Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1028175, 14 pages
https://doi.org/10.1155/2022/1028175

Research Article

An Optimized Feature Selection Method Using Ensemble Classifiers in Software Defect Prediction for Healthcare Systems

Uzma Ghulam Mohammad,[1] Salma Imtiaz,[1] Manoj Shakya,[2] Ahmad Almadhor,[3] and Fareeha Anwar[1]

[1]*Department of Computer Science and Software Engineering, International Islamic University, Islamabad 44000, Pakistan*
[2]*Department of Computer Science and Engineering, Kathmandu University, Nepal*
[3]*College of Computer and Information Sciences, Al Jouf University, Saudi Arabia*

Correspondence should be addressed to Salma Imtiaz; salma.imtiaz@iiu.edu.pk and Manoj Shakya; manoj@ku.edu.np

Received 1 April 2022; Revised 17 May 2022; Accepted 24 May 2022; Published 27 June 2022

Academic Editor: Kuruva Lakshmanna

The healthcare systems are extensively being used with increased focus on safety of patients. Software engineering for healthcare applications is an emerging research area. Detecting defects is a critical step of software development process of healthcare applications. The performance of the Software Defect Prediction model (SDP) depends on the features of healthcare system; irrelevant features decrease the performance of the model. An optimized feature selection technique is needed to recognize and remove the irrelevant features. In this study, a new optimized feature selection technique, i.e., multiobjective Harris Hawk Optimization (HHO), is proposed for binary classification problem with Adaptive Synthetic Sampling (ADASYN) Technique. Multiobjective HHO is proposed with two main objectives, one to reduce the total amount of selected features and the other to maximize the performance of the proposed model. The multiobjective feature selection technique helps to find the optimal solution to achieve the desired objectives and increase the classification performance in terms of accuracy, AUC, precision, recall, and F1-measure. The study conducts an experiment on a healthcare dataset. Six different search techniques (RF, SVM, bagging, adaptive boosting, voting, and stacking) are implemented on the dataset. The proposed model helps to predict the software defects with a significant classification accuracy of 0.990 and AUC score of 0.992.

1. Introduction

Software consultants and vendors develop high-quality healthcare systems such as middle-ware medical devices, hospital management systems, and electronic systems used in medical domain [1]. In recent years, software applications are playing a vital role in every organization and business. Companies rely on software applications for handling their daily operations [2]. These software applications and systems hold critical importance in the healthcare domain due to severe consequences associated with their malfunction. Therefore, healthcare applications are based on design rules and best practices for high-quality applications [1, 2].

Software Development Life Cycle (SDLC) (abbreviations and acronyms given in Table 1) provides a basic set of rules that are used in design, development, and testing of different software applications [2]. However, with increase in size and complexity of software, ensuring quality via testing results in increased cost [1–3]. In recent years, the healthcare industry is growing in both high- and low-level income countries [4–6]. These systems face many challenges, such as elicitation of user requirements to development, testing, and deployment of software applications [1–7]. It is important to consider the design rules of users and methodologies to increase the quality of software applications. Software developers are unable to perform exhaustive testing for high-quality healthcare software applications; therefore, a value-based approach to testing is required. There is a need to identify and test the most defective parts of the system. SDP has become the most investigated area in the field of software quality [8, 9]. Software defect prediction is a formal approach that has many different models, processes, and

Table 1: Abbreviation table.

Abbreviation	Full form
SDP	Software defect prediction
SFP	Software fault prediction
FS	Feature selection
HHO	Harris Hawks Optimization
RF	Random Forest
SVM	Support Vector Machine
DT	Decision tree
SDLC	Software Development Life Cycle
ML	Machine learning
ADASYN	Adaptive Synthetic Sampling
SMOTH	Synthetic Minority Oversampling
KNN	*K*-Nearest Neighbor
BPSO	Binary Particle Swarm Optimization
BACO	Binary Ant Colony Optimization
BGA	Binary Genetic Algorithm
MOFES	Multiobjective feature selection
QSA	Queuing Search Algorithm
ANP	Analytic Network Process
LR	Logistic Regression
NB	Naïve Bayes
LDA	Linear Discriminant Analysis
L-RNN	Layered Recurrent Neural Network
MLP	Multilayer Perception
PMAs	Pareto multiobjective algorithms
IBDA	Improved binary dragonfly algorithm
AUC	Area under the Curve
ET	Extra Tree
ANN	Artificial Neural Network

assessment standards [10]. It suffers from many challenges such as different datasets, problems in extracting the best features for defect prediction, and insufficient prediction models [8, 10, 11]. The defect prediction designs are frequently used by different industries such as healthcare systems to help in fault predication and effort estimation required to ensure reliability of healthcare software's [1–11]. Recognizing and removing the software defects in healthcare systems is a resource intensive activity. Preliminary defect prediction results in timely defects correction [12]. The purpose of SDP is to predict the possible defects and features of healthcare software systems [13]. Software defect prediction performance is affected by representation of defective data features [14]. Therefore, it is important to remove the irrelevant features while designing the software model [15]. Feature selection indents for enhancing the accuracy of SDP models by dropping irrelevant features and decreasing the time and complications of these algorithms. Three different feature selection techniques are wrapper technique, embedded technique, and filter technique [16–18]. Filter technique assigns the score to all features of the dataset. Wrapper technique uses the classifiers to estimate the results of feature selection. The literature shows that the results of wrapper techniques are better than the filter techniques [16]. Embedded techniques consider the selection of features as part of the learning classifiers [16, 17]. The SDP models are based on three components: machine learning (ML) algorithms, soft computing, and metrics of software [14]. The procedure of establishing the metrics model of the software is associated with gathering metrics features for predicting the defects. This technique does not satisfactorily work with diverse projects or diverse versions. Therefore, the researchers apply change metrics of software to control the problem and make a precise software defect prediction. However, the technique is inappropriate and takes more time with complicated systems in large industries. Prediction of defects at an early stage of software implementation process helps to decrease the cost of implementation and computation [14–16]. The existing techniques only identify the defects in the typical code base [10]. This research is designed to test the following hypothesis.

(i) Alternative hypothesis (H1): the selected features increase the software defect prediction model accuracy.

(ii) Null hypothesis (Ho): the selected features do not increase the software defect prediction model accuracy.

The most promising methods are ML algorithms such as the *K*-Nearest Neighbor (KNN), Support Vector Machine (SVM), Naïve Bayes (NB) and Logistic Regression (LR) [18–21], Ensemble classifiers [22–25], and other different feature selection techniques [16, 18, 20, 21] are also used in research [14]. Machine learning techniques are the soul of data mining, which are used successfully for solving the complicated problems either in industry or research [16].

In healthcare softwares, many defects remain undetected during the software development process by developers. This is because of the misinterpretation of the requirements of healthcare software, unreasonable process of development or the insufficient experience of development and less effective models [12–16]. The presence of defects results in decreased quality which may result in failure of healthcare software projects [2–18]. Effective techniques for identifying potential defective components as early as possible can be used to optimize the testing effort. There is a need to address these issues by using state-of-the-art techniques for defect prediction to enhance the quality of healthcare systems.

The research objective of the research is to identify relevant features to predict the software defects of healthcare systems and the ML algorithms that improves the accuracy of SDP. The main contributions of the research are given as follows:

(1) The proposed technique provides a better accuracy which uses a novel optimized feature selection technique with machine learning classifiers for healthcare systems

(2) Early and accurate detection of defects helps in achieving high accuracy and performance of defect prediction model

The paper is organized as follows: Section 2 gives the background of SDP research and describes the state-of-the-art techniques of ML, Section 3 describes the proposed methodology, and the results are presented in Section 3.

2. Literature Review

In software systems, the defects are detected by applying the diverse algorithms especially ML algorithms. The researchers have used different feature selection algorithms merging different classifiers for increasing the performance of the software systems. The literature of defect prediction in health care domain is limited in terms of machine learning algorithms. The literature is divided on the basis of healthcare software and general software for defect prediction.

