Incremental Learning of Fuzzy Decision Trees for Streaming Data Classification

Riccardo Pecori[a], Pietro Ducange[b] and Francesco Marcelloni[b]
[a]SMARTEST Research Centre, eCampus University, Via Isimbardi 10, 22060, Novedrate (CO), Italy
[b]Department of Information Engineering, University of Pisa, Largo L. Lazzarino 1, 56122 Pisa, Italy
contact: riccardo.pecori@uniecampus.it, pietro.ducange@unipi.it, francesco.marcelloni@unipi.it

Abstract

Data stream analysis is growing in popularity in the last years since several application domains require to continuously and quickly analyse data produced by sensors with the aim of, for instance, reacting immediately when problems arise, or detecting new trends. The specificity of these domains imposes strict temporal constraints on machine learning algorithms to be used for mining useful insights. The Hoeffding Decision Tree (HDT) is a well-known classification algorithm for efficient streaming data classification. In this paper, with the aim of improving HDT accuracy and capability of handling noisy data, we exploit the learning procedure proposed in HDT for adapting a recently proposed fuzzy decision tree to cope with streaming data classification problems. We tested the fuzzy approach on a benchmark dataset for the on-line learning of data stream classification models. Results show that, during the on-line learning process, the fuzzy approach outperforms HDT in terms of accuracy.

Keywords: Streaming Data Classification, Hoeffding Decision Tree, Fuzzy Decision Tree, Evolving Classifiers

1 Introduction

Data stream analysis is gaining more and more attention thanks to the daily usage of a huge number of heterogeneous sources of streaming data such as sensors, wearables, smartphones and all other smart devices of the Internet of Things [17] in different application domains, including measurements in network monitoring and traffic management [6], log records and click-streams in web surfing [2] as well as in virtual learning environments [12], manufacturing processes [19], and continuous monitoring of twitter posts [4]. Moreover, data stream analysis is part of the Big Data and Analytics enabling technology for Industry 4.0 [13].

In the data stream context, data are collected typically at high rate, and algorithms processing them are required to work under very strict constraints. Consequently, data streams pose some important challenges for designing proper machine learning/data mining algorithms. First of all, due to the availability of limited resources with respect to the (practically infinite) amount of data to be handled, such algorithms should scan and inspect instances only once, trying to reduce the data to be stored and adopted for the training stage. Second, dealing with data whose nature and distribution generally changes over time, the concept drift should be appropriately managed. Third, the models, although continuously updated as the data arrive, should be capable to make a prediction at any time [16]. Three main interdependent metrics are usually concerned when designing data stream mining models, namely *accuracy*, *amount of memory* for training data storage and *time* required for parameter learning [1]. To concurrently optimize the values of these metrics during the learning phase is a very challenging task, since an increase in accuracy is generally achieved by increasing the amount of memory and time, and vice-versa.

The Hoeffding Decision Tree (HDT) [5], also known as "Very Fast Decision Tree", represents a reference model for streaming data classification. HDT adopts an on-line learning strategy, which grows a decision tree incrementally. The strategy considers to split a leaf of the tree whenever it contains a minimum number of instances. The actual number of instances for generating a split depends on the Hoeffding bound. This statistic ensures, with a certain level of confidence, that the attribute selected for the split is the same as the one that would have been chosen if an infinite number of instances had arrived. The main strengths of HDT are: i) only the tree and its necessary statistics are stored in memory and ii) the learned model is asymptotically nearly identical to the one built by a non-incremental learner (if the number of training instances is large enough).

In the last years, a number of strategies have been proposed for improving HDT. For example, in [7] the authors have discussed an approach for dealing with continuous data and to face the concept drift phenomenon. Furthermore, the work in [21] has presented a methodology for handling noisy instances and controlling the dimension of the tree. Recently, in [11], the Vertical HDT, which is a parallel and distributed version of the on line learning procedure of the HDT, has been introduced and experimented on big streaming data.

