Hindawi
Complexity
Volume 2020, Article ID 6688075, 21 pages
https://doi.org/10.1155/2020/6688075

Research Article

Investigating Tree Family Machine Learning Techniques for a Predictive System to Unveil Software Defects

Rashid Naseem,[1] Bilal Khan,[2] Arshad Ahmad,[1] Ahmad Almogren,[3] Saima Jabeen,[1] Bashir Hayat,[4] and Muhammad Arif Shah[1]

[1]*Department of IT and Computer Science, Pak-Austria Fachhochschule: Institute of Applied Sciences and Technology, Mang Khanpur Road, Haripur 22620, Pakistan*
[2]*Department of Computer Science, City University of Science and Information Technology, Peshawar 25000, Pakistan*
[3]*Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia*
[4]*Institute of Management Sciences Peshawar, Peshawar 25000, Pakistan*

Correspondence should be addressed to Arshad Ahmad; yaarshad@gmail.com

Received 4 October 2020; Revised 15 November 2020; Accepted 20 November 2020; Published 30 November 2020

Academic Editor: M. Irfan Uddin

Software defects prediction at the initial period of the software development life cycle remains a critical and important assignment. Defect prediction and correctness leads to the assurance of the quality of software systems and has remained integral to study in the previous years. The quick forecast of imperfect or defective modules in software development can serve the development squad to use the existing assets competently and effectively to provide remarkable software products in a given short timeline. Hitherto, several researchers have industrialized defect prediction models by utilizing statistical and machine learning techniques that are operative and effective approaches to pinpoint the defective modules. Tree family machine learning techniques are well-thought-out to be one of the finest and ordinarily used supervised learning methods. In this study, different tree family machine learning techniques are employed for software defect prediction using ten benchmark datasets. These techniques include Credal Decision Tree (CDT), Cost-Sensitive Decision Forest (CS-Forest), Decision Stump (DS), Forest by Penalizing Attributes (Forest-PA), Hoeffding Tree (HT), Decision Tree (J48), Logistic Model Tree (LMT), Random Forest (RF), Random Tree (RT), and REP-Tree (REP-T). Performance of each technique is evaluated using different measures, i.e., mean absolute error (MAE), relative absolute error (RAE), root mean squared error (RMSE), root relative squared error (RRSE), specificity, precision, recall, F-measure (FM), G-measure (GM), Matthew's correlation coefficient (MCC), and accuracy. The overall outcomes of this paper suggested RF technique by producing best results in terms of reducing error rates as well as increasing accuracy on five datasets, i.e., AR3, PC1, PC2, PC3, and PC4. The average accuracy achieved by RF is 90.2238%. The comprehensive outcomes of this study can be used as a reference point for other researchers. Any assertion concerning the enhancement in prediction through any new model, technique, or framework can be benchmarked and verified.

1. Introduction

Software engineering (SE) is a discipline that is worrisome with all qualities of software development from the beginning period of software specification over to keeping up to the software maintenance after it has gone into practice [1]. In the field of SE, software defect prediction (SDP) is one of the most significant and dynamic research zones that assumes a significant job in the software quality assurance (SQA) [2, 3]. An SD is a flaw or insufficiency in a software system that roots the development of a spontaneous result. The rising convolutions as well as dependencies of software systems have increased the difficulty in delivering software with minimal effort, high caliber, and maintainability as well which increases the chances of introducing software defects (SDs) [4]. Generally, SDs are found in the testing stage of the Software Development Life Cycle (SDLC) [5]. An SD can moreover be the situation when the finalized software product does not meet the client's desire or client prerequisite [6] which causes the diminution of the software product quality and increases the development cost.

SDP is a momentous commotion to assure the substances of a software system that leads to adequate development cost and recover the quality by identifying defect-prone instances before testing [4]. It moreover embraces categorizing software components in new varieties of a software system, which constructs the testing progression supplementary by concentrating on testing and evaluating the components classified as defective [7]. Defects adversely affect software reliability and quality [8].

SDP in the primary period of SDLC is measured as the utmost thought-provoking aspect of SQA [9]. In SE, bug fixing and testing are very costly which also requires a massive amount of resources. Forecasting software defects in software development have been observed by numerous studies in the last decades. Amid all these studies, machine learning (ML) techniques are considered as the best approach toward SDPs [7, 10, 11].

Keeping the above issue related to SDP, various researchers evaluated and built SDP models utilizing diverse classification techniques. Still, it is quite challenging to sort any broad-spectrum preparation to inaugurate the usability of these techniques. Comprehensively, it was originated that despite some uniqueness in the studies, no specific SDP procedure conveys a higher precision to different methods slantingly on different datasets. Most of the researchers have utilized different evaluation measures to achieve a higher accuracy, but to the best of our knowledge, no one has worked on reducing error rate which is also an important factor for any prediction model [12, 13].

However, this study focuses on exploring Tree Family (TF) ML techniques for SDP. These TF-ML techniques include Credal Decision Tree (CDT), Cost-Sensitive Decision Forest (CS-Forest), Decision Stump (DS), Forest by Penalizing Attributes (Forest-PA), Hoeffding Tree (HT), Decision Tree (J48), Logistic Model Tree (LMT), Random Forest (RF), Random Tree (RT), and REP-Tree (REP-T). These techniques are employed on ten different datasets including AR1, AR3, CM1, KC2, KC3, MW1, PC1, PC2, PC3, and PC4. All the experiments are validated using mean absolute error (MAE), relative absolute error (RAE), root mean squared error (RMSE), root relative squared error (RRSE), specificity, precision, recall, F-measure (FM), G-measure (GM), Matthew's correlation coefficient (MCC), and accuracy.

A question may be raised for the reason of selecting TF-ML techniques. The motive for the selection of TF-ML techniques is well-thought-out to be one of the finest and ordinarily used supervised learning methods. Tree-based techniques empower predictive models with ease of interpretation, stability, and high accuracy [14]. Disparate linear models such as TF-ML techniques map nonlinear relationships pretty well. They are flexible at resolving several kinds of problems at hand (regression or classification). These techniques also work for both categorical and continuous input and output variables [15, 16]. TF is one of the wildest ways to categorize the utmost momentous variables and relation between two or more variables. TFs can produce new features that have improved power to forecast object variable. It involves fewer data cleaning contrasted to several other modeling techniques. It is not prejudiced via outliers and missing values to a rational amount [17–19].

The main contributions of this research are as follows:

(i) We exploited ten benchmarked TF-ML techniques (CDT, CS-Forest, DS, Forest-PA, HT, J48, LMT, RF, RT, and REP-T) for SDP.

(ii) We demeanor a series of tryouts on different datasets; i.e., AR1, AR3, CM1, KC2, KC3, MW1, PC1, PC2, PC3, and PC4.

(iii) To gain insights into the experimental outcomes, evaluation is accomplished using MAE, RAE, RMSE, RRSE, specificity, precision, recall, FM, GM, MCC, and accuracy.

(iv) To show the significance of the results, Friedman's two-way analysis of variance by ranks is performed.

Hereinafter, Section 2 presents the literature survey, Section 3 comprises the methodology and techniques, while experimental outcomes have conversed in Section 4, and Section 6 covers the inclusive conclusion.

2. Literature Survey

This section delivers an ephemeral study about existing techniques in the field of SDP. Several researchers have employed ML techniques for SDP at the initial phase of software development. Several particular studies have conversed here. Czibula et al. [11] presented a model grounded on Relational Association Discovery (RAD) for SDP. They applied all the investigations on the NASA dataset including MC2, KC1, KC3, JM1, MW1, PC3, PC4, CM1, PC1, and PC2. To assess the model by comparing it to other models using accuracy, probability of detection (PD), specificity, precision, and area under cover (ROC) assessment measure, the acquired outcomes presented that RAD performed well rather than other employed techniques.

Li et al. [20] recommended a framework for SDP named Defect Prediction through Convolutional Neural Network (DP-CNN). They evaluated DP-CNN on seven different open source projects that are camel, jEdit, Lucene, xalam, Xerces, synapse, and poi in terms of FM in defect predictions. Overall outcomes illustrate that, on average, DP-CNN enhanced the state-of-the-art method by 12%.

Jacob and Raju [21] introduced a hybrid feature selection (HFS) method for SDP. They also performed their analysis on NASA datasets including PC1, PC2, PC3, PC4, CM1, JM1, KC3, and MW1. The outcomes of HFS are benchmarked with Naïve Bayes (NB), neural networks (NNs), RF, Random Tree (RT), and J48. Benchmarking is carried out using sensitivity, specificity, accuracy, and Matthew's correlation coefficient (MCC). The analyzed outcome shows that HFS outperforms by improving classification accuracy from 82% to 98%.

Bashir et al. [22] presented a joined framework to improve the SDP model using data sampling (DS), ranker feature selection (FS) techniques, and iterative partition filter

(IPF) to conquest class imbalance, high dimensionality, and noise correspondingly. Seven ML techniques including NB, RF, KNN, MLP, SVM, J48, and decision stump are employed on CM1, JM1, KC2, MC1, PC1, and PC5 datasets for evaluations. The outcomes are carried out utilizing receiver operating characteristic (ROC) performance evaluation. Overall experimental outcomes of the proposed model outperformed other models.

In another study [7], the author projected a new approach for SDP utilizing hybridized gradual relational association and artificial neural network (HyGRAR) to classify the defective and nondefective objects. Experiments were achieved based on ten different open-source datasets that are JEdit 4.0, JEdit 4.2, JEdit 4.3, Anr 1.7, Tomcat 6.0, AR1, AR3, AR4, AR5, and AR6. For module evaluation, accuracy, sensitivity, specificity, and precision measures were utilized. The author concluded that HyGRAR achieved better outcomes compared to most of the foregoing projected approaches.

Alsaeedi and Khan [8] performed the comparison of supervised learning techniques including Bagging, SVM, Decision Tree (DT), and RF and ensemble classifiers on different NASA datasets that are KC2, KC3, PC1, PC3, PC4, PC5, JM1, CM1, MC1, and MC2. The basic learning and ensemble classifiers are evaluated using GM, specificity, F-score, recall, precision, and accuracy. The experimental results showed that RF, AdaBoost with RF, and DS with bagging outperformed other employed techniques.

A comparative exploration of several ML techniques for SDP is performed [9] on twelve NASA datasets that are PC1, PC2, PC3, PC4, PC5, MC1, MC2, JM1, CM1, MW1, KC1, and KC3, while the classification techniques include One Rule (OneR), NB, MLP, DT, RBF, kStar (K∗), SVM, KNN, PART, and RF. The performance of each technique is evaluated via MCC, ROC area, recall, precision, FM, and accuracy. It is imitated from the outcomes that neither the accuracy and nor the ROC can be utilized as an operative performance measure as both of these did not respond to the class imbalance problem.

Malhotra and Kamal [6] evaluated the efficiency of ML classifiers for SDP on twelve excessive NASA datasets by employing sampling methods and cost-sensitive classifiers. They examine five prevailing methods including J48, RF, NB, AdaBoost, and Bagging, as well as the SPIDER3 method for SDP. They have compared the performance based on accuracy, sensitivity, specificity, and precision. The outcomes show improvement in the prophecy competence of ML classifiers with the usage of oversampling methods. Moreover, the projected SPIDER3 method shows hopeful results.

Manjula and Florence [23] developed a hybrid model based on the genetic algorithm (GA) and deep neural network (DNN). GA is used for feature optimization while DNN is for classification. The performance of the projected technique is compared with NB, RF, DT, Immunos, ANN-artificial bee colony (ABC), SVM, majority vote, AntMiner+, and KNN. All the performances are carried out on datasets that include KC1, KC2, CM1, PC1, and JM1 and assessed via recall, F-score, sensitivity, precision, specificity, and accuracy. The experimental outcomes showed that the recommended technique outperformed as compared to other techniques in terms of achieving better accuracy.

Researchers have used various techniques to incredulous the boundaries of SDP on a variety of datasets. In each study, different evaluation measures are accomplished to evaluate the proposed techniques. The overall summary of the literature discussed above is given in Table 1 As shown in Table 1, each study has used different evaluation measures to achieve a higher accuracy, but no one made an effort to decrease the error rate which is a significant feature.

3. Research Methodology

This research aims to present the performance analysis of TF-ML techniques for SDP on various datasets including AR1, AR3, CM1, KC2, KC3, MW1, PC1, PC2, PC3, and PC4. The complete research is prepared via the procedure shown in Figure 1. All experimentation is performed using open source ML and DM tool Weka version 3.9 (https://machinelearningmastery.com/use-ensemble-machine-learningalgorithms-weka/). After the selection of datasets, a preprocessing step is applied on each dataset for two main purposes: replacing the missing values and changing the class attribute from numerical to categorical as some of the techniques do not work on numerical class attributes. Then, ML techniques are applied to each dataset, and the outcomes are assessed using different assessment measures to show the better performance of an individual technique. Eleven assessment measures named MAE [13, 24, 25], RMSE [8, 26, 27], RAE [22, 28, 29], RRSE [28], specificity [30–32], precision [9, 15, 33], recall [9, 10, 31], FM [9, 15, 20], GM [8, 34], MCC [9, 35, 36], and accuracy [37–39] are utilized to evaluate the performance of ML techniques on SDP datasets.

