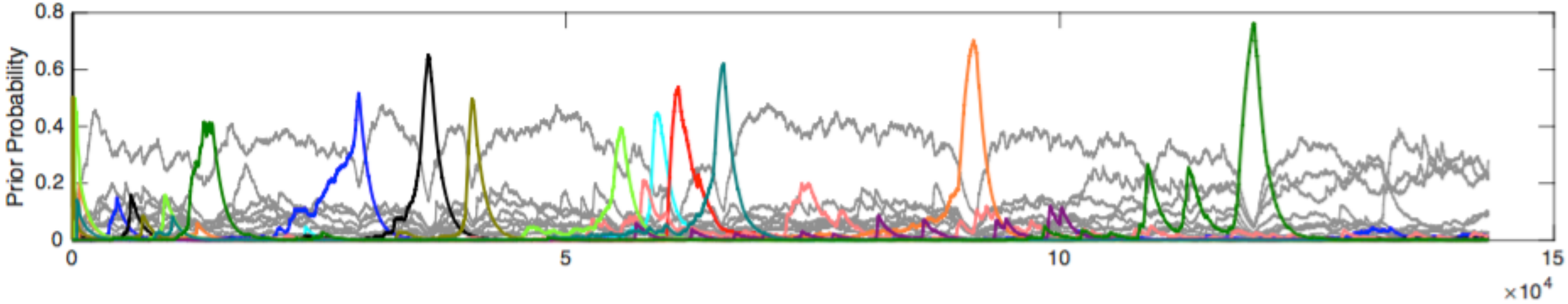
- Online problem: feedback that generates supervised samples is potentially instantaneous.
- Class imbalance.
- Concept drifts may affect p(y|**x**), though not so common.



Y. Sun, K. Tang, L. Minku, S. Wang and X. Yao. "Online Ensemble Learning of Data Streams with Gradually Evolved Classes", *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1532-1545, 2016.

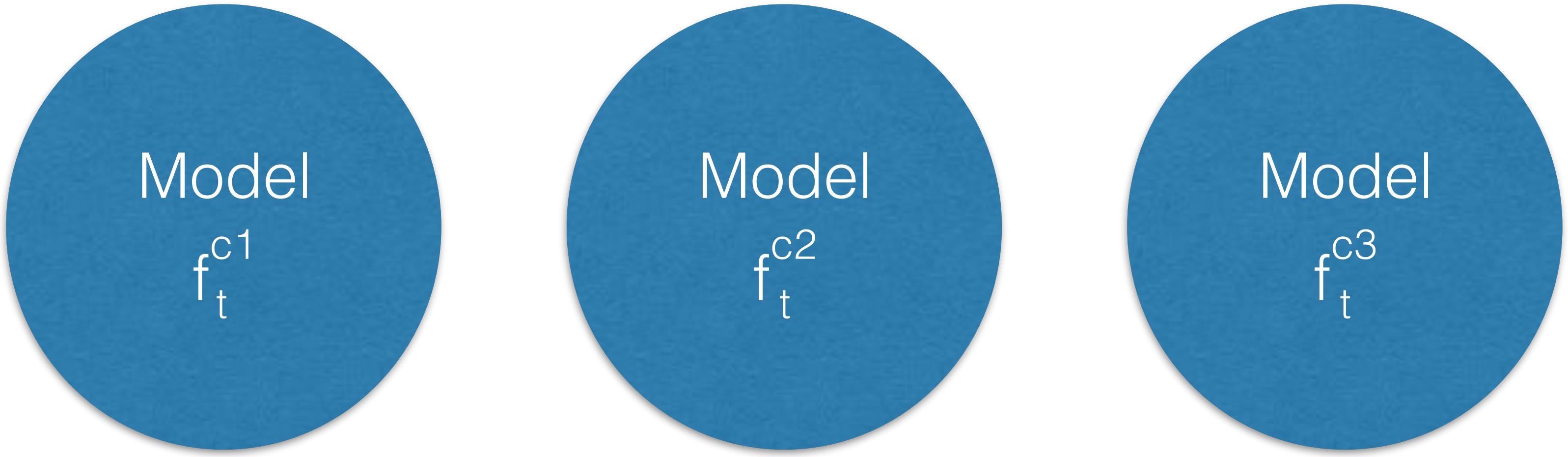
Characteristics of Tweet Topic Classification

- Gradual concept drifts affecting p(y) are very common.
- Gradual class evolution.
  - Recurrence is different from recurrent concepts, as it does not mean that a whole concept reoccurs.



