

# Incremental Decision Trees

Walter Gallego Gomez

# Outline

- Incremental learning
- ID5R and ITI algorithms
- VFDT algorithm
- HW implementation of VFDT
- VFDT with positive and unlabeled data

# Incremental learning

The background of the slide features an abstract geometric design. It consists of several overlapping triangles and polygons in various shades of blue and red. The colors range from light, airy blues to deep, saturated reds and dark blues. The shapes are layered, creating a sense of depth and movement. The overall composition is modern and minimalist, typical of a professional presentation slide.

# Incremental learning

- ▶ Training Samples arrive sequentially
- ▶ No distinct phases of learning and operation. The system be used for classification and later refined with new training samples.
- ▶ Often Real Time.
- ▶ When a new training sample arrives, is cheaper to update the system accordingly than to re-build from scratch.

# Incremental learning

Incremental learning is useful when:

- ▶ The application is intrinsically sequential.
- ▶ There is a huge volume of training data (stored for example in a hard disk) and it is not possible for the algorithm to evaluate it all at the same time, so it has to be presented sequentially.

# ID5R and ITI algorithms

## ID5R and ITI algorithms

**ID5R, proposed by Paul E. Utgoff in 1989**

Modification of the basic non incremental ID3 algorithm. Only works for two-classes applications.

Each time a new sample arrives, statistics on each node are updated, and the current tree is restructured if necessary.

**ITI, same author, 1994:** Improvements on ID5R, to handle numerical attributes, Multiple classes and missing values.

# ID5R and ITI algorithms

In Both algorithms:

- Example order doesn't affect the resulting tree. It is guaranteed that the tree is the same that would have been obtained using the ID3 algorithm.
- Needs to store all examples, in leaf nodes.  
In non-leaf nodes, statics are stored.
- Relatively slow because restructuring of the tree may be necessary when new samples arrive



# VFDT algorithm



# Very Fast Decision Tree learner

**VFDT proposed by Pedro Domingos and Geoff Hulten in 2000**

Inspect each training sample only once. There is no need to store the samples.

The only information kept in memory is the tree itself and some statics.

The algorithm can produce trees of the same quality as batch learned trees.

# Information Gain

To choose the best attribute, we search for the attribute  $X$ , with **higher information Gain**.

The attribute  $X$  splits the dataset  $S$  on subsets  $\{S_1, \dots, S_m\}$

$$\text{InfoGain}(X) = \text{Entropy}(S) - \text{Entropy}_X(S_1, \dots, S_m)$$

# Information Gain

To choose the best attribute, we search for the attribute  $X$ , with **higher information Gain**.

The attribute  $X$  splits the dataset  $S$  on subsets  $\{S_1, \dots, S_m\}$

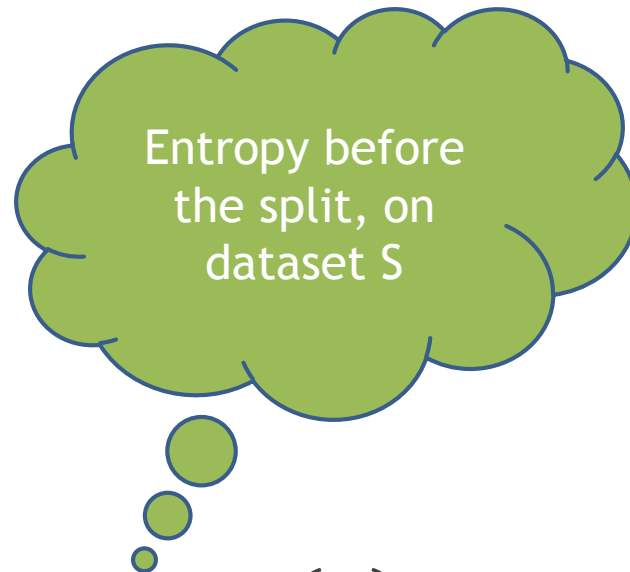


$$InfoGain(X) = Entropy(S) - Entropy_X(S_1, \dots, S_m)$$

# Information Gain

To choose the best attribute, we search for the attribute  $X$ , with **higher information Gain**.

