JACET

J. ADV COMP ENG TECHNOL, 7(1) Winter 2021

Scalable Fuzzy Decision Tree Induction Using Fast Data Partitioning and Incremental Approach for Large Dataset

Somayeh Lotfi[1], Mohammad Ghasemzadeh[2], Mehran Mohsenzadeh[3], Mitra Mirzarezaei[4]
1- Computer Engineering Department, Science and Research Branch, Islamic Azad University, Tehran, IRAN.
2- Computer Engineering Department, Yazd University, Yazd, IRAN. (m.ghasemzadeh@yazd.ac.ir)
3- Computer Engineering Department, Science and Research Branch, Islamic Azad University, Tehran, IRAN.
4- Computer Engineering Department, Science and Research Branch, Islamic Azad University, Tehran, IRAN.

Received (2020-12-29) Accepted (2021-04-11)

Abstract: *Decision tree is one of the popular methods for learning and reasoning through recursive partitioning of data space. To choose the best attribute in the case on numerical features, partitioning criteria should be calculated for individual values or the value range of each attribute should be divided into two or more intervals using a set of cut points. In partitioning range of attribute, the fuzzy partitioning can be used to reduce the noise sensitivity of data and to increase the stability of decision trees. Since the tree building algorithms need to keep in main memory the whole training dataset, they have memory restrictions. In this paper, we present an algorithm that builds the fuzzy decision tree on the large dataset. In order to avoid storing the entire training dataset in main memory and overcome the memory limitation, the algorithm builds DTs in an incremental way. In the discretization stage, a fuzzy partition was generated on each continuous attribute based on fuzzy entropy. Then, in order to select the best feature for branches, two criteria, including fuzzy information gain and occurrence matrix are used. Besides, real datasets are used to evaluate the behavior of the algorithm in terms of classification accuracy, decision tree complexity, and execution time as well. The results show that proposed algorithm without a need to store the entire dataset in memory and reduce the complexity of the tree is able to overcome the memory limitation and making balance between accuracy and complexity .*

Keywords: *Fuzzy Decision trees, Large dataset, Fuzzy entropy, Fuzzy partitioning*

How to cite this article:
Somayeh Lotfi, Mohammad Ghasemzadeh, Mehran Mohsenzadeh, Mitra Mirzarezaei. Scalable Fuzzy Decision Tree Induction Using Fast Data Partitioning and Incremental Approach for Large Dataset. J. ADV COMP ENG TECHNOL, 7(1) Winter 2021 : 55-66

I. INTRODUCTION

THE use of decision tree as a classification algorithm was successful in many applied fields such as security assessment, health system, road traffic and prediction learning styles [1,2]. The decision tree is popular because of the simplicity of their learning. In addition, it is an interpretable classification method, which is representative of output's extraction from the inputs. Moreover, the learning process of decision tree requires adjusting only a few parameters [3, 4]. In the past decades, a large number of algorithms have been proposed to create a decision tree, including ID3, C4.5, CART, and their developed examples [5, 6]. Given that data in the database often include many features and records, the basic methods are not useful for large datasets. Since the algorithms require to store the entire dataset in memory to build tree and the complexity of the tree despite a large amount of data, these algorithms are inefficient to build decision tree from large data [7].

The data with numerical values should be partition during the decision tree building. The most common method is to partition each feature into two intervals using all

values of the attribute. The other method is the conversion of each attribute to a few intervals through multiple cutting points [8, 9]. Therefore, the numerical features are divided into intervals and the discrete intervals behave like categorical values. The discretization results in 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 decision tree to noisy data. Also, if the dataset contains m numerical feature (with the maximum k difference value) for each branching the branch criteria should be calculated mk times. A solution to this problem is the use of soft discretization based on fuzzy theory.

Fuzzy decision tree (FDT) was built with the purpose of combining the decision tree with approximate reasoning provided by the fuzzy display. FDT uses the popularity and understandability of the decision tree in applications to learn from the samples and the ability to deal with the uncertain information in the fuzzy display [10]. In FDT, a node is specified by a fuzzy set instead of a set. Besides, the process of fuzzy reasoning allows two or more rules validate simultaneously and the final result is obtained by combining multiple results [11, 12]. Generally, FDT algorithms require a fuzzy partitioning on each continues feature. Consequently, continue features are converted to discrete values by optimizing a series of index [13]. The method of discretization affects the classification accuracy. In [14], the authors investigated the effect of discretization methods on accuracy and complexity of FDT. They studied several methods, including fuzzy partitioning, Boolean discretization, and different types of the membership function, and evaluated the effect of each method on accuracy and complexity of FDT. It should be noted that a typical decision tree is mainly employed to classify the small dataset and it is not suitable for a large amount of information [15, 16]. Among the simplest method to deal with large data, one can name sampling, list structure methods [17], incremental methods [18], parallel processing, and distributed calculations. A novel distributed fuzzy discretize, which generates strong fuzzy partitions for each continuous attribute and a distributed implementation of an FDT learning algorithm, are introduced in [19].

