Pattern Anal Applic (2017) 20:991–1004
DOI 10.1007/s10044-016-0542-2

THEORETICAL ADVANCES

A framework to induce more stable decision trees for pattern classification

Zahra Mirzamomen[1] · **Mohammad Reza Kangavari**[1]

Received: 14 November 2015 / Accepted: 23 February 2016 / Published online: 17 March 2016

Abstract Decision tree learning algorithms are known to be unstable, such that small changes in the training data can result in highly different output models. Instability is an important issue in the context of machine learning which is usually overlooked. In this paper, we illustrate and discuss the problem of instability of decision tree induction algorithms and propose a framework to induce more stable decision trees. In the proposed framework, the split test encompasses two advantageous properties: First, it is able to contribute multiple attributes. Second, it has a polylithic structure. The first property alleviates the race between the competing attributes to be installed at an internal node, which is the major cause of instability. The second property has the potential of improving the stability by providing the locality of the effect of the instances on the split test. We illustrate the effectiveness of the proposed framework by providing a complying decision tree learning algorithm and conducting several experiments. We have evaluated the structural stability of the algorithms by employing three measures. The experimental results reveal that the decision trees induced by the proposed framework exhibit great stability and competitive accuracy in comparison with several well-known decision tree learning algorithms.

Keywords Stability · Decision tree · Split measure · Split test

✉ Zahra Mirzamomen
mirzamomen@iust.ac.ir

[1] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

1 Introduction

Decision tree learning algorithms are one of the most widely used algorithms for the classification task. They can build high precision models from the labelled training instances effectively and efficiently. Moreover, the comprehensibility of the output model is an excellent property of the decision tree learning algorithms. On the other hand, decision tree learning algorithms are unstable [2, 14, 25]. An unstable learning algorithm is a highly data dependent algorithm, such that small changes in the training data can result in highly different output models.

Instability arouses suspicion about the validity of the output model, especially in domains such as medicine, where the goal is: "extracting knowledge from the model". The decision tree hypothesis may change heavily because of small changes in the training data. "The users view the learning algorithm as an oracle. Obviously, it is difficult to trust an oracle that says something radically different each time you make a slight change in the data" [28]. The consequence of such behaviour would be this concern: is the output model an integration of the information extracted from the training data or it is an artefact reacting to the training instances?

The rest of the paper is organized as follows: we first illustrate the instability problem which has motivated this research and then, we discuss about the causes of this issue and our approach for addressing this problem. Section 2 reviews the related works. Section 3 introduces the proposed framework. For the illustration purpose, Sect. 4 presents a new decision tree learning algorithm which complies with the proposed framework. Section 5 empirically evaluates the stability and the performance of the presented algorithm by comparing it with four

representative decision tree learning algorithms. Section 6 gives the conclusion.

1.1 Preliminaries of decision trees

Decision tree is a hierarchical model for supervised learning, which is composed of internal decision nodes and terminal leaves. "Each internal node m implements a test function $f_x(x)$ with discrete outcomes that label the branches" [29]. When an input x reaches to an internal node, the outcome of $f_x(x$ is calculated and the corresponding branch is taken. Throughout this paper, we have used split test in order to refer to this test function.

In univariate decision trees, a single attribute is employed at the internal nodes of the decision tree as the split test. This attribute is selected based on a split measure. Information gain is a well-known split measure based on information theory, which is used in several decision tree learning algorithms [18, 33]. In the following section, we illustrate the instability problem in the case of decision trees which employ this split measure. Meanwhile, although there may be small differences in the sensitivity of different split measures to the training data, but it is shown that the effect of this choice on the stability is not significant [7].

Let S be the set of the training instances at node N. The entropy of S is given by:

$$\text{Entropy}\ (S) = -\sum_{i=1}^{m} p_i log_2(p_i) \tag{1}$$

Where m is the number of classes and p_i is the probability that an arbitrary instance in S belongs to class C_i, which is estimated by $\frac{|C_{i,S}|}{|S|}$. Now, if the instances in S be partitioned into v disjoint subsets based on some attribute A, the entropy of the resulted partitions is given by:

$$\text{Entropy}_A(S) = \sum_{j=1}^{v} \frac{|S_j|}{|S|} \times \text{Entropy}(S_j) \tag{2}$$

Then, information gain is defined as the difference between the original entropy and the new entropy obtained after the split:

$$InfoGain(A) = Entropy(S) - Entropy_A(S). \tag{3}$$

In the case of using information gain as the split measure, information gain is calculated for every attribute and the attribute with the highest information gain is chosen as the splitting test for a node.

1.2 Motivating examples

Figure 1 illustrates the instability problem in univariate decision trees (which use information gain as the split measure) on a hypothetical 2-class classification problem. The vertical dashed line shows the split test at the root of the decision tree built on the black instances, while the horizontal line shows the split test at the root of the decision tree built on the black instances plus the one instance shown in red. In other words, on the original training data shown in black, the split test at the root of decision tree would be "Attribute 1" with 0.65 as the splitting-value, which becomes "Attribute 2" with 0.64 as the splitting-value by adding the training instance shown in red,. Therefore, we can see that even a single change in the training data can result in a highly different decision tree with a different split test (decision attribute) at the root.

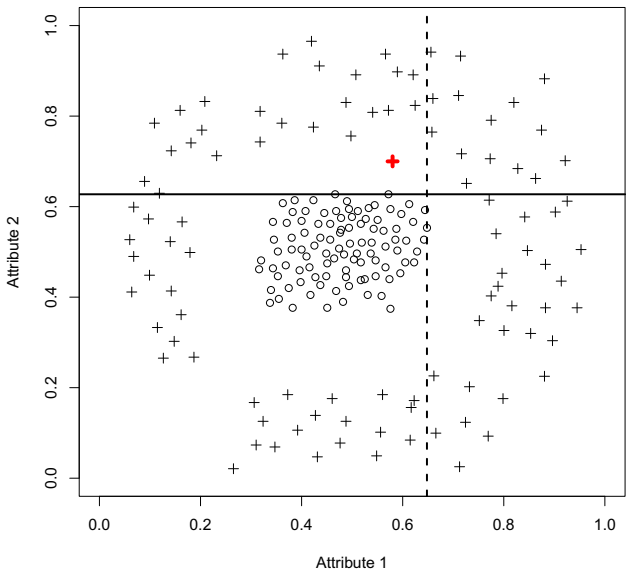


Fig. 1 Illustrating the instability problem in traditional decision tree learning algorithms on a hypothetical classification problem. The split test at the root node of the decision tree changes from the *vertical line* to the *horizontal line* by a single change in the input dataset, e.g. adding the instance shown boldface in *red* (color figure online)

Figure 2 illustrates the instability problem in the traditional decision tree learning algorithms on the Iris dataset. Iris dataset is commonly used in the literature. As it is shown, the split test at the root node has to be selected randomly, because "Petal Length" and "Petal Width" attributes have the same information gain. In other words, the split test at the root node of the decision tree may change on different runs of the algorithm, even when no change is occurred in the training data.

1.3 Discussion

In the traditional univariate decision trees, such as ID3 [18], C4.5 [19] and their descendants [16, 33], the split test is a single attribute. Consequently, the attributes compete for being selected as the split test at the internal nodes in

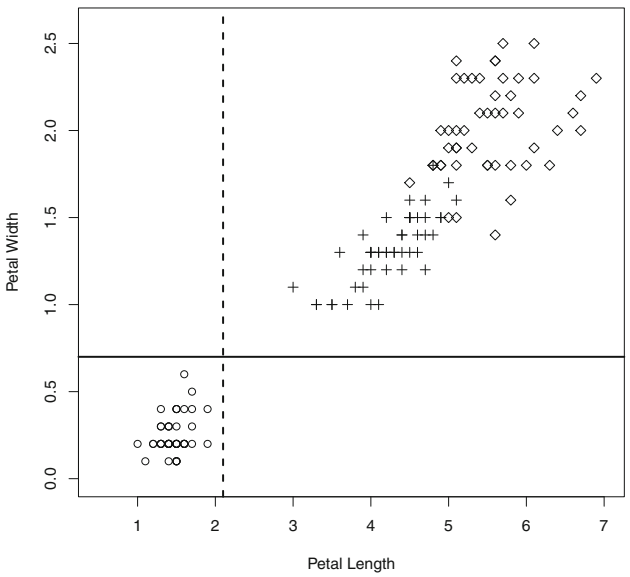


Fig. 2 Illustrating the instability problem in the traditional decision tree learning algorithms on Iris dataset. The split test at the root node of the decision tree has to be selected randomly between the "Petal Length" (the *dashed line*) and the "Petal Width" (the *solid line*), because the information gain of these attributes are equal ($\approxeq 0.9183$)

such decision tree learning algorithms and hence, when some attributes have close merit based on the heuristic split measure function, a little change in the training data may result in selecting a different winner. This brings about instability, i. e. a little change in training data may cause a highly different output model.

