

Scalable decision tree based on fuzzy partitioning and an incremental approach

Somayeh Lotfi¹, Mohammad Ghasemzadeh², Mehran Mohsenzadeh¹, Mitra Mirzarezaee¹

¹Computer Engineering Department, Science and Research Branch, Islamic Azad University, Tehran, Iran

²Computer Engineering Department, Yazd University, Yazd, Iran

Article Info

Article history:

Received Jul 18, 2021

Revised Dec 18, 2021

Accepted Jan 5, 2022

Keywords:

Discretization

Fuzzy decision tree

Fuzzy partitioning

Incremental method

ABSTRACT

Classification as a data mining material is the process of assigning entities to an already defined class by examining the features. The most significant feature of a decision tree as a classification method is its ability to data recursive partitioning. To choose the best attributes for partition, the value range of each continuous attribute should be divided into two or more intervals. Fuzzy partitioning can be used to reduce noise sensitivity and increase the stability of trees. Also, decision trees constructed with existing approaches, tend to be complex, and consequently are difficult to use in practical applications. In this article, a fuzzy decision tree has been introduced that tackles the problem of tree complexity and memory limitation by incrementally inserting data sets into the tree. Membership functions are generated automatically. Then fuzzy information gain is used as a fast-splitting attribute selection criterion and the expansion of a leaf is done attending only with the instances stored in it. The efficiency of this algorithm is examined in terms of accuracy and tree complexity. The results show that the proposed algorithm by reducing the complexity of the tree can overcome the memory limitation and make a balance between accuracy and complexity.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mohammad Ghasemzadeh

Department of Computer Engineering, Yazd University

Yazd, Iran

Email: m.ghasemzadeh@yazd.ac.ir

1. INTRODUCTION

Data mining is a motivating field of research in several disciplines including artificial intelligence, databases, statistics, visualization and high performance and parallel computing. The knowledge obtained by data processing is used for several applications including customer retention, marketing research, science, exploration and fraud detection to production control, health system, education, security assessment, and road traffic prediction and control [1]–[4]. The classification in data mining may be described as a supervised manner that given a training dataset with associated training labels, determine the suitable class labels for an unlabeled test instance. The decision tree (DT) may be a popular classification method that constructs a classification model within the form of a tree structure [5]. DT is a structure similar to a flowchart, in which each internal node represents a test of an attribute, each branch represents the result of the attribute test, and each leaf node represents a class label. The lead of the root node is zero, which means it has no leading edge. The tree implements classification by dividing the branches of the tree, where each division represents a test of the data attributes. This branch splitting continues to the last level, called the terminal level, where all the data tuples in a node involve samples of one class [6]. There are a variety of statistical algorithms that can be used to build decision trees, including ID3, classification and regression trees (CART), C4.5, chi-squared automatic interaction detection (CHAID), and quick, unbiased, efficient, statistical tree (QUEST) [7], [8].

One of the most challenging problems in the decision tree is developing scalable algorithms that can process large data sets whose size exceeds the memory capacity [9]. The size of decision trees tends to be dependent on the size of training data, and conventional decision tree building approaches are inefficient. Several algorithms have been developed to construct DTs from large data sets. Sampling, partitioning, distributed, parallel processing, and incremental methods are some of the basic techniques for processing large data [9]. For example, supervised learning in quest (SLIQ) [10], RainForest [11], classification for large or out-of-core datasets (CLOUDS) [12], bootstrapped optimistic algorithm for tree (BOAT) [13], very fast decision tree (VFDT) [14], decision tree using fast splitting attribute selection (DTFS) [15] trying to solve the same problems. Some of these algorithms store the entire data set in memory, while others only store part of the training data, but selecting a subset of the data is time-consuming and computationally expensive. Besides, in these methods, the reliability of the model is reduced due to the loss of part of the data. Some of these use lists store a set of data in the main memory. These methods assign a list to each attribute in the dataset. The problem is that some of these lists require more space than the one required to store the whole training set.