In [20], the partitioning and sampling techniques for big data analysis were reviewed. Although some of these algorithms deal with memory limitation by selecting parts of training data, in order to select desired data subset, they deal with time and computing cost. The results of this paper show that the accuracy of the results depends on the quality of the selected samples and in these methods, the reliability of the model is reduced due to loss of part of the data.

In some other methods, the building of the decision tree incrementally is done using entire training data [21]. In these methods, the data respectively, are used in the tree construction. Besides, these methods are employed for stream data. In order to deal with the mentioned problems in the decision tree, the present study aims to present an incremental scalable algorithm based on fast partitioning. By entering data into the tree incrementally, there is no need to store the entire dataset in main memory. Also a batch-incremental approach for mining data streams proposed in [22] that pre-processes this data by producing successive embedding on a stream of disjoint batches. Besides, in each node, the attribute with the highest probability for prediction is selected to create a new branch. With building the decision tree by entire training dataset without a 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. With local discretization on a dataset of each node, continuous data discretized through fuzzy discretization [12]. with the presence of the fuzzy discretization technique, the accuracy of classification is increased.

The structure of the present study is as follows: in section 2, the basic concepts of the decision tree, discretization of numerical values, the definition of fuzzy membership functions on it and tasks ha ve done in this case are presented. In section 3, the proposed algorithm to build the incremental fuzzy decision tree is described. The results of algorithm implementation are shown in section 4. Finally, in section 5, the conclusion of the research is presented.

II. BASIC CONCEPTS

1. Fuzzy Sets

Contrary to crisp or two-valued logics which whether they are true or false, the fuzzy logic is many-valued logic. The fuzzy logic provides the mathematical model on domains with non-sharp boundaries [23]. Assume that X is a global set of x variable; the fuzzy set of A on X is defined by membership functions as follow, which indicated the membership degree of x to A fuzzy set: $\mu_A(x): X \rightarrow [0,1]$

The most important issue in fuzzy logic is the definition of number, type of parameters, and membership function. A membership function describes the degree of membership for each sample in the corresponding fuzzy set. Besides membership functions is defined for both input and output data. Triangular, Trapezoidal, and Gaussian functions are the most common membership functions that are used to identify a pattern. In the present research, the triangular membership functions were employed because of its simplicity and application in fuzzy sets.

2. classification of continuous attributes

To build a decision tree by continuous-valued data, an appropriate threshold, such as T, is selected to divide the range of each attribute. Based on the value of the threshold, the value of a continuous attribute of A was divided into A1=[min(A), T] and A2= [T, Max(A)] intervals. According to the threshold, the A≤T condition is assigned to the left branch of the node and the A>T condition is assigned to the right branch [24]. The best threshold for partitioning is determined based on the information gain derived from the corresponded classification to that partition. If the samples in the node are sorted based on continuous feature values of A, both adjacent points will show a potential threshold of $T=(a_i+a_{i+1})/2$ to create a cutting point and a partitioning on A. The division results in the formation of crisp intervals.

In [9], it was proved that if samples are arranged in ascending order based on their continuous feature values, only the boundary points of the class can be the cutting points to obtain the maximum information gain in the classification. It means that if a_i and a_{i+1} belong to a class, the cutting point between them cannot lead to areas with maximum information gain. Therefore, it is possible to create a small set of cutting points from boundary points of the class. Based on an intuitive approach, in order to obtain the fuzzy intervals for each continuous feature, its domain is discretized into several crisp intervals. Then, a crisp interval can be fuzzy by assigning a suitable membership function to it. It is possible the membership function to be assigned by experts or through statistical data. In [14], the different methods were investigated to produce fuzzy parts. Besides, the effect of the methods on accuracy and complexity of fuzzy decision tree was studied. The results showed that fuzzy partitioning based on fuzzy entropy (FPFE) was the efficient method.

In the present study, FPFE method was employed to generate the triangular fuzzy partitions for large dataset. FPFE is a supervised recursive method, which produces the candidate fuzzy partitions and it is evaluated by fuzzy entropy. The algorithm selects the fuzzy partition to minimize the fuzzy entropy value [10]. Then, it divides the domain of the continuous feature into two subsets. The process is repeated for each produced subsets until the stop condition. Considering the time-consuming processes of sorting and evaluating for a large number of fuzzy partitions encountering large datasets, an approximate version of the fuzzy partitioning method (partitions with a same number of replication) was used.