Some elements of fuzzy logic and fuzzy set theory have been exploited for enhancing classification models based on decision trees, both in batch and streaming contexts. In particular, vague and noisy data can be handled very efficiently whenever fuzzy sets are adopted for discretizing the domains of the attributes used in the classifiers [18]. An approach for learning binary and multi-way fuzzy decision trees (FDTs) in batch mode for big data classification has been recently discussed in [20]. Specifically, the authors exploit a fuzzy partitioning of the input attributes, carried out by using a novel distributed discretizer, and adopt the fuzzy information gain for selecting the attributes at the decision nodes.

The Flexible Fuzzy Decision Tree (FlexDT) proposed in [9] represents one of the first examples of FDT for streaming data classification. FlexDT is based on the classical incremental learning algorithm of HDT, but it employs binary fuzzy partitions for each attribute at a decision node. Sigmoidal fuzzy sets are adopted and appropriate heuristics for tuning their parameters during the learning stage have been defined. The classical formulations of the Hoeffding bound and of the information gain are used, respectively, for deciding whether to expand or not the tree and to select the best input attribute to be used for the splitting at each node. Recently, an extension of FlexDT, denoted as Multi FlexDT, has been introduced in [10]. In Multi FlexDT multi-way splits rather than binary splits are allowed at each decision node. In Multi FlexDT, when a new decision node is created, a binary fuzzy partition is adopted as in FlexDT. Then, the incremental learning algorithm allows creating a new binary partition for the attribute at a specific decision node, whenever a new training instance arrives and its membership degree to each fuzzy set is lower than a prefixed threshold. Thus, the previous partition at the node is extended considering the two new sigmoidal fuzzy sets and two new splittings may be considered. Both FlexDT and Multi FlexDT generate, incrementally, noise-robust models and also allow managing efficiently missing values. As stated in [10], Multi FlexDT outperforms FlexDT in terms of accuracy and depth of the trees. However, linguistic labels cannot be defined a-priori for each input attribute and a good level of integrity of the fuzzy partitions is not ensured. Thus, the interpretability of the fuzzy decision trees may be compromised.

In this paper, we propose a novel approach for incremental learning of multi-way FDTs in classification tasks. We exploit the main workflow of HDT for adapting the FDT proposed in [20]. Unlike in [20], for the sake of simplicity, we adopt uniform fuzzy partitions for each input attribute: the selection of the best input attribute to be used for the splitting at each node is performed by using the fuzzy information gain defined in [20]. We experimented the proposed incremental learning procedure for building FDTs on a benchmark dataset for on-line learning of classifiers for data streams [3]. We compared the proposed streaming FDT, denoted as SFDT in the following, with HDT. The results show that, in most of the time intervals, SFDT outperforms HDT in terms of accuracy. Moreover, the use of uniform fuzzy partitions allows us to generate simple, accurate and explainable linguistic decision trees.

The rest of the paper is structured as follows. In Section 2, we summarize some preliminaries about fuzzy partitioning and the structure of FDT. In Section 3 we describe the proposed incremental procedure for learning SFDTs. Section 4 shows the preliminary results and discusses some comparisons with HDT. Finally, Section 5 draws some conclusions.

2 Preliminaries

In this section, we briefly introduce some preliminary concepts regarding fuzzy partitions and the structure of FDT.

2.1 Fuzzy Partitions

Let $X = \{X_1, \dots, X_F\}$ be the set of numerical input attributes and X_{F+1} be the output of a fuzzy classification model. Let U_f, with $f = 1, ..., F$, be the universe of the f^{th} attribute X_f. Let $P_f = \{A_{f,1}, \dots, A_{f,j}, \dots, A_{f,T_f}\}$ be a partition of X_f consisting of T_f fuzzy sets $A_{f,j}$. The output X_{F+1} is a categorical variable assuming values in the set $\Gamma = \{C_1, \dots, C_k, \dots, C_K\}$ of K possible classes C_k.

In this work, we adopt strong and uniform triangular fuzzy partitions [8], as shown in Figure 1: the partitions are formed of three triangular fuzzy sets $A_{f,j}$, whose membership function is defined by the tuples ($a_{f,j}$,$b_{f,j}$,$c_{f,j}$), where $a_{f,j}$ and $c_{f,j}$ correspond to the left and right extremes of the support of $A_{f,j}$, and $b_{f,j}$ to its core.