2.1. Defects Prediction Techniques for Healthcare Software. Different software development process models are used to predict the defects in healthcare softwares. The defect prediction technique such as model localization, static analysis, software metrics, and code review are used to identify the defects in healthcare applications [2, 3]. The modern analysis tools help to reduce the software maintenance cost by early detection of software defects in software development process, and the statistical analysis tool is used to analyze the software system without executing the software. [3]. The recall data of user interface errors in medical devices is analyzed. The two phases are carried out to recall the data. In the first phase, about 423 medical-related recalls are identified from user interface software errors. In this phase, the semiautomatic filtering process is applied to eliminate the recalls quickly that were not caused by software errors. In the second phase, the total number of 499 user interface software errors are identified and detail classification of the errors are established. The data is classified into 20 categories that is used by healthcare providers, device manufacturers, and regulatory authorities to raise the awareness of the impact of user interface software errors. The classification provides the evidence-based challenge to the stakeholder to increase the quality of the user interface software in different medical devices [7].

2.2. Machine Learning (ML) Techniques for Healthcare Software. Machine learning helps to diagnose the problem in different medical domains and analyze the most important clinical parameters for prediction [26–32]. The ML techniques are used for analysis of data such as data regularity detection which deals with data imperfection and continuous interpretation data that is used in intensive care unit. The ML algorithms help in the integration of the computer-based systems for healthcare software that increase the quality and efficiency of the healthcare systems [33, 34]. The ML techniques are used to explore the patterns from different medical data sources and help to predict the defective data appropriately [35]. Different ML techniques such as supervised learning, semisupervised learning, unsupervised learning, and reinforcement learning are reviewed to develop the efficient decision support system for healthcare software. The machine learning-based health protection system is capable to identify the data patterns efficiently [35]. The ensemble ML model is presented for predicting the time and behaviour of the oxide glasses for different medical applications. Data of 1300 records are used for an original glass dissolution experiment. The results demonstrate that the proposed model accurately predicts the chemical degradation behaviour of different glasses. It uses ML algorithms to handle and utilize the biomedical data from different perspectives [36]. The different Internet of Things (IoT) medical devices such as emergency medical equipment, medical drone, and ambulance face severe challenges like signal distortion and security issues [37, 38]. The paper represents the efficient lightweight encryption algorithm to design the secure image encryption technique for the healthcare industry. The proposed technique utilizes two different permutation techniques to secure the medical images. The proposed technique is analyzed, evaluated, and compared with the traditional encrypted ones on execution time. Multiple experiments are conducted that show that the proposed technique for image encryption provides better efficiency as compared to the traditional techniques. [38].

2.3. Machine Learning Techniques for Defect Prediction for General Software

2.3.1. Feature Selection (FS) Techniques. Different metaheuristic optimization algorithms are used as a search approach for feature selection techniques to predict the software defects. Researchers have used applied the base classifiers on different datasets and obtained the results presented in Table 2. In [17], the cluster hybrid feature selection technique is used for defect prediction. The proposed technique is applied on fifteen open source datasets, and the best average AUC value achieved by Pearson's correlation is 0.971, 0.809 achieved by MIC, 0.915 achieved by Spearman's correlation, and 0.915 achieved by Kendall's correlation. Reference [18] proposes the Multiobjective Feature Selection (MOFES) method. MOFES uses Pareto-based multiobjective optimizing algorithm. The AUC value is 1 and 107 second average computational cost. In [19], the improved versions of WOA by merging with a single-point cross-over technique are proposed that uses the five different FS techniques, i.e., random, tournament, Roulette wheel, stochastic universal sampling, and linear rank. The computational cost of the given model is high. In [20], a novel binary version of current HHO algorithm, i.e., EBHHO, for FS is proposed in [21]. The binary version of Queuing Search Algorithm (QSA) which is constructed on wrapper FS technique is proposed. The proposed model is applied on a 14 benchmark dataset, and the average AUC value based on F-rank test is 1.57; however, the prediction quality reduces when oversampling ratio is greater than 300 In [22], a relief F-based clustering and a cluster-based feature selection approach are proposed to identify and remove the redundant and irrelevant features. The proposed approach is applied on nine NASA SDP datasets and achieved the highest AUC value of RFC which is 0.767 for J48 classifier and 0.813 for Naive Bayes. In [23], a

Table 2: Literature review.

Reference	Techniques	Datasets	Results/findings	Limitation
Xu et al. [22]	RCF J48	9 NASA datasets	Highest AUC value of RFC is 0.767 for J48 classifier and 0.813 for Naïve Bayes	This approach is not focusing on the redundant features of high dimensional datasets
Hassouneh et al. [19]	BWOA KNN SVM, DT, and LDA	20 features of each datasets	Area under the Curve value based on *F*-ranked test is 2.88 of Linear Discriminant Analysis	Highest computational cost consumed by proposed model
Thaher et al. [21]	BQSA SMOTH	14 real world benchmark dataset	AUC value based on *F*-rank test is 1.57	Negative impact on prediction quality when oversampling ratio is greater than 300%
Iqbal and Aftab [23]	Artificial Neural Network, multi-layer perceptron	12 NASA datasets	Highest ROC value is 0.955, *F*-measure is 0.918, and MCC is 0.838	MLP-FS is not significantly improved the accuracy with sampling techniques
Ni et al. [18]	MOFES PMAs, KNN, LR, NB, and J48	10 datasets of PROMISE and 3 from RELINK	AUC value is 1 and 107 second average computational cost	Class imbalanced issue exists. Need to incorporate class imbalanced learning methods such as sampling technique into MOFES

framework is proposed for feature selection that uses Multilayer Filter Feature Selection approach and Feed Forward Artificial Neural Network (Multilayer Perception) for prediction of defective modules. The proposed framework is performed on 12 NASA datasets with oversampling technique; the ROC value is 0.955, *F*-Measure is 0.918, and MCC is 0.838 that are significantly improved, but with little improvement in accuracy. In [24], the hybrid technique is proposed that combines the feature selection ability of the Opt-aiNet algorithm with ML classifiers to detect the defects. The proposed technique is applied on 5 open source NASA datasets. The results indicate that DT provides the highest accuracy of 94.82and 0.90 AUC value for the JM1 dataset. Reference [25] proposes a new feature selection technique En-Binary Particle Swarm Optimization integrated with the ensemble classifiers for defect prediction that based on fitness function. The proposed approach is applied on SOFTLAB and MORPH datasets. The results reflect that the proposed approach achieves the best FM rank by comparing with other feature selection techniques. In [39], the Particle Swarm Optimization on object-oriented metrics for feature selection is proposed. Reference [40] proposed the improved binary dragonfly algorithm that is the extended version of dragonfly algorithm for feature selection.

2.3.2. Sampling Techniques. Adaptive Synthetic Sampling (ADASYN) technique is used [16, 20] to rebalance the dataset for increasing the quality of classifiers. Synthetic Minority Oversampling (SMOTH) is used [21, 25, 41] for balancing the dataset to increase the accuracy of the proposed model. The results find that the SMOTE method enhances the performance of defect prediction with highly imbalanced dataset. A study [23] uses the Random Oversampling technique on 12 NASA datasets that decrease the dataset imbalance ratio by copying the instances in minority class. This technique enhances the dataset volume due to application.

2.3.3. Ensemble Techniques. Machine learning algorithms show promising performance for solving the software defect prediction problem. Ensemble techniques are used for predicting software defects. In [41], adaptive boosting, random forest, bagging, and XGBoost are used as an ensemble. Bagging, boosting and stacking, and voting ensemble classifiers [42, 43] are used for predicting the software defects. [44] provides the empirical comparison of SDP models that are developed through different boosting-based ensemble techniques on three open source projects. Three ensemble techniques RUSBoost, SMOTEBoost, MSMOTEBoost are integrated with resampling methods that improve the performance of the model. Many base classifiers are used such as *K*-Nearest Neighbor (K-NN) [16, 18–20, 41, 42], decision tree (DT) [16, 19, 20, 40, 42], Linear Discriminant Analysis (LDA) [16–20], Logistic Regression (LR) [19], Naïve Bayes (NB) [18–41], Random Forest (RF) [41, 43, 45], and Support Vector Machine (SVM) [19, 39, 43, 45]. The researchers applied the base classifiers on different datasets and obtained the required results.

