Czestochowa University of Technology,
Institute of Computational Intelligence
Poland

Data Stream Mining

Leszek Rutkowski

WCCI’ 2016 Tutorial, Vancouver, July 24, 2016

Data stream mining

in cooperation with

Piotr Duda, Maciej Jaworski, Lena Pietruczuk

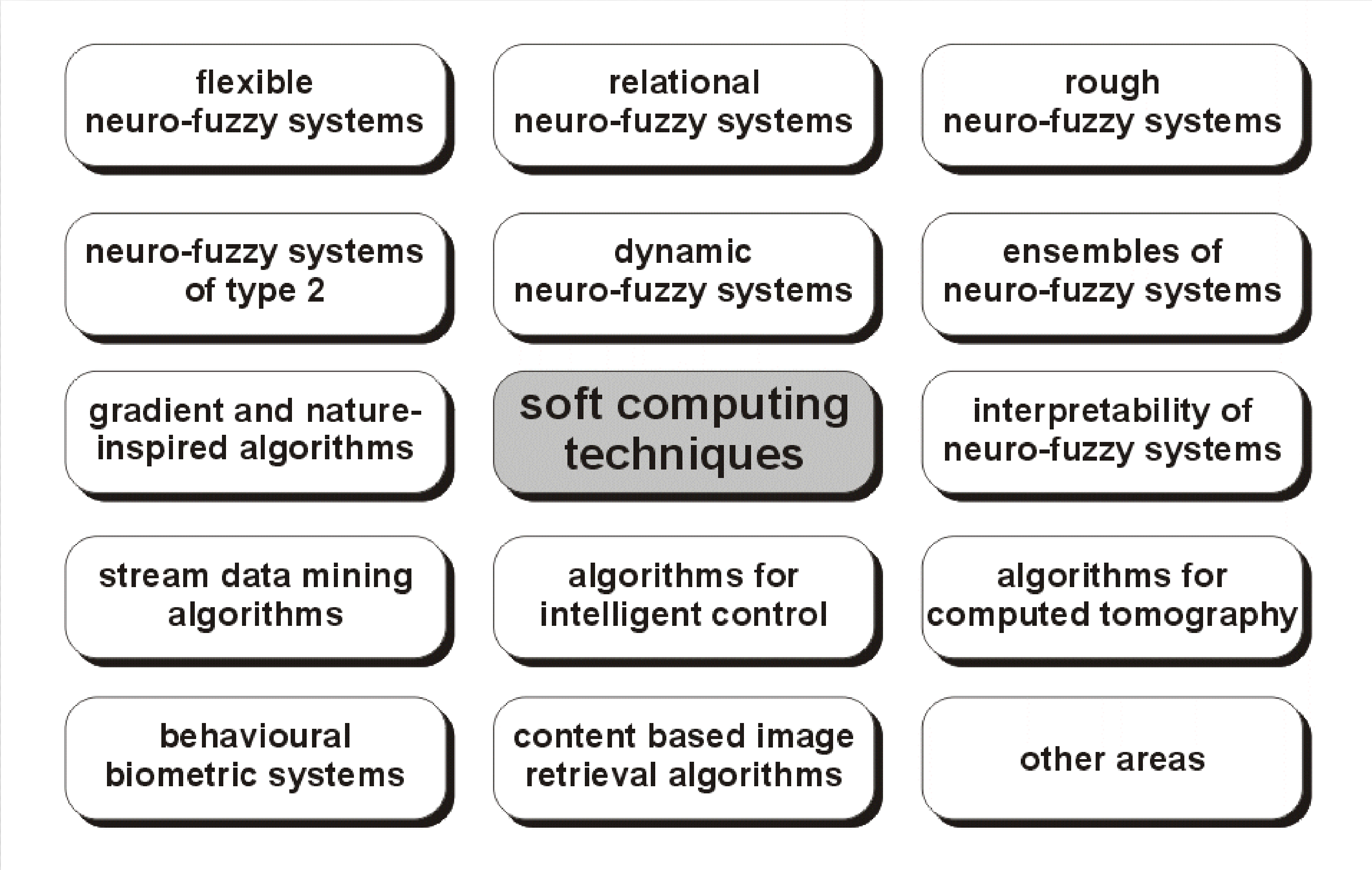
Czestochowa Univeristy of Technology, Poland

Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Overview of the research in the Institute of Computational Intelligence at CUT



flexible neuro-fuzzy systems

relational neuro-fuzzy systems

rough neuro-fuzzy systems

neuro-fuzzy systems of type 2

dynamic neuro-fuzzy systems

ensembles of neuro-fuzzy systems

gradient and nature-inspired algorithms

soft computing techniques

interpretability of neuro-fuzzy systems

stream data mining algorithms

algorithms for intelligent control

algorithms for computed tomography

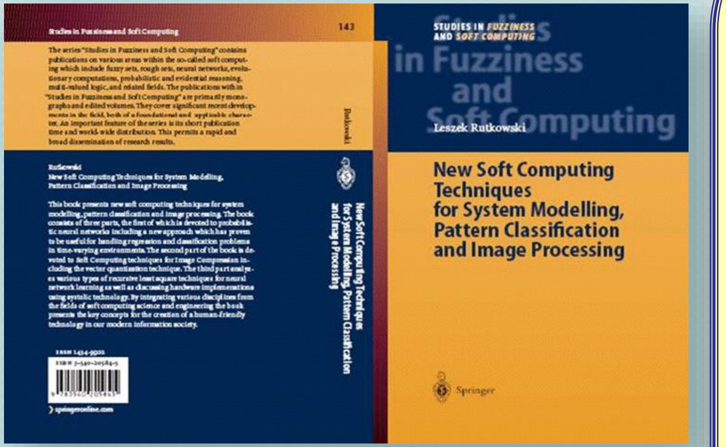
behavioural biometric systems

content based image retrieval algorithms

other areas

Overview of the research in the Institute of Computational Intelligence at CUT

Rutkowski L., New Soft Computing Techniques for System Modelling, Pattern Classification and Image Processing, Springer, 2004.



Studies in Fuzziness and Soft Computing 143

The series "Studies in Fuzziness and Soft Computing" contains publications on various areas within the so-called soft computing which include fuzzy sets, rough sets, neural networks, evolutionary computations, probabilistic and evidential reasoning, multi-valued logic, and related fields. The publications within "Studies in Fuzziness and Soft Computing" are primarily monographs and edited volumes. They cover significant recent developments in the field, both of a foundational and applicable character. An important feature of the series is its short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

Rutkowski
New Soft Computing Techniques for System Modelling, Pattern Classification and Image Processing

This book presents new soft computing techniques for system modelling, pattern classification and image processing. The book consists of three parts, the first of which is devoted to probabilistic neural networks including a new approach which has proven to be useful for handling regression and classification problems in time-varying environments. The second part of the book is devoted to Soft Computing techniques for Image Compression including the vector quantization technique. The third part analyses various types of recursive least square techniques for neural network learning as well as discussing hardware implementations using systolic technology. By integrating various disciplines from the fields of soft computing science and engineering the book presents the key concepts for the creation of a human-friendly technology in our modern information society.

ISSN 1434-9922
ISBN 3-540-20584-5

› springeronline.com

Rutkowski

New Soft Computing Techniques for System Modelling, Pattern Classification and Image Processing

STUDIES IN FUZZINESS AND SOFT COMPUTING

Studies in Fuzziness and Soft Computing

Leszek Rutkowski

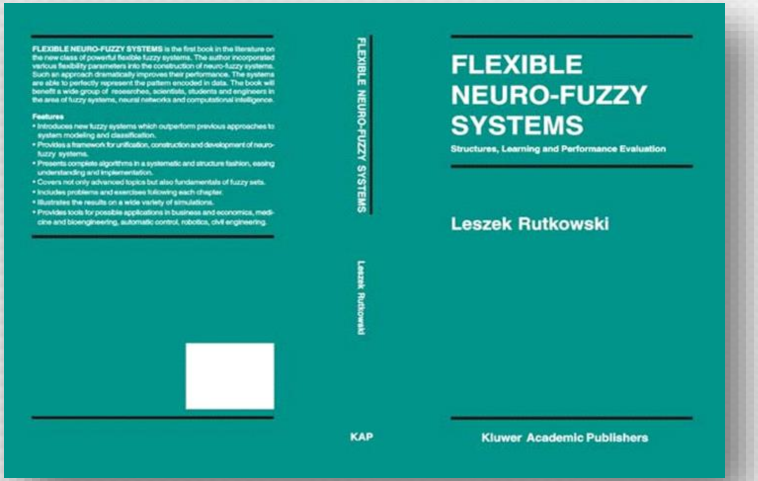
New Soft Computing Techniques for System Modelling, Pattern Classification and Image Processing

Springer

1. (1) Probabilistic neural networks to solve various problems of system modelling and classification in a non-stationary environment.
2. (2) Image compression methods.
3. (3) RLS learning algorithms for multilayer neural networks.
4. (4) Systolic architectures of multilayer neural networks.

Overview of the research in the Institute of Computational Intelligence at CUT

Rutkowski L., Flexible Neuro-Fuzzy Systems. Structures, learning and performance evaluation, Kluwer, 2004.



FLEXIBLE NEURO-FUZZY SYSTEMS is the first book in the literature on the new class of powerful flexible fuzzy systems. The author incorporated various flexibility parameters into the construction of neuro-fuzzy systems. Such an approach dramatically improves their performance. The systems are able to perfectly represent the pattern encoded in data. The book will benefit a wide group of researches, scientists, students and engineers in the area of fuzzy systems, neural networks and computational intelligence.

Features

- Introduces new fuzzy systems which outperform previous approaches to system modeling and classification.
- Provides a framework for unification, construction and development of neuro-fuzzy systems.
- Presents complete algorithms in a systematic and structure fashion, easing understanding and implementation.
- Covers not only advanced topics but also fundamentals of fuzzy sets.
- Includes problems and exercises following each chapter.
- Illustrates the results on a wide variety of simulations.
- Provides tools for possible applications in business and economics, medicine and bioengineering, automatic control, robotics, civil engineering.

FLEXIBLE NEURO-FUZZY SYSTEMS

Leszek Rutkowski

KAP

FLEXIBLE
NEURO-FUZZY
SYSTEMS
Structures, Learning and Performance Evaluation

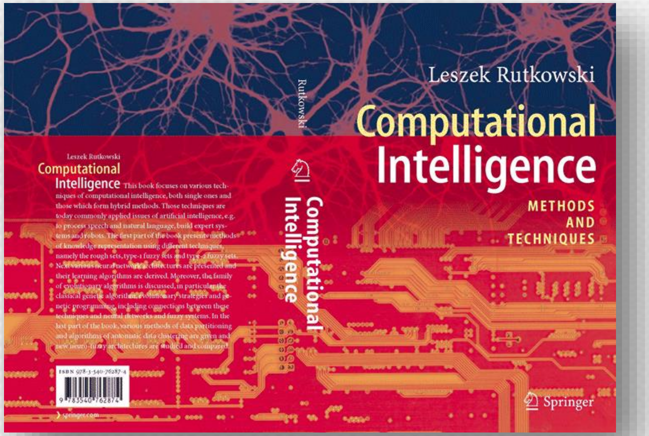
Leszek Rutkowski

Kluwer Academic Publishers

(1) New fuzzy systems which outperform previous approaches to system modelling and classification.
(2) Framework for unification, construction and development of neuro-fuzzy systems.
(3) Complete algorithms in a systematic and structured fashion, easing understanding and implementation.

Overview of the research in the Institute of Computational Intelligence at CUT

Rutkowski L., Computational intelligence,
Springer, 2008.



(1) Selected issues of artificial intelligence.
(2) Methods of knowledge representation using rough sets.
(3) Methods of knowledge representation using type-1 and type-2 fuzzy sets.
(4) Neural networks and their learning algorithms.
(5) Evolutionary algorithms.
(6) Data clustering methods.
(7) Neuro-fuzzy systems of Mamdani, logical and Takagi-Sugeno type.
(8) Flexible neuro-fuzzy systems.

Overview of the research in the Institute of Computational Intelligence at CUT

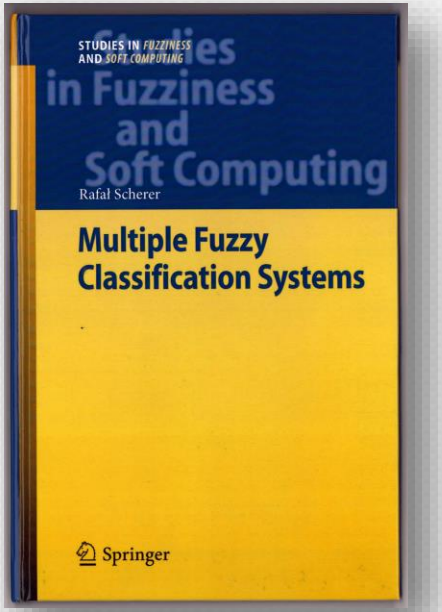
Rutkowski L., Duda P., Jaworski M., Pietruczuk L., Stream Data Mining: Algorithms and Their Probabilistic Properties, Studies in Big Data, Springer, 2017.

This book shows methods and algorithms which are mathematically justified.

(1) It shows how to adopt the static decision trees, like ID.3 or CART, to deal with data streams.
(2) A new technique, based on the McDiarmid bound, is developed.
(3) New decision trees are designed, by a proper combination of the Gini index and misclassification error impurity measures, leading to the original concept of the hybrid decision trees.
(4) The problem of designing ensembles and automatic choosing their sizes is described and solved.
(5) Nonparametric techniques based on the Parzen – kernels and orthogonal series, are adopted to deal with concept drift in the problem of non-stationary regressions and classification in time-varying environment. Nonparametric procedures are developed and their probabilistic properties are investigated.

Overview of the research in the Institute of Computational Intelligence at CUT

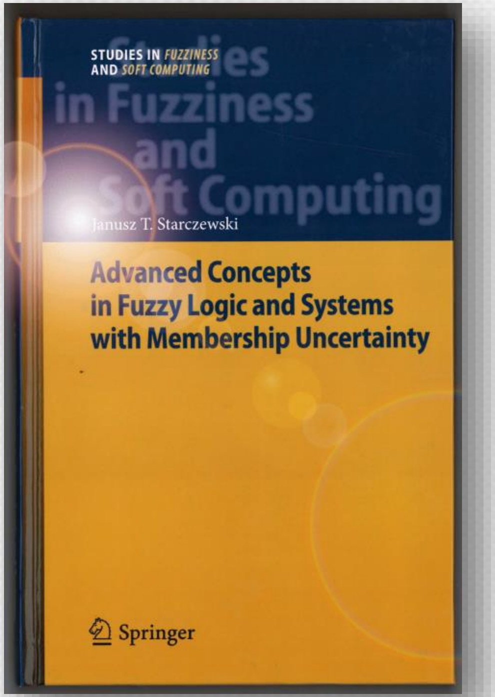
Scherer R., Multiple Fuzzy Classification Systems,
Springer, 2012.



(1) Ensemble techniques (bagging, boosting, negative correlation learning etc.).
(2) Relational modular fuzzy systems.
(3) Ensembles of the Mamdani, logical and Takagi-Sugeno fuzzy systems.
(4) Rough–neuro–fuzzy ensembles for classification with missing data.

Overview of the research in the Institute of Computational Intelligence at CUT

Starczewski J., Fuzzy Logic and Systems
with Membership Uncertainty, Springer, 2013.


STUDIES IN FUZZINESS AND SOFT COMPUTING

Janusz T. Starczewski

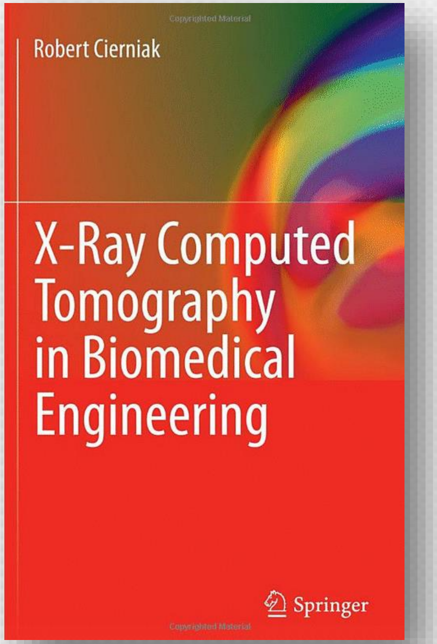
Advanced Concepts in Fuzzy Logic and Systems with Membership Uncertainty

Springer

1. Algebraic operations on fuzzy valued fuzzy sets.
2. Defuzzification of uncertain fuzzy sets. Generalized uncertain fuzzy logic systems.
3. Uncertainty generation in uncertain fuzzy logic systems.
4. Designing uncertain fuzzy logic systems.

Overview of the research in the Institute of Computational Intelligence at CUT

Cierniak R., X-Ray Computed Tomography in Biomedical Engineering, Springer, 2011



Copyrighted Material

Robert Cierniak

X-Ray Computed Tomography in Biomedical Engineering

Springer

Copyrighted Material

(1) Discusses X-ray CT tomography from a historical point of view, the design and physical operating principles of computed tomography apparatus, the algorithms
of image reconstruction and the quality assessment criteria of tomography scanners.

(2) Algorithms of image reconstruction from projections, a crucial problem in medical imaging, are considered in depth.

Data stream mining

- content

- **Data streams – introduction to the topic**
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Stream Data

Stream of data:

- huge volumes of continuous data,
- possibly infinite,
- multidimensional features,
- often fast changing,
- requiring fast, real-time responses.

Examples of data streams

- *Telecommunication calling records,*
- *Credit card transaction flows,*
- *Network monitoring and traffic engineering,*
- *Financial market,*
- *Audio and video recording of various processes,*
- *Computer network information flow,*
- *Web logs and Web page click streams,*
- *Satelite data flow.*

Stream Data

Static data	Stream of data
Fixed number of data elements (unlimited memory usage)	Potentially infinite number of data elements (memory limitation problem)
Stationary distribution of data	Changing data distribution (concept drift)
All data available at any time – multiple passes of data	One pass of data
Unlimited processing time	Processing time depends on rate of incoming data

Typical problems in data mining

$X_i = [x_i^1, \ldots, x_i^d, y_i] = [\boldsymbol{x}_i, y_i]$, $x_i^j \in A_j, y_i \in B$

$A_1 \times \cdots \times A_d$ - space of attributes values

$X_1, \ldots, X_N$ from probability distribution $\varrho(\boldsymbol{x}, y)$

Two types of supervised learning:

1) **Regression** - B continuous
2) **Classification** - B nominal

Typical problems in data mining

- Regression

$X_1, \ldots, X_N$ derived from $\varrho(\boldsymbol{x}, y), B \subset \mathbb{R}$

Aim: Construct a mapping function:

$$f\colon A_1 \times \cdots \times A_d \to B,$$

such that for any $\tilde{X} = [\tilde{\boldsymbol{x}}, \tilde{y}]$ from $\varrho(\boldsymbol{x}, y)$:

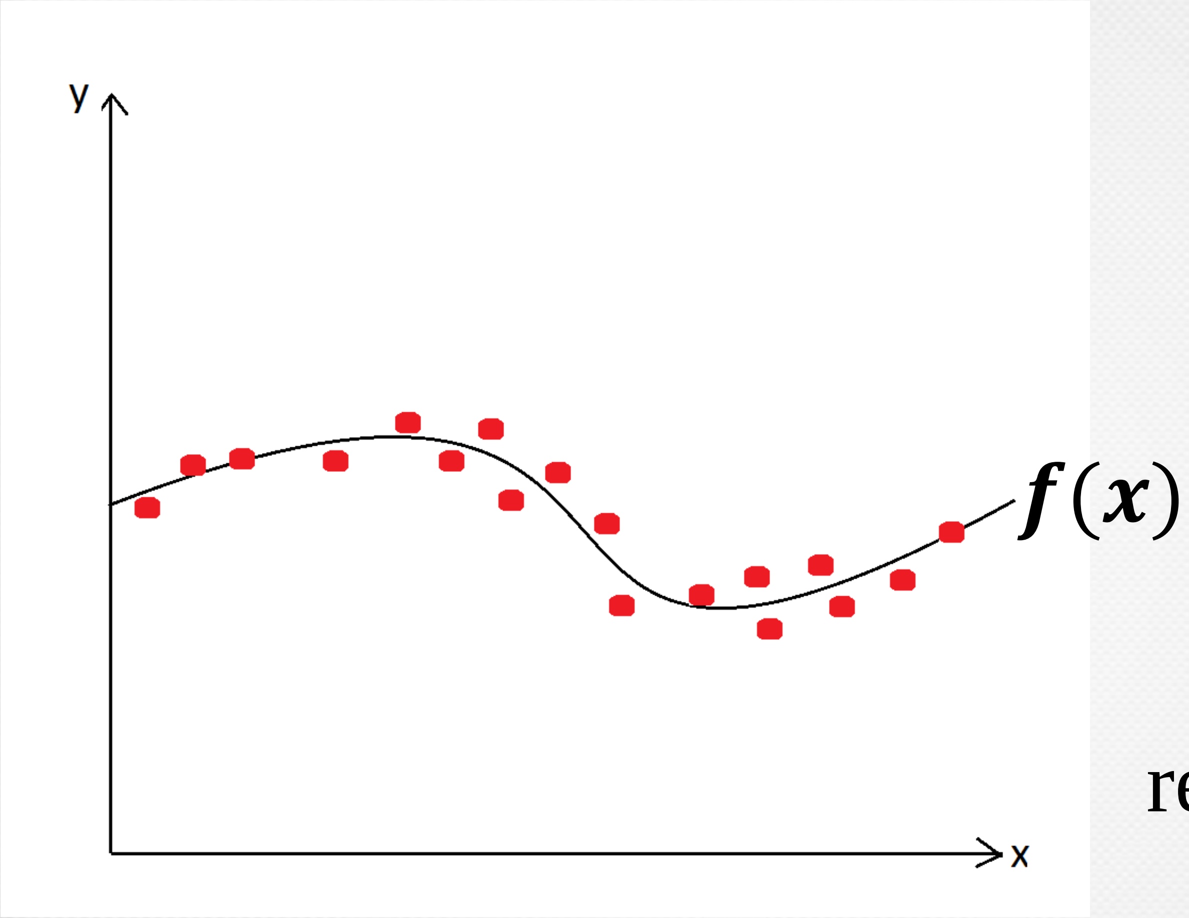
$$f = \underset{f^*}{\operatorname{argmin}}\{E[L(f^*(\tilde{\boldsymbol{x}}), \tilde{y})]\},$$

where L is a loss function, e.g.

$$L(f^*(\tilde{\boldsymbol{x}}), \tilde{y}) = (f^*(\tilde{\boldsymbol{x}}) - \tilde{y})^2$$

Typical problems in data mining

- Regression



one-dimensional regression problem

Typical problems in data mining

- Classification

$X_1, \ldots, X_N$ derived from $\varrho(\boldsymbol{x}, y)$, $B = \{C_1, \ldots C_K\}$

Aim: Construct a mapping function:

$$f\colon A_1 \times \cdots \times A_d \to B,$$

such that for any $\tilde{X} = [\tilde{\boldsymbol{x}}, \tilde{y}]$ from $\varrho(\boldsymbol{x}, y)$:

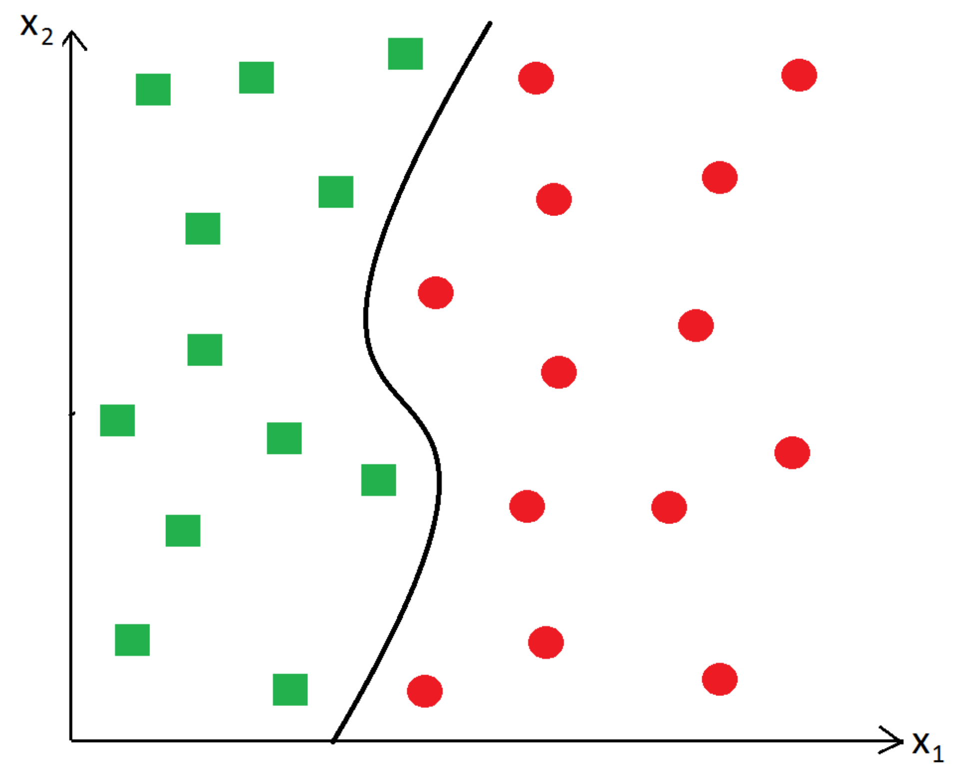
$$f = \underset{f^*}{\operatorname{argmin}}\{E[L(f^*(\tilde{\boldsymbol{x}}), \tilde{y})]\},$$

where $L(f^*(\tilde{\boldsymbol{x}}), \tilde{y}) = \begin{cases} 0, & f^*(\tilde{x}) = \tilde{y} \\ 1, & f^*(\tilde{x}) \neq \tilde{y} \end{cases}$

Supervised learning:

Classification

$$g(x) = 0$$



$$f(x) = \begin{cases} \text{●}, sgn(g(x)) \geq 0 \\ \text{■}, sgn(g(x)) < 0 \end{cases}$$

two-dimensional classification problem

Data stream mining

- content

- Data streams – introduction to the topic
- **Concept drift**
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Stream Data

- concept drift

CONCEPT DRIFT – it is a phenomenon describing the change in data distribution, character or their meaning, e.g. assigning e-mails to the ”spam” category.

Stream Data

- concept drift

In data streams the joint probability distribution $\varrho(\boldsymbol{x}, y)$ can vary over time:

X_1 from $\varrho_1(\boldsymbol{x}, y)$;
X_2 from $\varrho_2(\boldsymbol{x}, y)$;
…………………… ;
X_N from $\varrho_N(\boldsymbol{x}, y)$;

Stream Data

- concept drift

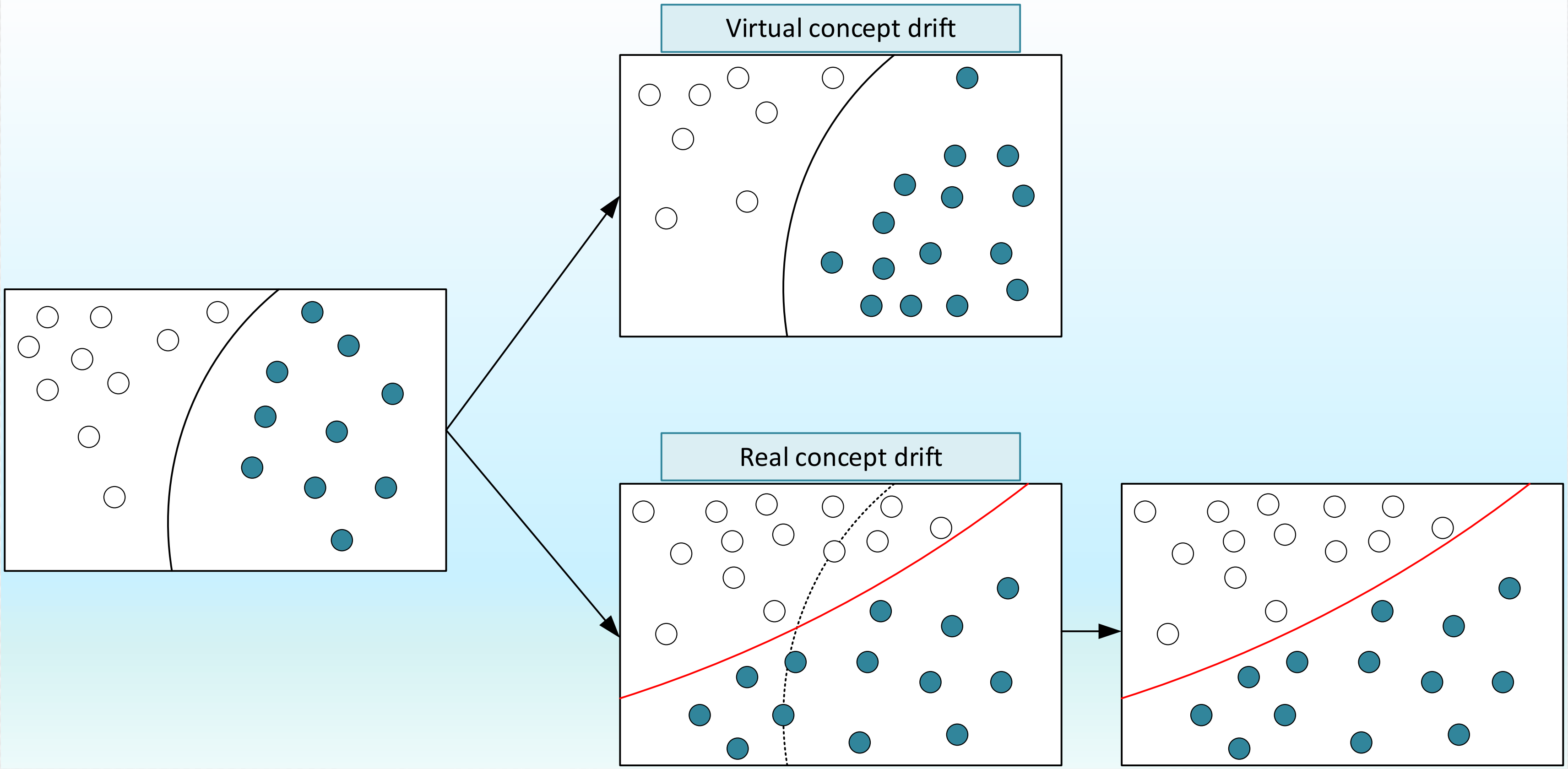
$\varrho_i(\boldsymbol{x}, y)$ - joint probability distribution

$$\varrho_i(\boldsymbol{x}, y) = \hat{\varrho}_i\ (y|\boldsymbol{x})\tilde{\varrho}_i(\boldsymbol{x})$$

Type of change:	**drift:**
$\hat{\varrho}_i\ (y\|\boldsymbol{x})$	real
$\tilde{\varrho}_i(\boldsymbol{x})$	virtual

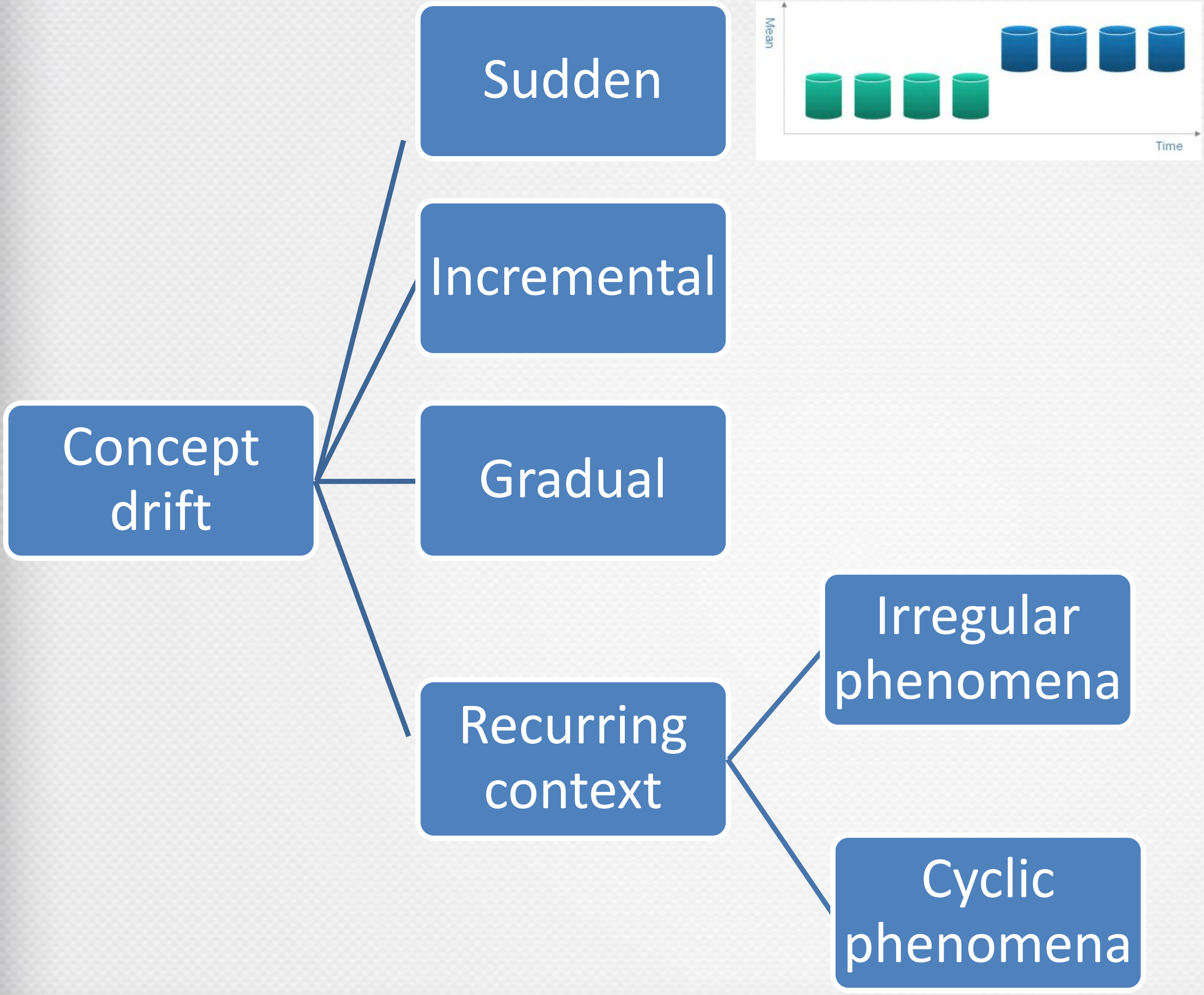
Stream Data

- concept drift



Stream Data

- concept drift



Concept drift

Sudden

Incremental

Gradual

Recurring context

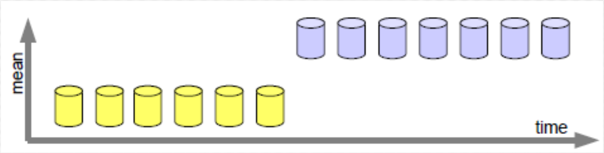
Irregular phenomena

Cyclic phenomena

Stream Data

- concept drift

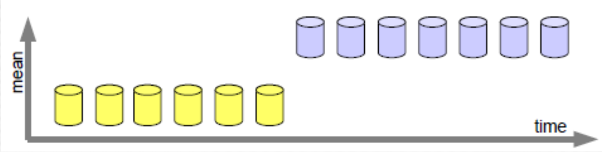
- Concept drift
  - Sudden
  - Incremental
  - Gradual
  - Recurring context
    - Irregular phenomena
    - Cyclic phenomena

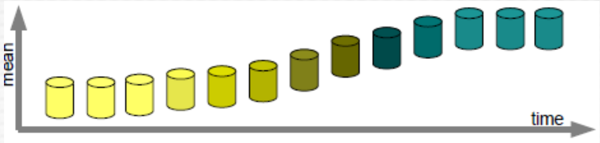


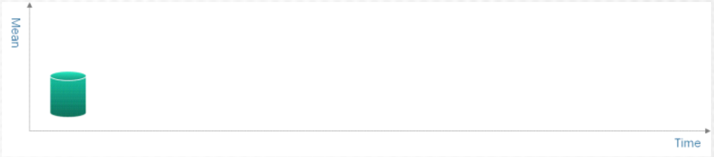
Stream Data

- concept drift

- Concept drift
  - Sudden
  - Incremental
  - Gradual
  - Recurring context
    - Irregular phenomena
    - Cyclic phenomena

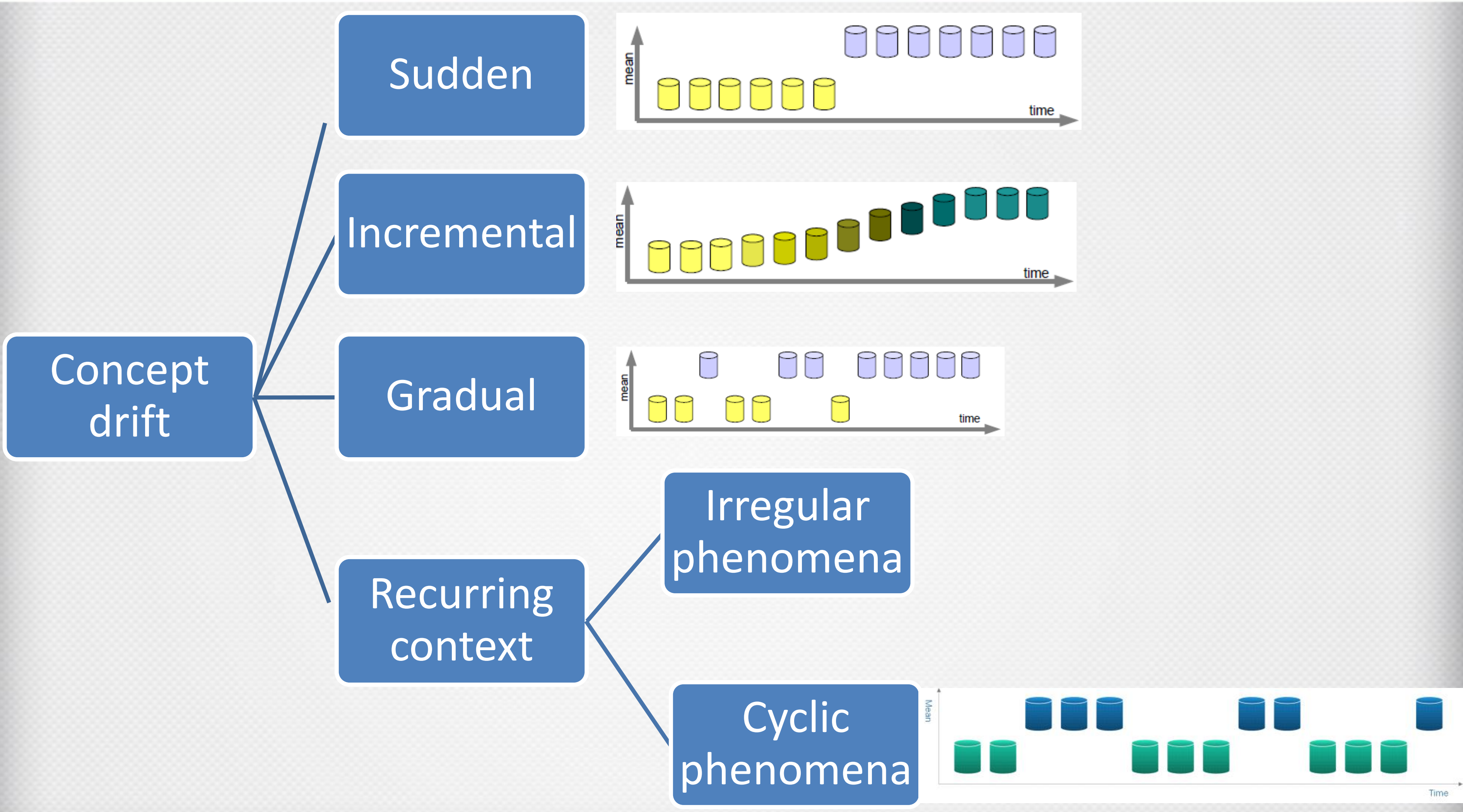






Stream Data

- concept drift



Concept drift

Sudden

Incremental

Gradual

Recurring context

Irregular phenomena

Cyclic phenomena

Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- **Various strategies of learning**
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Different kinds of learning*

(regarding time constraints and examples availability)

- Batch mode learning
- Incremental learning
- Online learning
- Anytime learning

* V. Lemaire, Ch. Slaperwyck, A. Bondu, *A Survey on Supervised Classification on Data Streams*, Lecture Notes in Business Information Processing, vol. 205, pp. 88-125, Srpinger, 2015

Batch mode learning

Batch mode learning:

consists of learning a model from a representative dataset which is fully available at the beginning of the learning stage. This type of algorithm is not appropriate for stream data.

Incremental learning

Incremental learning:

consists of receiving and integrating new examples without the need to perform a full learning phase from scratch. For any examples $X_1, X_2, \ldots,$ it generates the hypotheses $f_1, f_2, \ldots,$ such that f_{n+1}depends only on f_n and the current example x_{n+1}.

Online learning

Online learning:

an incremental learning for which the examples continuously arrive from data stream. The requirements in terms of time complexity are stronger than for the incremental learning. Concept drift must be managed by algorithms of this type.

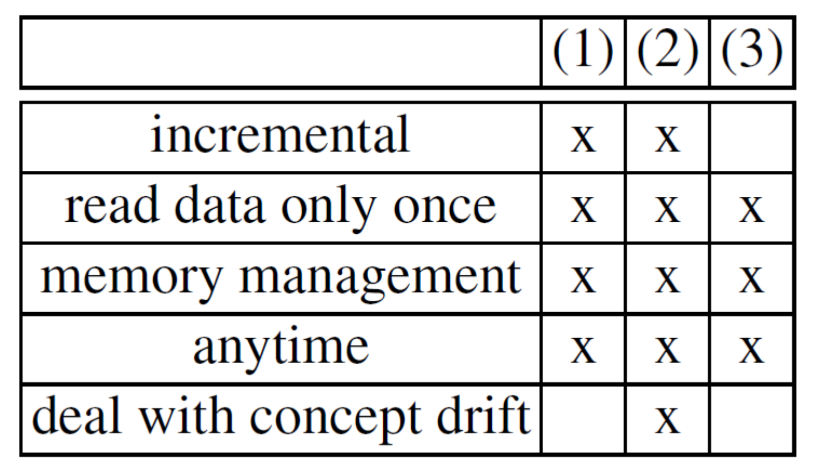
Anytime learning

Anytime learning:

algorithm is able to maximize the quality of the learned model (with respect to a particular evaluation criterion) until an interruption (which may be the arrival of a new example).

Learning on data streams

Desired properties*



	(1)	(2)	(3)
incremental	x	x	
read data only once	x	x	x
memory management	x	x	x
anytime	x	x	x
deal with concept drift		x	

(1) Fayyad, U.M. et al.: Advances in Knowledge Discovery and Data Mining. American Associacion for Artificial Intelligence, Menlo Park, CA, USA (1996);
(2) Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, NY, USA (2001) 97-106;
(3) Stonebraker, M. Çetintemel, U., Zdonik, S.: The 8 requrements of real-time stream processing. ACM SIGMOD Record 34(4) (December 2005) 42-47;
* **V. Lemaire, Ch. Slaperwyck, A. Bondu, *A Survey on Supervised Classification on Data Streams*, Lecture Notes in Business Information Processing, vol. 205, pp. 88-125, Srpinger, 2015**

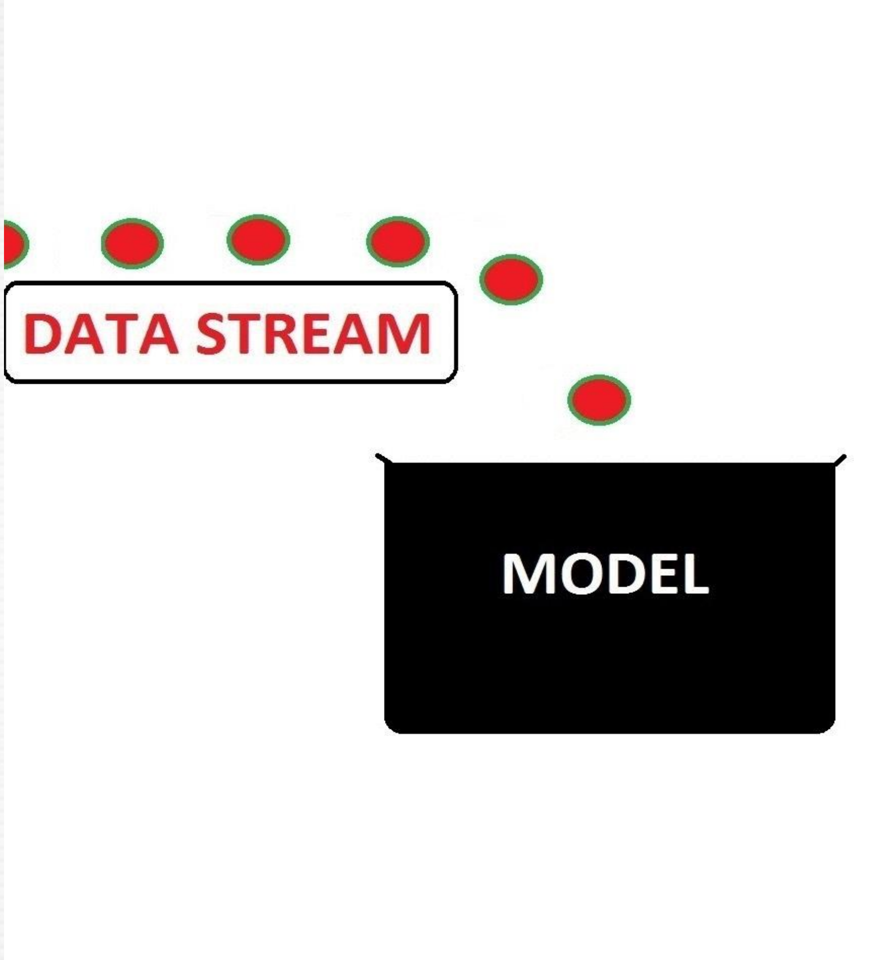
Data stream mining
- content

How to process stream data?

Online processing (instant incremental)

- The model (classifier) is updated after reading every single data element

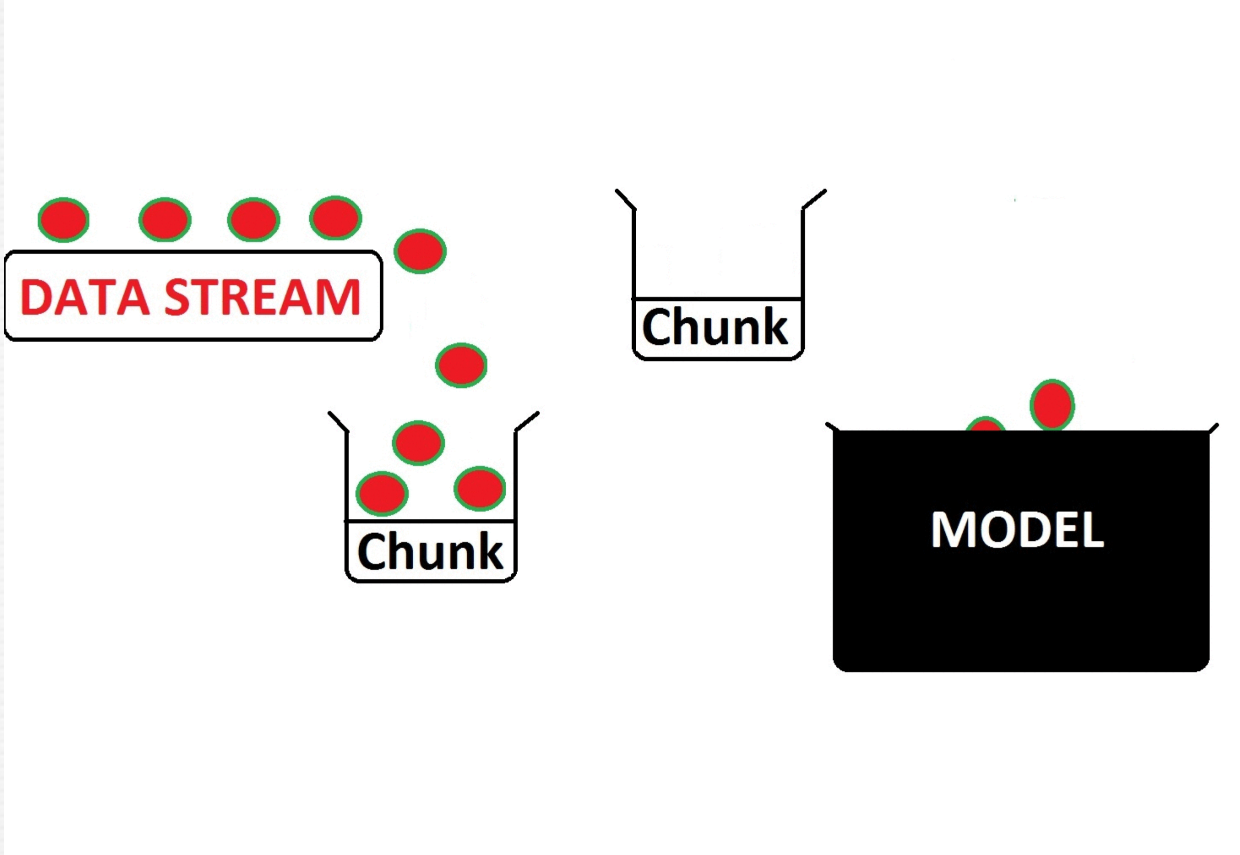
- Updating must be completed before the next data element arrives



How to process stream data?

Block (chunk) processing

- The model (classifier) is updated after reading every chunk of data
- Updating must be completed before the next chunk of data is collected



How to deal with concept drift?

How to deal with concept drift?

Sliding windows

Goal: limit the number of training examples to the most recent ones, consequently eliminate examples coming from an old concept

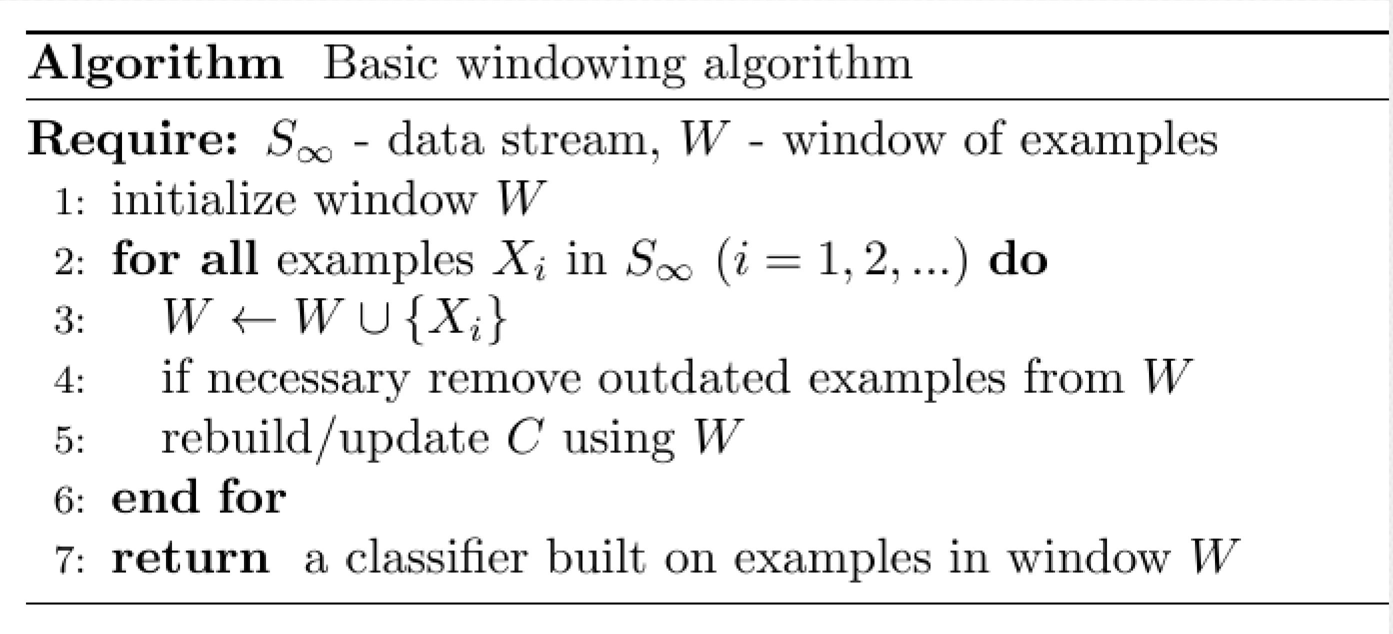
Two approaches:

- window of a fixed sized
  - classifier based on a small window reacts quickly to concept drifts but loses an accuracy in periods of stationary data
  - Classifier based on a large window fails to react quickly to concept drift
- window size adjusted heuristically (e.g. ADWIN)

* Alexey Tsymbal, Mykola Pechenizkiy, Padraig Cunningham, and Seppo Puuronen. Dynamic integration of classifiers for handling concept drift. Information Fusion, vol. 9, no 1, pp. 56–68, 2008.

How to deal with concept drift?

Sliding windows



Algorithm Basic windowing algorithm

Require: S_∞ - data stream, W - window of examples

```
initialize window $W$
for all examples $X_i$ in $S_\infty$ $(i = 1, 2, ...)$ do
    $W \leftarrow W \cup \{X_i\}$
    if necessary remove outdated examples from $W$
    rebuild/update $C$ using $W$
end for
return a classifier built on examples in window $W$
```

How to deal with concept drift?