The attribute  $X$  splits the dataset  $S$  on subsets  $\{S_1, \dots, S_m\}$

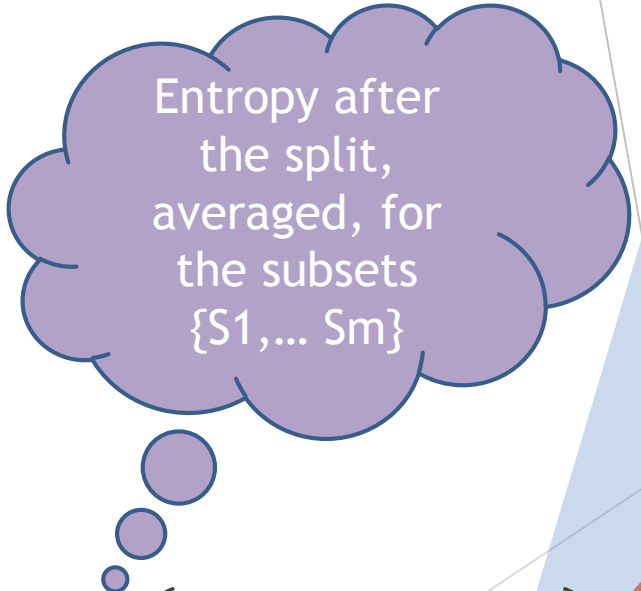


$$InfoGain(X) = Entropy(S) - Entropy_X(S_1, \dots, S_m)$$

# Information Gain

To choose the best attribute, we search for the attribute  $X$ , with **higher information Gain**.

The attribute  $X$  splits the dataset  $S$  on subsets  $\{S_1, \dots, S_m\}$



Entropy after  
the split,  
averaged, for  
the subsets  
 $\{S_1, \dots, S_m\}$