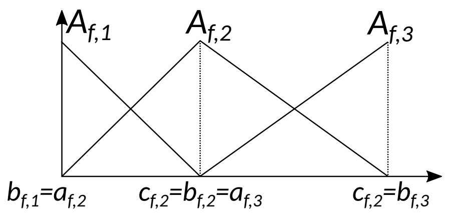

Figure 1: An example of a strong triangular fuzzy partition.

2.2 Fuzzy Decision Trees

A decision tree is a directed acyclic graph, where internal nodes represent a test on an attribute, branches denote the outcome of the test, and terminal nodes (leaves) contain instances belonging to one or more class labels. The topmost node is the root of the tree. In a general scenario, class $C_k \in \Gamma$ of each leaf is associated with a weight w_k, which determines the strength of class C_k in the leaf node.

Given a training set TS, the structure of a decision tree, in terms of nodes, branches and leaves, is generated recursively. First of all, one of the attributes is selected at the root, taking the overall TS into consideration. The attribute selection algorithm returns also a set of branches and corresponding nodes. A branch corresponds to a condition on the values of the attribute. For each node, a new attribute is selected from the set of the attributes, considering only the instances of the TS which satisfy the test associated with the branch. When no attribute can be selected, the node is considered as a leaf. The attribute selection algorithm is usually based on a specific metric, such as Gini Index or Information Gain [15].

In this paper, we exploit a multi-way FDT based on Fuzzy Information Gain [20]. First, each attribute is partitioned by using strong and uniform triangular fuzzy partitions. The recursive procedure for building the tree uses the Fuzzy Information Gain for the identification of the best splitting attribute. In multi-way FDTs, each test node produces exactly T_f branches, corresponding to all possible partitions of the attribute over which the test is being performed. Figure 2 shows an example of multi-way FDT, in which, for the sake of simplicity, we consider only two attributes: X_2 in the root and X_1 at the second level. Each attribute is partitioned into three fuzzy sets, and, as a consequence, produces exactly three output branches. In the example, each leaf is associated with just one class.

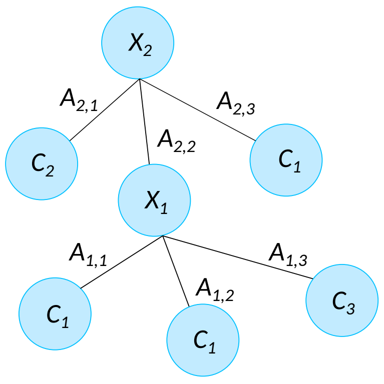

Figure 2: An example of multi-way FDT.

Once the tree has been generated, a given unlabeled instance $\hat{\mathbf{x}}$ can be assigned to a class $C_k \in \Gamma$ by following the activation path of nodes from the root to one or more leaves. In classical decision trees, each node represents a crisp set and each leaf is labeled with a unique class label. It follows that $\hat{\mathbf{x}}$ activates a unique path and is assigned to a unique class. In FDT, each node represents a fuzzy set. Thus, $\hat{\mathbf{x}}$ can activate multiple paths in the tree, reaching more than one leaf with different strengths of activation, called *matching degrees*. In this work, the chosen output class label for a certain unlabeled instance is obtained using the weighted vote approach, which considers the maximum total strength across all classes and the aforementioned weight w_k per class [20].

2.3 The Incremental Learning Procedure of HDT

In this Section, we briefly describe the iterative learning procedure of the classical HDT. In the data streaming scenario, in a certain time interval, we suppose that a stream of labeled training instances arrives. Thus, the iterative learning procedure is in charge of updating the current structure of the decision tree through two main stages: i) update of the statistics in the nodes and leaves, and ii) possible growth of the tree. We recall that the incremental learning procedure of HDT depends on a set of parameters including Grace Period (GP), Tie Threshold (TT), Split Confidence (usually denoted as δ) and Minimum Fraction (MF).

