International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

Design an Optimal Decision Tree based Algorithm to Improve Model Prediction Performance

Shabana Pathan[1], Dr. Sanjeev Kumar Sharma[2]
[1]Research scholar
Department ofComputerScience & Engineering
Oriental University
Indore, (M.P), India
Assistant Professor,
St.Vincent Pallotti College of Engineering & Technology,
Nagpur,India
sbn.pathan@gmail.com
[2]Associate Professor
Department of Computer Science and Engineering
Oriental Institute of Science and Technology
Bhopal, (M.P), India
spd50020@gmail.com

Abstract—Performance of decision trees is assessed by prediction accuracy for unobserved occurrences. In order to generate optimised decision trees with high classification accuracy and smaller decision trees, this study will pre-process the data. In this study, some decision tree components are addressed and enhanced. The algorithms should produce precise and ideal decision trees in order to increase prediction performance. Additionally, it hopes to create a decision tree algorithm with a tiny global footprint and excellent forecast accuracy. The typical decision tree-based technique was created for classification purposes and is used with various kinds of uncertain information. Prior to preparing the dataset for classification, the uncertain dataset was first processed through missing data treatment and other uncertainty handling procedures to produce the balanced dataset. Three different real-time datasets, including the Titanic dataset, the PIMA Indian Diabetes dataset, and datasets relating to heart disease, have been used to test the proposed algorithm. The suggested algorithm's performance has been assessed in terms of the precision, recall, f-measure, and accuracy metrics. The outcomes of suggested decision tree and the standard decision tree have been contrasted. On all three datasets, it was found that the decision tree with Gini impurity optimization performed remarkably well.

Keywords-Gini impurity optimization, decision tree, Uncertainty Estimation, Balanced Dataset.

I. INTRODUCTION

Data mining is useful in a variety of industries, including medical, marketing, finance, power, banking, manufacturing, and telecommunications. However, the data in these real-world applications is unreliable, confusing, inconsistent, and noisy, adding to the uncertainty. In addition, uncertainty emerges as a result of missing or insufficient data. Because of the ambiguities in the data, we have poor information, which makes data mining activities more difficult. Handling these uncertainties in data is a vital responsibility or task for a decision-making system to perform intelligent data analysis and make wise conclusions [1,2,3].

This paper identifies uncertain data analysis and provides data mining models, i.e. classification decision-making systems that employ decision tree models to address the aforementioned uncertainties in categorical and numerical data classification. The proposed research utilizes deep learning to handle uncertainty in datasets, followed by the development of three decision tree models for classification purposes.

When it comes to categorization and prediction, the decision tree is the workhorse. The decision tree uses a tree representation of the problem, where the class label is associated with each leaf node and inner node displays attributes of the tree, to reach a conclusionAn attribute or characteristic is represented by the centre node of a decision tree, a decision rule is represented by the branch, and the result is represented by each leaf node[4,5,6,7]. The leaf node figures out how to divide things apart according to their attribute values. Partitioning the tree in a recursive fashion can be done with a technique called recursive partitioning. For better decision-making, use this flowchart-like framework. It's a graphical representation, like a flowchart, that may be simply understood and used to represent human thought processes. The structure of a decision tree makes it intuitive to analyses

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

the data it contains. It can be trained faster than the neural network technique. The computing time for a decision tree is inversely correlated with the number of records and characteristics in the input data. A distribution-free (or non-parametric) statistical technique is one that does not rely on a certain distribution, such as the decision tree. Decision trees are able to deal with high-dimensional data effectively [8].

This is the fundamental principle underlying any decision tree algorithm:

1. Select the most relevant attribute using the Attribute Selection Measures (ASM) to segment the data.

2. Create a decision node based on that property and use it to partition the dataset.

3. Performs recursive procedure for each child node in the tree until one of conditions is met, hence initiating tree construction.

o Every tuple shares the same value for that attribute.

As of right now,

o There are no accessible further attributes..

o No further instances exist.