On the other hand, the divide-and-conquer nature of the decision tree learning algorithms and the way they select the split tests (by optimizing for all outcomes of the test) make them sensitive to the training instances [27]. It is shown that adding additional instances to the training set will have a strong influence on the selection of tests at all levels of tree [27]. That 's because even a single change in the training instances can change the split test at a node (even the root node) and such a change cascades to all of its descendant subtrees, because it alters the distribution of the instances that reach to them. The consequence is that each training instance can have a global influence on the learnt hypothesis.

Meanwhile, in a study on comparing the effect of adding new instances to the training set (via windowing) on the decision tree learning algorithms and the decision rule learning algorithms, it is experimentally shown that decision rule learning algorithms exhibit more stability than decision tree learning algorithms [27], which is justified by the fact that in the separate-and-conquer rule learning, adding new instances to the training set only affects the decision rules that cover those instances and hence, the parts of the hypothesis that have already been learned well would not be affected [27].

These deliberations assert that "to make the system stable, some kind of localization is necessary" [39].

1.4 Our approach

This paper proposes a new framework for inducing decision trees. In the proposed framework, the split test has advantageous properties which provides higher stability: first, it can contribute multiple attributes, which alleviates the race between the competing attributes to be installed at an internal node. Second, it is non-monolithic and hence, it can be designed such that the effect of the training instances on it be local. These properties help to prevent major changes in the decision tree structure due to small changes in the training data.

2 Related works

Decision trees are well-studied classification models and there are many works in the literature concerning about their fundamental aspects [4, 5, 8, 11, 16, 24], including "how a node should be split?", "how the effectiveness of a split should be evaluated?", "when a node should be turned into a leaf node?" and "how the class label should be assigned to the new instances which arrive at the leaf nodes?". However, despite the extensive usage of decision trees, the important problem of instability is usually overlooked and hence, there are few works in the literature concerning about this issue.

2.1 On improving the stability

Breiman [2] has been the first researcher who has identified decision trees as unstable predictors. Moreover, he studied the undesirable consequences of instability and he proposed bagging [1] as a way to improve stability. Bagging refers to a voting scheme in which multiple models are combined. Making an ensemble of decision trees, i. e. employing the voting scheme, is the common approach to improve stability. However, the induced ensemble models are incomprehensible and complex. Breiman asserted on the need for exploring the possibility of stabilization of algorithms by changing their structure instead of voting [2].

Turney [22] has propounded the importance of stability of decision trees too. He has discussed the relationships between the stability, the predictive accuracy and the bias of a learning algorithm and finally, he has asserted to consider stability as a criteria for evaluation of the classification algorithms.

Kohavi et. al. [13] have proposed an approach for improving stability of decision trees based on employing the voting schema in some special internal nodes, called

option nodes. "An option node is like an *or* node in *and–or* trees" [13], in which multiple split tests (attributes) are combined using the majority vote approach. However, It is shown that decision trees created by the option decision tree approach can be very large, such that the number of nodes in an option decision tree is comparable with the total number of nodes in an ensemble of decision trees.

Dannegger [26] has employed the re-sampling technique at the internal nodes to stabilize the selection of the split attribute, such that the attribute with the maximum number of winnings on a user-defined number of bootstraps at the node-level is selected as the split attribute.

Zimmermann [25] has studied employing an ensemble of rules within each internal node of the decision tree in order to improve stability. However, he reported that the experimental results did not advocate the expected improvement in the structural stability.

Dwyer [7] has studied the problem of instability of decision trees in the passive and the active learning settings. He proposed a new split measure, called DKM [7], and he investigated the effect of using this criteria on the stability of C4.5 algorithm [19]. He showed experimentally that to employ the DKM splitting criterion instead of information gain improves the stability of C4.5 on some datasets, but it does not show a significant improve in the general case [7].

2.2 On quantifying stability

An important issue for studying stability is how to quantify it. Some indicators are proposed in the literature for quantifying stability of classification algorithms [7, 17, 22, 25]. Two types of stability is defined in the literature: semantic stability and structural stability [7]. "Semantic stability measures the degree to which two classifiers make the same predictions, whereas structural stability measures the similarity between particular structural properties of two trees" [7]. Structural stability is more difficult to achieve than semantic stability, because when two hypotheses are structurally similar, they would produce the same predictions, but the converse is not true [7].

Turney [22] has proposed a method for quantifying semantic stability based on a measure of the agreement between the concepts built on samples from the same distribution. This metric which is called expected agreement [22] is used as a measure for semantic stability of decision trees in the consequent researches [7]. Paul et. al [17] have proposed the stability of the class prediction as an indicator for semantic stability of the classification algorithms.

Zimmermann [25] has considered the low variance in the size and depth of the trees in the cross-validation setting as an indicator for structural stability. Dwyer [7] has defined region stability metric as a novel measure for structural stability of decision trees. Briand et. al. [30] have proposed a similarity measure based on the placement of the attributes at the internal node of the decision tree to assess the stability of decision trees.

3 The proposed framework

Algorithm 1 presents the proposed framework. Given a data set D with attribute set $A = \{a_1, a_2, \ldots, a_n\}$ and a target attribute y, the goal is to induce an optimal decision tree ($DT(D)$) with minimum generalization error. However, as inducing an optimal decision tree from a given dataset is NP-complete, heuristic methods are used for solving the problem [28].

The existing top-down decision tree learning algorithms grow the decision tree in a greedy manner by recursively selecting an attribute as the split test [18, 19], which maximizes the heuristic split measure, such that: $a_i = argmax_{a_i \in A}\ H(a_i)$. However, as discussed before, when some attributes have close merit based on the heuristic split measure, trying to select the single best attribute causes such a race-condition which makes the whole tree structure very sensitive to the training instances.

Algorithm 1 The proposed framework for inducing stable decision trees.

function InduceTree(Dataset D, Parameters P) returns a tree;
  Let $A = \{a_1, a_2, \cdots, a_n\}$ be the attributes of D;
  Let $H(.)$ be a heuristic split measure function to measure the merit of the attributes;
  Let $R(.)$ be a feature selection mechanism;
  Let *S_Constructor* be the split test constructor;
  Let *L_Constructor* be the leaf-labeller constructor;

  if *Stop_Criteria*(D) **then**
    Build a model ξ using *L_Constructor* on D
    Return a Leaf node with ξ as the label assigner
  end if

  Select the contributing attributes $A' \subseteq A$ according to the $H(.)$ and $R(.)$
  Build a model ψ using *S_Constructor*(P) on D, by considering the attributes in A'
  Partition the instances in D using ψ
  For each partition D_i of D
    $Tree_i = InduceTree(D_i, P_i)$;

  Create *Root* as an internal node with ψ as the split test and all $Tree_i$s as the children
  return the *Root* as the induced Tree.
end function

In the proposed framework, we suggest to employ more general split tests in which the number of contributing attributes is not limited to be just one. In other words, we propose to use split tests with the ability to contribute multiple attributes, such that if several attributes achieve close merit (based on the split measure), there is no need to choose only one of them. As already discussed and illustrated, to choose a single attribute when there are some attributes with close merits would be a fragile and unstable decision.

More specifically, depending on the distribution of the instances at an internal node, the policy is as follows: if there is a single attribute which is definitely better than the

other attributes regarding the split measure, it would be considered as the single attribute to contribute in the split test, but if several attributes have close merits, we can allow all of them to contribute.

The remaining question is how to split a node by contributing multiple attributes? In the univariate decision trees which use a single attribute as the split test, the feature space is divided into axis-parallel regions by orthogonal hyperplanes. Oblique decision trees [40] employ multivariate split tests based on a linear combination of the attributes ($\Sigma_{i=1}^{n} e_i a_i + e_{n+1} > 0$), which cause the feature space to be partitioned linearly by the means of non-orthogonal hyperplanes. However, such a split test is monolithic and hence, it does not comply with our proposed framework and moreover, finding the best linear combination of the attributes is too computational expensive [40].