Another problem with tree-building algorithms is dealing with numerical features. The most common method is to partition each feature into two or more intervals using all values of the attribute through multiple cutting points. Therefore, the numerical features are divided into intervals and the discrete intervals behave like categorical values [16]. The discretization results are the generation of crisp intervals so that a feature value either belongs to an interval or not. One of the important disadvantages of the sharp cutting point is the sensitivity of the decision tree to noisy data. A solution to this problem is the use of soft discretization based on fuzzy theory [17]. The literature proposes some techniques to select partition attributes [18], [19], however, these techniques do not intend to deal with large data sets, because some of them must evaluate a large number of candidate partitions to select the best attributes, others use discretization methods to process numerical attributes, and some use expensive techniques to expand nodes.

To deal with the mentioned problems in the decision tree, the present study aims to present an incremental algorithm based on fuzzy partitioning. By entering data into the tree incrementally, there is no need to store the entire dataset in the main memory. The partition strategy involves dividing the training set based on all possible attribute values of the discrete attributes, resulting in a partition for each possible value of the selected attribute. For continuous attributes, we need a discretization step. In this paper, our focus is on a discretization of continuous values based on a fuzzy approach. With local discretization on a dataset of each node, the numerical value is converted to fuzzy values. Besides, in each node, the attribute with the highest probability prediction is selected to create a new branch. By building the decision tree using the entire training dataset without the need to store the entire data in memory and eliminate the used records after the development of each node, the memory loss is prevented, and the reliability of the model is increased.

2. THE COMPREHENSIVE THE ORETICAL BASIS

When there are continuous and nominal attributes within the data set, most rule induction techniques discretize the continual attributes into intervals and treat the discretized intervals like to the par value within the induction process [20]. The aim of attribute discretization is to search out concise data representations as categories that are sufficient for the training task to retain the maximum amount information as possible within the original continuous attributes. The foremost common method is to partition each feature into two or more intervals using all values of the attribute through multiple cutting points [16]. There will be a DA discretization scheme in the continuous attribute A, which divides this attribute into k discrete and disjoint intervals $\{[d_0, d_1], [d_1, d_2], \dots, [d_{k-1}, d_k]\}$, where d_0 and d_k are the minimum and maximum values respectively, and $P_A = \{d_1, d_2, \dots, d_{k-1}\}$ represents the set of cut points of A, arranged in ascending order. After evaluating cut points, intervals of continuous values are splitting according to some defined criterion. In the clear case, the discretization will result in a crisp range, and the attribute value either belongs to a certain range or does not belong to a certain range. Therefore, the discrete interval should be interpreted loosely. The intuitive way to obtain the fuzzy interval for each continuous attribute is to discretize its domain into several clear intervals [21]. The most important issue in fuzzy logic is the definition of number, type of parameters, and membership function. We use triangular fuzzy partitions, as shown in Figure 1.

In a crisp decision tree, the split point is chosen as the midpoint of the continual attribute values where the knowledge within the class changes. As long as the division attribute with a specific value is selected, there is no guarantee that this is the exact value that should be divided. There is always some fuzziness when choosing the value of the split point.

Assume that X is a global set of x variables; the fuzzy set of S on X is defined by membership functions as $\mu_s(x): X \rightarrow [0,1]$, which indicates the membership degree of x to S fuzzy set [22]. i) The set of attributes of a dataset is denoted by $X = \{X_1, \dots, X_k, Y\}$, where X_i is the i^{th} attribute, K is the number of input

attributes, Y be the output of a fuzzy classification model; ii) 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 $A_{f,j}$; iii) The output Y is a categorical variable assuming values in the set $Y = \{C_1, \dots, C_K\}$ of K possible classes C_k , iv) The examples in the fuzzy dataset S are denoted by $S = \{(X_1, \mu_S(X_1)), (X_2, \mu_S(X_2)), \dots, (X_n, \mu_S(X_n))\}$, where X_i is the i^{th} example, $\mu_S(X_i)$ is the membership degree of X_i in S , and n is the number of examples, and v) $\{F_i^{(1)}, F_i^{(2)}, \dots, F_i^{(r_i)}\}$ shows the fuzzy terms defined on the i^{th} attribute, where $F_i^{(j)}$ is the j^{th} fuzzy term defined on attribute X_i .