The root node property is defined at each branch in the decision tree, which is the main issue. It's common practise to use the term "set of attributes" to describe this procedure. There are two typical methods for choosing attributes:

(a) Knowledge Accumulation

(b) The Gini Index

(c) Gini Index,

a) Information Gain

When we divide the training using a branch in the decision tree, the entropy shifts to reflect the new structure. Entropy change may be measured via information gain. Entropy quantifies the impure nature of a large number of occurrences by measuring the fluctuation in a random variable. Entropy increases as the information value increases. A decision tree approach called ID3 (Iterative Dichotomiser) employs information gain. [9,10,11].

$$\text{Info}(D) = -\sum_{i=1}^{m} pi \log_2 pi$$

Pi is the likelihood that a D tuple belongs to Di.

$$\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|Dj|}{|D|} \text{ X Info}(D_j)$$

$$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$$

Where,

The typical amount of information required to determine the class label of a tuple is Info(D).

|Dj|/|D| acts as the weight of the jth partition.

Information that should be provided to insert a tupleinto one of A's partitions of D is denoted by InfoA(D).

At node N (), we pick the attribute A with the biggest information gain, Gain(A), to use as the separating characteristic.

b) Gini Ratio:

Information gain is biassed for the characteristic with numerous outcomes. It suggests that the property with a lot of various possible values is chosen. Take the example of a customer ID, which has zero info(D) owing to pure partition and is a unique identification. Through excessive segmentation, this maximises information acquisition. [12].

The gain ratio is an improvement over ID3 and an expansion of the information gain employed by C4.5. The Gain Ratio solves the issue of bias by employing Split Info to normalise the information gain.

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

Where,

|Dj|/|D| acts as the weight of the jth partition.

The total number of distinct values for Attribute A is represented by v.

Gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$

The characteristic with the largest gain ratio is chosen as the splitting attribute.

c) Gini Index

The Gini Index is a statistic that determines how often a variable is incorrectly defined at random. As a result, an attribute with a lower Gini index can be selected.

In the CART (Classification and Regression Tree) decision tree technique, split points are created using the Gini approach.

$$\text{Gini(D)} = 1 - \sum_{i=1}^{m} \text{Pi}^2$$

Pi provides the likelihood that a tuple in D belongs to class Ci.

The Gini Index investigates a binary split for every attribute. It is possible to determine a weighted sum of the impurities in each division. When data D is split into D1 and D2 as a consequence of a binary split on attribute A, the following is the Gini index for data D:

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

$$\text{Gini}_A(D)=\frac{|D1|}{|D|}\text{Gini}(D_1)+\frac{|D2|}{|D|}\text{Gini}(D_2)$$

Discrete-valued properties are selected by selecting the subset with the lowest gini index. When dealing with characteristics that have continuous values, the method entails picking a splitting point by comparing all possible splitting points between pairs of neighbouring values and selecting the one with the smallest gini index.

$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

The splitting attribute is the one with the lowest Gini index.

II. LITERATURE REVIEW

Classification learning for data streams is a hot issue. Data streams' underlying concepts can change over time, requiring rapid revisions to classifiers. Observing online classification accuracy can detect idea change. If accuracy falls below a threshold, a concept shift occurs. This methodology assumes that a decline in categorization accuracy indicates a concept shift. In reality, data streams often contain noise that reduces categorization accuracy.How to handle missing values is another categorization difficulty. How a learning model classifies new instances with missing values and updates itself is an unresolved topic. FlexDT introduces fuzzy logic to data stream classification to overcome these difficulties. Threefold benefits. First, FlexDT's flexible structure handles concept changes effectively [13].

Author [14] proposed an Ant Colony Optimization-based method for building decision trees (ACO). ACO is a metaheuristic that draws inspiration from how actual ants search for the best answers utilising local heuristics and prior knowledge.Good results of ant colony methods for combinatorial optimization problems imply their effectiveness in building decision trees. To improve decision tree accuracy, they devised the Ant Colony method (ACDT). A novel algorithm's heuristic function uses CART's splitting rule .The suggested approach is tested on many UCI Machine Learning benchmark data sets. Empirical data suggest that ACDT outperforms alternative approaches [15,16,17].