In the proposed framework, we have suggested to employ non-monolithic split tests based on multiple attributes. The polylithic structure of the split test provides several advantages: First, the split test can be designed such that the effect of the training instances on it be local, i. e. a training instance can not change the split test totally, but it may alter some parts of it. Second, such split tests can split the feature space non-linearly, which would result in less complex decision trees, especially for the bent complex classification problems [23, 31].

One way for having such non-monolithic split tests is to train a discriminative model (on the training instances that have reached to the node) in order to learn to determine the branch that an instance should be sent through. For this purpose, various machine learning classification and clustering methods can be considered. This approach leads to a hybrid classification model.

To make the split tests, we have considered a constructor, *S_Constructor*, which constructs a discriminative model based on the training instances that have reached to a node, using the attributes in A' as the variables. However, as such a constructor may have several parameters, the parameters (P) should be passed to this constructor. This enables tuning the constructor parameters at different levels of the decision tree. Afterwards, the constructed model should determine the branch that an instances should be sent through, as is done by the classical split tests.

As is suggested in the state-of-the-art studies [12, 32], we have considered using functional leaf nodes in the proposed framework. Therefore, we have considered another constructor, *L_Constructor*, which constructs a discriminative model based on the training instances that have reached to the leaf nodes. The induced discriminative model would be embedded in the leaf nodes and is used to assign the class label to the new instances in the classification phase.

Nevertheless, several aspects of the proposed framework should be made explicit by the decision tree learning algorithms which are about to be designed in accordance with this framework, including:

1. The way of selecting the contributing attributes (line 11). This can be conducted by employing suitable optimization techniques or heuristic functions.
2. The nature of the split tests at the internal nodes and its corresponding constructor (lines 5 and 12), along with the way of tuning the parameters of the constructor at each level of the decision tree (line 14).
3. The function to be used at the leaf nodes and its corresponding constructor (line 6).

Recently, novel hybrid decision tree learning algorithms with special split tests are designed [23, 31], which despite their distinct design motivations, they can be adapted to comply with the proposed framework. Fuzzy rule-based decision tree (FRDT) is a decision tree which employs fuzzy rules as the split test at the internal nodes [23]. On the other hand, Fuzzy Min-Max Decision Tree (FMMDT) is a decision tree learning algorithm, which employs a fuzzy min-max neural network as the split test at the internal nodes [31].

In both of the above algorithms, the split test is based on multiple attributes and it is not monolithic. More specifically, FMMDT employs split tests which are composed of several hyperboxes, while FRDT employs split tests which are composed of several fuzzy rules. Moreover, FMMDT provides the localization of the effect of the training instances on the split test because each training instance (x_i, y_i) only affects the nearest hyperbox in the feature space which corresponds to the same class (e. g. y_i) and has no effect on the other hyperboxes. The locality of the effect of the instances on the decision rules is already discussed in Sect. 1.3. In Sect. 4, we will discuss more about the FMMDT learning algorithm and we extend it to comply with the proposed framework.

Nevertheless, one of the outstanding benefits of traditional decision tree models over the other learning models (e.g. neural networks) is their simple hierarchical structure which is comprehensible by the domain expert. Moreover, because of the hierarchical structure of the tree, the traditional splits (which are based on solely a single attribute) provide a way of judging about the importance of the attributes for the classification task.

The proposed framework not only retains the possibility of inference about the importance of the attributes, but also improves it. This claim can be justified as follows: consider a classification task in which there are two correlated attributes, A and B, with competitive merit. Figure 3 presents a simple example of such a task. In classical decision trees, when one of these attributes (e.g. A) is selected at the

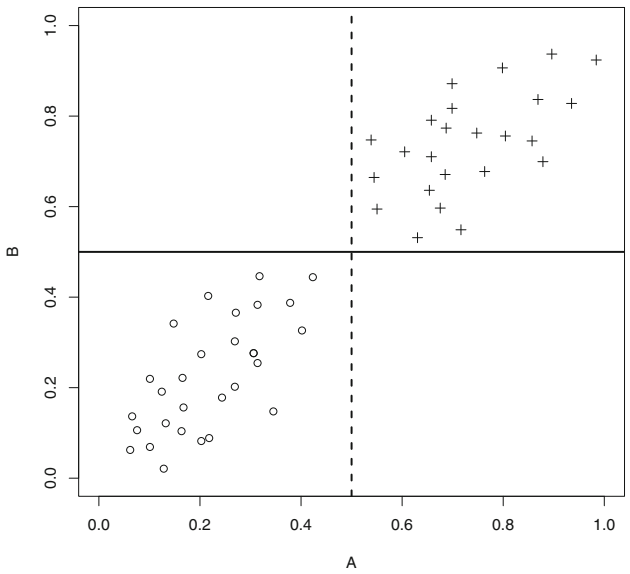


Fig. 3 A synthetic dataset with two correlated attributes with the same merits. In traditional decision trees, once A is selected as the split test, B would never appear in the corresponding subtree (and vice versa)

Lth level of the tree for the first time, the other attribute (e.g. B) would never be selected in the next levels. For example, if A be selected at the root of the tree, B would not be used in the decision tree model at all and hence, it may be considered as a poor and non-informative attribute for the classification task. In the proposed framework, this problem does not occur by contributing both attributes at the split test.

However, the comprehensibility of the model highly depends on the nature of the split test, e.g. the model build by the *S_Constructor*. Moreover, there is a trade-off between the number of contributing attributes in the internal nodes and the complexity of the internal nodes.

4 An illustrative example algorithm

In this section, we have presented a new decision tree learning algorithm which complies with the proposed framework. This algorithm is based on our previous research which contributed to introduce the FMMDT [31] as a hybrid decision tree learning algorithm. We avoid detailed description of the FMMDT algorithm, which can be found in [31].

In FMMDT, the split test is a classification model based on the fuzzy min-max neural networks. The split test is trained on the instances that have reached to the internal node. The granularity of the fuzzy min-max neural networks at the internal nodes decreases as a function of the depth of the decision tree. This enables the model to learn the boundaries between the classes [31]. It is shown experimentally that FMMDT can induce smaller decision trees with lower depth in comparison with the traditional decision trees [31].

In this paper, we have extended the FMMDT learning algorithm by adding a feature selection mechanism at the internal nodes. In the original FMMDT learning algorithm, all of the features are contributed in the split test. In this paper, we introduce a feature selection mechanism which enables detecting the features which are close to each other regarding the split measure. Our policy to improve the stability of decision tree is to contribute the best attribute along with all the other attributes close to the best attribute in the split test at the internal nodes. We have used a statistical test called Hoeffding bound [34] for the feature selection purpose, so we have called this new algorithm FMMDT-HB.

4.1 Selecting the contributing attributes at internal nodes

As stated before, our proposed policy to improve the stability of decision trees is to employ split tests in which not only the best attribute, but also the attributes which are close to it, are contributing. In this paper, we have used a heuristic based on the Hoeffding Bound [33, 34] statistical test to determine the features which are close to the best attribute regarding the split measure.

Hoeffding bound is a well-known statistical test which is commonly used for selecting the best attribute at the internal nodes of the state-of-the-art incremental decision tree learning algorithms. Assuming that G is the split measure to be maximized (e.g. information gain), the hoeffding bound can be computed as:

$$\epsilon = \sqrt{\frac{R^2 ln\left(\frac{1}{\delta}\right)}{2N}} \tag{4}$$

In which R is the range of the values that G can take, N is the number of training instances and δ is a user-defined confidence value.

Let X_a be the feature with the highest observed G after seeing N training instances, X_b be the second-best feature and ΔG be difference between the observed values. If $\Delta G > \epsilon$, then hoeffding bound guarantees that X_a remains the best attribute even after seeing additional training instances, with probability $1 - \delta$ [33, 34].

According to the above, if the difference between the first-best feature and the second-best (or another) feature does not exceed the hoeffding bound, then the difference between these features is not statistically significant. Therefore, such a feature which is currently the best one, can not be confidently considered as the feature which is really the best one. According to this discussion, we have designed the following heuristic to select the contributing

features (A'), out of the original set of features (A) at the internal nodes:

- The best feature (e.g. the feature with the highest information gain) always contributes in the split test.
- The features F that satisfy $\Delta G_F < \epsilon$ are considered to have close merit to the best feature and hence can contribute in the split test, in which ΔG_F is the difference between the value of the split measure for F and the value of the split measure for the best attribute (X_a).