As regards the first stage, starting from the root, the procedure determines the leaf reached by the current instance and its statistics for the calculation of the information gain are updated. Then, a set of conditions are checked for deciding whether to split or not the current leaf (CL). First of all, the splitting is not allowed if the total number of instances in CL, denoted as Total Cardinality (TC_{CL}), is lower than GP. Otherwise, attributes are ranked considering the Information Gain and the Hoeffding Bound of the leaf (HB_{CL}), calculated as follows:

$$HB_{CL} = \sqrt{\frac{R^2 \ln(1/\delta)}{2TC_{CL}}}, \quad (1)$$

where R is the log_2 of the number of classes contained in CL. If the difference between the information gains of the first two attributes in the ranking is higher than HB_{CL} or HB_{CL} is lower than TT, then the splitting of CL is allowed by using the attribute with the highest information gain. The splitting is actually carried out if the percentage of instances in each post-split branch is higher than MF. We recall that the use of the Hoeffding Bound as a threshold ensures, with high probability, that the attribute chosen for the splitting using n samples (where n is the TC_{CL}) is the same that would be chosen using infinite samples [5].

3 The Proposed Incremental Learning Procedure for SFDTs

The incremental procedure for the SFDT learning, introduced in this work, resembles the one used in the standard HDT, except for the updating of the leaf statistics and the use of the Fuzzy Information Gain for selecting the best splitting attribute. Furthermore, the adopted classification strategy is the weighted vote approach as described in [20].

As regards the first stage, starting from the root, the procedure determines the leaves reached by the current instance. While in classical HDT only one leaf is reached by the instance, in SFDT an instance can reach more than one leaf. Indeed, an instance can activate more than one branch since an instance has different membership degree to the fuzzy sets corresponding to the possible values of the attribute used for splitting the node. In our case, due to the uniform partition we adopt, two branches are activated at each decision node. Thus, the instance can travel different paths with different membership degree and therefore reach different leaves. Once a leaf is reached, the *matching degree* of the path of the tree is calculated. Further, the *fuzzy cardinalities* of the fuzzy sets activated by the instance are updated. More details on the calculation of fuzzy membership degrees, matching degrees and fuzzy cardinality can be found in [20].

In the second stage, the procedure verifies whether the conditions to grow SFDT by splitting some of the current leaves CLs are satisfied. First of all, the *Total Fuzzy Cardinality* (TFC_{CL}) of each current leaf CL is calculated, summing up its single fuzzy cardinalities per class. If the TFC_{CL} is higher than GP, the procedure ranks the attributes calculating, for each attribute X_f, the Fuzzy Information Gain as in [20]. Furthermore, the *Fuzzy Hoeffding Bound* of the current leaf is computed as follows:

$$FHB_{CL} = \sqrt{\frac{R^2 \ln(1/\delta)}{2TFC_{CL}}}, \quad (2)$$

If the difference between the fuzzy information gains of the first two attributes in the ranking is higher than FHB_{CL} or FHB_{CL} is lower than TT, then the splitting of CL is allowed by employing the attribute with the highest fuzzy information gain. Also in this case, the splitting is actually carried out if the percentage of instances in each post-split branch is higher than MF. Notice that, since the maximum value of the fuzzy cardinality of an instance in CL is equal to 1, TC_{CL} is an upper bound for TFC_{CL}. Thus, FHB_{CL} is an upper bound for HB_{CL} and, consequently, the splitting condition checked for the incremental learning of the SFDT is more restrictive than the one checked for HDT.

4 Simulations and Results

In this section, first we describe the used dataset and the considered simulation scenario. Then, we show and discuss the results achieved by the proposed SFDT and compare them to the ones obtained by the classical HDT.

4.1 The Dataset used in the Experiments

In order to assess the effectiveness of the proposed approach for data stream classification, we consider a real world classification dataset, but simulating its evolution over time. We used the *Optical Recognition of Handwritten Digits* dataset, extracted from the UCI machine learning repository[1]. The dataset contains normalized bitmaps of hand-written digits belonging to 10 classes (digits ranging from zero to nine): 32×32 bitmaps are divided into non-overlapping blocks of 4×4 and the number of pixels are counted in each block. This generates an input matrix of 8×8 where each element is an integer in the range [0..16]. As a consequence, each image is an instance featuring 64 numerical attributes with values in [0..16], and one class attribute in [0..9]. In the experiments, we have normalized the values of each attribute from the range [0..16] to the range [0..1], thus the universe U_f of each input attribute X_f ranges from zero to one. The complete dataset includes a total of 5620 samples, but we considered only the first 5600 ones as described in the following.