Fuzzy decision trees (FDT) combine decision trees with approximate reasoning offered by fuzzy representation to cater to uncertainties. FDT use fuzzy linguistic terms to specify the branching condition of nodes and permit examples to simultaneously follow down multiple branches with different satisfaction degrees ranged on $[0, 1]$. Each edge of FDT is annotated with a condition, and each leaf is annotated with a fuzzy set. During this paper, we exploit an FDT based on fuzzy information gain [16]. 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 [23] as (1) to (3):

$$\text{Fuzzy Information Gain}(S, A) = \text{Entropy}_f(s) - \sum_{v \in A} \left(\frac{|S_v|}{|S|} \right) \text{Entropy}_f(S_v) \tag{1}$$

$$\text{Entropy}_f(s) = - \sum_{i=1}^c \sum_{j=1}^n \frac{\mu_{ij}}{\sum_{k=1}^c \sum_{i=1}^n \mu_{ik}} \log_2 \sum_{j=1}^n \frac{\mu_{ij}}{\sum_{k=1}^c \sum_{i=1}^n \mu_{ik}} \tag{2}$$

$$\frac{|S_v|}{|S|} = \frac{\sum_{i=1}^n \mu_{iv}}{\sum_{k=1}^c \sum_{j=1}^n \mu_{kj}} \tag{3}$$

where N represents the number of samples, μ_{iv} is the membership degree of special value for i^{th} feature, and C_t is the number of fuzzy sets for the attribute in question.

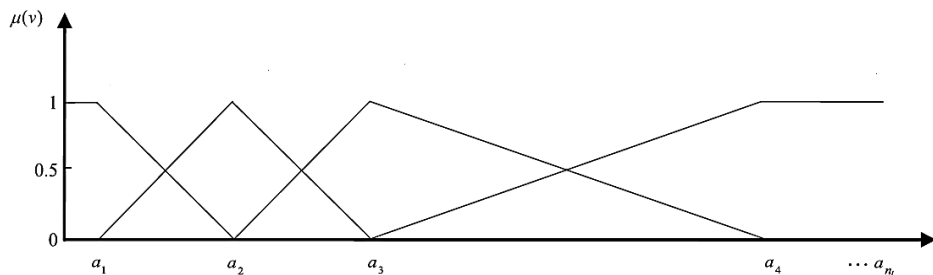


Figure 1. The triangular fuzzy partition

3. METHOD

This paper proposes an algorithm called IFDT for the development a decision tree supported the idea of incremental construction, while the training data in every node of the tree is processed incremental way. IFDT is that the fuzzy version of the IDT algorithm, that the continuous values are converted to fuzzy values, the attribute with the best classification ability is chosen for branching using evaluation criteria, and also the children are branched from the attribute. The IFDT algorithm is split into four stages and that they are described within the following subsections.

3.1. First stage

The first stage creates the root node of the decision tree, and the training object defines the input of this node, one by one, until the given conditions are met. N is the maximum number of objects that the root node can accept at any time. After reaching this value, the node needs to be developed. To develop the node, a feature with homogeneous partitioning ability on data should be chosen. For this purpose, training records are converted to fuzzy values using defined membership functions.

3.2. Second stage

In this stage, we want to find the fuzzy sets for j^{th} quantitative attribute. The range of considered attributes is from min to max. The set of $\{a_{1j}, a_{2j}, \dots, a_{kj}\}$ shows the median fuzzy points of j^{th} attribute. The provided methods to generate the membership functions act independently of sample distribution or based on

the distribution of samples. In the present study, the membership function is automatically generated using the presented method in [24]. The considered membership function is calculated as follow. First, the initial cut-off points are generated using crisp discretization. The points can be used to create a set of distinct intervals that are described using the mean of the crisp membership functions. If the value of the attribute is within the relevant interval, the value of 1 is assigned to the membership function and otherwise, the membership values are equal to 0. In the described overlapping intervals by fuzzy membership functions, a point near the cut-off point is assigned to two fuzzy sets with the membership degrees of less than 1 and greater than 0 for both membership functions. The sum of two adjacent membership functions is always one, and the points crossing these functions are coordinated with the cut-off point in the interval partition. Triangular membership functions can be generated as (4) where v is a value in the continuous feature A , L_j represents the j^{th} assigned fuzzy term to an attribute A . $\mu_{L_j}(V)$ is a fuzzy membership function which determines the membership degree of value v from the attribute A to the corresponding linguistic expression.