A sequential ensemble credit scoring model based on extreme gradient boosting (XGBoost) was proposed by the authors [18]. There are three steps to this process. The first step is data pre-processing, which includes normalisation and handling of missing values. Then, we utilise a model-based feature selection technique to filter out the extraneous information. Third, the XGBoost hyper-parameters are tuned in an iterative fashion using Bayesian hyper-parameter optimisation. Reference points for experiments might be things like hyper-parameter optimisation strategies or default classifiers. Bayesian hyper-parameter optimisation outperforms random, grid, and manual searches. The proposed model outperforms the state-of-the-art on four different metrics: accuracy, error rate, AUC-H, and Brier score. Improve your credit score with the use of feature significance scores and a decision chart [18].

Campagner's paradigm for dealing with Machine Learning, based on TWD and the trisecting-acting-outcome model (ML), was developed in response to the challenges posed by ambiguity. While TWD was used to let ML models sit on their hands if their output was uncertain, it was primarily employed to locate and properly account for uncertain occurrences in the input. After highlighting the benefits of the three-pronged approach, the authors gave a narrative analysis of the existing condition of TWD applications with regard to the various framework-identified areas of concern [19, 20].

Decision tree classifiers are regarded as the most popular methods for representing classifiers in data. Many academics from many fields and backgrounds have focused on the problem of growing a decision tree utilising readily available data, such as machine learning, pattern recognition, and statistics. Decision tree classifiers have been proposed for usage in a number of fields, including medical sickness analysis, text categorization, user smartphone classification, images, and many more. Also included are a thorough examination and explanation of the datasets, algorithms/approaches used, and outcomes of the article. All of the investigated approaches were subsequently investigated in order to further illustrate the authors' subjects and identify the most trustworthy classifiers.As a consequence, the applications of different dataset types are looked at, and the outcomes are provided.[21][22].

Authors have [23] created decision trees using K-means clustering. After clustering data into groups, it sorts data within each cluster. Each cluster constructs one layer of the tree. It applies depth growth constraint to restrict the size of the decision tree [24]. The author claims that clustering decision tree with a simple strategy provides better precision and less computational complexity. Recently many researchers used the clustering methods to build decision trees for large data with various factors of construction and evaluation. These algorithms aim making precise and comprehensible decision trees and can be applicable for real-time applications [25][26].

Random Feature Weights Ensemble: Each property is given a random weight from 0.0 to 1.0. By giving each characteristic a merit value (the Gini index multiplied with random weights), it creates a decision tree. The attribute with the greatest merit value is then chosen as the splitting attribute. This approach enables the creation of a random forest that is more versatile. The drawback of the random feature weight approach is that assigning positive weights to characteristics raises the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

likelihood that certain attributes will be chosen as the root node. So many identical trees might grow and limit the forest's variety. [27][28][29][30].

III.PROPOSED METHODOLOGY

In this research, a framework for uncertainty estimation and handling using several strategies is given, as shown in fig.1. Uncertainty estimation, uncertainty handling, optimal decision tree, and performance evaluation are the four phases of the framework. The raw dataset is first applied to the uncertainty estimation phase, where deep learning models are utilized to identify dataset uncertainty. If there is uncertainty in the dataset, it is resolved during the uncertainty handling phase, which employs a variety of strategies such as missing data treatment and transformation techniques. After the data has been cleaned, the next step is to classify it using a decision tree classifier, which uses parameter optimization and impurity optimization methods to get accurate results. The performance of the algorithms is assessed in the following phase using several accuracy metrics. For handling uncertainty, an optimum strategy is identified based on various comparisons. The flow chart of the entire process is shown in fig 1. The algorithm is also discussed in the next section.

