Evolutionary Intelligence (2022) 15:439–453
https://doi.org/10.1007/s12065-020-00519-0

RESEARCH PAPER

A novel approach to build accurate and diverse decision tree forest

Archana R. Panhalkar[1] · **Dharmpal D. Doye**[2]

Received: 19 February 2020 / Revised: 31 August 2020 / Accepted: 28 October 2020 / Published online: 3 January 2021

Abstract
Decision tree is one of the best expressive classifiers in data mining. A decision tree is popular due to its simplicity and straightforward visualization capability for all types of datasets. Decision tree forest is an ensemble of decision trees. The prediction accuracy of the decision tree forest is more than a decision tree algorithm. Constant efforts are going on to create accurate and diverse trees in the decision tree forest. In this paper, we propose Tangent Weighted Decision Tree Forest (TWDForest), which is more accurate and diverse than random forest. The strength of this technique is that it uses a more accurate and uniform tangent weighting function to create a weighted decision tree forest. It also improves performance by taking opinions from previous trees to best fit the successor tree and avoids the toggling of the root node. Due to this novel approach, the decision trees from the forest are more accurate and diverse as compared to other decision forest algorithms. Experiments of this novel method are performed on 15 well known, publicly available UCI machine learning repository datasets of various sizes. The results of the TWDForest method demonstrate that the entire forest and decision trees produced in TWDForest have high prediction accuracy of 1–7% more than existing methods. TWDForest also creates more diverse trees than other forest algorithms.

Keywords Decision tree forest · Random forest · C4.5 · Classification · Hyperbolic function · CART

1 Introduction

To mine useful information from massive data is a challenging field in today's era. Various data mining methods are available to analyze massive data collected from various sources. Classification is the method of finding to which group a new observation fits in. The classifier is trained using a set of data whose class is known. Many classifiers like K-Nearest Neighbor Classifier, Bayesian classifier, Rule-based classifier, decision tree and many more are used to classify data efficiently [1]. A decision tree is one of the best classifiers due to its simple interpretation and expressive quality. It is a flowchart like structure in which internal nodes are used for taking decisions to predict class indicated by leaf nodes. A decision tree is a supervised classifier that gets trained by using a training dataset and tested using unseen instances. It is used for solving classification and regression problems. A decision tree is used as a promising, nonparametric and supervised classifier for all size datasets. A small change in the data creates unstable decision trees, which might result in an entirely different tree. A decision tree follows the Greedy approach for construction of tree. Greedy approaches cannot assure to generate the globally optimal decision tree. This can be mitigated by several trees training, where the attributes and instances are randomly sampled. The main limitation of the decision tree is that it is prone to overfitting and fails to give a globally optimal solution. To tackle the given limitation of a decision tree, an ensemble of decision trees called decision tree forest is generated to avoid overfitting and finds globally optimal decisions [2]. Ensemble methods have extensive benefits in automated assessment making areas as compared to single expert decision [2]. The significant rule behind ensemble methods are for collection of weak learners can combine to generate a "strong learner".

Decision tree forest is a collection of decision trees generated from different training instances of a dataset. The main aim of collecting decision trees is to take an opinion of multiple decision trees, which leads to correct prediction of decision. In a real-time example, for the correct diagnosis of any disease, the opinion of many doctors is taken,

✉ Archana R. Panhalkar
archana10bhosale@rediffmail.com

[1] Shri Guru Gobind Singhji Institute of Engineering and Technology, Nanded, India

[2] Electronics and Telecommunication Engineering, Shri Guru Gobind Singhji Institute of Engineering and Technology, Nanded, India

so that chances of failure are very less than taking opinion from a single expert. This same concept is used in an ensemble of decision trees. It obtains the prediction from every tree in the forest and finally opts for the best solution through voting. The decision tree forest gives accurate results as compared to a single decision tree. Decision tree forest can handle hundreds of input features without any feature deletion. Researchers in literature performed variations to create decision tree forest. Every decision tree forest is qualified using two factors, namely accuracy and diversity of the forest. Each tree in the forest should be accurate and diverse. Importance and relation of accuracy and diversity are analyzed in literature [3, 4] and found that accuracy and diversity are essential factors for correct decisions. High ensemble accuracy and diversity should be balanced to create efficient decision forest [4].

In this paper, we propose a novel Tangent Weighted Decision Forest (TWDForest) building method to create accurate and diverse decision trees. TWDForest creates accurate decision trees compared to existing decision tree forest methods. It uses novel tangent weighting function, which is capable of giving accurate weight ranges. Due to these accurate weight ranges, the classification performance of decision tree forest and individual tree in the forest is increased. Another important contribution of this method is to utilize knowledge from each tree to create subsequent trees in the forest. Learning from previous trees leads to create decision trees which provide high classification capacity. As per the above discussion, the decision tree forest should be equally accurate and diverse. Diversity should not be compromised for accuracy of decision tree forest. TWDForest achieves variation (diversity) among different trees in the forest by avoiding consequent decision trees with the same root nodes. This variation in root node leads to create different decision nodes at different levels in each tree of the forest. The proposed method is analyzed on various publicly available datasets of the UCI machine learning repository [5]. The experimental results prove that TWDForest increases the accuracy of decision tree forest with improvement in diversity. The accuracy of each tree in the TWDForest is better, and each tree is different from other trees in the forest.

This paper is structured into the following sections. Section-2 gives a brief idea about various decision tree forest methods from literature along with its performance and limitations. In section-3, some basic terminologies used in the paper are revisited. The proposed method of constructing a decision tree forest is discussed in section-4. The experimental results with discussion are elaborated in section-5. Various real-time applications and implications are described in section-6. Finally, concluded with remarks of the proposed method in section-7.

2 Related work

Decision tree forest evolution started with concepts of Bagging (Bootstrap Aggregation) [6] and Boosting [7], which are simple and compelling ensemble methods. These are based on the concept that the number of weak learners can create a single strong learner. A classifier that is slightly associated with the correct classification is called a weak learner. Bagging [6] generates a new dataset by randomly selecting instances from the original dataset. The size of each newly generated dataset is the same as the original dataset. Boosting [7] also generates a new dataset by assigning weights to instances. AdaBoost [7] is a prominent basic boosting algorithm in the literature. AdaBoost works by putting extra weight on hard to classify instances and less on those instances previously classified. In the random subspace method [8], attributes are sampled in random order instead of instances as done in bagging. These features or attributes are sampled with a replacement for each decision tree learner. The random subspace method selects a subset of attributes either at tree level or at the node level. Due to this, every learner does not excess focus on attributes that emerges highly predictive in the training dataset but fall short to be as predictive outside that dataset. For this reason, these are a smart choice for applications where the quantity of features is larger than the number of training instances. The first method of random decision forests was proposed by Ho [9]. Ho created decision trees in a forest using the oblique hyperplane method. The use of hyperplanes can achieve better accuracy, and they build trees without the need for over-training. One of the key motivations is that random forest is very effectual because every tree in it is dissimilar to other trees in the forest. This condition is obtained due to the random sampling of data for the training of every individual tree. The major drawback of this method is that it fails to work if a data set has fewer features because after selecting a subset of features, very few attributes are available in each tree. So trees generated using oblique hyperplanes are not accurate and diverse.