3.1. Datasets Description

3.1. Datasets Description. Each dataset consists of some attributes along with a known output class. Datasets contain numerical data, while the total number of attributes and instances are different, as presented in Table 2, where the first column presents the datasets and second and third columns present the number of attributes and number of instances, respectively. The fourth and fifth columns represent the number of defective modules and the number of non-defective modules, while the last column shows the type of data in each dataset. However, Table 3 shows the list of all attributes (software metrics) according to each dataset utilized in this research, where "-" means that this attribute is not part of the dataset while "Y" represents the presence of an attribute in the dataset.

3.2. Performance Measurement Parameters. This section describes the calculation mechanism of each performance measurement parameter with a short description, where $|y_i - y|$ is the absolute error, n is the number of errors, T_j is the goal value for record Ji, P_{ij} is the value of forecast by the specific model I for record j (out of n records), TP is the number of true-positive classification, FN is the number of false-negative classification, TN is the number of true-negative classification, and FP is the number of false-positive classifications:

(A) MAE is the average of all absolute errors. It can be calculated as

$$\text{MAE} = \frac{1}{2}\sum_{j=1}^{n}\left|y_i - y\right|. \tag{1}$$

(B) RMSE is the quadratic scoring statute that similarly measures the average magnitude error that can be calculated as

$$\text{RMSE} = \sqrt{\frac{1}{2}\sum_{j=1}^{n}\left(y_i - 1\right)^2}. \tag{2}$$

(C) RAE is the same as a modest predictor, which is simply the average of the real values and can be found as

$$\text{RAE} = \frac{\sum_{j=1}^{n}\left|P_{ij} - T\right|}{\sum_{j=1}^{n}\left|T_j - \underline{T}\right|}. \tag{3}$$

(D) RRSE is known as the square root of the relative squared error (RSE) that mostly decreases the error to similar dimensions as the quantity being predicted. It can be found as

$$\text{RRSE} = \sqrt{\frac{\sum_{j=1}^{n}\left(P_{ij} - T_j\right)^2}{\sum_{j=1}^{n}\left(T_j - T\right)^2}}. \tag{4}$$

(E) Specificity (also called the true-negative rate) measures the proportion of actual negatives that are correctly identified as such (e.g., the percentage of healthy instances that are correctly identified as not having the condition). It can be calculated as

$$\text{specificity} = \frac{\text{TN}}{\text{FP} + \text{TN}}. \tag{5}$$

(F) Precision is the number of positive predictions divided by the total number of positive class values expected. It is also called the positive predictive value (PPV). It can be calculated as

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{6}$$

(G) Recall is defined as the ratio of TP modules with high opinion to the total number of positive modules. It can be found as

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{7}$$

(H) F-measure is also called the F-score. F1-score conveys the balance between precision and recall. It can be measured as

$$\text{FM} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{8}$$

(I) G-measure conveys the balance between the specificity and the recall. It can be calculated as

$$\text{GM} = \frac{2 * \text{recall} * \text{specificity}}{\text{recall} + \text{specificity}}. \tag{9}$$

(J) MCC is a correlation coefficient calculated using all four values in the confusion matrix. This can be found as

$$\text{MCC} = \frac{(\text{TN} * \text{TP}) - (\text{FN} * \text{FP})}{\sqrt{(\text{FP} + \text{TP})(\text{FN} + \text{TP})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}. \tag{10}$$

(K) Accuracy points to how much the forecast is accurate and can be calculated as

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{11}$$

3.3. Summarization of Employed Techniques. ML techniques are currently extensively used to excerpt significant knowledge commencing a massive volume of data in diverse areas. ML applications embrace numerous real-world situations such as cybersecurity, bioinformatics, detecting communities in social networks, and software process enhancement to harvest high-quality software systems [7]. ML as well as TF-ML-based solutions for SDP have also been investigated [6, 10, 34]. The following subsections briefly discuss TF-ML techniques employed in this research.

3.3.1. Credal Decision Tree. Credal Decision Tree (CDT) is a technique to design classifiers grounded on inexact possibilities and improbability measures [18]. Throughout the creation procedure of a CDT, to sidestep producing a too-problematical decision tree, a new standard was presented: stop once the total improbability increases due to the splitting of the decision tree. The function used in the total uncertainty dimension can be fleetingly articulated as in [14, 19].

3.3.2. Cost-Sensitive Decision Forest. CS-Forest practices cost-sensitive pruning as a substitute for the pruning used by C4.5. C4.5 prunes a tree if the probable number of

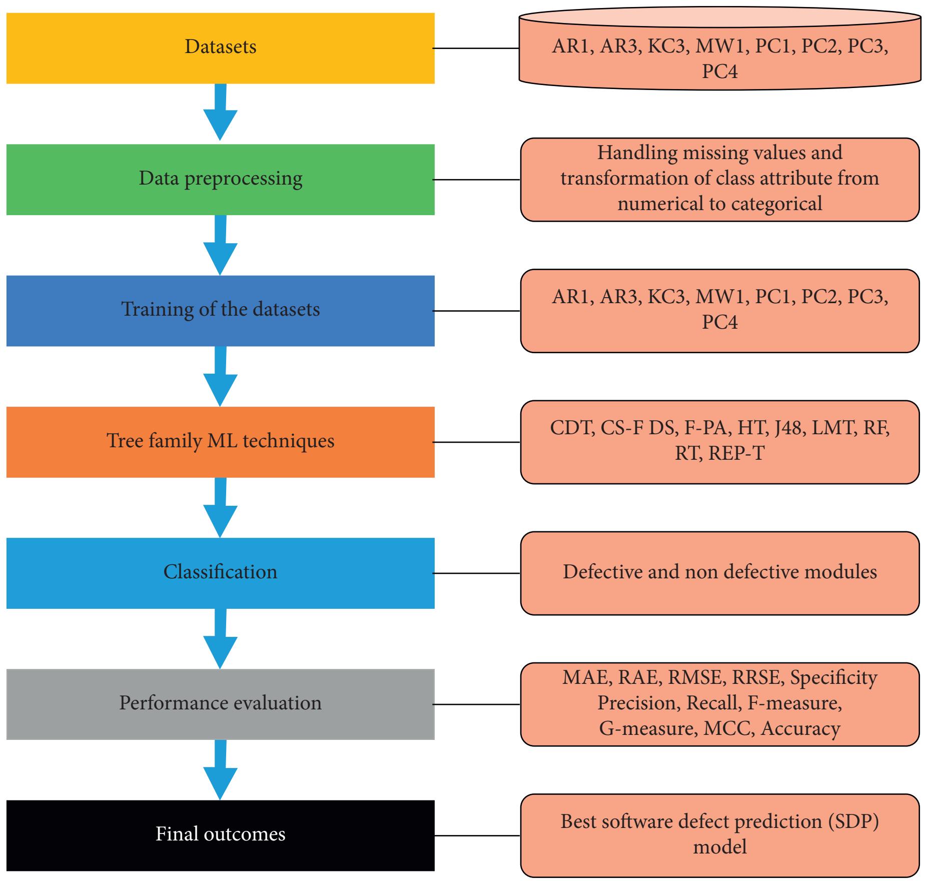
Datasets
AR1, AR3, KC3, MW1, PC1, PC2, PC3, PC4
Data preprocessing
Handling missing values and transformation of class attribute from numerical to categorical
Training of the datasets
AR1, AR3, KC3, MW1, PC1, PC2, PC3, PC4
Tree family ML techniques
CDT, CS-F DS, F-PA, HT, J48, LMT, RF, RT, REP-T
Classification
Defective and non defective modules
Performance evaluation
MAE, RAE, RMSE, RRSE, Specificity Precision, Recall, F-measure, G-measure, MCC, Accuracy
Final outcomes
Best software defect prediction (SDP) model

Figure 1: Methodology work flow.

Table 1: Summary of the literature survey.

Author	Technique/model	Datasets	Evaluation measures
Czibula et al. [11]	RAD	PC1, PC2, PC3, PC4, KC1, KC3, MC2, JM1, MW1, and CM1	Accuracy, specificity, precision, PD, and ROC
Li et al. [20]	DP-CNN	Camel, jEdit, Lucene, xalam, Xerces, synapse, and poi	FM
Jacob and Raju [21]	HFS, NB, NN, RF, RT, J48	PC1, PC2, PC3, PC4, CM1, MW1, KC3, and JM1	Specificity, sensitivity, MCC, and accuracy
Bashir et al. [22]	NB, RF, KNN, MLP, SVM, J48, and decision stump	CM1, JM1, KC2, MC1, PC1, and PC5	ROC
Miholca et al. [7]	HyGRAR	JEdit 4.2, JEdit 4.0, Anr 1.7, JEdit 4.3, Tomcat 6.0, AR1, AR3, AR4, AR5, and AR6	Accuracy, sensitivity, specificity, and precision
Alsaeedi and Khan [8]	Bagging, SVM, DT, and RF	PC1, PC3, PC4, PC5, JM1, KC2, KC3, MC1, MC2, and CM1	GM, specificity, F-score, recall, precision, and accuracy
Iqbal et al. [9]	OneR, NB, K*, MLP, RBF, SVM, KNN, DT, PART, and RF	PC1, PC2, PC3, PC4, PC5, KC1, KC3, CM1, JM1, MW1, MC1, and MC2	MCC, ROC area, FM, recall, precision, and accuracy
Malhotra and Kamal [6]	J48, RF, NB, AdaBoost, bagging, and SPIDER3	NASA datasets	Accuracy, sensitivity, specificity, and precision
Manjula and Florence [23]	GA, DNN, NB, RF, DT, ABC, SVM, and KNN	KC1, KC2, CM1, PC1, and JM1	Precision, sensitivity, specificity, recall, F-score, and accuracy

misclassification for forthcoming records does not increase expressively due to the pruning. However, CS-Forest prunes a tree if the probable classification cost for forthcoming records does not increase expressively due to the pruning. Moreover, unlike Cost-Sensitive Decision Tree (CS-Tree), CS-Forest tolerates a tree to first completely develop and then get pruned [40].

Table 2: Attributes, instances, defective, and nondefective modules of each utilized dataset.

S. no.	Datasets	No. of attributes	No. of instances	No. of defective modules		No. of nondefective modules		Data type
1	AR1	30	121	9	7.4%	112	92.6%	Numerical
2	AR3	30	63	8	12.7%	55	87.3%	Numerical
3	CM1	22	498	49	9.8%	449	90.2%	Numerical
4	KC2	22	522	107	20.5%	415	79.5%	Numerical
5	KC3	40	194	36	18.6%	158	81.4%	Numerical
6	MW1	41	403	31	8.0%	372	92.0%	Numerical
7	PC1	22	1109	77	6.9%	1032	93.1%	Numerical
8	PC2	41	5589	23	0.5%	5566	99.5%	Numerical
9	PC3	41	1563	160	10.1%	1403	89.9%	Numerical
10	PC4	41	1458	178	12.2%	1280	87.8%	Numerical

Table 3: List of attributes according to datasets.

Attributes		Datasets									
		AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
	Halstead content	Y	Y	—	–	Y	Y	–	Y	Y	Y
	Halstead difficulty	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Halstead effort	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Halstead error estimator	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Halstead length	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Halstead attributes	Halstead level	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Halstead program time	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Halstead volume	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Number of operands	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Number of operators	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Number of unique operands	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Number of unique operators	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Essential complexity	–	–	Y	Y	Y	Y	Y	Y	Y	Y
McCabe attributes	Cyclomatic complexity	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Design complexity	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Cyclomatic density	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Number of lines	–	–	Y	Y	Y	Y	Y	Y	Y	Y
	LOC total	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Size attributes	LOC executable	Y	Y	–	–	Y	Y	–	Y	Y	Y
	LOC comments	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	LOC code and comments	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	LOC blank	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Branch count	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Condition count	Y	Y	–	–	Y	Y	–	Y	Y	Y
	EDGE count	–	–	–	–	Y	Y	–	Y	Y	Y
	Parameter count	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Modified condition count	–	–	–	–	Y	Y	–	Y	Y	Y
	Multiple condition count	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Node count	–	–	–	–	Y	Y	–	Y	Y	Y
	Design density	Y	Y	–	–	Y	Y	–	Y	Y	Y
Other attributes	Essential density	–	–	–	–	Y	Y	–	Y	Y	Y
	Decision count	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Decision density	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Call pairs	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Global data complexity	–	–	–	–	Y	Y	–	Y	Y	Y
	Global data density	–	–	–	–	Y	Y	–	Y	Y	Y
	Maintenance severity	–	–	Y	Y	Y	Y	Y	Y	Y	Y
	Normalized cyclomatic complexity	Y	Y	–	–	Y	Y	–	Y	Y	Y
	Pathological complexity	–	–	–	–	–	Y	–	Y	Y	Y
	Percent comments	–	–	Y	Y	Y	Y	Y	Y	Y	Y
Class attribute	Defective	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

3.3.3. Decision Stump. DS is utilized as a base learner to construct ensemble models. DS is an ML model encompassing a one-level decision tree. This ensemble learning performs 1000 repetitions for accomplishing optimal performance [41]. DS is essentially decision trees with a solitary label. A stump is divergent to a tree that has various layers. It mostly stops after the first split. DS is commonly utilized in large data. Almost not, they also serve to create modest yes/no decision models for smaller datasets [39].

