MULTI-OBJECTIVE HYPERPARAMETER TUNING OF CLASSIFIERS FOR DISEASE DIAGNOSIS

Sunil Kumar

Assistant Professor, Department of Computer Science and Engineering,
Guru Jambheshwar University of Science and Technology, Haryana, India
skvermacse@gmail.com

Saroj Ratnoo

Professor, Department of Computer Science and Engineering,
Guru Jambheshwar University of Science and Technology, Haryana, India
ratnoo.saroj@gmail.com
http://www.gjust.ac.in

Abstract

Disease diagnosis is an arduous task due to the high cost associated with the misclassifications. The classifiers for disease diagnosis must be accurate, offer a tradeoff among sensitivity and specificity, and address the problem of class imbalance. Further, the ad-hoc hyperparameter settings deteriorate classifiers performance, specifically for disease diagnosis. For disease diagnosis systems, it is essential to address the issues of the setting of hyperparameters and class imbalance simultaneously. The objective of this paper is to propose a Multi-objective Hyperparameter Tuning approach, called MOHPT, in combination with resampling techniques. So far, the hyperparameters of classifiers for disease diagnosis have been often optimized based on a single objective criterion, i.e., accuracy. Multi-objective hyperparameter tuning in combination with resampling techniques is expected to enhance the efficacy of disease diagnosis.

We have used decision trees (CART) and random forest as the classifiers. The proposed system MOHPT obtains non-dominated optimal hyperparameter configurations of these classifiers using crowding based sorting technique from multi-objective NSGA-II. The MOHPT uses sensitivity (true positive rate) and specificity (true negative rate) as the two objective functions for optimization, and each configuration in the non-dominated front of hyperparameter configurations belongs to a different tradeoff between sensitivity and specificity. The suggested approach offers a choice to medical practitioners to prefer any of the hyperparameter configurations. However, we have selected the hyperparameter configuration based on the optimum g-mean, a product of sensitivity and specificity. The resampling techniques such as undersampling, oversampling, SMOTE, RWO, MWSMOTE, and ROSE have been implemented to tackle the class imbalance. The MOHPT is tested on 17 medical datasets. Several combinations of decision tree and random forest classifiers have been implemented in combination with hyperparameter tuning and resampling techniques. The results have been validated using appropriate statistical tests. The result shows that the application of hyperparameter tuning in combination with sampling techniques significantly enhances the performance of disease diagnosis. Overall, the performance of the random forest is superior to the other classifiers. The MOHPT comparison with the related works further proves its efficacy. The suggested technique has been successful in finding the set of non-dominated set of solutions for hyperparameter configurations along with addressing the class imbalance problem. The results show a performance improvement on several performance evaluation metrics such as sensitivity, specificity, AUC etc.

Keywords: **Hyperparameter Tuning, Multi-Objective Optimization, Disease Diagnosis, Decision Tree, and Random Forest Classifiers.**

1. Introduction

Medical data mining is gaining impetus because of the accessibility of a large amount of medical data. Early disease diagnosis can reduce the cost of treatment and risks of fatal consequences, including loss of lives. The healthcare systems all over the world are overloaded, and there is a scarcity of doctors. The recent spread of Covid-

19 has proved the limitations of the health systems all the more. In such circumstances, machine learning tools can boost the efficacy and efficiency of medical experts for disease diagnosis [1], [2]. However, automated medical diagnosis is high-risk area due to its direct influence on human lives. There are specific challenges to be addressed while applying machine learning algorithms for disease diagnosis [2]. For disease diagnosis, it is not enough to have produced a classifier model with high accuracy due to frequently occurring class imbalance in the medical data. The datasets with class imbalance often consist of many examples of the negative class, which signifies the absence of the disease and only a very few cases belonging to the positive class, which indicate the presence of the disease. For instance, it may be the case that 90 percent of the examples of a dataset concerning to the disease diagnosis falls in the negative class and only the remaining 10 percent of the instances fall in the positive class. Assuming that a classifier predicts every instance of the dataset into the negative class, it achieves 90 percent accuracy, which sounds reasonable. However, the classifier has misclassified all the positive class instances into the negative class. Since the classifier does not diagnose a single positive case, it is a complete failure as a disease diagnosis tool despite showing 90 percent accuracy. In this context, the sensitivity and specificity are far more important as the performance evaluation metrics than the accuracy for disease diagnosis systems [3], [4]. The other important performance valuation metrics for classifiers in the presence of class skew are geometric mean (G-mean) of the sensitivity and specificity and Area under ROC (AUC) [5], [6].

Disease diagnosis involves the tasks of classification. The aim of a classification algorithm is to learn the mapping between the predicting attributes and the target attribute from training data. The relationship is later used to forecast the class labels of the new instances. For disease diagnosis, the predicting attributes capture the data regarding the personal profile of the patients, their family history, the symptoms reported by them and the outcomes of various diagnostics tests. In contrast, the class labels involve diagnostic flags that mark the presence or absence of the disease. Many classification algorithms such as K-Nearest Neighbours (KNN), Support Vector Machine (SVM), Bayesian Networks Neural Networks (NNs) and Decision Trees have been applied for medical diagnosis including coronary heart disease, liver disease, cancer diseases, diabetes, Parkinson disease and many more diseases [7].