Weighted windows

The oldest examples are discarded by using a decay function assigning a weight to each example, e.g.

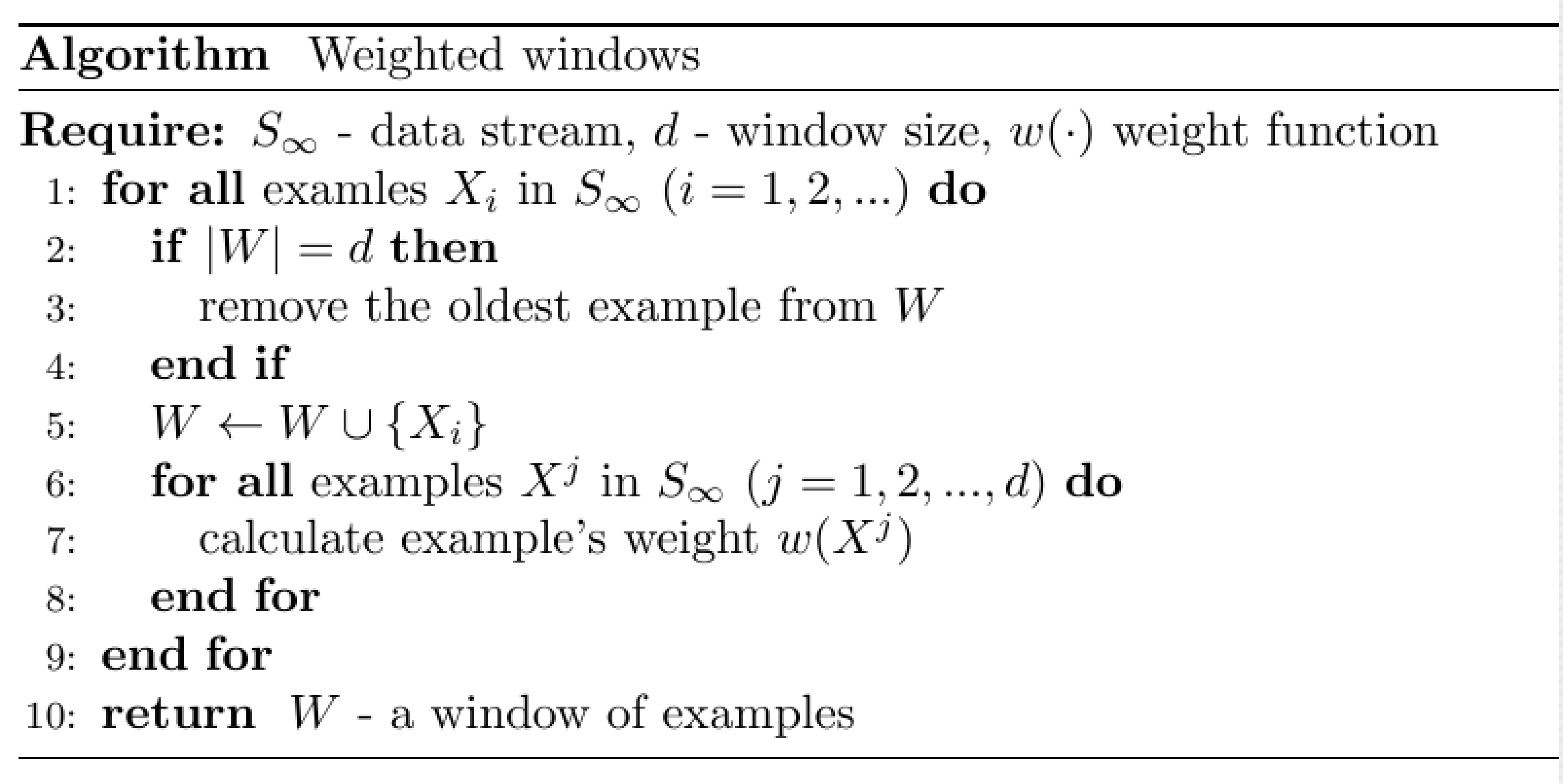
$$w_1(t) = e^{-\lambda t},\ \lambda > 0,$$
$$w_2(t) = t^{-\alpha},\ \alpha > 0,$$
$$w_3(t) = 1 - \frac{t}{|W|},$$

where t represents the age of an example ($t = 0$ for a new example and $t = |W| - 1$ for the last one).

* Edith Cohen, Martin J. Strauss, Maintaining time-decaying stream aggregates, J. Algorithms, vol. 59, no. 1, pp. 19–36, 2006.

How to deal with concept drift?

Weighted windows



```
Algorithm Weighted windows
Require: $S_\infty$ - data stream, $d$ - window size, $w(\cdot)$ weight function
for all examles $X_i$ in $S_\infty$ $(i = 1, 2, ...)$ do
    if $|W| = d$ then
        remove the oldest example from $W$
    end if
    $W \leftarrow W \cup \{X_i\}$
    for all examples $X^j$ in $S_\infty$ $(j = 1, 2, ..., d)$ do
        calculate example's weight $w(X^j)$
    end for
end for
return $W$ - a window of examples
```

How to deal with concept drift?

(a family of algorithms called FISH)

The FISH* algorithms are family of methods that take advantage of similarities between data elements both in time and space. Based on distances in space $d^{(s)}_{ij}$ and in time $d^{(t)}_{ij}$ we calculate the distance between X_i and X_j as follows:

$$D_{ij}=a_1d^{(s)}_{ij} + a_2d^{(s)}_{ij}$$

where a_1 and a_2 are the weight coefficients.

* Indre Žliobaite, Combining time and space similarity for small size learning under concept drift, In Foundations of Intelligent Systems, volume 5722 of Lecture Notes in Computer Science, pp.412–421, Springer Berlin Heidelberg, 2009.
** Indre Žliobaite, Adaptive Training Set Formation, PhD thesis, Vilnius University, Lithuania, 2010.

How to deal with concept drift?

(a family of algorithms called FISH)

In the first version of the algorithm, the number of data instances in the created training sample is determined by the user, while in FISH2** this number is not fixed. Moreover, FISH* builds separate data sets for each class, while the second version takes data from all classes based only on the closeness of the data elements. In the third version of this algorithm (FISH3**), a search for the best weight coefficients a_1 and a_2 is being conducted. In this case, the size of the data window is established dynamically.

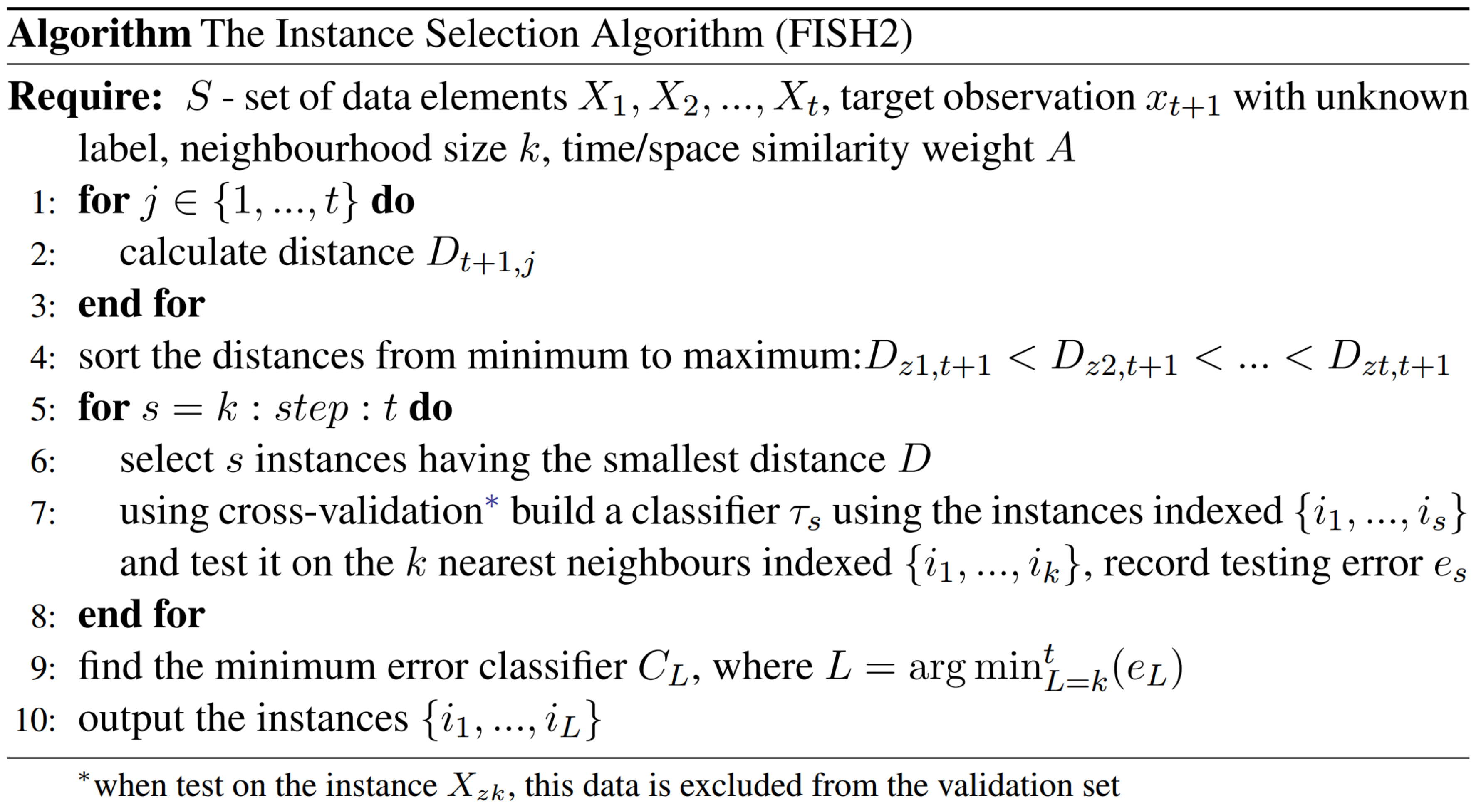
---

* Indre Žliobaite, Combining time and space similarity for small size learning under concept drift, In Foundations of Intelligent Systems, volume 5722 of Lecture Notes in Computer Science, pp.412–421, Springer Berlin Heidelberg, 2009.

** Indre Žliobaite, Adaptive Training Set Formation, PhD thesis, Vilnius University, Lithuania, 2010.

How to deal with concept drift?

(a family of algorithms called FISH)



```
**Algorithm** The Instance Selection Algorithm (FISH2)
**Require:** $S$ - set of data elements $X_1, X_2, ..., X_t$, target observation $x_{t+1}$ with unknown
    label, neighbourhood size $k$, time/space similarity weight $A$
for $j \in \{1, ..., t\}$ do
    calculate distance $D_{t+1,j}$
end for
sort the distances from minimum to maximum:$D_{z1,t+1} < D_{z2,t+1} < ... < D_{zt,t+1}$
for $s = k : step : t$ do
    select $s$ instances having the smallest distance $D$
    using cross-validation* build a classifier $\tau_s$ using the instances indexed $\{i_1, ..., i_s\}$
    and test it on the $k$ nearest neighbours indexed $\{i_1, ..., i_k\}$, record testing error $e_s$
end for
find the minimum error classifier $C_L$, where $L = \arg\min_{L=k}^{t}(e_L)$
output the instances $\{i_1, ..., i_L\}$
```

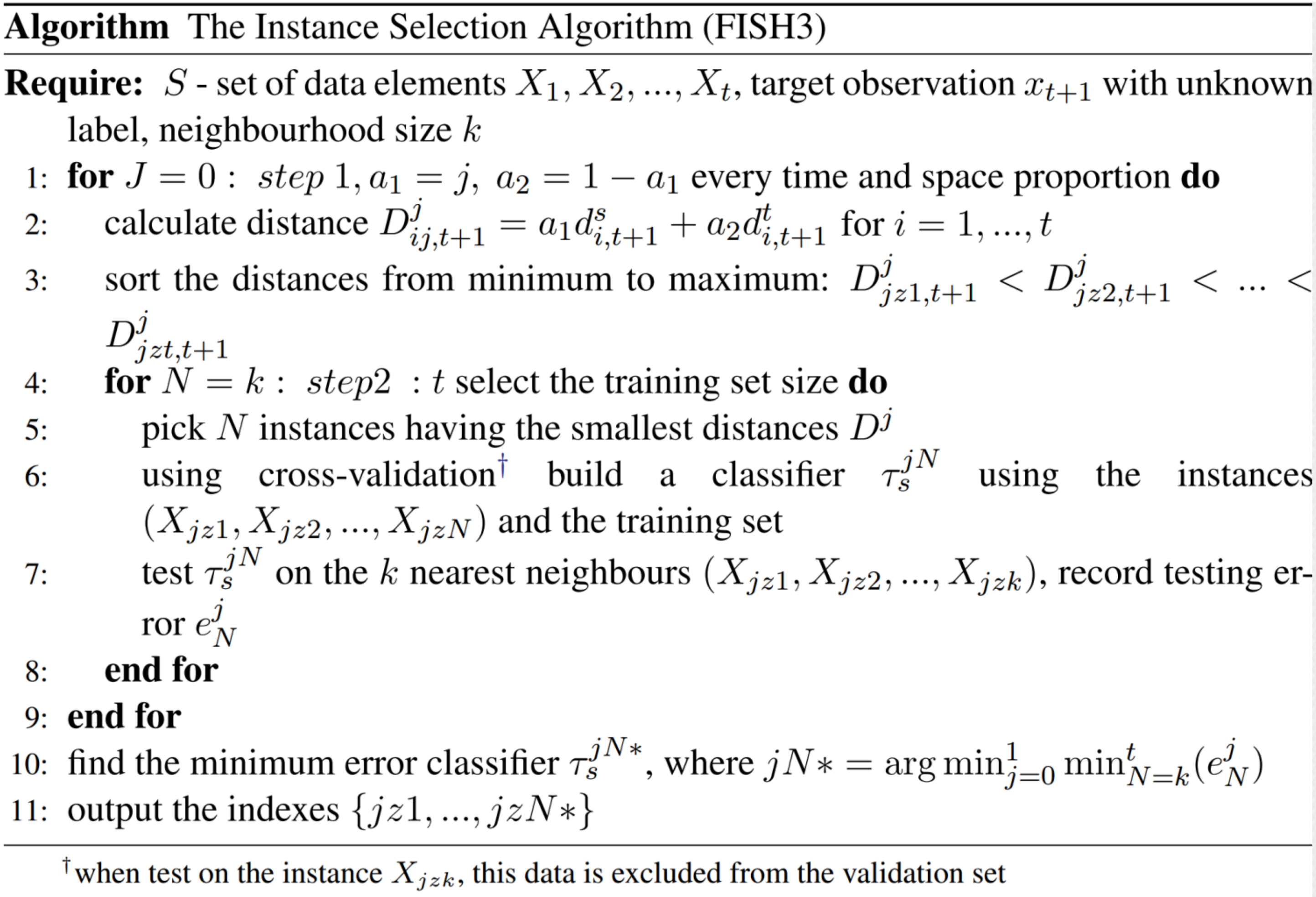
---

*when test on the instance X_{zk}, this data is excluded from the validation set

Indre Žliobaite, Adaptive Training Set Formation, PhD thesis, Vilnius University, Lithuania, 2010.

How to deal with concept drift?

(a family of algorithms called FISH)



Algorithm The Instance Selection Algorithm (FISH3)

```
Require: $S$ - set of data elements $X_1, X_2, ..., X_t$, target observation $x_{t+1}$ with unknown label, neighbourhood size $k$
for $J = 0 :\ step\ 1, a_1 = j,\ a_2 = 1 - a_1$ every time and space proportion do
    calculate distance $D^j_{ij,t+1} = a_1 d^s_{i,t+1} + a_2 d^t_{i,t+1}$ for $i = 1, ..., t$
    sort the distances from minimum to maximum: $D^j_{jz1,t+1} < D^j_{jz2,t+1} < ... < D^j_{jzt,t+1}$
    for $N = k :\ step2\ : t$ select the training set size do
        pick $N$ instances having the smallest distances $D^j$
        using cross-validation† build a classifier $\tau^{jN}_s$ using the instances $(X_{jz1}, X_{jz2}, ..., X_{jzN})$ and the training set
        test $\tau^{jN}_s$ on the $k$ nearest neighbours $(X_{jz1}, X_{jz2}, ..., X_{jzk})$, record testing error $e^j_N$
    end for
end for
find the minimum error classifier $\tau^{jN*}_s$, where $jN* = \arg\min^1_{j=0} \min^t_{N=k}(e^j_N)$
output the indexes $\{jz1, ..., jzN*\}$
```

† when test on the instance X_{jzk}, this data is excluded from the validation set

* Indre Žliobaite, Adaptive Training Set Formation, PhD thesis, Vilnius University, Lithuania, 2010.

How to deal with concept drift?

- the ADWIN

This algorithm was obtained to regulate the size of data elements windows. It uses hypothesis that is based on differences of mean value of windows W_0 and W_1 compared with the threshold value α. If we assume that $|W_0|$ and $|W_1|$ denote the number of data elements in windows W_0 and W_1 respectively, then we have to find ϵ_{cut} such that $|\mu\widehat{W}_0 - \mu\widehat{W}_1| < \epsilon_{cut}$. The value of ϵ_{cut} is determined as follows:

$$\epsilon_{cut} = \sqrt{\frac{1}{2m}\ln\frac{4}{\delta'}}$$

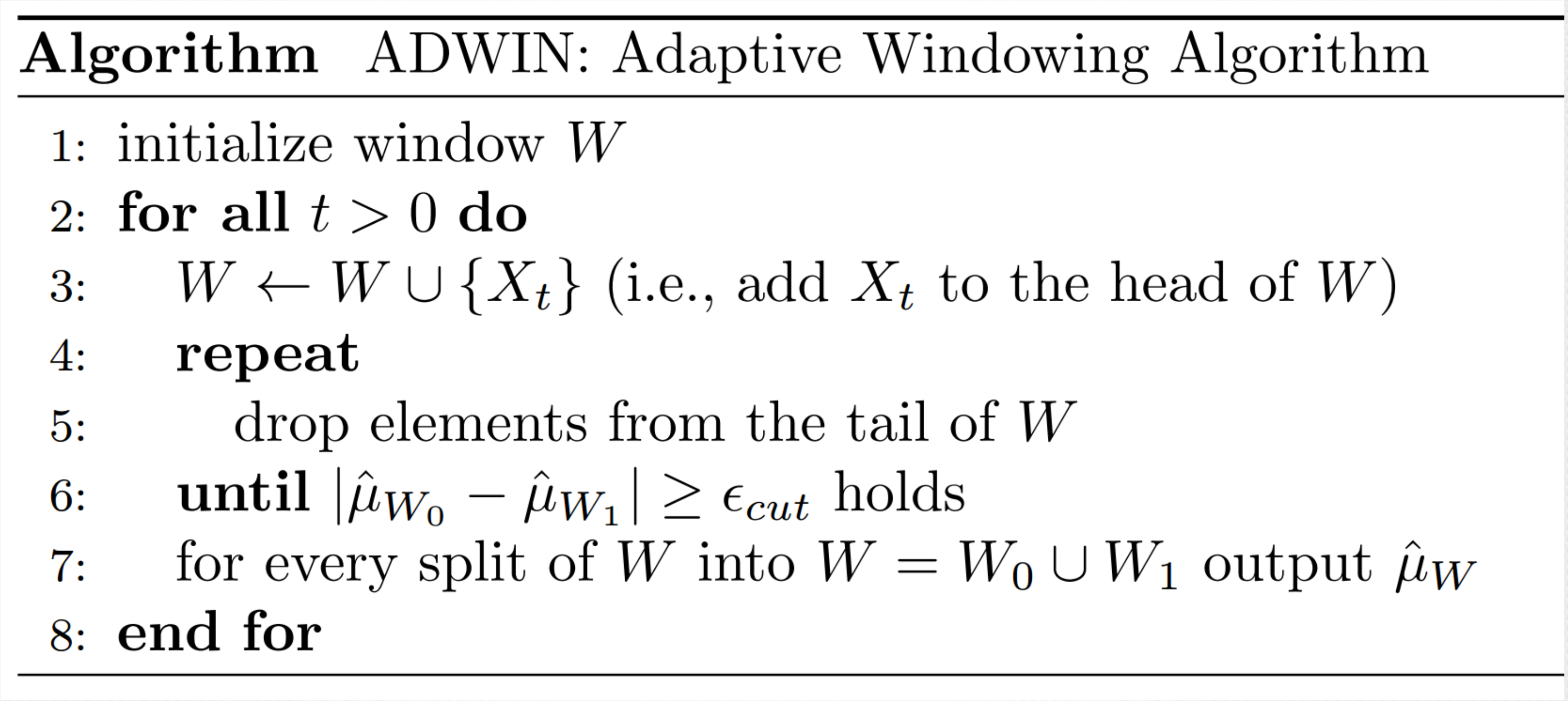
$$m = \frac{1}{\frac{1}{|W_0|}+\frac{1}{|W_1|}} \qquad \delta' = \frac{\delta}{|W_0| + |W_1|}$$

In this method we compare every possible split of entire window into windows W_0 and W_1.

* Bifet A., Avalda R., Kalman filters and adaptive windows for learning in data streams. In Proc. of the 9th int. conf. on Discovery science, DS. 29–40, 2006.

How to deal with concept drift?

- the ADWIN



Algorithm ADWIN: Adaptive Windowing Algorithm

```
initialize window $W$
for all $t > 0$ do
    $W \leftarrow W \cup \{X_t\}$ (i.e., add $X_t$ to the head of $W$)
    repeat
        drop elements from the tail of $W$
    until $|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| \geq \epsilon_{cut}$ holds
    for every split of $W$ into $W = W_0 \cup W_1$ output $\hat{\mu}_W$
end for
```

* Bifet A., Avalda R., Kalman filters and adaptive windows for learning in data streams. In Proc. of the 9th int. conf. on Discovery science, DS. 29–40, 2006.

How to deal with concept drift?

- the Drift Detection Method (DDM)

In the DDM algorithm the authors noticed that the class assigned by a classifier can be either true or false. Therefore they model the number of classification errors with a binomial distribution. Let p_i denote the probability of a false prediction, then the standard deviation is calculated as follows:

$$s_i = \sqrt{\frac{p_i(1-p_i)}{i}}$$

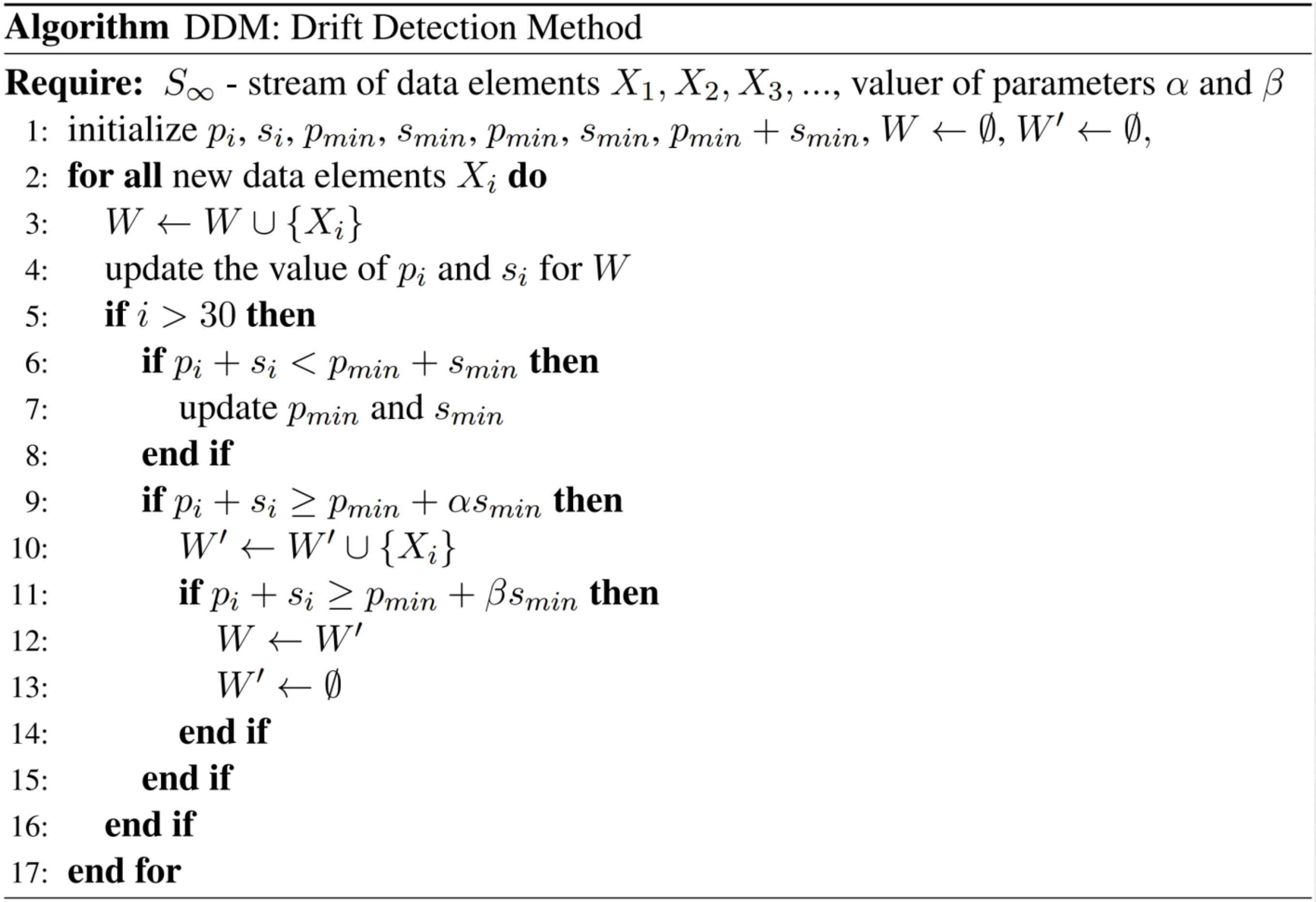
For a sufficiently large number of examples ($n > 30$), the binomial distribution can be approximated by a Gaussian distribution with the same mean and variance. The error rate is monitored by updating two registers: p_{min} and s_{min}.

- **warning level** $\boxed{p_i + s_i \geq p_{min} + \alpha \cdot s_{min}}$
- **alarm level** $\boxed{p_i + s_i \geq p_{min} + \beta \cdot s_{min}}$

João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, Learning with drift detection, In AnaL.C. Bazzan and Sofiane Labidi, editors, Advances in Artificial Intelligence – SBIA 2004, vol. 3171 of Lecture Notes in Computer Science, pp. 286–295, Springer Berlin Heidelberg, 2004.

How to deal with concept drift?

- the Drift Detection Method (DDM)



Algorithm DDM: Drift Detection Method

```
Require: $S_\infty$ - stream of data elements $X_1, X_2, X_3$, ..., valuer of parameters $\alpha$ and $\beta$
initialize $p_i$, $s_i$, $p_{min}$, $s_{min}$, $p_{min}$, $s_{min}$, $p_{min} + s_{min}$, $W \leftarrow \emptyset$, $W' \leftarrow \emptyset$,
for all new data elements $X_i$ do
    $W \leftarrow W \cup \{X_i\}$
    update the value of $p_i$ and $s_i$ for $W$
    if $i > 30$ then
        if $p_i + s_i < p_{min} + s_{min}$ then
            update $p_{min}$ and $s_{min}$
        end if
        if $p_i + s_i \geq p_{min} + \alpha s_{min}$ then
            $W' \leftarrow W' \cup \{X_i\}$
            if $p_i + s_i \geq p_{min} + \beta s_{min}$ then
                $W \leftarrow W'$
                $W' \leftarrow \emptyset$
            end if
        end if
    end if
end for
```

João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, Learning with drift detection, In AnaL.C. Bazzan and Sofiane Labidi, editors, Advances in Artificial Intelligence – SBIA 2004, vol. 3171 of Lecture Notes in Computer Science, pp. 286–295, Springer Berlin Heidelberg, 2004.

How to deal with concept drift?

- the Early Drift Detection Method (EDDM)

The main difference between DDM[*] and EDDM[**] is the definition of parameter that is being investigated. In this method p'_i defines the distance between two errors and s'_i is its standard deviation. It is expected that with increasing accuracy of the system this distance will increase.

- **warning level** $$\boxed{\frac{p'_i + 2s'_i}{p'_{max} + 2s'_{max}} < \alpha}$$
- **alarm level** $$\boxed{\frac{p'_i + 2s'_i}{p'_{max} + 2s'_{max}} < \beta}$$

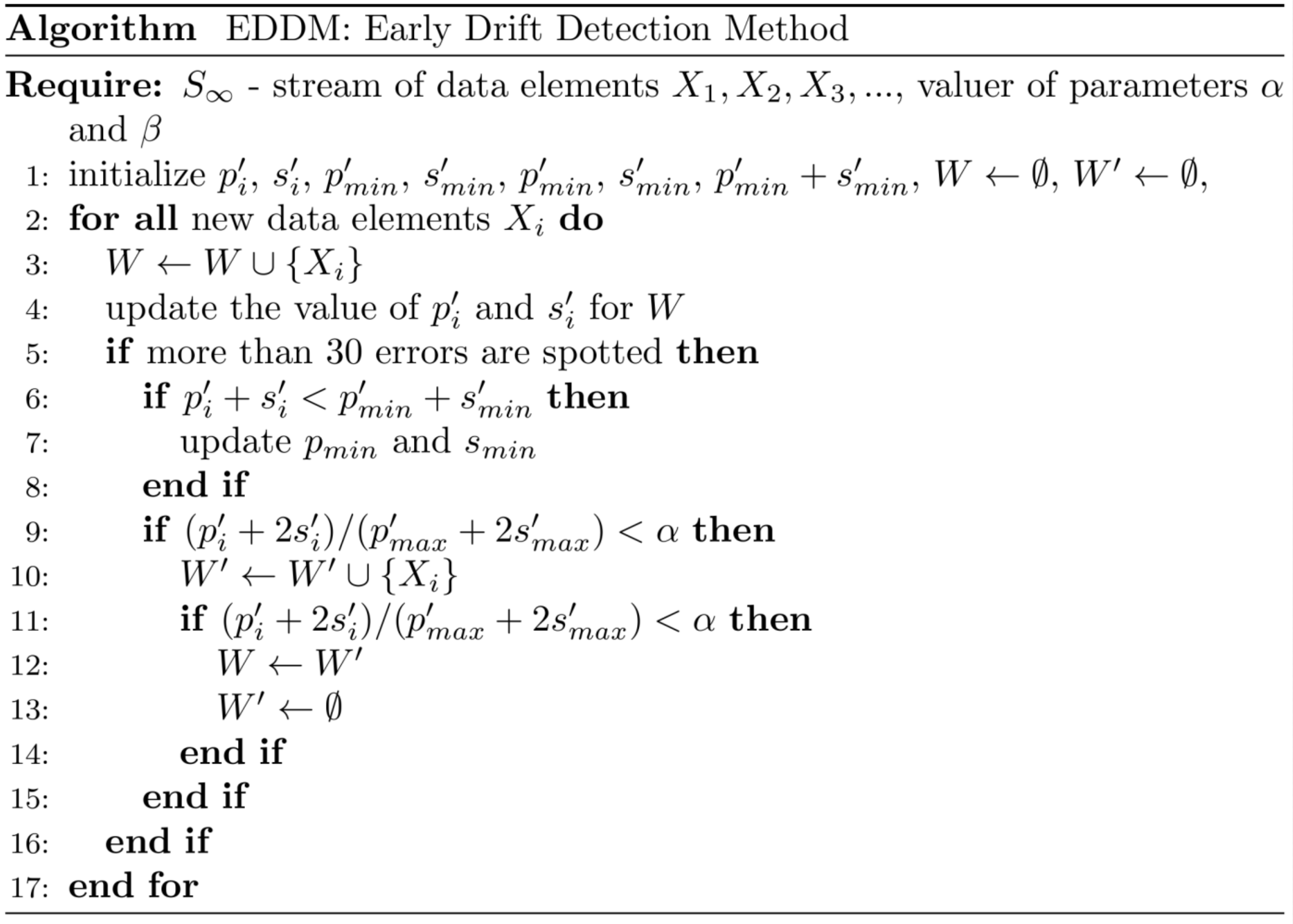
In the article authors proposed to set values of α and β to 2 and 3, respectively, which represents 95% and 90% of the distribution.

[*] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, Learning with drift detection, In AnaL.C. Bazzan and Sofiane Labidi, editors, Advances in Artificial Intelligence – SBIA 2004, volume 3171 of Lecture Notes in Computer Science, pp. 286–295, Springer Berlin Heidelberg, 2004.

[**] Manuel Baena-García, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, Ricard Gavaldá, and Rafael Morales-Bueno. Early drift detection method, In Fourth International Workshop on Knowledge Discovery from Data Streams, 2006.

How to deal with concept drift?

- the Early Drift Detection Method (EDDM)



```
Algorithm EDDM: Early Drift Detection Method
Require: $S_\infty$ - stream of data elements $X_1, X_2, X_3, ...$, valuer of parameters $\alpha$ and $\beta$
initialize $p'_i$, $s'_i$, $p'_{min}$, $s'_{min}$, $p'_{min}$, $s'_{min}$, $p'_{min} + s'_{min}$, $W \leftarrow \emptyset$, $W' \leftarrow \emptyset$,
for all new data elements $X_i$ do
    $W \leftarrow W \cup \{X_i\}$
    update the value of $p'_i$ and $s'_i$ for $W$
    if more than 30 errors are spotted then
        if $p'_i + s'_i < p'_{min} + s'_{min}$ then
            update $p_{min}$ and $s_{min}$
        end if
        if $(p'_i + 2s'_i)/(p'_{max} + 2s'_{max}) < \alpha$ then
            $W' \leftarrow W' \cup \{X_i\}$
            if $(p'_i + 2s'_i)/(p'_{max} + 2s'_{max}) < \alpha$ then
                $W \leftarrow W'$
                $W' \leftarrow \emptyset$
            end if
        end if
    end if
end for
```

* Manuel Baena-García, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, Ricard Gavaldá, and Rafael Morales-Bueno. Early drift detection method, In Fourth International Workshop on Knowledge Discovery from Data Streams, 2006.

How to deal with concept drift?

- the Page-Hinkley test

The Page-Hinkley test was originally used as a sequential analysis technique for change detection in signal processing. Recently it has been proposed as a drift detector*. It allows to efficiently detect changes in the normal behavior of a process established by a model. The cumulative variable U_T of this test is defined as the cumulative difference between the observed values X_i and their mean up to the current moment in time:

$$U_T = \sum_{i=1}^{T} (X_i - \bar{X}_T - \delta)$$

where $\bar{X}_T = 1/t \sum_{i=1}^{t} X_i$ and δ corresponds to the magnitude of changes that are allowed. In the drift detection we treat the classifier's error rate as the observed value.

---

* João Gama, Raquel Sebastião, Pedro P. Rodrigues, On evaluating stream learning algorithms, Machine Learning, vol. 90, no. 3, pp. 317–346, 2013

How to deal with concept drift?

- the Page-Hinkley test

The minimal U_T is defined as

$$U_T^{min} = \min\{U_T; i = 1, \dots, t\}.$$

The PH test calculates the difference between U_T^{min} and U_T

$$PH_T = U_T - U_T^{min}.$$

If this difference is higher than a user specified threshold λ, a change is flagged. The threshold λ depends on the admissible false alarm rate. Increasing its value will entail fewer false alarms, but might miss or delay some changes. Controlling this detection threshold parameter makes it possible to establish a trade-off between the false alarms and the miss detections.

How to deal with concept drift?

- the Welch's t-test

This test applies on two samples of size n_1 and n_2 and is an adaptation of the Student's t test. This test is used to statistically test the null hypothesis that the means of two populations $\bar{X}_1$ and $\bar{X}_2$, with unequel variances (s_2^1 and s_2^2), are equal. The formula of this test is:

$$p\text{-}value = \frac{\bar{X}_1 - \bar{X}_2}{\left(\sqrt{\frac{s_2^1}{n_1} - \frac{s_2^2}{n_2}}\right)}$$

The null hypothesis can be rejected depending on the *p-value*.

How to deal with concept drift?

- the Kolmogorov-Smirnov test

The Kolmogorov–Smirnov statistic determines a distance between the empirical distribution function of the sample and the reference distribution or between the empirical distribution functions of two samples. In the latter case we calculate

$$D_{n,n'} = \sup_x |F_{1,n}(x) - F_{2,n'}(x)|\,,$$

where $F_{1,n}(x)$ and $F_{2,n'}(x)$ are the empirical distribution functions of the first and the second sample respectively, and sup is the supremum function.

The null hypothesis that the samples are drawn from the same distribution is rejected with a confidence α if

$$D_{n,n'} > c(\alpha)\sqrt{\frac{n+n'}{nn'}}.$$

The value of $c(\alpha)$ can be found in the Kolmogorov-Smirnov table.

Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- **Data stream classification methods – short overview**
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Data stream classification methods

- Neural networks
- Support vector machine
- Naive Bayes classifier
- Rule-based systems
- Nearest neighbor (lazy learners)
- Decision trees
- Ensemble methods

Data stream classification methods

Neural networks

- João Gama, Pedro P. Rodrigues, Stream-based electricity load forecast, In Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, vol. 4702, Lecture Notes in Computer Science, pp 446–453, Springer, 2007.

- Daniel Leite, Pyramo Costa Jr., Fernando Gomide, Evolving granular neural network for semi-supervised data stream classification, In Proceedings of the 2010 International Joint Conference on Neural Networks, pp. 1–8, IEEE, 2010.

- Daniel Leite, Pyramo Costa Jr., Fernando Gomide, Evolving granular neural networks from fuzzy data streams, Neural Networks, vol. 38, pp. 1–16, 2013.

Data stream classification methods

Support vector machine

- Domeniconi C., Gunopulos D., Incremental support vector machine construction, In ICDM, pp. 589–592, 2001.
- Syed N., Liu H., Sung K., Handling concept drifts in incremental learning with supportvector machines, Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM New York, NY, USA pp. 317–321, 1999.
- Fung G., Mangasarian O., Incremental support vector machine classification, Proceed-ings of the Second SIAM International Conference on Data Mining, Arlington, Virginia, pp. 247–260, 2002.
- Bordes A., Bottou L., The Huller: a simple and efficient online SVM, Proceedings of the16th European Conference on Machine Learning (ECML2005), 2005.
- Bordes A., Ertekin, S., Weston J., Bottou L., Fast kernel classiffiers with online and activelearning, Journal of Machine Learning Research 6, pp. 1579–1619, 2005.

Data stream classification methods

Naive Bayes classifier

- Remco R. Bouckaert, Voting massive collections of bayesian network classifiers for data streams. In Proceedings of the 19th Australian Joint Conference on Artificial Intelligence, vol. 4304, Lecture Notes in Computer Science, pp. 243–252, Springer, 2006.
- João Gama, Ricardo Rocha, Pedro Medas, Accurate decision trees for mining high-speed data streams, In Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 523–528, ACM, 2003.
- João Gama, Pedro P. Rodrigues, Stream-based electricity load forecast, In Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, vol. 4702, Lecture Notes in Computer Science, pp. 446–453, Springer, 2007.
- Richard Kirkby, Improving Hoeffding Trees, PhD thesis, Department of Computer Science, University of Waikato, 2007.

Data stream classification methods

Rule-based systems

- Magdalena Deckert, Jerzy Stefanowski, RILL: algorithm for learning rules from streaming data with concept drift, In Proceedings of the 21st ISMIS International Symposium on Foundations of Intelligent Systems, vol. 8502, Lecture Notes in Computer Science, pp. 20–29, Springer, 2014.
- Francisco J. Ferrer-Troyano, Jesús S. Aguilar-Ruiz, José Cristóbal Riquelme Santos, Discovering decision rules from numerical data streams, In Proceedings of the 2004 ACM Symposium on Applied Computing, pp. 649–653, ACM, 2004.
- Petr Kosina, João Gama, Handling time changing data with adaptive very fast decision rules, In Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases, vol. 7523, Lecture Notes in Computer Science, pp. 827–842, Springer, 2012.
- Petr Kosina, João Gama, Very fast decision rules for classification in data streams, Data Min. Knowl. Discov., vol. 29, no. 1, pp. 168–202, 2015.
- Gerhard Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, In Machine Learning, pp. 69–101, 1996.

Data stream classification methods

Nearest neighbor (lazy learners)

- Law Y., Zaniolo C., An adaptive nearest neighbor classification algorithm for data streams, Lecture notes in computer science, vol. 3721, pp. 108-120, 2005
- Beringer, J., Hüllermeier, E., Efficient instance-based learning on data streams, IntelligentData Analysis, vol. 11, no 6, pp. 627–650, 2007
- Shaker, A., Hüllermeier, E., Iblstreams: a system for instance-based classification and re-gression on data streams, Evolving Systems, vol. 3, no. 4, pp. 235–249, 2012
- Yan-Nei Law, Carlo Zaniolo, An adaptive nearest neighbor classification algorithm for data streams, In Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, vol. 3721, Lecture Notes in Computer Science, pp. 108–120, Springer, 2005.

Data stream classification methods

Decision trees

- P. Domingos and G. Hulten, Mining high-speed data streams, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.
- G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01), ACM, New York, NY, USA, pp. 97-106, 2001
- Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, Piotr Duda, A new method for data stream mining based on the misclassification error, IEEE Transaction on Neural Networks and Learning Systems, vol. 26, pp. 1048-1059, no. 5, 2015.
- Leszek Rutkowski, Lena Pietruczuk, Piotr Duda, Maciej Jaworski, Decision trees for mining data streams based on the McDiarmid's bound, IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 6, pp. 1272-1279, June 2013.

Data stream classification methods

Ensemble methods

- Brzezinski D., Stefanowski J., Combining block-based and online methods in learning ensembles from concept drifting data streams, Information Sciences, vol. 265, pp. 50-67, 2014.
- Dewan Md. Farid, Li Zhang, Alamgir Hossain, Chowdhury Mofizur Rahman, Rebecca Strachan, Graham Sexton, Keshav Dahal, An adaptive ensemble classifier for mining concept drifting data streams, Expert Systems with Applications, vol. 40, no. 15, pp. 5895-5906, 2013.
- Ryan Elwell, Robi Polikar, Incremental learning of concept drift in nonstationary environments, IEEE Transactions on Neural Networks, vol. 22, no. 10, pp. 1517-1531, Oct 2011.

Data stream classification methods

- Neural networks
- Support vector machine
- Naive Bayes classifier
- Rule-based systems
- Nearest neighbor (lazy learners)
- Decision trees
- Ensemble methods

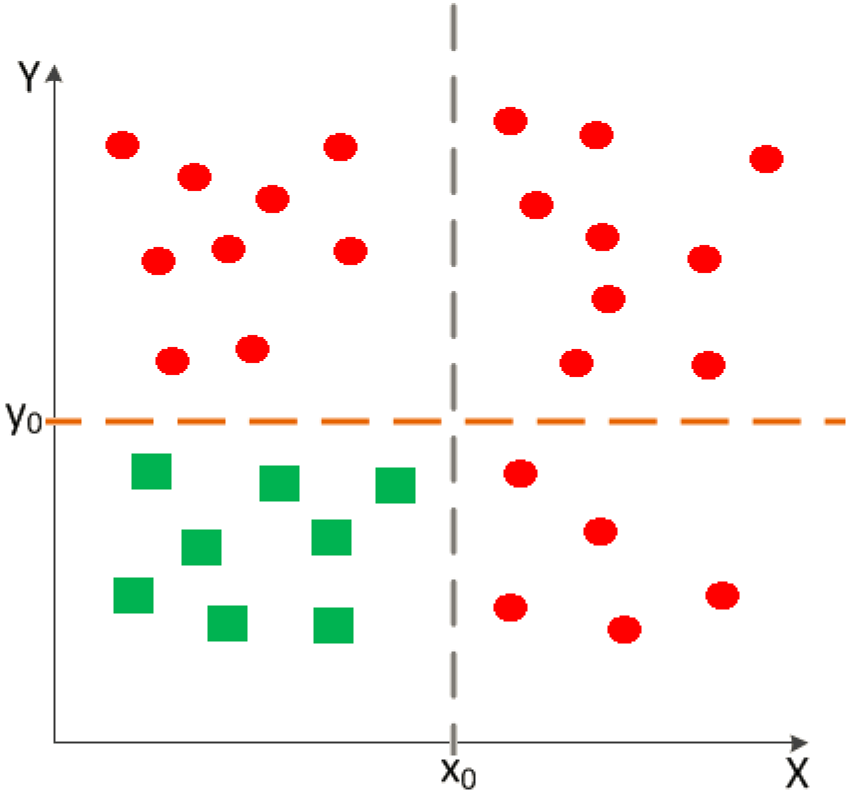
Data stream mining

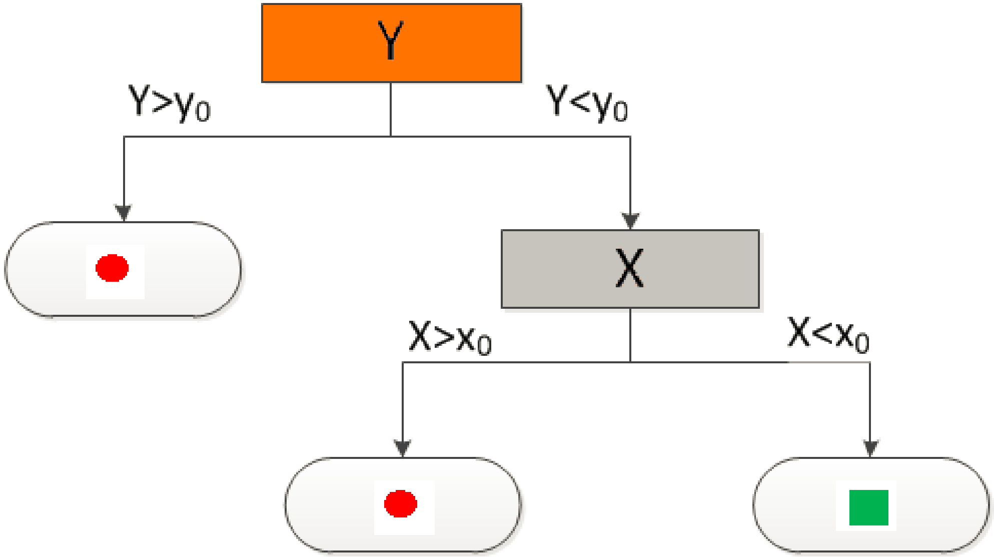
- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- **Decision trees for data streams (including new results 2016)**
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Decision trees for data mining (non-stream data)

DECISION TREES



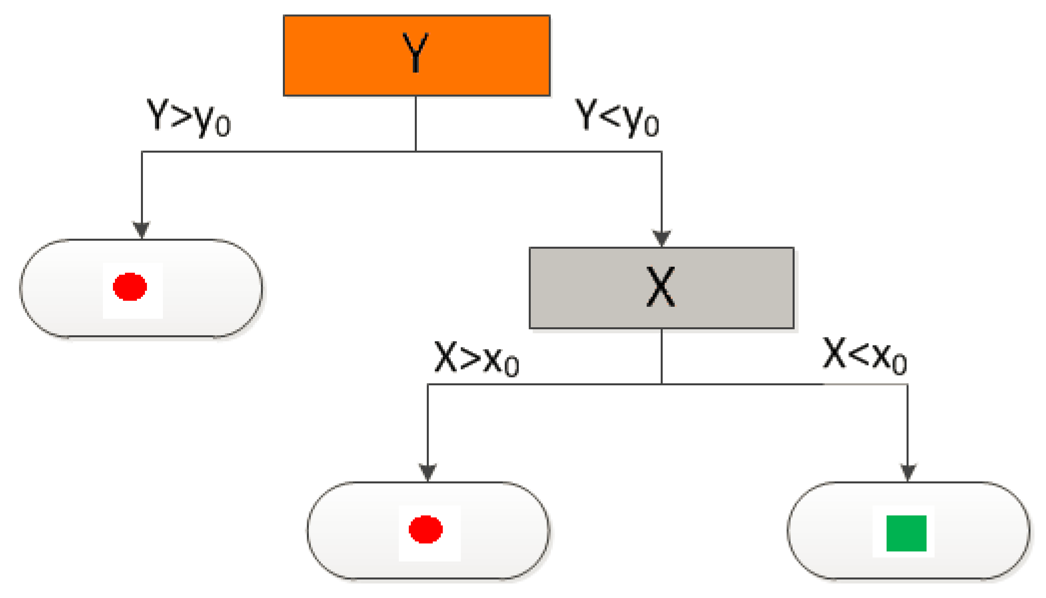


DECISION TREES

Commonly known decision tree algorithms:

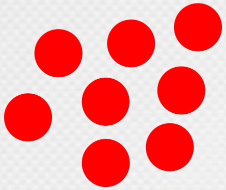
- ❖ **ID3 algorithm** – J. R. Quinlan, "Induction of decision tree," Machine Learning, Vol. 1, No. 1, pp. 81-106, 1986.
- ❖**C4.5 algorithm** – J. R. Quinlan, „C4.5: Programs for Machine Learning," Morgan Kaufmann Publishers, 1993
- ❖**CART algorithm** – L. Breiman, J. H. Friedman,R. A. Olshen, & C. J. Stone, „Classification and regression trees." Monterey, CA: Wadsworth & Brooks/Cole, 1984. Advanced Books & Software.

DECISION TREES

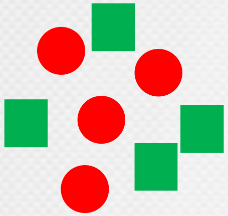


- The most critical point is the choice of a splitting attribute in each node;
- In ID3, C4.5, CART – the choice based on an impurity measure.