Y. Sun, K. Tang, L. Minku, S. Wang and X. Yao. “Online Ensemble Learning of Data Streams with Gradually Evolved Classes”, *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1532-1545, 2016.

Class-Based Ensemble for Class Evolution (CBCE)



Model f^{c1}_t

Model f^{c2}_t

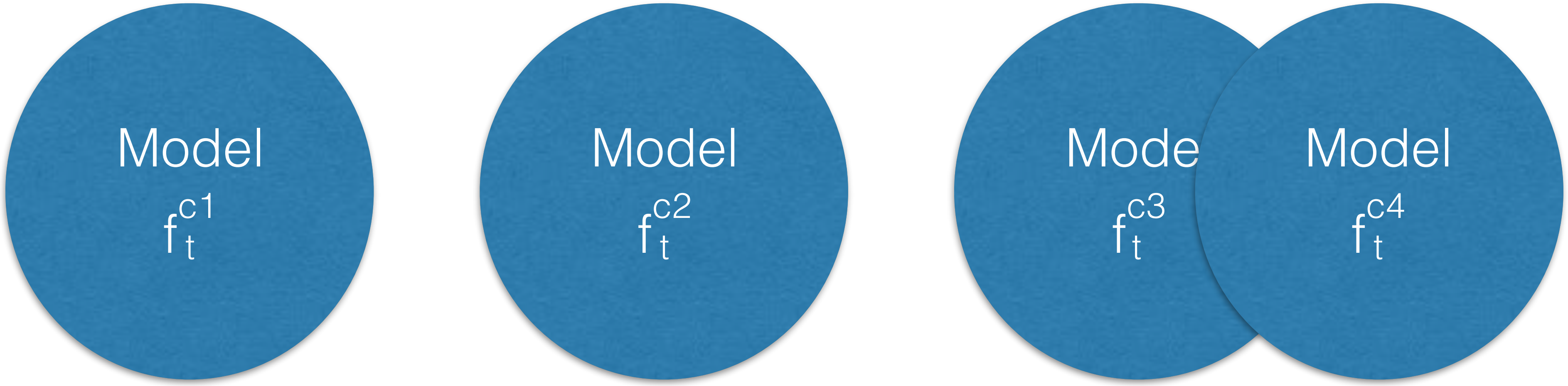
Model f^{c3}_t

- Each base model is a binary classifier which implements the one-versus-all strategy.
  - Class represented by the model is the positive +1 class.
  - All other classes compose the negative -1 class.

- The class c_i predicted by the ensemble is the class with maximum likelihood $p(\mathbf{x}|c_i)$.