3. Fuzzy decision tree

Classification is defined as assigning the class C_n to an unlabeled sample among a predefined set $C=\{C_1, C_2, ..., C_N\}$ of class N. It should be noted that each sample has numerical and categorical attributes. Assume that $X=\{X_1, X_2, \ldots, X_F\}$ shows the set of attributes. In the case of numerical features, X_f is defined in global set of $\bigcup_f \subset R$. Moreover, in the case of categorical features, X_f is defined as categorical values on $L_f=\{L_{f,1},..,L_{f,T}\}$ set. If $TR=\{(x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N)\}$ is assumed as training dataset, the continuous attributes for each sample (x_i,y_i) are {EMBED Equation. $y_i \in C$ and $x_{i,f} \in \bigcup_f$ the categorical attributes are $x_{i,f} \in L_f$, that i=1,2,…N and f=1,2,…,F. FDT

is generated in the top-down method by recursive partitioning of feature space. Generally, the FDT building algorithm includes choosing the appropriate feature for branching and creating a branch in each node [11,12]. At each stage of the tree construction, the feature using for branching in each node is selected. The feature selection should be done using the appropriate criterion. This criterion actually measures the difference between the uniform distribution of class labels in parent nodes and child nodes created by that branch. Selection of the weak features for branch results in poor performance of decision tree. The branching method determines the best feature for branch as well as number of child nodes. The method is used for multiple and binary branches. Several criteria are suggested to select the best attribute for branching in a decision tree [25]. In the present paper, the fuzzy information gain criterion is employed, which is described in section 3-1.

4. Look-ahead based fuzzy decision tree

Look-ahead based FDT is a special method to evaluate the abilities of attributes' classification along branches of a node to divide and create a smaller tree [26]. In Look-ahead based FDT, the ability of attributes' classification is measured using occurrence matrix. Accordingly, the distance between two samples in their circular neighborhood (r radius) is calculated by Eq. 1.

If $\mu_i^{(k)}(x)(1 \le k \le n, 1 \le i \le m_k, 1 \le x \le N)$ represents the membership degree of sample x to the i[th] value of k[th] attribute, the distance between sample x and y is defined as follow:

$$D_{xy} = \sum_{k=1}^{n} \sum_{i=1}^{m_k} | \mu_i^{(k)}(x) - \mu_i^{(k)}(y) | \qquad (1)$$

For each sample x in this set, we can limit its circular neighborhood to samples in radius r of x. Then, local occurrence matrix p is defined for sample x as follow. Assume that $\mu_i(x) = [\mu_1(x), ..., \mu_c(x)], 1 \le j \le c$ present the membership degree of sample x in class j, the local occurrence matrix of sample x is defined as follow:

$$p(x) = \sum_{y, D_{xy} \le r} \mu(x)^T \times \mu(y) \qquad (2)$$

Where $\mu(x)^T$ is a transpose matrix and r is the radius of circular neighborhood x. It is possible to calculate the occurrence matrix of each attribute by a local occurrence matrix. Local occurrence matrix of attribute k is calculated as follow:

$$W^{(k)} = \sum_{i=1}^{m_k} \sum_{x} p(x) \qquad (3)$$

The ability of classification of attribute k is calculated by Eq.4 as follow:

$$L^{(k)} = \sum_{i=1}^{c} W_{ii}^{(k)} - \sum_{i=1}^{c} \sum_{j=1}^{c} W_{ij}^{(k)} \qquad (4)$$

According to the value of $L^{(k)}$, it is possible to determine the highest ability of attribute classification to build a decision tree.

III. PROPOSED ALGORITHM

In order to deal with the problem of storing the training data in main memory and reduce computational overload, the presented algorithm enters the training data incrementally into the tree. Each record has traversed the tree and stored in leaves. Then, if the number of records in a leaf exceeds the number of s (user input), the algorithm decides to expand or update the leaf. If all the records in a node belong to the same class, that node is considered as a leaf and only the label of the input edge to the node will update, otherwise, the leaf will be expanded. At this stage, the best feature for branch should be selected. In a decision tree, the partitioning is done for all numerical attributes and the best attribute is selected based on information gain obtaining from classification. In the proposed method, at this stage, the continuous values are converted to fuzzy values, the attribute with highest classification ability is selected for branching using evaluation criteria, and the children are branched from the attribute. The advantage of this method is that it does not require all data to be stored in the main memory, and only with s record, the development is done. Moreover, the

soft discretization is performed on the attributes using the fuzzy values. Placing all members of a particular class in same direction results in the improper development of the tree and only the edges are updated. Therefore, in order to prevent this problem, a function is used to input the record into the tree randomly.

1. Selecting the feature for branch

One of the important branch criteria in the traditional and under memory decision tree algorithm is information gain [27]. For each of the attributes with numerical values, each dataset must be arranged in ascending order according to the attribute. Also, in order to have a binary classification, each time the boundary of the two classes is placed between two number and the amount of information gain resulting from classification is calculated based on that particular value. Then, the boundary with the highest accuracy of classification is selected. For categorical attributes, calculations must be made for each value of the attribute. Then, among the all information gain, the attribute with the maximum value is selected for branching. In some method, in order to reduce the computing value of the numerical feature, the mean values are used, but the accuracy of classification is reduced. In the present study, the values of the numerical attribute are converted to fuzzy values. Accordingly, a two-step algorithm is used to generate the membership function using partitioning method. In the first step, the range of each continuous feature is divided into multiple non-fuzzy intervals using discretization method. Then, in step two, each non-fuzzy section is converted into a fuzzy part by defining a membership function on it. In the first step, it is possible to use the different discretization criteria such as entropy, dependency, and accuracy [28].