Literature is discussed related to defect prediction techniques used in different medical devices and ML techniques to predict the defects for healthcare softwares and general softwares. Keeping in view the critical nature of healthcare systems and the consequences a defect can have, software defect predication is a much needed endeavour. More research is required for sophisticated, timely, and accurate defect prediction.

3. Proposed Methodology

The proposed model predicts the defects on the basis of selected features. For validation of the proposed model, a controlled experiment on python language is performed.The results are evaluated and compared by using AUC and accuracy measure. The performance optimization key parameters

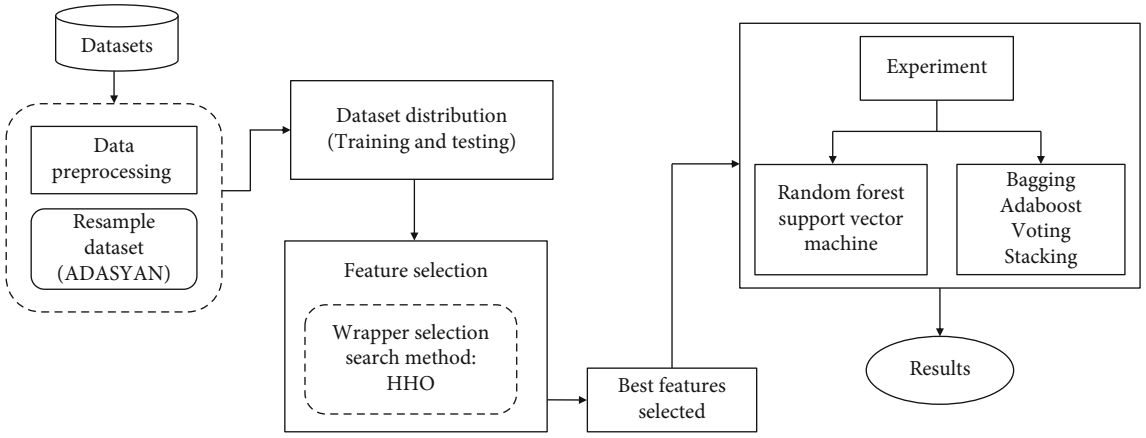

Figure 1: Proposed approach.

such as precision, recall, and F1-measure are also used to evaluate the model performance. The controlled experiment is a technical test that is mostly used by researchers for testing a unique variable. In this research, the independent variable (healthcare software defects) is used to test the effect on dependent variable (accuracy, AUC). The controlled variables are very important because variables change according to requirements that may impact on the behavior and relation among dependent and independent variables. In experiment, control variables are significant for testing the credibility of the results. The different optimization key metrics such as precision, recall, and F1-measure are used to check the performance of the model. The model consists of four different steps; the first step is to select the dataset. The second step is to perform the data preprocessing that consist of the main process to resample the imbalance data by using ADYSYN technique and then distributes the data into training and testing parts. The third step is to apply the wrapper feature selection technique HHO. The last step is to perform the experiment by using different ML algorithms RF, SVM, bagging, Adaboost, voting, and stacking. The proposed model is explained in Figure 1.

3.1. Experimental Setup. The experiment is held on a computer Intel(R) Core(TM) i7-2640M CPU 2.80 GHz with 4GB RAM and 64-bit Operating System. PYTHON is used as a tool for experiment.

(i) RQ1: what is the impact of multiobjective feature selection method on the software defect prediction model?

(ii) RQ2: does the bagging based ensemble classifier impact the accuracy of prediction of software defects?

3.2. Dataset Selection. The healthcare dataset is used for the experiment (https://www.kaggle.com/datasets/iqrayousaf/healthcare-dataset-for-defects-prediction). The dataset is created for healthcare systems by considering the defects in different medical applications [2] and articles related to software defects in healthcare software [3, 7]. The defective data in medical applications can cause death of patients. The most critical defects are predicted. The healthcare dataset has binary classes that is defective and nondefective. The dataset is available on Kaggle. The dataset is divided into two parts training and testing datasets, where 70% is used as training data to train the proposed methodology and 30% is considered testing data used to test the proposed model. The features in the healthcare dataset are given in Table 3.

3.3. Data Preprocessing. In this section, data preprocessing is performed to balance the highly imbalanced data and recognize and remove the irrelevant features. Data preprocessing step is the main part because it helps to achieve the more accurate features and increases the performance of the prediction model. It consists of two main steps: resampling dataset and feature selection.

3.4. Resample Dataset. The real life classification problem consists of large amount of data. Extracting the meaningful information becomes demanding in terms of space and computational time. Furthermore, irrelevant data may result in complicated and insufficient defect prediction models. Therefore, it is necessary to implement the preprocessing methods for boosting the classifiers performance [46]. The classifier performance is impacted by different factors such as the number of class type and number of samples. For data collection, the imbalance problem occurs when minority class is very compact as compared to majority class. The bright ML algorithms normally hurt when dataset is straight in direction of one class (minority or majority). Actually, most of collected data hurt by disproportion data that decreases the overall performance of the algorithms. There are two major techniques for handling the imbalanced data: the algorithm perspective and data perspective. The data perspective technique rebalances the distribution of class on the basis of resampling the space among data by using either undersampling or oversampling cases for majority classes or minority classes. The resampling technique tries to reduce the imbalance dataset issue either deterministically or randomly [16]. Different techniques are recommended to address the imbalanced data such as ADASYN [16–20], SMOTE [21, 39, 41], and Random Oversampling [23]. In

the experiment, the Adaptive Synthetic Sampling (ADASYN) is used for imbalanced dataset to enhance the performance of classifiers. This technique synthesizes the minority classes as per its distribution in training dataset, which pay attention to more data samples that are tough to learn and small data samples that are easy to learn. The ADASYN approach key is to find the distribution probability used as a criterion to synthesize the number of samples for all minority data samples and finally get the new dataset sample [47]. The algorithms are given in Algorithm 1.

3.5. Feature Selection (FS). Feature selection is the important preprocessing step for the classification tasks. The main objective of feature selection is to search the effective subset of feature which displays the raw data at the highest possible degree. The feature selection contributions are threefold: improved learning performance, decreased learning time, and simple model. The same does not happen with the feature subset; thus, the relationship between model performance and feature subset is nonlinear. To overcome these natural challenges, feature selection needs to optimize two different objectives as its focus is to decrease the total amount of features while enhancing the performance of the model [46]. The task of feature selection is defined as

$$\min o_1, \min o_2, \tag{1}$$

Subject to

$$\begin{aligned} o_1 &= |f| \\ o_2 &= \text{performance}(f), \end{aligned} \tag{2}$$

where $f \subseteq F$.

In the above equation, o_1 and o_2 show the first objective and second objective, respectively. Regarding these objectives, reduce the number of features and try to enhance the learning classifiers performance. Appropriately, when features subset (f) is selected from all features (F), o_1 equals to the total numbers of features in subset and o_2 equals to the classifiers accuracy obtained by the testing data after training the selected features only.

By considering these facts for feature selection problem, an ideal solution is to use a single feature that can separate the classes perfectly. Figure 2 [46] represents the sample solution for feature selection. Different sample solutions fs1, fs2, fs3, fs4, fs5, and fs6 are provided. The solutions fs1, fs2, and fs3 dominate the other solutions (fs4, fs5, and fs6) in both objectives; for example, fs1 feature subset solution selects less features and provides the best results as compared to fs4. Solution fs1 dominates solution fs4 and solution fs2, and fs3 also provides better results. Solutions fs1, fs2, and fs3 perform better in different objectives. Therefore, we have found the ideal solution (nondominated) that fits to the Pareto curve.

3.6. Proposed Multiobjective Harris Hawks Optimization Algorithm. The main aim of this research is to use the multiobjective optimization algorithm to search the most different Pareto optimal solution. The multiobjective optimization algorithm is used for optimization with two different objectives: one to reduce the total amount of selected features and the other to maximize the performance of the proposed model. The multiobjective feature selection technique Harris Hawk Optimization helps to find the optimal solution to achieve the desired objectives and increase the classification performance.