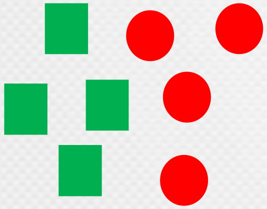
Impurity Measure



Lowest possible (0) impurity measure – maximally „pure” set



Highest possible impurity measure



Highest possible impurity measure

Impurity measure – information entropy

The entropy of *S* is defined as follows

$$\mathrm{H}(S) = -\sum_{i=1}^{K} p_i \log(p_i) \,,$$

S – training set $(X_1, \dots, X_N)$

K – number of classes

p_i – probability that element belongs to the i-th class (proportion of elements in *S* belonging to the i-th class)

Impurity measure – information entropy

$$\mathrm{H}(S) = -\sum_{i=1}^{K} p_i \log(p_i)\,,$$

- If $\boldsymbol{p_i = \frac{1}{K}, i = 1, \dots, K}$, then $\boldsymbol{H(S) = \log_2 K}$ - maximal value
- If $\boldsymbol{p_j = 1, p_{i \neq j} = 0}$, then $\boldsymbol{H(S) = 0}$ - minimal value

Split measure function

For each attribute the quality of potential split is measured by the **SPLIT MEASURE FUNCTION**

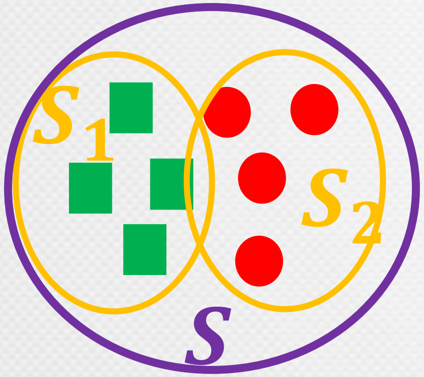
Split measure function – a reduction of an impurity measure

Split measure function

Split measure:

$$f(S) = g(S) - \frac{|S_1|}{|S|} g(S_1) - \frac{|S_2|}{|S|} g(S_2)$$

Example



S – high impurity measure $g(S)$

S_1 - low impurity measure $g(S_1)$

S_2 - low impurity measure $g(S_2)$

Split measure – information gain

$$H(S|a) = H(S) - \sum_{i=1}^{|a|} w_i H(S_i),$$

where

a – an attribute with values in set $\{a_1, \ldots, a_{|a|}\}$

$|a|$ – number of different values of attribute a

w_i - fraction of elements with value a_i

(probability that elements in S take value a_i for attribute a)

S_i - subset of set S, with data elements for which the value of attribute a is equal a_i

Information entropy and information gain – an example

married	age	uses computer at work	buy computer
Yes	young	Yes	Yes
Yes	young	Yes	Yes
No	young	Yes	Yes
Yes	middle	Yes	Yes
No	middle	Yes	Yes
No	old	Yes	Yes
Yes	old	No	No
Yes	old	No	No
Yes	young	No	Yes
Yes	middle	No	No

$$H(S) = -\left(\frac{7}{10} log_2\left({}^{7}/_{10}\right) + \frac{3}{10} log_2\left({}^{3}/_{10}\right)\right) = 0{,}88129$$

Partition according to „uses computer at work”

S_{Yes}

is married	age	uses computer at work	by computer
Yes	young	Yes	Yes
Yes	young	Yes	Yes
No	young	Yes	Yes
Yes	middle	Yes	Yes
No	middle	Yes	Yes
No	old	Yes	Yes

S_{No}

is married	age	uses computer at work	by computer
Yes	old	No	No
Yes	old	No	No
Yes	young	No	Yes
Yes	middle	No	No

Information gain: H(*S*|use computer at work)

S_{Yes}

is married	age	uses computer at work	by computer
Yes	young	Yes	Yes
Yes	young	Yes	Yes
No	young	Yes	Yes
Yes	middle	Yes	Yes
No	middle	Yes	Yes
No	old	Yes	Yes

S_{No}

is married	age	uses computer at work	by computer
Yes	old	No	No
Yes	old	No	No
Yes	young	No	Yes
Yes	middle	No	No

$$P(a = Yes) = {}^{6}/_{10} = w_1$$

$$P(a = No) = {}^{4}/_{10} = w_2$$

$$H(S_{Yes}) = 0$$

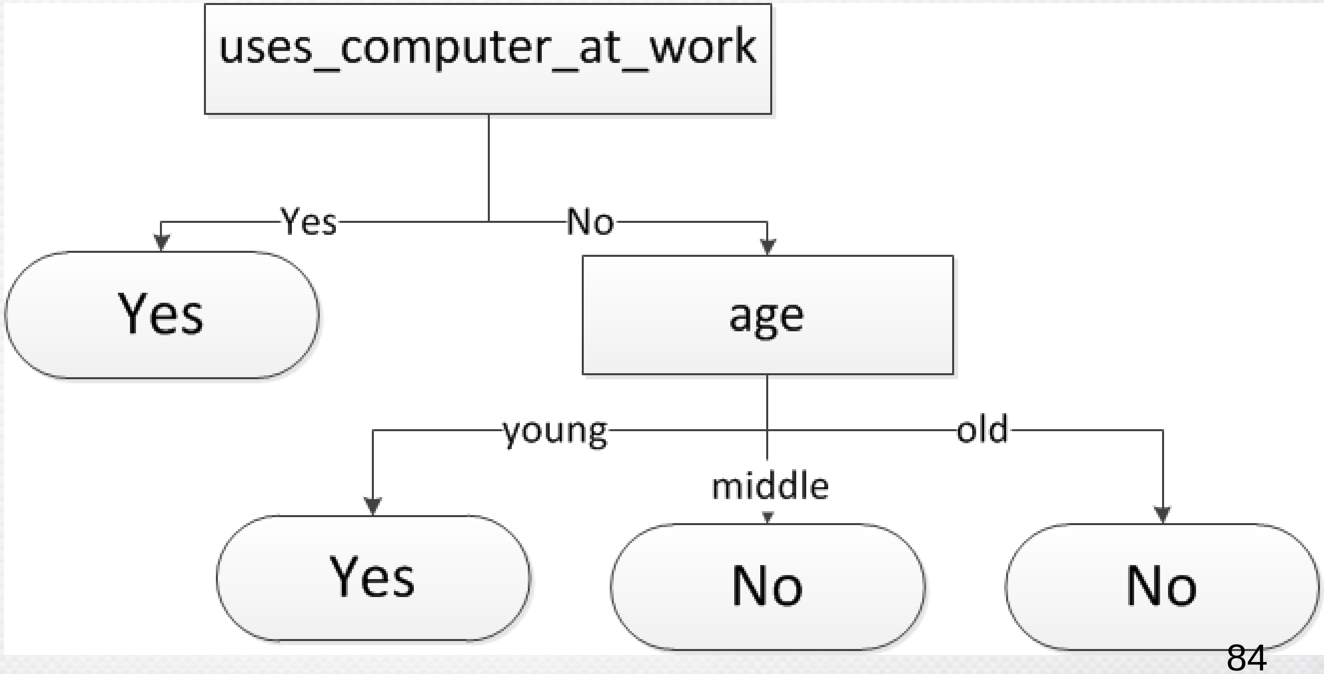
$$H(S_{No}) = 0{,}81128$$

$$H(S|a) = H(S) - {}^{6}/_{10} * 0 - {}^{4}/_{10} * 0{,}81128 = 0{,}55678$$

WCCI’ 2016 Tutorial, Vancouver, July 24, 2016

Obtained decision tree

attribute a	***H(S|a)***
is married	0,191631
age	0,330313
uses computer at work	0,55678



Other impurity measures

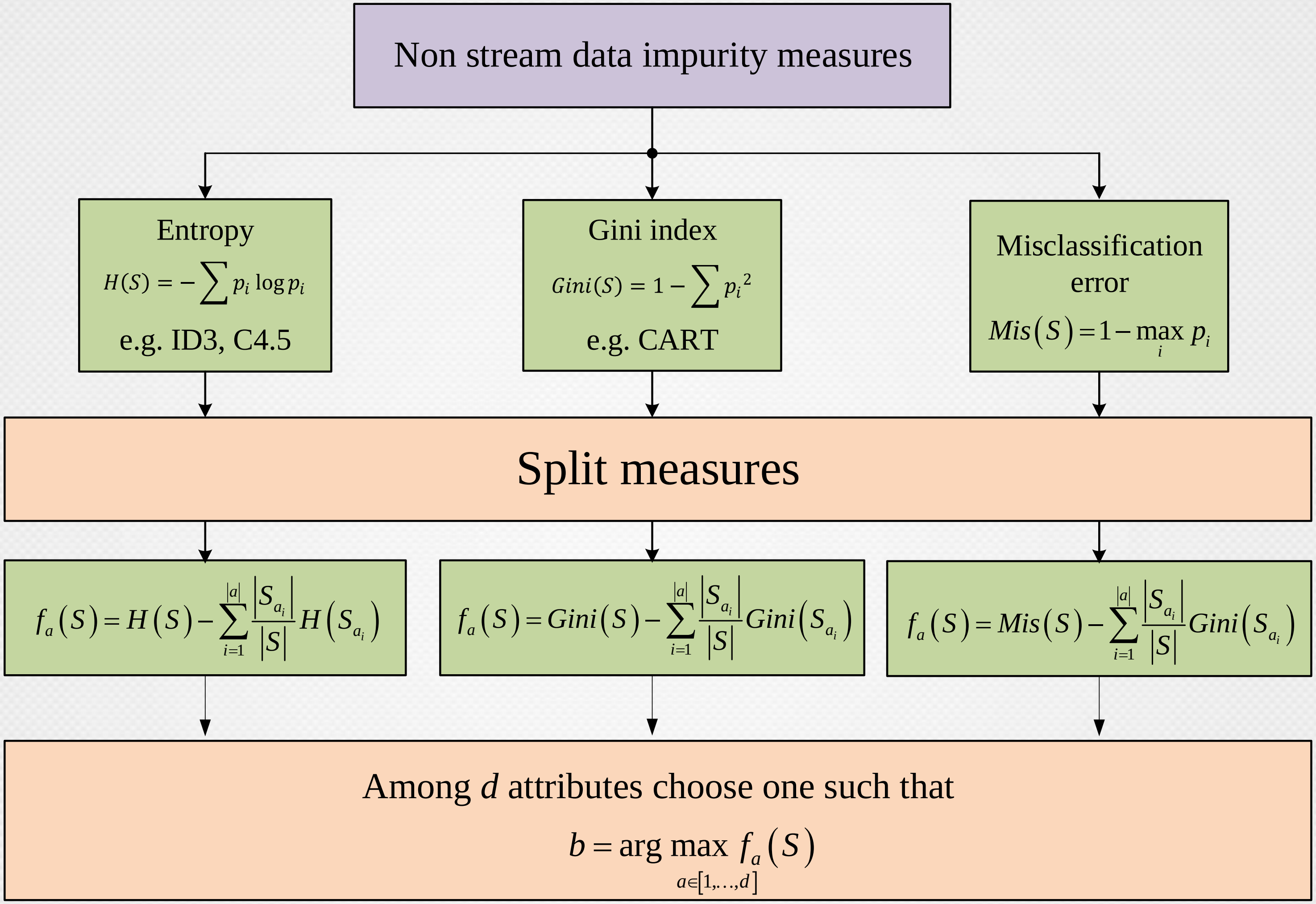
- Gini index:

$$\mathbf{Gini}(S) = \mathbf{1} - \sum_{i=1}^{K} (p_i)^2 \,,$$

- Misclassification error:

$$\mathbf{Mis}(S) = \mathbf{1} - \max_{i \in \{1,\ldots,K\}} p_i$$

Decision Trees for data mining (non-stream data)



Non stream data impurity measures

Entropy

$H(S) = -\sum p_i \log p_i$

e.g. ID3, C4.5

Gini index

$Gini(S) = 1 - \sum {p_i}^2$

e.g. CART

Misclassification error

$Mis(S) = 1 - \max_i p_i$

Split measures

$$f_a(S) = H(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} H(S_{a_i})$$

$$f_a(S) = Gini(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} Gini(S_{a_i})$$

$$f_a(S) = Mis(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} Gini(S_{a_i})$$

Among *d* attributes choose one such that

$$b = \arg\max_{a \in [1,\ldots,d]} f_a(S)$$

Decision Trees for data streams

Decision tree for data stream

In the case of data streams data arrive continuously to the considered node

Decision tree for static data:

1) Which attribute to choose?

Decision tree for stream data:

1) Which attribute to choose?
2) When to make a split?

Decision tree for data stream

The aim is to design a decision tree learning system, applied to data streams, which provides an output nearly identical to that induced by a conventional learner.

Decision tree for data stream

Hoeffding trees

P. Domingos and G. Hulten, ”Mining high-speed data streams”, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01), ACM, New York, NY, USA, pp. 97-106, 2001

Hoeffding Tree Algorithm - 2000

Main ideas applied in the Hoeffding tree algorithm:

1) **Sufficient statistics**
2) **Splitting criterion**

P. Domingos and G. Hulten, "Mining high-speed data streams", Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

Hoeffding tree algorithm: sufficient statistics

In each node information collected in a form of „sufficient statistics”:

N_{ijk} **- numer of data elements from the k-th class which take the j-th value of the i-th attribute**

Required memory per node: $M = K\sum_{i=1}^{d} v_i$.
For two-class binary problem: $M = 4d$.

P. Domingos and G. Hulten, ”Mining high-speed data streams”, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

Hoeffding tree algorithm: splitting criterion

1) Compute the split measure function $\boldsymbol{f_i(S)}$ for each attribute $\boldsymbol{i = 1, \dots, d}$ based on the currently collected data sample $\boldsymbol{S = X_1, \dots X_N}$

2) Find two attributes with the highest values of $\boldsymbol{f}$:

$$\boldsymbol{f_{a_{max1}}(S)}, \qquad \boldsymbol{f_{a_{max2}}(S)}$$

Hoeffding tree algorithm: splitting criterion

3) **If**

$$f_{a_{max1}}(S) - f_{a_{max2}}(S) > \varepsilon = \varepsilon(\mathrm{N}, \delta),$$

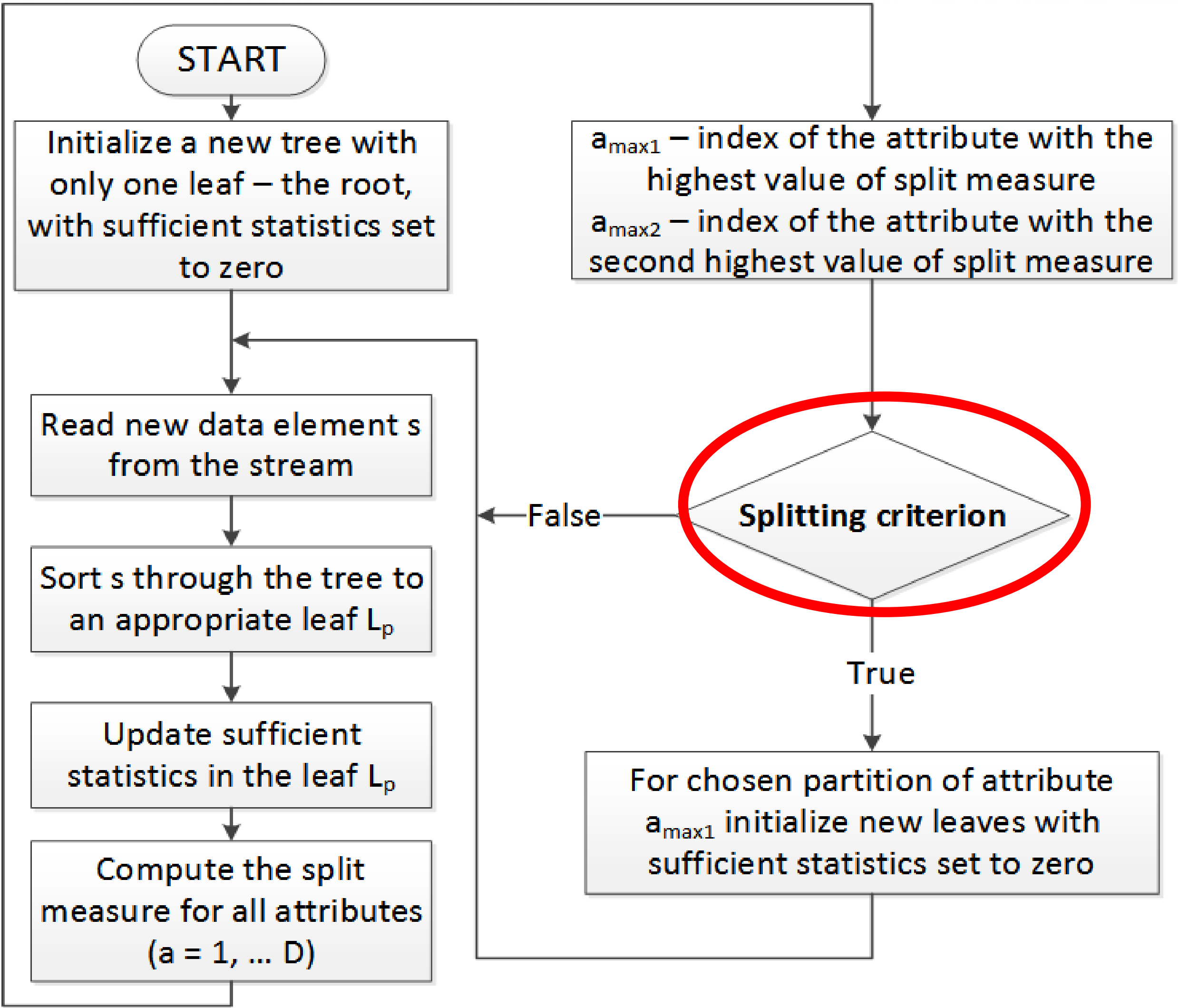
then with probability $1 - \delta$

$$E[f_{a_{max1}}(S)] > E[f_{a_{max2}}(S)]$$

and choose a_{max1} **as a splitting attribute**

Else, wait for more data elements in this node

Online Decision Tree – general algorithm



START

Initialize a new tree with only one leaf – the root, with sufficient statistics set to zero

Read new data element s from the stream

Sort s through the tree to an appropriate leaf L_p

Update sufficient statistics in the leaf L_p

Compute the split measure for all attributes (a = 1, … D)

a_{max1} – index of the attribute with the highest value of split measure

a_{max2} – index of the attribute with the second highest value of split measure

Splitting criterion

False

True

For chosen partition of attribute a_{max1} initialize new leaves with sufficient statistics set to zero

Splitting criterion:

$$f_{a_{max_1}} - f_{a_{max_2}} > \epsilon$$

Hoeffding tree algorithm: splitting criterion

Splitting criterion:

$$f_{a_{max1}}(S) - f_{a_{max2}}(S) > \varepsilon = \varepsilon(\mathrm{N}, \delta),$$

Challenge: How to find formula for $\varepsilon(\mathrm{N}, \delta)$???

Hoeffding's inequality - 1963

The commonly known algorithm called 'Hoeffdings Tree' was introduced by P. Domingos and G. Hulten in [1]. The main mathematical tool used in this algorithm was the Hoeffding's inequality [2]:

Theorem: ***If*** $X_1,\ X_2, \dots ,\ X_N$ ***are independent random variables and*** $a_i \le X_i \le b_i\ (i = 1, 2, \dots, N)$***, then for*** $\epsilon > 0$

$$P\{\bar{X} - E[\bar{X}] \ge \epsilon\} \le\ e^{-2N^2\epsilon^2 / \sum_{i=1}^{N}(b_i - a_i)^2}$$

where

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i \text{ and } E[\bar{X}] \text{ is expected value of } \bar{X}.$$

[1] P. Domingos and G. Hulten, "Mining high-speed data streams", Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

[2] W. Hoeffding, "Probability inequalities for sums of bounded random variables", Journal of the American Statistical Association, vol. 58, issue 301, pp. 13-30, March 1963.

Hoeffding’s inequality - 1963

Assuming that $P\{\bar{X} - E[\bar{X}] \geq \epsilon\} \leq \delta$
and $b_i - a_i = R, i = 1, \dots, N,$
the Hoeffding’s inequality is equivalent to:

$$P\left\{\bar{X} - E[\bar{X}] \leq \sqrt{\frac{R^2 \ln 1/\delta}{2N}}\right\} \geq 1 - \delta$$

[1] W. Hoeffding, ”Probability inequalities for sums of bounded random variables”, Journal of the American Statistical Association, vol. 58, issue 301, pp. 13-30, March 1963.

[2] P. Domingos and G. Hulten, ”Mining high-speed data streams”, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

Hoeffding's trees

P. Domingos and G. Hulten claimed that owing to the Hoeffding's inequality the form of ε is given by:

$$\boldsymbol{\varepsilon = \varepsilon(N, \delta) = \sqrt{\frac{R^2 \ln\left({}^{1}/_{\delta}\right)}{2N}}},$$

where R is a range of values of the applied split measure, e.g. $R = \log_2 K$ for information gain

[1] P. Domingos and G. Hulten, "Mining high-speed data streams", Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

Algorithms based on the Hoeffding's bound - 2000

Very Fast Decision Tree (VFDT)

The VFDT (Very Fast Decision Tree) algorithm makes several modifications to the Hoeffding tree algorithm to improve both speed and memory utilization. The modifications include:

- breaking near-ties during attribute selection more aggressively,
- computing function f_a after a number of training examples,
- deactivating the least promising leaves whenever memory running low,
- dropping poor splitting attributes,
- improving the initialization method.

[1] P. Domingos and G. Hulten, ”Mining high-speed data streams”, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

Algorithms based on the Hoeffding's bound - 2001

Concept-adapting Very Fast Decision Tree (CVFDT) for handling concept drift

CVFDT uses a sliding window approach, the main features are the following:

- it does not construct a new model from scratch each time,
- it updates statistics at the modes by incrementing the counts associated with new examples and decrementing the counts associated with old ones,

G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01), ACM, New York, NY, USA, pp. 97-106, 2001

Algorithms based on the Hoeffding's bound - 2001

Concept-adapting Very Fast Decision Tree (CVFDT) for handling concept drift

- If there is a concept drift, some nodes may no longer pass the Hoeffding bound. When this happens, an alternate subtree will be grown, with the new best splitting attribute at the root. As new examples stream in, the alternate subtree will continue to develop, without yet being used for classification. Once the alternate subtree becomes more accurate than the existing one, the old subtree is replaced.

G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01), ACM, New York, NY, USA, pp. 97-106, 2001

The Hoeffding's bound

The Hoeffding's bound is a wrong tool to solve the problem of choosing the best attribute to make a split in the node!!!

The Hoeffding's bound

Hoeffding’s inequaity is applicable only for sums (or arithmetic averages) of random variables.

Nonlinear impurity measures, like information entropy

$$\boldsymbol{H(S) = -\sum_{i=1}^{K} p_i(\mathrm{S}) \log_2 p_i(\mathrm{S})}$$

or Gini index

$$\boldsymbol{Gini(S) = 1 - \sum_{i=1}^{K} (p_i(\mathrm{S}))^2,}$$

can not be presented as a sum of elements.

The Hoeffding's bound

The idea presented in [1] violates the assumptions of the Hoeffding's theorem (see [2]) and the concept of Hoeffding Trees has no theoretical justification.

[1] P. Domingos and G. Hulten, ”Mining high-speed data streams”, Proc. 6th ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining, pp. 71-80, 2000.

[2] W. Hoeffding, ”Probability inequalities for sums of bounded random variables”, Journal of the American Statistical Association, vol. 58, issue 301, pp. 13-30, March 1963.

Challenge in Stream Data Mining

Find an appropriate, mathematically justified, form of the bound $\varepsilon = \varepsilon(\mathrm{N}, \delta)$ in the general splitting criterion:

$$f_{a_{max1}}(S) - f_{a_{max2}}(S) > \varepsilon = \varepsilon(\mathrm{N}, \delta)$$

WCCI’ 2016 Tutorial, Vancouver, July 24, 2016

What will be shown on the next slides???

a) We will study split measures based on the following impurity measures:

- information entropy,
- Gini index,
- misclassification error.

b) We will propose two different techniques to solve the problem:

- the McDiarmid's inequality,
- the Gaussian approximation.

What will be shown on the next slides???

c) We will propose the decision trees algorithms such that if

$$f_{a_{max_1}} - f_{a_{max_2}} > \varepsilon(N, \delta)$$

then

$$E\left[f_{a_{max_1}}\right] > E\left[f_{a_{max_2}}\right]$$

with probability $(1 - \delta)^{d-1}$, where d is the number of attributes.

WHAT IS THE VALUE OF ε ???

Papers concerning the issue

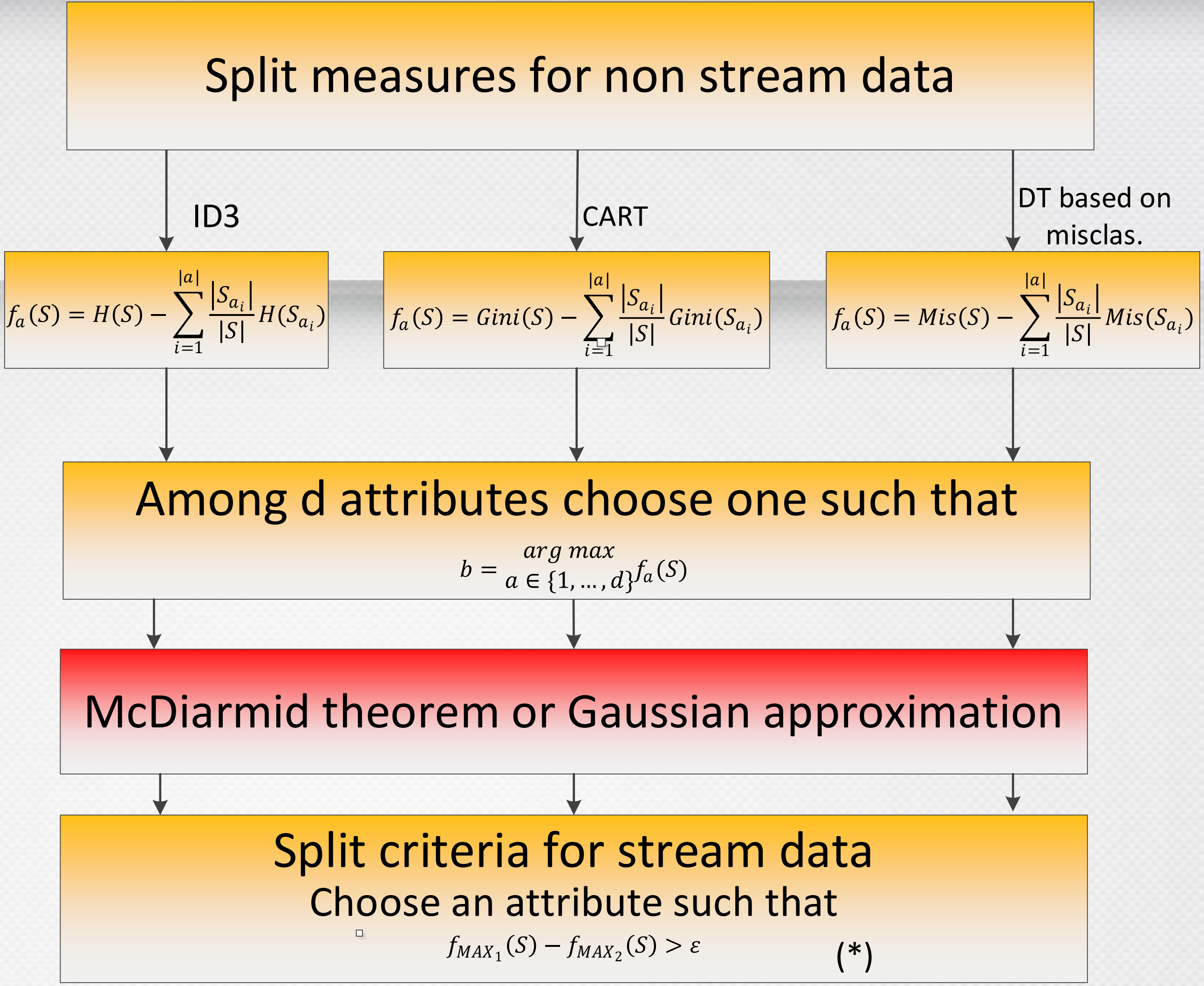
Leszek Rutkowski, Lena Pietruczuk, Piotr Duda, Maciej Jaworski, *Decision trees for mining data streams based on the McDiarmid's bound*, IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 6, pp. 1272–1279, 2013.

Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, Piotr Duda, *Decision trees for mining data streams based on the Gaussian approximation*, IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 1, pp. 108-119, Jan. 2014.

Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, Piotr Duda, *The CART decision tree for mining data streams*, Information Sciences, vol. 266, pp. 1 – 15, 2014.

Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, Piotr Duda, *A new method for data stream mining based on the misclassification error*, IEEE Transaction on Neural Networks and Learning Systems, vol. 26,PP 1048-1059, no. 5, 2015.

How to deal with stream data?



Split measures for non stream data

ID3

CART

DT based on misclas.

$f_a(S) = H(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} H(S_{a_i})$

$f_a(S) = Gini(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} Gini(S_{a_i})$

$f_a(S) = Mis(S) - \sum_{i=1}^{|a|} \frac{|S_{a_i}|}{|S|} Mis(S_{a_i})$

Among d attributes choose one such that

$$b = \underset{a \in \{1, \dots, d\}}{arg\ max} f_a(S)$$

McDiarmid theorem or Gaussian approximation

Split criteria for stream data

Choose an attribute such that

$$f_{MAX_1}(S) - f_{MAX_2}(S) > \varepsilon \qquad (*)$$

Under condition (*)

$$\boldsymbol{E\left[f_{a_{MAX_1}}\right] > E\left[f_{a_{MAX_2}}\right]}$$

with probability $(1 - \delta)^{d-1}$.

HOW TO DETERMINE THE VALUE OF ϵ ???

McDiarmid's inequality

Let $S = \{X_1, \ldots, X_N\}$ be the set of i.i.d. random variables, $X_i \in U_i$
Suppose that the (measurable) function
$\tilde{f}: \prod U_i \to \mathbb{R}$ satisfies

$$\sup_{X_1, \ldots, X_N, \widehat{X_i}} |\tilde{f}(X_1, \ldots, X_i, \ldots, X_N) - \tilde{f}(X_1, \ldots, \widehat{X_i}, \ldots X_N)| \le c_i,$$

for some constants $c_i, \mathrm{i} = 1, \ldots, \mathrm{N}$. Then

$$\Pr(\tilde{f}(\mathrm{S}) - \mathrm{E}|\tilde{f}(\mathrm{S})| \ge \varepsilon) \le \exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^{N} (c_i)^2}\right).$$

C. McDiarmid, On the Method of Bounded Differences, Surveys in Combinatorics, J. Siemons, ed., pp. 148-188, Cambridge Univ. Press, 1989

Application of the McDiarmid's inequality to stream data mining

The main result of our research is the following theorem stating that if the difference between the information gain estimates obtained for two attributes is greater than a specific value $\varepsilon(N, \delta)$, then with a fixed probability $1 - \delta$ there is, roughly speaking, a statistical difference between the expected values of information gain.

McDiarmid’s inequality - information gain

Theorem 1: Let $S = \{X_1, \dots, X_N\}$ be the set of independent random variables, with each of them taking values in the set $A_1 x \dots x A_d \times Y$. Then, for any fixed δ and any pair of attributes a and b, where $H(S|a) - H(S|b) > 0$, if

$$\varepsilon = C_{Gain}(K,N)\sqrt{\frac{\ln 1/\delta}{2N}}$$

where

$$C_{Gain}(K,N) = 6(K\log_2 eN + \log_2 2N) + 2\log_2 K$$

then

$$E[H(S|a)] > E[H(S|b)], \quad \text{with prob. } 1-\delta.$$

L. Rutkowski, L. Pietruczuk, P. Duda, M. Jaworski, *Decision trees for mining data streams based on the McDiarmid’s bound*, IEEE Transactions on Knowledge and Data Engineering, vol.25, no.6, pp.1272-1279, June 2013

McDiarmid’s inequality - Gini gain

Theorem 2: Let S= $\{X_1, \dots , X_N\}$ be the set of independent random variables, with each of them taking values in the set A_1 x … x $A_d \times Y$. Then, for any fixed δ and any pair of attributes a and b, where $Gini(S|a) - Gini(S|b) > 0$, if

$$\varepsilon = \sqrt{\frac{8\ln 1/\delta}{2N}}$$

then

$$E[Gini(S|a)] > E[Gini(S|b)],\ \text{with prob. } 1 - \delta.$$

L. Rutkowski, L. Pietruczuk, P. Duda, M. Jaworski, *Decision trees for mining data streams based on the McDiarmid’s bound*, IEEE Transactions on Knowledge and Data Engineering, vol.25, no.6, pp.1272-1279, June 2013

McDiarmid's inequality - Gini gain

Conclusion: If the number of data elements N satisfies the condition

$$\boxed{N > 64\frac{\ln 1/\delta}{2[Gini(S|a) - Gini(S|b)]}}$$

then the number of data elements is sufficient enough to say that attribute a is „better" (with probability $1 - \delta$ to make a split than attribute b.

Misclassification error

$$\boldsymbol{Mis(S) = 1 - \max_{i=1\ldots K}\{p_i\}}$$

or equivalently

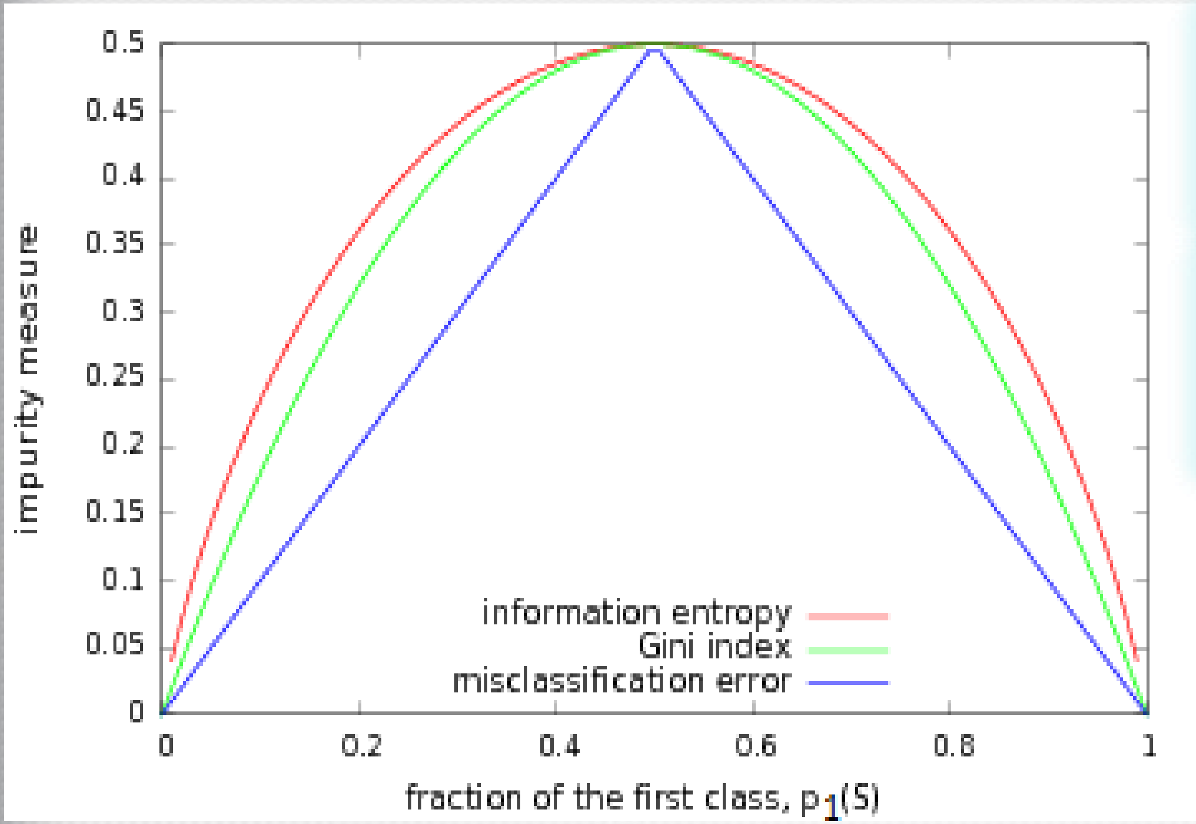
$$\boldsymbol{Mis(S) = 1 - \frac{\max_{i=1\ldots K}\{N^i\}}{N}}$$

where

$\boldsymbol{p_i}$ – probability that element belongs to the i-th class (proportion of elements in *S* belonging to the i-th class)

$\boldsymbol{N^i}$ - the number of data elements in S from the i-th class

Misclassification error



$$H(S) = -\sum_{i}^{K} p_i \log_2 p_i$$

$$Gini(S) = 1 - \sum_{i}^{K} (p_i)^2$$

$$Mis(S) = 1 - \max_{i=1\ldots K} \{p_i\}$$

Comparison for two-class problem

Split measure function based on the misclassification error

$$\boldsymbol{Mis(S|a) = Mis(S) - \sum_{i=1}^{|a|} \frac{|S_i|}{|S|} Mis(S_i)}$$

Analogously to the information gain:

$$\boldsymbol{H(S|a) = H(S) - \sum_{i=1}^{|a|} \frac{|S_i|}{|S|} H(S_i),}$$

And Gini gain:

$$\textbf{Gini}\boldsymbol{(S|a) = Gini(S) - \sum_{i=1}^{|a|} \frac{|S_i|}{|S|} Gini(S_i).}$$

Splitting criterion for misclassification-based split measure function – Gaussian approximation

Theorem 3: Let us consider two attributes a and b, for which the values of split measure function based on the misclassification error were calculated for set S. If the condition $\mathrm{Mis}(S|a) - Mis(S|b) > \varepsilon$ is satisfied, where

$$\boldsymbol{\varepsilon = z_{(1-\delta)} \frac{1}{\sqrt{2N}}},$$

$z_{(1-\delta)}$ is the $(1-\delta)$ –th quantile of the standard normal distribution $\mathcal{N}(0,1)$, then $\mathrm{E}[Mis(S|a)]$ is greater then $\mathrm{E}[Mis(S|b)]$ with probability $1-\delta$.

L. Rutkowski, M. Jaworski, L. Pietruczuk, P. Duda, A new method for data stream mining based on the misclassification error, IEEE Transaction on Neural Networks and Learning Systems, vol. 26,pp. 1048-1059, no. 5, 2015

Splitting criterion for misclassification-based split measure function – Hoeffding's inequality

Theorem 4: Let us consider two attributes a and b, for which we the values of split measure function based on the misclassification error were calculated for set S. If the condition $Mis(S|a) - Mis(S|b) > \varepsilon$ is satisfied, where

$$\boldsymbol{\varepsilon} = \sqrt{\frac{\mathbf{2\ ln\ 1/\delta}}{\boldsymbol{N}}},$$

then $\mathrm{E}[Mis(S|a)]$ is greater then $E[Mis(S|b)]$ with probability $1 - \delta$.

Remark. Theorem 4 is based on the Hoffeding's inequality!!!

Comparison of the newest results

Impurity measure Method	Information entropy	Gini index	Misclassification error
Hoeffding bound	$\varepsilon = \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$ Incorrectly obtained Domingos and Hulten, ACM 2000	$\varepsilon = \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$ Incorrectly obtained Domingos and Hulten, ACM 2000	$\varepsilon = \sqrt{\frac{2 \ln 1/\delta}{n}}$
McDiarmid bound	$\varepsilon = C_{Gain}(K,n)\sqrt{\frac{\ln 1/\delta}{2n}}$ $C_{Gain}(K,n) = 6(K\log_2 en + \log_2 2n) + 2\log_2 K$ Rutkowski et. al., IEEE Trans. on Knowledge and Data Engineering, 2013	$\varepsilon = 8\sqrt{\frac{\ln 1/\delta}{2n}}$ Rutkowski et. al., IEEE Trans. on Knowledge and Data Engineering, 2013	$\varepsilon = \sqrt{\frac{2 \ln 1/\delta}{n}}$
Gaussian approximation			$\varepsilon = z_{(1-\delta)}\sqrt{\frac{1}{2n}}$ Rutkowski et. al., IEEE Trans. on Neural Networks and Learning Systems, 2015

If $f_a(S) - f_b(S) > \varepsilon$,

then with probability $1 - \delta$ $E[f_a(S)] > E[f_b(S)]$

Hybrid Splitting Criteria

‚Single' Splitting Criteria

New result 2016

- Splitting criterion for Gini index:

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > \sqrt{\frac{8\,ln\left(\frac{1}{\delta}\right)}{n(Z)}},$$

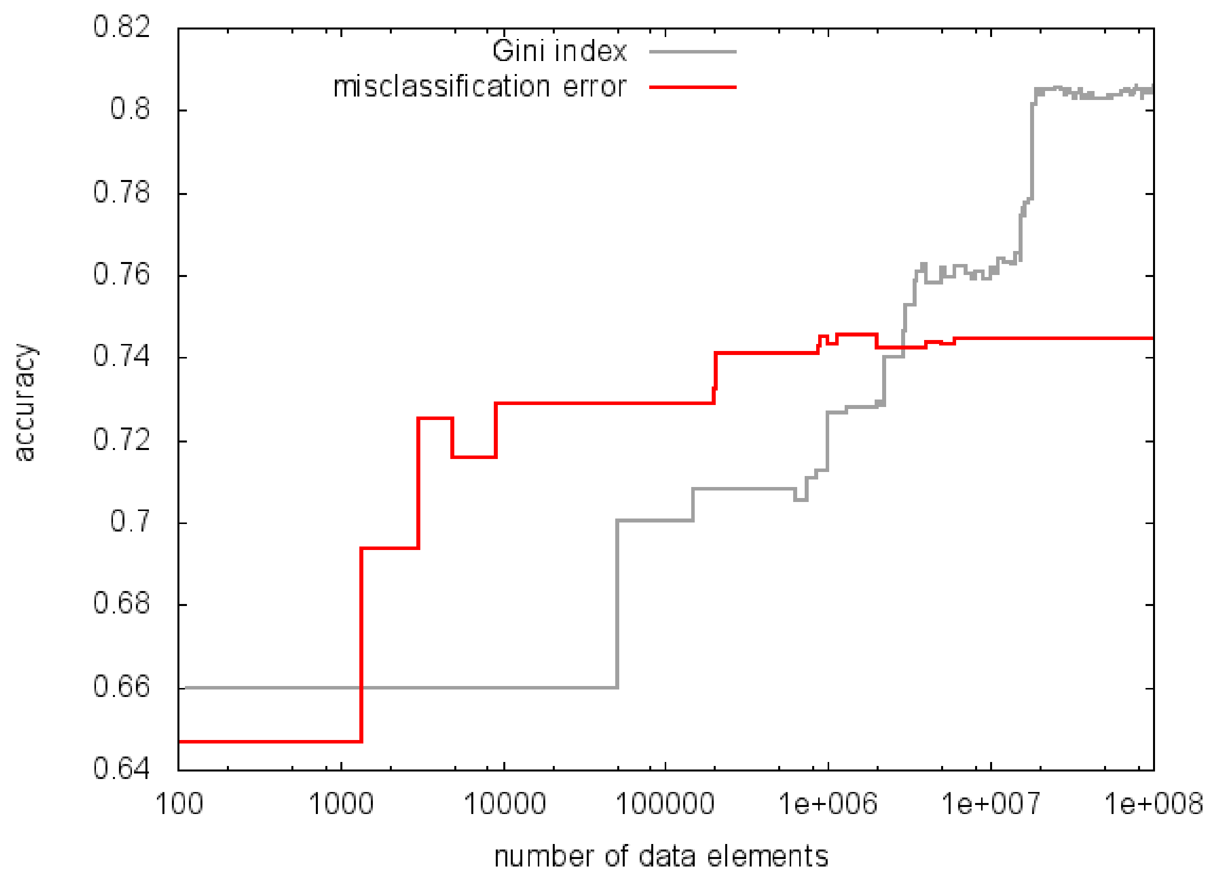
- Splitting criterion for mislcassification error:

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > z_{(1-\delta)}\sqrt{\frac{1}{2n(Z)}},$$

Experimental Results

New result 2016

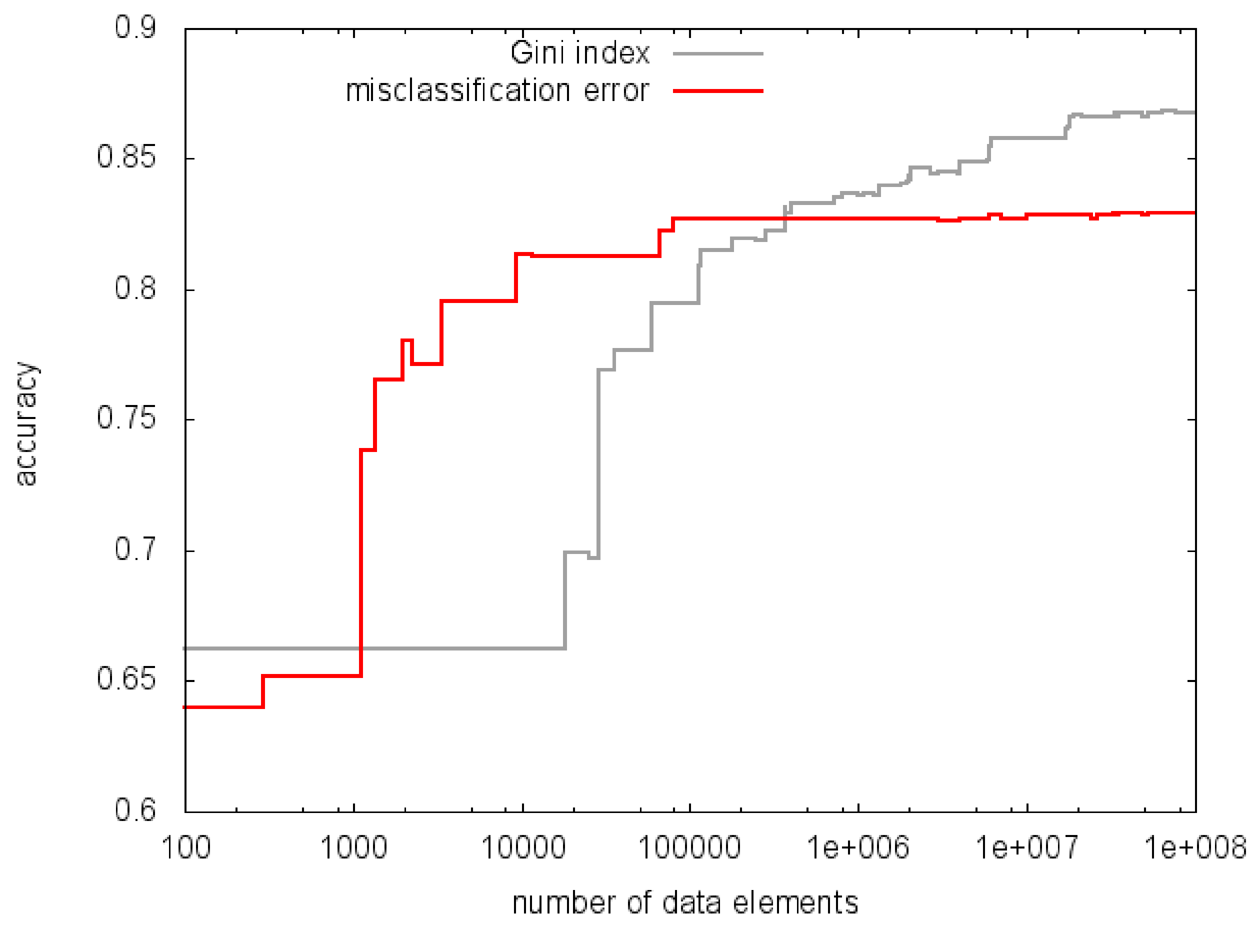
Accuracy vs numer of data elements: dataset no. 1



Experimental Results

New result 2016

Accuracy vs numer of data elements: dataset no. 2



Hybrid Splitting Criterion

New result 2016

$f^{G}{}_{i}(Z)$ - split measure for Gini index

$f^{M}{}_{i}(Z)$ - split measure for misclassification error

Hybrid criterion:

If:

Part 1

$$f^{G}{}_{a_{G,max}}(Z) - f^{G}{}_{a_{G,max2}}(Z) > \sqrt{\frac{8\ln\left(\frac{1}{\delta}\right)}{n(Z)}},$$

Choose the attribute with index $a_{G,max}$ to split the node.

Else, if:

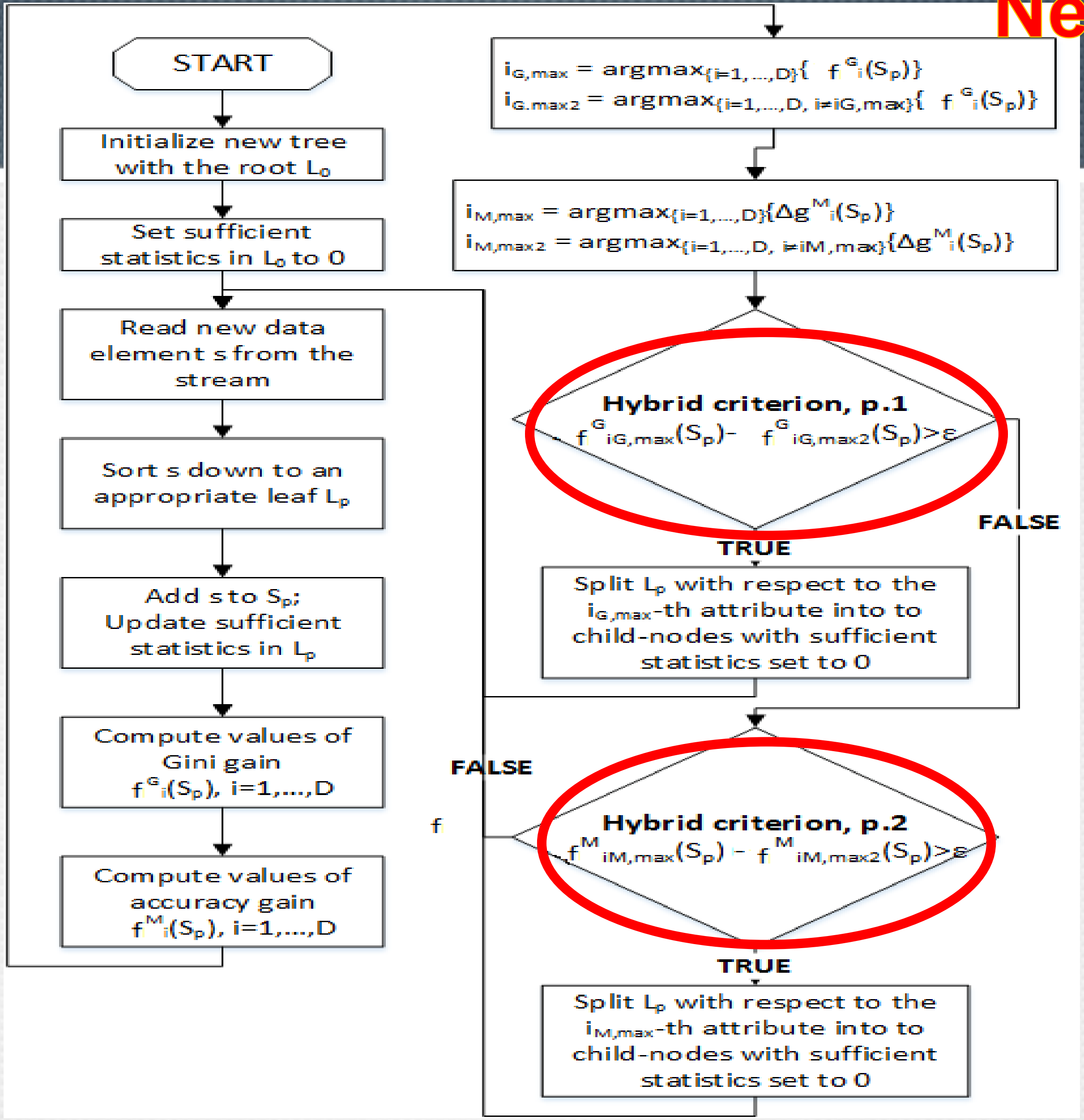
Part 2

$$f^{M}{}_{a_{M,max}}(Z) - f^{M}{}_{a_{M,max2}}(Z) > z_{(1-\delta)}\sqrt{\frac{1}{2n(Z)}},$$

Choose the attribute with index $a_{M,max}$ to split the node.