3.3.4. Forest by Penalizing Attributes. Forest-PA technique uses bootstrap samples and penalized attributes. It purposes to construct a set of highly accurate decision trees by manipulating the strong point of all nonclass attributes presented in a data set, not like certain current techniques that use a subset of the nonclass attributes. At a similar time to support robust assortment, Forest-PA enforces disadvantages (disadvantageous weights) to an individual's attributes that contributed to the newest tree to produce the subsequent trees. Forest-PA, moreover, has a contrivance to increase weights step by step from the attributes that have not been tested in the subsequent tree(s) [42].

3.3.5. Hoeffding Tree. HT is identified as the streaming decision tree generation. The term is resulting from the Hoeffding bound that is utilized in tree generation. The elementary idea is Hoeffding bound delivers a specific level of assurance on the finest attribute to riven the tree, which can be the baseline to create the finest model [39].

3.3.6. Decision Tree (J48). This is the basic C4.5 Decision Tree (DT) used for classification problems [37]. It is the deviation of information gain (IG), usually utilized to stun the result of biasness. An attribute with a maximum gain ratio is nominated in direction to shape a tree as a splitting attribute. Gain ratio- (GR-) based DT performs well as compared to IG, in terms of accuracy [43].

3.3.7. Logistic Model Tree. LMTs are classification trees using logistic regression functions at the leaves. This technique can compact with dualistic and multiclass objective variables, nominal and numeric attributes, and missing values. LMT is a classification model with an attendant supervised training technique. It syndicates decision tree learning and logistic forecasts. Logistic model trees use a decision tree that has linear regression models at its leaves to deliver fragment-wise linear regression model [39].

3.3.8. Random Forest. RF produces a set of techniques that involve constructing an ensemble or so-termed as a forest of decision trees from a randomized variation of tree induction techniques [44]. RF works by forming a mass of decision trees at the training period and outputting the class in the approach of the classes output by a single tree. It is deliberated as one of the utmost techniques which are extremely proficient for both classification and regression problems [45].

3.3.9. Random Tree. RT is the supervised and collective learning technique that creates numerous solitary learners. It uses a grasping idea to construct a set of random data for building a tree. In the standard tree, nearby every node is split using the best split amid all variables. In the RF, each node is divided utilizing a best amid the subset of predictors randomly selected at that node [46].

3.3.10. REP-Tree. REP-T uses a regression tree that produces numerous trees in diverse repetitions. Afterward, it chooses the best one from all created trees. REP-T constructs a decision/regression tree using entropy as an adulteration measure and prunes it employing reduced-error pruning. It merely sorts of values for numeric attributes [46].

4. Experimental Study

This section provides an experimental study for SDP employing ten ML techniques using a 10-fold cross-validation method which is a standard approach for assessment [47]. 10-fold cross-validation is the process that splits the complete data into ten subsets of equal sizes; one subset is used for testing, while others are used for training. This process is continued until each subset has been used for testing. The overall experiments are divided into three phases; these are experimental scenarios 1, 2, and 3. Experimental scenario 1 represents the analysis of CCI, ICI, TPR, and FPR, while experimental scenario 2 presents the performance analysis of absolute and squared error rates that are MAE, RAE, RMSE, and RRSE accordingly. However, experimental scenario 3 describes the performance achieved using accurate measurements that are specificity, precision, recall, F-measure, G-measure, MCC, and accuracy.

4.1. Experimental Scenario 1: Instances Classification and True-Positive and False-Positive Rates. Here, in this section, experiments carried out to find correctly classified and incorrectly classified instances are presented, as well as the true-positive and false-positive rate of each classifier over each solitary dataset. Tables 4 and 5 show the CCI and ICI analyses achieved, while Tables 6 and 7, respectively, present the TPR and FPR values of each technique on an individual dataset. In each of the mentioned tables, the first row represents the dataset utilized, while the first column represents the techniques employed. The rest of the table represents the outcome of CCI, ICI, TPR, and FPR, respectively.

The observation from CCI shows that RF correctly classified the instances on five datasets that are AR3, PC1, PC2, PC3, and PC4; CDT and HT do the same for three datasets, DS and REP-T do the same for two datasets, while other rest of CF-Forest and RT performs the same for only one dataset individually. In the case of ICI, each technique showed the same performance as CCI and ICI are contrasting each other. Figure 2 shows the inclusive analysis of CCI and ICI. However, the situation has changed for TPR and FPR. Calculating TPR, RF shows the best performance on five datasets, CDT, DS, HT, and LTM outperforms on three datasets individually, REP-T on two datasets, while Forest-PA and J48 do the same only on one individual dataset. However, measuring FPR, Forest-PA performs well on four datasets, DS, HT, and REP-T outperform on three datasets, respectively, CDT and RF show the best performance on two individual datasets, while J48 and LMT outperform only on one individual dataset. The overall analysis showed that RF performs well while calculating CCI, ICI, and TPR, while calculating FPR, Forest-PA outperformed other techniques. Figure 3 shows the inclusive analysis of TPR and FPR.

4.2. Experimental Scenario 2 (Error Rate Analysis and Results Discussion). This section describes all the error rates achieved utilizing TF-ML techniques on different datasets. Tables 8 and 9 show the absolute errors MAE and RAE, respectively. Firstly, we consider MAE to measure MAE where J48 outperformed other techniques and achieved best results on five datasets; RT achieved best results on four datasets and RF on two datasets, while CDT, DS, Forest-PA, and HT performed well only on one solitary dataset. Secondly, if we consider RAE, likewise MAE J48 outperformed other techniques achieving best results on four datasets, RT on three datasets, RF on two, while HT surpassed other techniques on one dataset. Figures 4 and 5 show the error bar with a standard deviation of MAE and RAE, respectively.

Error bar is a graphical demonstration of the inconsistency of data and used on graphs to specify the uncertainty or error in a described measurement. It provides an overall indication of how accurate a measurement is or, on the other hand, how distant from the described value the true (error free) value might be. Here, these analyses show the best performance of J48 and RT while reducing absolute error rates.

Tables 10 and 11 present the analysis of squared errors that are RMSE and RRSE. In each table, the first row represents the datasets, while the first column represents the employed techniques. The rest of the table cells shows the outcomes of each employed technique on individual datasets. On both squared error measures RRSE and RMSE, RF achieved better results on five datasets that are AR3, KC3, PC1, PC3, and PC4. LMT outperforms other techniques on two datasets, while CDT, DS, Forest-PA, and REP-T do the same only on one individual dataset. Figures 6 and 7 represent the error bar with a standard deviation of RMSE and RRSE. These outcomes present the best performance of RF to reduce the squared error rate.

4.3. Experimental Scenario 3 (Accuracy Analysis and Results Discussion). To measure the performance of any technique, accuracy is considered as one of the most important evaluation measures. Here, in this section, we present different measurements for accuracy that are specificity, precision, recall, F-measure, G-measure, MCC, and accuracy. All these measurements depend on the values of the confusion matrix shown in Table 12. There are two types of classes in which prediction is possible, i.e., class 1 (positive) and class 2 (negative). Class 1 represents that there is defect in the software, while Class 2 represents that there is no defect in the software. Here, TP is the case in which the software has positive (they have the defect) and FP is also the case of positive, but they do not actually have the defect, and it is also called type 1 error. FN is the negative cases, but they actually do have the defect, and it is also known as type 2 error. TN is a negative case, which shows that they do not have the defect.

Table 13 presents the specificity assessment of all employed techniques on various datasets. In this table, column one presents the list of techniques, while the rest of the columns represents the specificity achieved on an individual dataset. Instead of some values, there is a categorical message “#DIV/0!,” which is due to the “0” value in the confusion matrix. According to different equations, if there is a need to divide some values and that value becomes “0,” at that time as we know that “0” is not divisible, we will have this message. Here, in the tables, we used “?” instead of “#DIV/0!.”

The analysis of Table 13 shows that measuring specificity, DS outperforms on AR3 and KC3, J48 outperforms on AR1 and PC4, LMT outperforms on KC2 and MW1, while RF shows better performance on CM1 and PC3. CS-Forest and Forest-PA outperform other techniques on PC2 and PC1, respectively.

Table 14 presents the overall analysis of precision achieved by TF-ML techniques on each dataset. The outcomes show that, on one individual dataset, several techniques perform better. As we consider the AR1 dataset, three techniques perform well that are CDT, HT, and REP-T, while on CM1, PC1, and PC3 datasets, DS and HT produce the same results, whereas on the PC2 dataset, seven techniques show the same results outperforming the rest of the three techniques. However, on AR3, KC2, KC3, MW1, and PC4 datasets, CDT, LMT, DS, HT, and J48, respectively, outperform other techniques.

Table 15 shows the recall assessments of each technique. The analysis shows that CDT produces better results only on the KC3 dataset while RF does the same only on the AR3 dataset. However, CS-Forest outperforms other techniques and shows the best performance on eight datasets that are AR1, CM1, KC2, MW1, PC1, PC2, PC3, and PC4. This analysis recommended the CS-Forest technique for calculating recall. For better understanding of analysis taken over specificity, precision, and recall, Figures 8–10 represent the error bar and standard deviation lines correspondingly for specificity, precision, and recall.

The F-measure analysis is presented in Table 16 where RF generates better results on four datasets, DS and HT generate better outcomes on three datasets, respectively, while CDT, J48, LMT, and REP-T do the same for two datasets, respectively. However, Forest-PA outperforms other techniques only on one dataset. If we consider the PC2 dataset, seven techniques produce the same results and likewise in the case of precision and MAE too. Now a question may arise here that why more techniques present

Table 4: Correctly classified instances analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**112**	55	445	433	**159**	372	1037	**5566**	1394	1300
CS-Forest	103	53	411	413	158	356	1011	5562	1325	1296
DS	110	57	449	416	**159**	367	1032	**5566**	1403	1280
Forest-PA	111	56	448	436	155	371	1037	**5566**	1402	1313
HT	**112**	53	**449**	435	157	372	1032	**5566**	1403	1284
J48	109	55	438	425	154	371	1035	**5566**	1390	1303
LMT	111	55	444	**440**	154	**375**	1025	5565	1393	1317
RF	109	**58**	444	435	158	371	**1039**	**5566**	**1409**	**1322**
RT	108	55	415	422	137	349	1010	5544	1337	1264
REP-T	**112**	55	444	426	154	369	1038	**5566**	1401	1308

Table 5: Incorrectly classified instances analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**9**	8	53	89	**35**	31	72	**23**	169	158
CS-Forest	18	10	87	109	36	47	98	27	238	162
DS	11	6	49	106	**35**	36	77	**23**	160	178
Forest-PA	10	7	50	86	39	32	72	**23**	161	145
HT	**9**	10	**49**	87	37	31	77	**23**	160	174
J48	12	8	60	97	40	32	74	**23**	173	155
LMT	10	8	54	**82**	40	**25**	84	24	170	141
RF	12	5	54	87	36	32	**70**	**23**	**154**	**136**
RT	13	8	83	100	57	54	99	45	226	194
REP-T	9	8	54	96	40	34	71	**23**	162	150

Table 6: True-positive rate analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**0.926**	0.873	0.894	0.83	**0.82**	0.923	0.935	**0.996**	0.892	0.892
CS-Forest	0.851	0.841	0.825	0.791	0.814	0.883	0.912	0.995	0.848	0.889
DS	0.909	0.905	**0.902**	0.797	**0.82**	0.911	0.931	**0.996**	0.898	0.878
Forest-PA	0.917	0.889	0.9	0.835	0.799	0.921	0.935	**0.996**	0.897	0.901
HT	**0.926**	0.841	**0.902**	0.833	0.809	0.923	0.931	**0.996**	0.898	0.881
J48	0.901	0.873	0.88	0.814	0.794	0.921	0.933	**0.996**	0.889	0.894
LMT	0.917	0.873	0.892	**0.843**	0.794	**0.931**	0.924	**0.996**	0.891	0.903
RF	0.901	**0.921**	0.892	0.833	0.814	0.921	**0.937**	**0.996**	**0.901**	**0.907**
RT	0.893	0.873	0.833	0.808	0.706	0.866	0.911	0.992	0.855	0.867
REP-T	**0.926**	0.873	0.892	0.816	0.794	0.916	0.936	**0.996**	0.896	0.897

Table 7: False-positive rate analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	0.926	**0.873**	0.902	0.439	0.663	0.834	0.69	**0.996**	0.871	0.562
CS-Forest	0.625	0.45	0.583	0.206	0.814	0.631	0.523	0.953	0.355	0.78
DS	0.927	0.548	0.902	0.337	0.534	0.747	**0.931**	**0.996**	0.898	**0.878**
Forest-PA	0.926	0.657	0.902	0.479	**0.818**	**0.923**	0.822	**0.996**	**0.97**	0.609
HT	0.926	0.557	0.902	0.466	0.816	**0.923**	**0.931**	**0.996**	0.898	0.849
J48	0.723	0.446	0.849	0.422	0.562	0.775	0.714	**0.996**	0.649	0.368
LMT	0.926	0.659	0.885	0.484	0.626	0.775	0.895	**0.996**	0.882	0.565
RF	**0.928**	0.332	0.848	0.431	0.707	0.746	0.654	**0.996**	0.731	0.531
RT	0.724	0.446	0.673	0.459	0.689	0.691	0.584	0.953	0.664	0.497
REP-T	0.926	0.766	**0.903**	**0.526**	0.69	0.894	0.69	**0.996**	0.859	0.522

the same and good results on the PC2 dataset? The riposte here is that this is due to the PC2 dataset which contains a very less number of defective modules that are only 0.5%. If we consider the rest of the datasets, no one can have less than 6.9% defective modules which is why the performance of each technique is different on these datasets.