Table 3: Healthcare dataset feature detail.

Feature ID	Features
0	Duplication entities
1	Duplication features
2	Computational error
3	LineOfCode
4	LineofCodeandComments
5	Format mismatch
6	Misfielded value
7	Distortion
8	Branch count
9	Count statement
10	AvgCyclomatic
11	RatioCommenttoCode
12	Count semicolon
13	Lake of coupling in methods
14	Functional abstract measurement
15	CountLineCodeExecution
16	CountStatementExecution
17	Design complexity
18	Aggregation measurement
19	Time estimator

The HHO algorithm is recently presented that is motivated by chasing actions of the hawks. It proves to be an efficient metaheuristic technique for identifying the difficult optimization problems like feature selection [46]. The HHO algorithm is proposed by [48] that has a collaborative behavior and chasing style of Harris hawks and pounce on their prey.

Finding the prey, sudden pounce and different attacking plans perform the exploitative and explorative phases of the algorithm [49]. The flock of hawks pounce the prey collectively from different directions and approach the prey. The hunt completes by catching the target or try to go for other approaches. The Harris hawks contain a different set of patterns to follow the target. The strategy is changed when the head hawk bows at the target and the other hawks continue to pounce the target. These collective strategies fatigue the rabbit and enhance the target vulnerability. HHO is the population-based method. It is gradient free and can be implemented on any optimization issue. The HHO metaheuristic uses the exploitation and exploration phases that are motivated through the hawks' actions while searching the prey, with sudden attacks and diverse pounce plans. This activity of the Harris hawk is designed as the exploitation activity of the artificial hawk in algorithm. Figure 3

Input: Dataset S1, that contain n samples vi, wi, i = 1, 2, 3…n, vi is a n-dimensional space sample, label is wi 0, 1, wi = 0 shows minority class and wi = 1 shows majority class.
Output: Novel synthesized data samples.
1. Compute imbalance class
2. Compute S = (n1–n2) x ß that is total amount of data samples to be synthesized: where ß is a coefficient.
3. For all of minority data samples $v_{i,}$ note the points of K-nearest neighbor and compute the ratio $r_i = \Delta i/K$, i = 1, 2, 3 … n, Δi is the number of observations in K-nearest neighbor.
4. Regularization the r_i according to $r_i = r_i / \int_{i=1}^{no} ri$ therefore, r_i is the probability distribution $(\sum r_i = 1)$.
5. Compute $s_i = r_i$ x S which is the total number of synthetic samples that are required for all samples v_i of minority class.
6. The s_i samples are synthesized for all sample v_i of minority class.
7. For m= (1, 2, … , g_i)
8. Choose the sample v_m, from K nearest neighbor of v_i randomly.
9. Suppose represent the range of number [0, 1], for available $v_{im,}$ produce the synthetic sample according to $v_q = v_i + (v_i - v_{im})$.
10. Stop algorithm

Algorithm 1:Pseudocode of Adaptive Synthetic Algorithm.

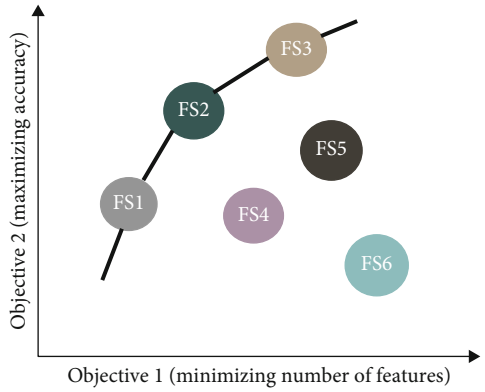

Figure 2: The Pareto optimal curve for multiobjective optimization algorithm.

represents how the phases of exploitation and exploration replace as per the prey energy level (E) and activities chance (q and r). The activity detail is given as follows [48]:

3.6.1. Exploration Phase. The Harris hawks have sharp eyes for tracking and detecting their target, but sometimes, it cannot easily find it. So the hawks detect the field to search the prey. Therefore, the hawks perch on a place and observe the prey with two different plans that are randomly used. The hawks perch as per to the location of other hawks and the rabbit when $q < 0.5$ or perch on the random tall tree $q \geq 0.5$. There are the equal chances for all strategies. The representation of solution (hawk) for select or not select the features is given in Figure 4 [48].

The HHO method can go from exploration phase to exploitation phase according to the prey escaping energy. With iterations, the rabbit energy (E) reduce according to the given formula:

$$E = 2Eo\left(1 - \frac{t}{T}\right), \tag{3}$$

where Eo is the initial energy of rabbit, t is the current iteration, and T is the total number of iterations. Eo is the random number initialized at each iteration with (-1, -1). The rabbit becomes strong when Eo value increases by 0 to 1.

The rabbit starts losing its energy when Eo value reduces by 0 to -1, as the number of iteration increases the escaping energy reduces. The HHO is at exploration step when $E \geq 1$ and at exploitation phase when $E < 1$. In short, the exploration and exploitation phase occur when $E \geq 1$ and $E < 1$, respectively [46–49].

3.6.2. Exploitation Phase. The prey normally can run away easily from risky conditions. So the hawks apply diverse chasing ways. In the exploitation phase, four different strategies are used according to the plan of hawks. Suppose r indicates the chances that the prey can escape $(r < 0.5)$ or chances of prey that it cannot escape $(r \geq 0.5)$. The soft and hard besiege performs to encircle the rabbit. The hawks encircle the rabbit by different locations according to the energy (E) of rabbit. The hawks together pounce on the prey for increasing the chance to grab the rabbit. The hawks increase the process of besiege to grab the rabbit when the prey starts losing energy. The soft besiege is used when $|E| \geq 0.5$, and the hard besiege is applied when $|E| < 0.5$ [46–49].

3.6.3. Soft Besiege. The rabbit has a good energy level when $|E| \geq 0.5$ and $r \geq 0.5$, and through some random bounces, the rabbit can escape. At the same time, the Harris hawks quietly encircle the rabbit to make it extremely tired and then execute the sudden pounce actions. The small random number of features from the original dataset, J, is created (that represent the movement of the rabbit in nature), and many of the features by the rabbit are copied to the selected hawks [46].

3.6.4. Hard Besiege. The prey has the low level of energy when $|E| < 0.5$ and $r \geq 0.5$ and cannot easily run away. The current position of Harris hawk is updated by the given equation:

$$X(i, d) = Xrb(0, d) - E * abs(\Delta X), \tag{4}$$

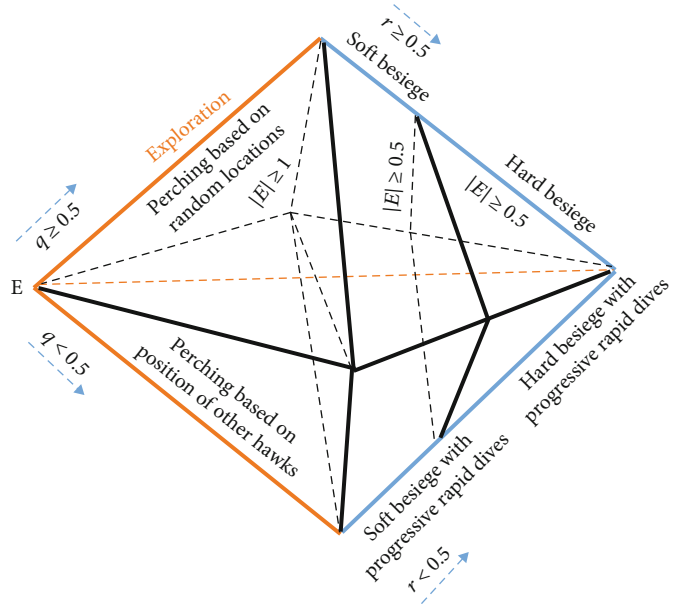

Figure 3: Process of exploration and exploitation phases.