Trusting the outcome of disease diagnostic machine learning algorithms is a question of life and death for medical experts. Whatsoever the accurate classification algorithm is, a medical expert would never believe its outcome unless s/he understands and is convinced of the whole process of the classifier that predicts the patients into positives and negative classes. Therefore, black box classifiers like SVM are not acceptable for medical diagnosis. Decision tree classifiers, one of the most widely used classifiers, are well known for their interpretability [7], [8]. Moreover, the random forest classifiers are not as comprehensible as the decision tree classifiers. However, random forest classifiers outperform the decision tree classifiers in all the evaluation metrics.

Due to class skew, the classification algorithms are favours the majority class (non-diseased cases), whereas predicting the instances for the minority class (positive cases) correctly is far more critical for disease diagnosis. Researchers have used many data resampling means for resolving the class imbalance problem. Some of the commonly used traditional sampling methods are oversampling, undersampling, Synthetic Minority Oversampling Techniques (SMOTE), Borderline SMOTE and Adaptive Synthetic Sampling Techniques (ADASYN), Minority Weighted [9]–[11]. Two relatively novel data resampling techniques to solve the issue of skewed class distribution in an effective manner are Majority Weighted Minority Oversampling Technique (MWMOTE) and a Random Walk Oversampling (RWO) [12], [13]. Selecting an appropriate resampling technique to address the class imbalance is an imperative decision that influences the outcome of medical diagnostic systems.

The weights of a NN classifier, the support vectors of an SVM classifier, and the estimated coefficients in a regression problem are known as parameters of these models. These parameters are estimated during the learning phase from training data. In contrast, the user specified parameters such as learning rate for NN classifiers, choice of kernel function for SVM, the size of the neighbourhood for KNN classifier and depth and splitting criteria for decision trees, etc. called hyperparameters. The hyperparameter configuration of a classification algorithm influences its performance to a great extent. For instance, the predictive performance and comprehensibility of a decision tree classifier depend upon the maximum permitted depth, the certain number of cases needed in a node for a further split, the amount of acceptable pruning and node splitting criteria adopted etc. Determining appropriate values for these hyperparameters for decision tree classifiers is a very crucial issue. Moreover, no hyperparameter setting works uniformly fine across all the datasets. A set of hyperparameter values may work well for one dataset, but the same may fail for another dataset. Therefore, hyperparameters need tuning for each and every dataset individually. Finding optimal values for hyperparameters for any machine learning algorithm is a difficult optimization problem with a large search space [14]–[16]. Hence, several efforts have been made to enhance the predictive power of classifiers through optimizing their hyperparameters by using evolutionary and other metaheuristic approaches. However, most of these approaches tune the parameters for optimizing a single performance criterion like accuracy or F-measure etc. [15]–[22].

For disease diagnosis, the classification algorithm needs to be optimized with respect to sensitivity as well as specificity. Ideally, we would like to maximize both. However, a tradeoff always exists between the sensitivity and specificity of a classifier. Any attempt to bring an increase in sensitivity of the classifier results is some decrease in specificity and vice-versa. In such a situation, there is no single optimized solution for the problem as such, and we need to depend upon the techniques that find a set of non-dominated set of solutions. A solution S2 is dominated by S1 solution if and only if S1 is not worse than S2 in any of the objectives, and S1 is strictly better than S2 in at least one of the objectives. A set of non-dominated solutions contains a set of diverse solutions which do not get dominated by any other solutions [23]. The field of disease diagnosis invites a multi-objective treatment. There are only a few works that optimize the hyperparameters of the classifier algorithms in a multi-objective framework [24]–[28].

This paper presents a multi-objective hyperparameter tuning approach, called MOHPT, for disease diagnostics. The class imbalance is dealt with by selecting an appropriate resampling technique for each dataset. Further, a set of diverse hyperparameter configurations is obtained instead of a single best hyperparameter configuration on a subset of data called validation dataset. Each set of hyperparameter configurations in the solution set corresponds to a specific trade-off between sensitivity and specificity. The overall algorithm returns the best combination of sampling technique and Pareto sets of hyperparameter configurations. Out of the pareto optimal solutions, the medical experts have the liberty to select one according to their preferences. Subsequently, a classifier model is built by using the combination of the selected sampling technique and hyperparameter configuration with the help of training data and its performance is evaluated on test data. We have applied the suggested MOHPT to 17 medical datasets for disease diagnosis and the results provide valuable insights on the use of decision tree (CART) and random forest classification algorithms in a multi-objective perspective for disease diagnosis.

The remaining paper contents are organized as follows. Section 2 gives the contextual details to better understand the research carried out in this paper. Section 3 describes the problem statement. Section 4 illustrates the proposed MOHPT with two variants decision tree (DT) and random forest (RF) method. Section 5 presents the results and the comparison of the suggested approach to similar works. Section 6 concludes the paper.