2. Feature selection based on information gain

The information gain is the common criterion to choose the feature for branching in the tree. The parameter selects a feature for branching to maximize the difference between the information gain of the target node and the nodes of the child. Information gain is calculated by definition of entropy. Entropy is a statistical feature, which shows the division quality of training examples through an attribute [29]. It is calculated by Eq.5.

$$Entropy\,(s) = -\sum_{i=1}^{c} p_i \log_2 p_i \qquad (5)$$

Where the variable S is samples set, C is the number of various value of the objective function, and p_i is the fraction of the samples in which the objective function has the value of i. The Eq.5 is converted to fuzzy entropy for calculation fuzzy values (Eq.6):

$$Entropy_f\,(s) = -\sum_{i=1}^{c} \left(\sum_{j=1}^{n} \frac{\mu_{ij}}{|s|}\right) \log_2 \sum_{j=1}^{n} \frac{\mu_{ij}}{|s|} \qquad (6)$$

Similar to entropy, the samples set is shown by S, and |S| is sample's count of the set of S, n is the number of entire samples, and C is the number of objective function's classes. The equation used to calculate the fuzzy entropy is slightly different from Eq.6, where instead of |S| the total membership degrees of all values of objective function was used in entire samples. Therefore, the entropy is calculated by Eq.7:

$$A = \sum_{j=1}^{n} \frac{\mu_{ij}}{\sum_{k=1}^{n}\sum_{i=1}^{c} \mu_{ik}}$$
$$Entropy_f\,(s) = -\sum_{i=1}^{c} A \log_2 A \qquad (7)$$

The information gain is the criterion to select the feature for classification of each group. The information gain is calculated by Eq.8.

$$InformationGain\,(S, A) =$$
$$Entropy\,(s) - \sum_{v \in A}^{m} \left(|s_v| / |s|\right) Entropy\,(s_v) \qquad (8)$$

Where A is an attribute, m is the number of the values of this attribute, and S_v is the samples set in which the attribute A has a special value v. If in the Eq.8, instead of the entropy the fuzzy

entropy is used, the relation is converted to fuzzy information gain:

$$Information\ Gain(S,A) = Entropy_f(s) - \sum_{v \in A}^{m} (|s_v| / |s|) Entropy_f(s_v) \quad (9)$$

What is to be considered in Eq.9 is the calculation of fuzzy entropy S_v. Because, certainly it cannot be said that in a specific sample, a feature such as A has a definite value, such as v, but the membership is shown with a membership degree. Accordingly, algebraic multiplication operators are employed. To calculate the ratio of S_v to S, the Eq.10 is used:

$$|s_v| / |s| = \sum_{i=1}^{n} \mu_{iv} \Big/ \sum_{k=1}^{n} \sum_{j=1}^{C_t} \mu_{kj} \quad (10)$$

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

3. Generating the membership function to fuzzy attribute classification

One of the important aspects of the construction of an FDT is the decision of fuzzy sets and the way of converting the values of numerical attributes to fuzzy values. In other words, the accuracy of the tree largely depends on the proportion of fuzzy sets with the given data. Many of the proposed algorithms for constructing an FDT assume that fuzzy sets and corresponding membership functions are specific. It means these algorithms trust an expert to manually determine the fuzzy sets. However, it is very difficult for an expert to estimate appropriate fuzzy sets. In this section, the way of fuzzifying the continuous features and creation of the membership functions for each fuzzy interval is explained.

Assuming that the set $\{m_1, m_2, ..., m_k\}$ represents the medians found in the database with n attributes and $m_i = \{a_{i1}, a_{i2}, ..., a_{in}\}$, is the i^{th} median, we want to find the fuzzy sets for j^{th} quantitative attribute. The range of considered attribute is from mini to maxi. The set of $\{a_{1j}, a_{2j}, \ldots, a_{kj}\}$ 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 [30]. The triangular membership functions were used due to their simplicity. 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 membership degree 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 follows.

$$\mu_{L_i}(V) = \begin{cases} 1 & v \le a_1 \\ \dfrac{a_2 - v}{a_2 - a_1} & a_1 < v < a_2 \\ 0 & v \ge a_2 \end{cases} \quad (11)$$

$$\mu_{L_j}(V) = \begin{cases} 0 & v \ge a_{j+1} \\ \dfrac{a_{j+1} - v}{a_{j+1} - a_j} & a_j < v < a_{j+1} \\ 1 & v = a_j \\ \dfrac{v - a_{j-1}}{a_j - a_{j-1}} & a_{j-1} < v < a_j \\ 0 & v \le a_{j-1} \end{cases} \quad (12)$$

$$\mu_{L_{n_l}}(V)=\begin{cases}1 & v \geq a_{n_l}\\ \dfrac{v-a_{n_v-1}}{a_{n_v}-a_{n_v-1}} & a_{n_l-1}<v<a_{n_l}\\ 0 & v \leq a_{n_l-1}\end{cases} \quad (13)$$