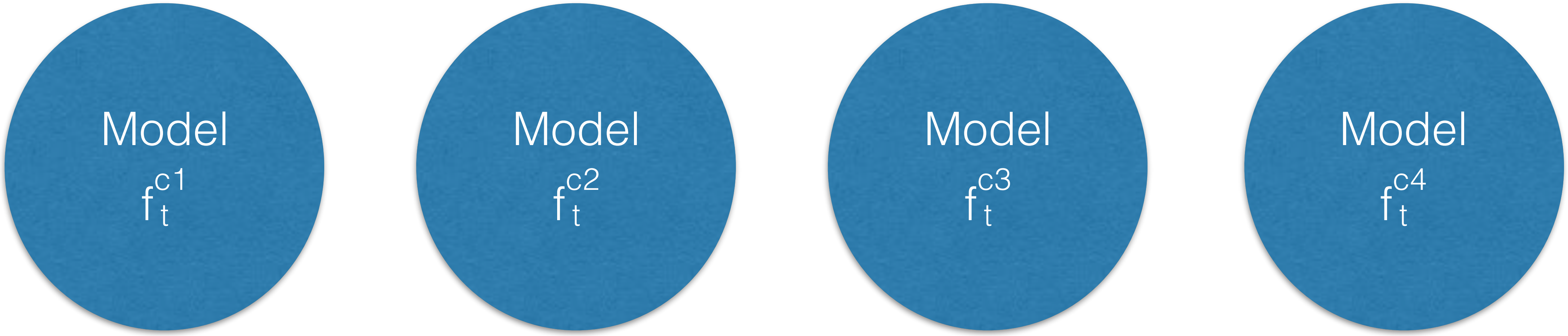
Dealing with Class Evolution

- The use of one base model for each class is a natural way of dealing with class emergence, disappearance and reoccurrence.



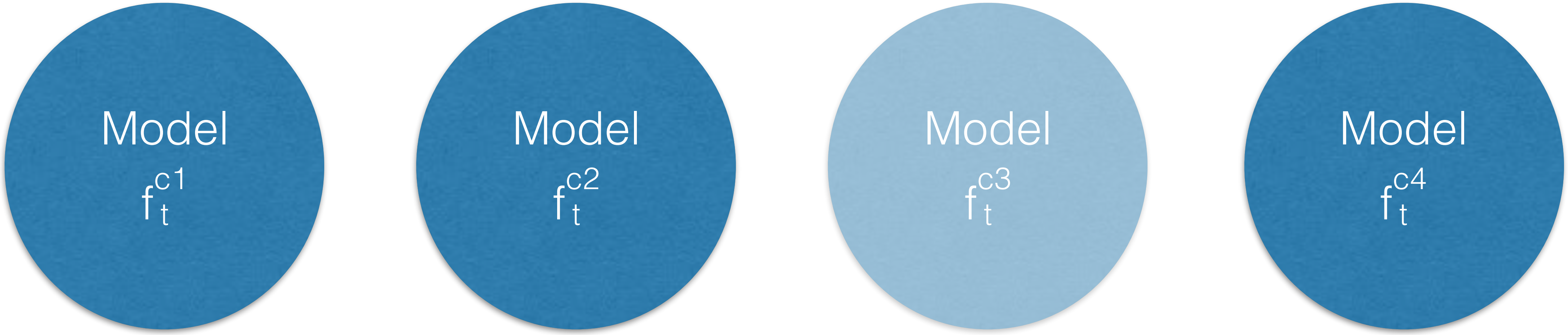
Dealing with Class Evolution

- The use of one base model for each class is a natural way of dealing with class emergence, disappearance and reoccurrence.



Dealing with Class Evolution

- The use of one base model for each class is a natural way of dealing with class emergence, disappearance and reoccurrence.



Dealing with Concept Drifts on p(y) and Class Imbalance

- Tracks proportion of examples of each class over time as OOB and UOB to deal with gradual concept drifts on p(y).
    - If a given class becomes too small, it is considered to have disappeared.
- Given the one-versus-all strategy, the positive classes are likely to be the minorities for each model.
    - Undersampling of negative examples for training when they are majority.

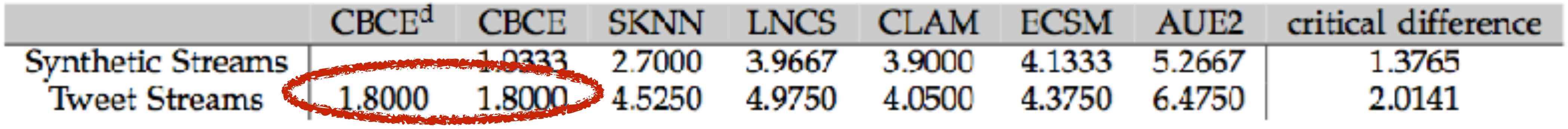
Dealing with Concept Drifts on p(y|**x**)

- DDM monitoring error of ensemble.
- Reset whole ensemble upon drift detection.

All these strategies are online, if the base learner is online.