Online Decision Tree with hybrid splitting criterion

New result 2016



START

Initialize new tree with the root L_0

Set sufficient statistics in L_0 to 0

Read new data element s from the stream

Sort s down to an appropriate leaf L_p

Add s to S_p; Update sufficient statistics in L_p

Compute values of Gini gain $f^G_i(S_p)$, $i=1,\dots,D$

Compute values of accuracy gain $f^M_i(S_p)$, $i=1,\dots,D$

$i_{G,max} = \text{argmax}_{\{i=1,\dots,D\}}\{ f^G_i(S_p)\}$

$i_{G,max2} = \text{argmax}_{\{i=1,\dots,D,\ i\neq iG,max\}}\{ f^G_i(S_p)\}$

$i_{M,max} = \text{argmax}_{\{i=1,\dots,D\}}\{\Delta g^M_i(S_p)\}$

$i_{M,max2} = \text{argmax}_{\{i=1,\dots,D,\ i\neq iM,max\}}\{\Delta g^M_i(S_p)\}$

Hybrid criterion, p.1

$f^G_{iG,max}(S_p) - f^G_{iG,max2}(S_p) > \varepsilon$

TRUE

FALSE

Split L_p with respect to the $i_{G,max}$-th attribute into to child-nodes with sufficient statistics set to 0

FALSE

f

Hybrid criterion, p.2

$f^M_{iM,max}(S_p) - f^M_{iM,max2}(S_p) > \varepsilon$

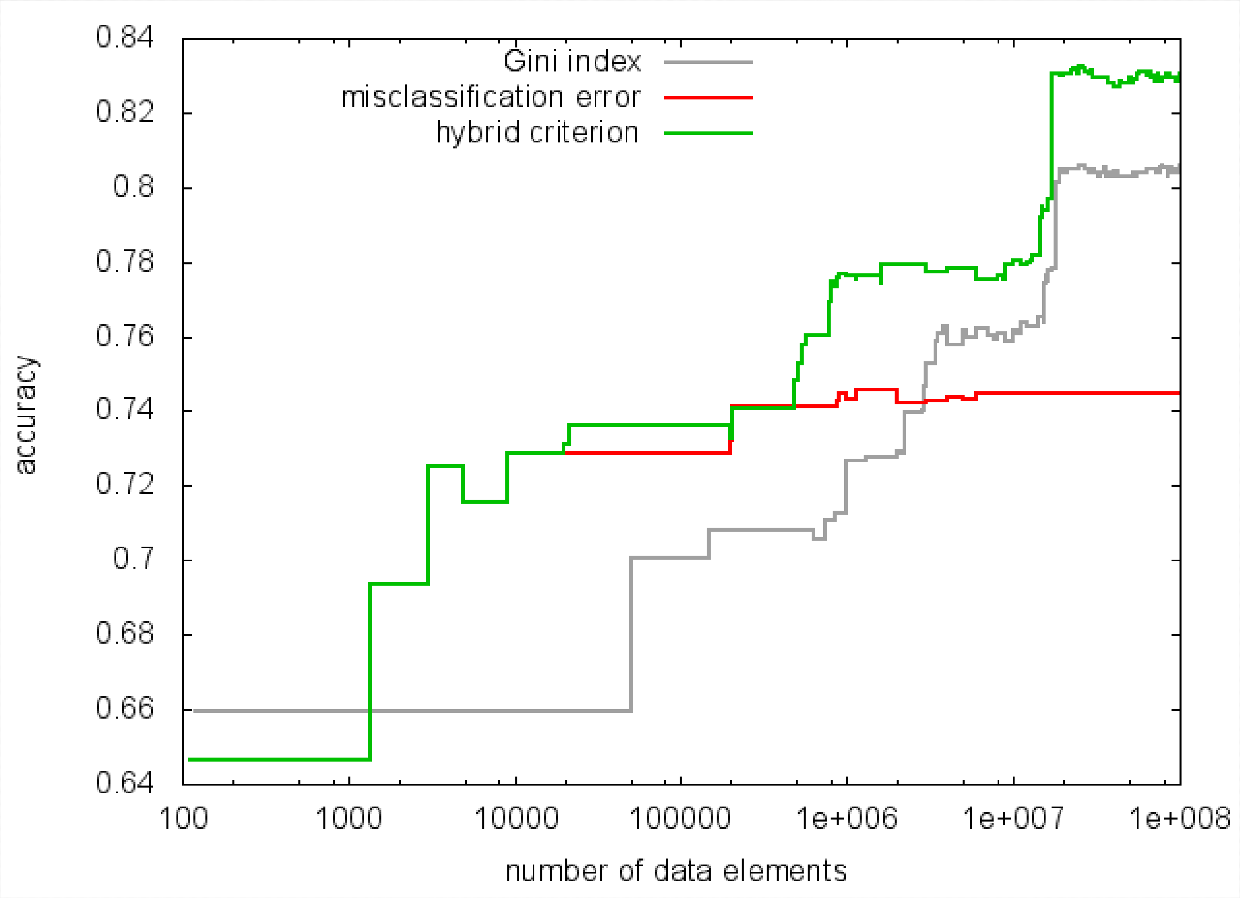
TRUE

Split L_p with respect to the $i_{M,max}$-th attribute into to child-nodes with sufficient statistics set to 0

Experimental Results

New result 2016

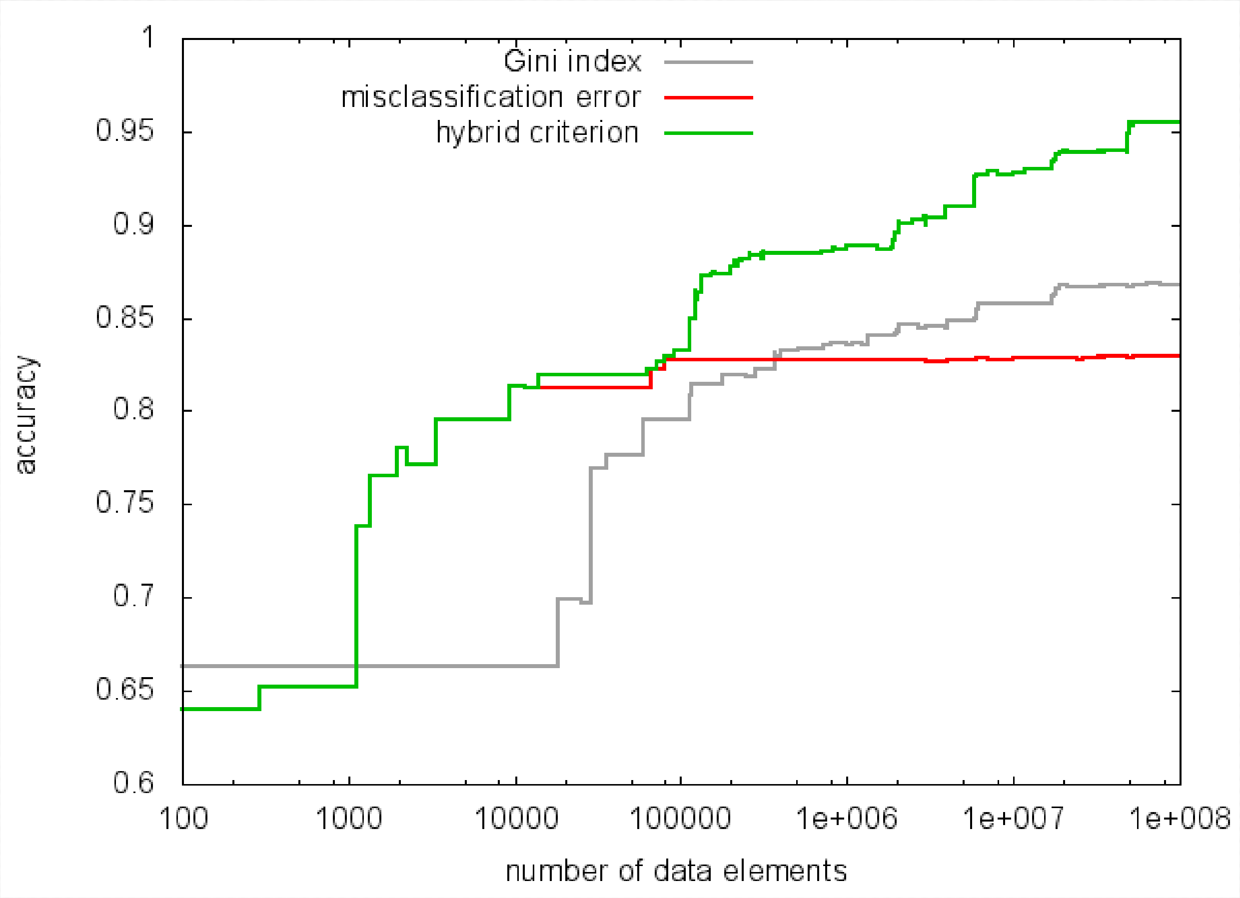
Accuracy vs numer of data elements: dataset no. 1



Experimental Results

New result 2016

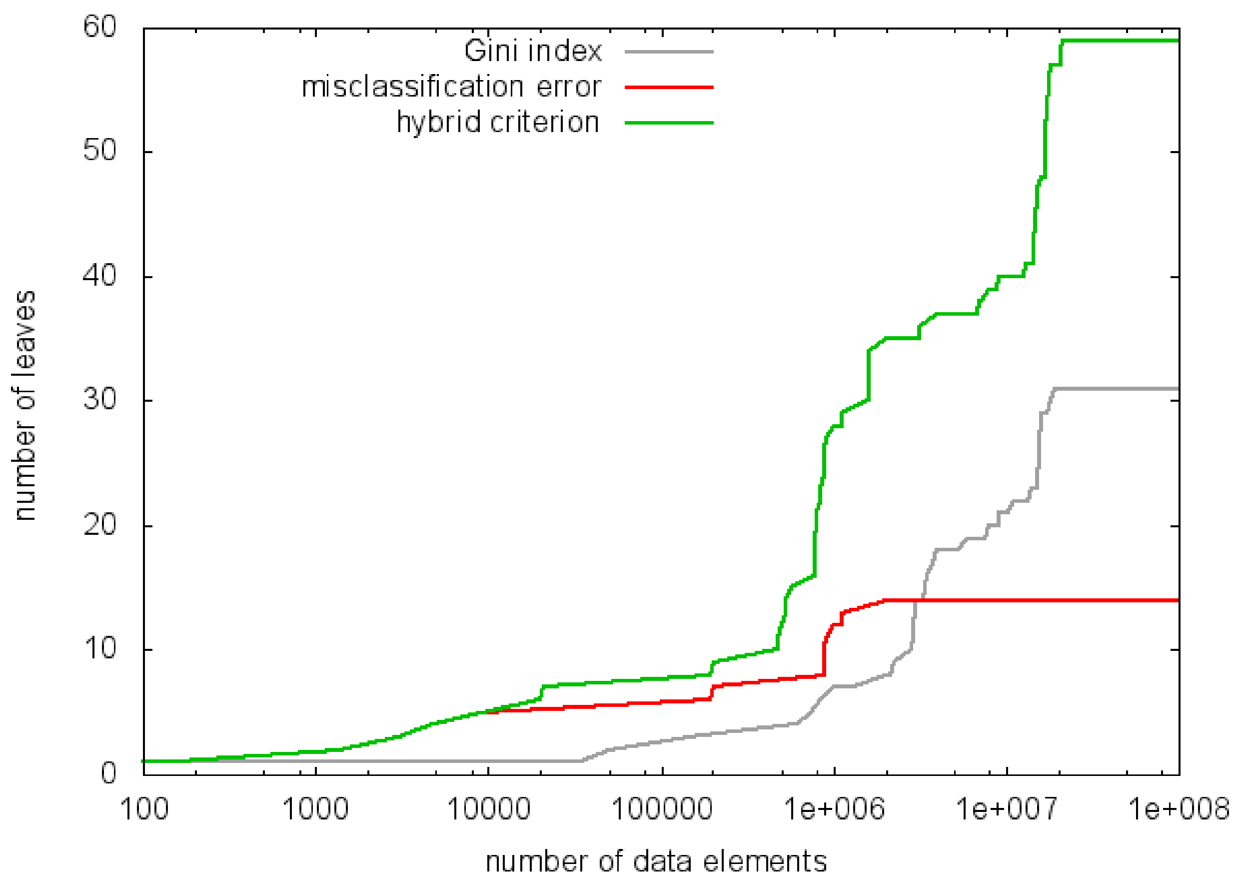
Accuracy vs numer of data elements: dataset no. 2



Experimental Results

New result 2016

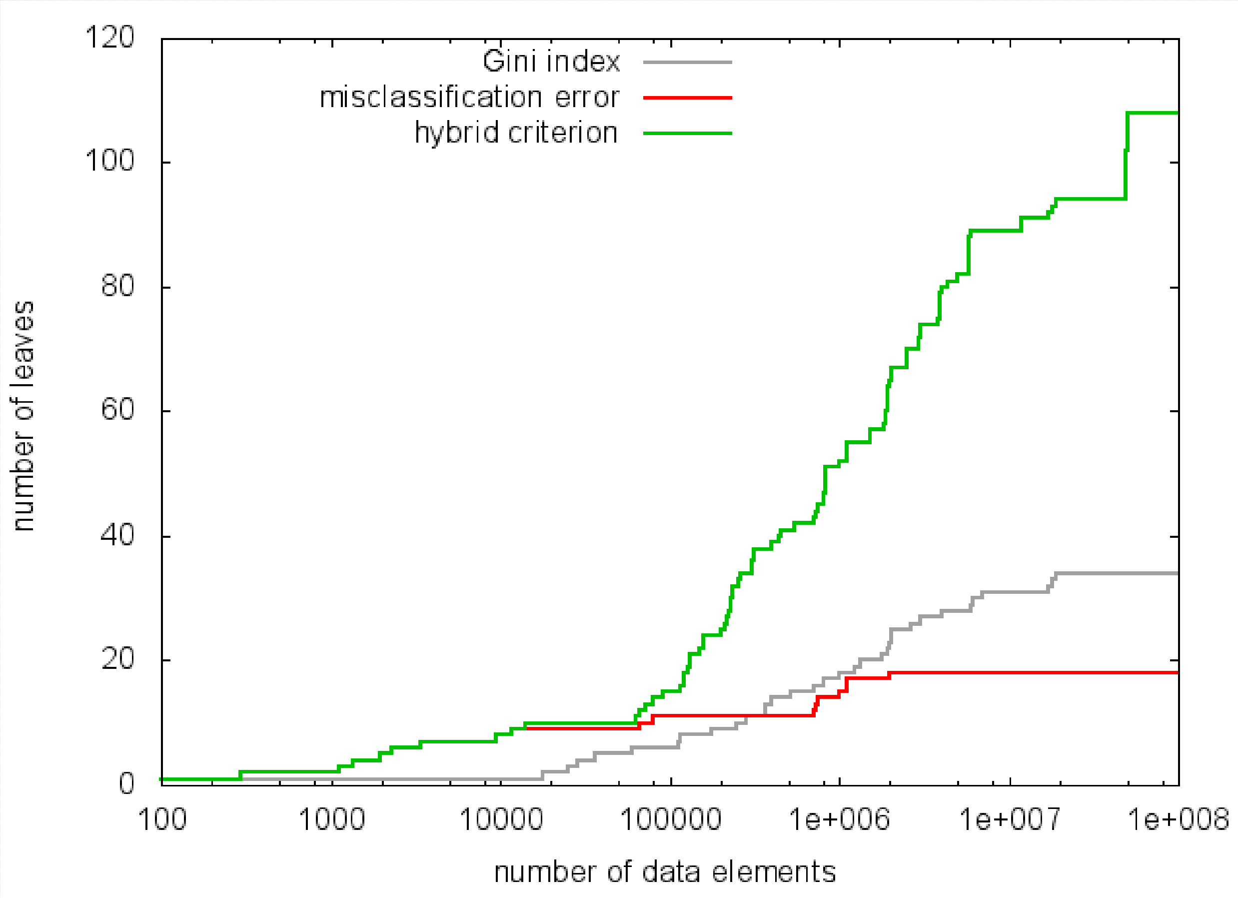
Number of leaves vs numer of data elements: dataset no. 1



Experimental Results

New result 2016

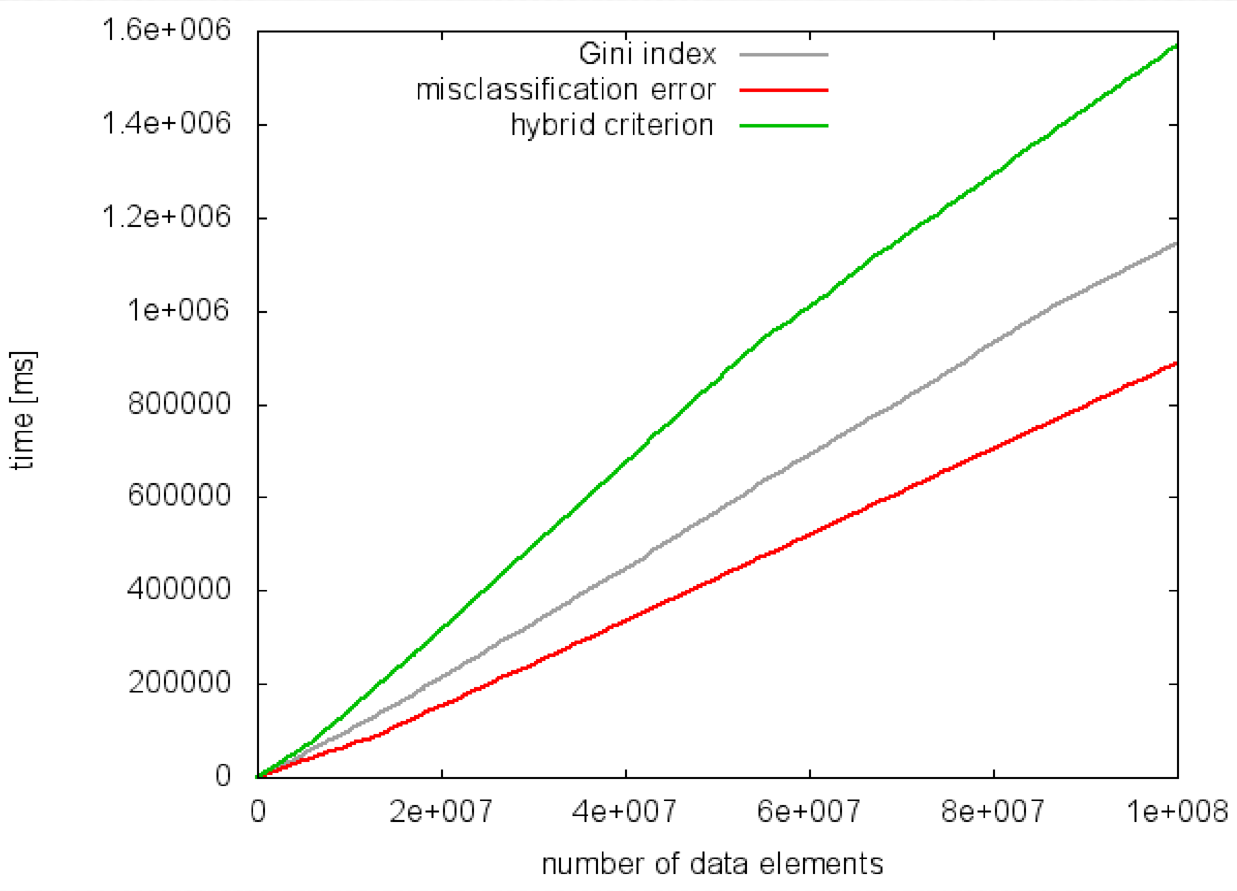
Number of leaves vs numer of data elements: dataset no. 2



Experimental Results

New result 2016

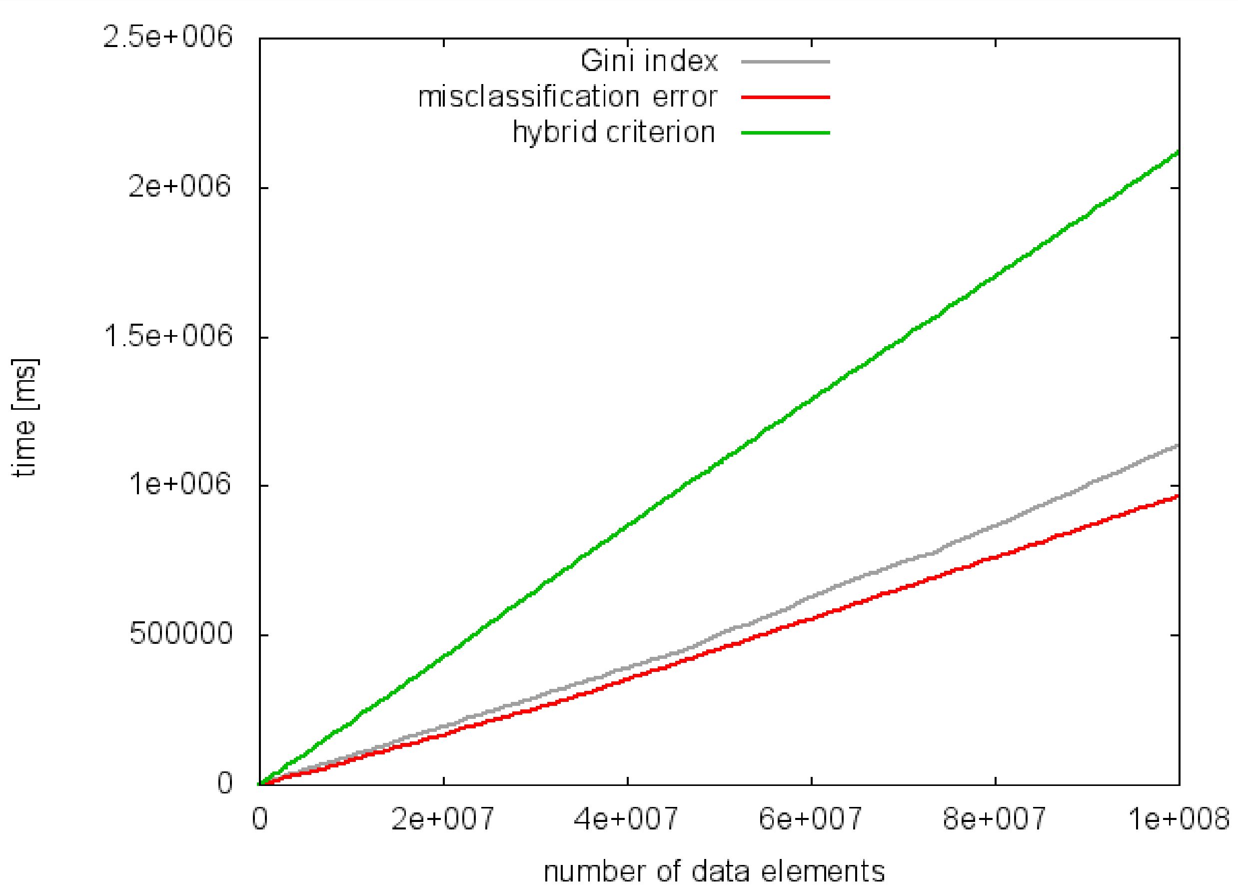
Processing time vs numer of data elements: dataset no. 2



Experimental Results

New result 2016

Processing time vs numer of data elements: dataset no. 2



New result 2016

More ‚single' splitting criteria

- Misclassification error + Hoeffding's inequality:

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > \sqrt{\frac{2\,ln\left(\frac{1}{\delta}\right)}{n(Z)}}$$

- Gini index + McDiarmid's bound + bias ([1])

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > \sqrt{\frac{8\,ln\left(\frac{1}{\delta}\right)}{n(Z)}} + \frac{8}{\sqrt{n(Z)}}$$

[1] R. De Rosa and N. Cesa-Bianchi Splitting with confidence in decision trees with application to stream data, *Proceedings of the International Joint Conference on Neural Networks*, 2015.

Online decision trees algorithms notations

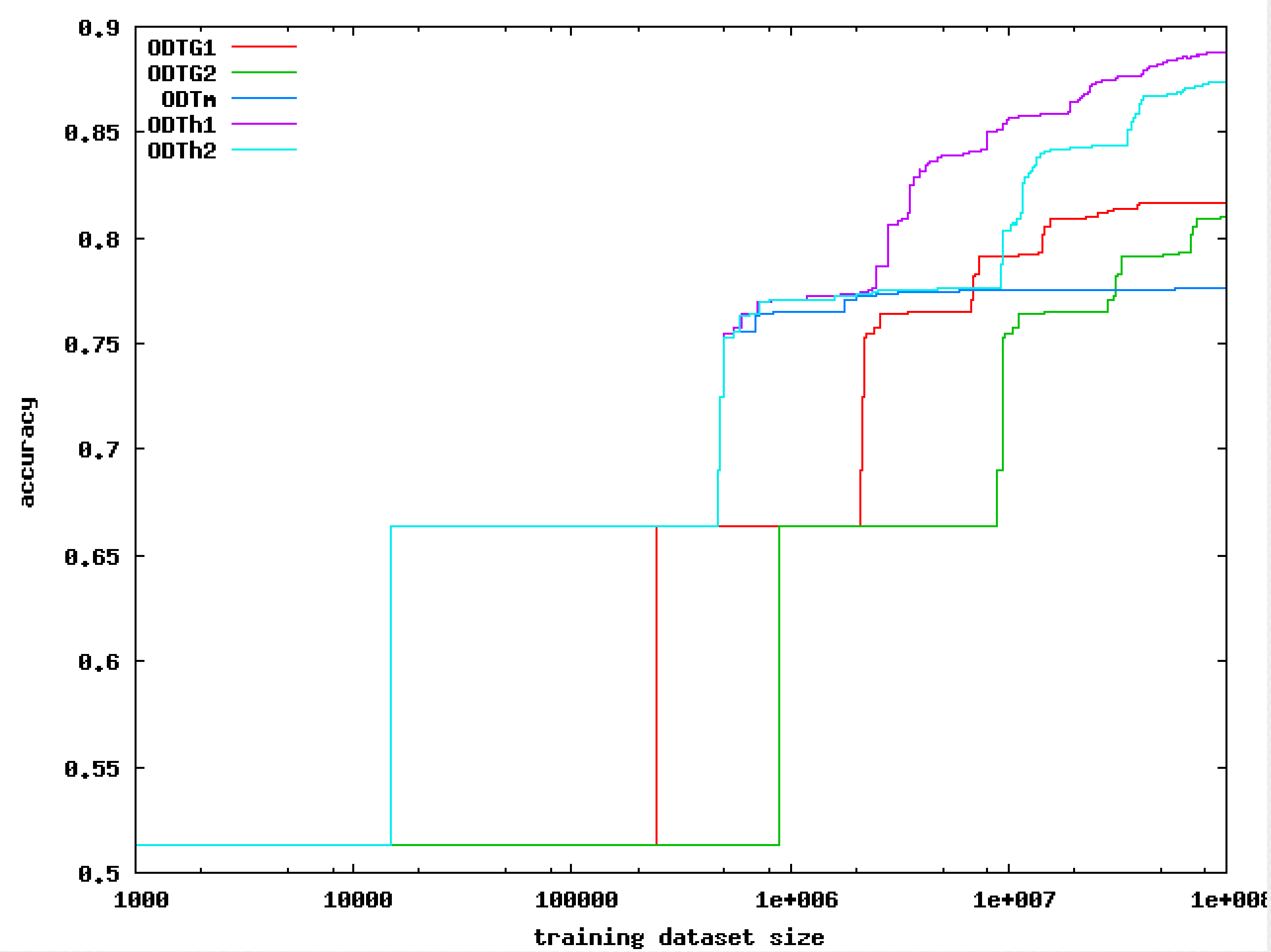
New result 2016

No.	Notation	Splitting criterion
1	ODTm	Misclassification error + Hoeffding’s inequality
2	OTDG1	Gini index + McDiarmid’s inequality
3	ODTG2	Gini index + McDiarmid’s inequality + bias
4	ODTh1	Hybrid: 1 + 2
5	ODTh2	Hybrid: 1 + 3

Experimental Results

New result 2016

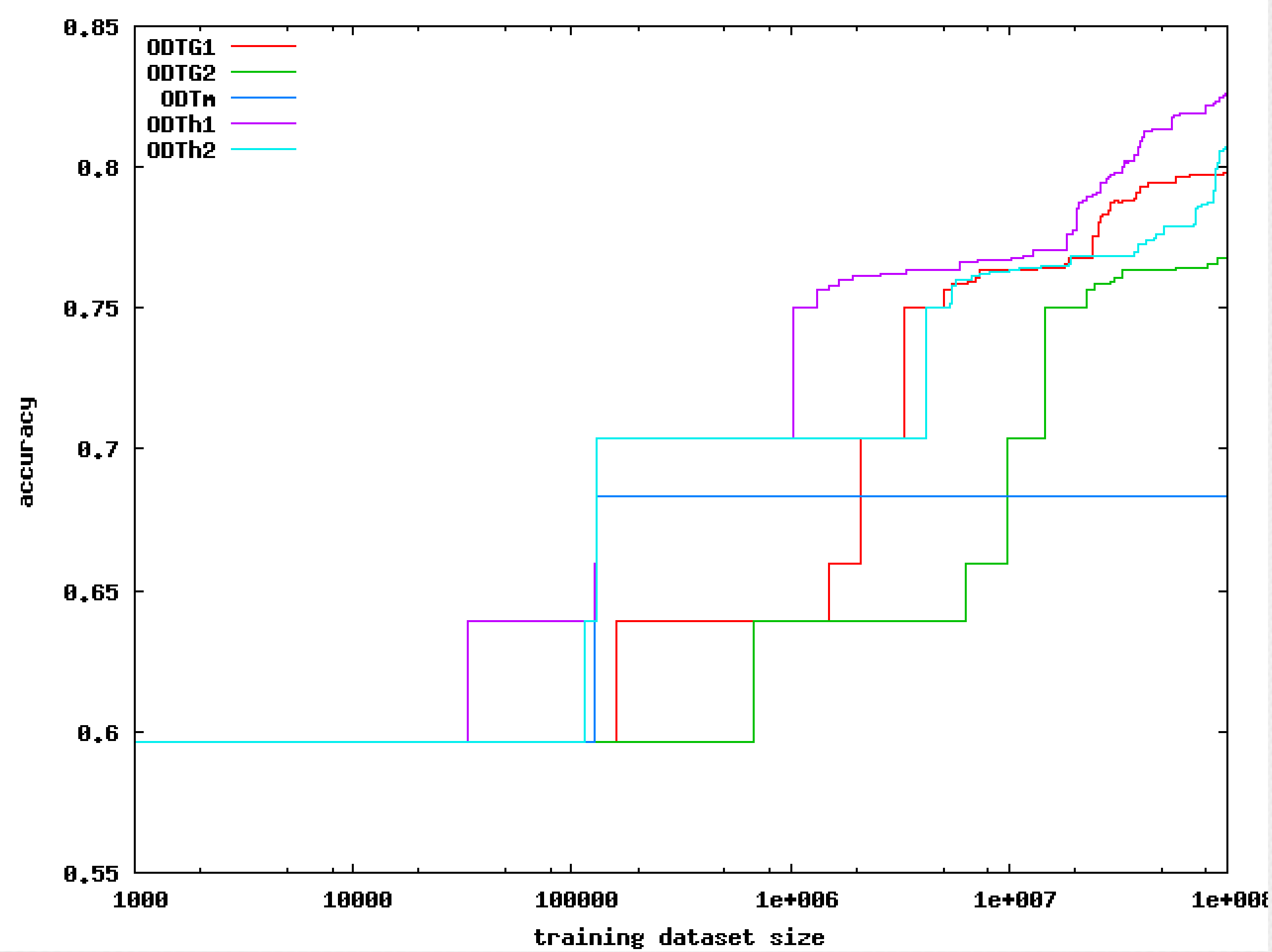
Accuracy vs numer of data elements: dataset no. 1



Experimental Results

New result 2016

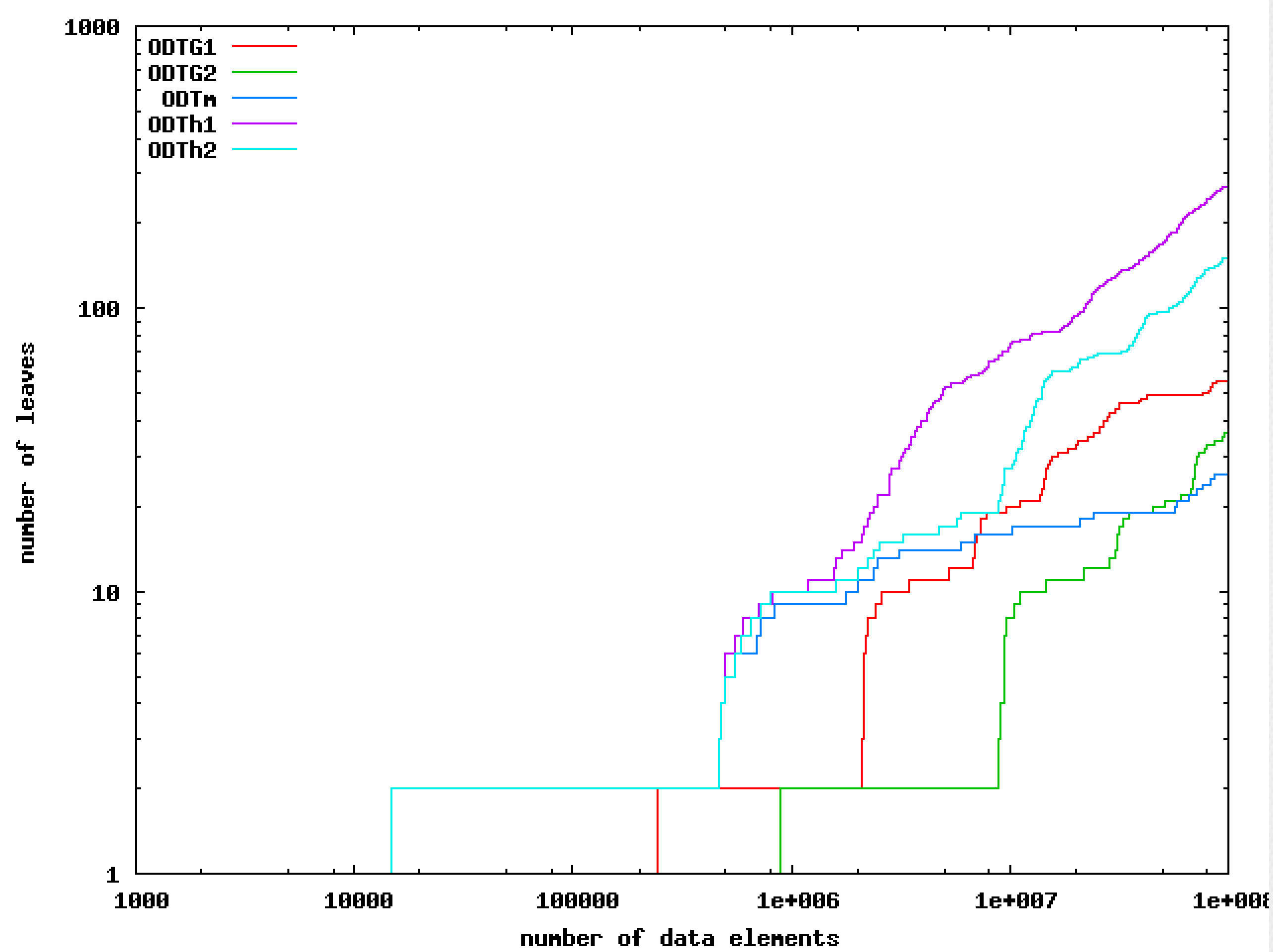
Accuracy vs numer of data elements: dataset no. 2



Experimental Results

New result 2016

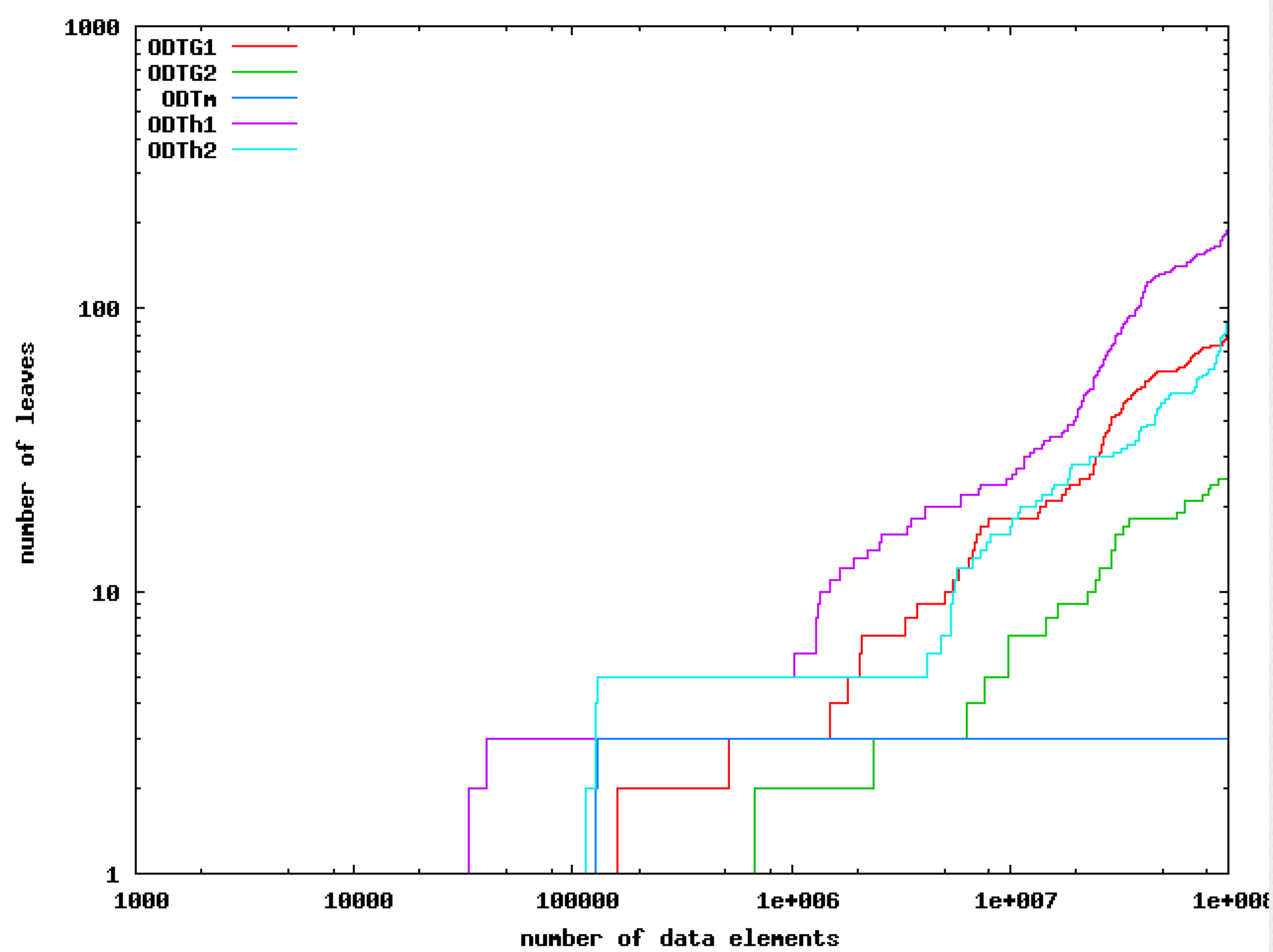
Number of leaves vs number data elements: dataset no. 1



Experimental Results

New result 2016

Number of leaves vs number data elements: dataset no. 2

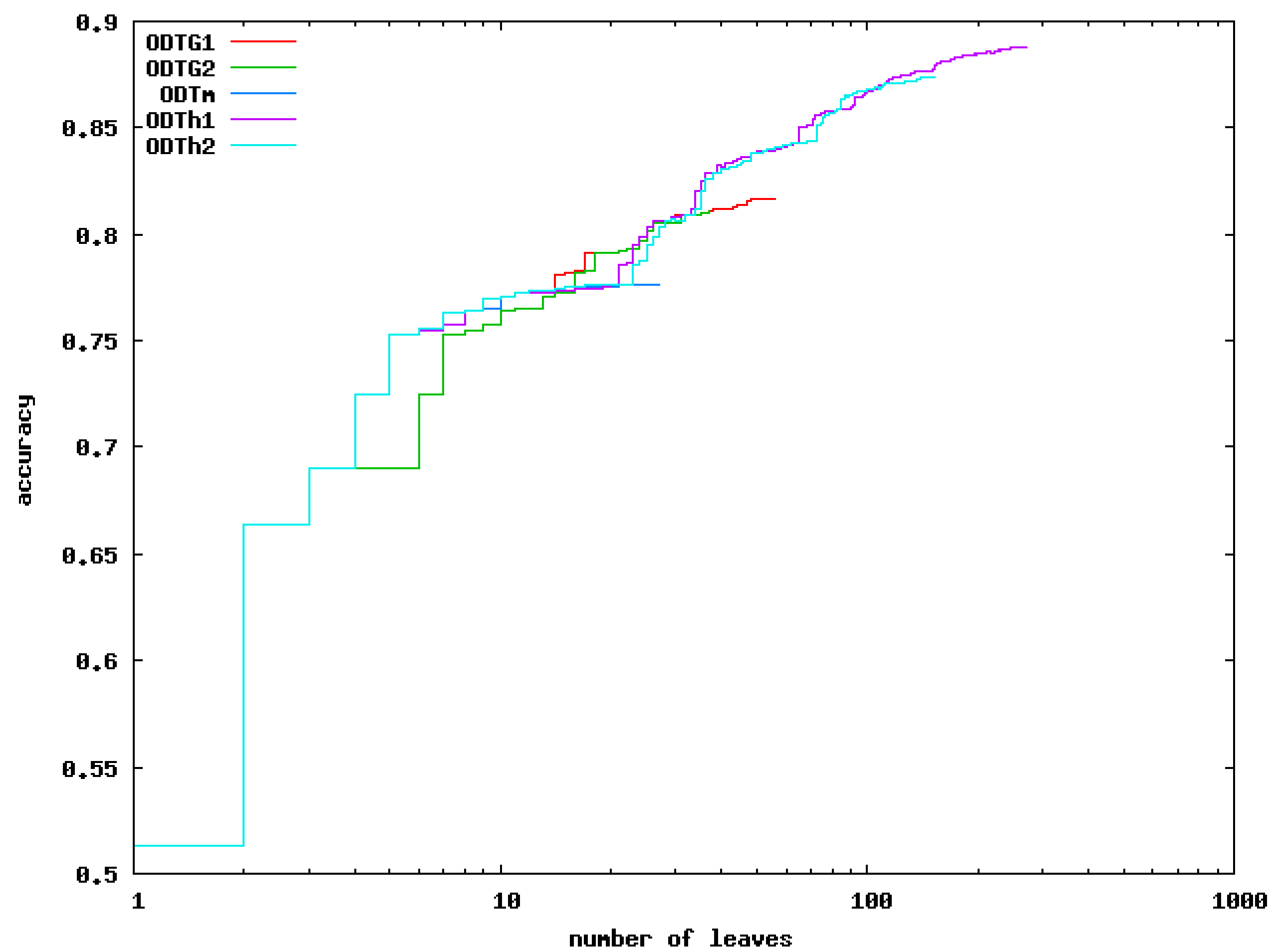


WCCI’ 2016 Tutorial, Vancouver, July 24, 2016

Experimental Results

New result 2016

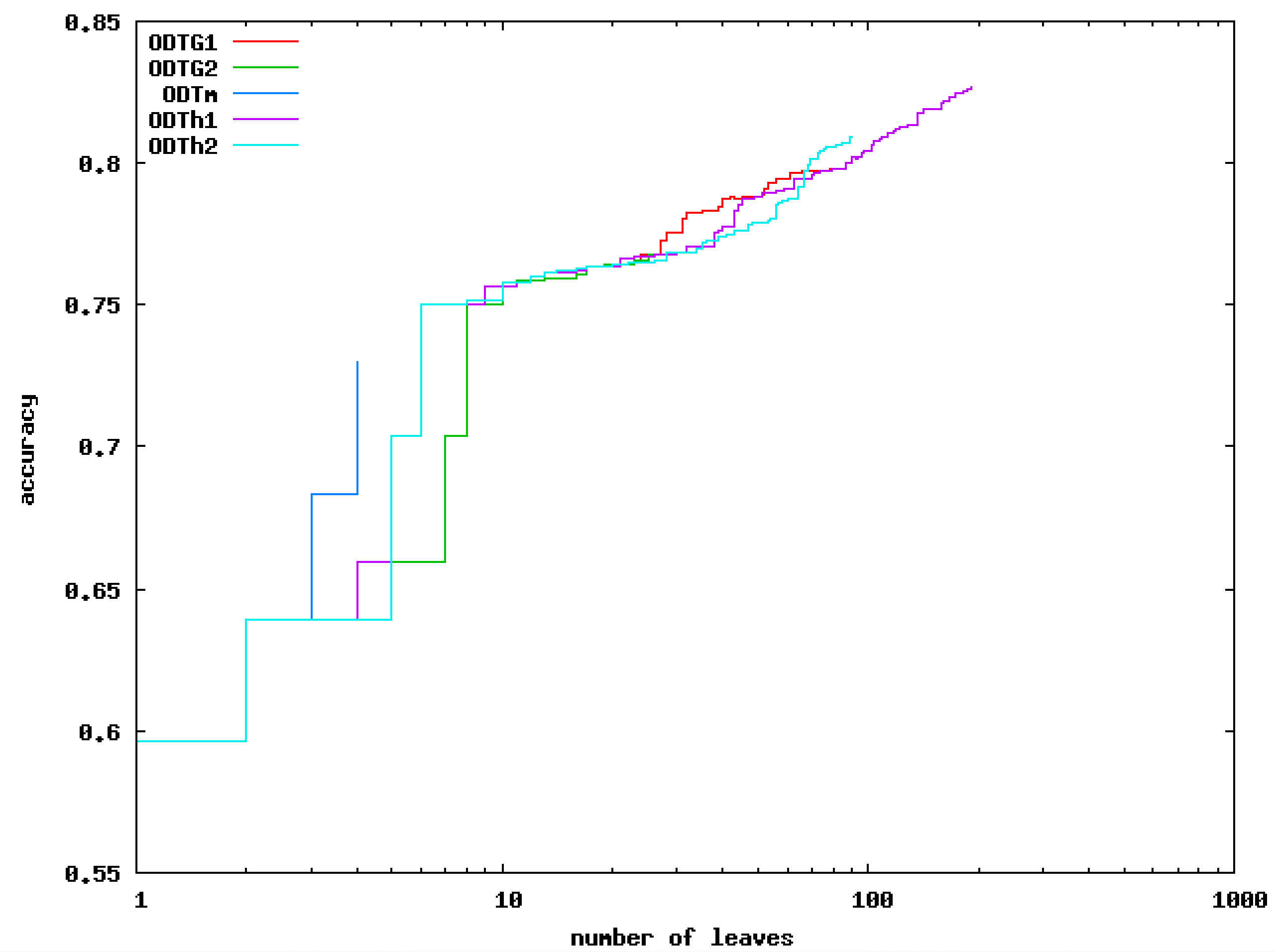
Accuracy vs numer of leaves: dataset no. 1



Experimental Results

New result 2016

Accuracy vs numer of leaves: dataset no. 2



Comparison with ‚Hoeffding Trees'

New result 2016

- HDT:

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > \sqrt{\frac{R\, ln\left(\frac{1}{\delta}\right)}{2n(Z)}}$$

(for Gini index R = 1)

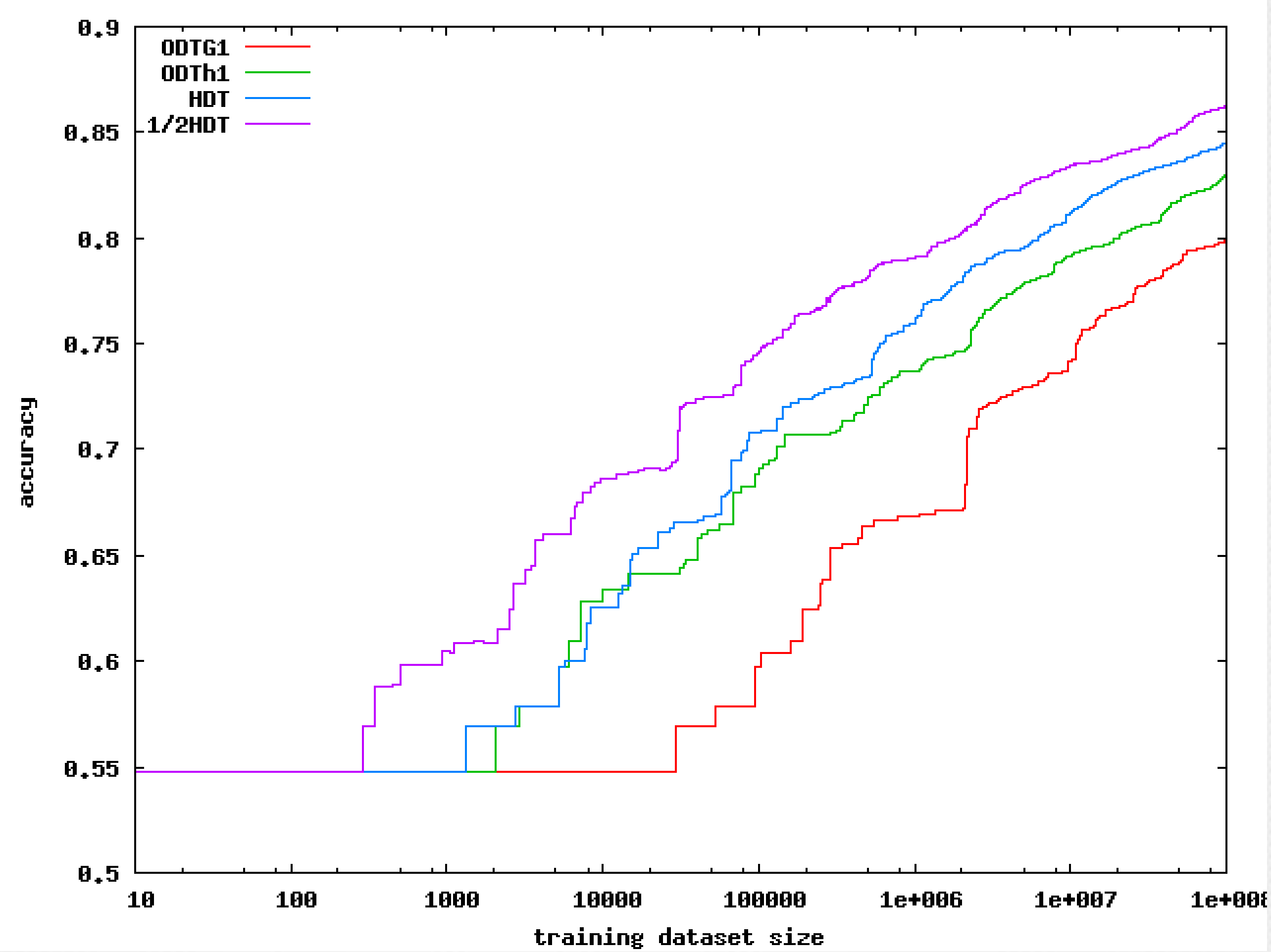
- 1/2HDT:

$$f_{a_{max}}(Z) - f_{a_{max2}}(Z) > 0.5\sqrt{\frac{R\, ln\left(\frac{1}{\delta}\right)}{2n(Z)}}$$

Experimental Results

New result 2016

Accuracy vs numer of data elements



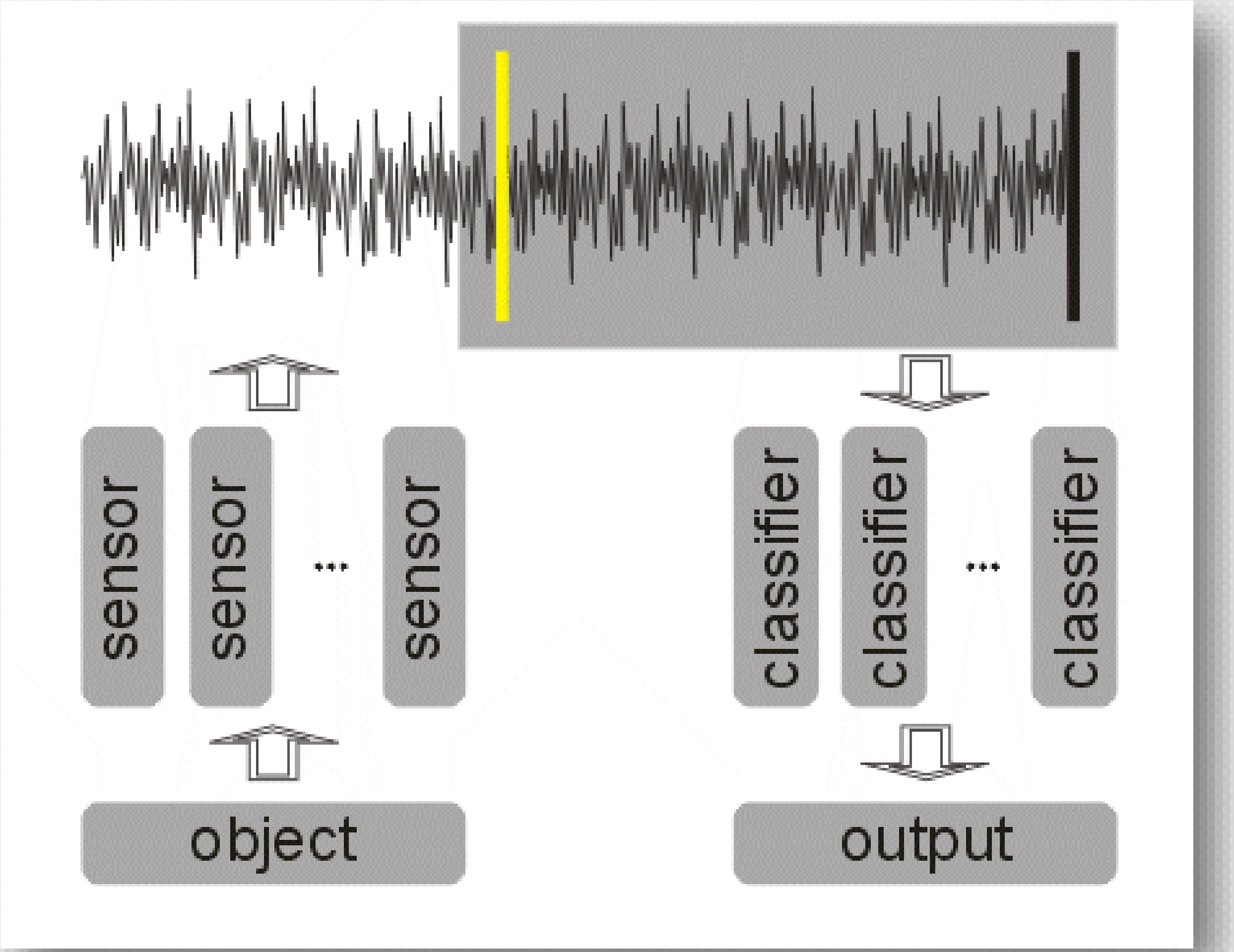
Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- **Ensemble methods for data streams (including new results 2016)**
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- References

Stream Data

- Ensemble methods

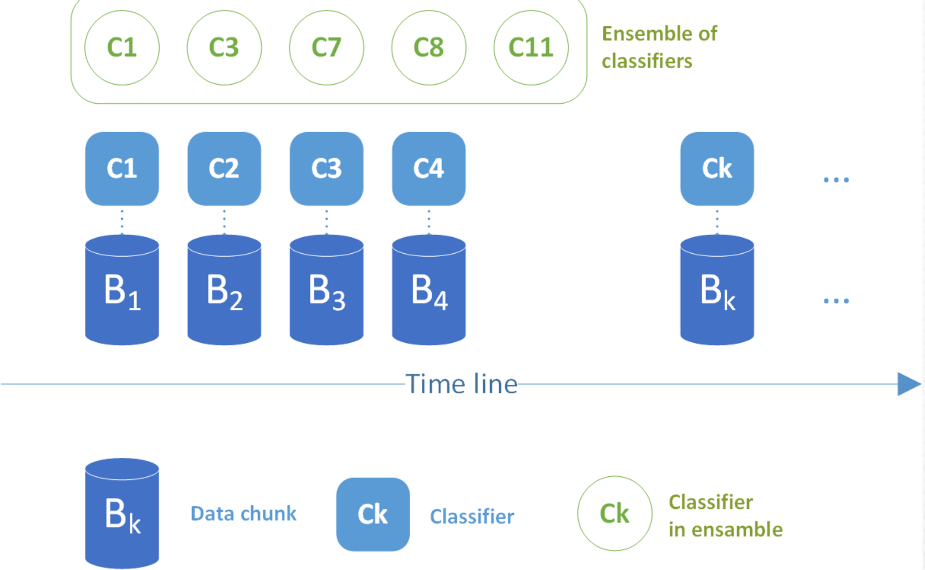


Stream Data

- Ensemble methods

Main steps of ensemble methods:

- Divide the data stream into equal sized chunks
- Train a new classifier based on current data chunk
- Keep the best *L* classifiers in the ensemble, e.g. *L*= 5



Stream Data

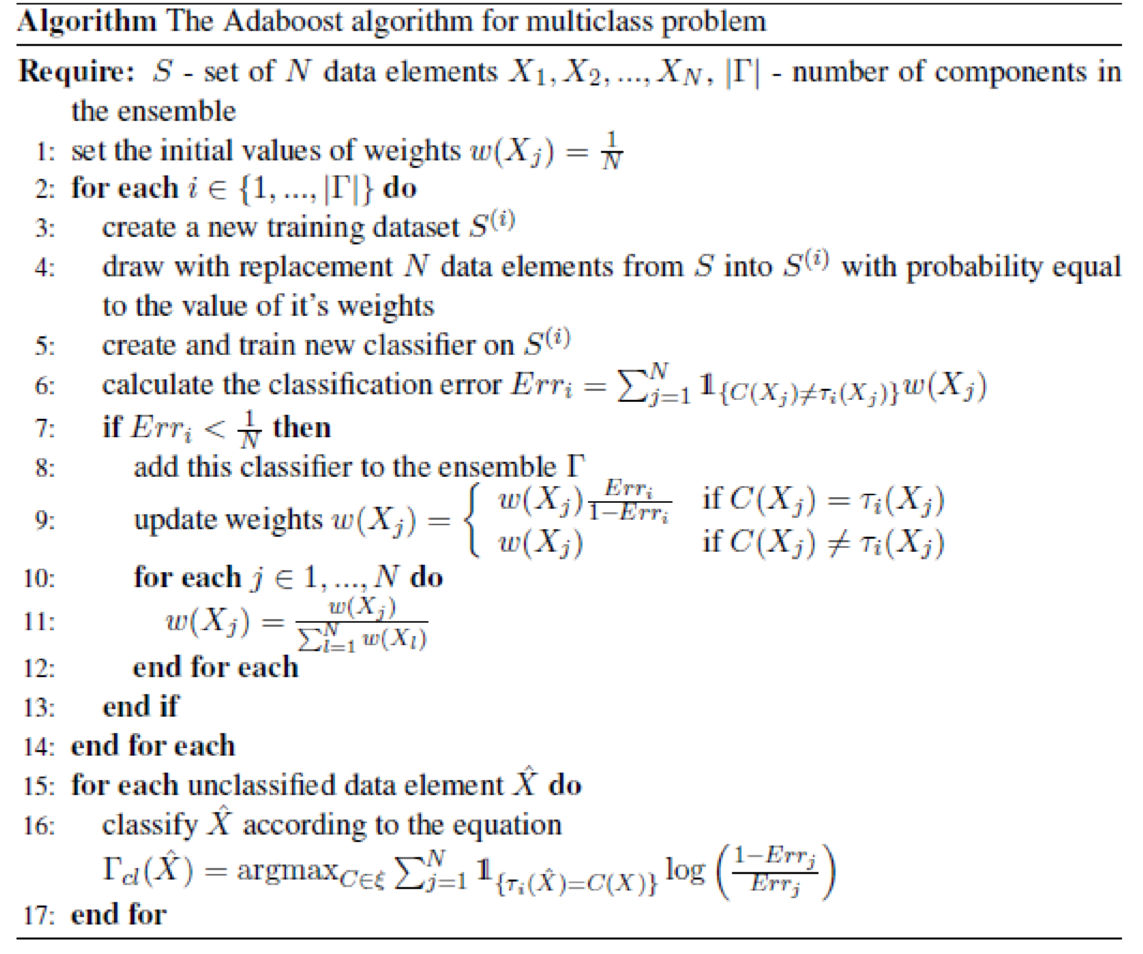
- Ensemble methods

Main questions with designing ensemble system for data stream mining:

- Which components to use?
- How to train the components?
- How big the ensemble should be?
- How to establish the usefulness of component to the ensemble?
- How to assign weights to the components?
- How to reduce the ensemble size?