Leo Breiman [10] extended Ho's random decision forest [9] and registered it as a "Random Forests" trademark. Random forest is a combination of bagging [6] and the random subspace method. Bagging also called bootstrap sampling is a method of generating new subsets by randomly selecting instances from the training dataset. The size of each subset is the same as the original training dataset and used to train a decision tree. As a result, an ensemble of different models is created in a random forest. It takes an average of all the predictions from these various trees. Output produced by bagging is more robust than a single tree. The random forest creates multiple decision trees and uses them together to get a more accurate and constant prediction. It creates a

tree from each bootstrap sample by using random subspace sampling. One of the main advantages of the random forest [10] is that it is most versatile, never over fit and gives better prediction accuracy than a single decision tree. It is found in the literature [11–13] that the size of the sub-space may not be evenly suitable for both low and high dimensional data sets. Cascading and Sharing method (CS4) [11] creates many decision trees. CS4 orders the attributes in descending order according to gain ratios. It creates i^{th} tree using the i^{th} best attribute. Then subtrees of i^{th} tree are created using the C4.5 decision tree algorithm [23]. The limitation of CS4 is that it generates a user input number of trees only when the number of attributes of a data set is greater than or equal to the user input number of trees. Otherwise, it builds as many trees as the number of non-class attributes in the data set. "Maximally Diversified Multiple Decision Tree Algorithm (MDMT)" [12] generates multiple trees with each tree contains a different set of attributes. All attributes that have been used in the previous tree are removed from the training data set and builds the next tree using the modified training data set. This continues until either all non-class attributes of the data set are removed or the user-defined numbers of trees are generated. The limitation of MDMT is that it is only applicable for low dimensional data sets. It is unable to build multiple trees so diversity is one of the issues of MDMT. To avoid such hyperparameter, Forest-RK [13] proposed a random selection of a subset of attributes between 1 to total attributes.

"Random Feature Weights (RFW) for decision tree ensemble construction" [14] assigns random weight within the range of 0.0 to 1.0 to each attribute. It builds a decision tree by assigning merit value (multiplication of the Gini index with random weight) to each attribute. Then it selects the attribute with the highest merit value as the splitting attribute. This algorithm allows creating a more general form of random forest. The limitation of the random feature weight algorithm is that assignment of random weights to good attributes, which increases their chance of frequently selected as the root node. So many similar trees may get generated and reduce the diversity of the forest. "Systematically developed Forest (SysFor)" [15] builds multiple trees (100 or user input number) from a data set, which causes an overfitting problem. It selects good attributes and their split points, which are found using user-defined separation and goodness threshold. After that SysFor algorithm builds decision trees by putting the good attributes at level 1 as the root attribute node. It can construct several trees equal to the number of good attributes. If a user requires more trees, then it builds more trees by using other Level 2 good attributes. The different better attributes are selected from the set of good attributes for Level 2 nodes.

Dynamic Random Forest [16] uses other boosting standards in the framework of Random Forest. Like boosting, initially, all instances of training data have the same weight. Whenever a next tree is produced, weights are assigned to entire records of the training data set. These weights are based on instances correctly classified by previously generated trees. Stratified Random Forest [17] used attributes stratified sampling for dealing with high dimensional data. The main concept in stratified sampling is that attributes are divided into two different groups. Good attributes are in one group, and bad attributes are in the other group. Attributes with high classification capacity than average classification capacity are considered as good attributes. Attributes with less classification capacity than average classification capacity are considered as bad attributes. It selects attributes randomly from both sets to create each tree in the decision forest.

Some variations of decision tree forests are proposed by Adnan et al. [18–20]. These decision forests are developed to make the forest more perfect, and the individual trees should be sufficiently diverse. One of the novel algorithms, "Forest by Continuously Excluding Root Node (Forest CERN)" [18] continuously excludes attributes that are used in the root nodes of prior trees. Forest CERN imposes unfavourable weights to the attributes so that they cannot appear in the subsequent trees. Limitation of Forest CERN is that it assigns very disadvantageous weights to the attributes of high dimensional datasets and which results in decreasing individual accuracy of tree in the forest. Usually, all trees in the forest are not equally useful to increase the ensemble accuracy of the decision forest. A large number of trees in a decision forest have high computational overhead for predicting unseen records. Therefore, it is significant to find an effective sub-forest to reduce storage and computational overhead, increase or maintain the ensemble accuracy. Decision forests can be optimized by selecting accurate and diverse trees. Adnan et. al [19]. used a genetic algorithm to optimize the decision tree forest. It selects equally accurate and diverse trees as the initial population which produces an efficient sub-forest.

"Forest by Penalizing Attributes (ForestPA)" [20] is proposed by Adnan et al. which systematically builds decision trees in the forest. ForestPA creates each tree *Ti* from one bootstrap sample *Di* of a dataset *D* by imposing more weight to nodes tested at a lower level than tested at higher levels. It calculates the merit value of each attribute in decision tree construction by multiplying the Gini index [21] with the weight assigned to the attribute in previous trees. ForestPA uses the weight assignment strategy based on the exponential function. The drawback of an exponential function is that it increases slowly initially, and later, it increases sharply. So, attributes tested at a lower level cannot get the uniform range value. The next drawback of ForestPA is that, if attribute *A* appears in i^{th} tree as the root node, then $i+2^{th}$ tree again becomes the root node in many trees. As shown

in Fig. 1, the root node of tree-1 *A* is repeated in tree-3 produces subsequent similar trees. Due to this frequent toggling of root nodes, the diversity of trees get affected and getting many subsequent similar trees.

The Extremely Randomized Trees [22] randomly selects cut points of a numerical attribute for adding more randomness for numerical attributes. Extremely Randomized Trees choose subspace default size as $\sqrt{n}$ where n is the size of the original training dataset. Like Random subspace, it is not applied to bootstrap samples. It uses the original training data set for constructing decision trees. One of the major drawbacks of all the above decision tree forest methods is that no tree will take opinion from previous trees for which samples it fails to predict so that these samples can be get utilized into the next tree for correct predictions. So this drawback and all limitations of ForestPA are overcome in the novel proposed method of Tangent Weighted Decision tree forest (TWDForest).

3 Basic concepts in decision forest

Decision forest is an ensemble of decision trees which gives better prediction accuracy and produce diverse results. Decision trees and bootstrap sampling are essential terminologies in the decision forest. These basic terminologies are explained in this section:

3.1 Decision tree

A decision tree is a flowchart structure in which leaves represent classes and internal nodes represent the testing attributes. To create a decision forest, an individual tree is created using CART [21] or C4.5 [23] algorithm. CART algorithm selects more discriminated features makes it suitable to create an efficient decision forest than C4.5 decision tree. In TWDForest, the Classification and regression tree (CART) approach [21] is used for building decision trees. The significant advantage of using the CART decision tree is that it is a nonparametric decision tree building method, so it does not rely on a particular type of input data. It is not majorly impacted by outliers in datasets. Due to all these significant properties, the CART decision tree building algorithm is used to create individual decision trees.

3.2 Bootstrap sampling

Bootstrap sampling [10] is the method of creating new datasets *Di* (bootstrap) from the original dataset *D* such that the size of bootstrap *Di* is the same as the original dataset *D*. Instances in the bootstrap sample are selected randomly from the original dataset such that some samples may select repeatedly, and some samples from the dataset are not selected at all.