Sample Results Using Online Kernelized Logistic Regression as Base Learner

Friedman Test (Nemenyi test at $\alpha = 0.05$) Result Considering the G-mean Values in Synthetic & Tweet Streams

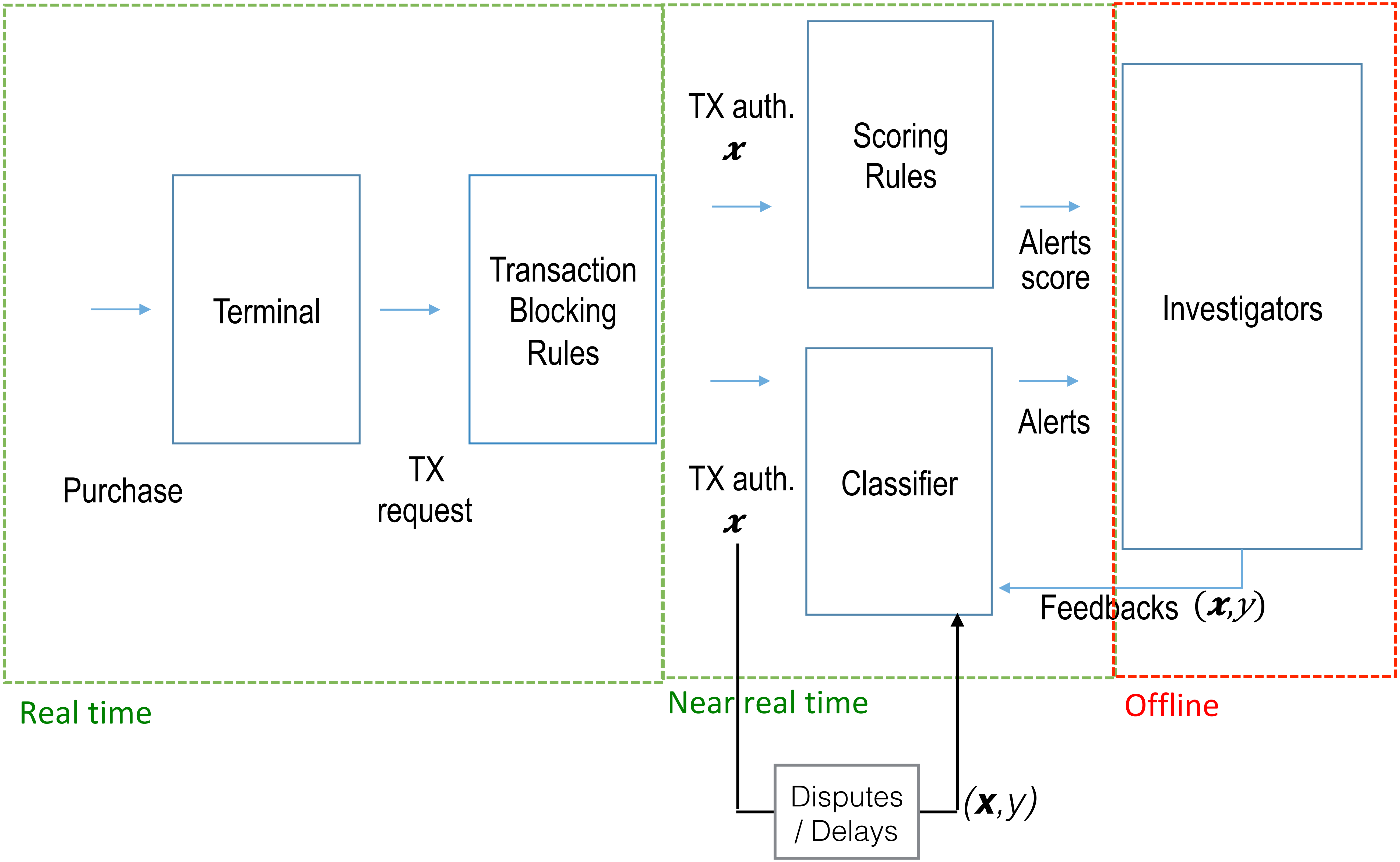


	CBCE^d	CBCE	SKNN	LNCS	CLAM	ECSM	AUE2	critical difference
Synthetic Streams		1.9333	2.7000	3.9667	3.9000	4.1333	5.2667	1.3765
Tweet Streams	1.8000	1.8000	4.5250	4.9750	4.0500	4.3750	6.4750	2.0141

CBCE outperformed the other approaches across data streams in terms of overall G-mean.