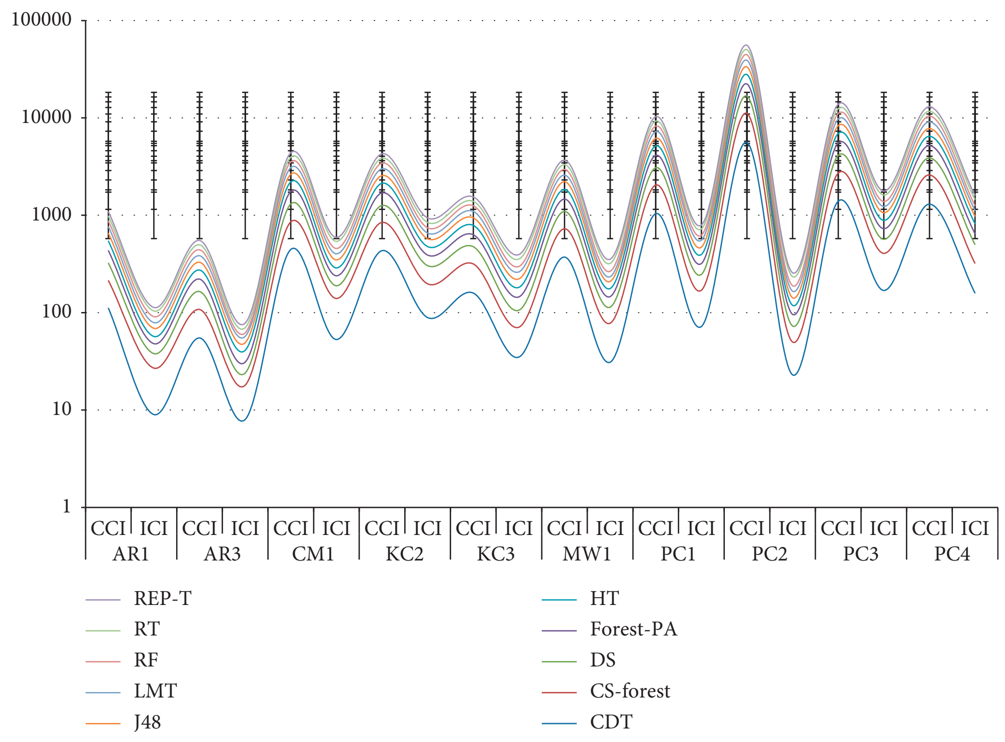

Figure 2: Overall analyses of CCI and ICI.

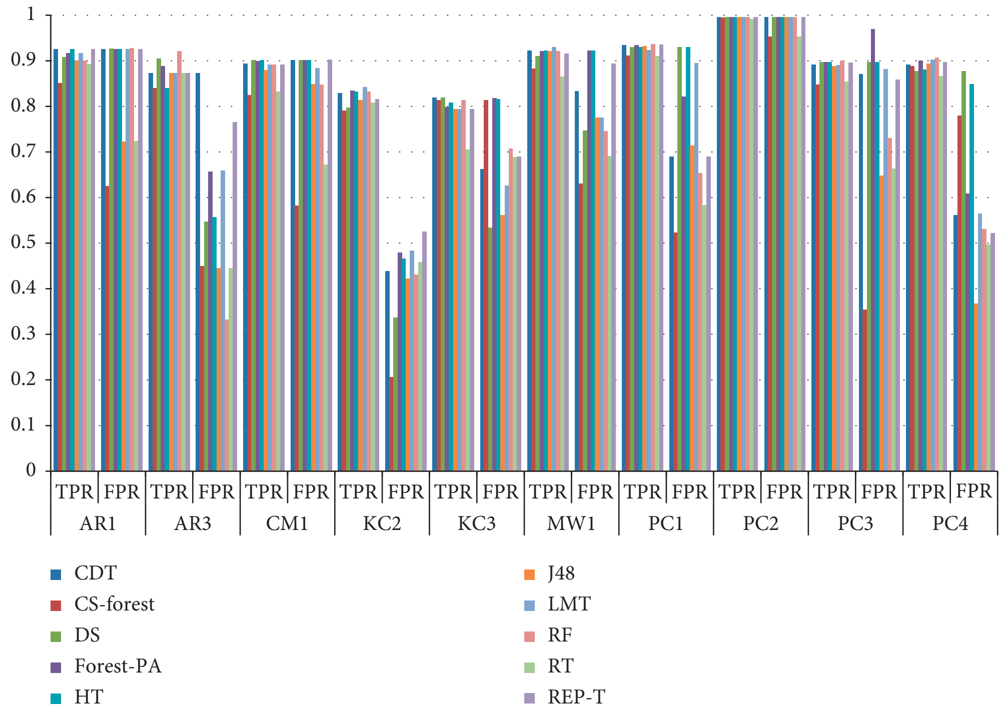

Figure 3: Inclusive analyses of TPR and FPR.

Table 17 shows the G-measure results. The outcome assessments show the best performance of CS-Forest, LMT, and RF for two individual datasets that are PC2 and PC4, KC2 and MW1, and CM1 and PC3 correspondingly, while CDT, DS, Forest-PA, and J48 beat other techniques on KC3, AR3, PC, and AR1 datasets, respectively. Furthermore,

Table 8: MAE analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	0.138	0.209	0.175	0.23	0.28	0.129	0.103	**0.008**	0.163	0.14
CS-Forest	0.184	0.254	0.218	0.292	0.299	0.156	0.127	0.009	0.183	0.244
DS	0.144	0.131	0.171	0.24	0.264	0.126	0.121	**0.008**	0.166	0.171
Forest-PA	0.139	0.157	0.167	0.225	0.269	0.127	0.113	**0.008**	0.163	0.149
HT	0.144	0.21	0.179	**0.165**	0.298	0.144	0.13	0.009	0.184	0.202
J48	0.127	0.161	0.176	0.237	**0.237**	**0.117**	0.105	**0.008**	**0.139**	**0.115**
LMT	0.436	0.191	0.367	0.229	0.259	0.12	0.194	0.356	0.302	0.141
RF	0.127	0.148	**0.163**	0.221	0.257	0.126	0.095	**0.008**	0.148	0.133
RT	**0.112**	**0.127**	0.168	0.206	0.294	0.135	**0.092**	**0.008**	0.144	0.134
REP-T	0.138	0.215	0.181	0.235	0.297	0.135	0.103	**0.008**	0.163	0.131

Table 9: RAE analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	95.475	90.104	97.589	70.29	91.982	89.352	79.525	97.671	88.203	64.957
CS-Forest	127.68	109.45	121.64	89.521	98.066	108.64	97.525	100.87	99.451	113.5
DS	100.05	56.431	95.674	73.372	86.727	87.145	92.794	95.159	90.149	79.504
Forest-PA	96.114	67.856	93.351	68.794	88.439	88.169	86.666	97.098	88.415	69.191
HT	100	90.312	100	**50.341**	97.885	100	100	104.2	100	94.236
J48	87.977	69.248	98.213	72.675	**77.888**	**81.303**	80.946	97.671	**75.478**	**53.491**
LMT	302.3	82.477	204.91	70.219	85.175	83.399	149.57	4238.8	163.77	65.806
RF	87.961	63.786	**91.195**	67.493	84.379	87.357	72.995	**93.456**	80.296	61.913
RT	**77.286**	**54.752**	93.743	63.146	96.462	93.925	**71.115**	99.202	78.198	62.266
REP-T	95.475	92.809	101.17	71.812	97.438	94.043	79.355	97.671	88.505	60.846

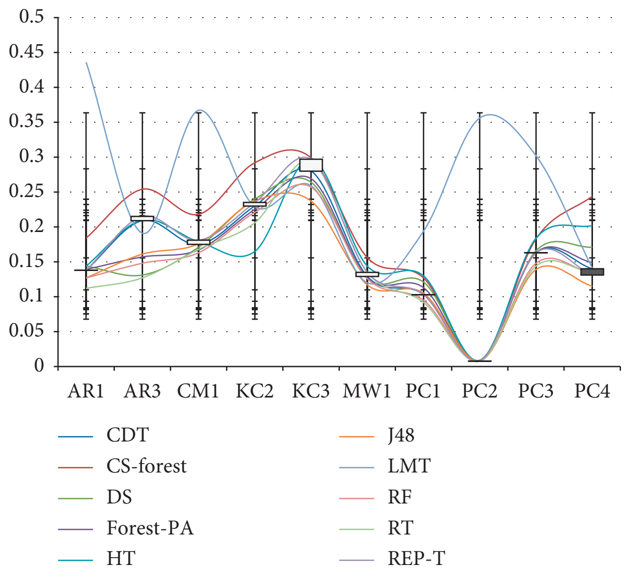

Figure 4: MAE analysis with error and standard deviation bar.

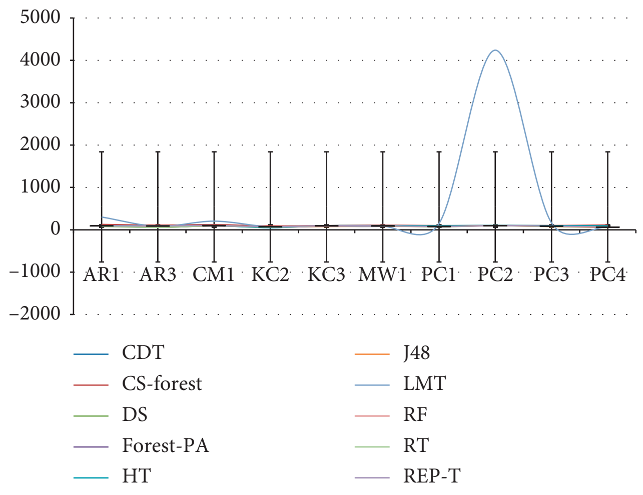

Figure 5: RAE analysis with error and standard deviation bar.

Table 18 represents the analysis of MCC measurement achieved using utilized TF-ML techniques. The outcomes show that utilizing CM1, KC2, PC2, and PC3 datasets, CS-Forest outperforms other techniques producing better results. Going on the KC3 dataset, the performance of the DS technique is better than the rest of the utilized techniques, while proceeding AR1 and PC4 datasets, the performance of J48 beats other employed techniques. Moreover, the performance of LMT is better then other techniques employing on MW1 datasset, while on AR3 and PC1 dataset, RF outperform well instead of other techniques.