As shown in Fig. 2, three bootstrap samples are generated from the original dataset, whose size is same as the original

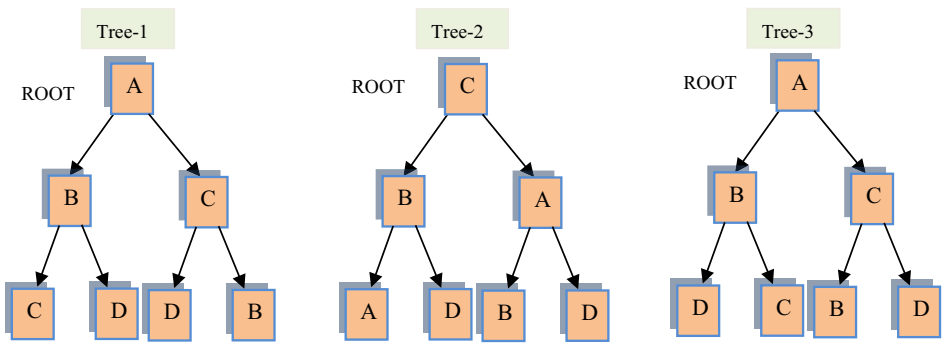

Fig. 1 Toggling of the Root node

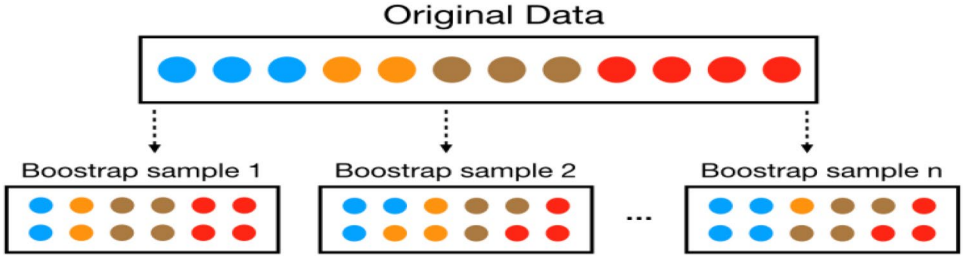

Fig. 2 Bootstrap sampling

dataset. The main advantage of using bootstrap sampling is that it increases the diversity of decision forest because a tree created from each bootstrap sample is diverse from a tree created from another bootstrap sample [24]. This leads to an increase in the diversity of TWDforest.

4 Tangent Weighted Decision tree Forest (TWDForest)

In this paper, the novel decision forest algorithm called Tangent Weighted Decision tree Forest is proposed, which overcomes limitations specified in Sect. 2. It uses a tangent hyperbolic function to calculate the weight range, which is more accurate than exponential function [25, 26].

4.1 Proposed Method

TWDForest improvement contributed to the following three proposed approaches:

a. *More accurate hyperbolic tangent function to calculate weight range*
b. *Avoid the toggling of the root node in subsequent decision trees.*
c. *Opinion of the tree to best fit the successor tree.*

TWDForest experiments show that these improvements not only increase the predictive accuracy of the forest but also increase the diversity of decision trees in the forest. Working of TWDForest is described using a block diagram in Fig. 3. The block diagram of TWDforest clearly shows the flow of TWDForest construction and novelty achieved in the proposed method. The major steps of TWDForest depicted in Algorithm-1 are described as follows.

Step 1: TWDForest reads the input training dataset D. It creates n. bootstrap samples $\{D_1, D_2, D_3, ..., D_n\}$ from the dataset D such that the size of bootstrap samples is the same as the training dataset D. The main aim of creating bootstrap samples is to create diverse decision trees. Then construct tree T_i from each bootstrap sample D_i. To compare TWDForest method with existing methods, 100 bootstrap

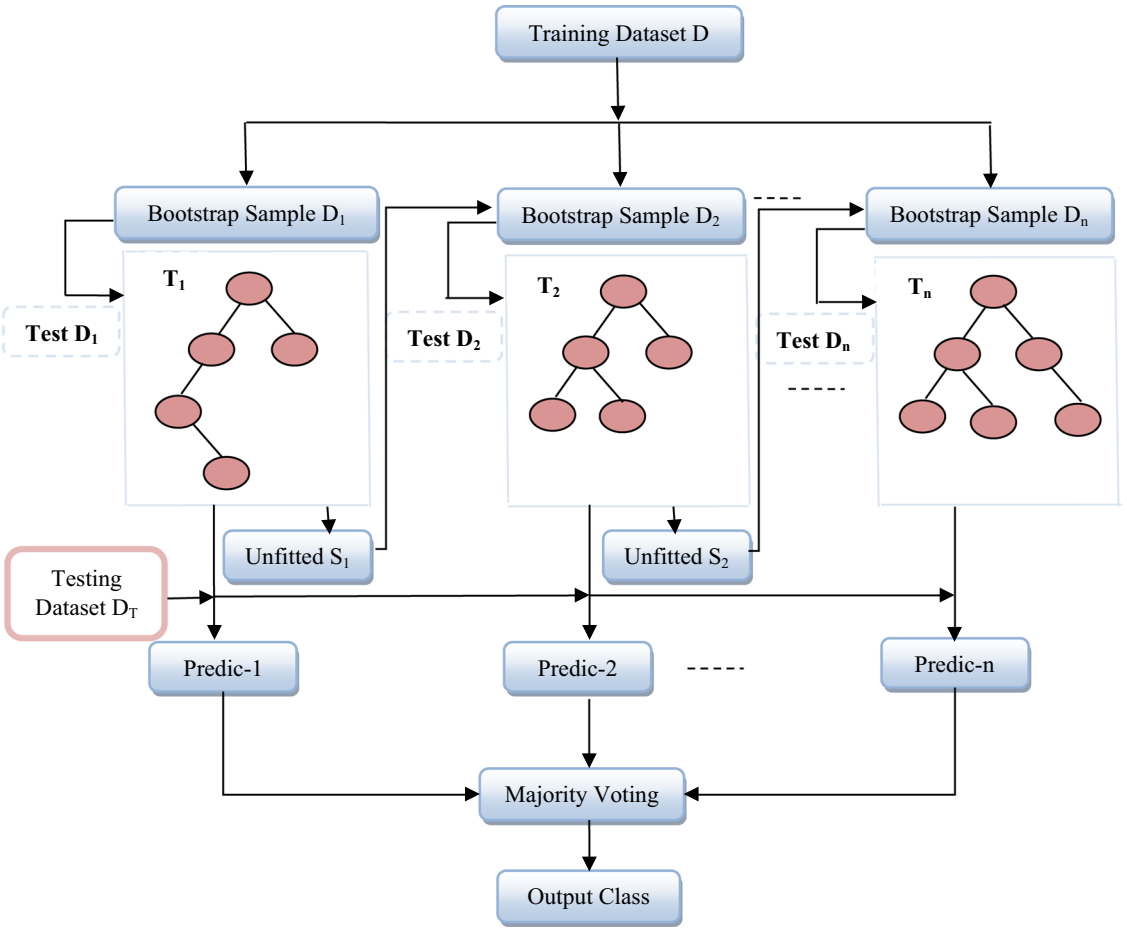

Fig. 3 Block Diagram of TWDForest

samples are generated. Bootstrap samples take nearly 63.2% original records from the training dataset.

Step 2: Initialize a uniform weight (ω) of 1.0 to each attribute in the dataset D. The weight of each attribute is multiplied with the Gini index of the attribute to create a decision tree from each bootstrap sample.

Step 3: Construct each tree T_i by calculating the merit value for each attribute A_j. Merit value is calculated by multiplying the Gini index value of an attribute and weight calculated by our novel method based on hyperbolic tangent function. After constructing the tree, it checks whether the root node of the current tree is already available in T_{i-1} or T_{i-2}. If the root node is present in any of the previous tree T_{i-1} or T_{i-2}, then select next best merit valued attribute as a root node in the current tree T_i.

From experiments, it is observed that the attribute appeared as the root node of a tree T_{i-1} again becomes the root node in the tree T_{i+1}. This is called the toggling of the root node. It shows that the same root node appears in many trees.