Therefore, we have: $1 \leq |A^{'}| \leq |A|$. However, as discussed in Sect. 3, there is a trade-off here: as the number of attributes that contribute in the split test increases, the complexity of the split test also increases and hence, it becomes more difficult to comprehend the model. So, it may seem rational to somehow limit the number of attributes that are allowed to contribute in the split test. Therefore, we have designed a parameter, L, which puts a limitation on the maximum number of attributes that are allowed to contribute. We have investigated the effect of this user-defined parameter on the stability of the decision trees in our experiments.

4.2 The split tests

In FMMDT, an adapted fuzzy min-max neural network model [21, 31] is employed at the internal nodes as the split test. This discriminative model satisfies all the above discussed advantageous properties to improve the stability. First, it can contribute as many attributes as desired, such that the number of attributes determines the dimensionality of the hyperboxes in the model. Second, each training instance can affect the model locally, depending on:

- It's class label, and
- It's position in the feature space.

More specifically, a training instance (x_i, y_i) may affect the model in one of the following ways:

1. No effect, if the instances is located inside an existing hyperbox with class label y_i in the feature space.
2. Expanding an existing hyperbox (*hb*), if *hb* belongs to the same class (y_i) and it is close enough to the instance, such that it can be expanded to include it.
3. Creation of a new hyperbox with the same class (y_i), otherwise.

4.3 The leaf function

FMMDT employs a naive bayes classifier at the leaf node of the decision tree. It is shown that decision trees which use functional leaf nodes usually outperform the decision trees in which the leaf nodes are simply labelled with the majority-class [12].

4.4 Illustrating the stability

In this section we illustrate the stability of the presented algorithm in comparison with the classical univariate decision trees on the same datasets that are used in Sect. 1.2. Figure 4 represents the division of the feature space at the root of the tree induced by the FMMDT-HB algorithm on the training data shown in black, which would not be changed by adding the instance shown in red. This reveals the merit of the proposed framework regarding the stability. Also, Fig. 5 represents the split test at the root of the tree induced by FMMD-HB on iris dataset, which does not include any random component.

On the other hand, in the extreme case of being near the boundary of two different classes in the pattern space, adding a new training instance can either alter the borders of a single hyperbox of the split test or create a new hyperbox and hence, it can not change the split test globally. This property (i.e. the locality of the effect of the instances on the split test) prevents major changes in the decision tree by small changes in the training data and hence, it brings about more stability.

5 Experimental analysis

To verify the stability and the performance of the presented algorithm, we have conducted several experiments. We have extensively used the WEKA [10] and MOA [38]

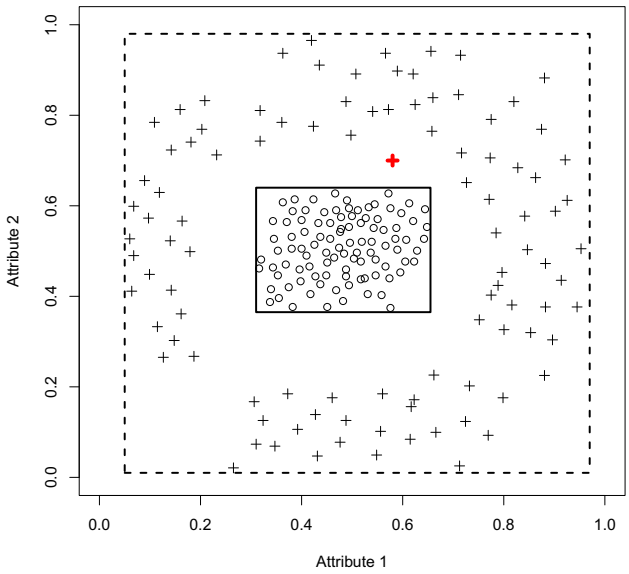


Fig. 4 Illustrating the stability of the adapted FMMDT based algorithm, as an example algorithm which complies with the proposed framework. The split test at the root node of the tree is not affected by adding the instance shown boldface in *red* (color figure online)

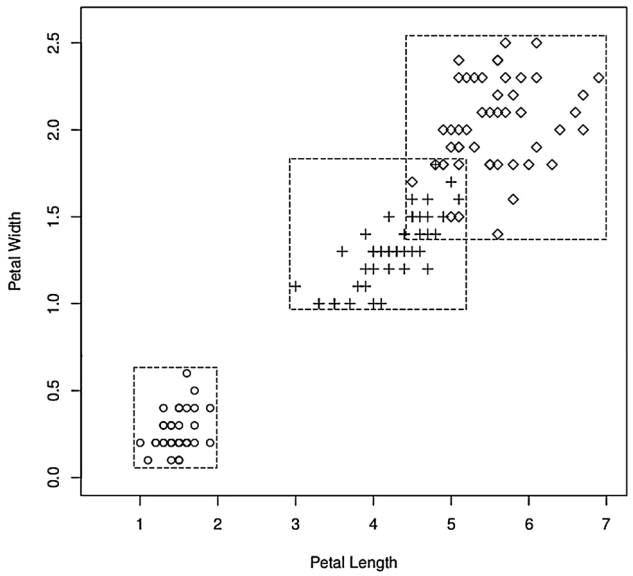


Fig. 5 Illustrating the stability of the adapted FMMDT based algorithm on the iris dataset. The split test at the root node of the tree is stable

frameworks to conduct the experiments. We have used the Weka implementation of the following decision tree learning algorithms for the comparison purpose: C4.5 [19], Naive Bayes Tree (NBT) [12], Simple Cart (SC) [3] and Best First Tree (BFT) [20]. We have set the value of the parameters of the algorithms to the default value in their Weka implementation.

The FMMDT-HB algorithm is implemented in Java under the MOA framework. For all the experiments, we have used the following parameter settings for FMMDT-HB: $q = 0.2$, $n_{min} = 10$, $\delta = 0.0001$ and $r = 0.5$. All the experiments are conducted on a 1.7GHz Core i5 Linux box with 8 GB of main memory. In the experiments, we have set different limits on the maximum number of contributing attributes in the internal nodes (as described in Sect. 4.1). We have named the resulted algorithms as FMMDT-HB-LX, in which X is a positive non-zero number which indicates the limit.

We have compared the presented algorithm with four well-known decision tree algorithms regarding both the stability and the performance. As suggested by [6], to compare multiple classifiers on multiple datasets based on average ranks, the significance of the observed differences in the metrics is tested with Friedman test [9]. The average rank of a classifier is calculated by taking the average of the ranks of it on different datasets. The rank of a classifier (based on a measure, e.g. accuracy or error) on a single dataset would be a number between 1 and the total number of classifiers in the experiment. When the null hypothesis is rejected, we use the posthoc Nemenyi test [6].

Table 1 Data sets: origin, name, number of attributes (NumAtts.), number of classes (Classes.), and number of instances (NumIntancess.)

No	Data set	NumAtts	Classes	NumInstances.
1	Balance	4	3	625
2	BreastCancer	9	2	683
3	Column3c	6	3	310
4	Haberman	3	2	306
5	Ionosphere	34	2	351
6	Iris	4	3	150
7	Liverdisorder	6	2	345
8	Newthyroid	5	3	215
9	SatelliteImage	36	6	4435
10	Sonar	60	2	208
11	Vehicle	18	4	846
12	Waveform	21	3	5000

In the following, we explain about the experimental setup and methodology, performance metrics, the datasets and the results.

5.1 Data sets

Table 1 presents the datasets used in our experiments. These datasets are among the commonly used data sets in the machine learning literature obtained from the UCI machine learning repository [15].

5.2 Stability analysis

We have conducted several experiments to assess the structural stability of the algorithms. We have evaluated the structural stability of the algorithms based on two methods: in the first set of the experiments, we have followed the method suggested by [25]. In the second one, we have adapted the similarity measure proposed in [30] to the case of decision trees with multiple contributing attributes in the internal nodes. The analysis of the results comes in the following subsections.

5.2.1 The variance measurements

Zimmermann [25] has considered the low variance in the size and the depth of the decision trees in the cross-validation setting as an indicator for structural stability. Following this criteria, we have done ten-fold cross-validation experiments for calculating the variance of the size and the depth of the induced decision trees. We have repeated each experiment ten times and hence, the reported results are the average results over ten runs of ten-fold cross-validation.