$$\mu_{L_j}(V) = \begin{cases} 0 & v \geq a_{j+1} \\ \frac{a_{j+1}-v}{a_{j+1}-a_j} & a_j < v < a_{j+1} \\ 1 & v = a_j \\ \frac{v-a_{j-1}}{a_j-a_{j-1}} & a_{j-1} < v < a_j \\ 0 & v \leq a_{j-1} \end{cases} \quad (4)$$

The values of a_j can be calculated by a set of cut-point d_k as (5)

$$1_j = \begin{cases} d_k - (d_{k+1} - d_k)/2 & j = k = 1 \\ a_{j-1} + 2*(d_{j-1} - a_{j-1}) & \forall j = k, d_{j-1} > a_{j-1} \end{cases} \quad (5)$$

3.3. Third stage

Then to select the best feature for branching, fuzzy information gain is computed as (3). To choose the best split attribute when the leaf needs to be expanded, only the instances stored on the leaf are considered. The idea is that if selected as a split attribute and a set of split values (one for each class), it is better to rebuild the partition defined by the instance class on the leaf to be extended. For a selected feature with numerical values, the edges of the nodes are created in proportion to the number of corresponding fuzzy values. Besides, considering the fuzzy set a label is assigned to each edge. In the case of categorical features, the edge is created in proportion to the possible values for the selected attribute. The tree traversing is performed using a selected variable and edge values. Then, the stored records in the node are deleted. In the following, the other instance is incrementally entered into the tree and converted to fuzzy values based on membership function. The entered sample surveys the tree edge until reaching the leaves based on membership degree of features, which satisfies the branching condition. A new sample with a different membership degree is stored in one or more leaves. The inference phase continues until all records of the training dataset are survived. The pseudocode of the proposed algorithm is shown in Figure 2.

```

IFDT(TS) // Incremental Fuzzy Decision Tree
Input: TS is the training dataset
ROOT = Create Node ()
For each I ∈ TS do
    UpdateFDT
End for
UpdateFDT (I, NODE)
    FI = FuzzyIns(I)
    AddInstanceToNode (NODE, I)
    Expand the node if the number of instances has reached s
    For each edge Rj ∈ NODE do
        memval[j] = ComputeMembershipValue (FI, NODE.Rj)
        if memval[j] != 0 then
            Update Node (FI, NODE.Edge)
        End.
    Expand Node (NODE)
    If NODE.Classes > 1 then
        NODE.BestAttr = ChooseBestAttribute ()
        for each FuzzyInterval in NODE.BestAttr
            Ri = Create Edge ()
            Leafi = Create Node ()
        End for
        Delete (NODE.Ins)
    else
        Update Edge (NODE.Ins, NODE.Input.Atr)
        Delete (NODE.Ins)
        NODE.numIns = 0
    End If
End.

```

Figure 2. Pseudo-code of the proposed algorithm

3.4. Fourth stage

To traverse the DT, an instance starts from the root and then descends through internal nodes until the instance reaches a leaf. To descend to a node, IFDT follows the path of the split attribute that best matches the corresponding value of the attribute of the traversal DT instance. It is done by calculating the minimum absolute difference between the instance value and the edge value. Like all FDT algorithms, the classification process in IFDT consists of traversing DTs with invisible instances until it reaches a table and assigning the class label associated with the table to the new instance.

4. RESULTS AND DISCUSSION

The general method of performing the tests is as follows: in the preprocessing step, the values of numerical features are discrete, and the fuzzy membership function is defined based on discrete intervals. Then, the dataset incrementally enters the tree and the numerical attributes in each node are converted to fuzzy values using defined membership function in preprocessing step. Next, the algorithm of the incremental FDT building is applied to fuzzified data. The datasets used for the experiments are described in Table 1. The datasets are characterized by varying amounts of instances, classes and attributes. For each dataset, the number of numeric and categorical attributes is specified.