Algorithm- 1. Tangent Weighted Decision Tree Forest (TWDForest)

Input: Training Dataset **D**
Number of trees $\boldsymbol{\eta}$
Output: TWDForest$\leftarrow \{\boldsymbol{T_1, T_2, T_3 \ldots \ldots T_n}\}$
Begin

1. Generate $\boldsymbol{\eta}$ bootstrap samples $\{\boldsymbol{D_1, D_2, D_3, \ldots \ldots D_\eta}\}$ of Training Dataset **D**

2. Initialize
   TWDForest $\leftarrow \{\emptyset\}$
   for *each* $\boldsymbol{A_j}$ in Dataset **D** do
   $\boldsymbol{\omega(A_j) \leftarrow 1.0}$
   end

3. **for** $\forall \boldsymbol{D_i}$ **in** bootstrap samples $\{\boldsymbol{D_1, D_2, D_3, \ldots \ldots D_\eta}\}$ do
   $\boldsymbol{T_i \leftarrow Make\_tree\_CART(D_i)}$
   if root($\boldsymbol{T_i}$) = root($\boldsymbol{T_{i-1}}$) **or** root($\boldsymbol{T_i}$) = root($\boldsymbol{T_{i-2}}$) **then**
   $\boldsymbol{T_i \leftarrow Make\_tree\_CART_{new} root(D_i)}$
   endif

4. **for** *each* $\boldsymbol{A_j}$ in tree $\boldsymbol{T_i}$ do
   Update $\boldsymbol{\omega(A_j)}$ using **Weight-Range** ($\boldsymbol{WR^\alpha}$)
   Calculated using a hyperbolic tangent function.
   end

5. **for** *each* $\boldsymbol{A_k}$ in tree $\boldsymbol{T_{i-1}}$ do
   Increment $\boldsymbol{\omega(A_k)}$ using Eq. 3
   end

6. **for** *each* $\boldsymbol{Inst_k}$ in $\boldsymbol{D_i}$ do
   if Classify($\boldsymbol{Inst_k, T_i}$) == $\boldsymbol{False}$ then
   $\boldsymbol{D_{i+1} \leftarrow D_{i+1} \cup Inst_k}$
   endif
   end
   Add the Tree $\boldsymbol{T_i}$ **in TWDForest**
   $\boldsymbol{TWDForest \leftarrow}$ **TWDForest** $\cup \{\mathbf{T_i}\}$

end

end

The main drawback of the toggling root node is that it decreases the diversity of the forest. In a decision tree forest, many trees may have the same attribute as the root node. If the root node is toggling then many similar trees get created as in ForestPA [20]. So to avoid this, toggling of the root node is avoided by selecting the next best attribute as a root node in a tree.

Step 4: Calculate new weights using Hyperbolic tangent function. It is more accurate than the exponential function, which is proven by researchers in literature [25, 26]. Weight of each attribute in tree T_i is obtained randomly from Weight-Range (WR^{α}). Consider $\propto$ is the level of attribute A_j, then weight range is calculated for an attribute A_j using Eqs. 1 and 2.

$$Ub = 1 + \frac{2 * \tanh\left(\frac{-1/\alpha}{2}\right)}{1 - \tanh\left(\frac{-1/\alpha}{2}\right)} \tag{1}$$

$$lb = 1 + \frac{2 * \tanh\left(\frac{-1/\alpha - 1}{2}\right)}{1 - \tanh\left(\frac{-1/\alpha - 1}{2}\right)} + 1 \tag{2}$$

Weight of root attribute A_j is calculated by randomly selecting a value from Weight-Range (WR^{α}) 0 to Ub. Weight of any attribute with level $\alpha > 1$ is calculated using Weight-Range (WR^{α}) lower bound (lb) to upper bound (Ub). The discussion of a novelty using the hyperbolic tangent function is given in the next section.

Step 5: For each attribute $A_k \in T_{i-1}$ but not in a tree T_i, increment its gradual weight γ_k using the following Eq. 3 where h is the height of a tree T_{i-1} and ω_k is the current weight of that attribute. Due to this, the weight of those attributes which are not in the latest tree is increased. This gradual weight increment helps attributes to appear in subsequent trees.

$$\gamma_k = \frac{1.0 - \omega_k}{(h-1) - \alpha} \tag{3}$$

Step 6: Opinion of the tree created in the previous step is taken by finding unfitted instances and transferring it in successor trees. This novel contribution helps to utilize knowledge from unfitted instances instead of ignoring them. Unfitted instances are instances which fail to classify using current tree T_i. To find unfitted instances, test the samples of D_i on tree T_i and collect unclassified instances into the unfitted set S_i. This unfitted set S_i is added into the next bootstrap sample D_{i+1}, as shown in Eq. 4 to create a tree T_{i+1}.

$$D_{i+1} = D_{i+1} \cup S_i \tag{4}$$

Repeat steps 1 to 6 for each bootstrap sample D_i.

To understand the TWDForest algorithm, a flow of execution is shown in Fig. 3. As shown in the block diagram, after creating a tree T_1, the bootstrap samples D_1 are tested on this tree. The unclassified instances from a tree are added in bootstrap samples D_2 and so on. The algorithm is contributed to use a more accurate hyperbolic function to calculate weight range and it takes opinion from the previous tree and avoids toggling of root nodes.

4.2 TWDForest achievements and discussion

Novel algorithm to construct a decision tree forest called TWDForest is proposed in this paper. The efficiency of any decision forest is determined using the prediction accuracy of an individual tree of the forest, ensemble accuracy and diversity of forest [14, 18, 20]. The reasons behind the efficiency achieved by TWDForest method are discussed below.

I. Bootstrap sampling is used to create TWDForest increases the diversity of the forest. Each tree is created using one bootstrap sample which contains ≅ 63% records from training dataset D and ≅ 27% records are duplicated. The main advantage of using bootstrap is that every tree i is different from other trees, so the diversity of forest gets improved.
II. Removing the toggling of the root node is one of the contributions which avoid the use of the same root node in subsequent trees in the forest. This method increases the diversity of the forest because it avoids generating similar trees.
III. Decision tree forests created in literature [10, 14, 18, 20], fails to utilize knowledge from trees constructed in the decision forest. In our novel contribution, after building each tree, the opinion of that tree is taken to best fit the next tree. This is achieved by finding unfitted instances from a tree and adding them to the next tree. Due to this, unfitted instances get more weight and the accuracy of subsequent trees is improved. This increases the accuracy of the individual tree and ensemble accuracy of the forest.
IV. Unlike random forest, an entire attribute set is used to create each tree in TWDForest and the use of the Gini index to calculate merit value leads to choose the most efficient attributes at higher levels of trees. This will create more accurate trees than random forests. One of the main advantages of a CART tree is

that it is used for regression also aswell TWDForest also applied for regression method.

V. The next line of defence is that TWDForest uses a hyperbolic tangent method to assign weights to attributes, which are defined to give more accurate decision trees. The characteristic of the tangent method is that it produces accurate results than an exponential function. Exponential function approximation near the argument 0, will be close to 1. This causes the loss of significant figures while computing the difference value $\exp(x) - 1$ with floating-point arithmetic. So it produces a large calculation error, possibly produces meaningless results [25–27].

When experimentations are carried out with tangent and exponential functions, it is observed that when calculating weight ranges using tangent function produce accurate forest than the exponential method. Due to tangent function precision, TWDForest calculates accurate weight values for each attribute. To avoid errors produced by e^x, Eq. 5 is used, which uses tangent based hyperbolic calculation.

$$\exp m(x) = \frac{2\tanh(x/2)}{1 - \tanh(x/2)} \tag{5}$$

In this section, with all lines of defence, it is proved that the TWDForest method not only increases the accuracy of an ensemble but also it performs best for individual accuracy. Due to avoiding toggling of root nodes in subsequent trees and best fitting of each tree, both accuracy and diversity are increased. The experimental results are discussed in the following section.

5 Experimental results

5.1 Dataset description

The performance of the proposed TWDForest method is evaluated on a popular 15 standard, well known UCI machine learning repository datasets, which are freely available [5]. To prove the scalability of TWDForest, various size datasets like small, medium and large scales are used to evaluate method. Experiments are carried out on small size datasets like Wine with 13 attributes, Sonar with 60 attributes, Glass with 9 attributes, Balance scale with 4 attributes, Credit Approval with 15 attributes, Liver disorder with 6 attributes and Ionosphere with 34. Medium size datasets like Car evaluation with 6 attributes, Yeast with 8 attributes, Tic-Tac-Toe with 9 attributes, diabetes with 8 attributes and vehicle with 18 attributes are evaluated using an algorithm. Large size datasets like chess, image segmentation and nursery with 36 and 19 attributes respectively used in TWDForest evaluation. These datasets are from different domains with different attributes like numerical and categorical which proves that TWDForest is efficient and accurate for all datasets. Detail description of datasets is given in Table 1.