Accuracy assessments of all employed TF-ML techniques are shown in Table 19. The outcome achieved on AR3, KC3, and PC2 datasets presents that CDT outperforms other techniques in terms of achieving a higher accuracy. Moreover, on AR1 and PC2 datasets, REP-T also outperforms other techniques while achieving the same results as CDT. Going on the PC2 dataset, CDT, REP-T well DS, Forest-PA, J48, and RF also produce the same results. This is due to the very less number of defective modules in this dataset. However, on CM1 and KC3 datasets, DS beats other employed techniques in tenure of achieving better accuracy results, as well as proceeding CM1 dataset; HT also

Table 10: RMSE analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**0.2627**	0.3377	0.3046	0.3627	0.3818	0.2605	0.2358	0.064	0.2971	0.278
CS-Forest	0.2985	0.3449	0.3307	0.379	0.3766	0.2919	0.2504	0.0645	0.3088	0.3036
DS	0.2995	0.3046	0.297	0.3569	0.3712	0.2565	0.2457	**0.0631**	0.2908	0.2923
Forest-PA	0.2667	0.2836	**0.292**	0.3422	0.3709	0.2538	0.2349	0.642	0.2867	0.2687
HT	0.2628	0.3871	0.2979	0.402	0.3987	0.2665	0.2542	0.0639	0.3031	0.3236
J48	0.2997	0.3424	0.3301	0.3968	0.43	0.2751	0.2441	0.064	0.3151	0.299
LMT	0.4646	0.3254	0.4322	**0.339**	0.3918	**0.243**	0.315	0.4199	0.3917	0.2683
RF	0.2856	**0.2724**	0.2951	0.349	**0.3667**	0.2613	**0.2223**	0.0647	**0.2715**	**0.247**
RT	0.3309	0.3563	0.4089	0.4392	0.542	0.3652	0.3014	0.0899	0.3786	0.3652
REP-T	**0.2627**	0.3438	0.3102	0.3733	0.4093	0.275	0.2365	0.064	0.2944	0.2768

Table 11: RRSE analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**99.9593**	100.959	102.252	89.8344	98.1628	97.7466	92.7793	99.9995	97.9979	84.9229
CS-Forest	113.601	103.118	111.028	93.8871	96.8295	109.526	98.4938	100.808	101.864	92.7438
DS	113.975	91.0648	99.7001	88.4132	95.4458	96.2335	96.6546	**98.5476**	95.9299	89.2681
Forest-PA	101.48	84.7947	**98.0292**	84.7724	95.3681	95.2364	92.397	100.288	94.5733	82.0605
HT	100	115.734	100	99.572	102.503	100	100	99.9618	100	98.8565
J48	114.05	102.391	110.822	98.2924	110.557	103.215	96.0121	99.9995	103.953	91.331
LMT	176.823	97.3056	145.11	**83.9701**	100.743	**91.1714**	123.907	655.858	129.23	81.9477
RF	108.688	**81.4368**	99.0872	86.4543	**94.2824**	98.0574	**87.469**	101.053	**89.569**	**75.4394**
RT	125.935	106.549	137.265	108.7792	139.372	137.037	118.574	140.375	124.894	111.563
REP-T	**99.9593**	102.791	104.148	92.4714	105.23	103.187	93.0209	99.9995	97.118	84.557

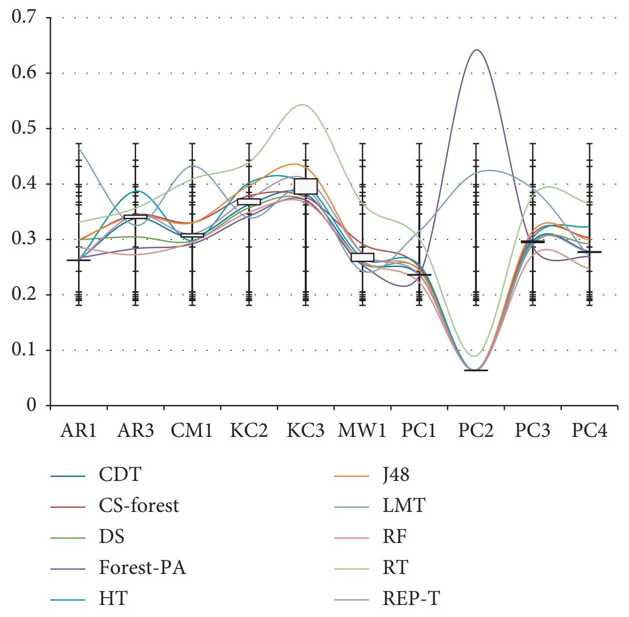

Figure 6: RMSE analyses with error and standard deviation bar.

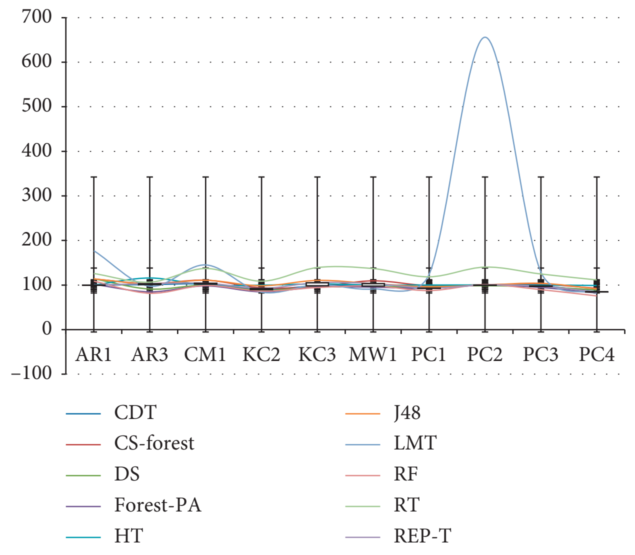

Figure 7: RRSE analyses with error and standard deviation bar.

Table 12: Confusion matrix.

	Positive 1	Negative 0
Positive 1	TP	FP
Negative 0	FN	TN

outperforms other techniques that turn out the same results such as DS. Furthermore, on AR3, PC1, PC3, PC4, and PC2 as well, RF performs better results to achieve a higher accuracy than other utilized techniques. The overall accuracy analysis suggests RF for a better measurement of accuracy. Moreover, the error bar and standard deviation line for the better understanding of F-measure, G-measure, MCC, and accuracy are presented in Figures 11–14 individually. The average accuracy is demonstrated in Figure 15 for the better understanding of outcomes analysis of each technique going through each utilized dataset.

5. Discussion on Overall Performance

A popular way to compare the overall performances of classifiers is to count (w) the number of data sets on which an algorithm is an overall winner, also known as the Count of Wins test. We have used 10 datasets, and no technique has given the best results for at least 10 datasets at $\alpha = 0.05$, according to the critical values in Table 3 of [48]. Since the Count of Wins test is also considered to be a weak testing procedure, therefore, we have a detailed matrix in Table 20, keeping in view Scenario 2. Table 20 shows the determined analysis of all evaluated absolute and squared error rates. It represents the measurement of absolute errors that are MAE and RAE on the employed datasets and TF-ML techniques. The results concluded that J48 and RT outperform other techniques. However, evaluating on squared errors that are RMSE and RRSE, the RF technique beats the rest of the techniques in terms of reducing squared error. Moreover, Table 21 shows the inclusive outcomes of experimental scenario 3. For specificity, the outcomes show the best performance of J48, DS, RF, and LMT for two datasets individually. On precision, HT beats other techniques on five datasets and DS on four datasets, while CDT beats others on only two datasets. At the moment of considering recall, CF-Forest shows the best performance on eight datasets, while on F-measure, RF outperforms the rest of the techniques, and on G-measure, RF, LMT, and CS-Forest perform better results on two individual datasets. Making an allowance for MCC, CS-Forest outperforms on four datasets and J48 and RF on two individual datasets. Finally, if we discuss the accuracy, RF produces the best results on five datasets, CDT on three datasets, while DS on two datasets. However, the performance of RF for error rate as well as for accuracy is better than the rest of the utilized TF-ML techniques.

Generally, we can say that the more trees in the forest the more robust the forest looks like. In the same way in the RF classifier, the higher the number of trees in the forest gives the high accuracy results. In other words, it is believed that RF ensures as an ensemble method of numerous trees, enhanced to knob categorical data when gaining the ultimate solution in the widely held voting system for the outcomes of respective trees is umpired [33, 45, 49]. RF not only delivers a dual classification of data facts and nevertheless also delivers the prospects for each factor to be appropriate to defective or nondefective categories [50]. It is deliberated as one of the utmost dominant techniques as it is extremely proficient in the accomplishment of both regression and classification [44].

5.1. Friedman Two-Way Analysis of Variance by Ranks.

To compare all applied ML techniques on multiple data sets, we have applied the statistical procedure as described by Sheskin [51], García, and Herrera (2008) [52]. The Friedman two-way analysis of variance by ranks (Friedman (1937) [53] is adopted with rank-order data in a hypothesis testing situation. A significant test indicates that there is a significant difference between at least two of the techniques in the set of k techniques. Friedman test checks whether the measured average ranks are significantly different from the mean rank (in our case, $Rj = 3.96$). The chi-square (χ^2) distribution is used to approximate the Friedman test statistic [51]. Friedman's statistic is $\chi^2 = 139.7985$.

To reject the null hypothesis, the computed value must be equal to or greater than χ^2 the tabled (table of the chi-square distribution) critical chi-square value at the prespecified level of significance [51]. The number of degrees of freedom $\mathrm{df} = k - 1$; thus, $\mathrm{df} = 10 - 1 = 9$. For $\mathrm{df} = 9$, the tabled critical $\alpha = 0.05$ chi-square values are $= 16.92$. Since the computed value $= 139.7985$ is greater than $\chi^2_{0.05} = 16.92$, the alternative hypothesis is supported at $\alpha = 0.05$. It can be concluded that there is a significant difference among at least nine of the ten ML techniques. This result can be summarized as follows: $\chi^2_{0.05}\ (9) = 139.7985,\ p < 0.05$.

Since the critical value is lower than the χ^2, we can continue with the post hoc tests to spot the significant pairwise differences among all the techniques. The results are shown in Table 22, where z is the corresponding statistics and p values for each hypothesis. Z is computed using the following equation:

$$z = \frac{(Ri - Rj)}{\mathrm{SE}}, \tag{12}$$

where Ri is the ith technique and the standard error is $\mathrm{SE} = \sqrt{(k(k+1)/6n)} = 0.\ 0.175$. Columns 5th and 6th represent Nemenyi's and Holm's statics procedure. The second last column lists the differences between the average ranks of ith and jth techniques. However, the last column shows the critical difference (CD), and it states that the performance of the two techniques is significantly different if the corresponding average ranks differ by at least the CD. CD can be calculated using the following equation:

$$\mathrm{CD} = q\alpha\sqrt{\frac{k(k+1)}{6n}}, \tag{13}$$

where critical values $q\alpha$ is given in Table 5(b) in (Demsar 2006) [48]. The notations "significant" and "insignificant" represent whether the difference in the average rank (Ri-Rj) is greater or less than the value of CD, respectively. Greater means a significant difference between two means. Here, the value of CD is $= 0.485$.

In Table 22, the family of hypotheses is ordered by their p values. As can be seen, Nemenyi's procedure rejects the first 28 hypotheses, whereas Holm's procedure also rejects the next 3 hypotheses since the corresponding p values are smaller than the adjusted NMα's and Holm. Therefore, we conclude that the performance of HT and LMT is comparable, and RT resulted in a lower performance. Besides, the obtained value $\mathrm{CD} = 0.485$ indicates that any difference between the average ranks of two techniques that is equal to or greater than 0.485 is significant. Concerning the pairwise comparisons in Table 22, the difference between the average ranks of two techniques which are greater than $\mathrm{CD} = 0.485$ is the first 33. Thus, it can be concluded that there is a significant difference between the average ranks of the first 33 pairs of techniques.

Table 13: Specificity analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	?	?	0	0.6098	0.8398	0.5	0.5714	?	0.2632	0.9162
CS-Forest	0.2	0.4	0.2432	0.4942	0.8144	0.2778	0.3820	**0.1667**	0.3587	0.8898
DS	0	**0.75**	?	0.5038	**0.8639**	0.3530	?	?	?	0.8779
Forest-PA	0	0.6667	0	0.6567	0.8115	0	**0.6923**	?	0.4545	0.9109
HT	?	0.375	?	0.6389	0.8135	?	?	?	?	0.8814
J48	**0.2857**	0.5	0.1765	0.5521	0.8554	0.4546	0.5455	?	0.4369	**0.9426**
LMT	0	0.5	0.1429	**0.7049**	0.843	**0.7143**	0.2308	0	0.1875	0.9166
RF	0	0.7143	**0.2727**	0.622	0.8315	0.4615	0.5897	?	**0.5556**	0.9212
RT	0.25	0.5	0.2167	0.5393	0.8176	0.2051	0.362	0.0417	0.2829	0.9229
REP-T	?	0.5	0	0.5846	0.8315	0.2	0.5882	?	0.4375	0.9216

Table 14: Precision analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**1**	**1**	0.9911	0.9229	0.1944	0.9919	0.9855	**1**	0.99	0.3652
CS-Forest	0.8929	0.8909	0.8753	0.7904	0	0.9301	0.9467	0.9991	0.8738	0.1124
DS	0.9821	0.9818	**1**	0.8434	**0.3611**	0.9704	**1**	**1**	**1**	0
Forest-PA	0.9911	0.9818	0.9978	0.9446	0	0.9973	0.9961	**1**	0.9957	0.309
HT	**1**	0.9091	**1**	0.9373	0	**1**	**1**	**1**	**1**	0.0337
J48	0.9554	0.9273	0.9688	0.8964	0.3333	0.9839	0.9855	**1**	0.9587	**0.5899**
LMT	0.9911	0.9636	0.9866	**0.9566**	0.25	0.9946	0.9903	0.9998	0.9907	0.3596
RF	0.9732	0.9636	0.9822	0.9253	0.1389	0.9812	0.9845	**1**	0.9829	0.3989
RT	0.9464	0.9273	0.8953	0.9012	0.1944	0.9167	0.9506	0.9959	0.9223	0.4438
REP-T	**1**	0.9818	0.9889	0.9349	0.1667	0.9892	0.9864	**1**	0.9936	0.4101