Where v is a value in the continuous feature A, L_j represents the j^{th} assigned fuzzy term to an attribute A. $\mu_{Lj}(v)$ is a fuzzy membership function which determines the membership degree of value v from the attribute A to the corresponding linguistic expression. In this case, the values of a_j should be determined. The values can be calculated by a set of cut-point ck as follow:

$$a_j=\begin{cases}c_k-(c_{k+1}-c_k)/2 & j=k=1\\ a_{j-1}+2*(c_{j-1}-a_{j-1}) & \forall j=k, c_{j-1}>a_{j-1}\end{cases} \quad (14)$$

4. Incremental FDT

Algorithm 1 shows the general scheme of Incremental Fuzzy Decision Tree construction. The algorithm starts the tree construction by an empty root node. Training records are converted to fuzzy value using membership function one by one. Then the values enter the tree and store in the root. After maximizing the number of stored sample in the root, the node is developed. To develop the node, it should choose a feature with homogeneous partitioning ability on data. In the proposed method to select the best feature to branch, the Eq. 9 and Eq.4 were used respectively. Algorithm 2 details the pseudo code of the update IFDT step.

Algorithm1: Incremental Fuzzy Decision Tree

```
procedure IFDT(TS, S)
    //TS is the training dataset
    //S is the maximum of instances in the nodes
      ReorganizeTS(TS)
      ROOT = CreateNode()
      for each I Є TS do
          UpdateFDT(I , ROOT)
      end for
end procedure
```

Algorithm2: Update Incremental Fuzzy Decision Tree

```
procedure UpdateFDT (I, NODE)
    //I is instance to be processed,
    // NODE is the node in the tree to be traversed
      FI = FuzzyIns(I)
      if NODE.numIns< S then
          AddInstanceToNode(NODE , I)
          NODE.numIns = NODE.NumIns + 1
          if  NODE.numIns= S then
            ExpandNode(NODE)
                    NODE.numIns = NODE.numIns  + 1
      endif
   else
          for each edge Rj Є NODE do
                    memval[j] = ComputeMembershipValue(FI
, NODE.Rj)
                    if memval[j] != 0 then
                    UpdateFDT( FI , NODE.Edge)
endif
end procedure
```

In order to employ the occurrence matrix, instead of selecting samples in neighborhood radius r of sample X, all samples in a node was considered as the neighborhood of X. It should be noted that the use of occurrence matrix to find the best feature results in the creation of the less complicated tree. 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 satisfy the branching condition. A new sample with a different membership degree is stored in one or more leaves. When the number of stored samples in a node reaches the maximum number s, one of the following conditions occurs:

- If all the stored records in a node are in the same class, a node is considered as a leaf and the input edge to the leaf is updated. Besides, all stored records in the leaf are deleted.

- If the records in the leaf are of the different classes, the updating process of the tree is recalled. The tree is traversed per entered sample until reaching the leaf. Then, the considered sample is stored in one or more leaves based on their membership value in corresponding fuzzy

function with features.

Inference phase continues until all records of training dataset are survived. Algorithms 3 detail the pseudo code of the expand a node in IFDT learning.

```
Algorithm3:  Expand Node
procedure ExpandNode(NODE)
  // NODE is the node in the tree to be expanded
    if NODE.Classes > 1 then
            NODE.BestAttr= chooseBestAttribute ()
             for each FuzzyInterval in NODE.BestAttr
                 Ri = CreateEdge()
                 Leafi = CteateNode()
             endfor
              Delete(NODE.Ins)
    else
          UpdateEdge(NODE.Ins,NODE.Input.Attr)
           Delete (NODE.Ins)
           NODE.numIns = 0
     Endif
              memval[j] = ComputeMembershipValue(FI , NODE.Rj)
              if memval[j] != 0 then
         UpdateFDT( FI , NODE.Edge)
      endif
end procedure
```

Fig. 1 shows the overall IFDT construction process.

5. Classification of new sample

After constructing the decision tree, a non-label sample such as x is assigned to class Cm by following the paths from the root to one or more leaves, which satisfy the branching conditions. In a classic decision tree, a node represents a crisp set and each leaf is labelled with a unique class label. In FDT, each node represents a fuzzy subset. Therefore, the sample x can active multiple paths in the tree and reach more than one leaf with different degrees of compliance. The membership degree of the test sample in a class is computed by multiplication of membership values for the considered feature of each path and membership value of each class in a relative node of the leaf. Then, the membership degree to that class is obtained for the new sample by summing the membership values of each considered class in all paths. The class with maximum membership values is assigned to a class of test sample. This process is conducted on all leaves.

In order to determine the class of non-labelled sample x, the corresponding class with the maximum sum of membership value is assigned to sample. The value of each class is calculated by summing the all belonging digress in per leaf for corresponding class.