We used 10-fold cross-validation for each dataset and algorithm. The data are divided in this manner so that 90% of the data is used for training and 10% is used for testing at each implementation. This method is repeated on each fold of data to validate the results. In all the experiments we evaluated the accuracy rate over a testing set. We have compared our algorithm with three algorithms of generalized fuzzy partition entropy-based fuzzy ID3 algorithm (GFIDT3) [25], fuzzy multi-way decision trees (FMDT) [26] and fuzzy binary decision trees (FBDTs) [26] in term of accuracy and tree complexity. The accuracy measure has been chosen based on the accuracy as (6).

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (6)$$

The accuracy of the algorithms is shown in Table 2. Another criterion for comparing decision trees is the complexity of the tree, expressed in terms of the number of nodes, the tree depth, and the number of leaves. The results for the FMDT and FBBDT ($\beta=15$) algorithms are taken from [26]. The number of whole nodes, leaves, and depth of the decision tree resulting from each technique is provided in Table 3 to assess the tree's complexity for each dataset.

As can be seen in the Table 3, the total number of nodes and the number of leaf nodes in the proposed algorithm is less than all other algorithms, which indicates a reduction in the complexity of the decision tree. But instead of reducing the complexity of the tree, the accuracy has not decreased much. Which represents the balance between the complexity and accuracy of the decision tree built into the proposed algorithm.

Table 1. The datasets used in tests

Dataset	Instances	Attributes	Classes
Poker-Hand	1025010	10 (cat:10)	2
ECO-E	4178504	16 (num: 16)	10
KDD99_2	4856151	41 (num:26, cat:15)	2
KDD99_5	4898431	41 (num:26, cat:15)	2
Susy	5000000	18 (num: 18)	2

Table 2. The accuracy achieved by algorithms

Dataset	GFIDT3	IFDT	FMDT	FBBDT
Poker-Hand	67.17	62.55	77.17	62.47
ECO-E	97.58	96.67	97.58	97.26
Susy	80.96	79.93	79.63	79.72
KDD99_2	99.99	99.80	99.98	99.99
KDD99_5	99.97	99.98	99.97	99.99

Table 3. Complexities of trees

Dataset	GFIDT3			IFDT			FMDT			FBBDT		
	Nodes	Leaves	Depth	Nodes	Leaves	Depth	Nodes	Leaves	Depth	Nodes	Leaves	Depth
Poker-Hand	30940	18561	18.60	29400	17340	18.20	30940	28561	4	44297	22149	14.75
ECO-E	16264	8448	20.73	15980	8005	19.50	222694	200048	2.73	17532	9370	24.23
Susy	18076	9754	30.46	18090	9650	29.80	805076	758064	3.46	21452	10723	14.62
KDD99_2	151	91	8.54	138	87	8.10	703	630	2.54	222	112	10.18
KDD99_5	654	302	10.65	609	286	10.04	2716	2351	2.6	779	389	10.65

5. CONCLUSION

In this paper, we have presented an algorithm for incremental induction of fuzzy decision trees. Proposed algorithm does not need to store the entire training set in memory but processes all instances of the

training set. In the non-fuzzy algorithm, to select the best attribute for the branching in continuous attributes, the calculation should be conducted on each of the values; therefore, the decision tree construction time is higher than a fuzzy algorithm. In the proposed algorithm the fuzzy information gain criterion is used to find the best attribute of the branch and the accuracy of the tree is high because of building the decision tree from the entire dataset. In general, the most important components of the proposed framework are: Achieve a balance between tree accuracy and complexity, Solve the memory limitation problem for a large dataset by entering data incrementally into the decision tree, increase model reliability by making a decision tree of all training data, lack memory and time overload due to the non-use of especially data structure.

The results of the implementation of incremental decision trees are compared with two non-incremental and non-fuzzy methods for large data sets. What can be deduced from the test results is that in the incremental method, the tree construction time is shorter because only the data from the same node is calculated when the best branch attribute is selected. Since the number of branches generated by each numeric attribute in FDT is as large as the number of fuzzy sets defined for that attribute, fewer nodes are created in the tree. Of course, it should be noted that FDT requires a preprocessing stage to determine the cutoff point and define the fuzzy set of numerical characteristics. In future research, it will be possible to evaluate the impact of each fuzzification method on the precision and complexity of the decision tree.