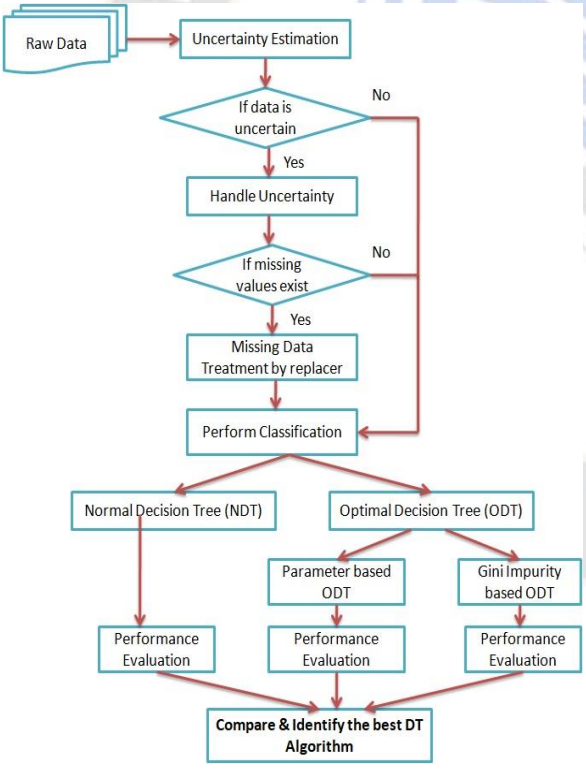

Figure 1.Flowchart of Decision Tree Optimization for Uncertainty handling

Algorithm

The algorithm for uncertainty handling is mentioned below. The input is raw dataset in which uncertainty should be identified and resolved if exists using different techniques. The output of algorithm is optimized method for decision tree classification with different performance measures.

Algorithm: Optimized Algorithm for Uncertainty Handling

Inputs: *Raw dataset*

Output: *Optimal decision tree, Accuracy, Precision, Recall, F-Score*

1.Dataset collection
Df →load dataset()
2. perform uncertainty estimation using neural network (NN)
Create NN architecture with dropout and without dropout layer
Plot uncertainty on data

- *model_without_dropout = architecture(layers_shape =[5,10,20,10,5], input_dim= 1, output_dim=1, dropout_proba=0, reg=0, act='relu', verbose=1)*

model_with_dropout = architecture(layers_shape=[5,10,20,10,5], input_dim= 1, output_dim=1, dropout_proba=0.05, reg=0.00475, act='relu', verbose=1)
3. Pre-process dataset
4. ***if*** *(missing data exists):*
treat_missingData()
apply replacer()
Else:
Proceed cleaned_data()
5. Apply transformation techniques
ss = StandardScaler()
mm = MinMaxScaler()
ma = MaxAbsScaler()
ns = Normalizer(norm = 'l2')
6. split dataset into train and test data with random state
7. Apply decision tree classifier
model = DecisionTreeClassifier()
8. Apply decision tree classifier with parameter optimization
clf = DecisionTreeClassifier(criterion="entropy", max_depth =3)
9. Apply decision tree classifier with impurity optimization

- *Calculate the GINI impurity for the provided binary class observations.*
- *Calculating the chances about seeing each of the classes*
- *Computing GINI*

*gini = 1 - (p1class ** 2 + p2class ** 2)*

- *Compute the optimum split of a decision tree for given "Xn" features and "Y" target.*
- *Generate dataset for split*

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

DataFrame = self.Xn.copy()
DataFrame ['Y'] = self.Y

- *Attain the base input's GINI impurity*

GINI_BS = self.get_GINIimpurity()

- *Determining which split produces the highest GINI gain*

Gain_max = 0

- *The default best feature, as well as a split*

best_feature = None
best_value = None

- *GINI impurity weighted computation*

*Wt_GINI = Wt_left * gini_left + Wt_right * gini_right*

- *GINI gain computation*

Gain_GINI = Base_GINI – Wt_GINI

- *Checking to see whether this is the best split we've seen thus far.*

If Gain_GINI > Gain_max:
best_feature = feature
best_value = value

- *Choosing the optimum gain for the current situation*

Gain_max = Gain_GINI
10. identify the execution time , f-score, recall ,accuracy, andprecision