Stream Data

- Ensemble methods (the Adaboost)



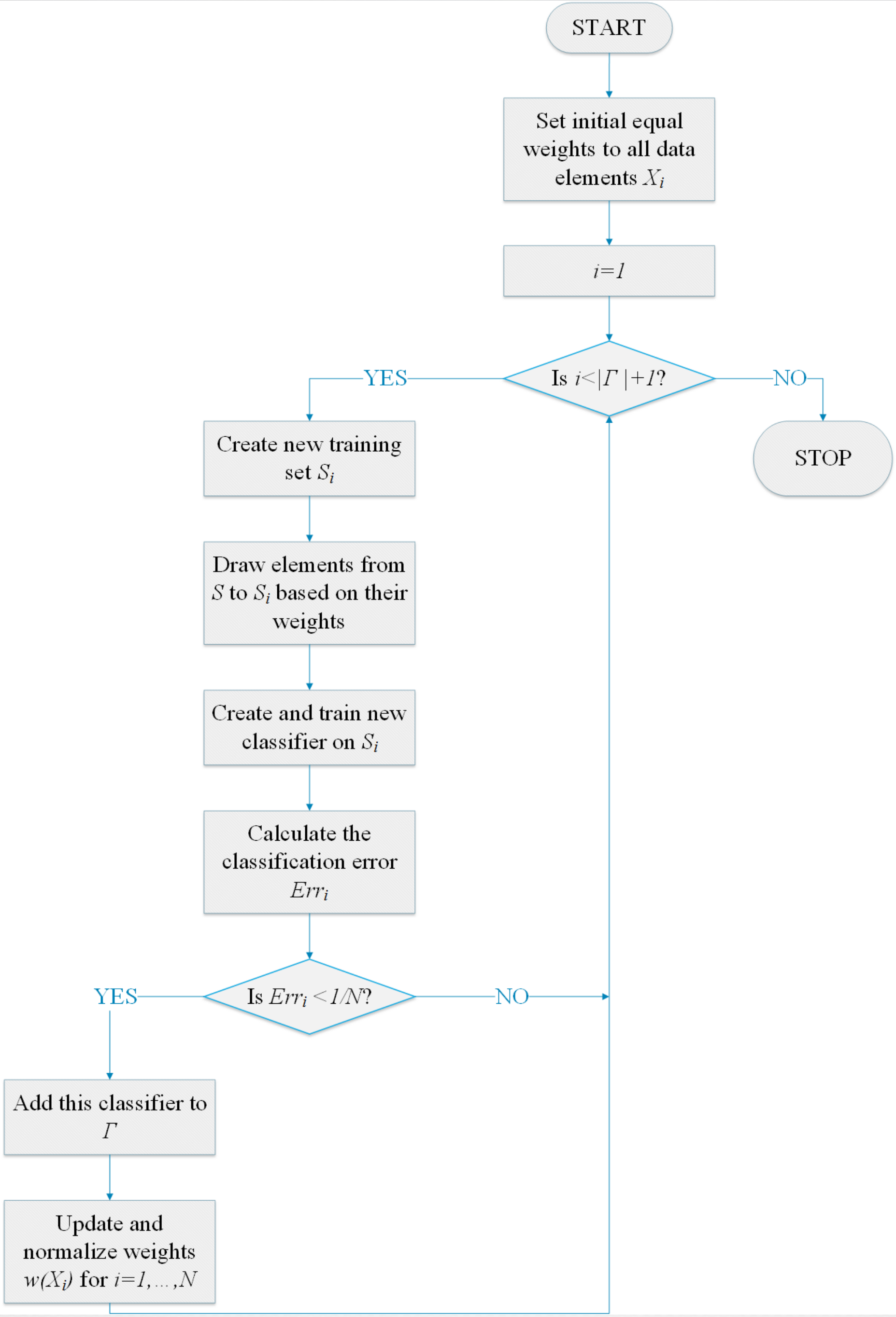
Algorithm The Adaboost algorithm for multiclass problem

```
Require: $S$ - set of $N$ data elements $X_1, X_2, ..., X_N$, $|\Gamma|$ - number of components in
    the ensemble
set the initial values of weights $w(X_j) = \frac{1}{N}$
for each $i \in \{1, ..., |\Gamma|\}$ do
    create a new training dataset $S^{(i)}$
    draw with replacement $N$ data elements from $S$ into $S^{(i)}$ with probability equal
    to the value of it's weights
    create and train new classifier on $S^{(i)}$
    calculate the classification error $Err_i = \sum_{j=1}^{N} \mathbb{1}_{\{C(X_j) \neq \tau_i(X_j)\}} w(X_j)$
    if $Err_i < \frac{1}{N}$ then
        add this classifier to the ensemble $\Gamma$
        update weights $w(X_j) = \begin{cases} w(X_j)\frac{Err_i}{1-Err_i} & \text{if } C(X_j) = \tau_i(X_j) \\ w(X_j) & \text{if } C(X_j) \neq \tau_i(X_j) \end{cases}$
        for each $j \in 1, ..., N$ do
            $w(X_j) = \frac{w(X_j)}{\sum_{l=1}^{N} w(X_l)}$
        end for each
    end if
end for each
for each unclassified data element $\hat{X}$ do
    classify $\hat{X}$ according to the equation
    $\Gamma_{cl}(\hat{X}) = \text{argmax}_{C \in \xi} \sum_{j=1}^{N} \mathbb{1}_{\{\tau_i(\hat{X}) = C(X)\}} \log\left(\frac{1-Err_j}{Err_j}\right)$
end for
```

* Yoav Freund and Robert E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119 –139, 1997.

Stream Data
- Ensemble methods

the Adabost

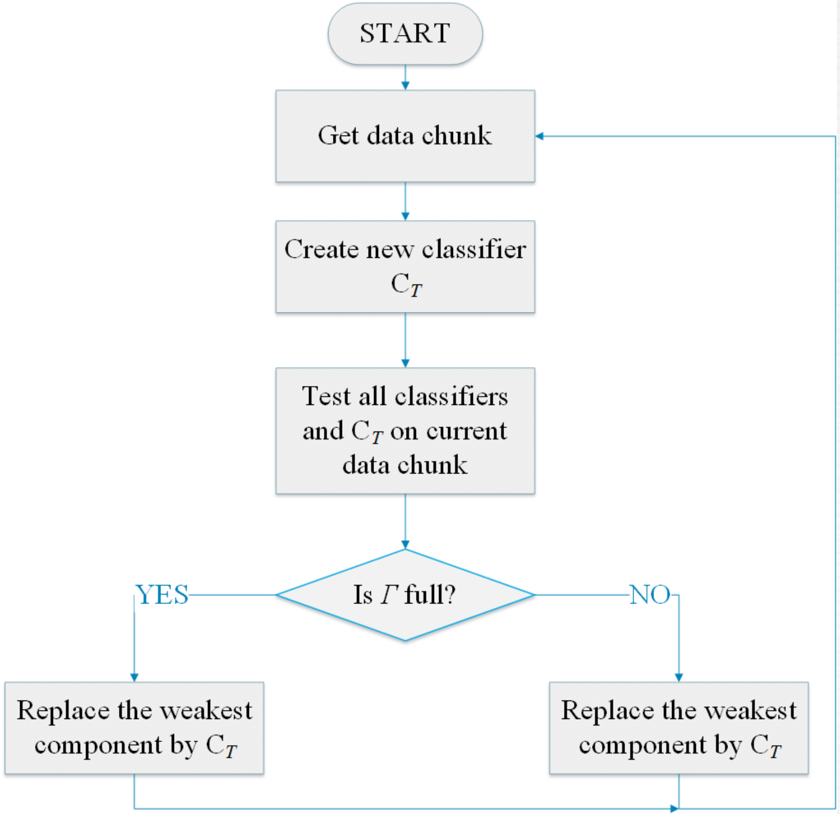


* Yoav Freund and Robert E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119 –139, 1997.

WCCI’ 2016 Tutorial, Vancouver, July 24, 2016

Stream Data

- Ensemble methods (the SEA)



P_1 denotes the percentage of votes for the most often occurring class,
P_2 denotes the percentage of votes obtained by the second most frequently occurring class,
P_C denotes the percentage obtained by the class of the correct class
P_T denotes the percentage of the prediction of the new classifier.

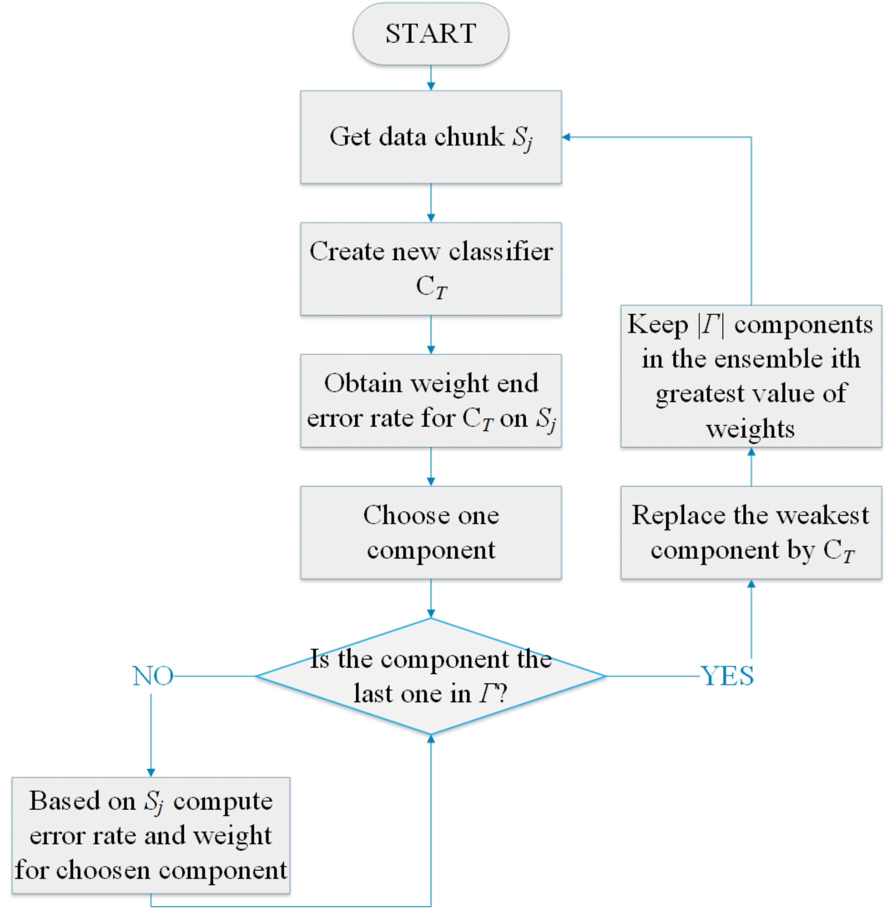
- the ensemble and the new tree give correct answers: the quality measure is increased by $1 - |P_1 - P_2|$
- the new tree gives correct answer but the ensemble is wrong: the quality measure is increased by $1 - |P_1 - P_C|$
- the tree gives wrong answer: the quality measure is decreased by $1 - |P_C - P_T|$

The obtained values of weights are used to determine if the newly created classifier should replace the least efficiently performing algorithm in the ensemble.

* W. Nick Street and Yong Seog Kim, A streaming ensemble algorithm (sea) for large-scale classification, In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’01, pp. 377–382, New York, NY, USA, 2001, ACM.

Stream Data

- Ensemble methods (the AWE)



Accuracy Weighted Ensemble (AWE)

to each component in the ensemble the weight is assigned $\mathcal{W}(\tau_j) = MSE_r - MSE_j$ where

$$MSE_j = \frac{1}{\left|S^{(j)}\right|} \sum_{X_k \in S^{(j)}} \left(1 - \omega(\tau_j, C)\right)^2,$$

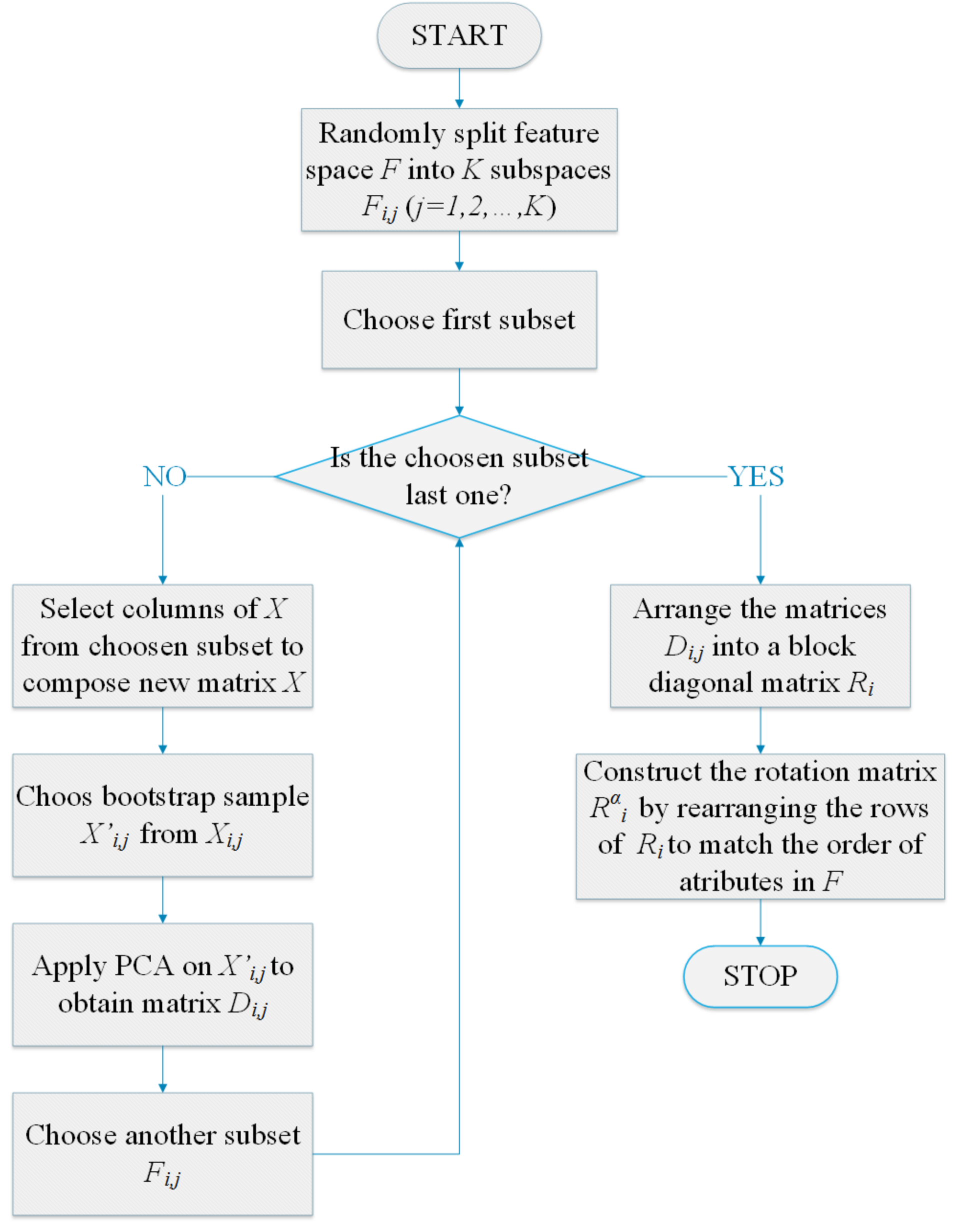
$$MSE_r = \sum_{C} \omega(C)\left(1 - \omega(C)\right)^2,$$

where $\left|S^{(j)}\right|$ denotes the number of data in the j-th chunk, $\omega(\tau_j, C)$ denotes the probability that classifier i will assign a correct class to data element and $\omega(C)$ denotes the probability of class C. In this case, MSE_r is the mean square error of randomly predicting classifier which defined the boundary for assessment of the minimal accuracy that the components in the ensemble should have. The members with lower accuracy are proven to decrease the performance of the whole ensemble method.

* Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, pp. 226–235, New York, NY, USA, 2003. ACM.

Stream Data
- Ensemble methods

the Rotation Forest



START

Randomly split feature space F into K subspaces $F_{i,j}$ $(j=1,2,\ldots,K)$

Choose first subset

Is the choosen subset last one?

NO

Select columns of X from choosen subset to compose new matrix X

Choos bootstrap sample $X'_{i,j}$ from $X_{i,j}$

Apply PCA on $X'_{i,j}$ to obtain matrix $D_{i,j}$

Choose another subset $F_{i,j}$

YES

Arrange the matrices $D_{i,j}$ into a block diagonal matrix R_i

Construct the rotation matrix R^{α}_{i} by rearranging the rows of R_i to match the order of atributes in F

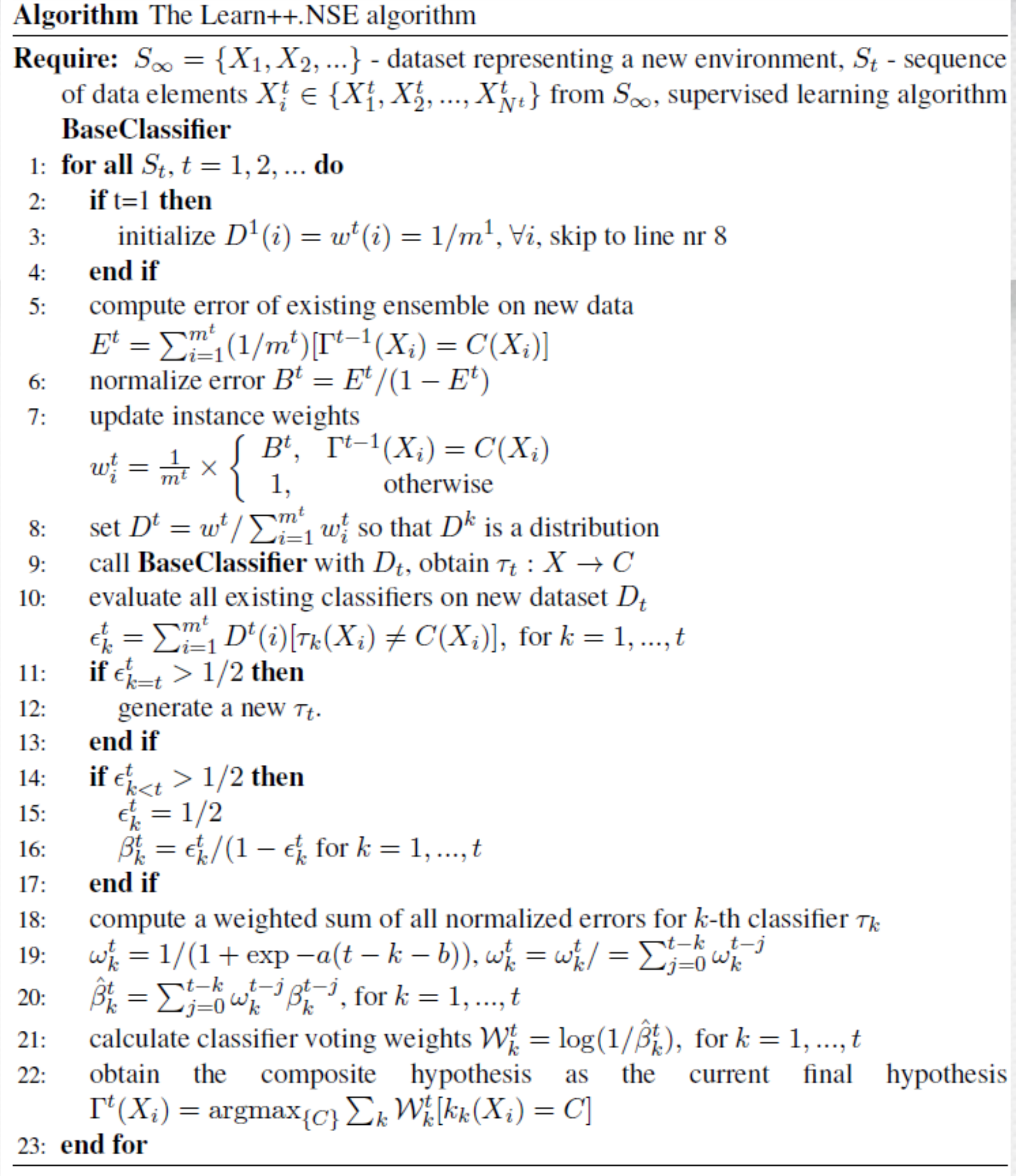
STOP

The main idea is to obtain different classifiers based on the sets of data that are transformed by the PCA algorithm. For construction of each decision tree a different mixture of features is taken into account and the principal component analysis (PCA) is applied on those sets. A new classifier is then trained based on data elements transformed linearly into new feature space. The diversity is obtained through the use of various extracted features which are the result of choosing different mixtures of feature components.

* Juan J. Rodriguez, Ludmila I. Kuncheva, and Carlos J. Alonso. Rotation forest: A new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 10, pp. 1619–1630, Oct 2006.

Stream Data
- Ensemble methods

the Learn++.NSE



Algorithm The Learn++.NSE algorithm

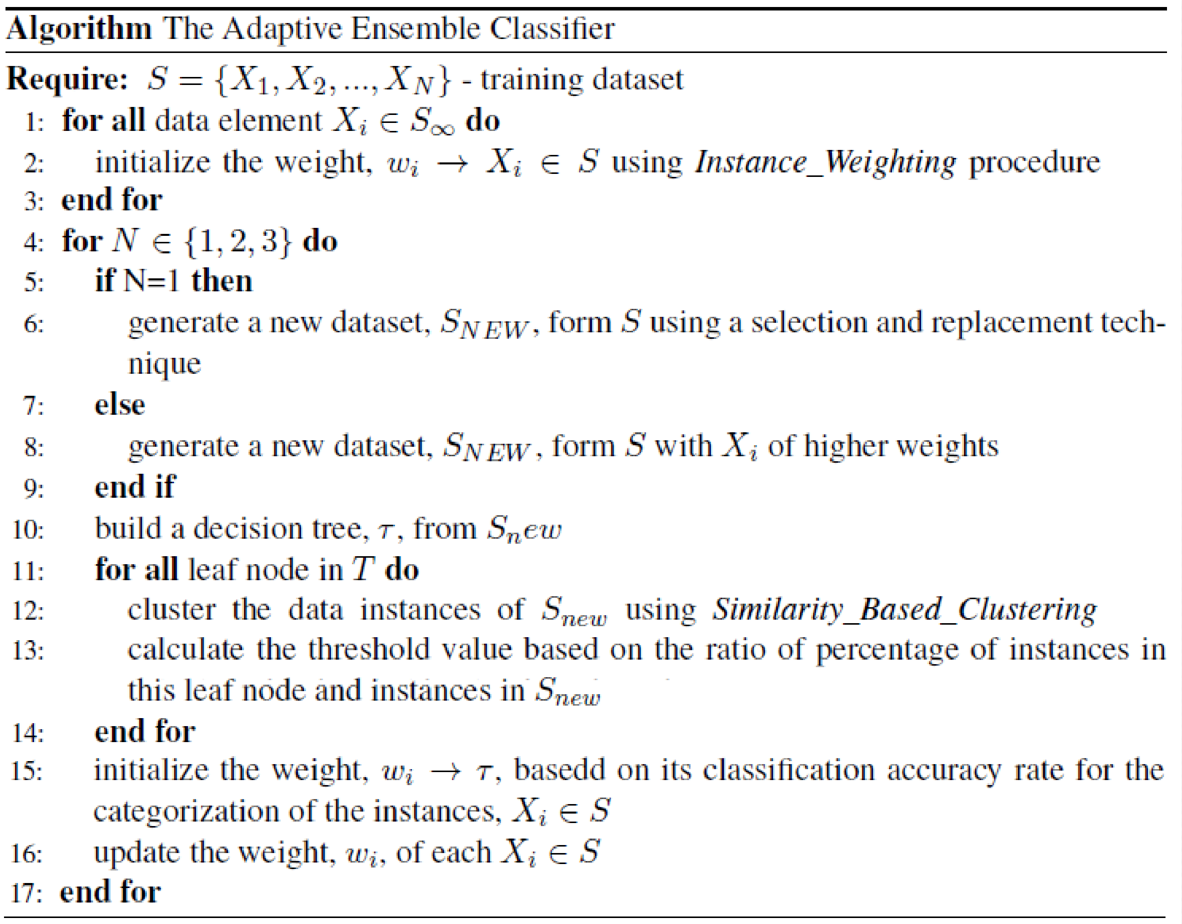
Require: $S_\infty = \{X_1, X_2, ...\}$ - dataset representing a new environment, S_t - sequence of data elements $X_i^t \in \{X_1^t, X_2^t, ..., X_{N^t}^t\}$ from S_∞, supervised learning algorithm **BaseClassifier**

```
1:  for all $S_t$, $t = 1, 2, ...$ do
2:      if t=1 then
3:          initialize $D^1(i) = w^t(i) = 1/m^1, \forall i$, skip to line nr 8
4:      end if
5:      compute error of existing ensemble on new data
        $E^t = \sum_{i=1}^{m^t}(1/m^t)[\Gamma^{t-1}(X_i) = C(X_i)]$
6:      normalize error $B^t = E^t/(1 - E^t)$
7:      update instance weights
        $w_i^t = \frac{1}{m^t} \times \begin{cases} B^t, & \Gamma^{t-1}(X_i) = C(X_i) \\ 1, & \text{otherwise} \end{cases}$
8:      set $D^t = w^t / \sum_{i=1}^{m^t} w_i^t$ so that $D^k$ is a distribution
9:      call BaseClassifier with $D_t$, obtain $\tau_t : X \to C$
10:     evaluate all existing classifiers on new dataset $D_t$
        $\epsilon_k^t = \sum_{i=1}^{m^t} D^t(i)[\tau_k(X_i) \neq C(X_i)]$, for $k = 1, ..., t$
11:     if $\epsilon_{k=t}^t > 1/2$ then
12:         generate a new $\tau_t$.
13:     end if
14:     if $\epsilon_{k<t}^t > 1/2$ then
15:         $\epsilon_k^t = 1/2$
16:         $\beta_k^t = \epsilon_k^t/(1 - \epsilon_k^t$ for $k = 1, ..., t$
17:     end if
18:     compute a weighted sum of all normalized errors for $k$-th classifier $\tau_k$
19:     $\omega_k^t = 1/(1 + \exp -a(t - k - b)), \omega_k^t = \omega_k^t / = \sum_{j=0}^{t-k} \omega_k^{t-j}$
20:     $\hat{\beta}_k^t = \sum_{j=0}^{t-k} \omega_k^{t-j} \beta_k^{t-j}$, for $k = 1, ..., t$
21:     calculate classifier voting weights $\mathcal{W}_k^t = \log(1/\hat{\beta}_k^t)$, for $k = 1, ..., t$
22:     obtain the composite hypothesis as the current final hypothesis
        $\Gamma^t(X_i) = \text{argmax}_{\{C\}} \sum_k W_k^t[k_k(X_i) = C]$
23: end for
```

* Ryan Elwell and Robi Polikar, Incremental learning of concept drift in nonstationary environments, IEEE Transactions on Neural Networks, vol. 22, no. 10, pp. 1517–1531, Oct 2011.

Stream Data

- Ensemble methods (the Adaptive Ensemble Classifier)

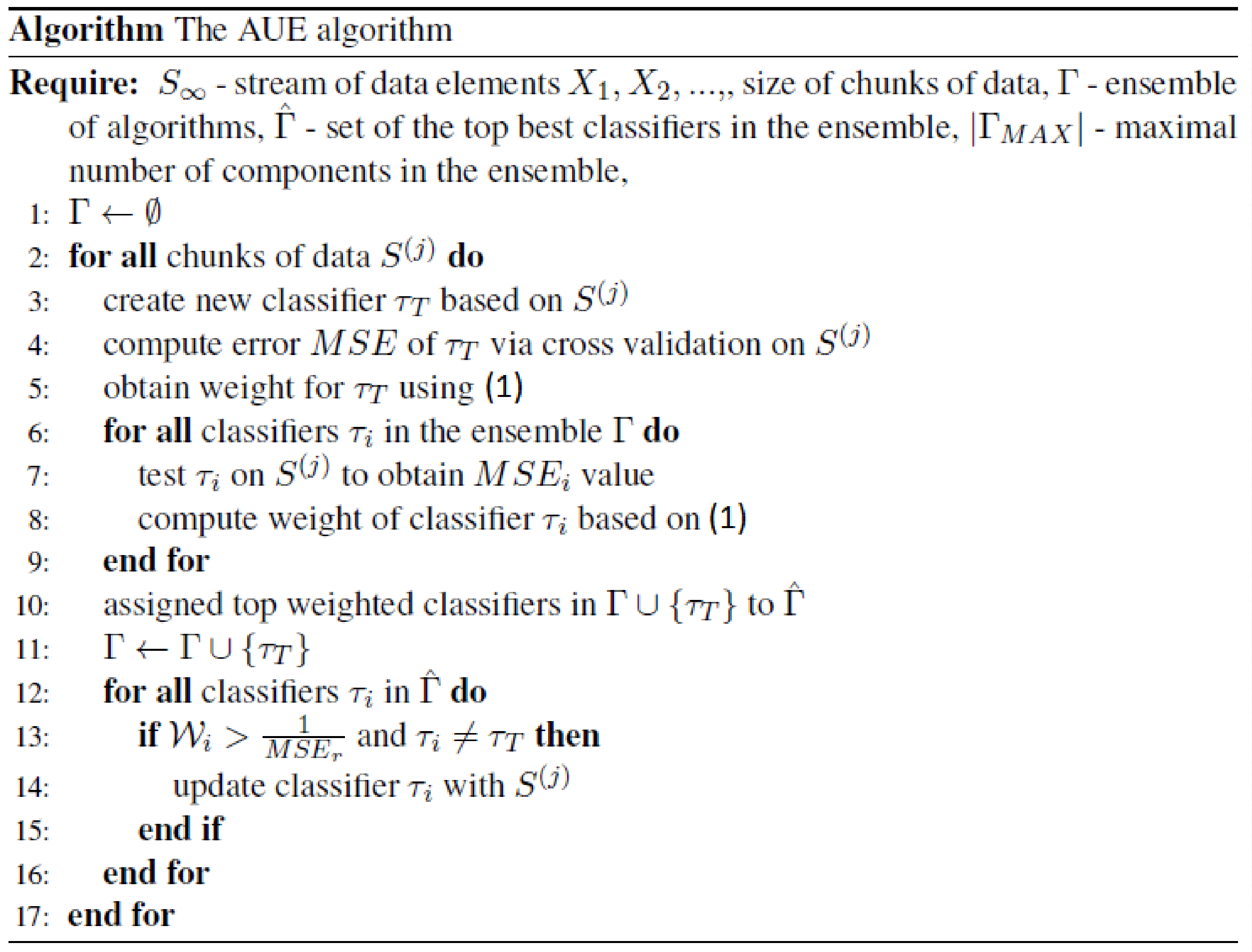


```
Algorithm The Adaptive Ensemble Classifier
Require: $S = \{X_1, X_2, ..., X_N\}$ - training dataset
for all data element $X_i \in S_\infty$ do
    initialize the weight, $w_i \rightarrow X_i \in S$ using Instance_Weighting procedure
end for
for $N \in \{1, 2, 3\}$ do
    if N=1 then
        generate a new dataset, $S_{NEW}$, form $S$ using a selection and replacement technique
    else
        generate a new dataset, $S_{NEW}$, form $S$ with $X_i$ of higher weights
    end if
    build a decision tree, $\tau$, from $S_new$
    for all leaf node in $T$ do
        cluster the data instances of $S_{new}$ using Similarity_Based_Clustering
        calculate the threshold value based on the ratio of percentage of instances in this leaf node and instances in $S_{new}$
    end for
    initialize the weight, $w_i \rightarrow \tau$, basedd on its classification accuracy rate for the categorization of the instances, $X_i \in S$
    update the weight, $w_i$, of each $X_i \in S$
end for
```

* Dewan Md. Farid, Li Zhang, Alamgir Hossain, Chowdhury Mofizur Rahman, Rebecca Strachan, Graham Sexton, and Keshav Dahal, An adaptive ensemble classifier for mining concept drifting data streams, Expert Systems with Applications, vol. 40, no. 15, pp. 5895 – 5906, 2013.

Stream Data

- Ensemble methods (the AUE)



Algorithm The AUE algorithm

Require: S_∞ - stream of data elements $X_1, X_2, \ldots,$ size of chunks of data, Γ - ensemble of algorithms, $\hat{\Gamma}$ - set of the top best classifiers in the ensemble, $|\Gamma_{MAX}|$ - maximal number of components in the ensemble,

```
$\Gamma \leftarrow \emptyset$
for all chunks of data $S^{(j)}$ do
    create new classifier $\tau_T$ based on $S^{(j)}$
    compute error $MSE$ of $\tau_T$ via cross validation on $S^{(j)}$
    obtain weight for $\tau_T$ using (1)
    for all classifiers $\tau_i$ in the ensemble $\Gamma$ do
        test $\tau_i$ on $S^{(j)}$ to obtain $MSE_i$ value
        compute weight of classifier $\tau_i$ based on (1)
    end for
    assigned top weighted classifiers in $\Gamma \cup \{\tau_T\}$ to $\hat{\Gamma}$
    $\Gamma \leftarrow \Gamma \cup \{\tau_T\}$
    for all classifiers $\tau_i$ in $\hat{\Gamma}$ do
        if $\mathcal{W}_i > \frac{1}{MSE_r}$ and $\tau_i \neq \tau_T$ then
            update classifier $\tau_i$ with $S^{(j)}$
        end if
    end for
end for
```

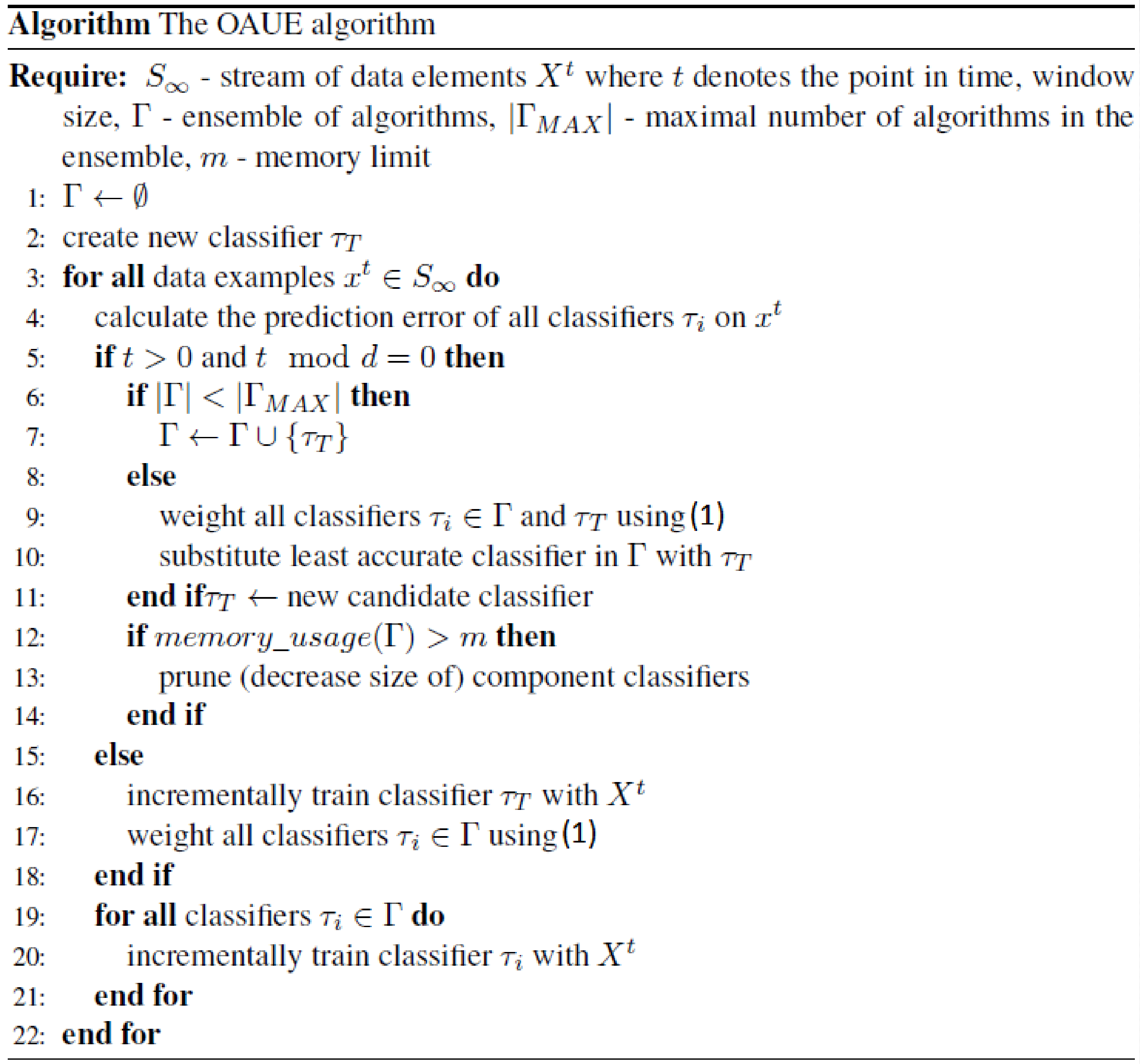
$$\mathcal{W}_j = \frac{1}{MSE_j + \epsilon}$$

where ϵ is a very small constant value which is used to avoid the problem of division by zero.

* Dariusz Brzezinski and Jerzy Stefanowski, Reacting to different types of concept drift: The accuracy updated ensemble algorithm, IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 81–94, Jan 2014.

Stream Data
- Ensemble methods

the OAUE



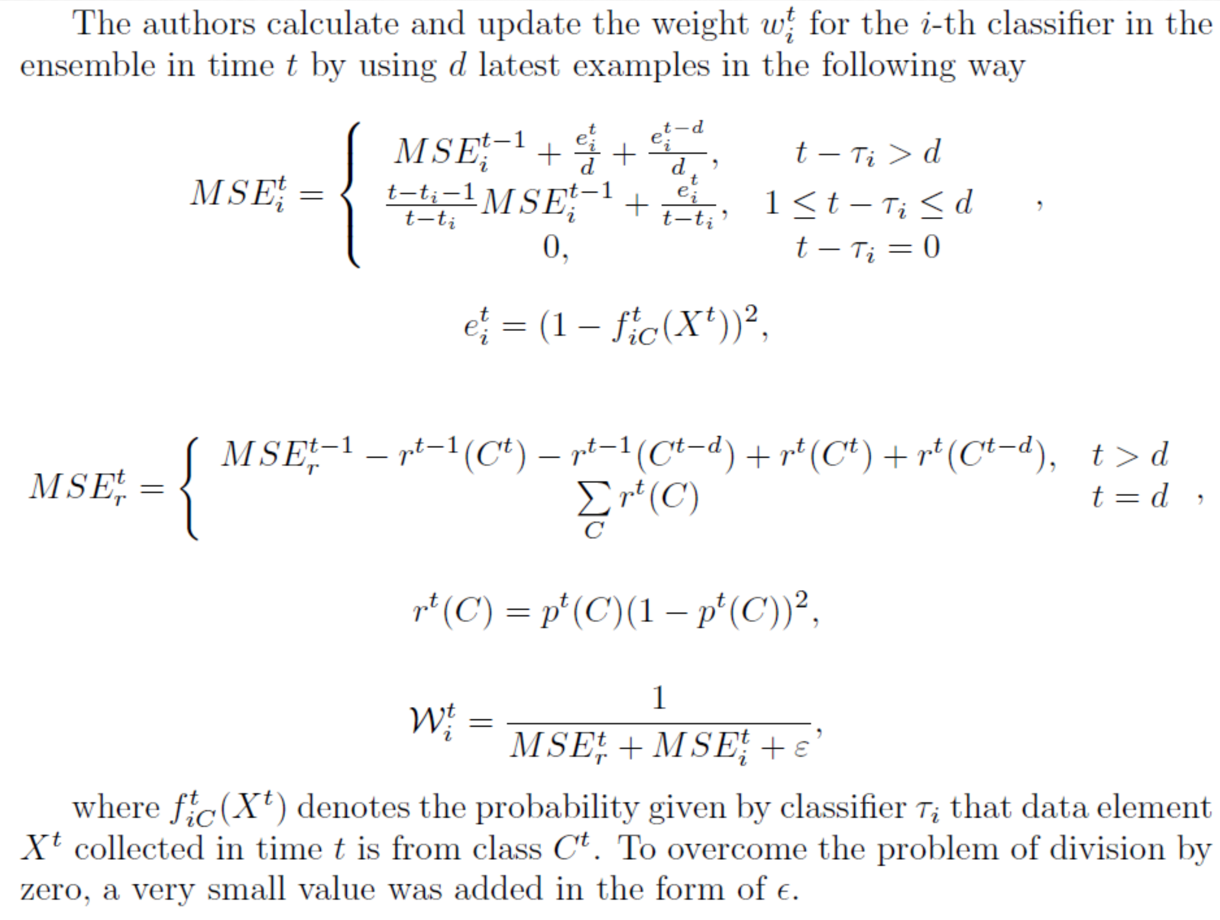
Algorithm The OAUE algorithm

```
Require: $S_\infty$ - stream of data elements $X^t$ where $t$ denotes the point in time, window
    size, $\Gamma$ - ensemble of algorithms, $|\Gamma_{MAX}|$ - maximal number of algorithms in the
    ensemble, $m$ - memory limit
$\Gamma \leftarrow \emptyset$
create new classifier $\tau_T$
for all data examples $x^t \in S_\infty$ do
    calculate the prediction error of all classifiers $\tau_i$ on $x^t$
    if $t > 0$ and $t \mod d = 0$ then
        if $|\Gamma| < |\Gamma_{MAX}|$ then
            $\Gamma \leftarrow \Gamma \cup \{\tau_T\}$
        else
            weight all classifiers $\tau_i \in \Gamma$ and $\tau_T$ using (1)
            substitute least accurate classifier in $\Gamma$ with $\tau_T$
        end if$\tau_T \leftarrow$ new candidate classifier
        if $memory\_usage(\Gamma) > m$ then
            prune (decrease size of) component classifiers
        end if
    else
        incrementally train classifier $\tau_T$ with $X^t$
        weight all classifiers $\tau_i \in \Gamma$ using (1)
    end if
    for all classifiers $\tau_i \in \Gamma$ do
        incrementally train classifier $\tau_i$ with $X^t$
    end for
end for
```

Brzezinski D., Stefanowski J., „Combining block-based and online methods in learning ensembles from concept drifting data streams", Information Sciences, Vol. 265, pp. 50-67, 2014.

Stream Data

- Ensemble methods (the OAUE)



The authors calculate and update the weight w_i^t for the i-th classifier in the ensemble in time t by using d latest examples in the following way

$$MSE_i^t = \begin{cases} MSE_i^{t-1} + \frac{e_i^t}{d} + \frac{e_i^{t-d}}{d}, & t - \tau_i > d \\ \frac{t-t_i-1}{t-t_i} MSE_i^{t-1} + \frac{e_i^t}{t-t_i}, & 1 \leq t - \tau_i \leq d \\ 0, & t - \tau_i = 0 \end{cases},$$

$$e_i^t = (1 - f_{iC}^t(X^t))^2,$$

$$MSE_r^t = \begin{cases} MSE_r^{t-1} - r^{t-1}(C^t) - r^{t-1}(C^{t-d}) + r^t(C^t) + r^t(C^{t-d}), & t > d \\ \sum_C r^t(C) & t = d \end{cases},$$

$$r^t(C) = p^t(C)(1 - p^t(C))^2,$$

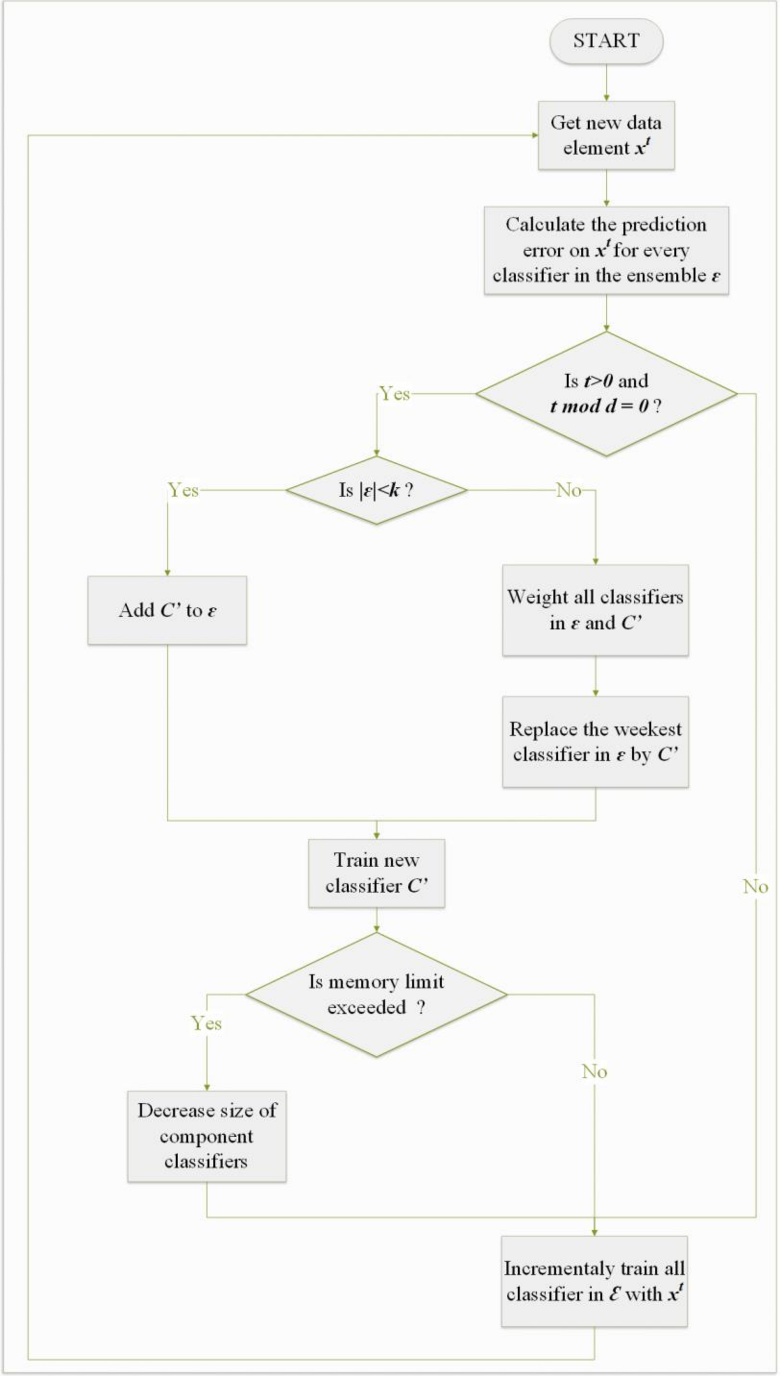
$$\mathcal{W}_i^t = \frac{1}{MSE_r^t + MSE_i^t + \varepsilon},$$

where $f_{iC}^t(X^t)$ denotes the probability given by classifier τ_i that data element X^t collected in time t is from class C^t. To overcome the problem of division by zero, a very small value was added in the form of ϵ.

Brzezinski D., Stefanowski J., „Combining block-based and online methods in learning ensembles from concept drifting data streams", Information Sciences, Vol. 265, pp. 50-67, 2014.

Stream Data
- Ensemble methods

the OAUE



Brzezinski D., Stefanowski J., „Combining block-based and online methods in learning ensembles from concept drifting data streams”, Information Sciences, Vol. 265, pp. 50-67, 2014

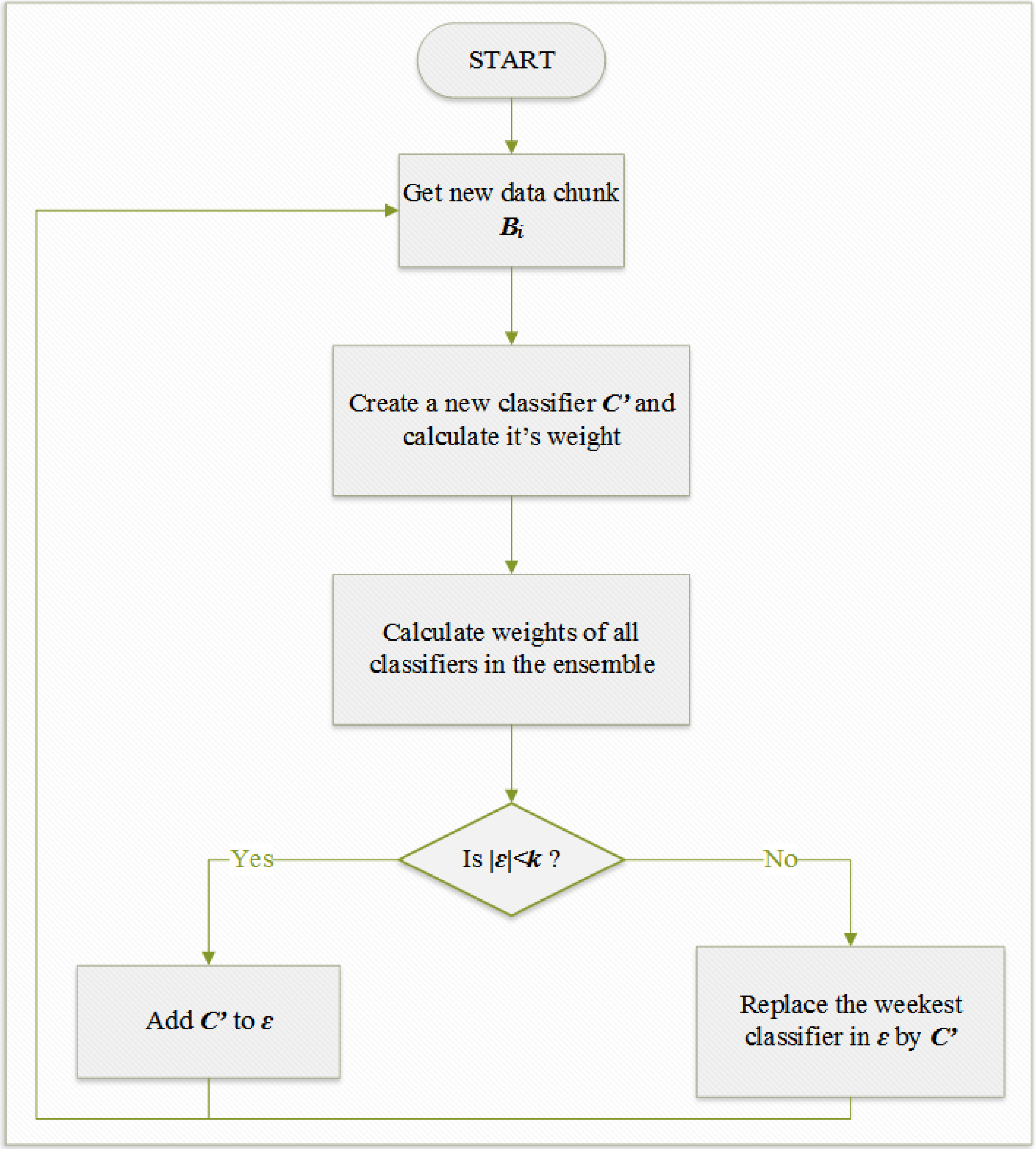
Stream Data

- Ensemble methods

General strategies for adaptation of algorithms for mining data streams based on ensemble methods.