For some twitter data streams, DDM helped and for some it did not help.

The Fraud Detection Pipeline

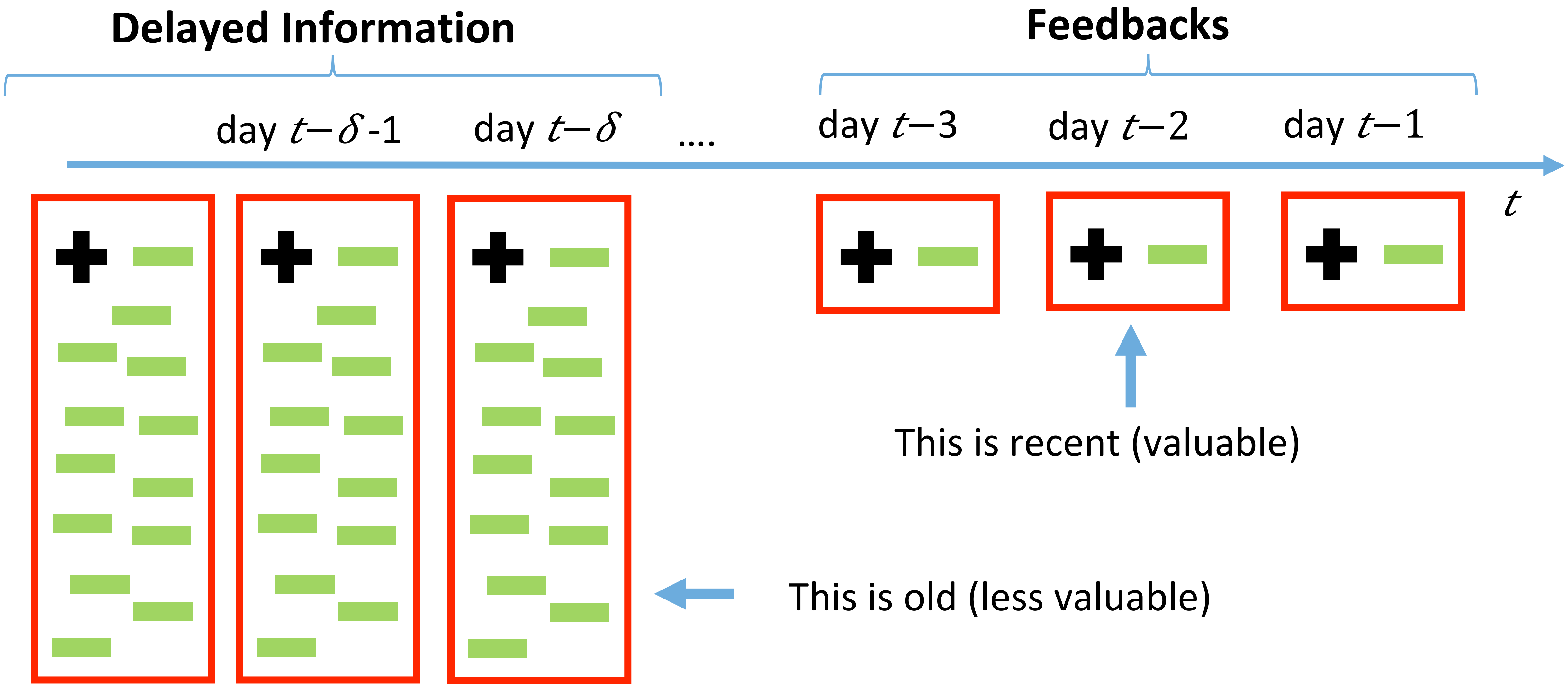


Characteristics of Fraud Detection Learning Systems

- Class imbalance (~0.2% of transactions are frauds).
- Concept drift may happen (customer habits may change, fraud strategies may change).
- Supervised information has a selection bias (feedback samples are transactions more likely to be fraud than the delayed transactions).
- Most supervised information arrives with a considerable delay (verification latency).