[1]https://archive.ics.uci.edu/ml/datasets/optical+recognition+of+handwritten+digits

4.2 Simulation Scenario

In order to simulate a data stream, we divided the aforementioned dataset into a fixed number of K data chunks and analysed the chunks in sequence. We simulated the data stream by evaluating the performance, on the data chunk at the current time interval T_k, with $k = 2..K$, of the current classification model, trained considering the stream of instances included into the data chunks arrived until the time interval T_{k-1}. During the first time interval T_1, the classification model is just built using the first chunk of the dataset. In the experiments, we set $K = 50$. Moreover, the class distribution was preserved in the chunks, thus resulting into balanced training and test sets.

For the simulation tests we employed a machine featuring an $x86\_64$ architecture, 4 cores, Intel(R) Core(TM) $i7-2600$ CPU @ $3.40GHz$ and $16GB$ RAM. The proposed incremental procedure for learning SFDTs was developed in JAVA and integrated into the WEKA Toolkit for machine learning[2], also exploiting the Weka implementation of the classical HDT.

Achieving optimal performance from a stream classifier is a challenging task and a fixed set of values of critical parameters depends on the particular considered dataset [14]. Therefore, we have carried out an intensive simulation campaign to optimize the main parameters of the incremental learning procedures of both SFDT and HDT. In Table 1, we summarize the values of the main aforementioned parameters for both classical HDT and the proposed SFDT. These values have been obtained to concurrently increase the accuracy and reduce the tree depth. As regards the SFDT, we achieved the best results, with the selected dataset, using a fuzzy partition of three triangular fuzzy sets for each input attribute.

The parameters influencing the growth of both trees and their classification accuracy, are the following:

- *Split Confidence*, involved in the computation of both fuzzy and classical Hoeffding Bound, and determining the closeness of performance statistics to those of a decision tree generated in batch.
- *Tie Threshold* specifying a threshold to break the ties forcibly whenever the difference of information gains, between the first two attributes with the highest gains, is too small. Indeed, in this latter case, too much time may be required to reach the splitting event.
- *Grace Period* determining, in the case of the classical HDT, the minimum number of instances (*total crisp cardinality*) that have to be contained by a node before being considered for splitting. In the case of SFDT, this threshold is compared, as discussed in Section 3, with the value of the total fuzzy cardinality.
- *Minimum Fraction* determining the minimum percentage of instances in a branch after a possible split.

[2]http://www.cs.waikato.ac.nz/ml/weka

Parameter	HDT	SFDT
Split Confidence	10^{-7}	10^{-7}
Tie Threshold	2.5	2.5
Grace Period	30	25
Minimum Fraction	0.01	0.01

Table 1: Main parameters for both HDT and SFDT.

4.3 Experimental Results

In this section, we show some preliminary results achieved by the proposed SFDT. Moreover, we compare SFDT with the classical HDT. To this aim, the following metrics are adopted for evaluating the two models: the *total accuracy* over all classes, the *tree depth*, computed as the maximum number of levels reached by the tree after the training phase, and the *total number of leaves*. For each of the two decision trees, these metrics have been extracted in correspondence of time interval T_k.

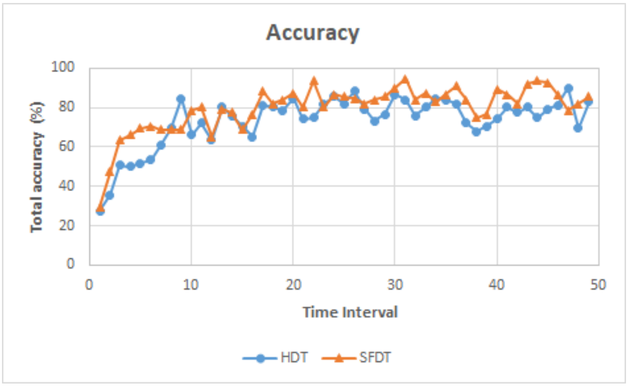

Figure 3: Trends of the total accuracy over all 10 classes for both HDT and SFDT.

Figure 3 shows the trends, along the different time intervals, of the total accuracy. It is worth noticing that, in most of the time intervals, the accuracy achieved by SFDT is higher than the one achieved by HDT.