- *accuracy = accuracy_score (y_tst, model.pred(x_test))*
- *precision = precision_score (y_tst, model.pred(x_test))*
- *recall = recall_score (y_tst, model.pred(x_test))*

11. Perform comparison between results of different transformation techniques
12. Perform comparison between base decision tree, parameterized and impurity based decision tree
13. Perform comparison between with uncertainty results and without uncertainty
14. Identify optimized algorithm for uncertainty handling

Gini Impurity Optimization

A machine learning technique known as the decision tree algorithm (DT for short) is used to categorise an observation given a collection of input features. The algorithm develops a set of guidelines at different levels of decision making such that a certain statistic is optimised. The "best" splits of the numerical variables are to be produced. The feature matrix will be labelled as X, and the target variable will be denoted as Y = 0 and 1. There are different parameters of decision tree which can be used for optimization of results, such as node, gini impurity, level and splitting.

The node is the basic building block of a decision tree. When looking at a standard decision tree schema (like the one in the title picture), the rectangles or bubbles that connect to other nodes downstream are called nodes.

The primary characteristics of each node are as follows:

Features of the node include:

- The Gini impurity score,
- The number of observations, the number of observations for each binary target class, and
- The feature matrix X.

IV.DATASET AND EXPERIMENTAL SETUP

Datasets

There are three types of datasets are used in this thesis work. Details of the dataset are presented as follows-

Titanic Dataset

Use the training set to create machine learning models. We present the result—also referred to as "ground truth"—for each passenger in the training set. Our model will be built on "features" like passenger class and gender. Additionally, new features may be created through feature engineering. The test set should be used to evaluate how well our model works when applied to novel data. We do not disclose the ground truth for each passenger in the test set. It is our responsibility to foresee these outcomes. Use the model we trained to calculate the probability that each test set participant survived the Titanic's sinking.

PIMA Indian diabetes dataset

This data comes from the National Institute of Diabetes and Digestive and Kidney Diseases. The diagnostic indicators in the dataset are meant to be used to make a diabetes diagnosis. The criteria for selecting these specific examples from the overall database were rather stringent. All of the patients at this clinic are 21-and-up Pima Indian women. The datasets each have one independent variable, outcome, and a number of dependent, medical predictors.

Heart Disease Dataset

Even though this dataset contains 76 features, all published studies employ only 14 of them as a subset of those attributes. The Cleveland database, in particular, is the only one that machine learning experts have used to date. The "target" field shows whether or not the patient is suffering from heart disease. It runs from 0 (no presence) to 4 (present) on a scale of one to four (present).

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

Datasets for the proposed research were gathered from a variety of intenet sources and their number is shown in the table 1 below :

Table 1: Datasets details

Dataset Name	Quantity	Training Data	Testing Data
Titanic	891	713	178
Heart Disease	304	243	61
PIMA	769	615	154

Experimental Setup

The suggested study is now being evaluated in Python using machine learning and classification strategies. Scikit-learn (Sklearn) is an efficient and powerful package for machine learning algorithms in Python. Dimensionality reduction, classification, regression, and clustering, among other techniques, are all accessible via the Python interface. These are all aspects of statistical modelling and machine learning. Python is the language used to create this library. SciPy, NumPy, and Matplotlib are the foundations upon which it stands.

Evaluation Parameters

Accuracy and F-measure metrics are used to assess the suggested approach. Accuracy in this context is defined as labels that are correctly categorized, while F-measure refers to average values computed from precision and recall. The formulae listed below are used to calculate these metrics.