Table 2 The average and standard deviation of the depth of the trees

Data set	FMMDT-HB-L2	BFT	C4.5	SC	NBT
Balance	3.91 ± **0.22**	9.28 ± 0.52	8.92 ± 0.68	6.74 ± 1.46	4.48 ± 2.01
BreastCancer	4.01 ± **0.49**	6.33 ± 1.38	5.72 ± 1.29	4.86 ± 1.33	1.46 ± 1.73
Column3c	4.0 ± **0.0**	6.47 ± 1.86	7.44 ± 1.34	4.66 ± 1.46	4.5 ± 1.12
Haberman	6.53 ± **0.74**	4.38 ± 3.75	1.78 ± 1.39	1.5 ± 2.51	0.64 ± 1.08
Ionosphere	1.0 ± **0.0**	5.11 ± 1.65	8.47 ± 1.28	3.48 ± 1.82	4.82 ± 0.9
Iris	2.23 ± **0.4**	3.67 ± 0.67	3.64 ± 0.56	3.25 ± 0.89	1.15 ± 1.32
LiverDisorder	6.99 ± **0.03**	7.01 ± 2.23	8.4 ± 1.39	5.03 ± 2.22	2.78 ± 1.44
Newthyroid	2.53 ± **0.5**	4.32 ± 1.05	5.38 ± 0.75	4.21 ± 1.38	2.78 ± 1.69
SatelliteImage	6.99 ± **0.03**	14.22 ± 1.36	20.63 ± 2.12	11.67 ± 1.03	0.24 ± 0.76
Sonar	5.96 ± **0.56**	4.2 ± 1.87	6.89 ± 0.86	3.12 ± 2.20	3.98 ± 0.78
Vehicle	6.02 ± **0.06**	11.7 ± 1.85	13.56 ± 2.0	12.7 ± 3.68	8.93 ± 1.54
Waveform	7.0 ± **0.0**	13.81 ± 1.69	16.78 ± 1.42	9.64 ± 1.65	5.43 ± 4.40

The smallest standard deviation is shown in bold

Table 3 The average and standard deviation of the size of the trees. The size is considered to be the total number of the nodes in the decision tree

Data set	FMMDT-HB-L2	BFT	C4.5	SC	NBT
Balance	16.64 ± **0.87**	128.78 ± 25.35	81.2 ± 8.26	55.1 ± 34.68	18.36 ± 10.02
BreastCancer	13.03 ± **1.48**	27.66 ± 8.68	20.36 ± 4.59	15.66 ± 5.95	5.16 ± 5.08
Column3c	17.0 ± **0.0**	25.48 ± 11.8	24.36 ± 5.59	12.56 ± 6.33	16.7 ± 4.98
Haberman	20.59 ± 2.23	18.84 ± 9.76	4.8 ± 2.7	4.76 ± 6.76	3.14 ± **1.89**
Ionosphere	4.0 ± **0.0**	15.7 ± 6.63	27.12 ± 4.07	9.9 ± 6.53	15.12 ± 3.15
Iris	9.92 ± **1.06**	9.26 ± 1.92	8.28 ± 1.13	7.5 ± 1.78	3.78 ± 2.72
LiverDisorder	21.97 ± **0.09**	42.76 ± 25.35	48.96 ± 12.35	24 ± 17.91	7.5 ± 3.20
Newthyroid	11.12 ± **1.96**	12.98 ± 2.72	15.24 ± 1.97	11.42 ± 3.68	7.42 ± 3.83
SatelliteImage	49.93 ± **0.22**	251.4 ± 50.61	391.04 ± 17.45	123.52 ± 34.22	5.28 ± 13.53
Sonar	18.88 ± **1.68**	17.02 ± 7.59	28.48 ± 3.19	11.88 ± 7.33	13.7 ± 2.54
Vehicle	31.1 ± **0.32**	123.62 ± 17.61	136.9 ± 19.43	91.84 ± 41.02	56.14 ± 11.29
Waveform	29.0 ± **0.0**	333.72 ± 91.46	539.62 ± 31.31	119.56 ± 54.31	41.94 ± 37.67

The smallest standard deviations are shown in bold

Table 2 shows the average and the variance of the depth of the decision trees induced by the rival algorithms in the cross-validation setting. It can be observed that for all of the cases, the variance of the depth of the trees induced by FMMDT-HB with $L = 2$ is the smallest value, in comparison with the other algorithms.

Table 3 shows the average and the variance of the size of the trees induced by the algorithms. The size is considered to be the total number of the nodes in the decision tree. Again, it can be observed that for almost all of the cases, the FMMDT-HB algorithm with $L = 2$ presents the smallest variance in the size of the tree.

These results indicate that FMMDT-HB algorithm provides a higher degree of structural stability in comparison with the rival algorithms, because it presents the minimum change in the structure of the model by the changes in the training data.

5.2.2 *The similarity measurements*

Briand et. al. [30] have proposed new similarity measures to assess the stability of classification trees. The basic version of the measure only takes into account the internal nodes of the tree, because they determine the hierarchy of splits which would be translated into a decision making process [30]. According to this measure, two decision trees are considered to be identical if they employ the same split tests at the corresponding internal nodes.

They have also proposed adapted versions of this measure to nominal and continuous attributes. However, as the FMMDT-HB algorithm only supports continuous attributes, we have used the basic version of the measure which works with continuous attributes. Let A_1 and A_2 be the decision trees. Suppose that $t_0, t_1, \ldots, t_T$ are the various internal nodes of the trees, numbered in descending order from the root node and from left to right and T is the number of the internal nodes of the trees. According to [30], the dissimilarity between the two decision trees A_1 and A_2 is calculated as:

$$d(A_1, A_2) = 1 - \sum_{t=0}^{T} q_t S_t$$

in which S_t $(0 \leq S_t \leq 1)$ is the similarity between A_1 and A_2 at node t, and $q_0, q_1, \ldots, q_t$ are user-supplied non-negative weights summing to 1. Therefore, $0 \leq d(A_1, A_2) \leq 1$ and $d(A_1, A_2) = 0$ if the trees are identical. If at a node t, the split test of A_1 be x_k with σ_1 as the cut-point and the split test of A_2 be $x_{k'}$ with σ_2 as the cut-point, the S_t is the similarity between A_1 and A_2 at node t [30]:

$$S_t = I\{k = k^{'}\}\left(1 - \frac{|\sigma_1 - \sigma 2|}{range(x_k)}\right)$$

in which, $I\{k = k^{'}\} \in \{0, 1\}$ is the indicator of $x_k = x_k^{'}$. Therefore $S_t = 0$ when the splits are made on different attributes, and $S_t = 1$ when the splits are made on the same attributes with the same cut-point, and $S_t \in (0, 1)$ otherwise.

In this paper, we have adapted this measure as follows: first, we have extended the $I\{k = k^{'}\}$ for the splits based on more than one attributes, in which x_k and $x_{k'}$ are the set of the attributes that contribute in the split test of the node t of A_1 and A_2 respectively. Second, we have neglected the difference in the cut–points to simplify the calculations. More specifically, we have defined S_t as:

$$S_t = I^{'}\{k = k^{'}\}$$

where $I^{'}\{k = k^{'}\}$ is a real–valued function, which can take a value in the range [0, 1], defined as:

$$I^{'}\{k = k^{'}\} = \frac{|x_k \bigcap x_{k'}|}{MAX(|x_k|, |x_{k'}|)}$$

For example, suppose that A_1 splits on $x_k = \{a_1, a_2\}$ at node t. If A_2 split on the same set of attributes, then $I^{'}\{k = k^{'}\} = 1$. If A_2 split on $x_{k'} = \{a_1, a_3\}$, then $I^{'}\{k = k^{'}\} = 0.5$ and if A_2 split on $x_{k'} = \{a_3, a_4\}$, then $I^{'}\{k = k^{'}\} = 0$. Moreover, if A_2 split on $x_{k'} = \{a_1, a_2, a_3\}$, then $I^{'}\{k = k^{'}\} = \frac{2}{3}$.

To calculate the dissimilarity measure for each algorithm, we have again employed the 10-fold cross-validation setting. In a single run of 10-fold cross-validation for an algorithm, ten decision trees are created. For each algorithm, we have computed the dissimilarity measure on each pair of these ten trees and hence, we get $(10 \times 9)/2 = 45$ measurements. Afterwards, we calculate the average of these measurements. We have repeated this method ten times for each algorithm and the reported results are the average over all the runs.