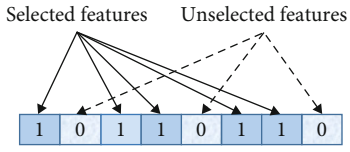

Figure 4: Representation of solution (in feature selection).

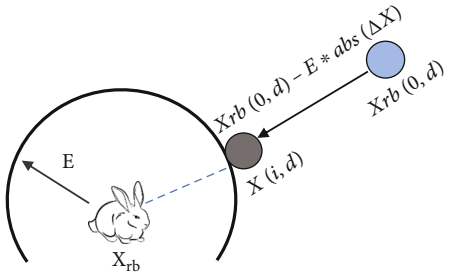

Figure 5: Process of hard besiege.

where ΔX indicates the difference among the current location of rabbit and iterations of hawk. For hard besiege in the proposed model, for the current hawk, the single feature of rabbit is copied. Figure 5 represents this step.

3.6.5. Soft Besiege with Progressive Rapid Dives. The prey can run away when $r < 0.5$ and $|E| \geq 0.5$, and before sudden pounce, the soft besiege can be applied. To design this sample of the prey in HHO algorithm, use the levy distribution for high-level perturbation. According to the level of energy of rabbit, the features are selected by the given solution that is not similar as the current hawk [46]. The features are selected through greedy selection method that selects the best features at that moment and solve the problems that arise later. By greedy feature selection, the classifiers become more efficient, and the defect prediction model provides more precise results.

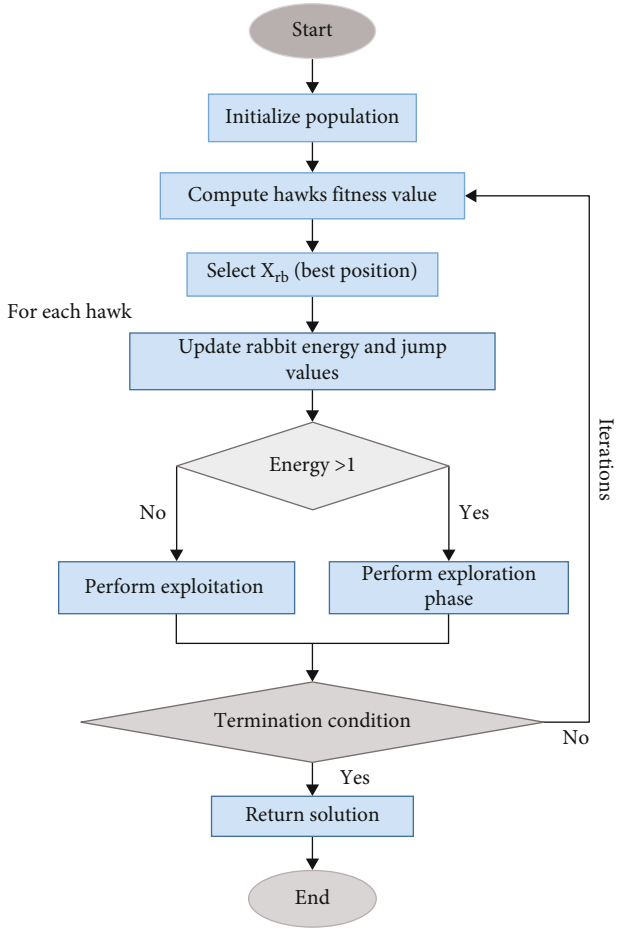

Figure 6: Flow chart of the proposed model for binary classification.

Input: Population size is M, the maximum number of iterations T and the number of the iteration t.
Output: Best solution
1. The population initialization of Hawks Xi (i = 1, 2, 3, ⋯, M)
2. While (t ≤ T)
3. Calculate the fitness value of hawks
4. Search the best location and set this location as the location of rabbit Xrb
5. for(each hawk (Xi)) do
6. Update the initial energy Eo
7. Update the level of energy of rabbit (Equation. (3))
8. if (|E| ≥ 1) then
9. Perform Exploration
10. else if ((|E| < 1)
11. Perform the Exploitation
12. If (|E| ≥ 0.5 and r ≥ 0.5) then
13. Perform soft besiege
14. else if (|E| < 0.5 and r ≥ 0.5) then
15. Perform Hard besiege with update the hawk's position (Equation. (4))
16. else if (|E| ≥ 0.5 and r < 0.5) then
17. Perform Soft besiege with progressive rapid dives with Greedy feature selection
18. else if (|E| < 0.5 and r < 0.5) then
19. Perform Hard besiege with progressive rapid dives with Greedy feature selection
20. Calculate fitness value of the updated Hawk
21. Result ⟵ Best feature subset
22. Return Result

Algorithm 2: Pseudo code of multi-objective Harris Hawk Optimization

3.6.6. Hard Besiege with Progressive Rapid Dives. The hard besiege is used when $r < 0.5$ and $|E| < 0.5$, and the prey cannot run away. This case is the same as the soft besiege. The hawks go to reduce the gap with the prey. According to the energy level of the rabbit, from rabbit, different features are selected and they are copied to a hawk randomly chosen from the population. The less number of features are selected to stop the high level of disturbance and make the model stable [46]. The representation of proposed model is given in Figure 6 [46].

The multiobjective Harris Hawks Optimization algorithm is provided in provided in Algorithm 2. The steps performed in algorithms are given in detail. The population size$M$, the maximum number of iteration$T$, and the number of iteration t are taken as inputs. The Hawk Xi population is initialized by using Feature Selection Function (Jfs), and calculate the fitness value of hawks by using fitness function. After this search the best location of the target and set this location of the rabbit location Xrb with update the initial energy of rabbit. Update the level of the energy of the rabbit, and then perform the exploration and exploitation phases. Soft besiege and hard besiege are performed with updating the position of hawks. Progressive rapid dives with soft and hard besiege are performed with greedy feature selection that selects the relevant features which increase the performance of the model.

3.7. Classification Algorithms. To differentiate software defect modules with tender ones, two main classifiers are used which are Support Vector Machine (SVM) and Random Forest (RF). These classification methods are important and mainly used in machine learning (ML) and also manifest the important performance of classification. The main aim of the classifiers is to take out pattern that discloses particular class; all data instance is associated in an available dataset [22].

3.7.1. Random Forest. Random Forest is a supervised classification method that has a collection of trees to create the forest [50]. Random Forest chooses the features randomly for creating the model by using decision tree. For this, it builds different decision trees (random forest) by choosing random variable and data. From the chosen attributes, randomly select the number of instances and allocate to the classification learning algorithm [41]. The random forest algorithm is split into two methods [50].

(1) Random Forest Creation. By using the random sample, all tree is trained and replaced by a training set. The pseudocode of creation of random forest is given below.

Select random "m" features from the total number of "n" features. Between "m" features, by using the finest split point, compute the node "t."

By using the best split, divide the node into subnodes.

Repeat the above steps until the number of nodes reached at "1."

By repeating all the above steps, create the forest for the "k" number of times for creating the "k" number of trees.

(2) Created Tree Prediction. For training separate trees, the randomly selected features are used to search for dividing. The random allocation decreases the relation between trees that enhance the performance of prediction. The following procedure is used for prediction performance:

Follow the rules of all randomly built decision trees with grasp of the test features.

For all predicted targets, compute the votes.

Consider the final prediction to the predicted target that obtain high votes.

3.7.2. Support Vector Machine. Support Vector Machine (SVM) uses the kernel trick technique for solving the nonlinear separable issue by plotting the points into a high-dimensional area. It solves the overfitting problem innate in learning algorithms. The important aim of SVM is to search the optimal hyperplane among the dataset classes by increasing the gap between the closest points of the classes. The maximum hyperplane gap provides the maximum distance among both classes [46].

3.7.3. Bagging. Bagging is an ensemble approach that enhances the accuracy of ML techniques by integrating prediction of different weak classifiers. It gives the better results for unstable classifiers with small changes in a training set and results in high prediction performance. Bagging predicts the results many times by several training sets which are integrated by voting. For explaining the bagging algorithm, suppose the dataset with M instances and the binary label of class. The procedure of bagging algorithm is given [44].