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN)$$

$$\text{Precision} = TP/(TP+FP)$$

$$\text{Recall} = TP/(TP+FN)$$

$$\text{F-Measure} = 2TP/(2TP+FP+FN)$$

where True Positive, True Negative, False Positive, and False Negative correspond to the notation TP, TN, FP, and FN, respectively. The accuracy and F-score measurement interval lies between zero and one. Increased values improve the efficacy of Twitter sentiment analysis. We use the Adam optimisation concept, wherein training is used to get optimal outcomes in opinion ranking.

V. Results and Discussions

A GINI impurity score is assigned to each node. The target variable's distribution in the node, or only the number of Y=1 and Y=0 observations in a node, is all that is required to compute the GINI impurity. Impurity in GINI is officially defined as follows: Gini impurity quantifies the frequency with which a randomly selected element from the set would be erroneously classified if it were randomly tagged in accordance with the distribution of labels in the subset.

Table 2 Performance Evaluation of proposed method on different datasets

Method	Metrics	Titanic	PIMA	Heart
Decision Tree + Uncertainty Handling	Accuracy (%)	68	65	54
	Precision(%)	39.54	11.08	37.89
	Recall(%)	42.74	12.46	43.01
	F-Score (%)	40.78	11.62	39.98
Decision Tree + Uncertainty Handling + Gini Impurity Optimization	Accuracy (%)	73.8	76	82
	Precision(%)	70.24	63.20	79.80
	Recall(%)	74.49	60.02	74.00
	F-Score(%)	72.26	62.92	77.79

Performance evaluation of proposed methodology is shown in table 2, in which Decision trees incorporating simple approaches to dealing with uncertainty have been found to produce only middling outcomes. An excellent outcome was obtained by using a decision tree with Gini impurity optimization. Optimal results in predicting coronary disease. It has been demonstrated that a decision tree can be utilized to maximize predictions made using the Gini impurity. The prediction accuracy of classifiers is enhanced by a novel method devised for the optimal decision tree.

VI. CONCLUSIONS

The suggested approach aims to raise the quality of performance prediction and categorization by employing decision tree and Gini impurity optimization. The conventional decision tree-based method serves as a classification tool and is applicable to a wide range of unreliable data formats. The missing data treatment and other uncertainty handling processes were applied to the uncertain dataset to generate the balanced dataset before it was ready for classification. The suggested technique has been evaluated using data from three distinct real-time datasets: the Titanic dataset, the PIMA Indian Diabetes dataset, and the heart disease datasets. Evaluation parameters have all been used to evaluate the effectiveness of the proposed approach. The proposed optimal decision tree's outcomes have been compared to those of the traditional decision tree. We discovered that the decision tree optimized for Gini impureness performed exceptionally well across all three datasets.

References

[1] Meyer ,P. E. & Bontempi, G. On the use of variable complementarity for feature selection in cancer classification. Workshops on applications of evolutionary computation, 2006. Springer, 9102.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7295
***Article Received**: 24 March 2023 **Revised**: 30 April 2023 **Accepted**: 16 June 2023*

[2] Mingers, J. 1989. An empirical comparison of pruning methods for decision tree induction. Machine learning, 4, 227-243.

[3] Painsky, A. & Rosset, S. 2016. Cross-validated variable selection in tree-based methods improves predictive performance. IEEE transactions on pattern analysis and machine intelligence, 39, 2142-2153.

[4] Pande, A., Li, L., Rajeswaran, J., Ehrlinger, J., Kogalur, U. B., Blackstone, E. H. & Ishwaran, H. 2017. Boosted multivariate trees for longitudinal data. Machine learning, 106, 277-305.

[5] Panhalkar, A. R. & Doye, D. D. 2021. Optimization of decision trees using modified African buffalo algorithm. Journal of King Saud University-Computer and Information Sciences.

[6] Peng, H., Long, F. & Ding, C. 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence, 27, 1226-1238.

[7] Probst, P. & Boulesteix, A.-L. 2017. To tune or not to tune the number of trees in random forest. J. Mach. Learn. Res., 18, 6673-6690.