Brzezinski D., Stefanowski J., „Combining block-based and online methods in learning ensembles from concept drifting data streams", Information Sciences, Vol. 265, pp. 50-67, 2014.

Generic block ensemble training scheme



START

Get new data chunk B_i

Create a new classifier ***C’*** and calculate it’s weight

Calculate weights of all classifiers in the ensemble

Is $|\varepsilon|<k$?

Yes

No

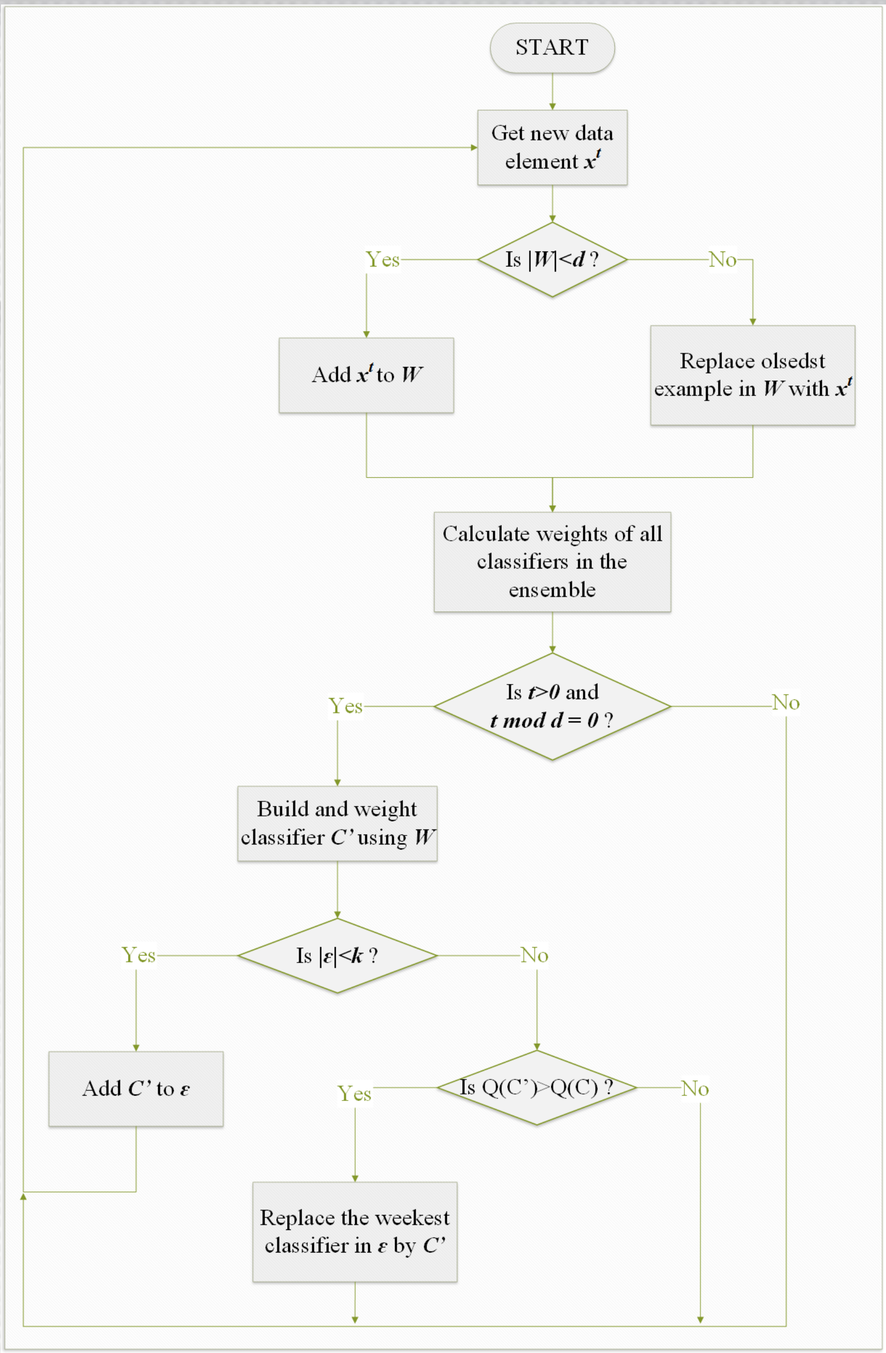
Add ***C’*** to ε

Replace the weekest classifier in ε by ***C’***

Stream Data

- General Ensemble Strategies

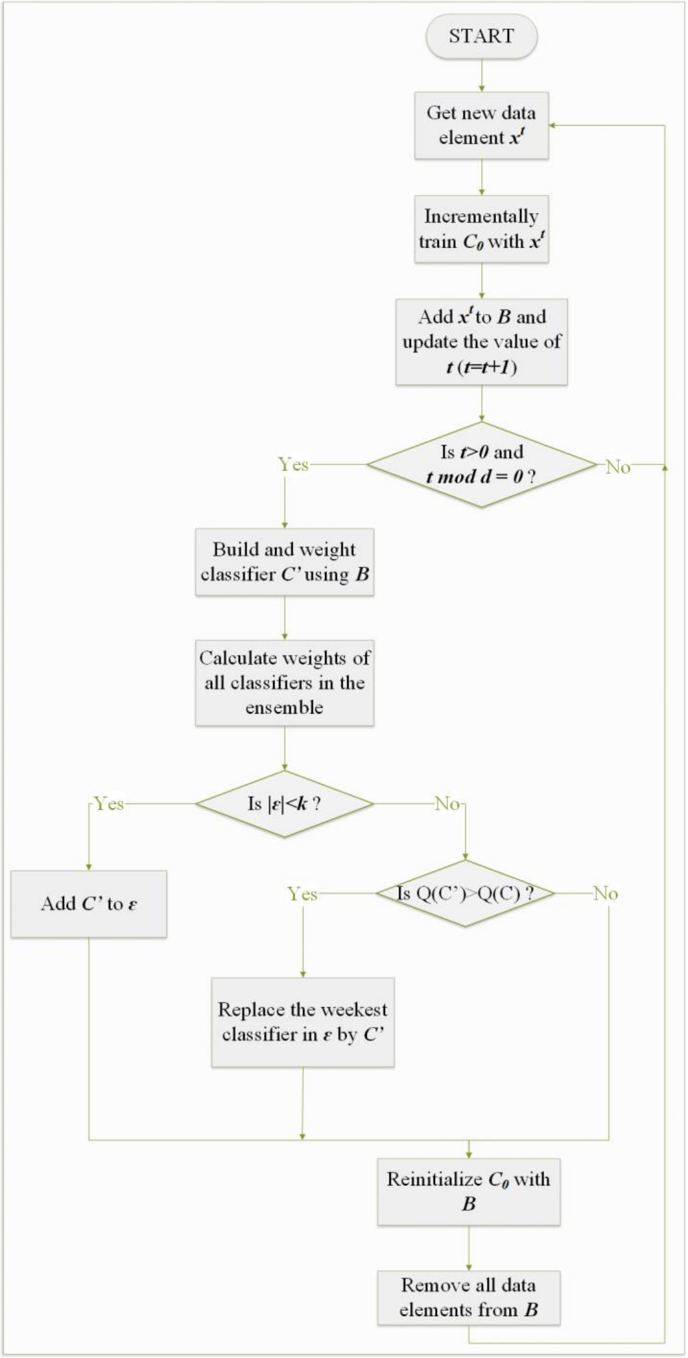
Windowing strategy



Stream Data

- General Ensemble Strategies

Additional incremental learner strategy



START

Get new data element x^t

Incrementally train C_0 with x^t

Add x^t to B and update the value of t (t=t+1)

Is t>0 and $t \ mod \ d = 0$?

Yes

No

Build and weight classifier C' using B

Calculate weights of all classifiers in the ensemble

Is $|\varepsilon|<k$?

Yes

No

Add C' to ε

Is Q(C')>Q(C) ?

Yes

No

Replace the weekest classifier in ε by C'

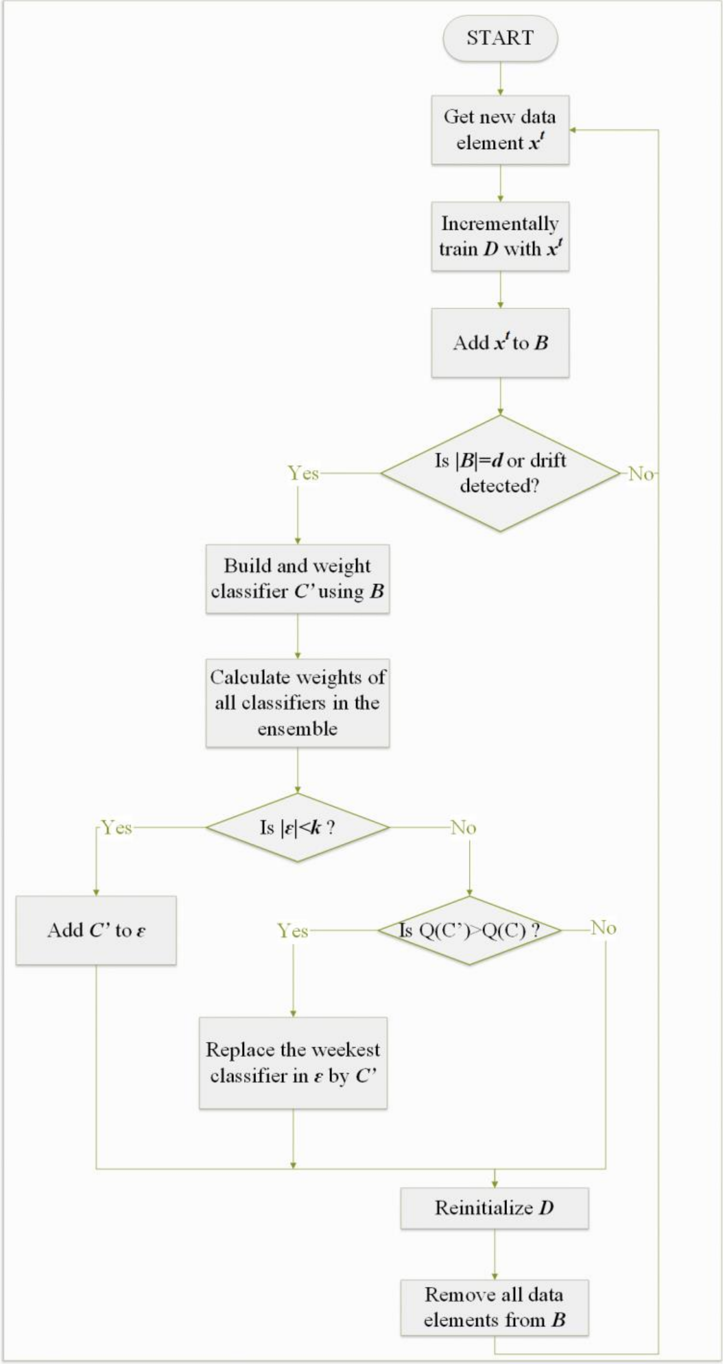
Reinitialize C_0 with B

Remove all data elements from B

Stream Data

- General Ensemble Strategies

Drift detector strategy



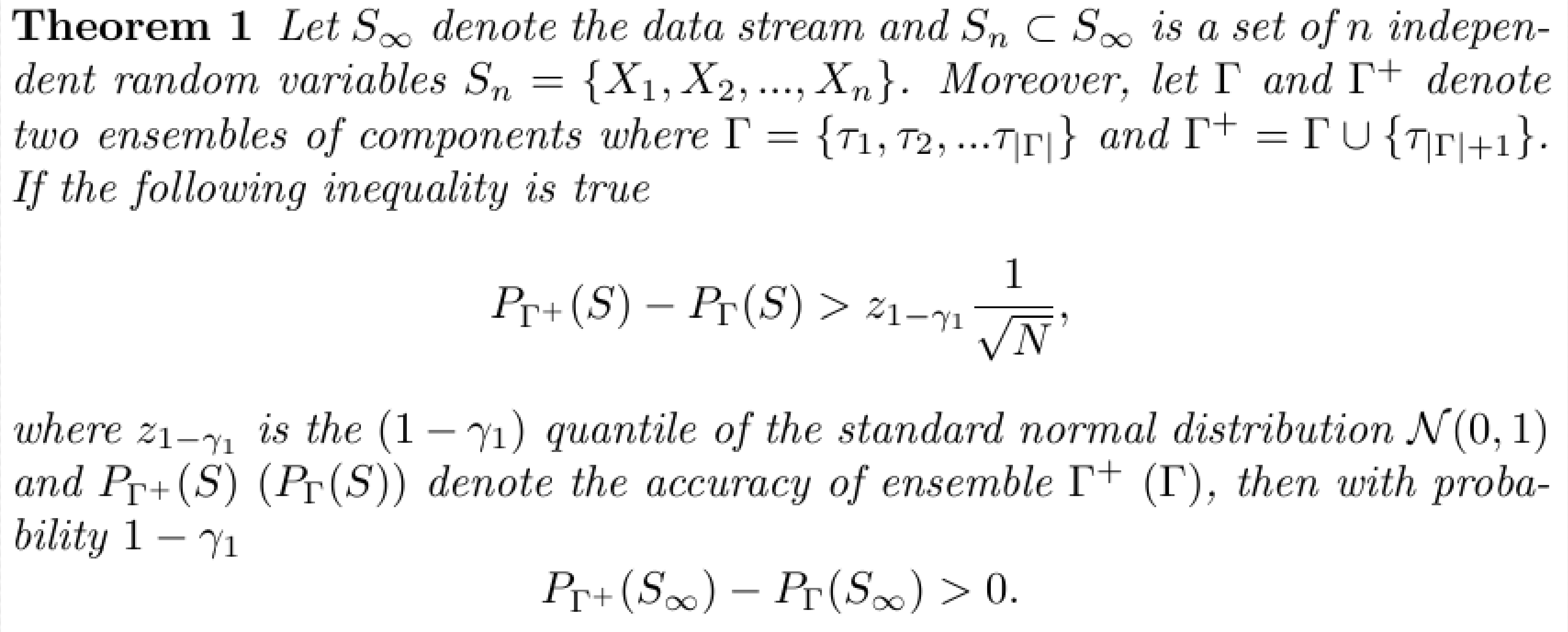
Stream Data
- Ensemble methods

How to decide if a new component should be added to the ensemble?

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)



Theorem 1 *Let* S_∞ *denote the data stream and* $S_n \subset S_\infty$ *is a set of* n *independent random variables* $S_n = \{X_1, X_2, ..., X_n\}$. *Moreover, let* Γ *and* Γ^+ *denote two ensembles of components where* $\Gamma = \{\tau_1, \tau_2, ...\tau_{|\Gamma|}\}$ *and* $\Gamma^+ = \Gamma \cup \{\tau_{|\Gamma|+1}\}$. *If the following inequality is true*

$$P_{\Gamma^+}(S) - P_\Gamma(S) > z_{1-\gamma_1}\frac{1}{\sqrt{N}},$$

where $z_{1-\gamma_1}$ *is the* $(1-\gamma_1)$ *quantile of the standard normal distribution* $\mathcal{N}(0,1)$ *and* $P_{\Gamma^+}(S)$ $(P_\Gamma(S))$ *denote the accuracy of ensemble* Γ^+ (Γ), *then with probability* $1-\gamma_1$

$$P_{\Gamma^+}(S_\infty) - P_\Gamma(S_\infty) > 0.$$

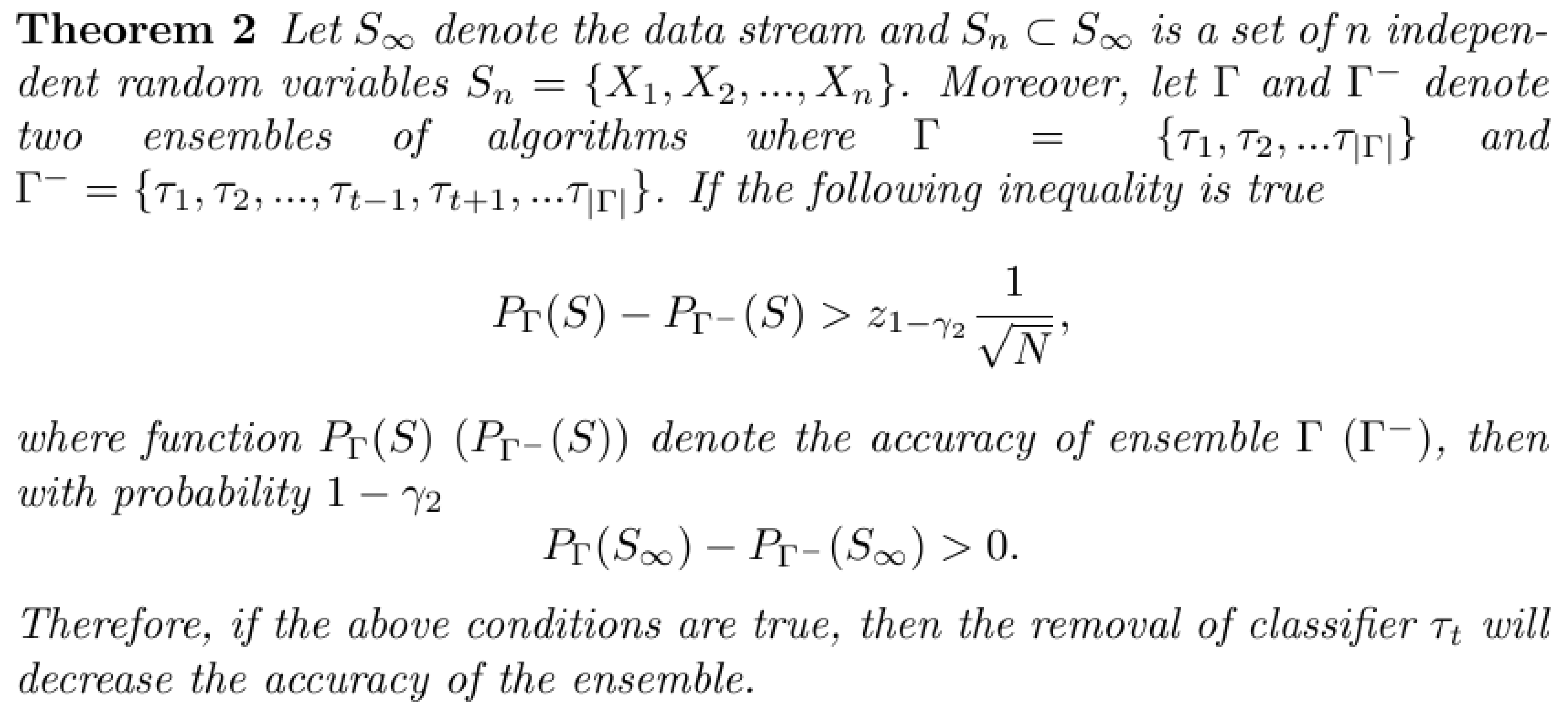
Stream Data
- Ensemble methods

How to decide if we should to remove a component from the ensemble?

New result
2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)



Theorem 2 *Let* S_∞ *denote the data stream and* $S_n \subset S_\infty$ *is a set of* n *independent random variables* $S_n = \{X_1, X_2, ..., X_n\}$. *Moreover, let* Γ *and* Γ^- *denote two ensembles of algorithms where* $\Gamma = \{\tau_1, \tau_2, ...\tau_{|\Gamma|}\}$ *and* $\Gamma^- = \{\tau_1, \tau_2, ..., \tau_{t-1}, \tau_{t+1}, ...\tau_{|\Gamma|}\}$. *If the following inequality is true*

$$P_\Gamma(S) - P_{\Gamma^-}(S) > z_{1-\gamma_2} \frac{1}{\sqrt{N}},$$

where function $P_\Gamma(S)$ $(P_{\Gamma^-}(S))$ *denote the accuracy of ensemble* Γ (Γ^-), *then with probability* $1 - \gamma_2$

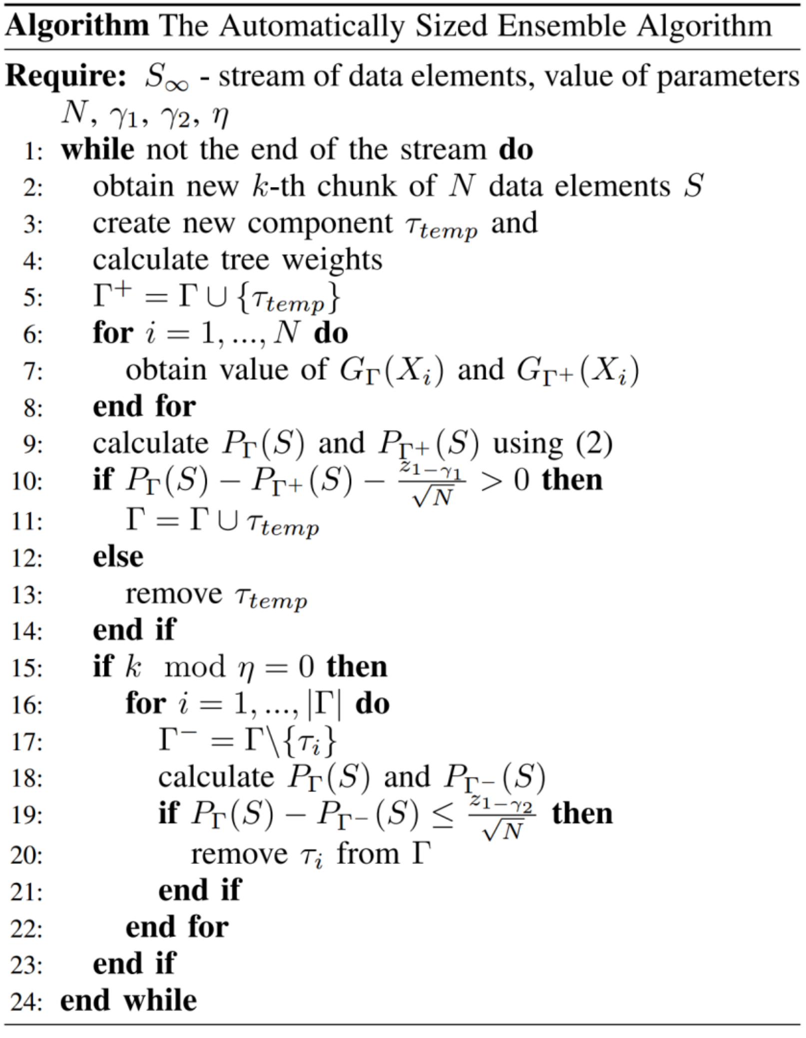
$$P_\Gamma(S_\infty) - P_{\Gamma^-}(S_\infty) > 0.$$

Therefore, if the above conditions are true, then the removal of classifier τ_t *will decrease the accuracy of the ensemble.*

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)



Algorithm The Automatically Sized Ensemble Algorithm

Require: S_∞ - stream of data elements, value of parameters N, γ_1, γ_2, η

```
while not the end of the stream do
    obtain new $k$-th chunk of $N$ data elements $S$
    create new component $\tau_{temp}$ and
    calculate tree weights
    $\Gamma^+ = \Gamma \cup \{\tau_{temp}\}$
    for $i = 1, ..., N$ do
        obtain value of $G_\Gamma(X_i)$ and $G_{\Gamma^+}(X_i)$
    end for
    calculate $P_\Gamma(S)$ and $P_{\Gamma^+}(S)$ using (2)
    if $P_\Gamma(S) - P_{\Gamma^+}(S) - \frac{z_{1-\gamma_1}}{\sqrt{N}} > 0$ then
        $\Gamma = \Gamma \cup \tau_{temp}$
    else
        remove $\tau_{temp}$
    end if
    if $k \mod \eta = 0$ then
        for $i = 1, ..., |\Gamma|$ do
            $\Gamma^- = \Gamma \backslash \{\tau_i\}$
            calculate $P_\Gamma(S)$ and $P_{\Gamma^-}(S)$
            if $P_\Gamma(S) - P_{\Gamma^-}(S) \leq \frac{z_{1-\gamma_2}}{\sqrt{N}}$ then
                remove $\tau_i$ from $\Gamma$
            end if
        end for
    end if
end while
```

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)

	Bayes	AUE	AWE	Bagging	Bagging Adwin	Boosting	WMA	Hoefding Adapive Tree	Hoeffding Opt.Tree	Hoeffding Tree	AASE η=20	AASE η=40
GEN1	80.86%	**95.55%**	96.86%	96.19%	96.19%	95.90%	95.23%	94.13%	95.62%	92.17%	90.68%	90.23%
GEN2	50.00%	87.18%	86.33%	89.69%	88.71%	87.80%	89.35%	86.91%	89.16%	89.35%	**92.42%**	92.30%
GEN3	86.85%	70.94%	69.27%	79.72%	70.02%	70.26%	80.19%	71.93%	80.08%	80.08%	**91.31%**	91.06%
GEN4	74.88%	73.61%	73.87%	74.45%	73.63%	73.91%	73.73%	73.12%	73.81%	73.73%	**90.46%**	90.30%

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)

	GEN1		GEN2		GEN3		GEN4	
$\gamma 1$	max tree number	end tree number	max tree number	end tree number	max tree number	end tree number	max tree number	end tree number
0.01	7	5	5	2	6	3	9	2
0.05	9	6	7	2	7	5	14	5
0.10	14	9	10	4	11	7	21	7
0.15	17	11	14	4	13	9	19	9
0.20	22	13	18	6	18	10	29	12
Accuracy								
0.01	91.89%		92.45%		91.91%		87.84%	
0.05	92.37%		93.19%		92.62%		88.97%	
0.10	93.08%		93.96%		93.16%		89.93%	
0.15	93.24%		94.19%		93.41%		90.27%	
0.20	93.43%		94.38%		93.67%		90.47%	

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)

	GEN1		GEN2		GEN3		GEN4	
$\gamma 2$	max tree number	end tree number	max tree number	end tree number	max tree number	end tree number	max tree number	end tree number
0.01	21	13	18	6	18	10	26	11
0.05	22	13	18	6	18	10	29	12
0.10	24	15	17	6	17	11	31	11
0.15	23	15	17	5	17	10	27	11
0.20	23	15	17	7	17	12	26	11
Accuracy								
0.01	93.43%		94.33%		93.68%		90.48%	
0.05	93.43%		94.38%		93.67%		90.47%	
0.10	93.30%		94.33%		93.65%		90.46%	
0.15	93.28%		94.26%		93.53%		90.56%	
0.20	93.27%		94.29%		93.64%		90.47%	

New result 2016

Stream Data

- The Automatically Adjusting Size of Ensemble Algorithm (ASE)

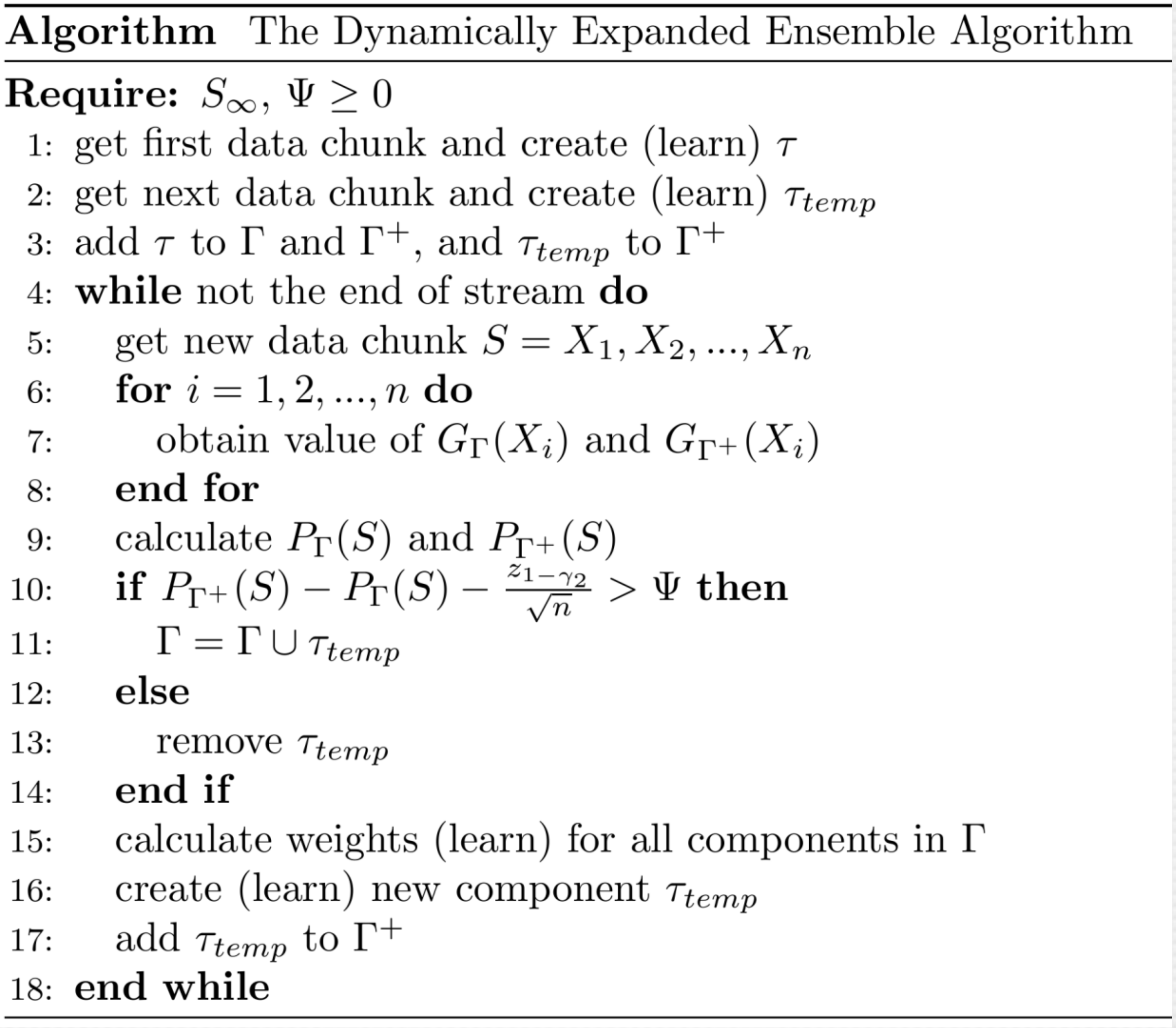
	GEN1	GEN2	GEN3	GEN4
0.01	539	458	484	613
0.05	539	432	472	620
0.10	517	410	475	615
0.15	521	399	454	605
0.20	509	397	438	569

difference	**30**	**61**	**46**	**44**

New result
2016

Stream Data

- The Dynamically Expanded Ensemble Algorithm (DEEA)



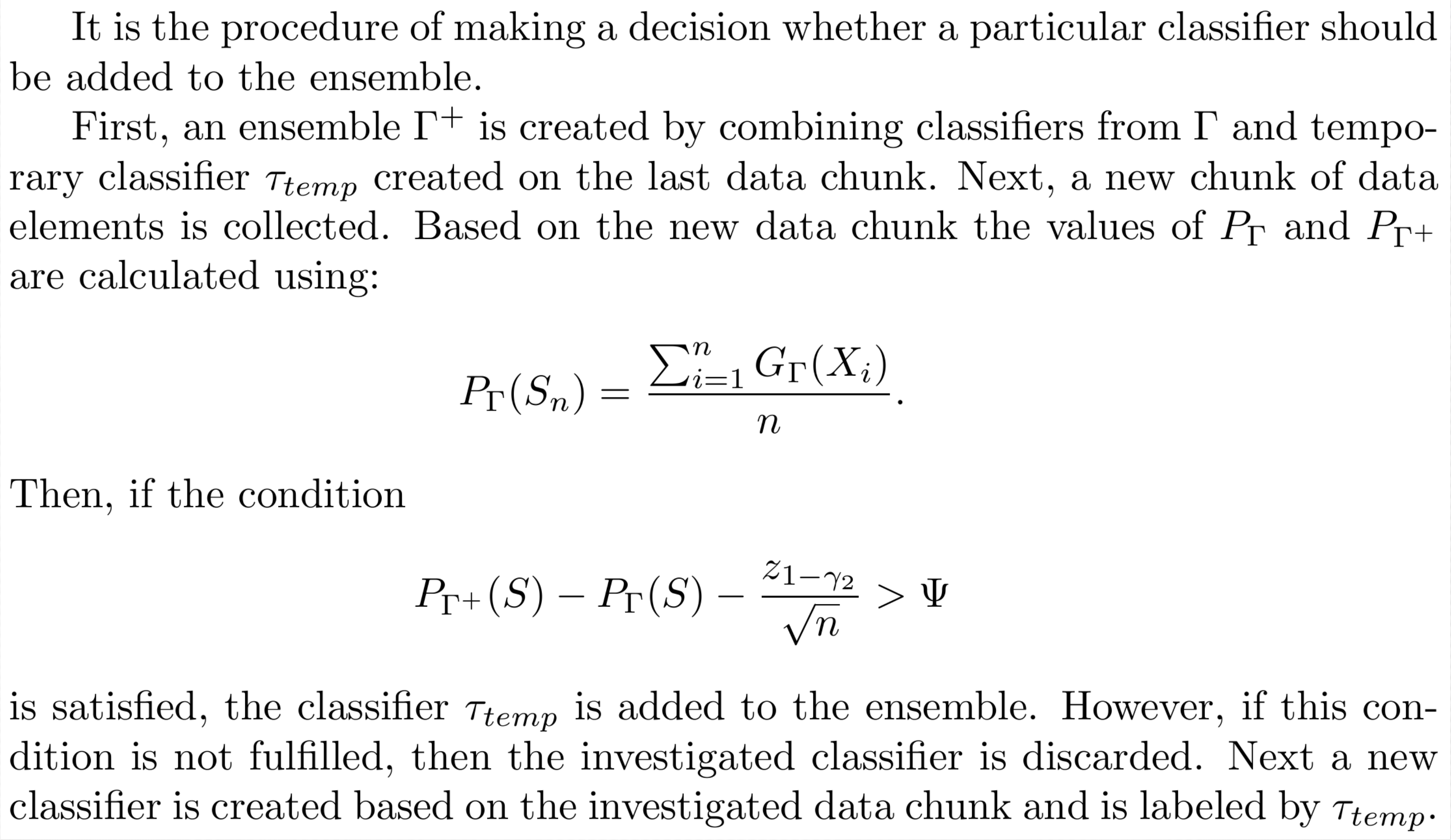
Algorithm The Dynamically Expanded Ensemble Algorithm

```
Require: $S_\infty$, $\Psi \geq 0$
get first data chunk and create (learn) $\tau$
get next data chunk and create (learn) $\tau_{temp}$
add $\tau$ to $\Gamma$ and $\Gamma^+$, and $\tau_{temp}$ to $\Gamma^+$
while not the end of stream do
    get new data chunk $S = X_1, X_2, ..., X_n$
    for $i = 1, 2, ..., n$ do
        obtain value of $G_\Gamma(X_i)$ and $G_{\Gamma^+}(X_i)$
    end for
    calculate $P_\Gamma(S)$ and $P_{\Gamma^+}(S)$
    if $P_{\Gamma^+}(S) - P_\Gamma(S) - \frac{z_{1-\gamma_2}}{\sqrt{n}} > \Psi$ then
        $\Gamma = \Gamma \cup \tau_{temp}$
    else
        remove $\tau_{temp}$
    end if
    calculate weights (learn) for all components in $\Gamma$
    create (learn) new component $\tau_{temp}$
    add $\tau_{temp}$ to $\Gamma^+$
end while
```

New result 2016

Stream Data

- The Dynamically Expanded Ensemble Algorithm (DEEA)



It is the procedure of making a decision whether a particular classifier should be added to the ensemble.

First, an ensemble Γ^{+} is created by combining classifiers from Γ and temporary classifier τ_{temp} created on the last data chunk. Next, a new chunk of data elements is collected. Based on the new data chunk the values of P_{Γ} and $P_{\Gamma^{+}}$ are calculated using:

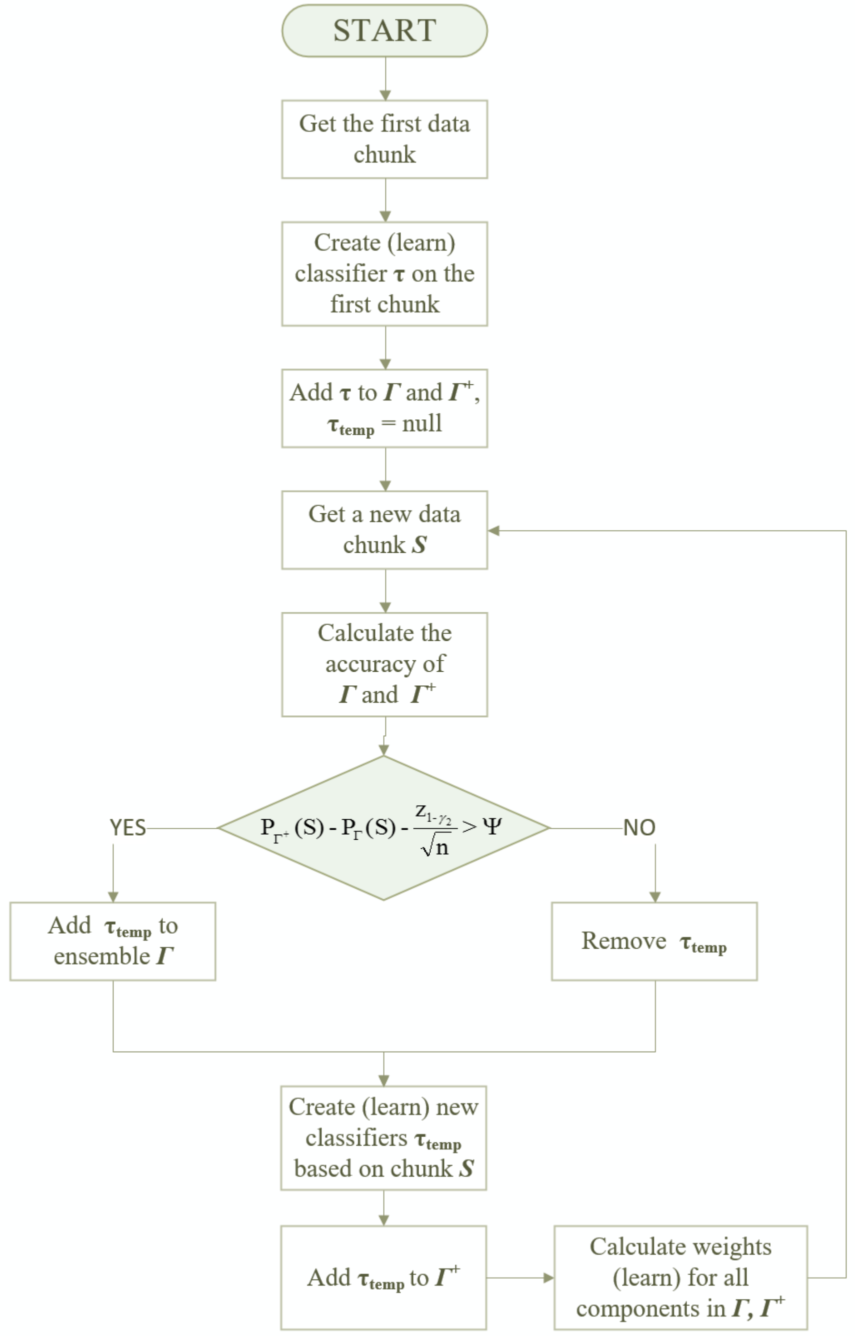
$$P_{\Gamma}(S_n) = \frac{\sum_{i=1}^{n} G_{\Gamma}(X_i)}{n}.$$

Then, if the condition

$$P_{\Gamma^{+}}(S) - P_{\Gamma}(S) - \frac{z_{1-\gamma_2}}{\sqrt{n}} > \Psi$$

is satisfied, the classifier τ_{temp} is added to the ensemble. However, if this condition is not fulfilled, then the investigated classifier is discarded. Next a new classifier is created based on the investigated data chunk and is labeled by τ_{temp}.

Stream Data - The Dynamically Expanded Ensemble Algorithm (DEEA)



START

Get the first data chunk

Create (learn) classifier $\boldsymbol{\tau}$ on the first chunk

Add $\boldsymbol{\tau}$ to $\boldsymbol{\Gamma}$ and $\boldsymbol{\Gamma}^{+}$, $\boldsymbol{\tau}_{\mathbf{temp}}$ = null

Get a new data chunk $\boldsymbol{S}$

Calculate the accuracy of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Gamma}^{+}$

$$P_{\Gamma^{+}}(S) - P_{\Gamma}(S) - \frac{z_{1-\gamma_2}}{\sqrt{n}} > \Psi$$

YES

Add $\boldsymbol{\tau}_{\mathbf{temp}}$ to ensemble $\boldsymbol{\Gamma}$

NO

Remove $\boldsymbol{\tau}_{\mathbf{temp}}$

Create (learn) new classifiers $\boldsymbol{\tau}_{\mathbf{temp}}$ based on chunk $\boldsymbol{S}$

Add $\boldsymbol{\tau}_{\mathbf{temp}}$ to $\boldsymbol{\Gamma}^{+}$

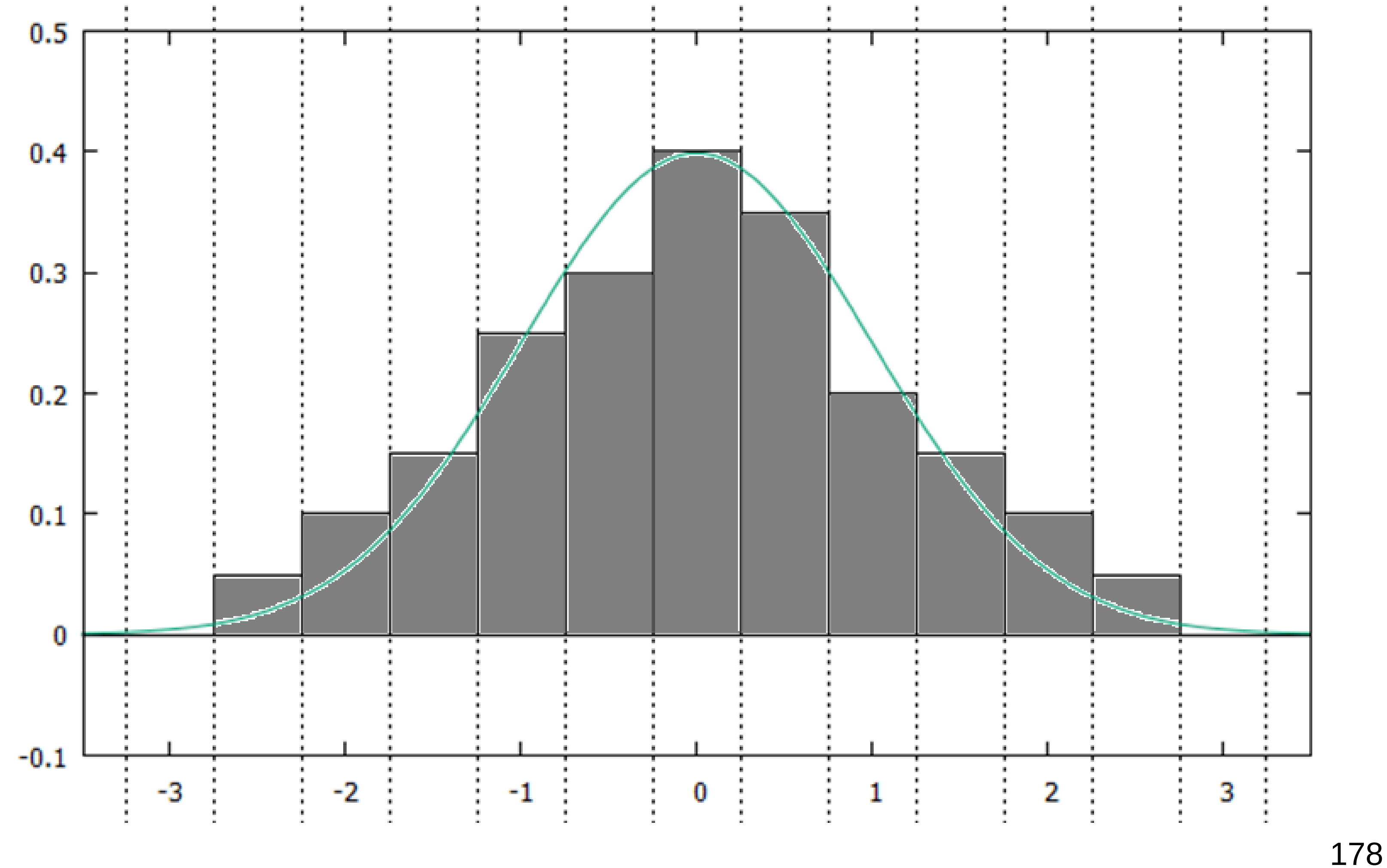
Calculate weights (learn) for all components in $\boldsymbol{\Gamma}$, $\boldsymbol{\Gamma}^{+}$

Data stream mining
- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- **Probabilistic neural networks for stream data mining (including new results 2016)**
- Final remarks and challenging problems
- References

Probabilistic Neural Networks (PNN) for stream data mining

PNN for stream data mining
Histograms



PNN for stream data mining

Non-recursive procedures

Let $X_1, \ldots, X_n$ be a sequence of independent, identically distributed random variables taking values in $A \subset \mathbb{R}^p$ and having a probability density function f. The general estimator of the probability density function f is given by the following formula

$$\hat{f}_n(x) = \frac{1}{n} \sum_{i=1}^{n} K_n(x, X_i)$$

where

$$K_n(x, X_i) = \frac{1}{h_n} K(x, X_i)$$

PNN for stream data mining

Non-recursive procedures

Parzen (1962):

$$\hat{f}_n(x) = \frac{1}{nh_n} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h_n}\right)$$

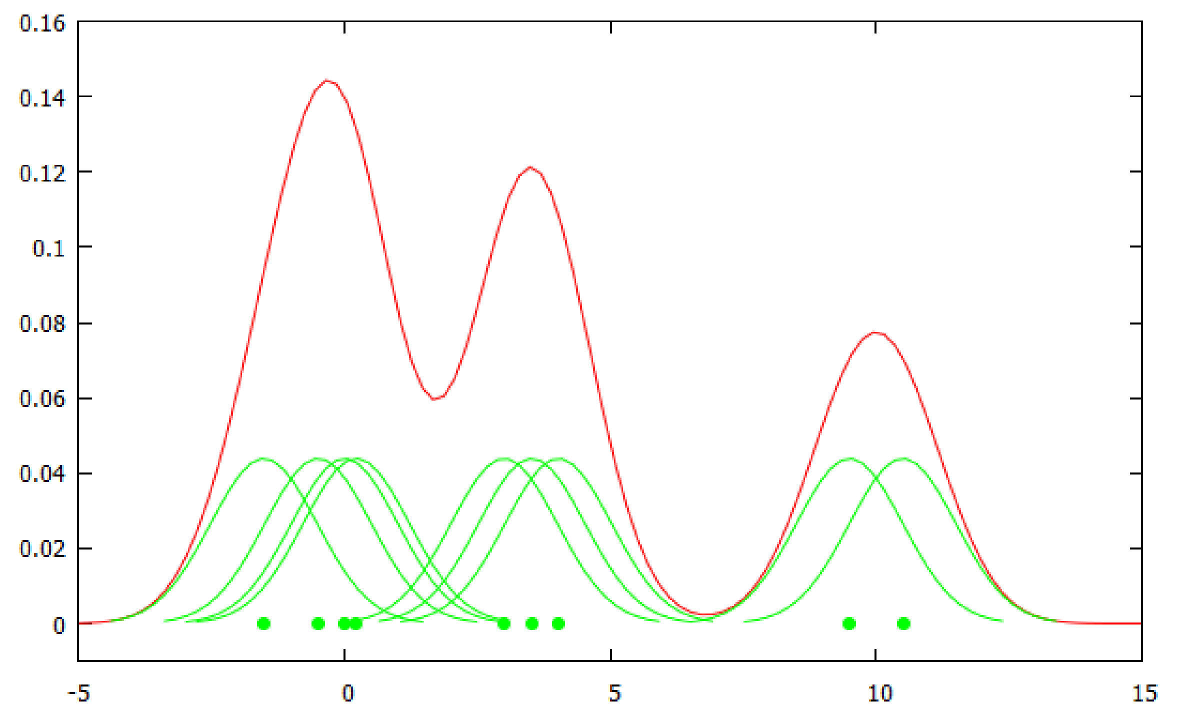
Cacullous (1965):

$$\hat{f}_n(x) = \frac{1}{nh_n^p} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h_n}\right)$$

$$h(n) > 0, \quad h_n \xrightarrow{n} 0, \quad nh_n^p \xrightarrow{n} \infty$$

PNN for stream data mining

Non-recursive procedures

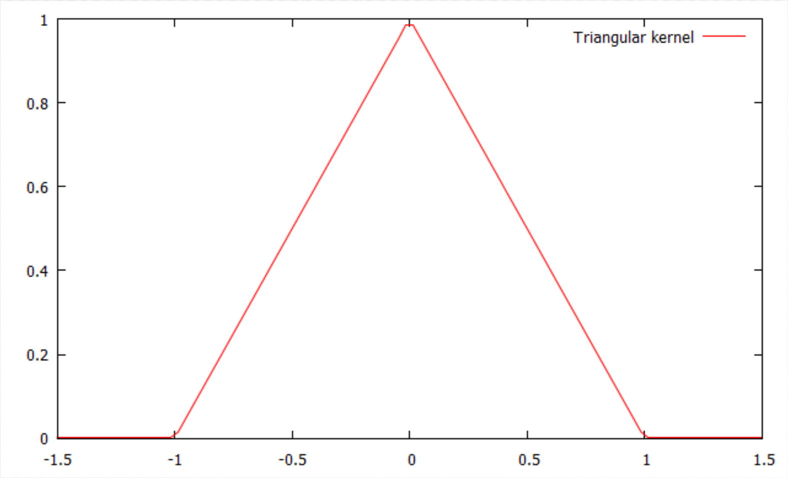


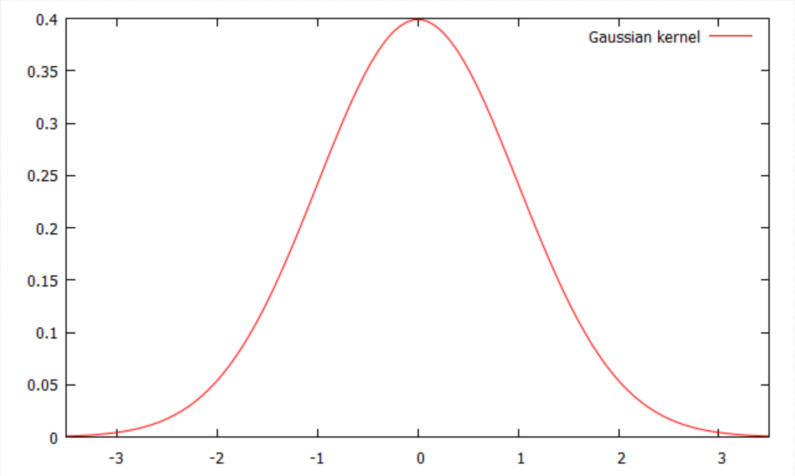
Parzen (1962):

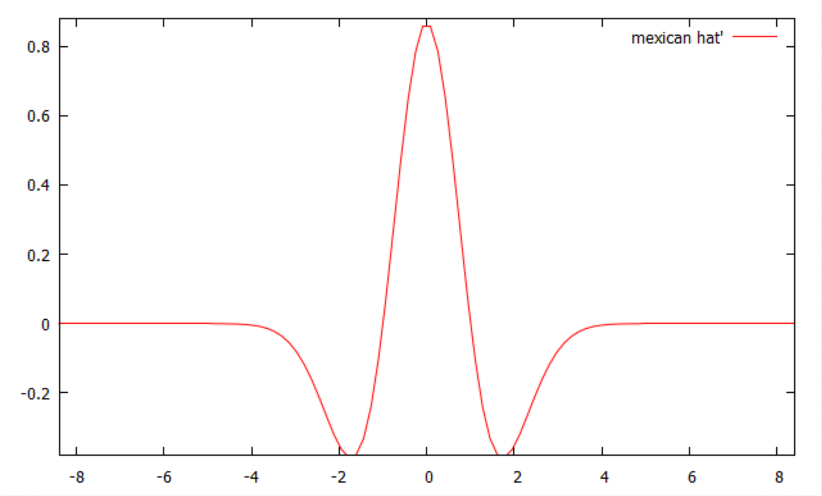
$$\hat{f}_n(x) = \frac{1}{nh_n} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h_n} \right)$$

PNN for stream data mining

Example of kernels







PNN for stream data mining

Non-recursive procedures

Example (Gaussian kernel)

$$K_n(x,u) = h_n^{-p} K\left(\frac{x-u}{h_n}\right)$$

$$K(x) = \prod_{j=1}^{p} H\left(x^{(j)}\right)$$

$$K_n(x,u) = h_n^{-p} \prod_{j=1}^{p} H\left(\frac{x^{(j)} - u^{(j)}}{h_n}\right)$$