Figure 4 regards the depth of the trees, which is related to the tree complexity. By analyzing the trends over the time intervals, the SFDT grows up, in terms of levels, slower than HDT. Since the *GP* for SFDT has been set to a lower value than the one set for HDT, this

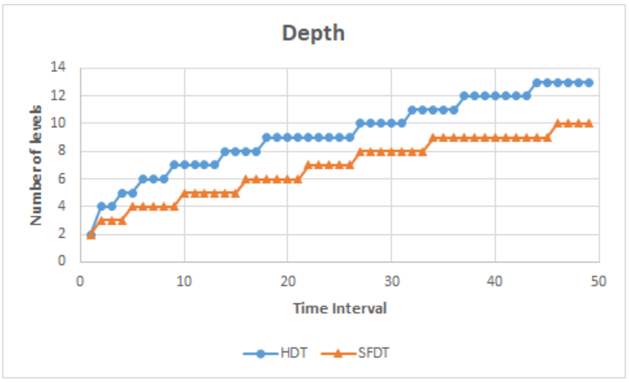

Figure 4: Trends of the tree depth for both HDT and SFDT.

result may appear quite strange. Indeed, in general, the lower the value of GP the higher the probability of allowing a split in a leaf and, thus, of generating deep trees. However, we have to consider that the value of the fuzzy cardinality is mostly lower than the classical cardinality. For this reason, if we set the same value for the GP for both HDT and SFDT, a split in a leaf of an SFDT will be allowed with a lower probability than in HDT. In conclusion, as shown by the experimental results, setting the values of GP as in Table 1 does not ensure that HDT will growth slower than SFDT.

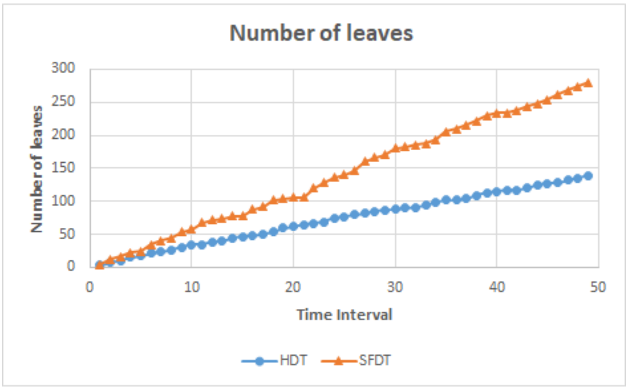

Figure 5: Trends of the number of leaves for both HDT and SFDT.

Finally, Figure 5 shows the tree growth trends in terms of number of leaves. Also this metric is related to the tree complexity. The reader can observe that the number of leaves in SFDT increases faster than in HDT. This phenomenon is due to the fact that each split in the SFDT can generate up to three leaves, while in the classical HDT just binary splits are allowed. However, the order of magnitude of the number of total leaves for both SFDT and HDT is the same.

For the sake of completeness, in Table 2 we show some of the obtained results for both SFDT and HDT in terms of accuracy, tree depth and number of leaves. The results are sampled at different time intervals, from T_5 to T_{50}, with a step of five units. We highlighted in bold the best values of the accuracy. It is worth noticing that, in most of the cases, SFDT outperforms HDT in terms of accuracy. On the other hand, as discussed above, the classical HDT is always less complex, in terms of number of leaves, than SFDT. However, the number of levels associated with SFDT is always lower than the one of HDT.

In the following, we show some examples of the obtained fuzzy rules, extracted from the SFDT generated at T_{45}, which is the one achieving the highest overall accuracy. We extracted three different paths from the SFDT, which generated three rules characterized by different levels of complexity, in terms of number of conditions in the antecedent. In the following, we show a long rule (with 8 conditions), a medium rule (with 7 conditions) and a short rule (with 5 conditions). We labeled the fuzzy sets, adopted in the partitioning of the input attributes, as “Low”, “Medium”, and “High” (L, M, and H, respectively). As a consequent of the rules, we considered the class with the highest weight in the specific leaf. In the antecedent, $atti$ represents the attribute describing the number of pixels contained in the i^{th} block of the image.