[8] Rahman, M. G. & Islam, M. Z. 2013. Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques. Knowledge-Based Systems, 53, 51-65.

[9] Rahman, M. G. & Islam, M. Z. 2016. Missing value imputation using a fuzzy clustering-based EM approach. Knowledge and Information Systems, 46, 389-422.

[10] Rokach, L. 2016. Decision forest: Twenty years of research. Information Fusion, 27, 111-125.

[11] Setiawan, N. A., Venkatachalam, P. A. & Hani, A. F. M. Missing attribute value prediction based on artificial neural network and rough set theory. 2008 international conference on bioMedical engineering and informatics, 2008. IEEE, 306-310.

[12] Sinharay, S., Stern, H. S. & Russell, D. 2001. The use of multiple imputation for the analysis of missing data. Psychological methods, 6, 317.

[13] Song, Q., NI, J. & Wang, G. 2011. A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE transactions on knowledge and data engineering, 25, 1-14.

[14] SONG, Y.-Y. & YING, L. 2015. Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27, 130.

[15] Tibshirani, R. & Hastie, T. 2007. Margin Trees for High-dimensional Classification. Journal of Machine Learning Research, 8.

[16] Tsang, S., Kao, B., Yip, K. Y., Ho, W.-S. & Lee, S. D. 2009. Decision trees for uncertain data. IEEE transactions on knowledge and data engineering, 23, 64-78.

[17] Tutz, G. & Ramzan, S. 2015. Improved methods for the imputation of missing data by nearest neighbor methods. Computational Statistics & Data Analysis, 90, 84-99.

[18] Wang, S., Aggarwal, C. & Liu, H. 2018. Random-forest-inspired neural networks. ACM Transactions on Intelligent Systems and Technology (TIST), 9, 1-25.

[19] Wang, S., Jiang, L. & Li, C. 2015. Adapting naive Bayes tree for text classification. Knowledge and Information Systems, 44, 77-89.

[20] Wang, T., Qin, Z., Jin, Z. & Zhang, S. 2010. Handling over-fitting in test cost-sensitive decision tree learning by feature selection, smoothing and pruning. Journal of Systems and Software, 83, 1137-1147.

[21] Wang, Y. & Zhang, N. 2014. Uncertainty analysis of knowledge reductions in rough sets. The Scientific World Journal, 2014.

[22] Windeatt, T. & Ardeshir, G. An empirical comparison of pruning methods for ensemble classifiers. International Symposium on Intelligent Data Analysis, 2001. Springer, 208-217.

[23] Xia, Y., Liu, C., Li, Y. & Liu, N. 2017. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. Expert Systems with Applications, 78, 225-241.

[24] Yager, R. R., Zadeh, L. A., Kosko, B. & Grossberg, S. 1994. Fuzzy sets, neural networks, and soft computing.

[25] Yan, J., Zhang, Z., Xie, L. & Zhu, Z. 2019. A unified framework for decision tree on continuous attributes. IEEE Access, 7, 11924-11933.

[26] Ye, Y., Wu, Q., Huang, J. Z., Ng, M. K. & Li, X. 2013. Stratified sampling for feature subspace selection in random forests for high dimensional data. Pattern Recognition, 46, 769-787.

[27] Zadeh, L. A. 1996. Soft computing and fuzzy logic. Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi a Zadeh. World Scientific.

[28] Zhang, Z., Zhao, Z. & Yeonm, D.-S. 2020. Decision Tree Algorithm-Based Model and Computer Simulation for Evaluating the Effectiveness of Physical Education in Universities. Complexity, 2020.

[29] Zhao, L., Lee, S. & Jeong, S.-P. 2021. Decision Tree Application to Classification Problems with Boosting Algorithm. Electronics, 10, 1903.

[30] Zharmagambetov, A., Gabidolla, M. & Carreira-Perpinan, M. A. Improved multiclass AdaBoost for image classification: The role of tree optimization. 2021 IEEE International Conference on Image Processing (ICIP), 2021. IEEE, 424-428.