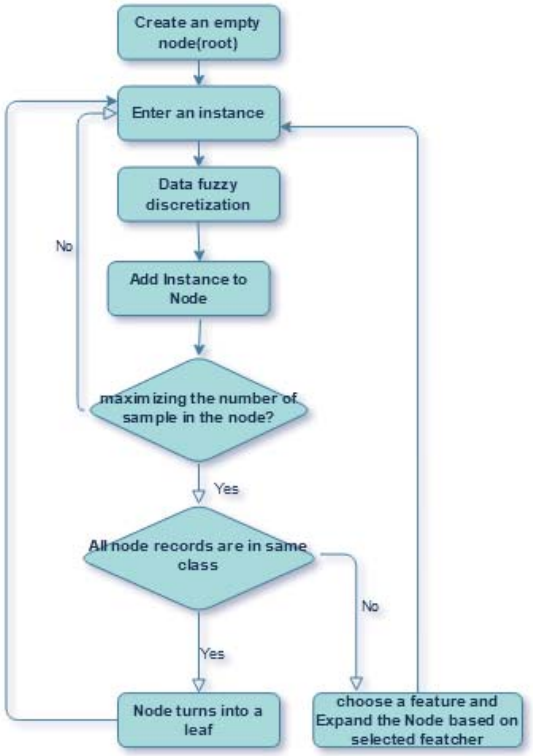

Fig. 1. Incremental FDT

IV. EXCREMENTAL RESULT

The general method of performing the tests is as follows: in the preprocessing step, the values of numerical features are discrete and fuzzy membership function is define based on discrete intervals. Then, the dataset incrementally enters to the tree and the numerical attributes are converted to fuzzy values using defined membership function in preprocessing step.

Next, the algorithm of the incremental FDT building is applied on fuzzified data. The generated decision tree was compared with two incremental non-fuzzy decision tree and non-incremental FDT in terms of accuracy, complexity, and runtime for building DTs. In all simulations, the class label is not fuzzy and each sample belongs to a single class. In incremental FDT was tested on two following scenarios:

1- The use of fuzzy gain ratio base on Eq.9 to find the best feature for branching

2- Calculation of occurrence matrix to find an attribute with highest classification ability using Eq.1 to 4.

As shown in Table I, we employed 5datasets freely UCI2 repository. The datasets are characterized by different numbers of instances, classes and attributes. For each dataset, the number of numeric and categorical attributes is specified.

TABLE I
THE DATASETS USED IN TESTS

Dataset	Instances	Attributes	Classes
Poker-Hand	1025010	10 (cat:10)	10
ECO_E	4178504	16(num: 16)	10
KDD99_2	4856151	41 (num:26, cat:15)	2
KDD99_5	4898431	41 (num:26, cat:15)	5
Susy	5000000	18(num: 18)	2

In order to implement the proposed method and other decision tree algorithms, a system with RAM of 10GB, CPU3.8GHz, and 64-bit Win10 operating system is used. During the comparison of the algorithms, we tried to ensure that all the conditions were met.

For each dataset and for each algorithm, 10-fold cross-validation is done on the dataset. In this method, the data is divided so that at each implementation, 90% of the data for training and 10% for the testing are considered. To validate the results, this method is repeated on each fold of data, and in incremental algorithms with different values of s and the average results is reported.

The accuracy of the proposed algorithm to choose the best attribute for the branch by the two scenarios and comparison of DTFS [21], GFIDT3[31], FMDT [19] and FBDT [19] algorithms are shown in Table 2. The results for the FMDT and FBDT(β=15) algorithm are taken from [19].

In Table 3, the number of entire nodes, leaves and the depth of the decision tree resulting from each component of the algorithms is shown to evaluate the complexity of the tree for each dataset. Construction time of each decision tree in terms of second is presented in Table 4.

Since non-incremental algorithms in each branching should put the entire dataset in the main memory to find the best attribute, according to the reported results of Table 2 to 4, the algorithms faced with memory limitation. Besides, in the non-incremental method, branching criterion is calculated on entire dataset of the considered branch, which leads to an increase in execution time. It should be noted that the accuracy of the tree is higher than others because of making decision tree from the entire dataset. In the non-fuzzy algorithm, in order to select the best attribute for the branching in continues attributes, the calculation should be conducted on each of the values;

TABLE II
AVERAGE ACCURACY ACHIEVED BY ALGORITHMS

Dataset	DTFS	GFIDT3	IFDT with Fuzzy Info Gain	IFDT with Occurrence Matrix	FMDT	FBDT
Poker-Hand	58.47	67.17	62.55	62.47	77.17	62.47
ECO_E	97.26	97.58	96.67	95.56	97.58	97.26
Susy	79.72	80.96	79.93	79.30	79.63	79.72
KDD99_2	99.95	99.99	99.80	99.98	99.98	99.99
KDD99_5	99.94	99.97	99.98	99.94	99.97	99.99