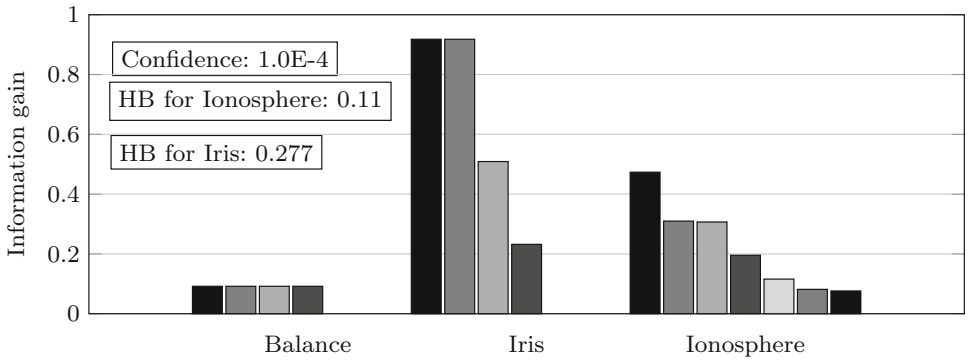
Table 4 presents the average value of the dissimilarity measure for the rival algorithms. Figure 7 depicts these results. The results show that the value of the dissimilarity measure is large (close to 1) for the normal univariate decision trees on most of the datasets, which reveals the high difference in the structure of such decision trees due to the changes that cross-validation method imposes on the training data. On the other hand, the results show that the FMMDT-HB algorithm achieves the smallest value for the dissimilarity measure on almost all datasets. The results also show that the value of the dissimilarity measure for FMMDT-HB is close to zero on most of the cases. The last row of Table 4 presents the average value of the dissimilarity measure cross the datasets. These results advocate the superiority of FMMDT-HB algorithm in producing more similar decision trees, regarding the split attributes at the internal nodes.

The high values of the dissimilarity measure for normal univariate decision trees in Table 4 can be explained by investigating the merit of the attributes of each dataset. Figure 6 shows the merit of the top attributes based on

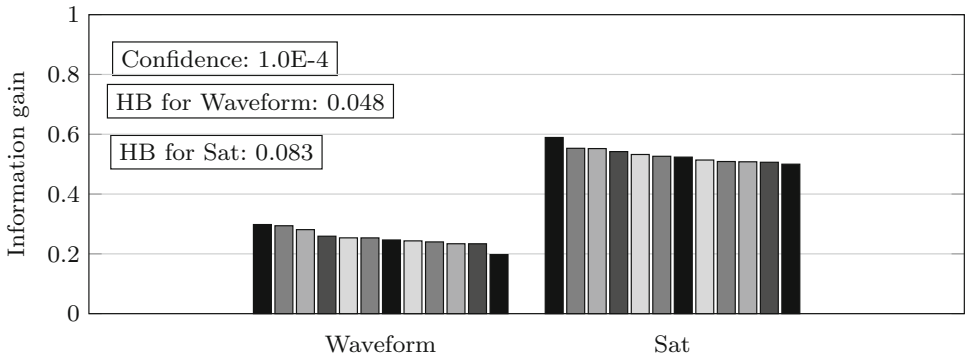
Table 4 The average and standard deviation of the value of the dissimilarity measure for the decision tree learning algorithms

Data set	FMMDT-HB	BFT	C4.5	SC	NBT
Balance	**0.09 ± 2.0e–3**	0.91 ± 4.0e–4	0.90 ± 1.9e–4	0.91 ± 4.7e–4	0.78 ± 1.3e–2
BreastCancer	**0.37 ± 4.9e–4**	0.79 ± 3.9e–3	0.65 ± 7.1e–3	0.73 ± 2.3e–3	0.38 ± 2.3e–2
Column3c	**0.05 ± 1.4e–4**	0.83 ± 4.7e–4	0.78 ± 3.7e–3	0.69 ± 9.9e–3	0.86 ± 1.6e–3
Haberman	**0.02 ± 8.2e–4**	0.51 ± 2.8e–2	0.34 ± 2.1e–2	0.2 ± 1.7e–2	0.29 ± 3.2e–2
Ionosphere	**0.01 ± 1.0e–3**	0.73 ± 1.9e–4	0.82 ± 1.2e–3	0.57 ± 1.6e–2	0.97 ± 3.5e–4
Iris	**0.04 ± 1.9e–3**	0.59 ± 6.4e–3	0.29 ± 4.9e–3	0.5 ± 1.0e–2	0.38 ± 4.4e–2
LiverDisorder	**0.06 ± 1.0e–3**	0.84 ± 1.0e–3	0.89 ± 1.4e–3	0.79 ± 2.3e–3	0.91 ± 7.4e–4
Newthyroid	**0.05 ± 3.7e–5**	0.76 ± 1.9e–3	0.72 ± 6.4e–3	0.83 ± 2.1e–3	0.69 ± 2.2e–2
SatelliteImage	0.28 ± 1.5e–3	0.87 ± 2.2e–4	0.93 ± 8.2e–5	0.78 ± 5.5e–4	**0.01 ± 7.9e–4**
Sonar	**0.01 ± 3.7e–4**	0.86 ± 7.0e–4	0.86 ± 7.0e–4	0.82 ± 5.9e–4	0.95 ± 1.9e–4
Vehicle	**0.14 ± 2.1e–4**	0.83 ± 9.2e–4	0.81 ± 4.9e–4	0.83 ± 1.1e–3	0.96 ± 9.4e–5
Waveform	**0.22 ± 2.7e–3**	0.95 ± 9.7e–5	0.96 ± 3.1e–5	0.89 ± 5.0e–4	0.58 ± 7.6e–3
Average	0.11	0.79	0.75	0.71	0.65

Decision trees with smaller values of the measure show more similarity and are shown in bold



(a) The vertical axis shows the value of information gain.



(b) The vertical axis shows the value of information gain.

Fig. 6 The best split attributes based on information gain for five datasets. The attributes are sorted according to the information gain in descending order. According to the hoeffding bound (computed with confidence = 1.0E–4 and using all the instances in each dataset), the following attributes are considered to have close merit: all the 4 attributes for Balance dataset, the first 2 attributes for Iris dataset, the first attribute for Ionosphere dataset, the first 6 attributes for Waveform dataset and the first 11 attributes for Sat dataset

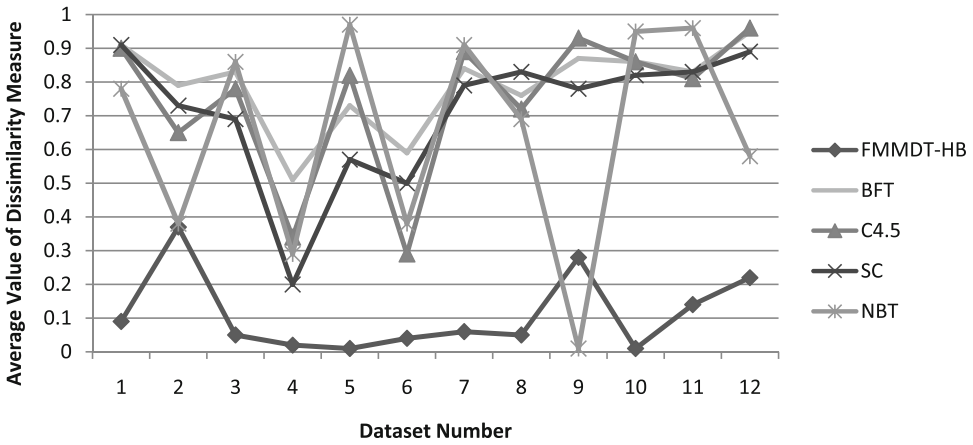


Fig. 7 The average value of the dissimilarity measure obtained for the five decision tree learning algorithms on the 12 datasets used in the experiments

information gain for the Balance, Iris, Ionosphere, Sat and Waveform datasets.