REFERENCES




- [1] C. Dong and Y. Guo, "Improved differentiation classification of variable precision artificial intelligence higher education management," *Journal of Intelligent & Fuzzy Systems*, pp. 1–10, May 2021, doi: 10.3233/JIFS-219036.
- [2] G. Saranya and A. Pravin, "A comprehensive study on disease risk predictions in machine learning," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 4, pp. 4217–4225, Aug. 2020, doi: 10.11591/ijece.v10i4.pp4217-4225.
- [3] W. Alajali, W. Zhou, S. Wen, and Y. Wang, "Intersection traffic prediction using decision tree models," *Symmetry*, vol. 10, no. 9, Sep. 2018, doi: 10.3390/sym10090386.
- [4] M. A. S. Barote and A. O. Bang, "Security risks, limitations and operational problems in mobile computing," *International Journal of Advent Research in Computer and Electronics*, vol. 2015, pp. 76–79, 2015.
- [5] Priyanka and D. Kumar, "Decision tree classifier: a detailed survey," *International Journal of Information and Decision Sciences*, vol. 12, no. 3, pp. 246–269, 2020.
- [6] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp. 20–28, 2021.
- [7] Y.-Y. Song and L. U. Ying, "Decision tree methods: applications for classification and prediction," *Shanghai archives of psychiatry*, vol. 27, no. 2, pp. 130–135, 2015, doi: 10.11919/j.issn.1002-0829.215044.
- [8] I. Jenhani, N. Ben Amor, and Z. Elouedi, "Decision trees as possibilistic classifiers," *International Journal of Approximate Reasoning*, vol. 48, no. 3, pp. 784–807, Aug. 2008, doi: 10.1016/j.ijar.2007.12.002.
- [9] S. B. Kotsiantis, "Decision trees: a recent overview," *Artificial Intelligence Review*, vol. 39, no. 4, pp. 261–283, Apr. 2013, doi: 10.1007/s10462-011-9272-4.
- [10] M. Mehta, R. Agrawal, and J. Rissanen, "SLIQ: A fast scalable classifier for data mining," in *International conference on extending database technology*, Springer, 1996, pp. 18–32.
- [11] J. Gehrke, R. Ramakrishnan, and V. Ganti, "RainForest—a framework for fast decision tree construction of large datasets," *Data Mining and Knowledge Discovery*, vol. 4, no. 2, pp. 127–162, 2000, doi: 10.1023/A:1009839829793.
- [12] S. Ranka and V. Singh, "CLOUDS: A decision tree classifier for large datasets," in *Proceedings of the 4th knowledge discovery and data mining conference*, 1998, vol. 2, pp. 1–34.
- [13] B. Yang, T. Wang, D. Yang, and L. Chang, "BOAI: Fast alternating decision tree induction based on bottom-up evaluation," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2008, pp. 405–416.
- [14] P. Domingos and 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, doi: 10.1145/347090.347107.
- [15] A. Franco-Arcega, J. A. Carrasco-Ochoa, G. Sánchez-Díaz, and J. F. Martínez-Trinidad, "Decision tree induction using a fast splitting attribute selection for large datasets," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14290–14300, Jun. 2011, doi: 10.1016/j.eswa.2011.05.087.
- [16] R. Thaiphon and T. Phetkaew, "Comparative analysis of discretization algorithms on decision tree," in *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, Jun. 2018, pp. 63–67, doi: 10.1109/ICIS.2018.8466449.
- [17] M. Zeinalkhani and M. Eftekhari, "Fuzzy partitioning of continuous attributes through discretization methods to construct fuzzy decision tree classifiers," *Information Sciences*, vol. 278, pp. 715–735, Sep. 2014, doi: 10.1016/j.ins.2014.03.087.
- [18] J. Ouyang, N. Patel, and I. Sethi, "Induction of multiclass multifeature split decision trees from distributed data," *Pattern Recognition*, vol. 42, no. 9, pp. 1786–1794, Sep. 2009, doi: 10.1016/j.patcog.2009.01.033.
- [19] B. Chandra and P. Paul Varghese, "Moving towards efficient decision tree construction," *Information Sciences*, vol. 179, no. 8, pp. 1059–1069, Mar. 2009, doi: 10.1016/j.ins.2008.12.006.
- [20] S. Ramírez-Gallego *et al.*, "Data discretization: taxonomy and big data challenge," *WIREs Data Mining and Knowledge Discovery*, vol. 6, no. 1, pp. 5–21, Jan. 2016, doi: 10.1002/widm.1173.
- [21] Z. Qin and J. Lawry, "Decision tree learning with fuzzy labels," *Information Sciences*, vol. 172, no. 1–2, pp. 91–129, Jun. 2005, doi: 10.1016/j.ins.2004.12.005.
- [22] R. Pecori, P. Ducange, and F. Marcelloni, "Incremental learning of fuzzy decision trees for streaming data classification," 2019, doi: 10.2991/eusflat-19.2019.102.
- [23] Xiaomeng Wang and C. Borgelt, "Information measures in fuzzy decision trees," in *2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No.04CH37542)*, 2004, vol. 1, pp. 85–90, doi: 10.1109/FUZZY.2004.1375694.
- [24] A. A. Afify, "A fuzzy rule induction algorithm for discovering classification rules," *Journal of Intelligent & Fuzzy Systems*, vol. 30, no. 6, pp. 3067–3085, Apr. 2016, doi: 10.3233/JIFS-152034.
- [25] C. Jin, F. Li, and Y. Li, "A generalized fuzzy ID3 algorithm using generalized information entropy," *Knowledge-Based Systems*,