Table 1 Summary of datasets

Dataset	Instances	Attributes	Classes
Car Evaluation	1728	6	4
Chess	3196	36	2
Image Segmentation	2310	19	7
Nursery	12,960	8	5
Yeast	1484	8	10
Balance Scale	625	4	3
Wine	178	13	3
Sonar	208	60	2
Glass	214	9	6
Liver Disorder	345	6	2
Ionosphere	351	34	2
Credit Approval	653	15	2
Tic-Tac-Toe	958	9	2
Pima Indian Diabetes	768	8	2
Statlog Vehicle	846	18	4

5.2 Comparison of different decision forest algorithms

In TWDForest, 100 trees are created from 100 bootstrap samples of each dataset. Results are tested using a tenfold cross-validation of the dataset and best results are stressed through **bold-face**. The original dataset is divided into 10 parts. In each fold, 9 parts are used for training and the remaining one part is used for testing. Results are calculated for all folds, and then the average result of accuracy and diversity is calculated. Before applying algorithms, records with missing values are removed from a dataset. To build an individual tree in TWDForest, the CART [21] decision tree method is used. To prove the performance of proposed TWDForest, a comparison of results is done with various strategies like Random Subspace [8], Random Forest [10], Random Feature weights (RFW) [14], Forest CERN [18] and ForestPA [20]. These methods are the best algorithms to create a decision forest using weighting strategies, which creates the ensemble of decision trees.

TWDForest is tested for two measures, first is a prediction accuracy and second is diversity. While calculating the accuracy of TWDforest, the accuracy of the individual decision tree and ensemble accuracy of the forest is calculated. Ensemble Accuracy (EA) is one of the best measures to check the performance of every decision forest. To calculate ensemble accuracy, the majority voting strategy [10] is

Table 2 Comparison of individual accuracy of tree in TWDForest with other methods

Dataset	Random subspace	Random forest	RFW	Forest CERN	ForestPA	TWDForest
Car Evaluation	86.0	75.6	82.6	77.4	82.4	**87.3**
Chess	65.3	67.9	95.1	78.9	96.6	**97.8**
Image Segmentation	93.3	92.0	88.3	88.8	90.5	**94.5**
Nursery	70.4	70.2	91.7	**95.2**	91.1	94.6
Yeast	52.1	47.9	46.4	47.0	49.1	**54.7**
Balance Scale	67.4	64.9	64.8	65.0	65.2	**69.1**
Wine	**90.4**	88.9	87.9	85.1	87.9	88.0
Sonar	72.4	68.9	70.1	65.9	69.2	**75.3**
Glass	65.3	60.5	55.4	54.8	58.2	**66.7**
Liver Disorder	62.6	60.1	57.2	59.1	58.9	**67.3**
Ionosphere	88.9	88.0	88.1	86.7	87.9	**89.2**
Credit Approval	**80.9**	74.0	81.4	70.5	78.8	78.1
Tic-Tac-Toe	40.5	54.3	66.6	59.6	59.3	**68.0**
Pima Indian Diabetes	71.8	70.0	68.7	66.6	70.2	**72.8**
Statlog Vehicle	68.6	66.7	63.3	63.9	65.1	**71.2**

Table 3 Comparison of ensemble accuracy (EA) of TWDForest with other methods

Dataset	Random subspace	Random forest	RFW	Forest CERN	ForestPA	TWDForest
Car Evaluation	93.5	91.2	93.7	93.8	94.2	**96.4**
Chess	95.1	95.2	97.1	**98.0**	97.7	**98.0**
Image Segmentation	97.6	97.1	97.5	96.8	97.8	**98.4**
Nursery	94.9	95.1	97.6	97.5	**97.8**	**97.8**
Yeast	58.6	59.5	57.9	58.8	61.4	**68.6**
Balance Scale	72.2	80.5	81.1	82.3	83.0	**85.2**
Wine	97.2	97.2	97.8	98.0	**98.4**	96.9
Sonar	84.6	83.1	80.6	84.9	84.1	**85.3**
Glass	73.2	74.1	73.2	71.8	73.6	**77.3**
Liver Disorder	69.8	71.5	71.0	71.0	71.8	**75.5**
Ionosphere	93.4	93.7	93.7	94.3	94.0	**95.8**
Credit Approval	85.9	86.1	86.4	87.0	**87.1**	86.9
Tic-Tac-Toe	80.7	84.5	85.8	87.1	86.7	**89.2**
Pima Indian Diabetes	76.2	75.9	75.6	74.0	76.9	**79.1**
Statlog Vehicle	73.5	74.1	74.6	75.1	76.3	**80.4**

used to test results. Results of individual accuracy are shown in Table 2 and results of ensemble accuracy are shown in Table 3. Accuracy is calculated using a number of instances classified correctly for the given test dataset.

From Table 2, it is observed that the individual accuracy of each decision tree in TWDForest is better than other methods of decision forest. The average of individual accuracies of each tree in all 10 cross-validations is calculated and shown in Table 2. Many methods give better ensemble accuracy but fail to perform well for individual trees. So to test the performance of TWDforest, individual accuracies of each tree are compared. TWDForest gives the best performance for all individual decision trees of almost all data types. The random subspace works well for Wine and Credit approval dataset. As compared to ForestPA, TWDForest shows 1 to 9% increment on all 15 datasets. ForestCERN works well for nursery data. For all datasets, TWDForest gives 1 to 4% more individual tree accuracy than other methods.

Figure 4 shows a graphical representation of the comparison of average individual accuracies of TWDForest with other methods in percentage. From Fig. 4, it is clear that the accuracy of each tree in TWDForest is greater than other weighting decision forest methods.

Decision tree forest is found to be efficient when it is both accurate and diverse. Ensemble accuracy evaluation

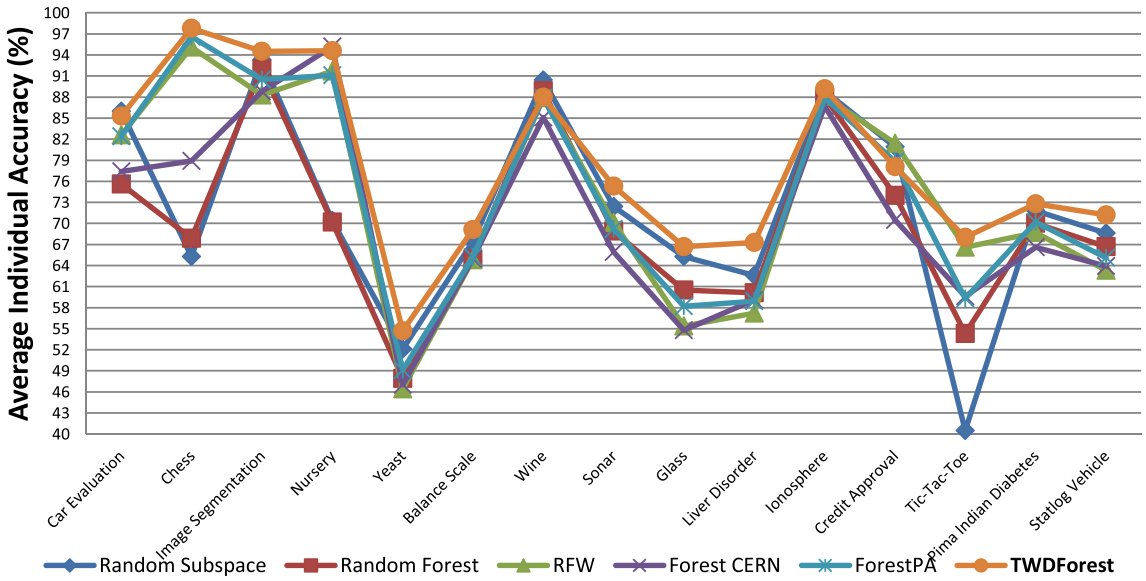

Fig. 4 Comparison of average individual accuracies of TWDForest with other decision forest method

is carried out for all 15 datasets. Ensemble accuracy from Table 3 shows that our TWDForest method gives better results for all datasets except Wine and Credit approval. The overall ensemble accuracy is better as compared with other methods. TWDForest gives the best ensemble accuracy for Yeast, Glass, Liver Disorder, Tic-Tac-Toe and Statlog vehicle dataset. All these datasets are of various sizes with different numbers of attributes. ForestPA gives better ensemble accuracy for Nursery, Wine and Credit Approval dataset. Random Subspace, Random Forest, RFW and ForestCERN increase individual tree accuracy but fails in ensemble accuracy. Decision forest is found to be promising when it gives better individual and ensemble accuracy. TWDForest performs well for both individual accuracy and ensemble accuracy of decision forest.