$$\text{InfoGain}(X) = \text{Entropy}(S) - \text{Entropy}_X(S_1, \dots, S_m)$$

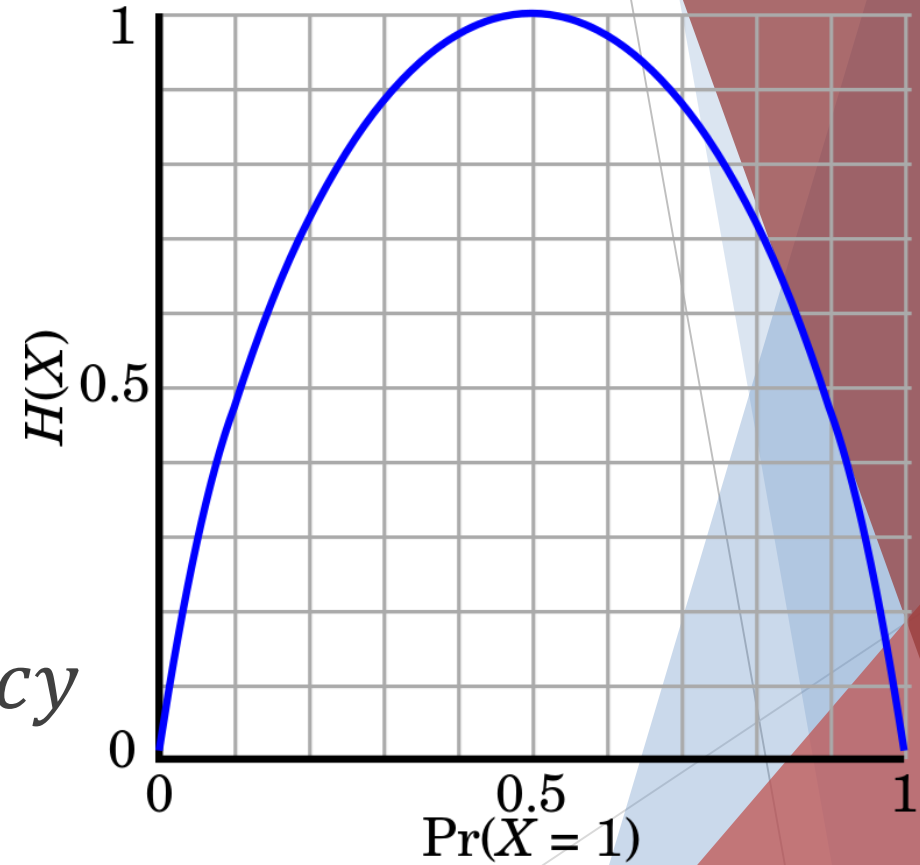
# Entropy

$$\text{InfoGain}(X) = \text{Entropy}(S) - \text{Entropy}_X(S_1, \dots, S_m)$$

$$\text{Entropy}(S) = - \sum_{i=1}^K P_i \log P_i$$

$k$ : Number of classes

$P_1, P_2, \dots, P_k$ : Classes relative frequency



# Averaged Entropy of subsets.

$$\text{InfoGain}(X) = \text{Entropy}(S) - \text{Entropy}_X(S_1, \dots, S_m)$$

$$\text{Entropy}_X(S_1, \dots, S_m) = - \sum_{i=1}^m \frac{n_i}{n} \text{Entropy}(S_i)$$

$n$ : Number of samples on  $S$ .



# Hoeffding bound trees

How can we find the attribute with the Highest information Gain for a node without examining all the training data?

If the algorithm is receiving samples sequentially, how many samples are required to evaluate the infoGain?

For each leaf-node, read samples sequentially, until I can choose the best attribute for a split with enough confidence. Future samples will be used in nodes downstream.

# Hoeffding bound trees

**Hoeffding bound:** With a probability(confidence)  $1 - \delta$ , the true mean of a random variable of range  $R$ , will not differ from the estimated mean after  $n$  independent observations for more than:

$$\epsilon = \sqrt{\frac{R^2 \ln \left( \frac{1}{\delta} \right)}{2n}}$$

# Hoeffding bound trees

**Hoeffding bound:** With a probability(confidence)  $1 - \delta$ , the true mean of a random variable of range  $R$ , will not differ from the estimated mean after  $n$  independent observations for more than:

$$\epsilon = \sqrt{\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2n}}$$



The larger  $n$ , the lower the error.

# Hoeffding bound trees

**Hoeffding bound:** With a probability(confidence)  $1 - \delta$ , the true mean of a random variable of range  $R$ , will not differ from the estimated mean after  $n$  independent observations for more than:

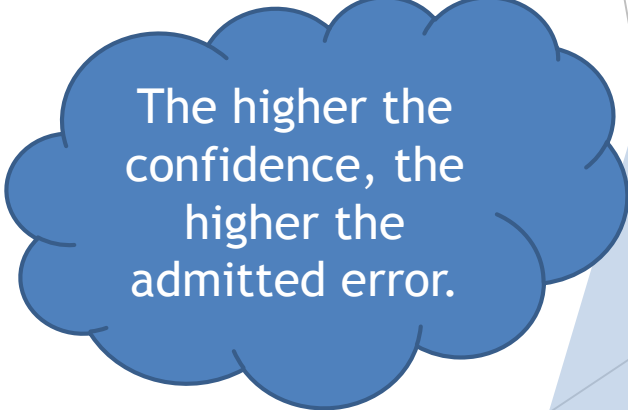
The larger the range, the larger the error.

$$\epsilon = \sqrt{\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2n}}$$

# Hoeffding bound trees

**Hoeffding bound:** With a probability(confidence)  $1 - \delta$ , the true mean of a random variable of range  $R$ , will not differ from the estimated mean after  $n$  independent observations for more than:

$$\epsilon = \sqrt{\frac{R^2 \ln \left( \frac{1}{\delta} \right)}{2n}}$$



The higher the confidence, the higher the admitted error.