PNN for stream data mining

Non-recursive procedures

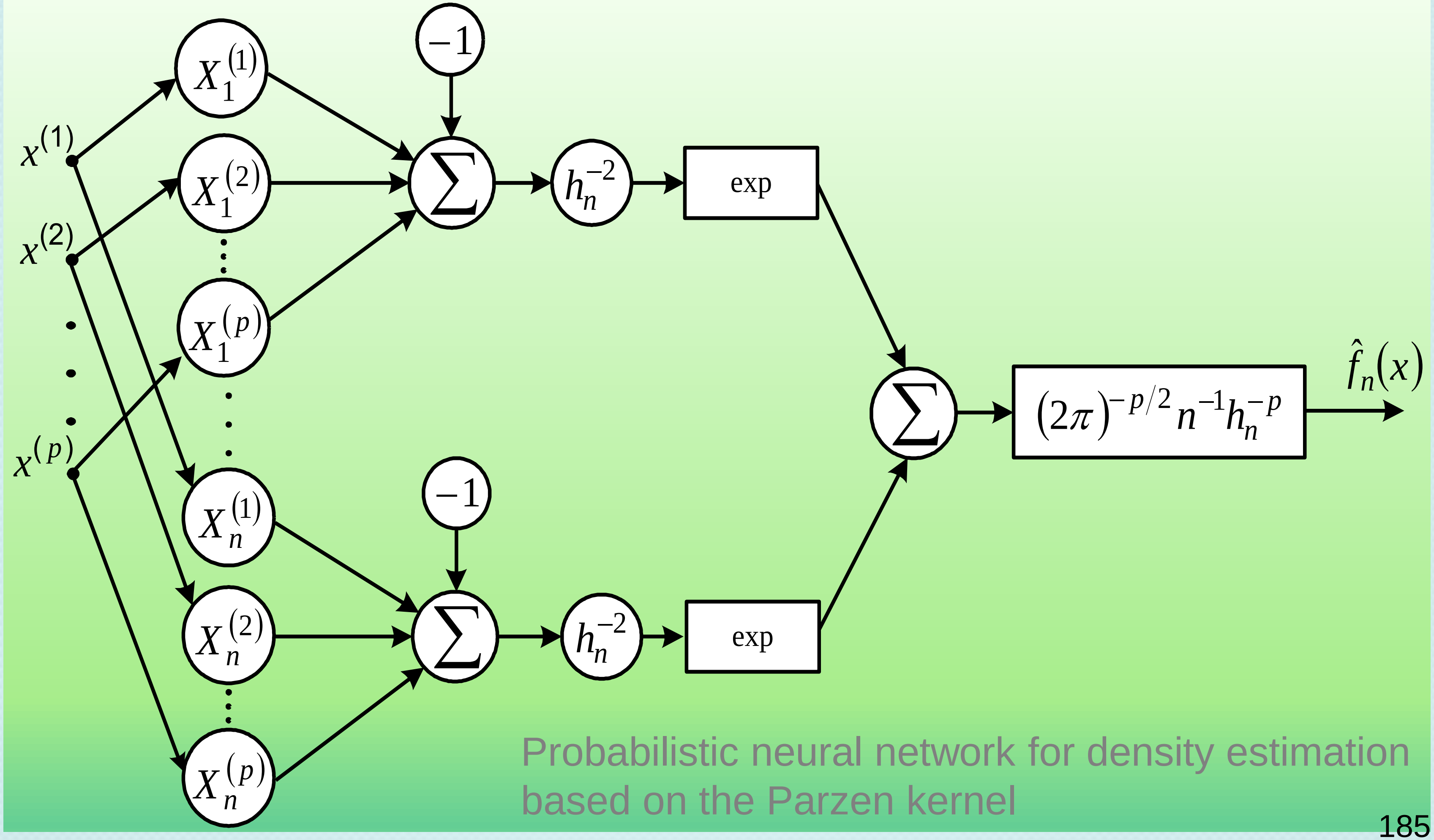
$$H(\upsilon)=(2\pi)^{-\frac{1}{2}}e^{-\frac{1}{2}\upsilon^2}$$

$$K_n(x,u)=h_n^{-p}(2\pi)^{-\frac{1}{2}}\prod_{j=1}^{p}\exp\left(-\frac{1}{2}\left(\frac{x^{(j)}-u^{(j)}}{h_n}\right)^2\right)$$

$$\hat{f}_n(x)=\frac{1}{(2\pi)^{\frac{1}{2}}nh_n^p}\sum_{i=1}^{n}\prod_{j=1}^{p}\exp\left(-\frac{1}{2}\left(\frac{x^{(j)}-X_i^{(j)}}{h_n}\right)^2\right)$$

PNN for stream data mining

Recursive procedures



Probabilistic neural network for density estimation based on the Parzen kernel

PNN for stream data mining

Non-recursive procedures

Let us modify estimator as follows

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n} K_n(x, X_i) \rightarrow \hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n} K_i(x, X_i)$$

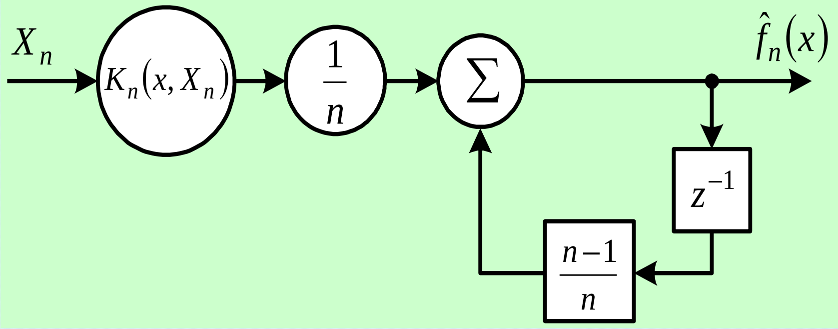
PNN for stream data mining

Recursive procedures

Observe that estimator is computationally equivalent to the recursive procedure

$$\hat{f}_{n+1}(x) = \hat{f}_n(x) + \frac{1}{n+1}\left[K_{n+1}(x, X_{n+1}) - \hat{f}_n(x)\right]$$

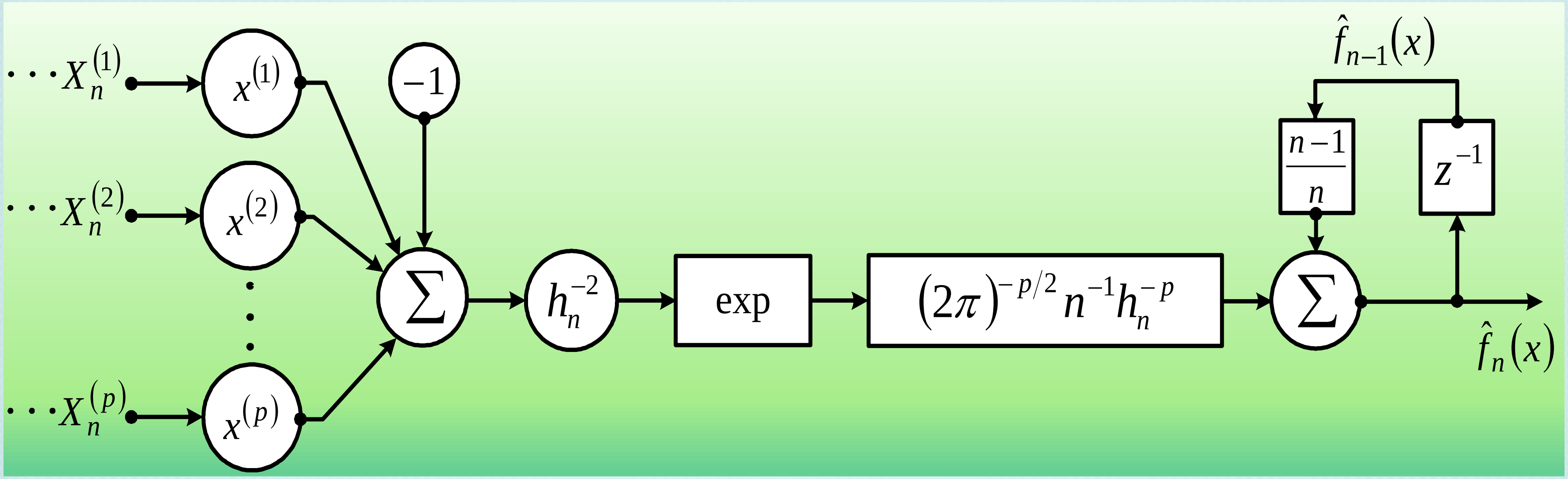
$$\hat{f}_0(x) = 0$$



Recursive probabilistic neural network for density estimation

PNN for stream data mining

Recursive procedures



Recursive probabilistic neural network for estimation based on the Parzen kernel

PNN for stream data mining

for pattern classification

Let $(X, Y), (X_1, Y_1), \ldots, (X_n, Y_n)$ be a sequence of i.i.d. pairs of random variables, Y takes values in the set of classes $S = \{1, \ldots, M\}$, whereas x takes values in $A \subset \mathbb{R}^p$. The problem is to estimate Y from x and w_n, where $w_n = (X_1, Y_1), \ldots, (X_n, Y_n)$ is a learning sequence. Suppose that p_m and f_m, $m = 1, \ldots, M$ are the prior class probabilities and class conditional densities, respectively. We define a discriminant function of class j:

$$d_j(x) = p_j f_j(x)$$

PNN for stream data mining

for pattern classification

Let $L(i, j)$ be the loss incurred in taking action $i \in S$ when the class is j. We assume $0 - 1$ loss function. For a decision function $\varphi: A \rightarrow S$ the expected loss is

$$R(\varphi) = \sum_{j=1}^{M} p_j \int_A L(\varphi(x), j) f_j(x) dx$$

PNN for stream data mining

for pattern classification

A decision function $\boldsymbol{\varphi}^*$ which classifies every as coming from any class m for which

$$p_m f_m(x) = \max_j \; p_j f_j(x) = \max_j \; d_j(x)$$

is a Bayes decision function and

$$R^* = R(\varphi^*) = \sum_{j=1}^{M} p_j \int_A L(\varphi^*(x), j) f_j(x) dx$$

is the minimal Bayes risk. The function $\boldsymbol{d_m}(\boldsymbol{x})$ is called the Bayes discriminant function.

PNN for stream data mining

for pattern classification

A decision function φ^* which classifies every $x \in A$ as coming from any class m for which

$$p_m f_m(x) = \max_j \; p_j f_j(x) = \max_j d_j(x)$$

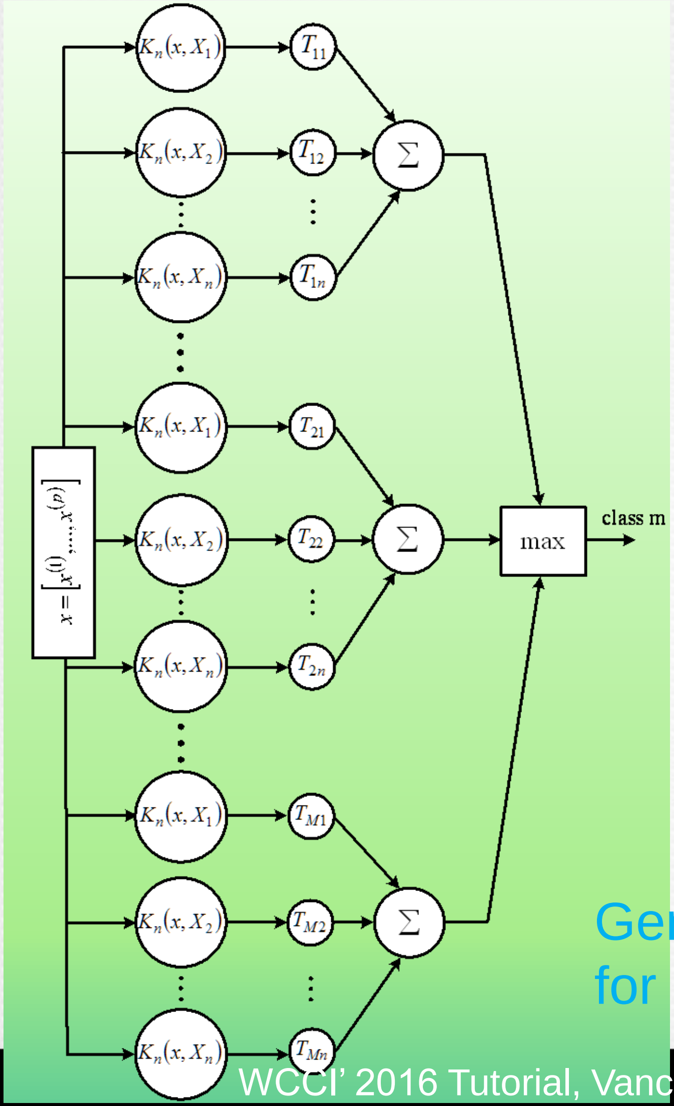
is a Bayes decision function and

$$R^* = R(\varphi^*) = \sum_{j=1}^{M} p_j \int_A L(\varphi^*(x), j) f_j(x) dx$$

is the minimal Bayes risk. The function $d_m(x)$ is called the Bayes discriminant function.

PNN for stream data mining

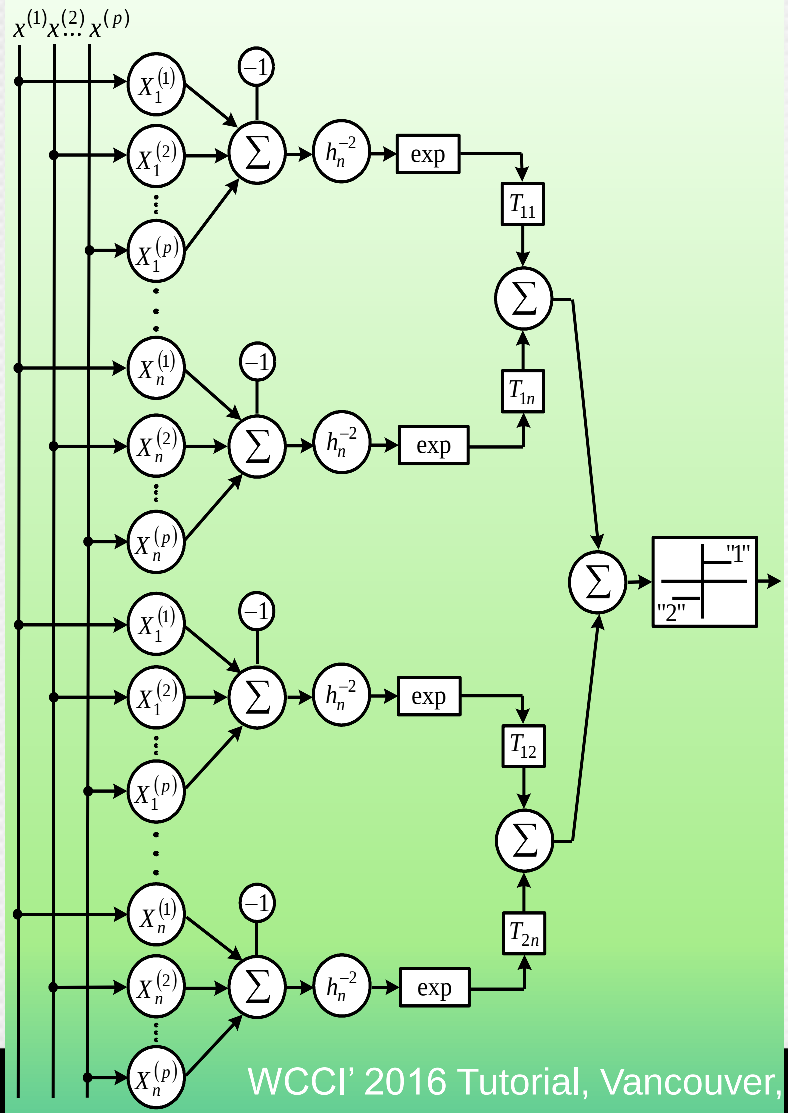
for pattern classification



Generalized regression neural network for pattern classification

PNN for stream data mining

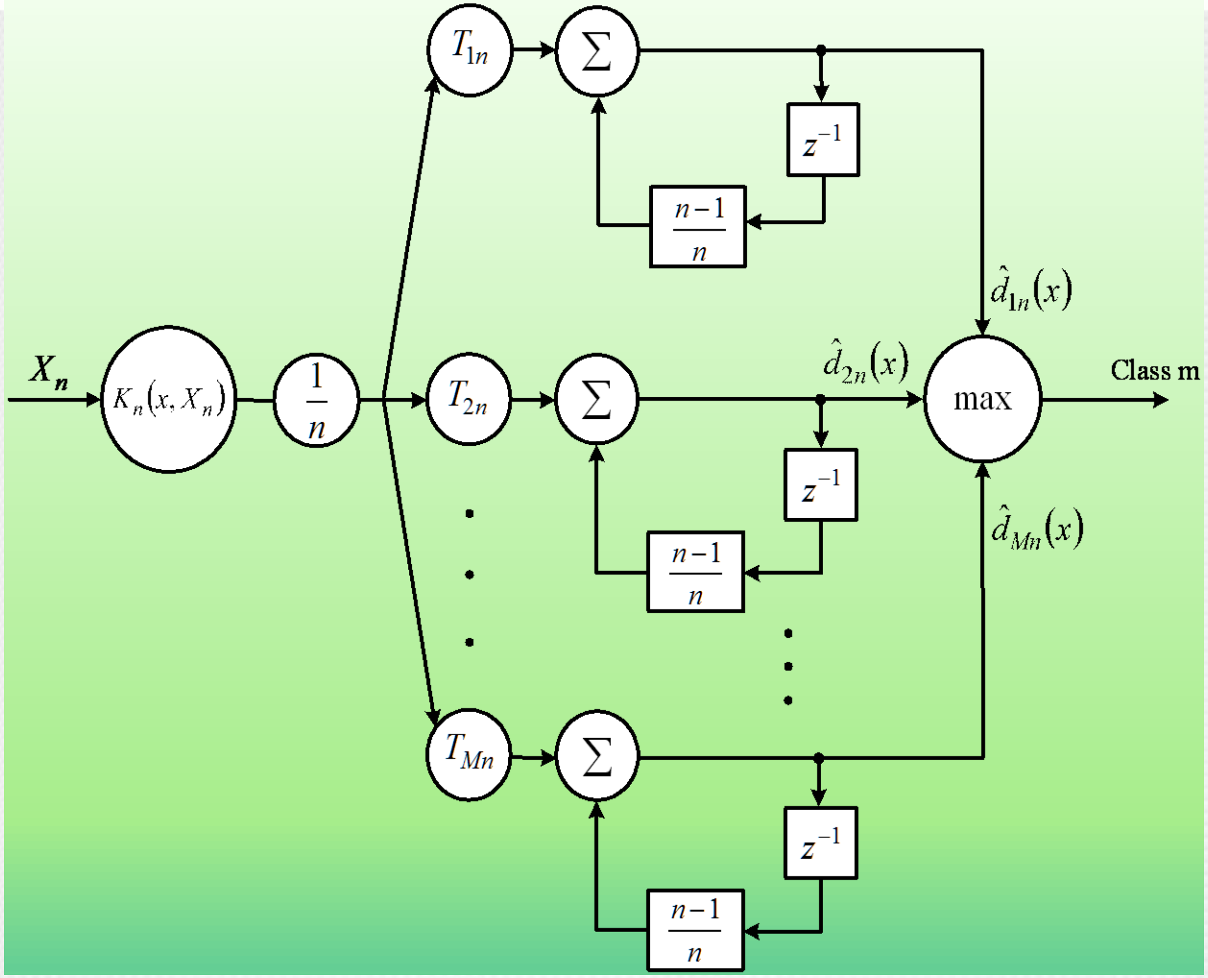
for pattern classification



Generalized regression neural network based on the Parzen kernel for pattern classification (M=2)

PNN for stream data mining

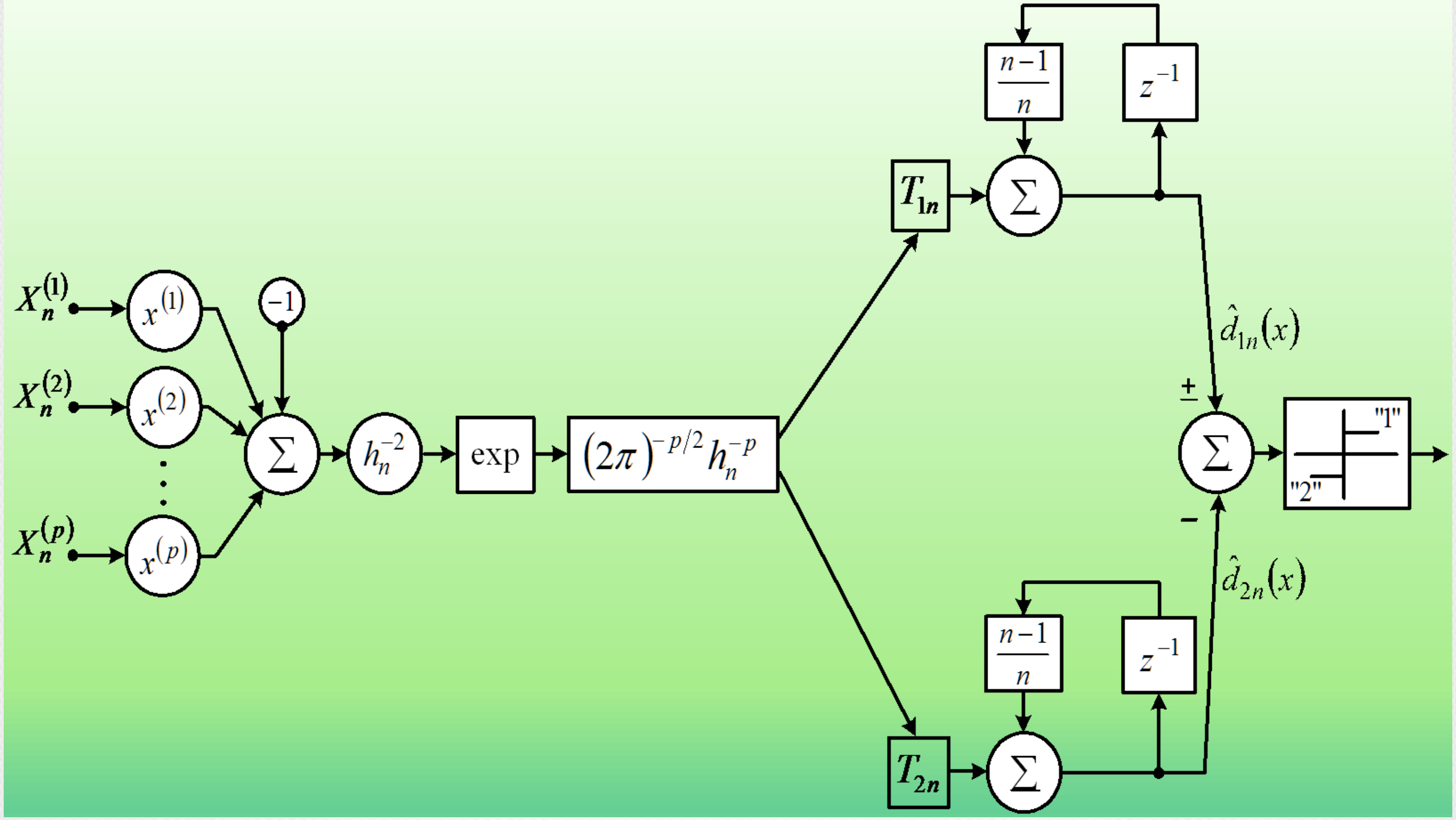
for pattern classification



Recursive generalized regression neural network for pattern classification

PNN for stream data mining

for pattern classification



Recursive generalized regression neural network based on the Parzen kernel for pattern classification (M=2)

PNN for stream data mining

for pattern classification in a time-varying environment

K – **number of classes**

Concept Drift:

$f_{mn}(x)$ – **time-varying probability density of class m ($\boldsymbol{m} = 1, . . , \boldsymbol{K}$) at the instant n ($\boldsymbol{n} = 1, 2, \ldots$)**

p_{mn} – **time-varying a priori probabilities**

PNN for stream data mining

for pattern classification in a time-varying environment

We want to estimate

$$d_{mn}(x) = p_{mn}\, f_{mn}(x), \qquad m = 1,\ldots,K, \quad n = 1,2,\ldots,$$

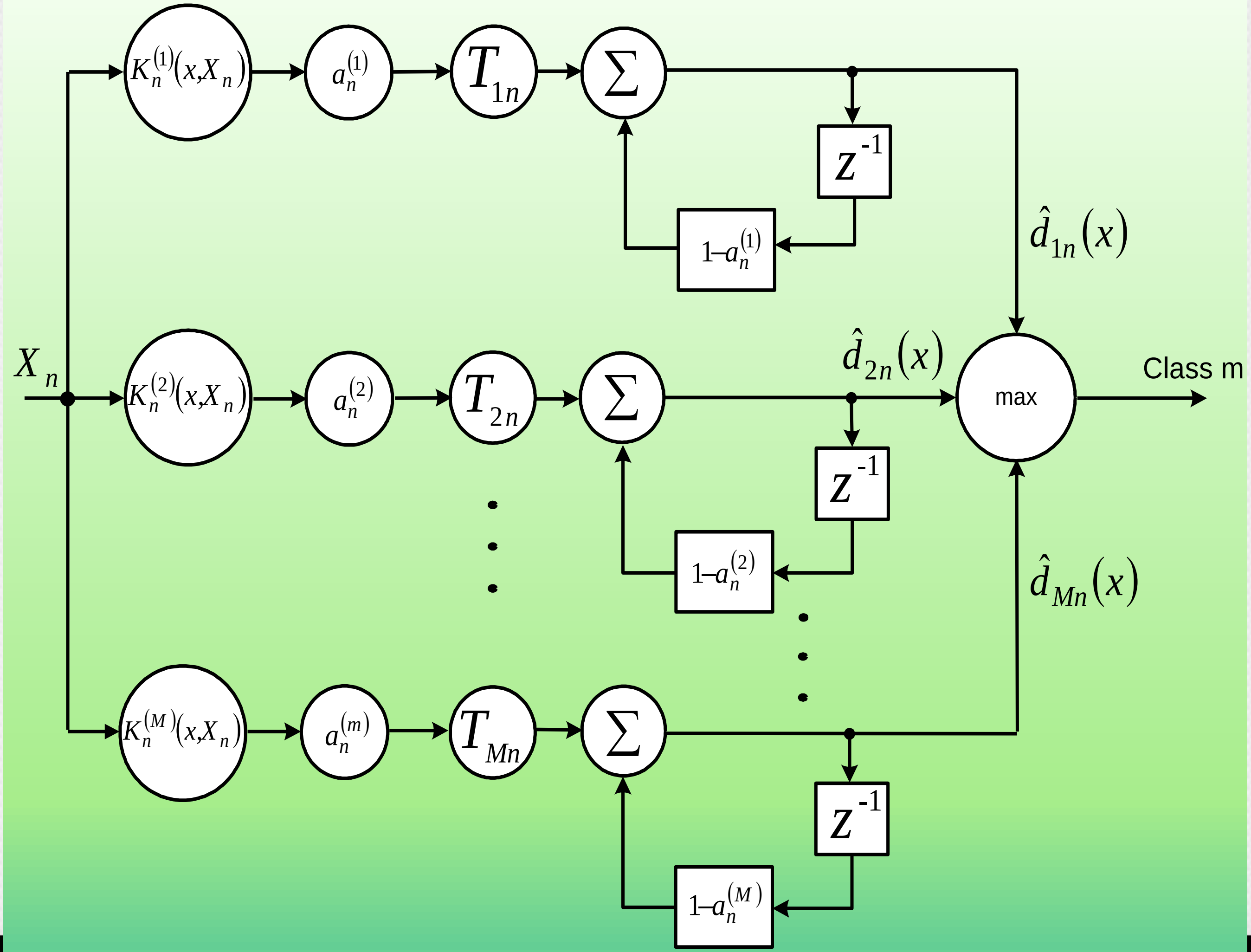
Find estimate $\hat{d}_{mn}$ of d_{mn} such that

$$\left|\hat{d}_{mn}(x) - d_{mn}(x)\right| \xrightarrow{n} 0$$

in probability
(with probability one)

PNN for stream data mining

for pattern classification in a time-varying environment



PNN for stream data mining

for pattern classification in a time-varying environment

Example

Consider a two-category classification problem with $p_{1n} = p_{2n} = \frac{1}{2}$ and

$$f_{1n}(x) = f_1(x - n^t), f_{2n}(x) = f_2(x - n^t)$$

where

$$f_1(x) = \mathcal{N}(0,1), f_2(x) = \mathcal{N}(2,1)$$

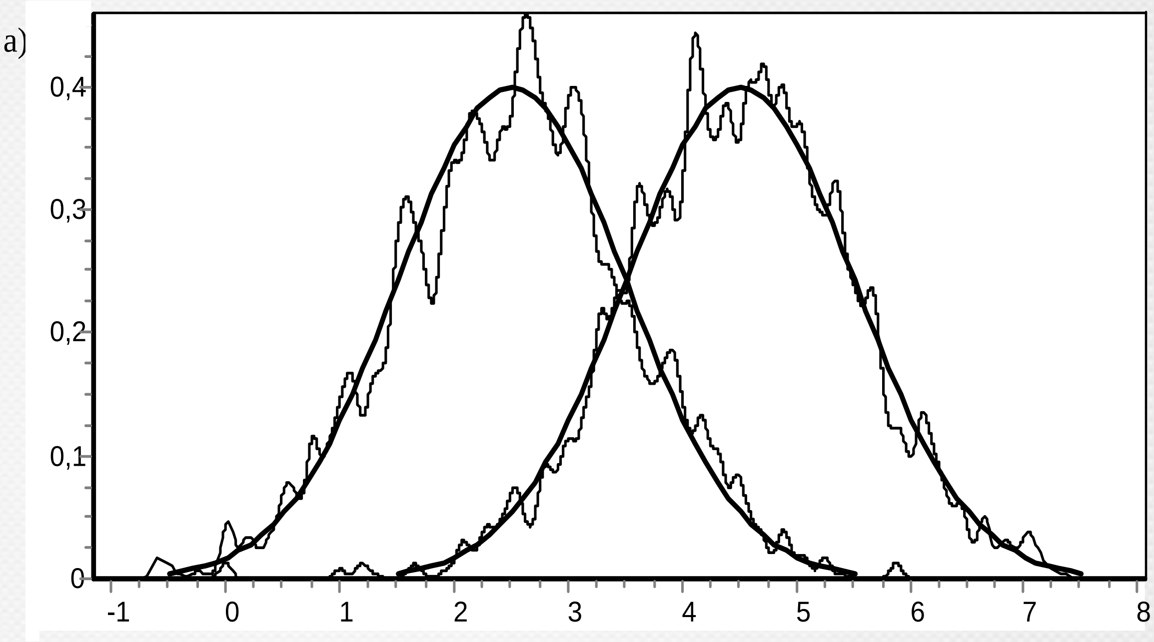
In this case the minimum probability of error is given by

$$P_e = \frac{1}{\sqrt{2\pi}} \int_1^\infty e^{-u^{2/2}} du = 0.159$$

PNN for stream data mining

for pattern classification in a time-varying environment

n	Error
1000	0.1559
2000	0.1602
3000	0.1573
4000	0.1691
5000	0.1566
6000	0.1598
7000	0.1562
8000	0.1607
9000	0.1561
10000	0.1572

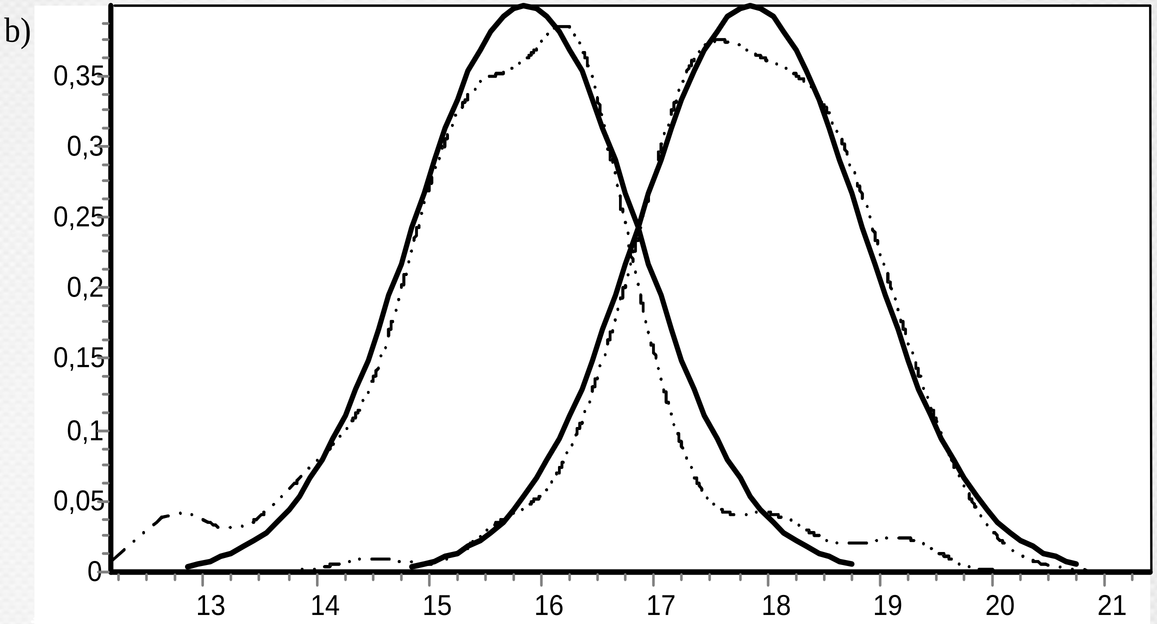


Empirical probability of misclassification
for *t*=0.1, *a*=0.7, *H*=0.5

PNN for stream data mining

for pattern classification in a time-varying environment

n	Error
1000	0.1639
2000	0.1648
3000	0.1687
4000	0.1649
5000	0.1564
6000	0.1611
7000	0.1558
8000	0.1586
9000	0.1569
10000	0.1562



Empirical probability of misclassification
for *t*=0.3, *a*=0.4, *H*=0.3

PNN for stream data mining

Generalized regression neural networks

Let (X, Y) be a pair of random variables X takes values in a Borel $A, A \subset \mathbb{R}^p$, whereas Y takes values in $\mathbb{R}$. Let f be the marginal Lebesgue density of X. Based on a sample $(X_1, Y_1), \dots, (X_n, Y_n)$ of independent observations of (X, Y) we wish to estimate the regression function

$$\phi(x) = E\left[Y \middle| X = x\right]$$

PNN for stream data mining

Generalized regression neural networks

To estimate function ϕ we propose the following formula

$$\hat{\phi}_n(x) = \frac{\hat{R}_n(x)}{\hat{f}_n(x)}$$

where

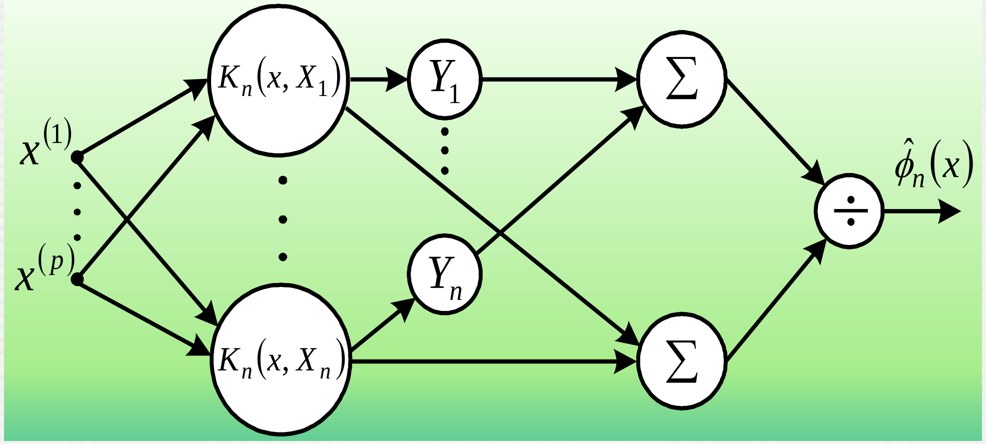
$$\hat{R}_n(x) = \frac{1}{n}\sum_{i=1}^{n} Y_i K_i(x, X_i)$$

and

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n} K_i(x, X_i)$$

PNN for stream data mining

Generalized regression neural networks



Scheme of generalized regression neural network

PNN for stream data mining

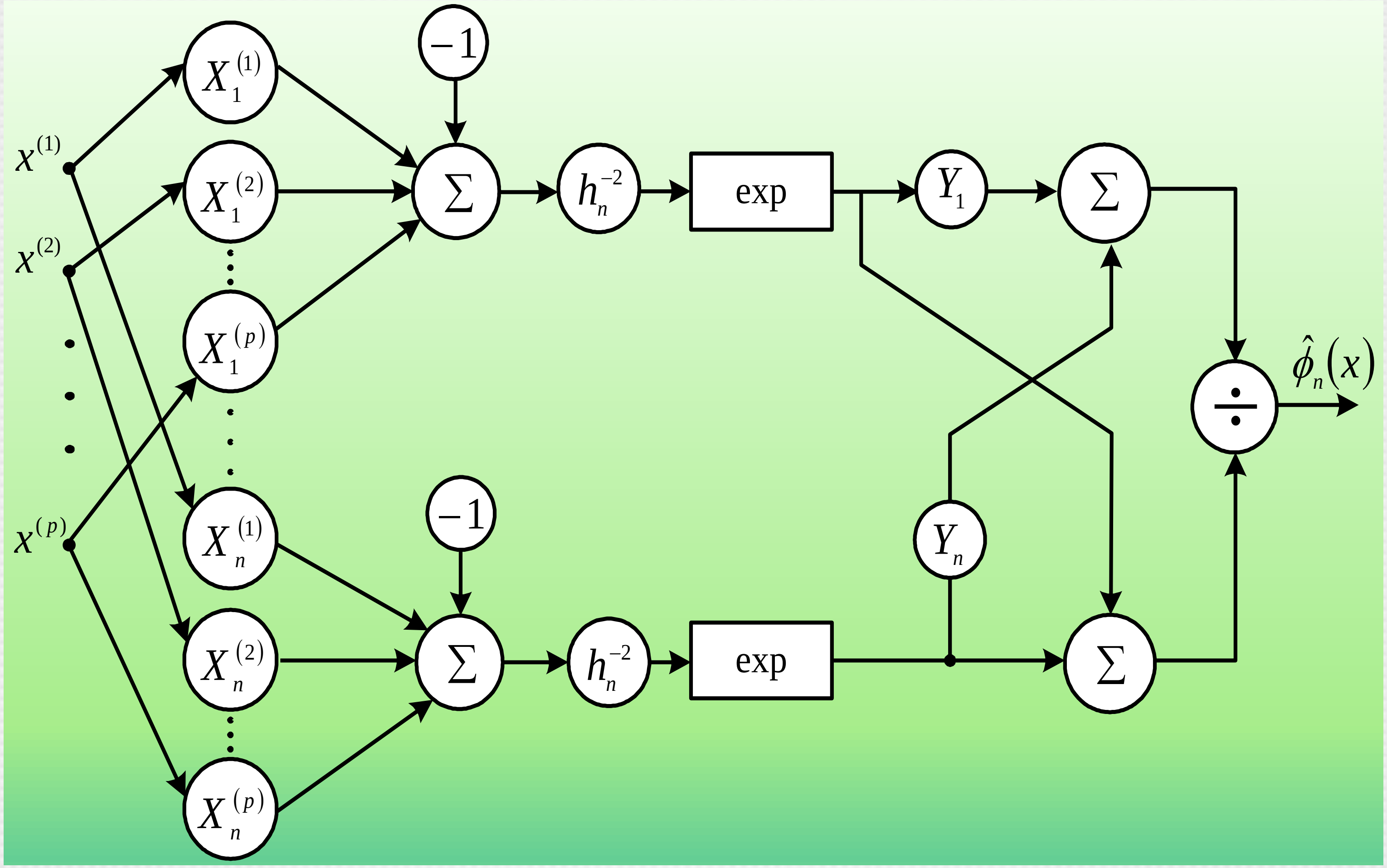
Generalized regression neural networks

Example (Nadaraya and Watson). Applying the Parzen kernel we get

$$\hat{\phi}_n(x) = \frac{\sum_{i=1}^{n} Y_i K\left(\frac{x - X_i}{h_n}\right)}{\sum_{i=1}^{n} K\left(\frac{x - X_i}{h_n}\right)}$$

PNN for stream data mining

Generalized regression neural networks



Generalized regression neural network based on the Parzen kernel

PNN for stream data mining

Generalized regression neural networks

The recursive version of procedure is given as follows

$$\hat{\phi}_n(x) = \frac{\hat{R}_n(x)}{\hat{f}_n(x)}$$

where

$$\hat{R}_n(x) = \frac{1}{n}\sum_{i=1}^{n} Y_i K_i(x, X_i)$$

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n} K_i(x, X_i)$$

PNN for stream data mining

Generalized regression neural networks

$$\hat{R}_{n+1}(x) = \hat{R}_n(x) + \frac{1}{n+1}\left[Y_{n+1}K_{n+1}(x, X_{n+1}) - \hat{R}_n(x)\right]$$

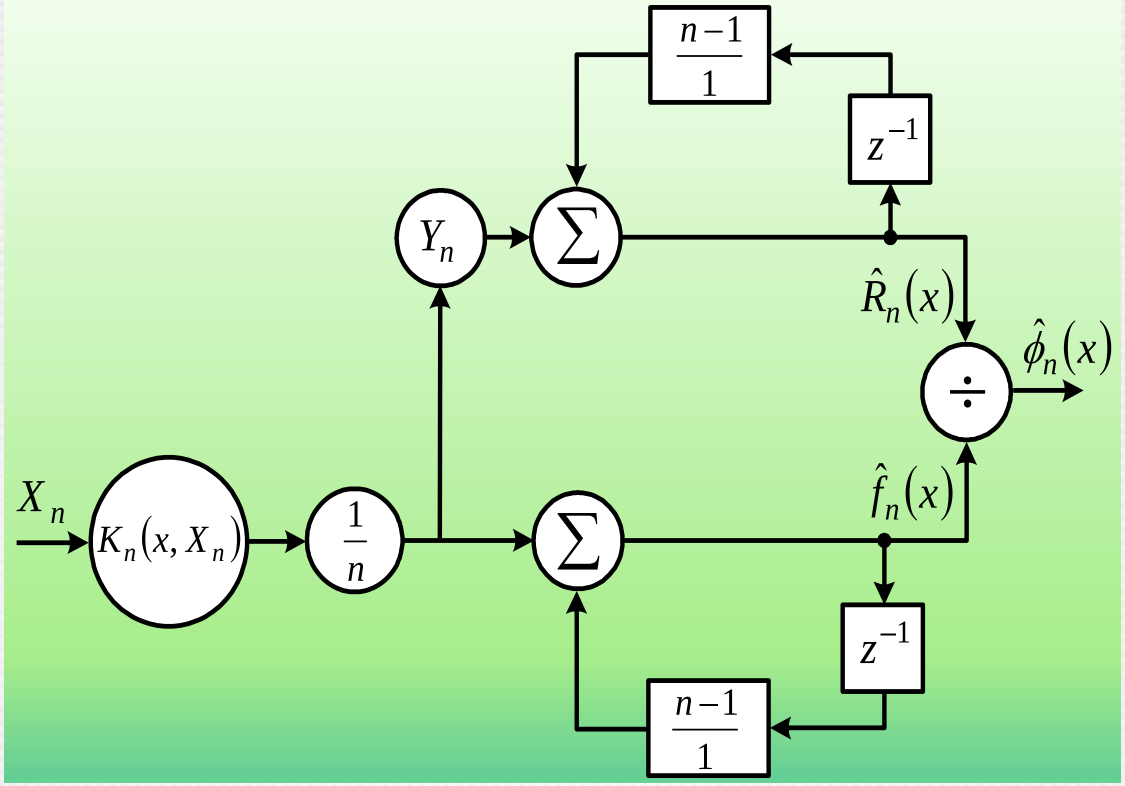
$$\hat{f}_{n+1}(x) = \hat{f}_n(x) + \frac{1}{n+1}\left[K_{n+1}(x, X_{n+1}) - \hat{f}_n(x)\right]$$

$$\hat{f}_0(x) = 0$$

where $\hat{R}_0(x) = 0$ and $\hat{f}_0(x) = 0$

PNN for stream data mining

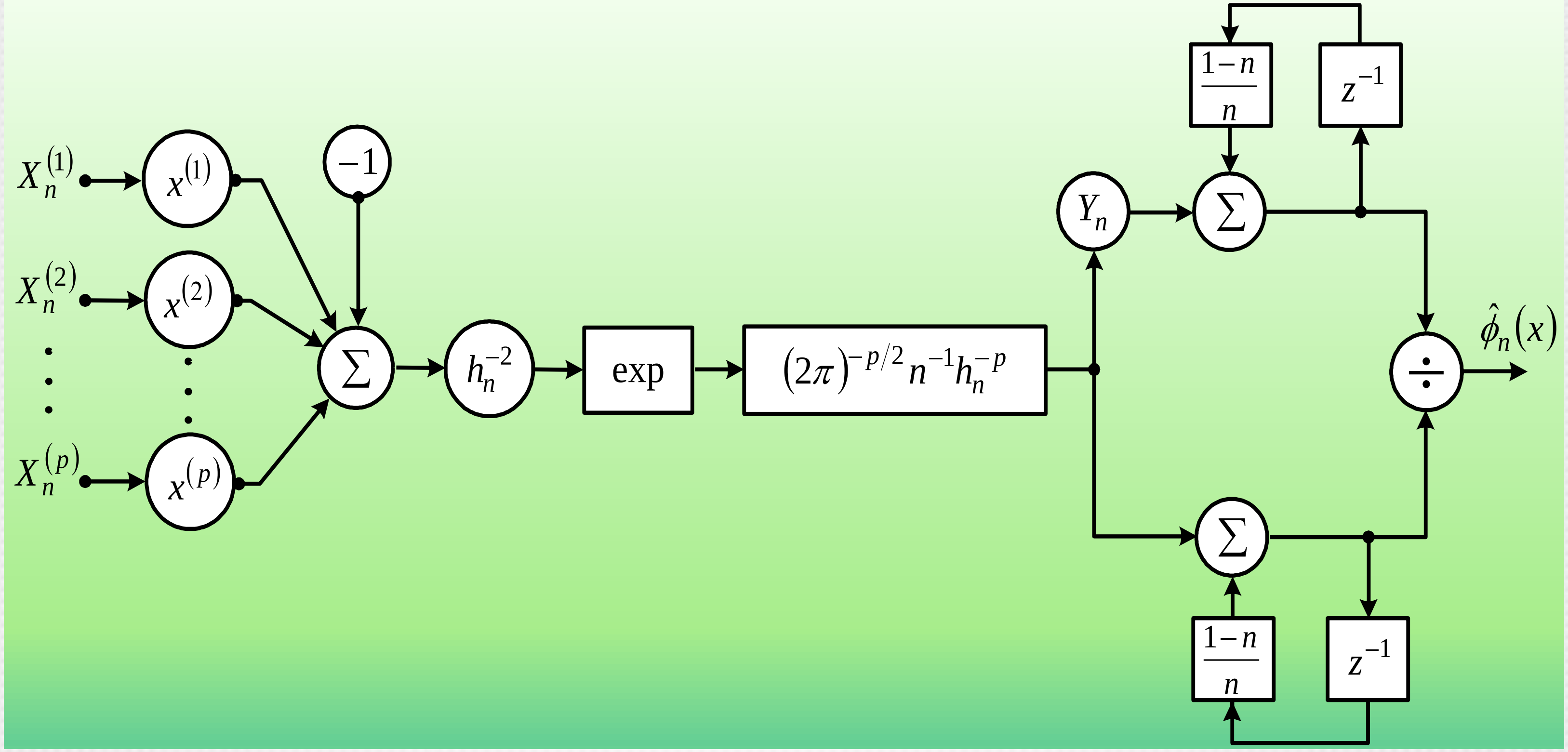
Generalized regression neural networks



Recursive generalized regression neural network

PNN for stream data mining

Generalized regression neural networks



Recursive generalized regression neural network based on the Parzen kernel

PNN for stream data mining

Generalized regression neural networks in time-varying environment

In the non-stationary regression we consider a sequence of random variables $(X_n, Y_n), n = 1,2, \dots,$ having time-varying cumulative probability density functions $f_n(x, y)$. The problem is to find a measurable function $\phi_n\colon \mathbb{R}^p \to \mathbb{R}$ such that the L_2 risk

$$E\left[\phi_n\left(X_n\right) - Y_n\right]^2$$

attains minimum. The solution is the regression function

$$\phi_n^*\left(x\right) = E\left[Y_n \middle| X_n = x\right], \quad n = 1, 2, \dots,$$

changing with time.

PNN for stream data mining

Generalized regression neural networks in time-varying environment

For the illustration of the capability of our GRNN we may consider an application to modelling of non-stationary plants described by

$$Y_n = \phi_n^*(X_n) + Z_n$$

where concept drift $\phi_n{}^*$ is given by:

(i) $\phi_n^*(x) = \alpha_n \phi(x)$

(ii) $\phi_n^*(x) = \phi(x) + \beta_n$

(iii) $\phi_n^*(x) = \phi(x\omega_n)$

(iv) $\phi_n^*(x) = \phi(x - \lambda_n)$

PNN for stream data mining

Generalized regression neural networks in time-varying environment

The problem of non-parametric regression boils down to finding an adaptive algorithm that could follow the changes of optimal characteristics expressed by formula. This algorithm should be constructed on the basis of a learning sequence, i.e. observations of the following random variables $(X_1, Y_1), (X_2, Y_2), \ldots$

We assume that pairs of the above random variables are independent.

In points x, where $f(x) \neq 0$, the characteristics of the best model can be expressed as

$$\phi_n^*(x) = R_n(x) / f(x), \quad n = 1, 2, \ldots,$$

where $R_n(x) = \phi_n^*(x) f(x)$.