Create the M size training set randomly, and replace it with data.

By applying any classification algorithm to train the random training set, allocate the class to all node. Repeat the above steps several times. Use the voting for the prediction of label of class.

3.7.4. Adaboost. Adaboost is the mostly used boosting algorithm that slowly increases the weights of classifier of classification error. Create the new classifier in all iterations to overcome the failure of old classifier, and then associate the created classifier with the voting process together. So the Adaboost essence promotes the weak classifier to strong classifier that is an adaptive lifting technique.

Therefore, the classification error rate reduces as the number of training data increases. The following steps are used in the Adaboost algorithm [51].

Takes the training dataset and all training samples learn through this got the first weak leaning classifier, also provides the maximum number of iterations (M).

The incorrect classification of sample and other data is integrated to represent the new training dataset, and at the same time, adjust the sample weight.

Repeat it to M number of times. The new training data samples are created for the next iteration learning classifier that is based on new weight, and finally, the strong classifier is created with better classification effect.

3.7.5. Voting. The voting algorithm is mostly used with learning classifiers combination. The basic idea behind the voting is for classification that uses the diverse probability estimation combinations. In voting, the integrated classifiers vote for the label of class [52].

3.7.6. Stacking. Stacking is the beneficial ensemble machine learning technique. The basic idea behind the stacking ensemble algorithm is as features utilize the confidence score in integrating different models and train the metaclassifier for helping to integrate the prediction of different learning classifiers [53].

3.8. Evaluation Metrics. The prediction and classification problems have different evaluation measurements such as accuracy, specificity, sensitivity, precision, and receiver operating characteristics-Area under the Curve. In this experiment, we evaluate the proposed model based on accuracy and AUC value. The ROC-AUC evaluation method is mostly used in software defect prediction. The AUC value calculation relies on the ratio among false positive verses true positive rates. The AUC value relies on two methods: specificity and sensitivity. The accuracy is the evaluation metrics that distinguish the defective and nondefective part of the software correctly. To estimate the accuracy, calculate the proportion of true positive and true negative in all evaluated cases. The optimization key parameters such as precision, recall, and F1-measure are used to evaluate the performance of the model.

$$\begin{aligned}
\text{Accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}},\\
\text{Specificity} &= \frac{\text{TN}}{\text{N}},\\
\text{Sensitivity} &= \frac{\text{TP}}{\text{P}},\\
\text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}},\\
\text{Recall} &= \frac{\text{TP}}{\text{TN} + \text{FN}},\\
\text{F1} - \text{measure} &= \frac{2 \times \text{Precision} + \text{Recall}}{\text{Precision} + \text{Recall}}.
\end{aligned} \tag{5}$$

4. Experimental Results

In this section, we evaluated the performance of the proposed model. The proposed model is measured on different parameters. These measurements are executed to check whether the proposed model is better than the existing techniques and to check whether the proposed model is suitable for healthcare application defect prediction or not. The proposed model is applied on healthcare dataset, and six different search techniques are implemented on this dataset. The proposed model reveals the best results as compared to other state-of-the-art techniques. The selected features of the healthcare dataset are given in Table 4.

The Area under the Curve and accuracy are found based on the selected features using RF, SVM as a base classifiers and ensemble classifiers bagging, Adaboost, Voting and Stacking given in Table 5. The obtained results indicate that the best AUC value is 0.992% and 0.957% of RF and Adaboost as a base and ensemble classifiers, respectively. The RF as a base provides the best accuracy of 0.990%, and

Table 4: Selected features.

Algorithms	Selected features
RF	LineofCodeandComments, format mismatch, misfielded value, AvgCyclomatic, FunctionalAstrctMeasurement, design complexity
SVM	Format mismatch, AvgCyclomatic, time estimator
Bagging	Duplication entities, LineOfCode, FunctionalAstrctMeasurement, CountStatementExe
Adaboost	Duplication features, LineofCodeandComments, misfielded value, AvgCyclomatic, Count Semicolon
Voting	Computational error, format mismatch, distortion, count statement, CountLineCodeExe
Stacking	Duplication entities, duplication features, computational error, LineOfCode, LineofCodeandComments, AvgCyclomatic, RatioCommenttoCode, lcm, FunctionalAstrctMeasurement, CountStatementExe, time estimator

stacking as an ensemble classifier gives the best accuracy of 0.976%.

In Table 6, the proposed model results are provided without feature selection. It can clearly see that the proposed model does not perform better without feature selection for defect prediction.

In Table 7, the performance of the optimization parameters precision, recall, and F1-measure is provided to evaluate the performance of the proposed optimization feature selection algorithm. Accuracy and AUC parameters are used for comparison.

The comparison of AUC value of different classification algorithms with and without feature selection is provided in Figure 7. It can be seen that RF as a base classifier and Adaboost as an ensemble classifier with feature selection perform better for healthcare software defect prediction. The overall performance of the proposed model is better with feature selection for defect prediction of healthcare software.

In Figure 8, the accuracy comparison of all classifiers is provided and it represents that the proposed approach performs better with feature selection. RF as a base classifier and stacking as an ensemble classifiers with feature selection perform better for healthcare software defect prediction.

4.1. Findings and Discussions. The experiment demonstrates that the proposed model performs best on the healthcare dataset. For checking the performance of the proposed model, two metrics accuracy and Area under the Curve (AUC) are used. The optimization key parameters such as precision, recall, and F1-measure are also used to evaluate the performance of the model. As base classifiers, the RF gives the best AUC result and stacking as a base classifier provides the best accuracy. The SVM and RF give the best precision, recall, and F1-measure results, respectively. The overall performance of the proposed model is better for the healthcare application defect prediction.

Table 5: Proposed approach results (with feature selection) of different classification algorithms.

Algorithms	Area under the Curve	Accuracy
RF	0.992	0.990
SVM	0.987	0.989
Bagging	0.954	0.963
Adaboost	0.957	0.968
Voting	0.956	0.966
Stacking	0.954	0.976

Table 6: Proposed approach results (without feature selection) of different classification algorithms.

Algorithms	Area under the Curve	Accuracy
RF	0.519	0.987
SVM	0.426	0.977
Bagging	0.505	0.861
Adaboost	0.623	0.880
Voting	0.532	0.876
Stacking	0.511	0.852

Table 7: Performance of optimization parameters.

Algorithms	Precision	Recall	F1-measure
RF	0.903	0.875	0.888
SVM	0.941	0.615	0.744
Bagging	0.917	0.804	0.857
Adaboost	0.861	0.684	0.763
Voting	0.901	0.720	0.801
Stacking	0.841	0.709	0.769

4.1.1. RQ 1: What Is the Impact of Multiobjective Feature Selection Method on the Software Defect Prediction Model? To evaluate the effectiveness of the multiobjective HHO, we must examine the effects of the multiobjective feature selection technique. Using the HHO algorithm, relevant features are selected by using various population sizes (i.e., 5, 10, 15, 20, and 30) and iterations (i.e., 10, 20, 30, 40, and 50). The number of population size and iterations play an important role in performance of prediction model. The proposed model gives the best results when the population size is 10 with 30 iterations, and the worst performance of the model is when the population size is 20 and 30 with 40 and 50 iterations, respectively. To obtain the best results, it is important to tune the parameters of feature selection technique carefully. The dataset variations also impact the feature selection method. Before the experiment, the datasets were highly imbalanced, and after application of HHO on imbalance datasets, it does not provide the relevant features. In this experiment, to get the relevant features and best results, balance the dataset by using adaptive synthetic sampling technique before feature selection. The multiobjective feature selection method surely impacts on software defect prediction model. Without feature selection, the model does not provide the best performance. The results of the

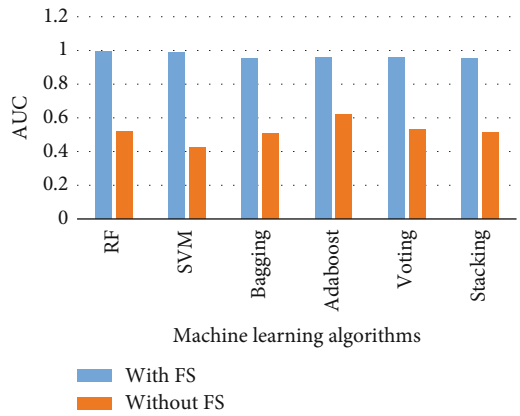

Figure 7: Comparison of AUC (with and without FS) of the proposed approach.