# Hoeffding bound trees

The random variable being estimated is:

$$\Delta G = G(X_a) - G(X_b)$$

*G: Info Gain (or some other metric)*

*X<sub>a</sub>: Best attribute*

*X<sub>b</sub>: Second best attribute*

If we can estimate with confidence that  $\Delta G$  is positive, then  $X_a$  is indeed the best attribute.

# Hoeffding bound trees

To estimate, make  $n$  observations until:

$$\overline{\Delta G} = \overline{G(X_a)} - \overline{G(X_b)} > \epsilon$$

With this, we know that, with a probability of  $1 - \delta$ ,  $X_a$  is the best attribute.

For example, if  $\overline{\Delta G} = 0.3$  and  $\epsilon = 0.1$ , it means that  $G(X_a)$  is at least 0.2 larger than  $G(X_b)$

# Hoeffding bound trees

When a new training sample arrives, it traverses tree until it reaches a leaf node.

In the leaf, statistics about previously seen samples are stored.

If the samples do not belong to the same class:

Gain info for each attribute and the Hoeffding error are recomputed.

If the condition  $\overline{\Delta G} > \epsilon$  holds:

the leaf is transformed into a node, using  $X_a$  as attribute.



# VFDT Hardware implementation



# Acceleration of Data Streaming Classification Using Reconfigurable Technology - 2015

FPGA-based architecture for streaming data processing. The implemented architecture maps the VFDT algorithm, on a reconfigurable platform with very promising performance results.

- This is the first hardware-based architecture to build the VFDT data streaming classifier on a high-end FPGA platform.
- The architecture takes advantage of the fine-grained and the coarse-grained parallelism that reconfigurable hardware can offer.

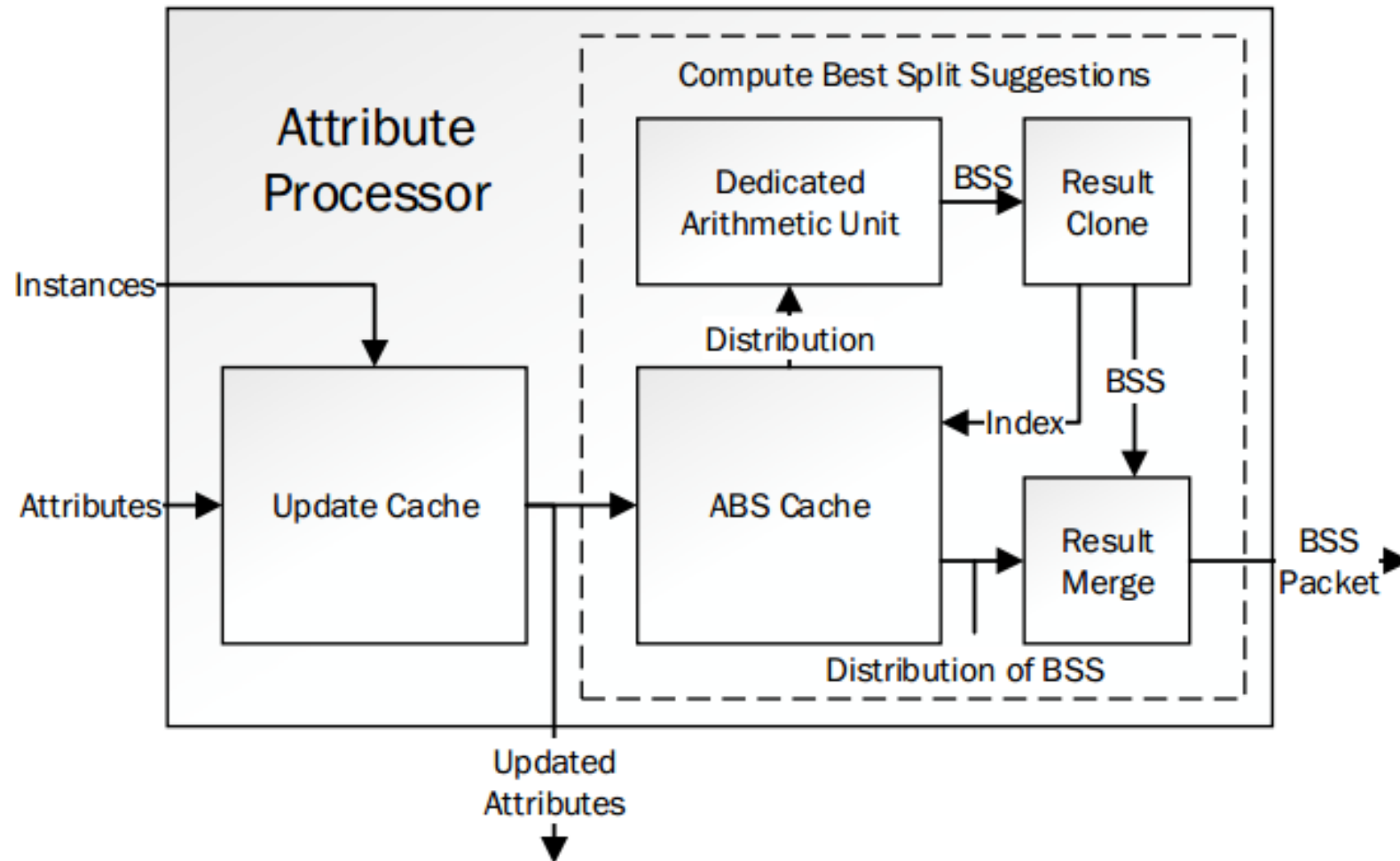
# Acceleration of Data Streaming Classification Using Reconfigurable Technology - 2015