PNN for stream data mining

Generalized regression neural networks in time-varying environment

The algorithm has the form

$$\boxed{\hat{\phi}_n(x) = \hat{R}_n(x) / \hat{f}_n(x)}$$

$$\hat{R}_{n+1}(x) = \hat{R}_n(x) + a_{n+1}\left[Y_{n+1} K_{n+1}(x, X_{n+1}) - \hat{R}_n(x)\right]$$

$$\hat{f}_{n+1}(x) = \hat{f}_n(x) + \frac{1}{n+1}\left(K_{n+1}(x, X_{n+1}) - \hat{f}_n(x)\right)$$

where $\hat{R}_0{}^{*}(x)$=0 and $\hat{f}_0{}^{*}(x)$=0

PNN for stream data mining

Generalized regression neural networks in time-varying environment

Let

$$a_n > 0, \quad a_n \xrightarrow{n} 0, \quad \sum_{n=1}^{\infty} a_n = \infty$$

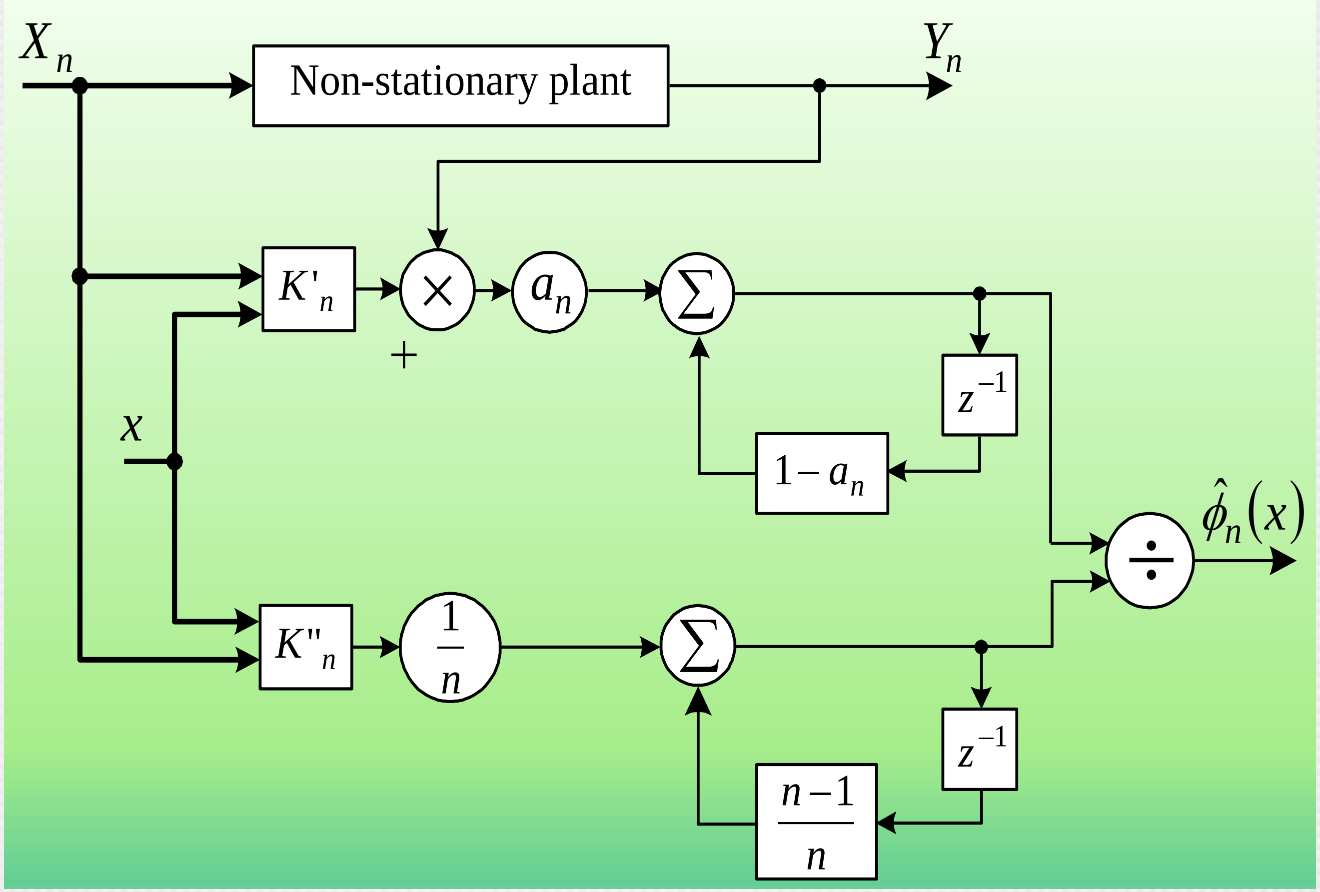
Under specific conditions imposed on parameters a_n and h_n and the rate of change of function ϕ_n the following holds

$$\left|\hat{\phi}_n(x) - \phi_n(x)\right| \xrightarrow{n} 0$$

in probability
(with probability one)

PNN for stream data mining

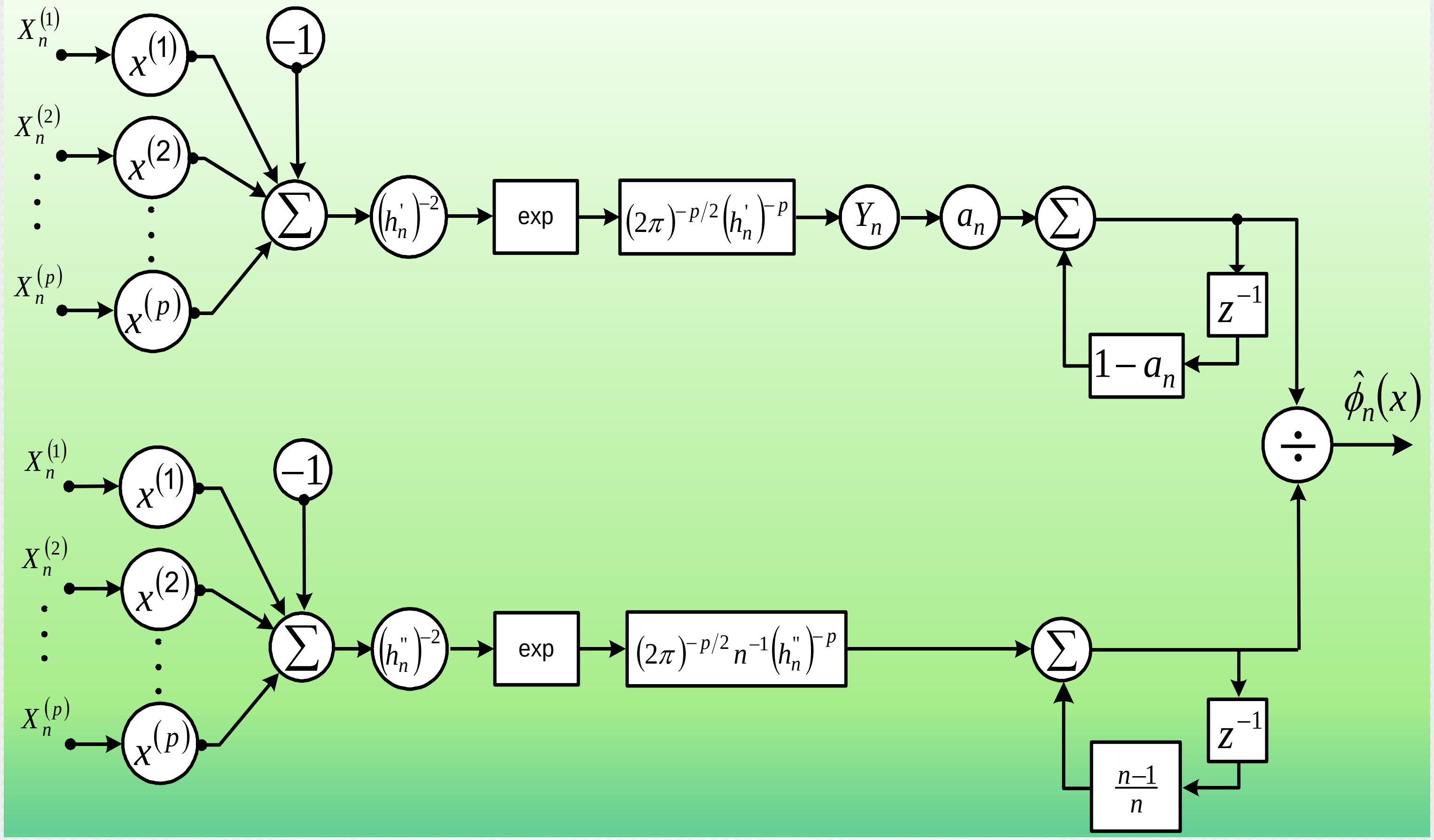
Generalized regression neural networks in time-varying environment



Block diagram of the GRNN applied to modelling of non-stationary plant

PNN for stream data mining

Generalized regression neural networks in time-varying environment



Block diagram of the GRNN based on the Parzen kernel

PNN for stream data mining

Generalized regression neural networks in time-varying environment

Example

In order to track changes of the system described by

$$y_n = \left(c_1 n^t + c_2 \log n + c_3\right) \phi\left(x_n\right) + z_n$$

where $t > 0$ is an unknown parameter and ϕ is an unknown function, it is possible to use algorithm if

a) $0 < t < \frac{1}{3}$ for week convergence

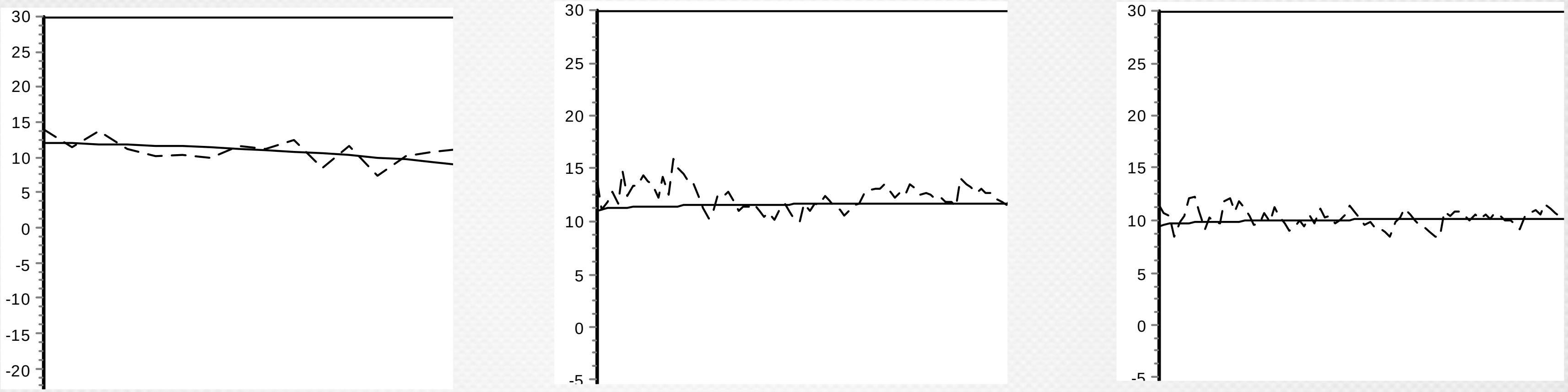
b) $0 < t < \frac{1}{6}$ for strong convergence

PNN for stream data mining

Generalized regression neural networks in time-varying environment

Example

$$\phi_n{}^{*}(x_n)=\mathbf{10}\cos(x_n)+\boldsymbol{n}^{\mathbf{0.1}}+\boldsymbol{z_n}$$



a) n=1000 b) x=0.2 c) x=0.6

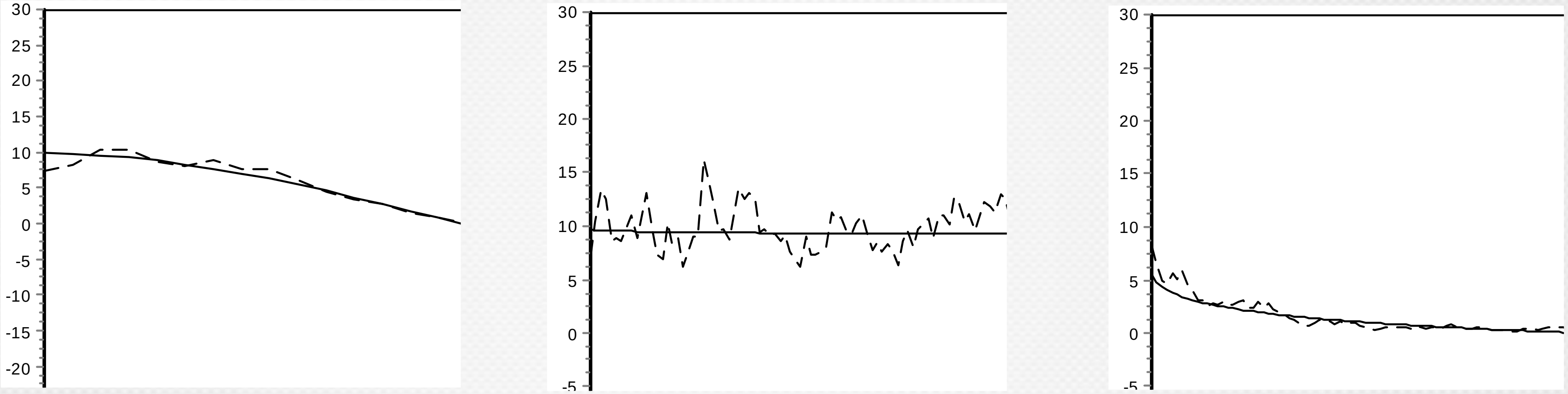
GRNN for modeling regressions with additive non-stationarity

PNN for stream data mining

Generalized regression neural networks in time-varying environment

Example

$$\phi_n^{\ *}(x_n)=\mathbf{10}\cos\left(x_n\boldsymbol{n}^{\mathbf{0.1}}\right)+\boldsymbol{z_n}$$



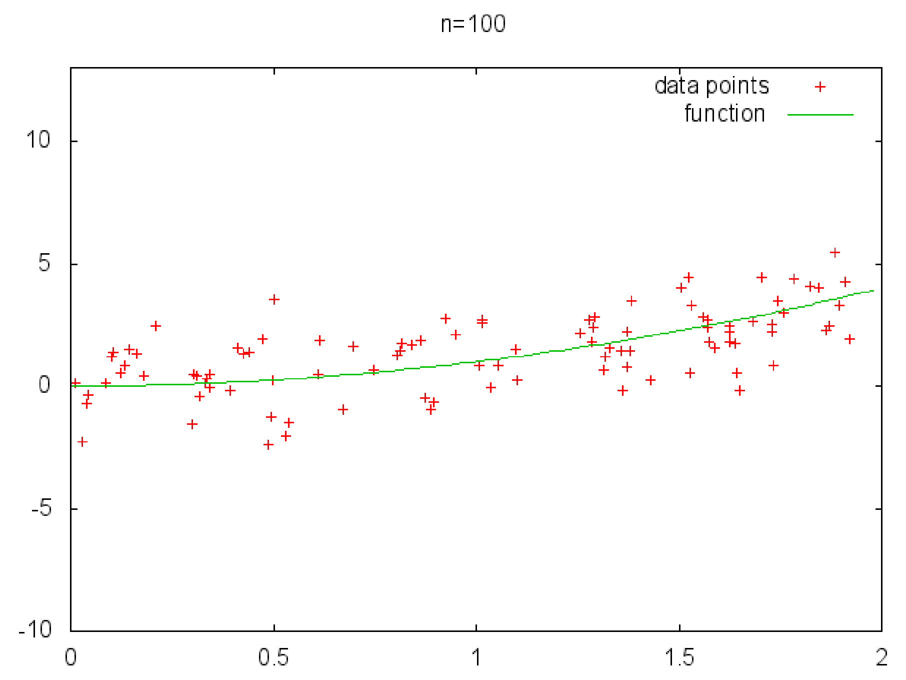
a) *n*=1000 b) *x=0.2* c) *x=0.8*

GRNN for modeling regressions with non-stationarity of the type “scale change”

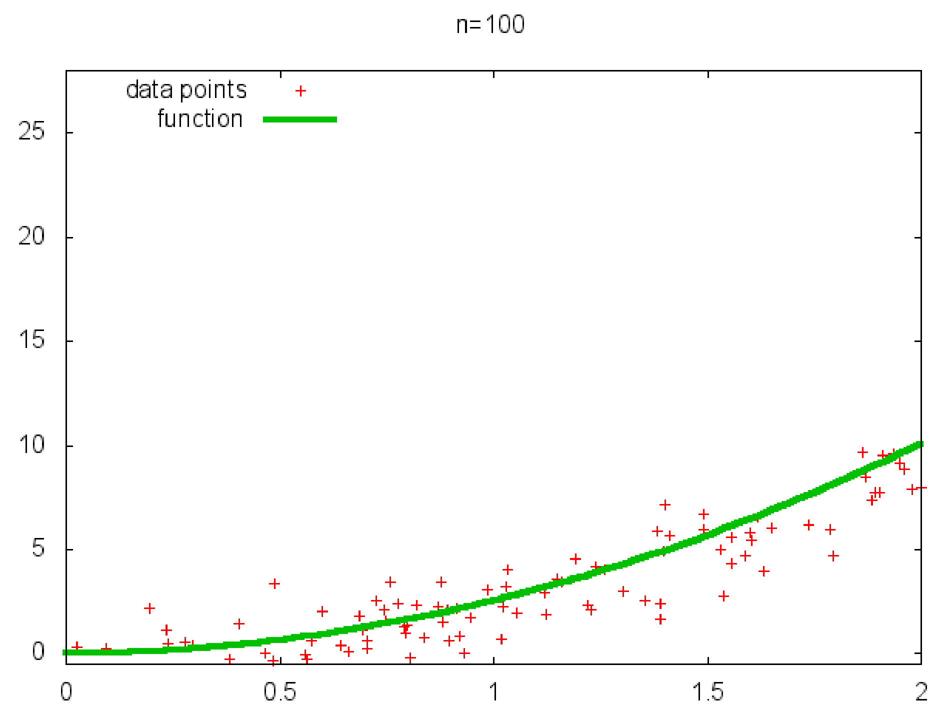
New result 2016

Three types of non-stationarities

1) $Y_n = \phi(X_n) + Z_n$
where variance of Z_n is changing over time



2) $Y_n = \phi_n(X_n) + Z_n$



3) $Y_n = \phi_n(X_n) + Z_n$ with changing variance of Z_n

New result 2016

Forgetting mechanism

$$\Phi_n^\lambda(x) = \frac{\sum_{i=1}^{n} \lambda^{n-i} Y_i \frac{1}{h_i^\lambda} K\left(\frac{x-X_i}{h_i^\lambda}\right)}{\sum_{i=1}^{n} \lambda^{n-i} \frac{1}{h_i'^\lambda} K\left(\frac{x-X_i}{h_i'^\lambda}\right)}$$

New result 2016

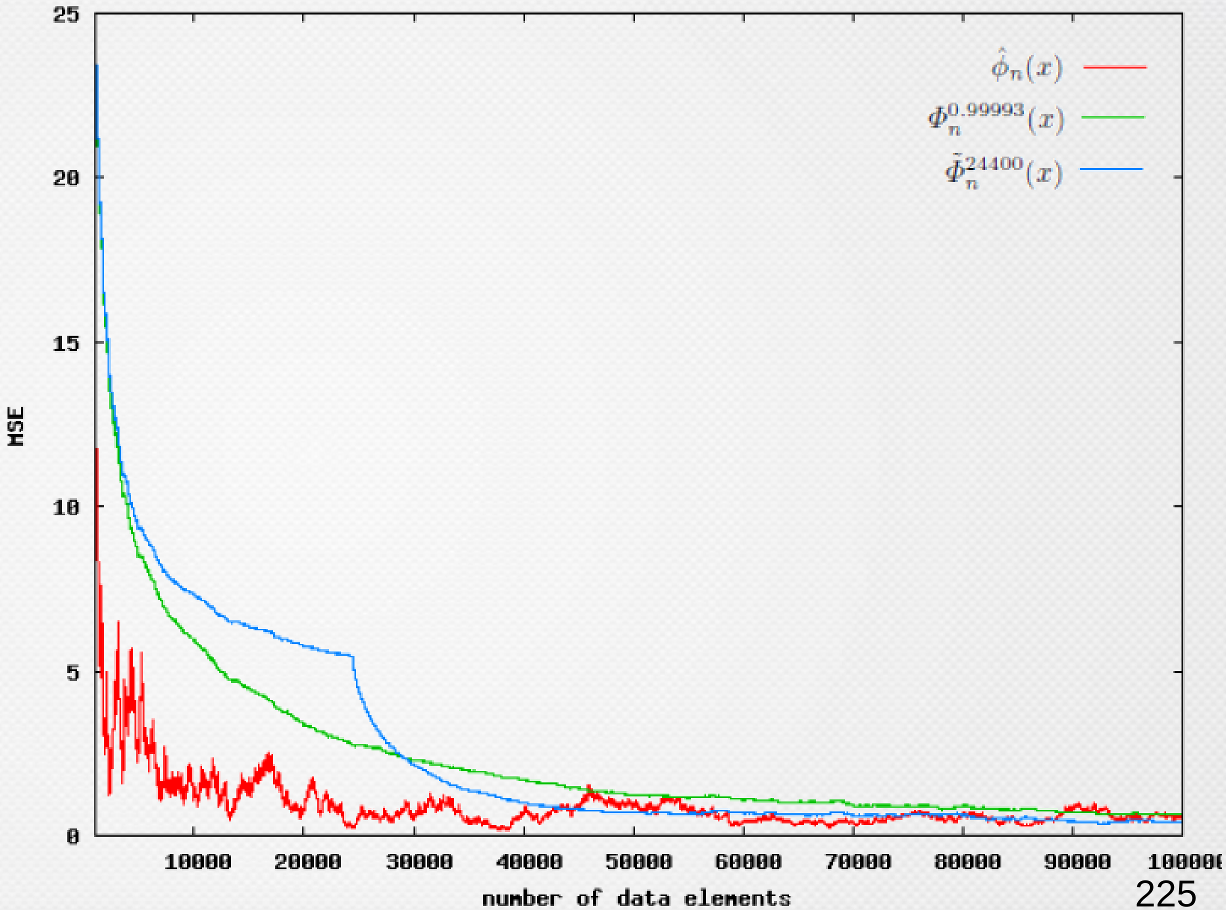
Sliding window

$$\tilde{\Phi}_n^W(x) = \frac{\sum_{i=n-W+1}^{n} Y_i \frac{1}{h_i^W} K\left(\frac{x-X_i}{h_i^W}\right)}{\sum_{i=n-W+1}^{n} \frac{1}{h_i'^W} K\left(\frac{x-X_i}{h_i'^W}\right)}$$

New result 2016

Experimental results

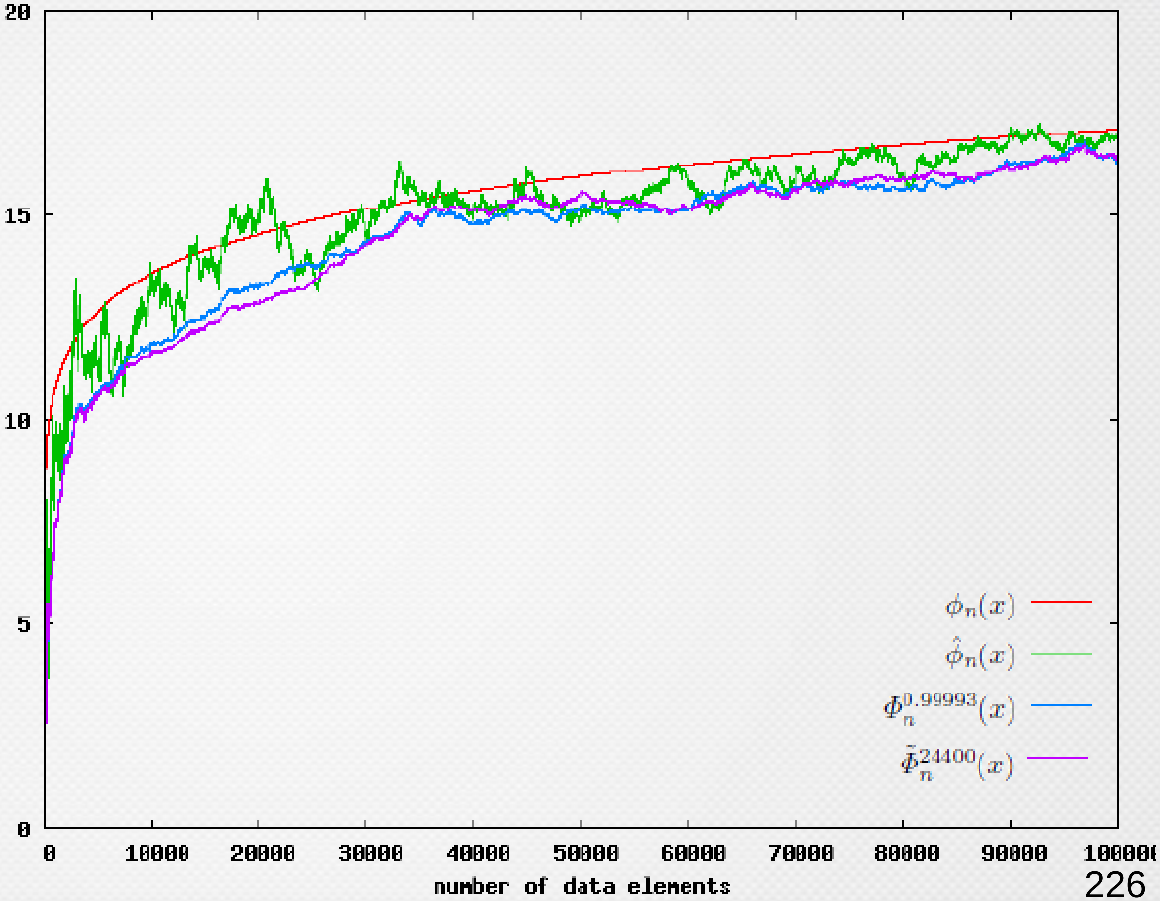
$$\phi_n(x) = 10 \cdot \cos(x) \cdot n^{0.1}$$



New result 2016

Experimental results

$$\phi_n(x) = 10 \cdot \cos(x) \cdot n^{0.1}$$



Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- **Final remarks and challenging problems**
- References

Data stream mining

- Final remarks and challenging problems

To be presented at the end of the tutorial.

Data stream mining

- content

- Data streams – introduction to the topic
- Concept drift
- Various strategies of learning
- How to deal with concept drift?
- Data stream classification methods – short overview
- Decision trees for data streams (including new results 2016)
- Ensemble methods for data streams (including new results 2016)
- Probabilistic neural networks for stream data mining (including new results 2016)
- Final remarks and challenging problems
- **References**

References

Stream data mining algorithms (concept drift)

Rutkowski L., Duda P., Jaworski M., Pietruczuk L., Stream Data Mining: Algorithms and Their Probabilistic Properties, Studies in Big Data, Springer, 2017.

This book shows methods and algorithms which are mathematically justified.

(1) It shows how to adopt the static decision trees, like ID.3 or CART, to deal with data streams.

(2) A new technique, based on the McDiarmid bound, is developed.

(3) New decision trees are designed, by a proper combination of the Gini index and misclassification error impurity measures, leading to the original concept of the hybrid decision trees.

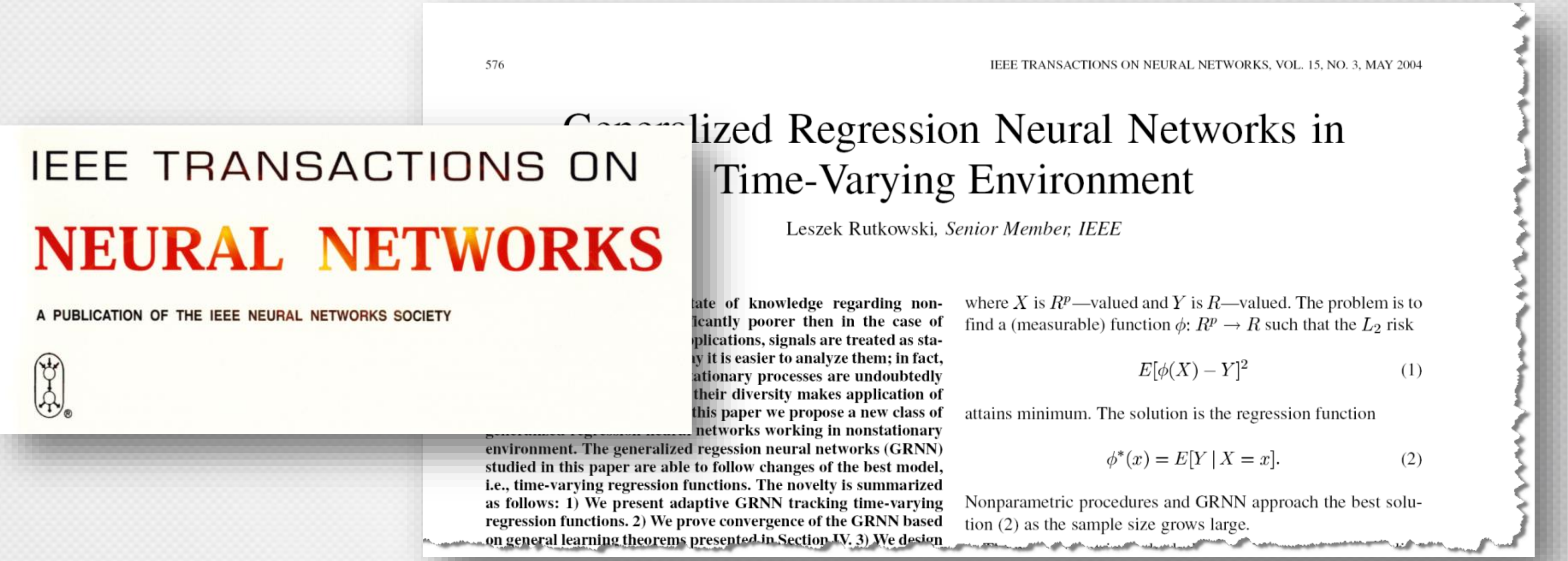
(4) The problem of designing ensembles and automatic choosing their sizes is described and solved.

(5) Nonparametric techniques based on the Parzen – kernels and orthogonal series, are adopted to deal with concept drift in the problem of non-stationary regressions and classification in time-varying environment. Nonparametric procedures are developed and their probabilistic properties are investigated.

References

Stream data mining algorithms (concept drift)

Rutkowski L., Generalized regression neural networks in time-varying environment, IEEE Transactions on Neural Networks, vol. 15, pp. 576-596, 2004.



IEEE TRANSACTIONS ON

NEURAL NETWORKS

A PUBLICATION OF THE IEEE NEURAL NETWORKS SOCIETY

576 IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 15, NO. 3, MAY 2004

lized Regression Neural Networks in
Time-Varying Environment

Leszek Rutkowski, *Senior Member, IEEE*

tate of knowledge regarding non-
ficantly poorer then in the case of
pplications, signals are treated as sta-
y it is easier to analyze them; in fact,
ationary processes are undoubtedly
their diversity makes application of
this paper we propose a new class of
networks working in nonstationary
environment. The generalized regession neural networks (GRNN) studied in this paper are able to follow changes of the best model, i.e., time-varying regression functions. The novelty is summarized as follows: 1) We present adaptive GRNN tracking time-varying regression functions. 2) We prove convergence of the GRNN based on general learning theorems presented in Section IV. 3) We design

where X is R^p—valued and Y is R—valued. The problem is to find a (measurable) function ϕ: $R^p \to R$ such that the L_2 risk

$$E[\phi(X) - Y]^2 \quad (1)$$

attains minimum. The solution is the regression function

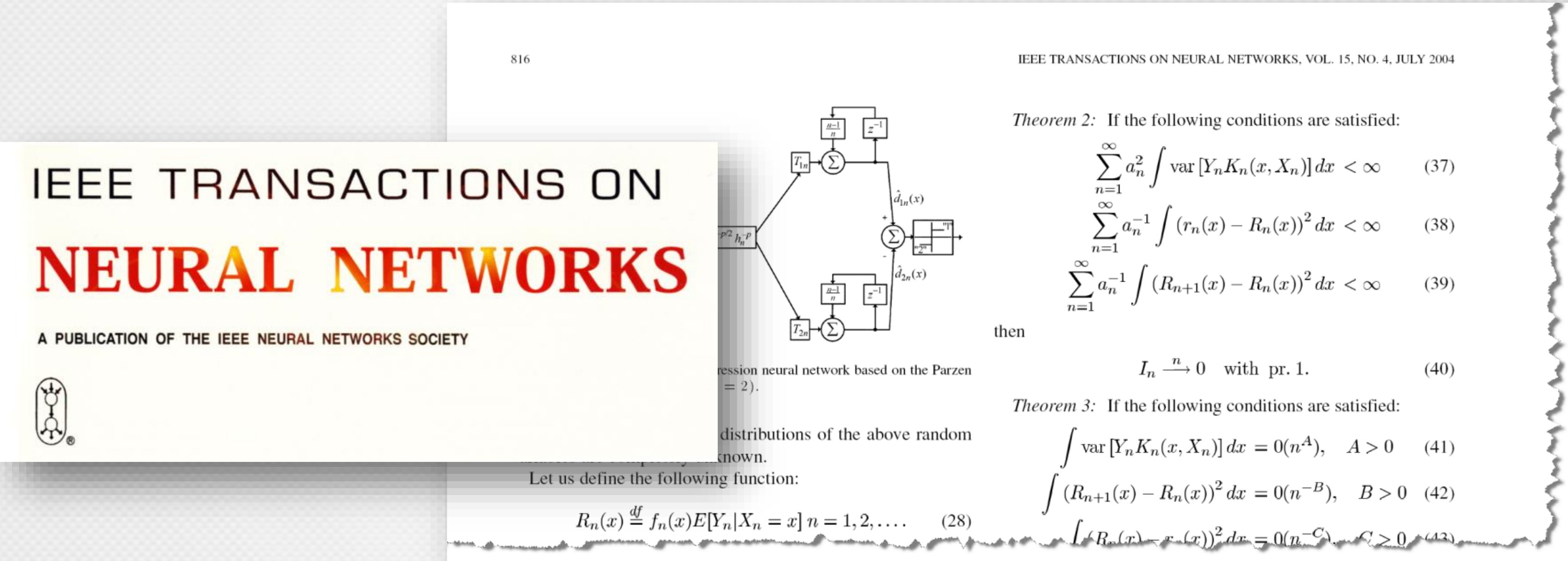
$$\phi^*(x) = E[Y \mid X = x]. \quad (2)$$

Nonparametric procedures and GRNN approach the best solution (2) as the sample size grows large.

References

Stream data mining algorithms (concept drift)

Rutkowski L., Adaptive probabilistic neural-networks for pattern classification in time-varying environment, IEEE Transactions on Neural Networks, vol. 15, 2004.



IEEE TRANSACTIONS ON

NEURAL NETWORKS

A PUBLICATION OF THE IEEE NEURAL NETWORKS SOCIETY

816 IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 15, NO. 4, JULY 2004

ression neural network based on the Parzen = 2).

distributions of the above random known.

Let us define the following function:

$$R_n(x) \overset{df}{=} f_n(x)E[Y_n|X_n = x]\, n = 1, 2, \ldots . \quad (28)$$

Theorem 2: If the following conditions are satisfied:

$$\sum_{n=1}^{\infty} a_n^2 \int \mathrm{var}\,[Y_n K_n(x, X_n)]\, dx < \infty \quad (37)$$

$$\sum_{n=1}^{\infty} a_n^{-1} \int (r_n(x) - R_n(x))^2\, dx < \infty \quad (38)$$

$$\sum_{n=1}^{\infty} a_n^{-1} \int (R_{n+1}(x) - R_n(x))^2\, dx < \infty \quad (39)$$

then

$$I_n \xrightarrow{n} 0 \quad \text{with pr. 1.} \quad (40)$$

Theorem 3: If the following conditions are satisfied:

$$\int \mathrm{var}\,[Y_n K_n(x, X_n)]\, dx = 0(n^A), \quad A > 0 \quad (41)$$

$$\int (R_{n+1}(x) - R_n(x))^2\, dx = 0(n^{-B}), \quad B > 0 \quad (42)$$

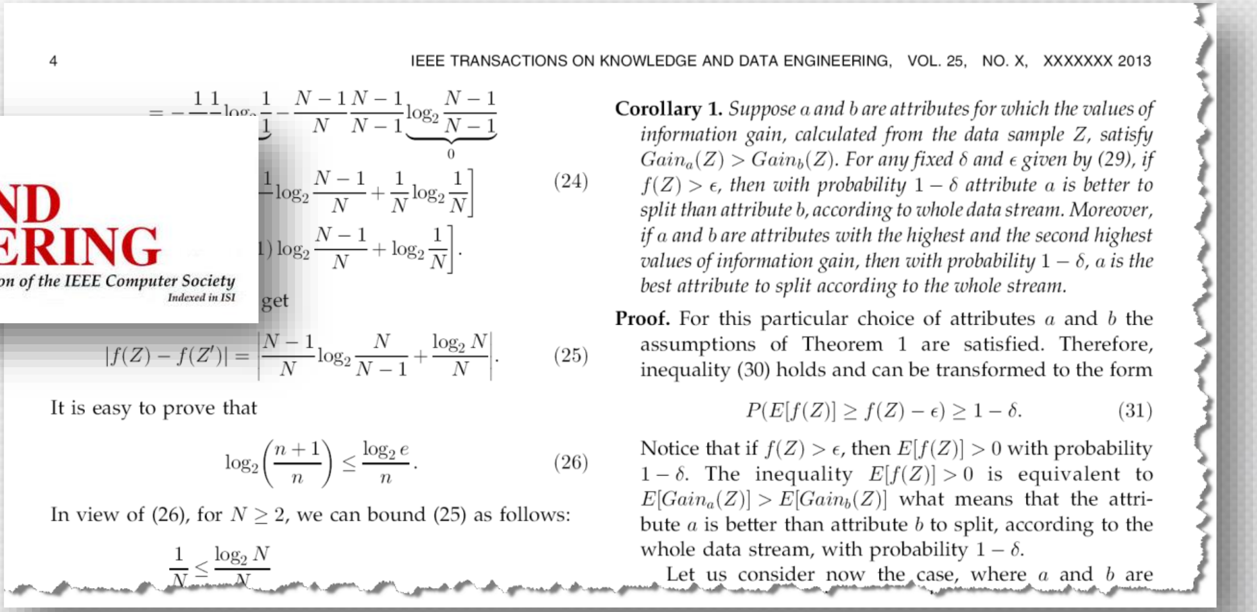
$$\int (R_n(x) - r_n(x))^2\, dx = 0(n^{-C}), \quad C > 0 \quad (43)$$

References

Stream data mining algorithms (concept drift)

Rutkowski L., Pietruczuk L., Duda P., Jaworski M., Decision Trees for Mining Data Streams Based on the McDiarmid's Bound, IEEE Transactions on Knowledge and Data Engineering, vol. 25, pp. 1272 – 1279, 2013.

IEEE TRANSACTIONS ON
KNOWLEDGE AND DATA ENGINEERING
A publication of the IEEE Computer Society
Indexed in ISI



4 IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. X, XXXXXXX 2013

$$= -\frac{1}{}\frac{1}{}\log_2\frac{1}{1} - \frac{N-1}{N}\frac{N-1}{N-1}\underbrace{\log_2\frac{N-1}{N-1}}_{0}$$

$$\log_2\frac{N-1}{N} + \frac{1}{N}\log_2\frac{1}{N}\Bigg] \qquad (24)$$

$$1)\log_2\frac{N-1}{N} + \log_2\frac{1}{N}\Bigg].$$

get

$$|f(Z) - f(Z')| = \left|\frac{N-1}{N}\log_2\frac{N}{N-1} + \frac{\log_2 N}{N}\right|. \qquad (25)$$

It is easy to prove that

$$\log_2\left(\frac{n+1}{n}\right) \leq \frac{\log_2 e}{n}. \qquad (26)$$

In view of (26), for $N \geq 2$, we can bound (25) as follows:

$$\frac{1}{N} \leq \frac{\log_2 N}{N}$$

Corollary 1. *Suppose a and b are attributes for which the values of information gain, calculated from the data sample Z, satisfy $Gain_a(Z) > Gain_b(Z)$. For any fixed δ and ϵ given by (29), if $f(Z) > \epsilon$, then with probability $1 - \delta$ attribute a is better to split than attribute b, according to whole data stream. Moreover, if a and b are attributes with the highest and the second highest values of information gain, then with probability $1 - \delta$, a is the best attribute to split according to the whole stream.*

Proof. For this particular choice of attributes a and b the assumptions of Theorem 1 are satisfied. Therefore, inequality (30) holds and can be transformed to the form

$$P(E[f(Z)] \geq f(Z) - \epsilon) \geq 1 - \delta. \qquad (31)$$

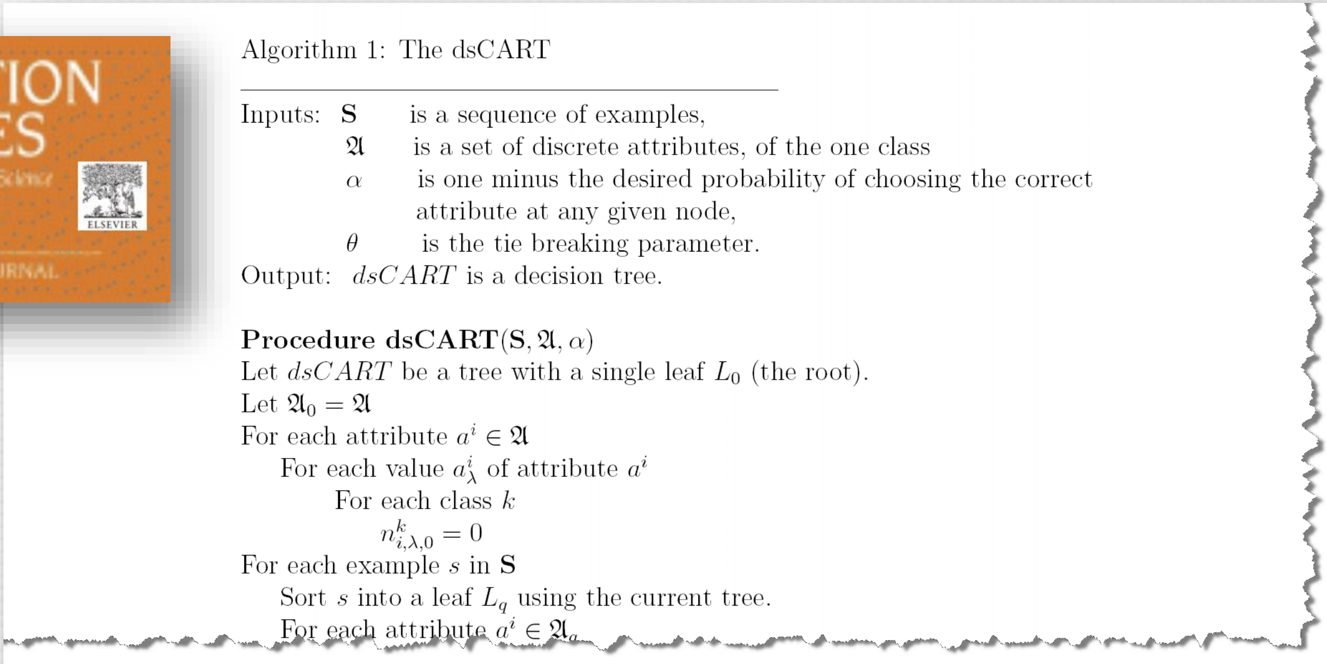
Notice that if $f(Z) > \epsilon$, then $E[f(Z)] > 0$ with probability $1 - \delta$. The inequality $E[f(Z)] > 0$ is equivalent to $E[Gain_a(Z)] > E[Gain_b(Z)]$ what means that the attribute a is better than attribute b to split, according to the whole data stream, with probability $1 - \delta$.

Let us consider now the case, where a and b are

References

Stream data mining algorithms (concept drift)

Rutkowski L. Jaworski M., Pietruczuk L., Duda P.,
The CART decision tree for mining data streams,
Information Sciences, vol. 266, pp. 1–15, 2014.



Algorithm 1: The dsCART

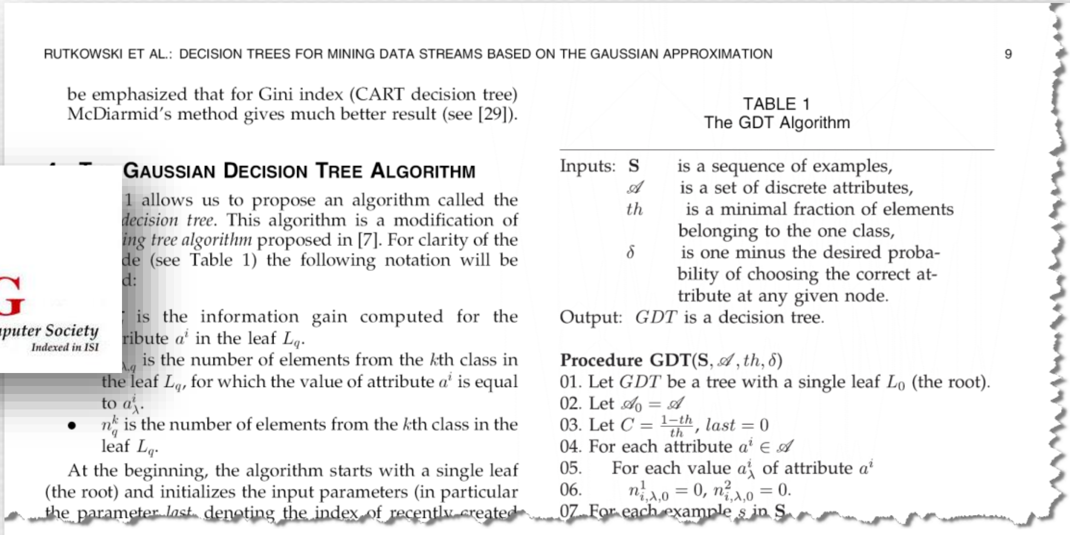
```
Inputs:  $\mathbf{S}$       is a sequence of examples,
         $\mathfrak{A}$       is a set of discrete attributes, of the one class
         $\alpha$       is one minus the desired probability of choosing the correct
                 attribute at any given node,
         $\theta$       is the tie breaking parameter.
Output:  $dsCART$ is a decision tree.

Procedure dsCART($\mathbf{S}, \mathfrak{A}, \alpha$)
Let $dsCART$ be a tree with a single leaf $L_0$ (the root).
Let $\mathfrak{A}_0 = \mathfrak{A}$
For each attribute $a^i \in \mathfrak{A}$
    For each value $a^i_\lambda$ of attribute $a^i$
        For each class $k$
            $n^k_{i,\lambda,0} = 0$
For each example $s$ in $\mathbf{S}$
    Sort $s$ into a leaf $L_q$ using the current tree.
    For each attribute $a^i \in \mathfrak{A}_q$
```

References

Stream data mining algorithms (concept drift)

Rutkowski L. Jaworski M., Pietruczuk L., Duda P., Decision Trees for Mining Data Streams Based on the Gaussian Approximation, IEEE Transactions on Knowledge and Data Engineering, vol. 26 , no. 1, pp. 108–119, 2014.

IEEE TRANSACTIONS ON
KNOWLEDGE AND DATA ENGINEERING
A publication of the IEEE Computer Society
Indexed in ISI



RUTKOWSKI ET AL.: DECISION TREES FOR MINING DATA STREAMS BASED ON THE GAUSSIAN APPROXIMATION 9

be emphasized that for Gini index (CART decision tree) McDiarmid's method gives much better result (see [29]).

GAUSSIAN DECISION TREE ALGORITHM

1 allows us to propose an algorithm called the *decision tree*. This algorithm is a modification of *ing tree algorithm* proposed in [7]. For clarity of the de (see Table 1) the following notation will be d:

is the information gain computed for the ribute a^i in the leaf L_q.

is the number of elements from the kth class in the leaf L_q, for which the value of attribute a^i is equal to a^i_λ.

- n^k_q is the number of elements from the kth class in the leaf L_q.

At the beginning, the algorithm starts with a single leaf (the root) and initializes the input parameters (in particular the parameter *last* denoting the index of recently created

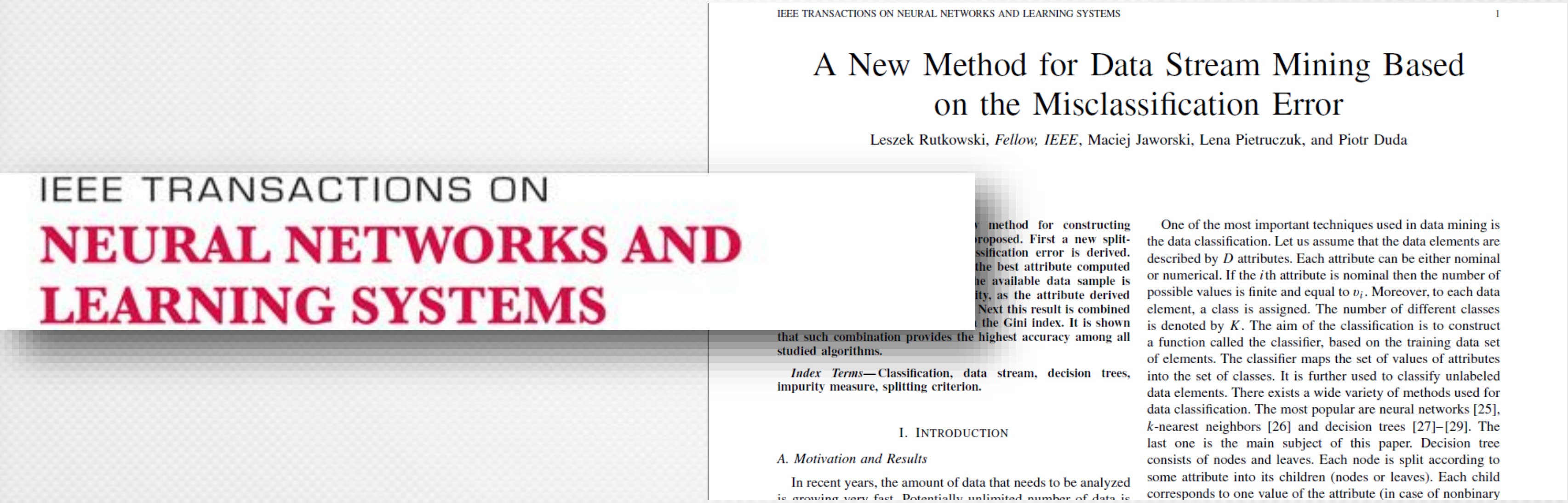
TABLE 1
The GDT Algorithm

```
Inputs:  S     is a sequence of examples,
         $\mathscr{A}$     is a set of discrete attributes,
         $th$    is a minimal fraction of elements
               belonging to the one class,
         $\delta$     is one minus the desired proba-
               bility of choosing the correct at-
               tribute at any given node.
Output:  $GDT$ is a decision tree.

Procedure GDT(S, $\mathscr{A}, th, \delta$)
01. Let $GDT$ be a tree with a single leaf $L_0$ (the root).
02. Let $\mathscr{A}_0 = \mathscr{A}$
03. Let $C = \frac{1-th}{th}$, $last = 0$
04. For each attribute $a^i \in \mathscr{A}$
05.     For each value $a^i_\lambda$ of attribute $a^i$
06.         $n^1_{i,\lambda,0} = 0$, $n^2_{i,\lambda,0} = 0$.
07. For each example $s$ in S
```

References

Stream data mining algorithms (concept drift)

Rutkowski L., Jaworski M., Pietruczuk L., Duda P.,
A new method for data stream mining based on the misclassification error, IEEE Transaction on Neural Networks and Learning Systems, vol.26, no. 5,
pp. 1048-1059, 2015



IEEE TRANSACTIONS ON
NEURAL NETWORKS AND LEARNING SYSTEMS

IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 1

A New Method for Data Stream Mining Based on the Misclassification Error

Leszek Rutkowski, *Fellow, IEEE*, Maciej Jaworski, Lena Pietruczuk, and Piotr Duda

method for constructing
roposed. First a new split-
ssification error is derived.
the best attribute computed
e available data sample is
ity, as the attribute derived
Next this result is combined
the Gini index. It is shown
that such combination provides the highest accuracy among all studied algorithms.

***Index Terms*—Classification, data stream, decision trees, impurity measure, splitting criterion.**

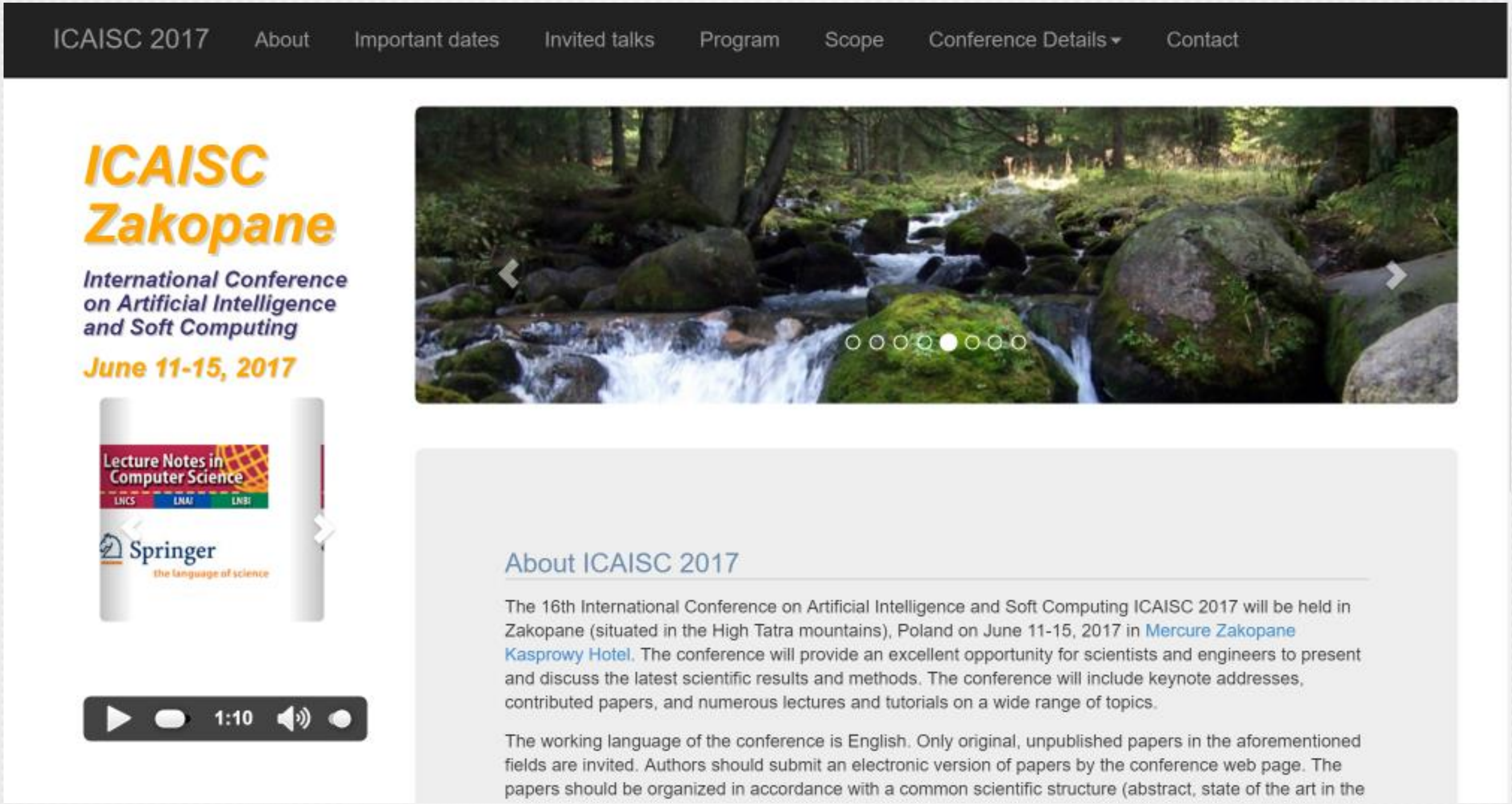
I. INTRODUCTION

A. Motivation and Results

In recent years, the amount of data that needs to be analyzed is growing very fast. Potentially unlimited number of data is

One of the most important techniques used in data mining is the data classification. Let us assume that the data elements are described by D attributes. Each attribute can be either nominal or numerical. If the ith attribute is nominal then the number of possible values is finite and equal to v_i. Moreover, to each data element, a class is assigned. The number of different classes is denoted by K. The aim of the classification is to construct a function called the classifier, based on the training data set of elements. The classifier maps the set of values of attributes into the set of classes. It is further used to classify unlabeled data elements. There exists a wide variety of methods used for data classification. The most popular are neural networks [25], k-nearest neighbors [26] and decision trees [27]–[29]. The last one is the main subject of this paper. Decision tree consists of nodes and leaves. Each node is split according to some attribute into its children (nodes or leaves). Each child corresponds to one value of the attribute (in case of nonbinary

The 16th International Conference on Artificial Intelligence and Soft Computing ICAISC 2017



About ICAISC 2017

The 16th International Conference on Artificial Intelligence and Soft Computing ICAISC 2017 will be held in Zakopane (situated in the High Tatra mountains), Poland on June 11-15, 2017 in Mercure Zakopane Kasprowy Hotel. The conference will provide an excellent opportunity for scientists and engineers to present and discuss the latest scientific results and methods. The conference will include keynote addresses, contributed papers, and numerous lectures and tutorials on a wide range of topics.

The working language of the conference is English. Only original, unpublished papers in the aforementioned fields are invited. Authors should submit an electronic version of papers by the conference web page. The papers should be organized in accordance with a common scientific structure (abstract, state of the art in the

Thank you for your attention