Table 15: Recall analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	0.9256	0.873	0.9008	0.8705	**0.5385**	0.9295	0.9469	0.9959	0.8996	0.5909
CS-Forest	**0.9434**	0.9245	**0.9269**	**0.9371**	?	**0.9428**	**0.9578**	**0.9961**	**0.9526**	**0.8333**
DS	0.9244	0.9153	0.9016	0.8951	0.52	0.9352	0.9306	0.9959	0.8976	?
Forest-PA	0.925	0.9	0.9014	0.8615	0	0.9229	0.938	0.9959	0.9001	0.7143
HT	0.9256	0.9091	0.9016	0.8644	0	0.9231	0.9306	0.9959	0.8976	0.75
J48	0.9386	0.9273	0.9044	0.8732	0.4286	0.9337	0.9452	0.9959	0.9212	0.5615
LMT	0.925	0.8983	0.9022	0.8612	0.4091	0.9343	0.9325	0.9959	0.8985	0.7033
RF	0.9237	**0.9464**	0.9055	0.8727	0.5	0.9359	0.9495	0.9959	0.9139	0.71
RT	0.9381	0.9273	0.9178	0.8637	0.2	0.9368	0.9534	0.996	0.9171	0.454
REP-T	0.9256	0.8852	0.9006	0.849	0.375	0.9246	0.947	0.9959	0.9011	0.6186

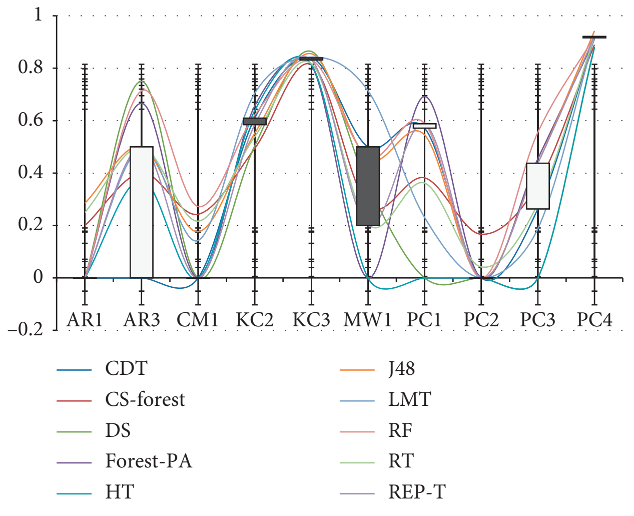

Figure 8: Specificity analyses with error and standard deviation bar.

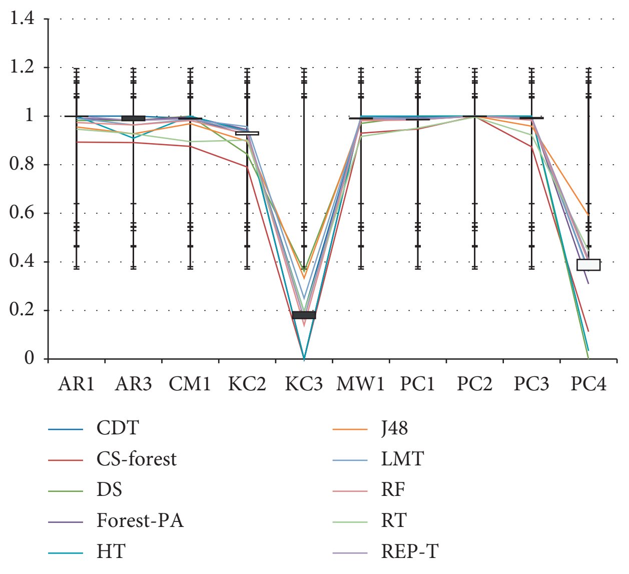

Figure 9: Precision analyses with error and standard deviation bar.

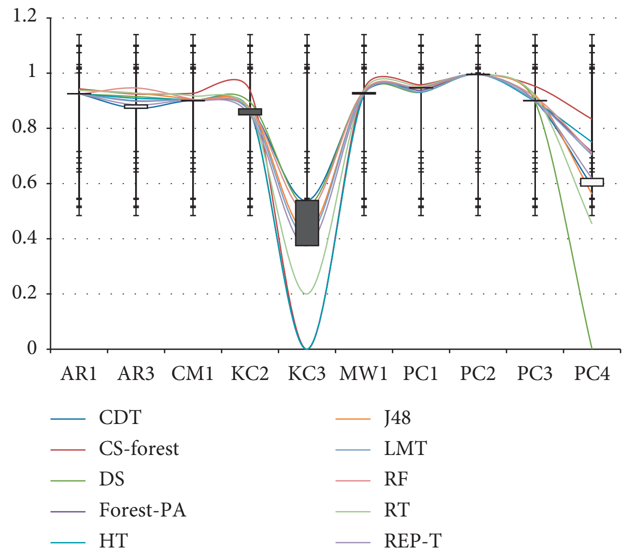

Figure 10: Recall analyses with error and standard deviation bar.

Table 16: F-measure analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	**0.9614**	0.9322	0.9438	0.8959	0.2857	0.9597	0.9658	**0.9979**	0.9427	0.4514
CS-Forest	0.9174	0.9074	0.9003	0.8575	?	0.9364	0.9522	0.9976	0.9115	0.198
DS	0.9524	0.9474	**0.9483**	0.8685	**0.4262**	0.9525	0.964	**0.9979**	0.9461	?
Forest-PA	0.9569	0.9391	0.9471	0.9011	?	0.9587	0.9662	**0.9979**	0.9455	0.4314
HT	**0.9614**	0.9091	**0.9483**	0.8994	?	0.96	0.964	**0.9979**	0.9461	0.0645
J48	0.9469	0.9273	0.9355	0.8847	0.375	0.9581	0.9649	**0.9979**	0.9396	**0.5753**
LMT	0.9569	0.9298	0.9426	**0.9064**	0.3103	**0.9635**	0.9605	0.9978	0.9424	0.4758
RF	0.9478	**0.955**	0.9423	0.8982	0.2174	0.958	**0.9667**	**0.9979**	**0.9471**	0.5108
RT	0.9422	0.9273	0.9064	0.8821	0.1972	0.9266	0.952	0.996	0.9197	0.4489
REP-T	**0.9614**	0.931	0.9427	0.8899	0.2308	0.9558	0.9663	**0.9979**	0.9451	0.4932

Table 17: G-measure analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	?	?	0	0.7171	**0.6562**	0.6502	0.7128	?	0.4072	0.7184
CS-Forest	0.33	0.5584	0.3854	0.6471	?	0.4291	0.546	**0.2856**	0.5212	**0.8607**
DS	0	**0.8244**	?	0.6447	0.6492	0.5125	?	?	?	?
Forest-PA	0	0.766	0	0.7453	0	0	**0.7966**	?	0.6041	0.8007
HT	?	0.531	?	0.7347	0	?	?	?	?	0.8104
J48	**0.4381**	0.6497	0.2953	0.6765	0.571	0.6114	0.6917	?	0.5927	0.7038
LMT	0	0.6424	0.2467	**0.7752**	0.5509	**0.8096**	0.37	0	0.3103	0.7959
RF	0	0.8141	**0.4192**	0.7263	0.6245	0.6182	0.7276	?	**0.691**	0.8019
RT	0.3948	0.6497	0.3506	0.664	0.3214	0.3366	0.5253	0.08	0.4324	0.6086
REP-T	?	0.6391	0	0.6924	0.5169	0.3289	0.7257	?	0.589	0.7403

Table 18: MCC analysis by each TF-ML technique on individual dataset.

Technique	AR1	AR3	CM1	KC2	KC3	MW1	PC1	PC2	PC3	PC4
CDT	?	?	−0.0297	0.4329	0.2433	0.1952	0.3565	?	0.0588	0.4091
CS-Forest	0.1801	0.3562	**0.2032**	**0.5022**	?	0.2361	0.3633	**0.0832**	**0.3916**	0.2811
DS	−0.0367	0.4872	?	0.4285	**0.3309**	0.2174	?	?	?	?
Forest-PA	−0.0259	0.3624	−0.0148	0.4294	−0.0598	−0.0144	0.2669	?	0.0978	0.4271
HT	?	0.2841	?	0.4299	−0.0344	?	?	?	?	0.1425
J48	**0.1996**	0.4273	0.0493	0.4082	0.2567	0.2374	0.328	?	0.2931	**0.5148**
LMT	−0.0259	0.2917	0.0178	0.4505	0.2056	**0.318**	0.0691	−0.0009	0.0286	0.4581
RF	−0.0452	**0.6236**	0.088	0.4459	0.1886	0.2635	**0.3908**	?	0.2828	0.4873
RT	0.1781	0.4273	0.147	0.3755	0.0174	0.1575	0.3215	0.0385	0.1955	0.3732
REP-T	?	0.2029	-0.0333	0.3547	0.1461	0.0518	0.363	?	0.1124	0.4501

Table 19: Results of different TF-ML techniques in terms of accuracy along with the rank values.

Datasets	CDT	CS-forest	DS	Forest-PA	HT	J48	LMT	RF	RT	REP-T
AR1	92.56 (1.3)	85.12 (6)	90.90 (3)	91.73 (2.5)	92.56 (1.3)	90.08 (4.5)	91.73 (2.5)	90.08 (4.5)	89.25 (5)	92.56 (1.3)
AR3	87.30 (4.2)	84.12 (5.2)	90.4762 (2)	88.88 (3)	84.12 (5.2)	87.30 (4.2)	87.30 (4.2)	92.06 (1)	87.30 (4.2)	87.30 (4.2)
CM1	89.35 (3)	82.53 (7)	90.16 (1.2)	89.95 (2)	90.16 (1.2)	87.95 (5)	89.15 (4.3)	89.15 (4.3)	83.33 (6)	89.15 (4.3)
KC2	82.95 (4)	79.11 (9)	79.69 (8)	83.52 (2)	83.33 (3.5)	81.41 (6)	84.29 (1)	83.33 (3.5)	80.84 (7)	81.60 (5)
KC3	81.95 (1.2)	81.44 (2.5)	81.95 (1.2)	79.89 (4)	80.92 (3)	79.38 (5.3)	79.38 (5.3)	81.44 (2.5)	70.61 (6)	79.38 (5.3)
MW1	92.30 (2.5)	88.33 (6)	91.06 (5)	92.05 (3.3)	92.30 (2.5)	92.05 (3.3)	93.05 (1)	92.05 (3.3)	86.60 (7)	91.56 (4)
PC1	93.50 (3.5)	91.16 (7)	93.05 (5.2)	93.50 (3.5)	93.05 (5.2)	93.32 (4)	92.42 (6)	93.68 (1)	91.07 (8)	93.59 (2)
PC2	99.58 (1.14)	99.51 (3)	99.58 (1.14)	99.58 (1.14)	99.58 (1.14)	99.58 (1.14)	99.57 (2)	99.58 (1.14)	99.19 (4)	99.58 (1.14)
PC3	89.18 (5)	84.77 (9)	89.76 (2.5)	89.69 (3)	89.76 (2.5)	88.93 (7)	89.12 (6)	90.14 (1)	85.54 (8)	89.63 (4)
PC4	89.16 (6)	88.88 (7)	87.79 (9)	90.05 (3)	88.06 (8)	89.36 (5)	90.32 (2)	90.67 (1)	86.69 (10)	89.71 (4)
Sum (rank)	31.84	61.7	38.24	27.44	33.54	45.44	34.3	23.24	65.2	35.24
Average (rank)	3.18	6.17	3.82	2.74	3.35	4.54	3.43	2.32	6.52	3.52
Sum (Acc)	897.88	865.02	894.46	898.91	893.89	889.41	896.36	902.23	860.45	894.10
Average (Acc)	89.78	86.50	89.44	89.89	89.38	88.94	89.63	90.22	86.04	89.41

It ranks the technique for each data set separately, the best performing algorithm getting the rank of 1 and the second-best rank 2. Last two columns present the sum and average of ranks for each technique.

Table 20: Decision table for experimental scenario 2.

Datasets	Evaluation measurements			
	MAE	RAE	RMSE	RRSE
AR1	RT	RT	CDT	CDT
AR3	RT	RT	RF	RF
CM1	RF	RF	Forest-PA	Forest-PA
KC2	HT	HT	LMT	LMT
KC3	J48	J48	RF	RF
MW1	J48	J48	LMT	LMT
PC1	RT	RT	RF	RF
PC2	Seven techniques	RF	DS	DS
PC3	J48	J48	RF	RF
PC4	J48	J48	RF	RF

Table 21: Decision table for experimental scenario 3.