A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi. "Credit Card Fraud Detection: a Realistic Modeling and a Novel Learning Strategy", *IEEE Transactions on Neural Networks and Learning Systems,* 2017 (in press).

Characteristics of Fraud Detection Learning Systems



Delayed Information

Feedbacks

day $t-\delta$-1

day $t-\delta$

....

day $t-3$

day $t-2$

day $t-1$

t

This is recent (valuable)

This is old (less valuable)

Learning-Based Solutions for Fraud Detection

Rationale: “Feedback and delayed samples are different in nature and should be exploited differently”

Two types of learners:

- Learn examples created from investigators’ feedback:

$$\mathcal{F}_t = \text{TRAIN}\left(\{\text{F}_t, \dots, \text{F}_{t-(Q-1)}\}\right)$$

- Learn examples with delayed labels.

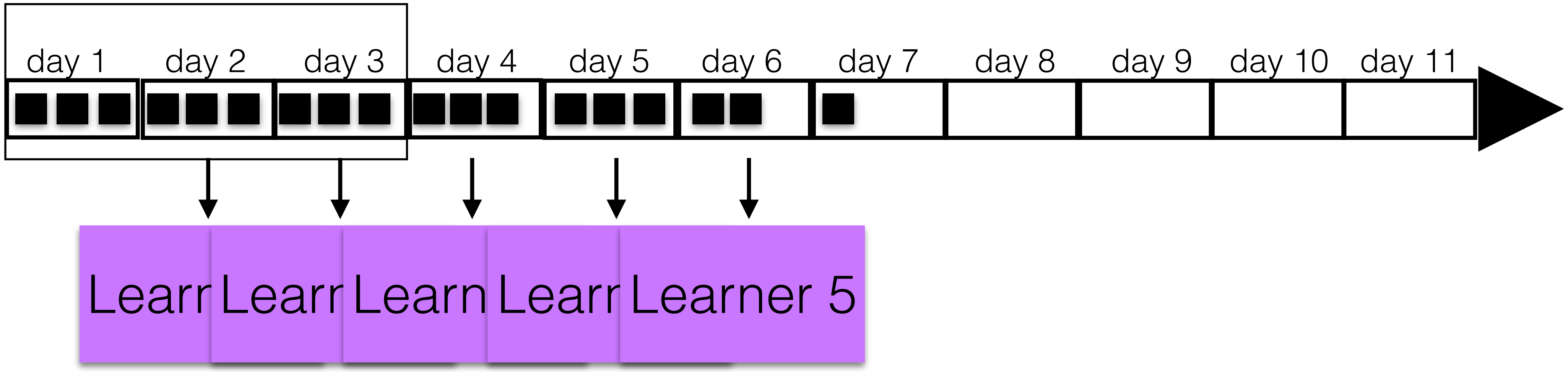
$$\mathcal{D}_t = \text{TRAIN}\left(\{\text{D}_{t-\delta}, \dots, \text{D}_{t-(\delta+M-1)}\}\right)$$

Combination rule:

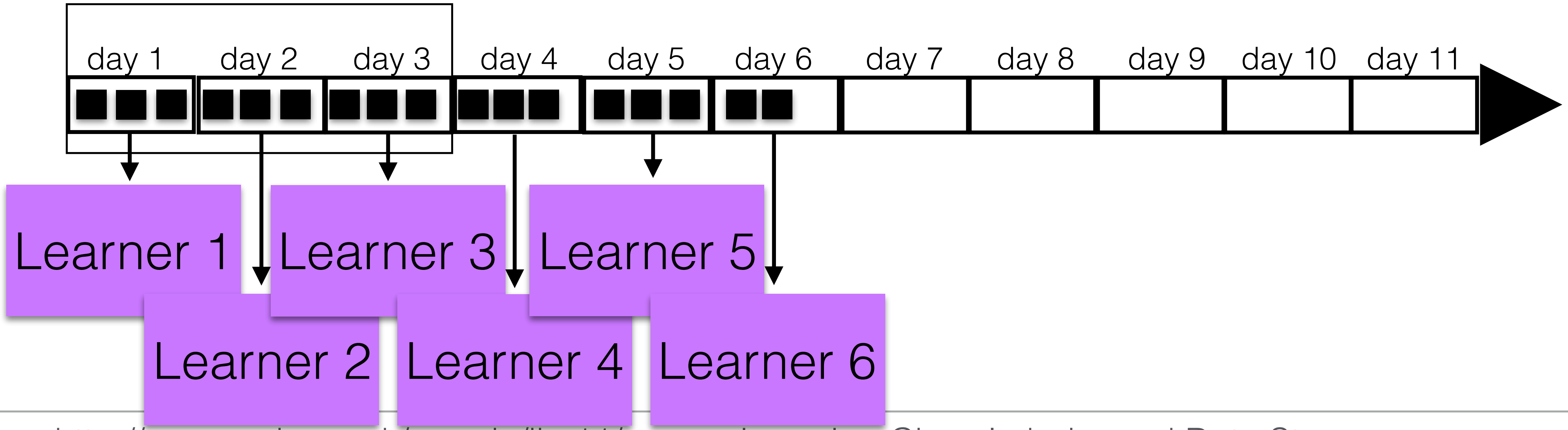
$$\mathcal{P}_{\mathcal{A}_t}(+|x) = \alpha \mathcal{P}_{\mathcal{F}_t}(+|x) + (1-\alpha)\mathcal{P}_{\mathcal{D}_t}(+|x)$$

Adaptation Strategies for Delayed Data

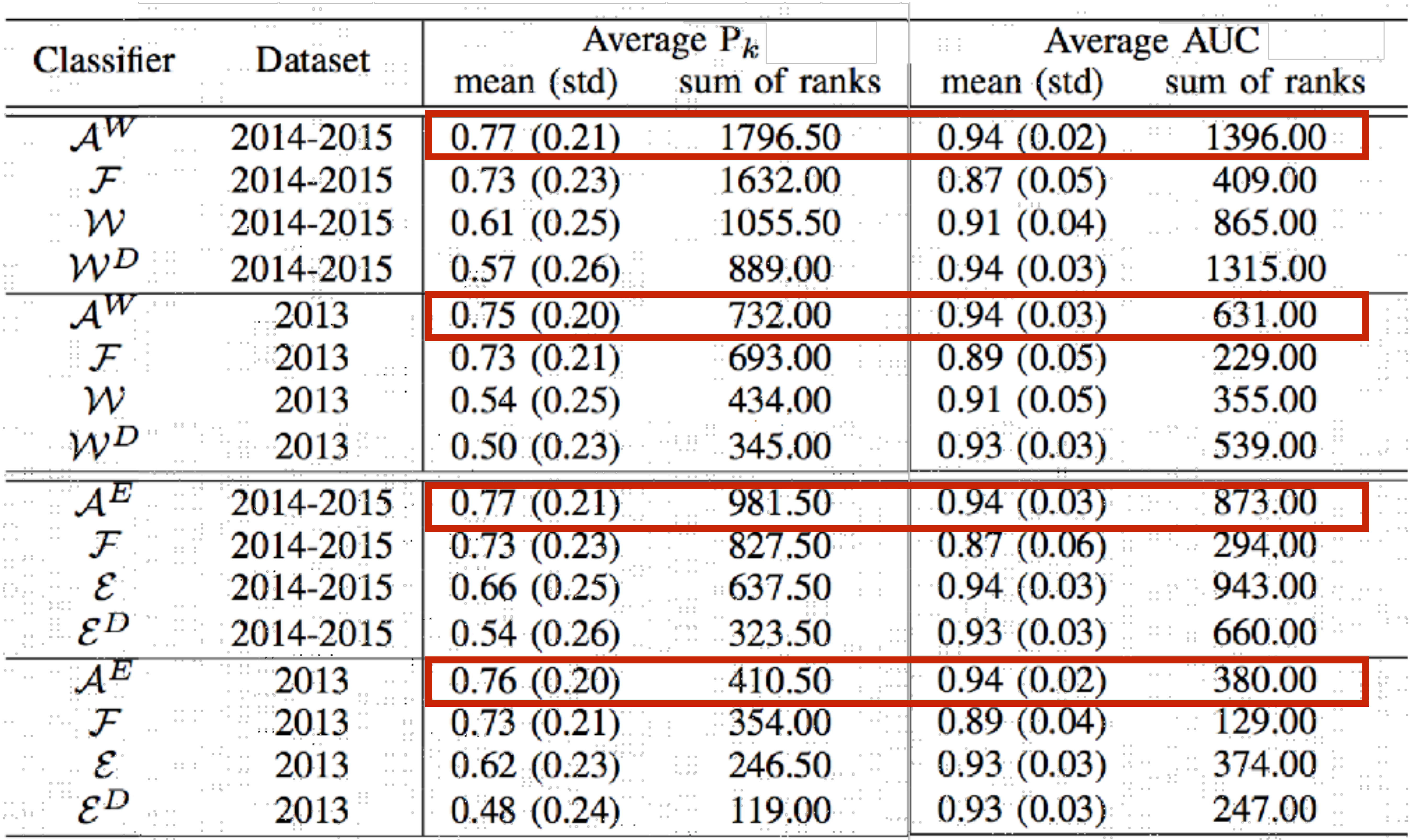
- Sliding windows:



- Ensemble



Sample Results Using Random Forest as Base Learner



Classifier	Dataset	Average P_k		Average AUC	
		mean (std)	sum of ranks	mean (std)	sum of ranks
$\mathcal{A}^W$	2014-2015	0.77 (0.21)	1796.50	0.94 (0.02)	1396.00
$\mathcal{F}$	2014-2015	0.73 (0.23)	1632.00	0.87 (0.05)	409.00
$\mathcal{W}$	2014-2015	0.61 (0.25)	1055.50	0.91 (0.04)	865.00
$\mathcal{W}^D$	2014-2015	0.57 (0.26)	889.00	0.94 (0.03)	1315.00
$\mathcal{A}^W$	2013	0.75 (0.20)	732.00	0.94 (0.03)	631.00
$\mathcal{F}$	2013	0.73 (0.21)	693.00	0.89 (0.05)	229.00
$\mathcal{W}$	2013	0.54 (0.25)	434.00	0.91 (0.05)	355.00
$\mathcal{W}^D$	2013	0.50 (0.23)	345.00	0.93 (0.03)	539.00
$\mathcal{A}^E$	2014-2015	0.77 (0.21)	981.50	0.94 (0.03)	873.00
$\mathcal{F}$	2014-2015	0.73 (0.23)	827.50	0.87 (0.06)	294.00
$\mathcal{E}$	2014-2015	0.66 (0.25)	637.50	0.94 (0.03)	943.00
$\mathcal{E}^D$	2014-2015	0.54 (0.26)	323.50	0.93 (0.03)	660.00
$\mathcal{A}^E$	2013	0.76 (0.20)	410.50	0.94 (0.02)	380.00
$\mathcal{F}$	2013	0.73 (0.21)	354.00	0.89 (0.04)	129.00
$\mathcal{E}$	2013	0.62 (0.23)	246.50	0.93 (0.03)	374.00
$\mathcal{E}^D$	2013	0.48 (0.24)	119.00	0.93 (0.03)	247.00

Outline

- Background and motivation
- Problem formulation
- Challenges and core techniques
- Online approaches for learning class imbalanced data streams
- Chunk-based approaches for learning class imbalanced data streams
- Performance assessment
- Two real world problems
- Remarks and next challenges

Remarks and Next Challenges

- Overview of core techniques to deal with challenges posed by data streams.

- Learning class imbalanced data streams require a combination of several different core techniques to be used.

- Each technique has potential advantages and disadvantages based on the application to be tackled.

- Still, there are several challenges requiring more attention, when adopting more realistic scenarios, e.g.:
  - Class evolution.
  - Scarce supervised information.
  - Large delays in supervised information (verification latency).
  - Biased samples.

- Not many datasets are available in realistic conditions.