vol. 64, pp. 13–21, Jul. 2014, doi: 10.1016/j.knosys.2014.03.014.




- [26] A. Segatori, F. Marcelloni, and W. Pedrycz, "On distributed fuzzy decision trees for big data," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 1, pp. 174–192, Feb. 2018, doi: 10.1109/TFUZZ.2016.2646746.

BIOGRAPHIES OF AUTHORS






Somayeh Lotfi    received the B.Eng. and M.S degrees in computer engineering from Najaf Abad University in 2004 and Shiraz University, Iran, in 2009, respectively. She is a PhD student in Computer Engineering at the Islamic Azad University Science and Research Branch (Tehran, Iran). She is also a member of faculty of Computer science and Engineering at Islamic Azad University, Bandar Abbas Branch. Her research interests include Data mining, Machine Learning, Algorithms Design and Programming. She can be contacted at email: s.lotfi@iauba.ac.ir.






Mohammad Ghasemzadeh    is an Associate Professor of Computer science and Engineering at Yazd University in Iran. Received his BSc degree in computer science and Engineering from Shiraz University, Shiraz, Iran in 1989, the MSc degree in AI and robotics from Amirkabir University of Technology (Tehran Polytechnic) 1995. For his PhD research program, he worked at the University of Trier and at the University of Potsdam both in Germany, from Feb. 2002 to Feb. 2006. He completed his PhD degree in Theoretical Computer Science, in Nov. 2005. He also spent his sabbatical as guest/postdoc researcher at HPI in Germany, from Feb. to Nov. 2016. He can be contacted at email: m.ghasemzadeh@yazd.ac.ir.



Mehran Mohsenzadeh    is an Assistant Professor in the Department of Computer Engineering, Science and Research Branch, IAU University of Iran. Received his B.E degree (Software Engineering) in 1997 from Shahid Beheshti University and M.E (in 1999), and PhD in Software Engineering 2004 from IAU University, Tehran. His major interests are Cloud Computing, Software Engineering and Big Data and has published more than eighty papers (author/co-author) in International Conferences and journals. He can be contacted at email: mohsenzadeh@srbiau.ac.ir.



Mitra Mirzarezaee    is an Assistant Professor of Computer Engineering at Islamic Azad University of Iran, Science and Research Branch. Her areas of interests include Intelligent Systems, Bioinformatics, Statistical Pattern Recognition, Structural Pattern Recognition and Machine Learning. She can be contacted at email: mirzarezaee@srbiau.ac.ir.