Figure 6 shows that the four attributes of the Balance dataset have identical information gain and hence at the root of the decision tree, the single best attribute either has to be chosen randomly or the changes in the training data (e.g. removing one fold out of the ten fold of the data in the cross-validation setting) may cause to select a different one of them, each time, as the winner. As discussed earlier, such a change at the root of decision tree would change the distribution of the instances that are sent to the children and hence, it may cause changes in the subsequent levels of the tree. Similarly, the Waveform and the BreastCancer datasets include several identical and near-identical attributes. For the Iris dataset, the competition is between two attributes with identical information gain and hence, the

Table 5 The average and standard deviation of the value of the dissimilarity measure for FMMDT with different settings

Data set	FMMDT-HB-L1	FMMDT-HB-L2	FMMDT-HB-L3	FMMDT-HB-L4	FMMDT-HB
Balance	0.75 ± 1.4e−3	0.52 ± 1.6e−3	0.36 ± 5.2e−4	**0.09 ± 2.0e−3**	**0.09 ± 2.0e−3**
BreastCancer	**0.31 ± 1.8e−3**	0.48 ± 8.6e−4	0.39 ± 3.2e−4	0.38 ± 3.4e−4	0.37 ± 4.9e−4
Column3c	**0.0 ± 0.0**	0.05 ± 1.3e−4	0.05 ± 1.4e−4	0.05 ± 1.4e−4	0.05 ± 1.4e−4
Haberman	0.04 ± 3.1e−3	0.16 ± 3.5e−3	**0.02 ± 8.2e−4**	**0.02 ± 8.2e−4**	**0.02 ± 8.2e−4**
Ionosphere	**0.0 ± 0.0**	0.01 ± 1.0e−3	0.01 ± 1.0e−3	0.01 ± 1.0e−3	0.01 ± 1.0e−3
Iris	0.39 ± 3.7e−3	0.12 ± 3.1e−3	0.08 ± 2.1e−3	**0.04 ± 1.9e−3**	**0.04 ± 1.9e−3**
LiverDisorder	0.6 ± 2.0e−3	0.41 ± 9.6e−4	0.33 ± 2.1e−3	0.19 ± 1.3e−3	**0.06 ± 1.0e−3**
Newthyroid	0.44 ± 8.7e−3	0.42 ± 1.8e−3	0.13 ± 1.1e−3	**0.04 ± 6.0e−4**	0.05 ± 3.7e−5
SatelliteImage	**0.0 ± 0.0**	0.24 ± 1.4e−3	0.06 ± 1.5e−3	0.18 ± 2.2e−3	0.28 ± 1.5e−3
Sonar	0.56 ± 3.1e−3	0.64 ± 1.7e−3	0.57 ± 4.2e−4	0.56 ± 1.1e−3	**0.01 ± 3.7e−4**
Vehicle	0.07 ± 1.7e−3	0.04 ± 5.6e−4	**0.0 ± 0.0**	0.005 ± 9.7e−5	0.14 ± 2.1e−4
Waveform	0.07 ± 1.2e−2	**0.0 ± 0.0**	**0.0 ± 0.0**	0.15 ± 4.8e−4	0.22 ± 2.7e−3

The smaller values indicate more similarity and are shown in bold

dissimilarity results are smaller in comparison with the Balance dataset.

We have investigated the effect of the parameter L (which sets a limit on the maximum number of contributing attributes at an internal node) on the stability of the FMMDT-HB decision trees. Table 5 presents the value of the dissimilarity measure for the FMMDT-HB algorithms with L varying from 1 to 4. The last column of this table presents the results for the FMMDT-HB, in which hoeffding bound is used to determine the contributing attributes, but no limitation is imposed on the maximum number of contributing attributes.

In Table 5, there are some cases for which the dissimilarity value does not change by increasing the L. These observations can be explained depending on the dataset. For example, FMMDT-HB-L4 and FMMDT-HB present identical dissimilarity results on the Balance and Iris datasets. The reason is that these datasets contain only four attributes and hence, setting $L = 4$ is equivalent to the no-limit case. However, FMMDT-HB-L2, FMMDT-HB-L3, FMMDT-HB-L4 and FMMDT-HB present identical dissimilarity results for the Ionosphere dataset (which contains 34 attributes). Here, the reason can be explained by the difference between the merit of the attributes and the hoeffding bound. As shown in Fig. 6, the difference between the first-best attribute of the Ionosphere dataset and its second-best attribute exceeds the hoeffding bound and hence, the merit of the first attribute is significantly better than the second one. In such situations, increasing the L does not have any effect, because L only limits the attributes which pass the initial hoeffding bound test.

Figure 8 depicts the data presented in Table 5. It can be seen that there is no general correlation between the L and the similarity measure values, such that the increase in the L may cause increase or decrease in the dissimilarity measure, depending on the dataset. However, we can see that the diagram of the FMMDT-HB (e.g. when no limitation is imposed) has less variance in comparison with the others.

The above results reveal an interesting point. Overall, it may be expected that the trees become more identical as the L increases. However, as the results show, this is not always the case.

For the Balance dataset, which includes 4 attributes with identical merit (according to the Fig. 6), the similarity between the decision trees increases by increasing the L. This is rational because as L gets closer to 4, the need for randomly discarding some of those qualified attributes decreases.

On the other hand, for the Waveform and Sat datasets, which are large datasets and include many attributes with close information gain (Fig. 6), the experimental results show that increasing the L sometimes makes the results worse. The experiments show that for the Waveform and Sat datasets, the best similarity results are achieved by setting $L = 2$ (or 3) and $L = 1$ respectively. These observations can be justified as follows.

Let G be a specific split measure to be maximized and A, B and C be the top three attributes of a dataset respectively based on G, which are sorted in descending order of G, such that all of them have close merit based the hoeffding bound. Let D and E be the next attributes of this dataset, with significantly less merit in comparison with A based the hoeffding bound. In such a situation, the F-gap between the attributes, $\Delta(A, B) = F(A) - F(B)$, affects the similarity results obtained by varying the L. For example, suppose $\Delta(A, B) = \Delta(B, C) = \Delta(C, D) = \Delta(C, E) = \eta$ and hoeffding bound is equal to 2η. In this case, setting $L = 3$ does not necessarily improve the similarity measure in

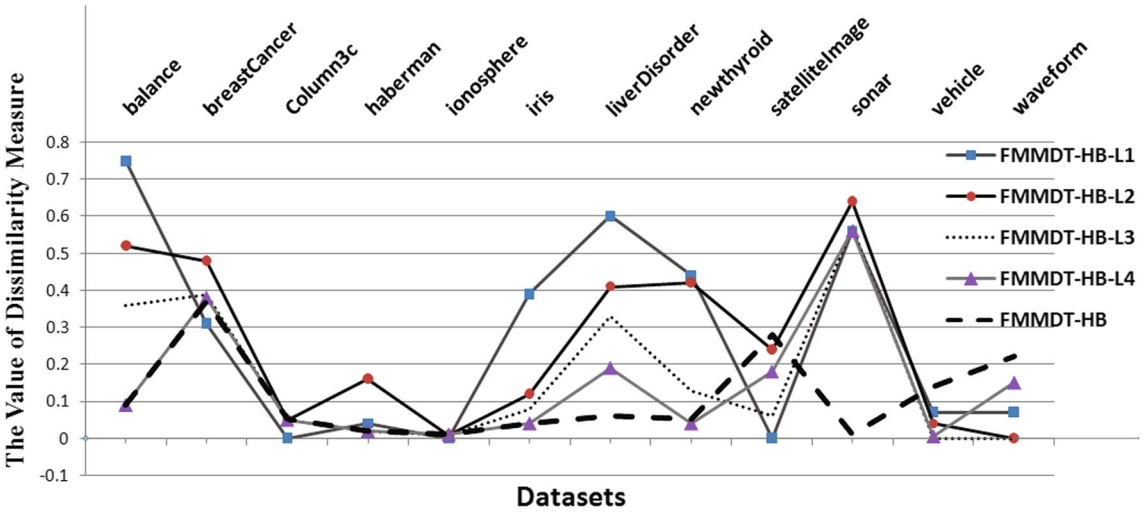


Fig. 8 The value of the dissimilarity measure for the FMMDT-HB algorithms with varying levels of L on each dataset

Table 6 The number of misclassified instances and standard deviation for the algorithms