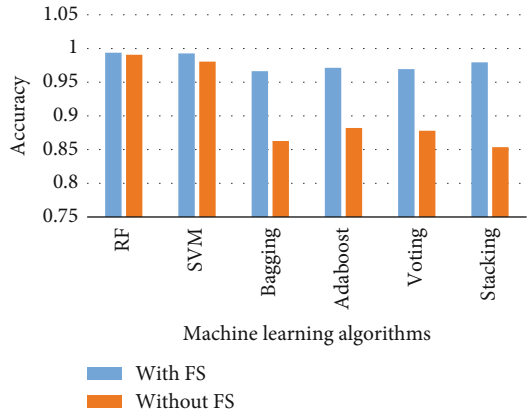

Figure 8: Comparison of accuracy (with and without FS) of proposed approach.

proposed model without features selection are provided in Table 6. The results clear that the healthcare defect prediction model provides the worst performance without feature selection. Therefore, in this experiment, the multiobjective feature selection method is used to focus on the two objectives; one is to select the relevant features provided in Table 4, and the other objective is to achieve the best results given in Table 5.

4.1.2. RQ 2: Does the Bagging-Based Ensemble Classifier Impact the Accuracy of Prediction of Software Defects? The ensemble classifiers Bagging, Adaboost, voting, and stacking are implemented on healthcare dataset to check the accuracy impact on defect prediction on healthcare applications. In Table 5, the accuracy of ensemble classifiers is provided. The best accuracy provides the stacking for healthcare application dataset, and the worst result bagging is provided.

4.2. Treats to Validity. In this experiment, the different threats to validity are explained.

4.2.1. Internal Validity. HHO is used for feature selection, but for the software defect prediction, other feature selection techniques can be used. The performance of the proposed approach can vary if there are different feature selection techniques such as GWO, DF, and GA. In addition to defect prediction, two classifiers (RF and SVM) and four ensemble models (bagging, Adaboost, voting, and stacking) are used in the experiment to achieve the best results. In spite of that, more models such as deep learning techniques can be used.

4.2.2. External Validity. The external threats to validity are minimized by creating a dataset from the data of different healthcare applications. The experiment is executed on health care application dataset, and the proposed approach performs best on the dataset. However if different open source projects and closed source projects are used, then the results can vary.

4.2.3. Construct Validity. The construct threats to validity are the selection technique for HHO to select the subset of features. In the experiment, a greedy selection technique is used to select the best subset of features that provides the best experimental results on the base of selected features. The usage of other selection techniques such as the best first search, random-based tournament, and roulette wheel can impact on the results of proposed approach.

5. Conclusion

Software engineering is used by many software consultants and vendors to develop high-quality healthcare systems such as middle-ware medical devices, patient record management systems, and electronic systems of medical devices. To design the healthcare applications, the software development process models are required to detect the defects in a timely manner. In this paper, an optimized feature selection multiobjective HHO is proposed with two objectives, i.e., minimize the total amount of selected features and increase the performance of the classifiers with the Adaptive Synthetic Sampling method to predict the software defects. The multiobjective HHO works as the wrapper-based feature selection method, and the ADASYN technique is used to increase the quality of datasets. Different machine learning techniques such as RF and SVM and ensemble classifiers such as bagging, Adaboost, voting, and stacking are used for defect prediction. The optimization key parameters such as precision, recall, and f1-measure are used to evaluate the performance of optimization model. After solving the imbalanced issue of dataset, the proposed model enhances the performance of all the algorithms. The RF as a base classifier and Adaboost as an ensemble classifier perform better for healthcare software defect prediction based on AUC while RF and stacking perform better than other classifiers based on accuracy. The proposed model helps to predict the software defects with a significant classification accuracy of 0.990 and an AUC score of 0.992. The obtained results clearly indicate that the proposed model is best for healthcare software defect prediction. In the future, work can be carried with deep learning algorithms CNN and ANN, for checking the model performance with more datasets form open source and closed source projects.

Data Availability

The (Healthcare Dataset for Defect prediction) data used to support the findings of this study have been deposited in the Kaggle repository (https://www.kaggle.com/datasets/iqrayousaf/healthcare-dataset-for-defects-prediction).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] N. Lalband and D. Kavitha, "Software engineering for smart healthcare applications," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 6S4, pp. 325–331, 2019.

[2] Y. Zhang and G. Koru, "Understanding and detecting defects in healthcare administration data: toward higher data quality to better support healthcare operations and decisions," *Journal of the American Medical Informatics Association*, vol. 27, no. 3, pp. 386–395, 2020.

[3] R. Jetley and B. Chelf, "Diagnosing medical device software defects using static analysis," in *Coverity*, Published in MD&DI, 2009.

[4] M. Rizwan, A. Shabbir, A. R. Javed et al., "Risk monitoring strategy for confidentiality of healthcare information," *Computers and Electrical Engineering*, vol. 100, article 107833, 2022.

[5] S. M. Akhtar, M. Nazir, K. Saleem et al., "A multi-agent formalism based on contextual defeasible logic for healthcare systems," *Frontiers in Public Health*, vol. 10, 2022.

[6] A. Mubashar, K. Asghar, A. R. Javed et al., "Storage and proximity management for centralized personal health records using an IPFS-based optimization algorithm," *Journal of Circuits, Systems and Computers*, vol. 31, no. 1, p. 2250010, 2022.

[7] Y. Zhang, P. Masci, P. Jones, and H. Thimbleby, "Research: user interface software errors in medical devices: study of U.S. recall data," *Biomedical Instrumentation & Technology*, vol. 53, no. 3, pp. 182–194, 2019.

[8] M. Kondo, C.-P. Bezemer, Y. Kamei, A. E. Hassan, and O. Mizuno, "The impact of feature reduction techniques on defect prediction models," *Empirical Software Engineering*, vol. 24, no. 4, pp. 1925–1963, 2019.

[9] L. H. Son, N. Pritam, M. Khari, R. Kumar, P. Phuong, and P. Thong, "Empirical study of software defect prediction: a systematic mapping," *Symmetry*, vol. 11, no. 2, p. 212, 2019.

[10] M. B. R. Pandit and N. Varma, "A deep introduction to ai based software defect prediction (SDP) and its current challenges," in *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, pp. 284–290, Kochi, India, 2019.

[11] C. L. Prabha and N. Shivakumar, "Software defect prediction using machine learning techniques," in *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184)*, pp. 728–733, Tirunelveli, India, 2020.

[12] R. Malhotra and K. Khan, "A study on software defect prediction using feature extraction techniques," in *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pp. 1139–1144, Noida, India, 2020.

[13] Y. Qiao, J. Qian, S. Jiang, W. Zhenhua, and G. Zhang, "An empirical study on the effectiveness of feature selection for crossproject defect prediction," *IEEE Access*, vol. 7, pp. 35710–35718, 2019.

[14] H. Turabieh, M. Mafarja, and X. Li, "Iterated feature selection algorithms with layered recurrent neural network for software fault prediction," *Expert Systems with Applications*, vol. 122, pp. 27–42, 2019.

[15] P. Kumar, G. P. Gupta, and R. Tripathi, "Toward design of an intelligent cyber attack detection system using hybrid feature reduced approach for IoT networks," *Arabian Journal for Science and Engineering*, vol. 46, no. 4, pp. 3749–3778, 2021.

[16] I. Tumar, Y. Hassouneh, H. Turabieh, and T. Thaher, "Enhanced binary moth flame optimization as a feature selection algorithm to predict software fault prediction," *IEEE Access*, vol. 8, pp. 8041–8055, 2020.