TABLE III
COMPLEXITIES OF ALGORITHMS:
A) NUMBER OF NODES

Dataset	DTFS	GFIDT3	IFDT with Fuzzy Info Gain	IFDT with Occurrence Matrix	FMDT	FBDT
Poker-Hand	44297	30940	29400	28194	30940	44297
ECO_E	17532	16264	15980	14970	222694	17532
Susy	21452	18076	18090	17876	805076	21452
KDD99_2	222	151	138	121	703	222
KDD99_5	779	654	609	544	2716	779

B) NUMBERS OF LEAVES

Dataset	DTFS	GFIDT3	IFDT with Fuzzy Info Gain	IFDT with Occurrence Matrix	FMDT	FBDT
Poker-Hand	22149	18561	17340	15651	28561	22149
ECO_E	8741	8448	8005	7809	200048	9370
Susy	10723	9754	9650	8954	758064	10723
KDD99_2	112	91	87	61	630	112
KDD99_5	389	302	286	272	2351	389

C) AVERAGE TREE DEPTH

Dataset	DTFS	GFIDT3	IFDT with Fuzzy Info Gain	IFDT with Occurrence Matrix	FMDT	FBDT
Poker-Hand	21.75	18.60	18.20	16.60	4	14.75
ECO_E	23.14	20.73	19.50	17.54	2.73	24.23
Susy	33.62	30.46	29.80	27.46	3.46	14.62
KDD99_2	10.18	8.54	8.10	8.54	2.54	10.18
KDD99_5	11.68	10.65	10.04	9.56	2.6	10.65

TABLE IV
THE EXECUTION TIMES (IN SECONDS) FOR ALGORITHMS

Dataset	DTFS	GFIDT3	IFDT with Fuzzy Info Gain	IFDT with Occurrence Matrix
Poker-Hand	15	25	10	13
ECO_E	360	690	320	340
Susy	400	850	390	410
KDD99_2	68	120	66	70
KDD99_5	95	136	90	100

Therefore, the decision tree construction time is higher than a fuzzy algorithm. Also, selection of a numerical feature result in the creation of binary branches in the non-fuzzy tree and the other values of feature should be evaluated at other levels of the tree. The tree has more node and depth compared to a fuzzy tree, which shows the complexity of the tree.

Since non-incremental algorithms in each branching should put the entire dataset in the main memory to find the best attribute, according to the reported results of Table 2 to 4, the algorithms faced with memory limitation. Besides, in the non-incremental method, branching criterion is calculated on entire dataset of the considered branch, which leads to an increase in execution time. It should be noted that the accuracy of the tree is higher than others because of making decision tree from the entire dataset. In the non-fuzzy algorithm, in order to select the best attribute for the branching in continues attributes, the calculation should be conducted on each of the values; therefore, the decision tree construction time is higher than a fuzzy algorithm. Also, selection of a numerical feature result in the creation of binary branches in the non-fuzzy tree and the other values of feature should be evaluated at other levels of the tree. The tree has more node and depth compared to a fuzzy tree, which shows the complexity of the tree.

V. CONCLUSION

In the present study, the incremental method was provided to build the decision tree on a large dataset. In the proposed algorithm, due to the incremental entrance of data to the tree, there is no need to store the entire dataset in main memory. On the other hand, in the decision-tree-building algorithm, the branching criterion should be calculated for all values per numerical attribute. Therefore, the calculation of branching criterion is time-consuming. Accordingly, in the proposed algorithm, the fuzzified value of the attribute was used and the two criteria of fuzzy information gain and occurrence matrix were employed to find the best attribute for the branch. The results of the implementation of incremental decision tree were compared with two non-incremental

and non-fuzzy methods on large datasets. The experimental results show that in the incremental method, the tree construction time is less because, in choosing the best branching attribute, the calculations only are performed on the data of the same node. Since the number of the generated branch for each numerical attribute in FDT is as large as the number of fuzzy sets defined on that attribute, the created node in this tree is less. Of course, it should be noted that FDT requires the preprocessing stage to determine the cut-points and define the fuzzy sets on numerical features. In the future research, it is possible to evaluate the effect of each fuzzified method on the accuracy and complexity of decision tree.

REFERENCES

1. Khakata, E., Omwenga, V. and Msanjila, S., 2020. Prediction of Student Learning Styles using Data Mining Techniques. Journal of Advances in Computer Engineering and Technology, 6(2), pp.107-118.

2. Goetz, T., 2010. The decision tree: taking control of your health in the new era of personalized medicine. Rodale.

3. Han, J., Pei, J. and Kamber, M., 2011. Data mining: concepts and techniques. Elsevier.

4. Rokach, L. and Maimon, O.Z., 2008. Data mining with decision trees: theory and applications (Vol. 69). World scientific.

5. Quinlan, J.R., 2014. C4. 5: programs for machine learning. Elsevier.

6. Breiman, L., Friedman, J., Stone, C.J. and Olshen, R.A., 1984. Classification and regression trees. CRC press.

7. Kotsiantis, S.B., 2013. Decision trees: a recent overview. Artificial Intelligence Review, 39(4), pp.261-283.

8. Fayyad, U. and Irani, K., 1993. Multi-interval discretization of continuous-valued attributes for classification learning.