Diversity may not be ignored for accuracy [3, 28, 30]. To calculate the diversity of the TWDForest, Kappa statistics visualization [29] is the best measure of defining how diverse decision trees are in a forest is used. Kappa statistics used to calculate diversity between two different trees. Kappa is also called as inter-rater reliability. To calculate the diversity of each tree in the decision tree forest, the Kappa of each tree is compared with remaining all trees in the forest. Consider T_i is a single tree then Kappa of tree T_i is calculated with $T - T_i$, where T is the decision forest. Prediction of tree T_i and prediction of remaining trees $T - T_i$ combine (By majority voting) is calculated. Consider P_o is a relative observed agreement between tree T_i and the tree $T - T_i$. P_e is the hypothetical probability of chance agreement (expected agreement) between tree T_i and forest $T - T_i$. Kappa of tree T_i is calculated using Eq. 6.

$$K = \frac{p_o - p_e}{1 - p_o} \tag{6}$$

If K = 0, then a tree disagrees on each testing record with other trees. If K = 1, then a tree is agreed on each example with remaining trees. This K value indicates that Kappa value is lower for diverse trees.

In Table 4, the average individual kappa for decision forests is calculated. To calculate kappa values, tenfold cross-validation method is used. The results show that for some datasets like yeast, Balance scale, Glass, Liver Disorder and Ionosphere diversity of TWDForest is better than Random Forest, Random Subspace, Forest CERN and ForestPA. Forest CERN method produces more diverse trees for some datasets like Car Evaluation, Wine, sonar, Credit Approval and Pima Indian Diabetes. But Average individual accuracy and Ensemble accuracy of Forest CERN is less as compared to TWDForest. Decision forest with better accuracy and diversity is efficient than only accurate or diverse decision forests. TWDForest fulfills all criteria of better ensemble accuracy, individual accuracy and diversity. Other methods fail in balancing both. Forest CERN and RFW are more diverse but the accuracy of the ensemble is less. The accuracy of Random Forest and ForestPA is better but diversity is low. Fig-5 shows a comparison of individual kappa values of RFW, Forest CERN and proposed method TWDForest for different datasets. TWDForest method elaborated on all aspects of creating efficient decision tree forest. TWDForest gives better accuracy and diversity than all other weight-based decision tree forest creation methods.

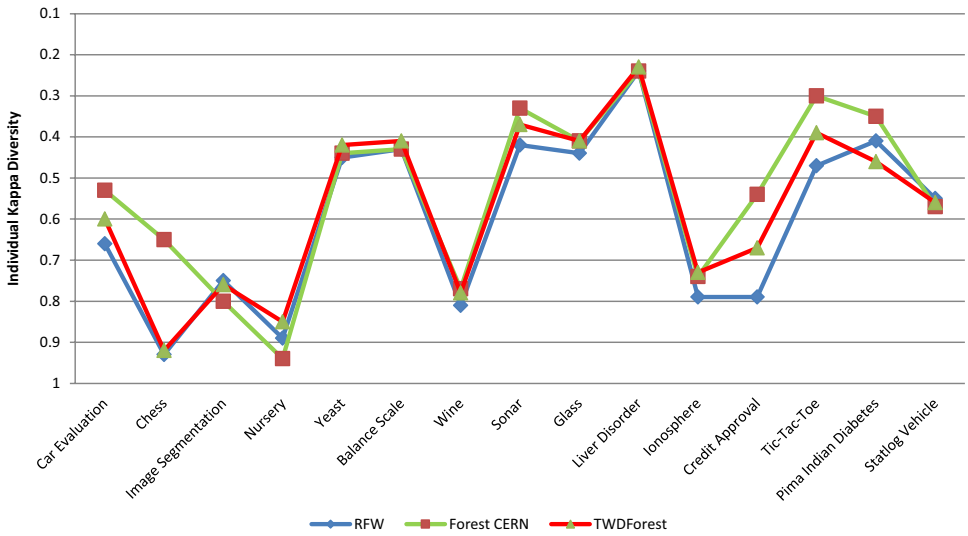

Fig. 5 Comparison of average individual kappa of TWDForest with RFW and ForestCERN

Table 4 Comparison of diversity of TWDForest with other methods

Dataset	Random subspace	Random forest	RFW	Forest CERN	ForestPA	TWDForest
Car Evaluation	0.76	0.54	0.66	**0.53**	0.65	0.60
Chess	**0.47**	0.49	0.93	0.65	0.94	0.92
Image Segmentation	0.90	0.90	**0.75**	0.80	0.81	0.76
Nursery	0.64	**0.63**	0.89	0.94	0.88	0.85
Yeast	0.60	0.47	0.45	0.44	0.47	**0.42**
Balance Scale	0.61	0.43	0.43	0.43	0.43	**0.41**
Wine	0.85	0.83	0.81	**0.77**	0.81	0.78
Sonar	0.47	0.40	0.42	**0.33**	0.42	0.37
Glass	0.61	0.53	0.44	0.41	0.45	**0.41**
Liver Disorder	0.49	0.31	0.24	0.24	0.27	**0.23**
Ionosphere	0.82	0.79	0.79	0.74	0.78	**0.73**
Credit Approval	0.75	0.64	0.79	**0.54**	0.69	0.67
Tic-Tac-Toe	**0.28**	0.31	0.47	0.30	0.40	0.39
Pima Indian Diabetes	0.58	0.49	0.41	**0.35**	0.49	0.46
Statlog Vehicle	0.67	0.62	**0.55**	0.57	0.60	0.56

5.3 TWDForest parameter evaluations

Decision tree forest performance is not much influenced by parameters. In TWDForest few parameters like number of bootstrap samples (η), number of trees in a forest, Weight-Range (WR^{α}) and gradual weight (γ_k) are important. The number of trees created in TWDforest is based on bootstrap samples. In TWDForest, 100 diverse trees are generated using 100 bootstrap samples. Results from Table 3 and Table 4 prove that TWDForest performs outstandingly with 100 trees. TWDForest is also evaluated on various bootstrap samples like 50, 60, 70, 80 and 90. Table 5 shows the ensemble accuracy of TWDForest with varied numbers of trees. It is observed that the number of trees affects the accuracy of TWDForest. If trees are more in a forest then accuracy is better for a maximum dataset.

Table 6 shows the diversity of TWDForest with varied numbers of trees. From results shown in Table 6, TWDForest achieves more diversity when it contains many trees. When 100 trees are created TWDForest consist of diverse trees for a maximum dataset.