Data set	FMMDT-HB-L2	BFT	C4.5	SC	NBT
Balance	**90.1 ± 8.9**	132.4 ± 7.2	139.2 ± 8.1	131.8 ± 7.2	150.4 ± 7.4
BreastCancer	31.9 ± 5.8	34.0 ± 3.6	32.6 ± 5.3	32.3 ± 3.1	**21.5 ± 4.7**
Column3c	**54.1 ± 4.0**	63.0 ± 7.6	57.8 ± 8.3	57.4 ± 7.5	59.8 ± 5.1
Haberman	**76.9 ± 8.7**	88.0 ± 7.8	90.4 ± 7.9	88.0 ± 9.6	86.2 ± 8.3
Ionosphere	64.9 ± 2.4	35.2 ± 4.0	**34.1 ± 3.5**	37.2 ± 4.3	35.9 ± 3.6
Iris	**7.8 ± 2.0**	8.2 ± 1.9	8.1 ± 2.9	8.8 ± 2.9	10.0 ± 2.8
LiverDisorder	123.7 ± 8.8	**115.1 ± 10.5**	119.0 ± 7.1	116.9 ± 11.4	125.3 ± 9.6
Newthyroid	**11.1 ± 2.9**	16.9 ± 3.0	14.7 ± 3.2	18.8 ± 4	14.0 ± 5.1
SatelliteImage	672.59 ± 22.9	616.6 ± 19.2	623.4 ± 22.4	**604.8 ± 21.2**	790.7 ± 34.5
Sonar	63.5 ± 8.1	61.8 ± 4.8	58.6 ± 7.5	61.4 ± 6.1	**49.8 ± 5.5**
VCehicle	402. ±	246.3 ± 10.7	**236.0 ± 10.8**	244.8 ± 10	247 ± 14.7
Waveform	957.2 ± 19.0	1159.6 ± 23.0	1164.8 ± 35.3	1145.5 ± 31.1	**939.4 ± 28.4**
Average rank	**2.67**	3.29	3.0	2.87	3.17

The best results are shown in bold

comparison with $L = 2$, because as the changes in training data may cause the C to win over the B in the $L = 2$ setting, it also may cause the D or E to win over the C in the $L = 3$ setting.

5.3 Performance analysis

Table 6 shows the total number of misclassified instances and standard deviation for the rival algorithms on twelve datasets. The results are the average results over ten runs of ten-fold cross validation. Comparing the average rank of the rival algorithms, we can see that FMMDT with $ds = 2$ achieves the best average rank among the rival algorithms. However, comparing the average ranks with Friedman test, we obtain $\chi_F^2 = 1.15$ and $F_F = 0.27$ with critical value 2.58 at the 0.05 critical level and so, we can not reject the null hypothesis. Therefore we can conclude that the difference in the performance of the algorithms is not statistically significant.

6 Conclusion

In this paper we focused on the problem of instability of decision tree learning algorithms. To alleviate this problem, we proposed to employ special split tests at the internal nodes of the decision trees which satisfy two advantageous properties. We illustrated our proposed framework by introducing a complying decision tree

learning algorithm. We measured the structural stability of the proposed learning algorithm by several metrics. The results confirmed the superiority of the proposed algorithm in creating decision trees with stable structure, in comparison with four well-known decision tree algorithms. In addition, the proposed algorithm achieved the best average rank regarding the total number of miss-classified instances, but its superiority was not statistically significant.

References

1. Breiman L (1996) Bagging predictors. Mach Learn 24:123–140
2. Breiman L (1996) Heuristics of instability and stabilization in model selection. Ann Stat 24(6):2350–2383
3. Breiman L, Friedman JH, Olshen RA, Stone CJ (1984) Classification and regression trees. Wadsworth International Group, Belmont, California
4. Brodley CE, Utgoff PE (1995) Multivariate decision trees. Mach Learn 19:45–77
5. Chandra B, Kothari R, Paul P (1995) A new node splitting measure for decision tree construction. Patt Recogn 43:2725–2731
6. Demšar, J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 1–16
7. Dwyer KD (2007) Decision tree instability and active learning, Ph.D. thesis. University of Alberta, Edmonton
8. Fong PK, Weber-Jahnke J (2012) Privacy preserving decision tree learning using unrealized data sets. IEEE Trans Know Data Eng 24:353–364
9. Friedman M (1940) A Comparison of Alternative Tests of Significance for the Problem of m Rankings. Ann Math Stat 11:86–92
10. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The weka data mining software: An update. SIGKDD Explor. Newsl 11:10–18
11. Hu Q, Che X, Zhang L, Zhang D, Guo M, Yu D (2012) Rank entropy-based decision trees for monotonic classification. IEEE Trans Know Data Eng 24:2052–2064
12. Kohavi R (1996) Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid, Proceedings of the second international conference on knowledge discovery and data mining, AAAI Press, 202–207
13. Kohavi R, Kunz C (1997) Option decision trees with majority votes, Proceedings of the Fourteenth International Conference on Machine Learning, ICML '97
14. Last M, Maimon O, Minkov E (2002) Improving stability of decision trees. Int J Patt Recogn Art Intell 16:145–159
15. Lichman M (2013) UCI machine learning repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science
16. Maher PE, St.Clair D (1993) Uncertain reasoning in an ID3 machine learning framework, Second IEEE International Conference on Fuzzy Systems, 7–12
17. Paul J, Verleysen M, Dupont P (2012) The stability of feature selection and class prediction from ensemble tree classifiers, ESANN2012 Special Session on Machine Ensembles
18. Quinlan JR (1986) Induction of Decision Trees. Mach Learn 1(1):81–106
19. Quinlan JR (1993) C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco
20. Shi H (2007) Best-first decision tree learning, Master's thesis. University of Waikato
21. Simpson PK (1992) Fuzzy min-max neural networks. I. Classification. IEEE Trans Neural Networks 3:776–786
22. Turney P (1995) Technical note: Bias and the quantification of stability. Mach Learn 20:23–33
23. Wang X, Liu X, Pedrycz W, Zhang L (2015) Fuzzy rule based decision trees. Patt Recogn 48:50–59
24. Yi W, Lu M, Liu Z (2011) Multi-valued attribute and multi-labelled data decision tree algorithm. Int J Mach Learn Cyber 2:67–74
25. Zimmermann A (2008) Ensemble-trees: leveraging ensemble power inside decision trees. Discovery Science, Springer, Berlin Heidelberg, Lecture Notes in Computer Science 5255:76–87
26. Dannegger F (2000) Tree stability diagnostics and some remedies against instability. Stat Med 19:475–491
27. Furnkranz J (1998) Integrative windowing. Stat Med 8:129–164
28. Rokach L, Maimon O (2008) Data Mining with Decision Trees: Theory and Applications, World Scientific Publishing Co. Pte. Ltd, volume 69 series in machine learning and artificial intelligence
29. Alpaydin E (2010) Introduction to Machine Learning, The MIT Press, 2nd edition
30. Briand B, Ducharme GR, Parache V, Mercat-Rommens C (2009) A similarity measure to assess the stability of classification trees. Comp Stat Data Anal 53(4):1208–1217
31. Mirzamomen Z, Kangavari M (2016) Fuzzy Min-Max neural network based decision trees, Intelligent Data Analysis, to appear soon in. 20(4)
32. Gama J (2004) Functional trees. Mach Learn 55(3):219–250
33. Domingos P, Hulten G (2000) Mining high-speed data streams. Proceedi Sixth ACM SIGKDD Int Conf Know Dis Data Mining, KDD 00:71–80
34. Hulten G, Spencer L, Domingos P (2001) Mining time-changing data streams, Proceedings of the 2001 ACM SIGKDD International Conference On Knowledge Discovery and Data Mining, 97–106
35. Gama J, Rocha R, Medas P (2003) Accurate decision trees for mining high-speed data streams, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, New York, USA, 523–528
36. Hashemi S, Yang Y (2009) Flexible decision tree for data stream classification in the presence of concept change, noise and missing values. Data Mining Know Dis 19:95–131
37. Bifet A, Gavaldá R (2009) Adaptive learning from evolving data streams. Advances in Intelligent Data Analysis VIII, Lecture Notes in Computer Science, Springer, Berlin Heidelberg 5772:249–260
38. Bifet A, Holmes G, Kirkby R, Pfahringer B (2010) Moa: massive online analysis. J Mach Learn Res 11:1601–1604
39. Zhao Q (2005) Learning with data streams: an nntree based approach, Embedded and Ubiquitous Computing. In: T Enokido, L Yan , B Xiao, D Kim, Y Dai, L Yang (eds) Lecture Notes in Computer Science. Springer, Berlin, Heidelberg vol 3823, pp 519–528
40. Heath David, Kasif Simon, Salzberg Steven (1993) Induction of oblique decision trees. J Arti Intell Res 2(2):1–32