The training process is a much more computationally intensive process than the classification one.

“The most time consuming part of the algorithm is the computation of the split suggestions at the leaves. Thus, we focused on accelerating the leaf operations by mapping these operations on reconfigurable technology.”

# Attribute processor

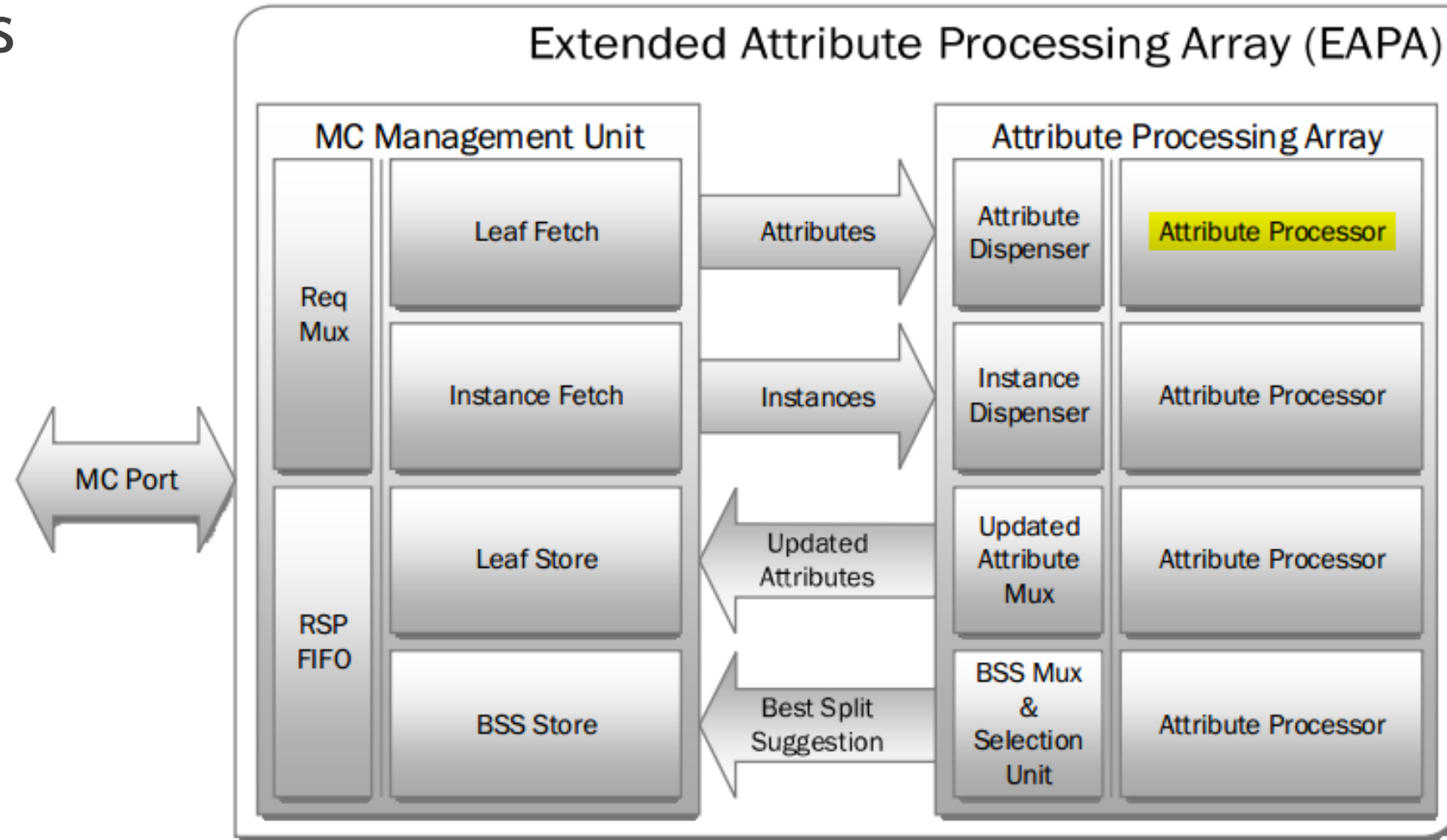
Updates the statics of a single node when a new instance arrives and calculates best split suggestions.