[17] J. A. Wang and Z. Zou, "A clusterbased hybrid feature selection method for defect prediction," in *2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)*, pp. 1–9, Sofia, Bulgaria, 2019.

[18] C. Ni, X. Chen, W. Fangfang, Y. Shen, and G. Qing, "An empirical study on Pareto based multi-objective feature selection for software defect prediction," *Journal of Systems and Software*, vol. 152, pp. 215–238, 2019.

[19] Y. Hassouneh, H. Turabieh, T. Thaher, I. Tumar, H. Chantar, and J. Too, "Boosted whale optimization algorithm with natural selection operators for software fault prediction," *IEEE Access*, vol. 9, pp. 14239–14258, 2021.

[20] T. Thaher and N. Arman, "Efficient multi-swarm binary Harris Hawks optimization as a feature selection approach for software fault prediction," in *2020 11th International Conference on Information and Communication Systems (ICICS)*, pp. 249–254, Irbid, Jordan, 2020.

[21] T. Thaher, M. Mafarja, B. Abdalhaq, and H. Chantar, "Wrapper-based feature selection for imbalanced data using binary queuing search algorithm," in *2019 2nd international conference on new trends in computing sciences (ICTCS)*, pp. 1–6, Amman, Jordan, 2019.

[22] X. U. Xiaolong, C. H. E. N. Wen, and W. A. N. G. Xinheng, "RFC: a feature selection algorithm for software defect prediction," *Journal of Systems Engineering and Electronics*, vol. 32, no. 2, pp. 389–398, 2021.

[23] A. Iqbal and S. Aftab, "A classification framework for software defect prediction using multi-filter feature selection technique and MLP," *International Journal of Modern Education & Computer Science*, vol. 12, no. 1, pp. 18–25, 2020.

[24] B. Mumtaz, S. Kanwal, S. Alamri, and F. Khan, "Feature selection using artificial immune network: an approach for software defect prediction," *Intelligent Automation and Soft Computing*, vol. 29, no. 3, pp. 669–684, 2021.

[25] R. Malhotra, A. Budhiraja, A. K. Singh, and I. Ghoshal, "A novel feature selection approach based on binary particle swarm optimization and ensemble learning for heterogeneous defect prediction," in *2021 3rd Asia Pacific Information Technology Conference*, pp. 115–121, Bangkok Thailand, 2021.

[26] A. R. Javed, M. U. Sarwar, M. O. Beg, M. Asim, T. Baker, and H. Tawfik, "A collaborative healthcare framework for shared healthcare plan with ambient intelligence," *Human-centric Computing and Information Sciences*, vol. 10, no. 1, pp. 1–21, 2020.

[27] M. Fayyaz, A. A. Farhan, and A. R. Javed, "Thermal comfort model for HVAC buildings using machine learning," *Arabian Journal for Science and Engineering*, vol. 47, no. 2, pp. 2045–2060, 2022.

[28] K. Saleem, M. Saleem, R. Zeeshan et al., "Situation-aware BDI reasoning to detect early symptoms of covid 19 using smartwatch," *IEEE Sensors Journal*, p. 1, 2022.

[29] A. Amanat, M. Rizwan, A. R. Javed et al., "Deep learning for depression detection from textual data," *Electronics*, vol. 11, no. 5, p. 676, 2022.

[30] M. Rizwan, A. Shabbir, A. R. Javed, M. Shabbir, T. Baker, and D. al-Jumeily Obe, "Brain tumor and glioma grade classification using Gaussian convolutional neural network," *Access*, vol. 10, pp. 29731–29740, 2022.

[31] A. R. Javed, "Automated cognitive health assessment in smart homes using machine learning," *Sustainable Cities and Society*, vol. 65, article 102572, 2021.

[32] T. M. Ghazal, M. Anam, M. K. Hasan et al., "Hep-pred: hepatitis c staging prediction using fine Gaussian SVM," *Computers, Materials & Continua*, vol. 69, no. 1, pp. 191–203, 2021.

[33] K. Shailaja, B. Seetharamulu, and M. A. Jabbar, "Machine learning in healthcare: a review," in *2018 Second international conference on electronics, communication and aerospace technology (ICECA)*, pp. 910–914, Coimbatore, India, 2018.

[34] T. M. Ghazal, S. Abbas, S. Munir et al., "Alzheimer disease detection empowered with transfer learning," *Computers, Materials & Continua*, vol. 70, no. 3, pp. 5005–5019, 2022.

[35] G. D. Magoulas and A. Prentza, "Machine learning in medical applications," in *Advanced Course on Artificial Intelligence*, pp. 300–307, Springer, 2001.

[36] T. Han, N. Stone-Weiss, J. Huang, A. Goel, and A. Kumar, "Machine learning as a tool to design glasses with controlled dissolution for healthcare applications," *Acta Biomaterialia*, vol. 107, pp. 286–298, 2020.

[37] M. K. Hasan, S. Islam, I. Memon et al., "A novel resource oriented DMA framework for internet of medical things devices in 5g network," *IEEE Transactions on Industrial Informatics*, p. 1, 2022.

[38] M. K. Hasan, S. Islam, R. Sulaiman et al., "Lightweight encryption technique to enhance medical image security on internet of medical things applications," *Access*, vol. 9, pp. 47731–47742, 2021.

[39] R. Malhotra, N. Nishant, S. Gurha, and V. Rathi, "Application of particle swarm optimization for software defect prediction using object oriented metrics," in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 88–93, Noida, India, 2021.

[40] J. Li, H. Kang, G. Sun et al., "IBDA: improved binary dragonfly algorithm with evolutionary population dynamics and adaptive crossover for feature selection," *IEEE Access*, vol. 8, pp. 108032–108051, 2020.

[41] H. Alsawalqah, N. Hijazi, M. Eshtay et al., "Software defect prediction using heterogeneous ensemble classification based on segmented patterns," *Applied Sciences*, vol. 10, no. 5, p. 1745, 2020.

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

[43] F. Matloob, S. Aftab, and A. Iqbal, "A framework for software defect prediction using feature selection and ensemble learning techniques," *International Journal of Modern Education & Computer Science*, vol. 11, no. 12, pp. 14–20, 2019.

[44] R. Malhotra and J. Jain, "Handling imbalanced data using ensemble learning in software defect prediction," in *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 300–304, Noida, India, 2020.

[45] 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.

[46] T. Dokeroglu, A. Deniz, and H. E. Kiziloz, "A robust multiobjective Harris' Hawks Optimization algorithm for the binary classification problem," *Knowledge-Based Systems*, vol. 227, article ???, 2021.

[47] H. He, X. Zhang, Q. Wang et al., "Ensemble multiboost based on ripper classifier for prediction of imbalanced software defect data," *IEEE Access*, vol. 7, pp. 110333–110343, 2019.

[48] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris Hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[49] M. Abdel-Basset, W. Ding, and D. El-Shahat, "A hybrid Harris Hawks optimization algorithm with simulated annealing for feature selection," *Artificial Intelligence Review*, vol. 54, no. 1, pp. 593–637, 2021.

[50] Y. N. Soe, P. I. Santosa, and R. Hartanto, "Software defect prediction using random forest algorithm," in *2018 12th South East Asian Technical University Consortium (SEATUC)*, pp. 1–5, Yogyakarta, Indonesia, 2018.

[51] R. Li, L. Zhou, S. Zhang, H. Liu, X. Huang, and Z. Sun, "Software defect prediction based on ensemble learning," in *Proceedings of the 2019 2nd International conference on data science and information technology*, pp. 1–6, Seoul Republic of Korea, 2019.

[52] M. A. Mabayoje, A. O. Balogun, A. O. Bajeh, and B. A. Musa, "Software Defect Prediction: Effect of Feature Selection and Ensemble Methods," *FUW Trends in Science & Technology Journal*, vol. 3, no. 2, pp. 518–522, 2018.

[53] M. A. I. Aquil and W. H. W. Ishak, "Predicting software defects using machine learning techniques," *International Journal*, vol. 9, no. 4, pp. 6609–6616, 2020.