Along with many trees in a forest, Weight-Range (WR^{α}) is one of the important parameters. It is calculated using the novel tangent function. This parameter gives varied weight to each attribute such that no two attributes get the same weight. Another important advantage of the weight range is that, while building a tree, each attribute's merit value is calculated using its Gini index and random value in weight range as shown in step-4 of Algorithm-1. To form diverse

Table 5 TWDForest ensemble accuracy (EA) in % with a varied number of trees

Dataset	50 Trees	60 Trees	70 Trees	80 Trees	90 Trees	100 Trees
Car Evaluation	94.67	94.35	95.23	96.15	96.20	**96.4**
Chess	96.02	96.54	96.58	97.80	**97.89**	98.0
Image Segmentation	97.16	97.34	97.45	97.98	98.3	**98.4**
Nursery	96.19	96.23	97.32	97.70	**97.85**	97.8
Yeast	67.26	67.90	67.93	68.27	68.41	**68.6**
Balance Scale	82.06	83.20	84.20	84.25	85.17	**85.2**
Wine	95.88	96.54	97.08	97.86	96.04	**96.9**
Sonar	85.41	82.54	84.06	84.27	84.87	**85.3**
Glass	75.10	76.37	76.54	76.45	77.01	**77.3**
Liver Disorder	73.69	74.08	74.29	74.89	75.1	**75.5**
Ionosphere	92.0	92.47	93.45	94.78	95.29	**95.8**
Credit Approval	84.27	84.98	85.94	86.05	86.69	**86.9**
Tic-Tac-Toe	86.12	87.23	87.41	88.12	88.94	**89.2**
Pima Indian Diabetes	72.11	73.21	73.68	77.85	**79.21**	79.1
Statlog Vehicle	75.94	77.82	78.37	77.21	78.16	**80.4**

Table 6 Diversity of TWDForest with a varied number of trees

Dataset	50 Trees	60 Trees	70 Trees	80 Trees	90 Trees	100 Trees
Car Evaluation	0.68	0.67	0.653	0.64	0.64	**0.60**
Chess	0.978	0.96	0.94	0.935	0.93	**0.92**
Image Segmentation	0.83	0.813	0.80	0.79	0.77	**0.76**
Nursery	0.92	0.891	0.87	0.86	**0.84**	0.85
Yeast	0.517	0.50	0.486	0.487	0.48	**0.42**
Balance Scale	0.48	0.464	0.45	0.44	0.43	**0.41**
Wine	0.89	0.86	0.82	0.81	0.80	**0.78**
Sonar	0.46	0.45	0.42	0.42	0.40	**0.37**
Glass	0.52	0.48	0.45	045	0.44	**0.41**
Liver Disorder	0.36	0.32	0.29	0.30	0.29	**0.23**
Ionosphere	0.84	0.812	0.81	0.79	0.78	**0.73**
Credit Approval	0.78	0.74	0.73	0.72	0.70	**0.67**
Tic-Tac-Toe	0.48	0.45	0.41	0.417	**0.39**	**0.39**
Pima Indian Diabetes	0.68	0.62	0.57	0.52	0.50	**0.46**
Statlog Vehicle	0.62	0.62	0.60	0.59	**0.56**	**0.56**

trees in TWDForest, the weight of attributes that are in the latest tree is increased using the weight range. Gradual weight increment (γ_k) is done for attributes that are not tested in the latest tree using Eq. 3. Due to gradual weight increment, attributes get a chance to select in subsequent trees.

5.4 Evaluating the complexity of TWDForest

The computational complexity of any algorithm is the number of resources required to run it. Particular focus is given on time required for an algorithm execution. Tuning parameters of decision tree forest may result in a computational overhead, particularly for large data sets, with hundreds and thousands of instances and attributes [31]. TWDForest is evaluated for large datasets with many attributes. To create each node in a tree, each attribute's merit value is calculated, which increases the computational complexity of an algorithm. One of the novelties in TWDForest is that after the creation of each tree, testing data is classified and misclassified instances are added in the next bootstrap samples for creating the next tree. Due to these operations, the time required to train TWDForest is more. The computational complexity of TWDForest is evaluated as follows: Consider TWDForest creates η trees, m average number of attributes in each tree and average instances in each bootstrap samples are k. The computational complexity of TWDForest is $O(\eta XcXmXkX\log(k))$. TWDforest is a nonparallel algorithm, so computational complexity cannot be optimized. Table 7 shows the time required for a different dataset for

Table 7 Comparison of Training time (in Seconds) of TWDForest with other algorithms

Dataset	Random Forest	ForestPA	TWDForest
Car Evaluation	**0.24**	13.06	16.30
Chess	**0.28**	12.26	13.40
Image Segmentation	**4.03**	28.78	34.56
Nursery	**1.35**	268.6	305.2
Yeast	**2.88**	18.34	20.41
Balance Scale	**0.24**	1.82	2.07
Wine	**0.13**	0.56	1.29
Sonar	**0.28**	3.82	4.05
Glass	**0.26**	0.95	1.03
Liver Disorder	**0.23**	1.25	1.85
Ionosphere	**0.45**	3.39	4.03
Credit Approval	**0.68**	10.49	11.12
Tic-Tac-Toe	**0.18**	6.86	7.15
Pima Indian Diabetes	**0.85**	3.17	3.81
Statlog Vehicle	**2.05**	7.77	9.01

TWDForest training. It shows that time required to train TWDForest is more than Random Forest and ForestPA which are contending algorithms. The time required for a Random Forest is minimum than ForestPA and TWDForest. After evaluating the computational complexity of TWDForest, all these 100 trees are created sequentially, so the time required to compute the algorithm is more. This is the main limitation of the TWDForest method which can overcome by applying some parallel computation on building forest.

6 Real-time applications of TWDForest

Decision tree forests are used in modeling predictions, behavior analysis and are constructed using decision trees. These are unexcelled in prediction accuracy, mathematically simple among current algorithms of classifications, and runs efficiently on large databases. As compared to a single decision tree, decision tree forest has promising prediction capacity due to the diverse nature of the ensemble classifier. Decision tree forests are so powerful and frequently used to appear in virtually every commercial and open-source software package that supports predictive analytics. From experiments, it is proved that TWDForest can be applicable for small to large scale various types of datasets with efficient performance. TWDForest could be applied to various real-time applications like banking, stock market, economy, social media analysis, E-commerce, ubiquitous analysis, weather forecasting, and in many areas. Now a day during the COVID-19 Pandemic, everyone is more attached to social media for various purposes like entertainment, knowledge, job, teaching, etc. TWDForest is one of the best algorithms to predict desired outcomes from real-world social media data [32–37]. Liu et al. [32] presented fortune tellers to predict the career path from data collected from various social media sources. TWDForest may achieve higher prediction accuracy in deciding a career path as it generates diverse ensemble classifiers.

Prediction of age and gender using online social media data [33], audience attributes of articles [34], detecting influential bloggers [35], vehicle sales prediction using sentiment analysis twitter and stock market [36], and political ideology through twitter posts [37] are some real-time applications where TWDForest can be the best predictor. Along with stated real-time applications, TWDForest may achieve better accuracy for some real-time data prediction methods like urban water quality prediction [38] and multi- appliance recognition system [39]. TWDForest is not influenced by many parameters so it a less parametric algorithm. TWDForest will perform better in the risk prediction of different medical fields. A random forest can be replaced with TWDForest is risk prediction for adverse health events [40] for achieving accurate prediction. Predicting COVID-19 patient health [41, 42] and drug discovery is an essential and significant area of research in this pandemic. TWDForest will be the best classifier in COVID-19 research.

7 Conclusion

TWDForest is a decision tree forest that is more accurate and diverse than existing decision tree forest methods. The novelty in this method is to use the Tangent hyperbolic function to construct accurate decision trees in the forest. The effectiveness of TWDForest lies in taking opinions from previous trees to construct subsequent trees, which create more accurate decision trees in the forest.