# Attribute processor

It consists of a series of parallel Attribute Processing Modules.

The DMA modules residing in MCMU feed APA with attributes and instances while the dispensers deliver the data to the Attribute Processors



# Acceleration of Data Streaming Classification Using Reconfigurable Technology - 2015

Some results reported by the authors:

<b>No. of attributes</b>	<b>MOA Hoeffding Tree Classification Execution Time (sec)</b>	<b>FPGA-based Hoeffding Tree Classification Execution Time (sec)</b>	<b>Speedup</b>
32	73.84	0.51	145x
64	147.89	0.96	154x
128	295.54	1.85	160x
256	590.95	3.63	163x
512	1182.55	7.19	164x
1024	2363.11	14.31	165x



VFDT with positive and unlabeled data

# Learning very fast decision tree from uncertain data streams with positive and unlabeled samples- 2012

Semi-supervised learning.

The idea is to modify the CVFDT (Concept adapting) algorithm, to make possible the learning from positive and unlabeled samples.

It works for applications with only two possible classes: “Positive” and “Negative”. Some of the training samples have been labeled as positive. The others are unlabeled.



## Learning very fast decision tree from uncertain data streams with positive and unlabeled samples- 2012

“ We argue that, in many practical situations, elements of the target concept may be abundant and cheap to collect. For instance, consider one diagnosis of diseases: in order to obtain an fully-labeled examples, it is necessary to systematically detect the disease on a representative sample of patients and this task may be quite expensive (or impossible). On the other hand, it may be easy to collect the medical files of patients who have the disease. Also, unlabeled data are any pool of patients possibly having the disease.”

## Learning very fast decision tree from uncertain data streams with positive and unlabeled samples- 2012

“A series of experiments on both synthetic and real-life dataset show that the proposed puuCVFDT algorithm has strong capabilities to learn from uncertain data streams with positive and unlabeled samples and tackle concept drift. Even when only 10% of the samples in the stream are positive, the classification accuracy of puuCVFDT is still very close to that of CVFDT”