$$\begin{aligned} R_{Long}: \textbf{IF}\ & att43\ \textbf{is}\ L\ \textbf{AND} & att19\ \textbf{is}\ H\ \textbf{AND} \\ & att22\ \textbf{is}\ L\ \textbf{AND} & att6\ \textbf{is}\ H\ \textbf{AND} \\ & att11\ \textbf{is}\ H\ \textbf{AND} & att21\ \textbf{is}\ L\ \textbf{AND} \\ & att27\ \textbf{is}\ H\ \textbf{AND} & att23\ \textbf{is}\ H \\ & \textbf{THEN}\ Digit\ \textbf{is}\ 7 & \end{aligned}$$

$$\begin{aligned} R_{Medium}: \textbf{IF}\ & att43\ \textbf{is}\ L\ \textbf{AND} & att11\ \textbf{is}\ M\ \textbf{AND} \\ & att44\ \textbf{is}\ H\ \textbf{AND} & att22\ \textbf{is}\ L\ \textbf{AND} \\ & att21\ \textbf{is}\ L\ \textbf{AND} & att62\ \textbf{is}\ H\ \textbf{AND} \\ & att30\ \textbf{is}\ L & \textbf{THEN}\ Digit\ \textbf{is}\ 6 \end{aligned}$$

$$\begin{aligned} R_{Short}: \textbf{IF}\ & att43\ \textbf{is}\ L\ \textbf{AND} & att29\ \textbf{is}\ M\ \textbf{AND} \\ & att38\ \textbf{is}\ L\ \textbf{AND} & att27\ \textbf{is}\ L\ \textbf{AND} \\ & att45\ \textbf{is}\ L & \textbf{THEN}\ Digit\ \textbf{is}\ 8 \end{aligned}$$

As the reader can appreciate, the linguistic rules shown above are characterized by a very high level of interpretability.

	HDT			SFDT		
Time Interval	**Accuracy**	**Depth**	**Leaves**	**Accuracy**	**Depth**	**Leaves**
T_{50}	83.04%	13	139	**85.71%**	10	280
T_{45}	75%	13	125	**93.75%**	9	248
T_{40}	70.54%	12	113	**76.78%**	9	230
T_{35}	**84.82%**	11	99	83.04%	9	194
T_{30}	76.78%	10	87	**85.71%**	8	170
T_{25}	85.71%	9	75	**86.61%**	7	136
T_{20}	78.57%	9	60	**83.93%**	6	104
T_{15}	75.89%	8	44	**77.68%**	5	5
T_{10}	**84.82%**	7	30	68.75%	4	4
T_5	50%	5	15	**66.07%**	3	3

Table 2: Accuracy, tree depth and number of leaves extracted at different time intervals.

5 Conclusions

In this paper, we have discussed a novel procedure for incremental learning of a fuzzy decision tree from streaming data. We have integrated some elements of the fuzzy logic and fuzzy set theory into the classical incremental learning procedure of the Hoeffding Decision Tree (HDT). Specifically, first, we have discretized each input attribute using a fuzzy strong uniform partition. Then, we have defined the fuzzy Hoeffding bound for controlling the growth of the tree while streams of instances arrive. Finally, we have adopted the fuzzy information gain as a metric for selecting the best input attribute for the splitting nodes. We have experimented the proposed incremental learning procedure on a benchmark dataset for data stream classification. We have compared the achieved results with the ones achieved by the classical HDT. The results have shown that, even though the fuzzy decision tree is a bit more complex than HDT, the accuracies achieved by HDT are, in most of the cases, lower than the ones obtained by the fuzzy approach. Moreover, the rules that may be extracted from the fuzzy decision tree are characterized by a high level of interpretability.

Acknowledgement

This work was partially supported by Tuscany Region, in the context of the projects Talent and Sibilla in the framework of regional program “FESR 2014-2020”, and by the Italian Ministry of Education and Research (MIUR), in the framework of the CrossLab project (Departments of Excellence).

References

[1] A. Bifet, Mining big data in real time, Informatica (Slovenia) 37 (1) (2013) 15–20.