9. Fayyad, U.M. and Irani, K.B., 1992. On the handling of continuous-valued attributes in decision tree generation. Machine learning, 8(1), pp.87-102.

10. Zheng, H., He, J., Zhang, Y., Huang, G., Zhang, Z. and Liu, Q., 2019. A general model for fuzzy decision tree and fuzzy random forest. Computational Intelligence, 35(2), pp.310-335.

11. Yu, H., Lu, J. and Zhang, G., 2017, November. Learning a fuzzy decision tree from uncertain data. In 2017 12th International Conf. on Intelligent Systems and Knowledge Engineering (pp. 1-7).

12. Hishamuddin, M.N.F., Hassan, M.F. and Mokhtar, A.A., 2020, February. Improving Classification Accuracy of Random Forest Algorithm Using Unsupervised Discretization with Fuzzy Partition and Fuzzy Set Intervals. In Proceedings of the 2020 9th International Conference on Software and Computer Applications (pp. 99-104).

13. Garcia, S., Luengo, J., Sáez, J.A., Lopez, V. and Herrera, F., 2012. A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. IEEE Trans. on Knowledge and Data Engineering, 25(4), pp.734-750.

14. Zeinalkhani, M. and Eftekhari, M., 2014. Fuzzy partitioning of continuous attributes through discretization methods to construct fuzzy decision tree classifiers. Information Sciences, 278, pp.715-735.

15. Lomax, S. and Vadera, S., 2013. A survey of cost-sensitive decision tree induction algorithms. ACM Computing Surveys (CSUR), 45(2), pp.1-35.

16. Lomax, S. and Vadera, S., 2013. A survey of cost-sensitive decision tree induction algorithms. ACM Computing Surveys (CSUR), 45(2), pp.1-35.

17. Mehta, M., Agrawal, R. and Rissanen, J., 1996, March. SLIQ: A fast scalable classifier for data mining. In International conference on extending database technology (pp. 18-32). Springer, Berlin, Heidelberg.

18. Hulten, G. and Domingos, P., 2016. Mining Decision Trees from Streams. In Data Stream Management (pp. 189-208). Springer, Berlin, Heidelberg.

19. Segatori, A., Marcelloni, F. and Pedrycz, W., 2017. On distributed fuzzy decision trees for big data. IEEE Transactions on Fuzzy Systems, 26(1), pp.174-192.

20. Mahmud, M.S., Huang, J.Z., Salloum, S., Emara, T.Z. and Sadatdiynov, K., 2020. A survey of data partitioning and sampling methods to support big data analysis. Big Data Mining and Analytics, 3(2), pp.85-101.

21. Franco-Arcega, A., Carrasco-Ochoa, J.A., Sánchez-Díaz, G. and Martínez-Trinidad, J.F., 2011. Decision tree induction using a fast splitting attribute selection for large datasets. Expert Systems with Applications, 38(11), pp.14290-14300.

22. Bahri, M., Pfahringer, B., Bifet, A. and Maniu, S., 2020, April. Efficient batch-incremental classification using umap for evolving data streams. In International Symposium on Intelligent Data Analysis (pp. 40-53). Springer, Cham.

23. Couso, I., Borgelt, C., Hullermeier, E. and Kruse, R., 2019. Fuzzy sets in data analysis: From statistical foundations to machine learning. IEEE Comput-ational Intelligence Magazine, 14(1), pp.31-44.

24. Peng, Y. and Flach, P., 2001. Soft discretization to enhance the continuous decision tree induction. Integrating Aspects of Data Mining, Decision Support and Meta-Learning, 1(34), pp.109-118.

25. Chen, Y.L., Wang, T., Wang, B.S. and Li, Z.J., 2009. A survey of fuzzy decision tree classifier. Fuzzy Information and Engineering, 1(2), pp.149-159.

26. Dong, M. and Kothari, R., 2001. Look-ahead based fuzzy decision tree induction. IEEE Transactions on fuzzy systems, 9(3), pp.461-468.

27. Wang, X. and Borgelt, C., 2004, July. Information measures in fuzzy decision trees. In 2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No. 04CH37542) (Vol. 1, pp. 85-90). IEEE.

28. Wang, X., Liu, X., Pedrycz, W. and Zhang, L., 2015. Fuzzy rule based decision trees. Pattern Recognition, 48(1), pp.50-59.

29. Grossi, V., Romei, A. and Turini, F., 2017. Survey on using constraints in data mining. Data mining and knowledge discovery, 31(2), pp.424-464.

30. Afify, A.A., 2016. A fuzzy rule induction algorithm for discovering classification rules. Journal of Intelligent & Fuzzy Systems, 30(6), pp.3067-3085.

31. Jin, C., Li, F. and Li, Y., 2014. A generalized fuzzy ID3 algorithm using generalized information entropy. Knowledge-Based Systems, 64, pp.13-21.