To create diverse trees in the forest, a novel idea to avoid the toggling of the root node in subsequent trees is applied. Empirical experiments are carried out on 15 well-known datasets from the UCI machine learning repository. These datasets are of various sizes and a varied number of features. The use of the CART decision tree algorithm in TWDForest makes this suitable to use for creating regression trees. It is capable to give promising results for a numeric, categorical and mixed dataset. The performance of TWDForest is compared with various popular decision tree forest algorithms like Random Subspace, Random Forest, RFW, ForestCERN and ForestPA. Results shows that TWDForest is more accurate and diverse than other decision forest methods.

TWDForest creates decision trees sequentially, which increases the computational burden. It is observed specifically for a big dataset that computation time is more. In the future, the main aim is to use parallel computation to create decision trees which will significantly decrease the

overhead of computation. Further analysis of all trees created in TWDForest will give scope to find similar trees and remove these trees to produce an optimized forest. TWDForest can be made more diverse by using other weight assignment strategies.

Acknowledgements We thank Md Nasim Adnan, Professor, Department of Computer Science and Engineering, JSTU, Jessore, Bangladesh for his guidance at all stages in implementation and evaluating the results of this research work. His support to understand concepts and research directions helped us to do research.

References

1. Han J, Kamber M (2006) Data mining: concepts and techniques. In: Morgan Kaufmann Publishers, 3rd Edition, pp 223–357
2. Polikar R (2006) Ensemble based systems in decision making. IEEE Circuits Syst Mag 6:21–45
3. Kuncheva LI (2005) Using diversity measures for generating error-correcting output codes in classifier ensembles. Pattern Recogn Lett 26(1):83–90
4. Shipp CA, Kuncheva LI (2002) Relationships between combination methods and measures of diversity in combining classifiers. Inform Fusion 3:135–148
5. Dua D, Graff C (2019) UCI machine learning repository. School of Information and Computer Science, University of California, Irvine, CA. http://archive.ics.uci.edu/ml
6. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
7. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. Proc 13th Int Conf Mach Learn 96:148–156
8. Ho TK (1998) The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 20:832–844
9. Ho TK (1995) "Random decision forests", In: Proceedings of 3rd international conference on document analysis and recognition, vol. 1, pp. 278–282. IEEE
10. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
11. Jinyan L, Huiqing Liu L (2003) "Ensembles of cascading trees", In: Proceedings of the third IEEE international conference on Data Mining (ICDM 03), IEEE, pp. 9–17
12. Hong H, Jiuyong L, Wang H, Daggard G, Shi M (2006) "A maximally diversified multiple decision tree algorithm for microarray data classification", In: Proceedings of the Workshop on Intelligent Systems for Bioinformatics (WISB 2006), Conferences in Research and Practice in Information Technology (CRPIT), vol. 73
13. Bernard S, Heutte L, Adam S (2008) "Forest-RK: A new random forest induction method", In: Advanced Intelligent Computing Theories and Applications, Lecture Notes in Computer Science, pp. 430–437
14. Maudes J, Rodriguez JJ, Osorio CG, Pedrajas NG (2012) Random feature weights for decision tree ensemble construction. Inform Fusion 13:20–30
15. Islam Z, Giggins H (2011) Knowledge discovery through SysFor: a systematically developed forest of multiple decision trees. Proc 9th Austr Data Mining Conf 121:195–204
16. Bernard S, Adam S, Heutte L (2012) Dynamic random forests. Pattern Recogn Lett 33:1580–1586
17. Ye Y, Wu Q, Huang JZ, Ng MK, Li X (2014) Stratified sampling of feature subspace selection in random forests for high dimensional data. Pattern Recogn 46:769–787
18. Adnan M. N., Islam M. Z. (2016) "Forest CERN: A new decision forest building technique", In: Proceedings of the 20th Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 304–315
19. Adnan MN, Islam MZ (2016) Optimizing the number of trees in a decision forest to discover a subforest with high ensemble accuracy using a genetic algorithm. Knowl-Based Syst 110:86–97
20. Adnan MN, Islam MZ (2017) Forest PA: Constructing a decision forest by penalizing attributes used in previous trees. Expert Syst Appl 89:389–403
21. Breiman L, Friedman J, Olshen R, Stone C (1985) Classification and Regression Trees. U.S.A, Wadsworth International Group, CA
22. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63:3–42
23. Quinlan JR (1993) C4.5: Programs Programming for Machine Learning. Morgan Kaufman, San Francisco
24. Munoz GM, Suarez A (2010) Out-of-bag estimation of the optimal sample size in bagging. Pattern Recogn 43:143–152
25. Kirby KN, Maraković NN (1995) Modeling myopic decisions: Evidence for hyperbolic delay-discounting within-subjects and amounts. Organ Behav Hum Decis Process 64(1):22–30
26. Vuchinich RE, Simpson CA (1998) Hyperbolic temporal discounting in social drinkers and problem drinkers. Exp Clin Psychopharmacol 6(3):292–305
27. Beebe, H. F. Nelson, "The mathematical-function computation handbook - programming using the MathCW portable software library", Springer International Publishing AG, 1st Edition, pp. 273–282. DOI:https://doi.org/10.1007/978-3-319-64110-2. ISBN 978–3–319–64109–62017
28. Tang EK, Suganthan PN, Yao X (2006) An analysis of diversity measures. Machine Learning 65:247–271
29. Margineantu DD, Dietterich TG (1997) "Pruning adaptive boosting", In: Proceedings of the 14th International Conference on Machine Learning, pp. 211–218
30. Bhatnagar V, Bhardwaj M, Sharma S, Haroon S (2014) Accuracy-diversity based pruning of classifier ensembles. Prog Artif Intell 2:97–111
31. Biauand G, Scornet E (2016) A random forest guided tour. Test 25(2):197–227
32. Liu Y, Zhang L, Nie Yan Y, Rosenblum DS (2016) "Fortune teller: predicting your career path", In: Thirtieth AAAI conference on artificial intelligence
33. Peersman C, Daelemans W, Van Vaerenbergh L (2011) "Predicting age and gender in online social networks", In: 3rd international workshop on Search and mining user-generated contents, pp. 37–44
34. Bin Tareaf R, Berger P, Hennig P, Jung J, Meinel C (2017) "Identifying audience attributes: predicting age, gender and personality for enhanced article writing", In: 2017 International Conference on Cloud and Big Data Computing, pp. 79–88
35. Asim Y, Raza B, Malik AK, Shahaid AR, Alquhayz H (2019) "An adaptive model for identification of influential bloggers based on case-based reasoning using random forest", IEEE Access, pp. 87732–87749.
36. Soonthornphisaj N, Sira-Aksorn T, Suksankawanich P (2018) "Social media comment management using smote and random forest algorithms", In: 9th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp. 129–134, IEEE.
37. Preoţiuc-Pietro D, Liu Y, Hopkins D, Ungar L (2017) "Beyond binary labels: political ideology prediction of twitter users", In: 55th Annual Meeting of the Association for Computational Linguistics, Vol. 1, pp. 729–740
38. Liu Y, Zheng Y, Liang Y, Liu S, Rosenblum DS (2016) "Urban water quality prediction based on multi-task multi-view learning", In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, pp 2576- 2582

39. Ying-Xun L, Lai C-F, Huang Y-M, Chao H-C (2013) "Multi-appliance recognition system with hybrid SVM/GMM classifier in ubiquitous smart home", Information Sciences, Vol. 230, ISSN 0020-0255, pp. 39–55
40. Cafri G, Li L, Paxton EW, Fan J (2018) Predicting risk for adverse health events using random forest. J Appl Stat 45(12):2279–2294
41. Iwendi C, Bashir AK, Peshkar A, Sujatha R, Chatterjee JM, Pasupuleti S, Mishra R, Pillai S, Jo O (2020) COVID-19 patient health prediction using boosted random forest algorithm. Front Public Health 8:357
42. Malki Z, Atlam ES, Hassanien AE, Dagnew G, Elhosseini MA, Gad I (2020) "Association between weather data and COVID-19 pandemic predicting mortality rate: Machine learning approaches", Chaos, Solitons & Fractals, pp.110–137