[2] K. K. Bina Bhandari, R. H. Goudar, Quine-mccluskey: A novel concept for mining the frequency patterns from web data, International Journal of Education and Management Engineering 8 (1) (2018) 40–47.

[3] G. Casalino, G. Castellano, A. M. Fanelli, C. Mencar, Enhancing the dissfcm algorithm for data stream classification, in: International Workshop on Fuzzy Logic and Applications, Springer, 2018, pp. 109–122.

[4] E. D’Andrea, P. Ducange, A. Bechini, A. Renda, F. Marcelloni, Monitoring the public opinion about the vaccination topic from tweets analysis, Expert Systems with Applications 116 (2019) 209–226.

[5] P. Domingos, G. Hulten, Mining high-speed data streams, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD’00, 2000, pp. 71–80.

[6] P. Ducange, G. Mannarà, F. Marcelloni, R. Pecori, M. Vecchio, A novel approach for internet traffic classification based on multi-objective evolutionary fuzzy classifiers, in: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2017, pp. 1–6.

[7] J. Gama, R. Fernandes, R. Rocha, Decision trees for mining data streams, Intell. Data Anal. 10 (1) (2006) 23–45.

[8] S. Guillaume, B. Charnomordic, Fuzzy inference systems: An integrated modeling environment for collaboration between expert knowledge and data using fispro, Expert Systems with Applications 39 (10) (2012) 8744–8755.

[9] S. Hashemi, Y. Yang, Flexible decision tree for data stream classification in the presence of concept change, noise and missing values, Data Mining and Knowledge Discovery 19 (1) (2009) 95–131.

[10] A. Isazadeh, F. Mahan, W. Pedrycz, Mflexdt: multi flexible fuzzy decision tree for data stream classification, Soft Computing 20 (9) (2016) 3719–3733.

[11] N. Kourtellis, G. De Francisci Morales, A. Bifet, A. Murdopo, VHT: Vertical Hoeffding Tree, in: 2016 IEEE International Conference on Big Data (Big Data), 2016, pp. 915–922.

[12] R. Pecori, A virtual learning architecture enhanced by fog computing and big data streams, Future Internet 10 (1) (2018) 4.

[13] R. S. Peres, A. D. Rocha, P. Leitao, J. Barata, Idarts–towards intelligent data analysis and real-time supervision for industry 4.0, Computers in Industry 101 (2018) 138–146.

[14] B. R. Prasad, S. Agarwal, Critical parameter analysis of Vertical Hoeffding Tree for optimized performance using SAMOA, International Journal of Machine Learning and Cybernetics 8 (4) (2017) 1389–1402.

[15] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106.

[16] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, F. Herrera, A survey on data preprocessing for data stream mining: Current status and future directions, Neurocomputing 239 (2017) 39 – 57.

[17] R. Ranjan, O. Rana, S. Nepal, M. Yousif, P. James, Z. Wen, S. Barr, P. Watson, P. P. Jayaraman, D. Georgakopoulos, M. Villari, M. Fazio, S. Garg, R. Buyya, L. Wang, A. Y. Zomaya, S. Dustdar, The next grand challenges: Integrating the internet of things and data science, IEEE Cloud Computing 5 (3) (2018) 12–26.

[18] M. Ricatto, M. Barsacchi, A. Bechini, Interpretable cnv-based tumour classification using fuzzy rule based classifiers, in: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, ACM, 2018, pp. 54–59.

[19] O. Savković, E. Kharlamov, M. Ringsquandl, G. Xiao, G. Mehdi, E. G. Kalayc, W. Nutt, I. Horrocks, Semantic diagnostics of smart factories, in: R. Ichise, F. Lecue, T. Kawamura, D. Zhao, S. Muggleton, K. Kozaki (Eds.), Semantic Technology, Springer International Publishing, Cham, 2018, pp. 277–294.

[20] A. Segatori, A. Bechini, P. Ducange, F. Marcelloni, A distributed fuzzy associative classifier for big data, IEEE Transactions on Cybernetics 48 (9) (2018) 2656–2669.

[21] H. Yang, S. Fong, Moderated VFDT in stream mining using adaptive tie threshold and incremental pruning, in: Data Warehousing and Knowledge Discovery, Springer Berlin Heidelberg, 2011, pp. 471–483.