Datasets	Evaluation measurements						
	Specificity	Precision	Recall	F-measure	G-measure	MCC	Accuracy
AR1	J48	CDT, HT, REP-T	CS-Forest	CDT, HT, REP-T	J48	J48	CDT, REP-T
AR3	DS	CDT	RF	RF	DS	RF	RF
CM1	RF	DS, HT	CS-Forest	DS, HT	RF	CS-Forest	DS, HT
KC2	LMT	LMT	CS-Forest	LMT	LMT	CS-Forest	LMT
KC3	DS	DS	CDT	DS	CDT	DS	CDT, DS
MW1	LMT	HT	CS-Forest	LMT	LMT	LMT	LMT
PC1	Forest-PA	DS, HT	CS-Forest	RF	Forest-PA	RF	RF
PC2	CS-Forest	Seven techniques	CS-Forest	Seven techniques	CS-Forest	CS-Forest	Seven techniques
PC3	RF	DS, HT	CS-Forest	RF	RF	CS-Forest	RF
PC4	J48	J48	CS-Forest	J48	CS-Forest	J48	RF

Table 22: Family of hypotheses ordered by the p value and adjusting α by Nemenyi and Holm's procedures, considering an initial $\alpha = 0.05$.

Sr. no.	Algo	Algo	z	p	Nm 0.05	Holm	*Ri-Rj*	Inference from CD
1	RF	RT	24.00437415	$1.81E-09$	0.001	0.0011111	4.196	Significant
2	CS-Forest	RF	22.00210271	$3.91E-09$	0.001	0.0011364	3.846	Significant
3	Forest-PA	RT	21.60164842	$4.60E-09$	0.001	0.0011628	3.776	Significant
4	CS-Forest	Forest-PA	19.59937698	$1.09E-08$	0.001	0.0011905	3.426	Significant
5	CDT	RT	19.08450719	$1.37E-08$	0.001	0.0012195	3.336	Significant
6	HT	RT	18.11197535	$2.17E-08$	0.001	0.00125	3.166	Significant
7	LMT	RT	17.6771964	$2.69E-08$	0.001	0.0012821	3.09	Significant
8	RT	REP-T	17.1394435	$3.53E-08$	0.001	0.0013158	2.996	Significant
9	CDT	CS-Forest	17.08223575	$3.63E-08$	0.001	0.0013514	2.986	Significant
10	CS-Forest	HT	16.10970391	$6.06E-08$	0.001	0.0013889	2.816	Significant
11	CS-Forest	LMT	15.67492497	$7.69E-08$	0.001	0.0014286	2.74	Significant
12	DS	RT	15.42321084	$8.86E-08$	0.001	0.0014706	2.696	Significant
13	CS-Forest	REP-T	15.13717207	$1.04E-07$	0.001	0.0015152	2.646	Significant
14	CS-Forest	DS	13.42093941	$2.95E-07$	0.001	0.0015625	2.346	Significant
15	J48	RF	12.70012169	$4.74E-07$	0.001	0.0016129	2.22	Significant
16	J48	RT	11.30425246	$1.28E-06$	0.001	0.0016667	1.976	Significant
17	Forest-PA	J48	10.29739596	$2.80E-06$	0.001	0.0017241	1.8	Significant
18	CS-Forest	J48	9.301981021	$6.51E-06$	0.001	0.0017857	1.626	Significant
19	DS	RF	8.581163303	$1.26E-05$	0.001	0.0018519	1.5	Significant
20	CDT	J48	7.780254728	$2.76E-05$	0.001	0.0019231	1.36	Significant
21	RF	REP-T	6.864930643	$7.35E-05$	0.001	0.002	1.2	Significant
22	HT	J48	6.807722887	$7.84E-05$	0.001	0.0020833	1.19	Significant
23	J48	LMT	6.372943947	$1.29E-04$	0.001	0.0021739	1.114	Significant
24	LMT	RF	6.327177742	$1.37E-04$	0.001	0.0022727	1.106	Significant
25	DS	Forest-PA	6.178437578	$1.63E-04$	0.001	0.002381	1.08	Significant
26	HT	RF	5.892398802	$2.31E-04$	0.001	0.0025	1.03	Significant
27	J48	REP-T	5.835191046	$2.48E-04$	**0.001**	0.0026316	1.02	Significant
28	CDT	RF	4.919866961	$8.25E-04$	0.001	0.0027778	0.86	Significant
29	Forest-PA	REP-T	4.462204918	$1.57E-03$	0.001	0.0029412	0.78	Significant
30	DS	J48	4.118958386	$2.60E-03$	0.001	0.003125	0.72	Significant
31	Forest-PA	LMT	3.924452017	$3.49E-03$	0.001	0.0033333	0.686	Significant
32	CDT	DS	3.661296343	$5.22E-03$	0.001	0.0035714	0.64	Significant
33	Forest-PA	HT	3.489673077	$6.83E-03$	0.001	0.0038462	0.61	Significant
34	DS	HT	2.688764502	$2.48E-02$	0.001	0.0041667	0.47	Insignificant
35	CDT	Forest-PA	2.517141236	$3.29E-02$	0.001	0.0045455	0.44	Insignificant
36	Forest-PA	RF	2.402725725	$3.97E-02$	0.001	0.005	0.42	Insignificant
37	DS	LMT	2.253985561	$5.07E-02$	0.001	0.0055556	0.394	Insignificant
38	CS-Forest	RT	2.002271437	$7.63E-02$	0.001	0.00625	0.35	Insignificant
39	CDT	REP-T	1.945063682	$8.36E-02$	0.001	0.0071429	0.34	Insignificant
40	DS	REP-T	1.716232661	$1.20E-01$	0.001	0.0083333	0.3	Insignificant
41	CDT	LMT	1.407310782	$1.93E-01$	0.001	0.01	0.246	Insignificant
42	CDT	HT	0.972531841	$3.56E-01$	0.001	0.0125	0.17	Insignificant
43	HT	REP-T	0.972531841	$3.56E-01$	0.001	0.0166667	0.17	Insignificant
44	LMT	REP-T	0.5377529	$6.04E-01$	0.001	0.025	0.094	Insignificant
45	HT	LMT	0.434778941	$6.74E-01$	0.001	0.05	0.076	Insignificant

5.2. *Threats to Validity.*

In this section, we converse the effects that may anguish the validity of this research work.

5.2.1. *Internal Validity.*

The exploration of this paper is grounded on diverse very familiar evaluation standards that are used in the past in various studies. Amid these standards, several methods are used to assess the error rate, while certain approaches are used to assess accuracy. So, the treat can be that the renewal of new evaluation standards as a replacement for utilized standards can decrease the accuracy. Furthermore, the techniques used in this research can be supplanted with some newer techniques that can be hybridized with each other and can harvest enhanced outcomes than the employed techniques.

5.2.2. *External Validity.*

We piloted investigations on various datasets. A threat to validity may arise if we relate the projected techniques in the other real data composed from the different software development organizations using surveys etc. or replace these datasets with some other datasets, which may distress the outcomes while growing the error rates. Likewise, the projected technique may not be capable of harvesting better forecasts in outcomes using some other SDP datasets. Hence, this study concentrated on AR1, AR3, CM1, KC2, KC3, MW1,

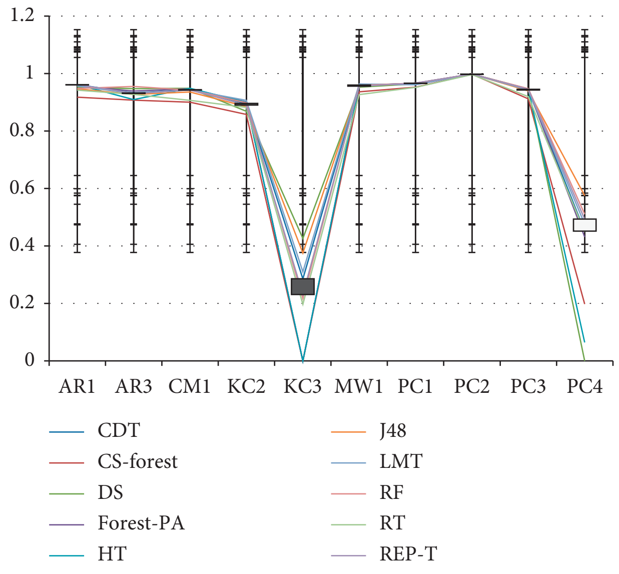

Figure 11: F-measure analyses with error and standard deviation bar.

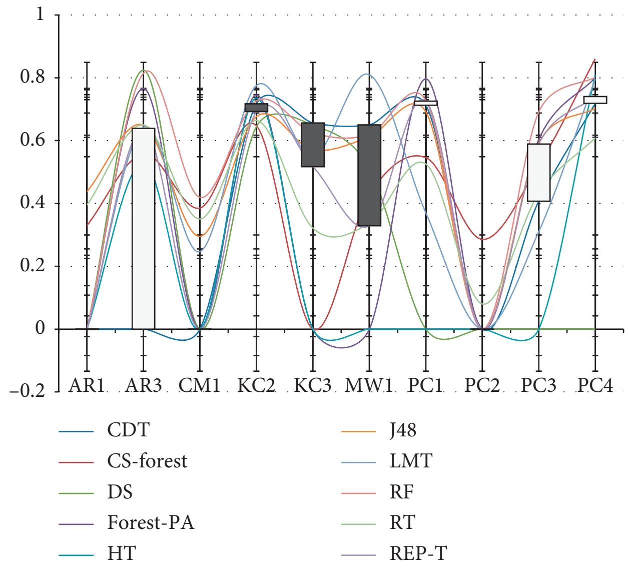

Figure 12: G-measure analysis with error and standard deviation bar.

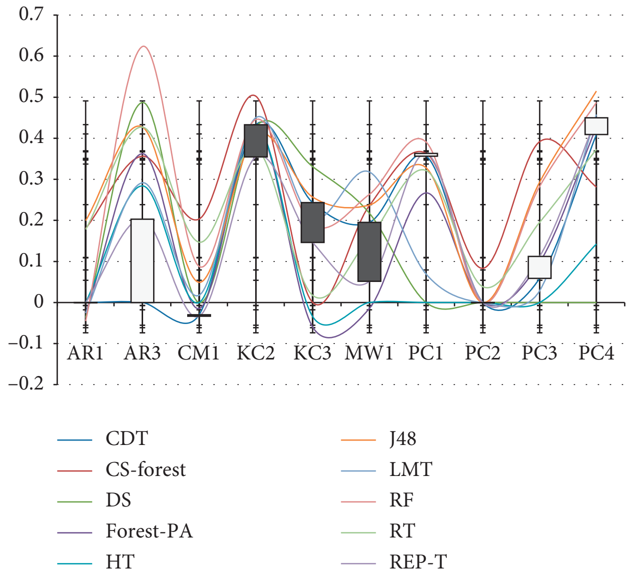

Figure 13: MCC analyses with error and standard deviation bar.

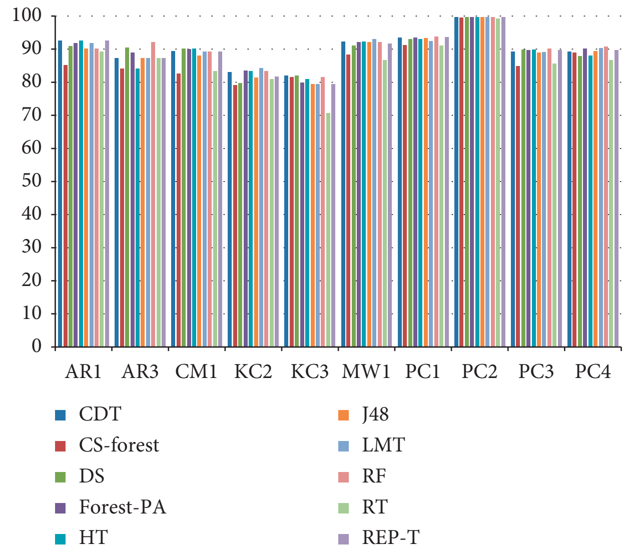

Figure 14: Inclusive analysis of accuracy achieved by individual TF-ML technique.

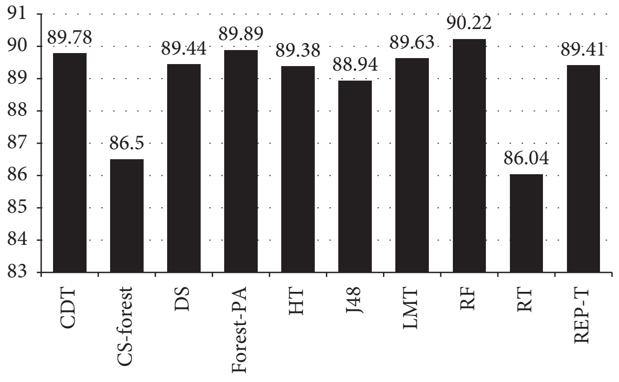

Figure 15: Overall average-accuracy achieved by individual TF-ML technique.

PC1, PC2, PC3, and PC4 datasets to measure the performance of the utilized techniques.

5.2.3. Construct Validity. In this study, diverse TF-ML techniques are benchmarked with each on various datasets based on several assessment measures. The assortments of techniques utilized in this study are at the center of their progressive features over the other techniques that have been exploited by the researchers in the last decades. However, the threat can be that if we put on some other new techniques, then it can be the probability that these new techniques can exhaust the projected techniques. Furthermore, any change in the dataset splitting (increasing or decreasing the number of K-Folds) may change the current outcomes. It also can be promising that using the newest evaluation standards creates improved outcomes that can beat the current accomplished outcomes.

6. Conclusion

Nowadays, SDP using ML techniques is dignified as one of the emerging research areas. As the identification of software defects at the primary stage of SDLS is a challenging task, nevertheless it can subsidize the provision of high-quality software systems. This paper considered ten extensively used publically available datasets to compare ten famous TF-ML techniques: CDT, CS-Forest, DS, Forest-PA, HT, J48, LMT, RF, RT, and REP-T, which are broadly used for SDP. The performance is evaluated utilizing different measures such as MAE, RAE, RMSE, RRSE, specificity, precision, recall, FM, GM, MCC, and accuracy. The inclusive results of this paper recommended RF technique by providing the best results in terms of reducing error rates as well as increasing accuracy on five datasets that include AR3, PC1, PC2, PC3, and PC4, where the accuracy rate for each of these datasets is 92.0635%, 93.688%, 99.5885%, 90.1472%, and 90.6722%, respectively. However, CDT and DS are best in terms of increasing accuracy on three individual datasets. CDT accuracy outcomes are 92.562%, 81.9588%, and 99.5885% correspondingly going on AR1, KC3, and PC2, while DS shows an accuracy performance of 90.1606%, 81.9588%, and 99.5885% individually on CM1, KC3, and PC2.

The outcomes obtainable in this research can be recycled as a baseline for other studies and researchers so that the outcomes of any projected technique, model, or framework can be benchmarked and simply confirmed. For future work, class imbalance matters ought to be committed to these datasets. Furthermore, to increase the enactment, feature selection and ensemble learning techniques should also be explored.

Data Availability

The datasets used in this research are taken from UCI ML Learning Repository available at https://archive.ics.uci.edu/.

Conflicts of Interest

The authors declare that they have no conflicts of interest related to this study.

Acknowledgments

The authors would like to thank and acknowledge the support provided by the King Saud University, Riyadh, Saudi Arabia, through researchers supporting project number RSP-2020/184.

References

[1] A. O. Balogun, A. O. Bajeh, V. A. Orie, and A. W. Yusuf-asaju, "Software defect prediction using ensemble learning: an ANP based evaluation method," *Journal of Engineering Technology*, vol. 3, no. 2, pp. 50–55, 2018.

[2] T. Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener, "Defect prediction from static code features: current results, limitations, new approaches," *Automated Software Engineering*, vol. 17, no. 4, pp. 375–407, 2010.

[3] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*, vol. 38, no. 6, pp. 1276–1304, 2012.

[4] Z. Li, X.-Y. Jing, and X. Zhu, "Progress on approaches to software defect prediction," *IET Software*, vol. 12, no. 3, pp. 161–175, 2018.

[5] H. Wang, "Software defects classification prediction based on mining software repository," *IEEE Transactions on Software Engineering*, vol. 44, no. 6, pp. 534–550, 2014.

[6] R. Malhotra and S. Kamal, "An empirical study to investigate oversampling methods for improving software defect prediction using imbalanced data," *Neurocomputing*, vol. 343, pp. 120–140, 2019.

[7] D.-L. Miholca, G. Czibula, and I. G. Czibula, "A novel approach for software defect prediction through hybridizing gradual relational association rules with artificial neural networks," *Information Sciences*, vol. 441, pp. 152–170, 2018.

[8] A. Alsaeedi and M. Z. Khan, "Software defect prediction using supervised machine learning and ensemble techniques : a comparative study," *Journal of Software Engineering and Applications*, vol. 12, no. 5, pp. 85–100, 2019.

[9] A. Iqbal, S. Aftab, U. Ali et al., "Performance analysis of machine learning techniques on software defect prediction using NASA datasets," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, pp. 300–308, 2019.

[10] T. Menzies, A. Dekhtyar, J. Distefano, and J. Greenwald, "Problems with precision: a response to "comments on data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 9, pp. 637–640, 2007.

[11] G. Czibula, Z. Marian, and I. G. Czibula, "Software defect prediction using relational association rule mining," *Information Sciences*, vol. 264, pp. 260–278, 2014.

[12] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, no. 1, pp. 79–82, 2005.

[13] H. Alasker, S. Alharkan, W. Alharkan, A. Zaki, and L. S. Riza, "Detection of kidney disease using various intelligent classifiers," in *Proceedings of the 2017 3rd International Conference on Science in Information Technology (ICSITech)*, pp. 681–684, Bandung, Indonesia, October 2017.

[14] J. Abellán and A. R. Masegosa, "An ensemble method using credal decision trees," *European Journal of Operational Research*, vol. 205, no. 1, pp. 218–226, 2010.

[15] M. Bilal, H. Israr, M. Shahid, and A. Khan, "Sentiment classification of roman-Urdu opinions using Naïve Bayesian, decision tree and KNN classification techniques," *Journal of King Saud University-omputer and Information Sciences*, vol. 28, no. 3, pp. 330–344, 2016.

[16] K. Phakhounthong, P. Chaovalit, P. Jittamala et al., "Predicting the severity of dengue fever in children on admission based on clinical features and laboratory indicators: application of classification tree analysis," *BMC Pediatrics*, vol. 18, no. 1, 2018.

[17] S. Moral-García, C. J. Mantas, J. G. Castellano, M. D. Benítez, and J. Abellán, "Bagging of credal decision trees for imprecise classification," *Expert Systems with Applications*, vol. 141, p. 112944, 2020.

[18] C. J. Mantas and J. Abellán, "Credal decision trees in noisy domains," in *Proceedings of the 22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 683–688, Bruges, Belgium, 2014.

[19] Q. He, Z. Xu, S. Li et al., "Novel entropy and rotation forest-based credal decision tree classifier for landslide susceptibility modeling," *Entropy*, vol. 21, no. 2, 2019.

[20] J. Li, P. He, J. Zhu, and M. R. Lyu, "Software defect prediction via convolutional neural network," in *Proceedings of the 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, pp. 318–328, Prague, Czech Republic, July 2017.

[21] S. Jacob and G. Raju, "Software defect prediction in large space systems through hybrid feature selection and classification," *International Arab Journal of Information Technology*, vol. 14, no. 2, pp. 208–214, 2017.

[22] K. Bashir, T. Li, C. W. Yohannese, and Y. Mahama, "Enhancing software defect prediction using supervised-learning based framework," in *Proceedings of the 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, Nanjing, China, November 2017.

[23] C. Manjula and L. Florence, "Deep neural network based hybrid approach for software defect prediction using software metrics," *Cluster Computing*, vol. 22, no. 4, pp. 9847–9863, 2019.

[24] B. Khan, R. Naseem, M. Ali, M. Arshad, and N. Jan, "Machine learning approaches for liver disease diagnosing," *International Journal of Data Science and Advanced Analytics*, vol. 1, no. 1, pp. 27–31, 2019.

[25] S. A. Lauer, K. Sakrejda, E. L. Ray et al., "Prospective forecasts of annual dengue hemorrhagic fever incidence in Thailand, 2010-2014," *Proceedings of the National Academy of Sciences*, vol. 115, no. 10, pp. E2175–E2182, 2018.

[26] K. Balasaravanan and M. Prakash, "Detection of dengue disease using artificial neural network based classification techniquetion," *International Journal of Engineering and Technology*, vol. 7, no. 1, pp. 13–15, 2018.

[27] S. Chae, S. Kwon, and D. Lee, "Predicting infectious disease using deep learning and big data," *International Journal of Environmental Research and Public Health*, vol. 15, no. 8, 2018.

[28] C. G. Raji and S. S. Vinod Chandra, "Graft survival prediction in liver transplantation using artificial neural network models," *Journal of Computational Science*, vol. 16, pp. 72–78, 2016.

[29] N. L. Woolever, R J Schomberg, S Cai, R. A Dierkhising, A. S Dababneh, and R. C Kujak, "Pharmacist-driven MRSA Nasal PCR screening and the duration of empirical Vancomycin therapy for suspected MRSA respiratory tract

infections," *Mayo Clinic Proceedings: Innovations, Quality & Outcomes*, vol. 4, no. 5, pp. 550–556, 2020.

[30] A. Amini, O. Varsaneux, H. Kelly et al., "Diagnostic accuracy of tests to detect hepatitis B surface antigen: a systematic review of the literature and meta-analysis," *BMC Infectious Diseases*, vol. 17, no. 1, 2017.

[31] C. Davi, A. Pastor, T. Oliveira et al., "Severe dengue prognosis using human genome data and machine learning," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 10, pp. 2861–2868, 2019.

[32] H.-H. Rau, C.-Y. Hsu, Y.-A. Lin et al., "Development of a web-based liver cancer prediction model for type II diabetes patients by using an artificial neural network," *Computer Methods and Programs in Biomedicine*, vol. 125, pp. 58–65, 2016.

[33] H. Jin, S. Kim, and J. Kim, "Decision factors on effective liver patient data prediction," *International Journal of Bio-Science and Bio-Technology*, vol. 6, no. 4, pp. 167–178, 2014.

[34] J. Chen, Y. Yang, K. Hu, Q. Xuan, Y. Liu, and C. Yang, "Multiview transfer learning for software defect prediction," *IEEE Access*, vol. 7, pp. 8901–8916, 2019.

[35] Q. Song, Y. Guo, and M. Shepperd, "A comprehensive investigation of the role of imbalanced learning for software defect prediction," *IEEE Transactions on Software Engineering*, vol. 45, no. 12, pp. 1253–1269, 2019.

[36] H. Tong, B. Liu, and S. Wang, "Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning," *Information and Software Technology*, vol. 96, pp. 94–111, 2018.

[37] M. M. Saritas, "Performance analysis of ANN and naive Bayes classification algorithm for data classification," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 7, no. 2, pp. 88–91, 2019.

[38] C. Wu, S.-C. Kao, C.-H. Shih, and M.-H. Kan, "Open data mining for Taiwan's dengue epidemic," *Acta Tropica*, vol. 183, pp. 1–7, 2018.

[39] N. Nahar and F. Ara, "Liver disease prediction by using different decision tree techniques," *International Journal of Data Mining & Knowledge Management Process*, vol. 8, no. 2, 2018.

[40] M. J. Siers and M. Z. Islam, "Cost sensitive decision forest and voting for software defect prediction," *Lecture Notes in Computer Science*, vol. 8862, pp. 929–936, 2014.

[41] U. R. Acharya, H. Fujita, S. Bhat et al., "Decision support system for fatty liver disease using GIST descriptors extracted from ultrasound images," *Information Fusion*, vol. 29, pp. 32–39, 2016.

[42] N. Adnan and Z. Islam, "PT US CR," *Expert Systems With Applications*, vol. 89, 2017.

[43] S. Perveen, M. Shahbaz, K. Keshavjee, and A. Guergachi, "A systematic machine learning based approach for the diagnosis of non-alcoholic fatty liver disease risk and progression," *Scientific Reports*, vol. 8, no. 1, pp. 1–12, 2018.

[44] A. N. Arbain and B. Y. P. Balakrishnan, "A comparison of data mining algorithms for liver disease prediction on imbalanced data," *International Journal of Data Science and Analytics*, vol. 1, no. 1, 2019.

[45] A. Gulia, R. Vohra, and P. Rani, "Liver patient classification using intelligent techniques," vol. 5, no. 4, pp. 5110–5115, 2014.

[46] K. Shaukat Dar and S. M. Ulya Azmeen, "Dengue fever prediction: a data mining problem," *Journal of Data Mining Genomics Proteomics*, vol. 6, no. 3, 2015.

[47] K. A. Otunaiya and G. Muhammad, "Performance of data-mining techniques in the prediction of chronic kidney disease," *Computer Science and Information Technology*, vol. 7, no. 2, pp. 48–53, 2019.

[48] J. Dem̆, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[49] T. M. Carvajal, K. M. Viacrusis, L. F. T. Hernandez, H. T. Ho, D. M. Amalin, and K. Watanabe, "Machine learning methods reveal the temporal pattern of dengue incidence using meteorological factors in metropolitan Manila, Philippines," *BMC Infectious Diseases*, vol. 18, no. 1, pp. 1–15, 2018.

[50] D. Bowes, T. Hall, and J. Petrić, "Software defect prediction: do different classifiers find the same defects?" *Software Quality Journal*, vol. 26, no. 2, pp. 525–552, 2018.

[51] D. J. Sheskin, "Parametric and Nonparametric Statistical Procedures," *Chapman & Hall/CRC*, Boca Raton, FL, USA, 2000.

[52] S. Garc, "An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[53] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 37–41, 1937.