2. The Related Work

There is an ample number of works published on disease diagnosis, and it is not possible to consider all of them in a single work. Some review papers available in the data mining and machine learning field provide a fairly good discussion of the classifiers that have been extensively applied for diagnosing a diverse spectrum of diseases [2], [29]. In the context of the disease diagnosis domain, the review papers incorporate diagnosis of diseases concerning cancer, appendicitis, cardio-vascular, skin, liver, thyroid, fertility, brain tumours, Parkinson's disease, hepatitis and many more [8], [30]. Recently, a widespread investigation of 32 classification approaches has been implemented to identify the most suitable method for disease diagnosis [7]. The authors concluded that the tree-based classification methods outperformed the other 31 methods. Being the most comprehensible model across the numerous existing classification methods, decision trees are the preferred predictive models, particularly for the domains like medical diagnosis, where the main focus is on the reasons that cause the prediction rather than the prediction itself [31].

However, decision trees are created using top-down greedy approach; these may result in sub-optimal performance. Different training data of the same datasets may lead to different decision tree classifiers. This problem has moved the focus of researchers towards ensemble classifiers for improved prediction performance. Random forest algorithms are the outcome of ensemble learning of decision trees where multiple trees are grown from different training samples and classification of new instances is carried out by majority vote [32]. Ensemble of classifiers is ascertained to be a very successful model to improve classification accuracy since these can overcome the uncorrelated miscalculation by a single classifier through combining the estimations of various base classifiers. Random Forest (RF) classifiers have given improved performances in disease prediction [33], [34]. Trees in RF are grown by the CART method with no pruning. Some authors have also combined sampling techniques with RF classifiers to overcome class imbalance and achieved promising results [35], [36]. We use CART and random forest classifiers in this research.

2.1 *Sampling techniques for disease diagnosis*

As described earlier, medical datasets have a skewed distribution of class labels, i.e., only have a few diseases cases (positive class) in contrast to the non-diseases cases (negative class). Since the classifier has a learning bias towards the negative class, the classifier fails in making accurate predictions for the positive class cases, and hence the classifier has a high false negative rate. A high false negative rate poses a serious problem in disease diagnosis [37].

Researchers have used many sampling techniques to deal with the class imbalance problem [38]–[40]. The sampling techniques mainly include undersampling and oversampling. These techniques intend to balance the ratio of the positive and negatives cases in the data. The undersampling technique reduces the examples of the majority class. As undersampling may remove some vital information related to positive class, it may deteriorate the accuracy of a classifier in the majority class. From the review of the literature, it is evident that oversampling techniques outperform undersampling techniques. On the other hand, oversampling methods scaled up the data samples for the minority class. Random oversampling of the minority class example may end up adding some redundant data examples, which leads to an overfitted model [38], [41].

There is another class of oversampling techniques called synthetic oversampling. These techniques add synthetically generated examples of minority classes to balance the imbalanced class distribution. The popular examples of synthetic oversampling are Synthetic Minority Oversampling Techniques (SMOTE), Borderline SMOTE and Adaptive Synthetic Sampling Techniques (ADASYN) [9]–[11]. Synthetic oversampling methods allocate weights to the minority class examples. Examples of minority classes with more weight participate more in generating synthetic examples to cover class imbalance. The performance of synthetic oversampling techniques obviously depends upon the assignment of proper weights of the minority class samples. SMOTE, Border-line SMOTE and ADASYN do not guarantee that the new samples generated will always be of the minority class when samples from both classes have overlapping values. These techniques may also generate noisy instances as well. Barua et al. [42] have devised an advanced Majority Weighted Minority Oversampling Technique (MWMOTE) to improve the sample selection and synthetic example creation schemes. MWMOTE initially discovers the most critical and difficult to learn samples of minority classes and allots them weights corresponding to their Euclidean distance of the closest majority class examples. Subsequently, it creates the synthetic example of minority class from the weighted explanatory minority class samples employing clustering. The technique generates new samples in such a way that these lie in some clusters of the minority class, and it also removes the noisy instances from the newly sampled synthetic dataset. MWMOTE has been evaluated extensively and the result showed that it performs better than the other existing approaches [12]. Zhang and Li [13] proposed another synthetic oversampling technique named Random Walk Oversampling. The technique uses the mean and variance of various attributes for balancing the unequal distribution of data by using the central limit theorem. It generates synthetic data for positive class based on standard deviation and mean of the attribute values that fall in the positive class.

2.2 *Hyperparameter optimization of tree classifiers*

Like other Machine Leaning methods, decision tree and random forest classifiers also have hyperparameters that directly impact the functioning of these algorithms. The performance of decision trees and random forest algorithms can be significantly improved by tuning their hyperparameters. Due to the huge number of combinations for these hyper-parameter values, researchers use optimization techniques to find the optimal hyperparameter configuration. There are many search strategies for tuning hyperparameters. Some of these list all the possible candidate hyperparameter configurations (discretized) in advance and pursue the best one. Random search and grid search are two such strategies. Some other methods applied in the literature include model based optimization (MBO), Gaussian Process Approach, Tree-structured Parzen Estimator Approach, Sequential Search and Random Search [16], [18], [20], [21], [43].

Finding optimal hyperparameter configuration is a combinatorial optimization problem, and therefore, many authors have used meta-heuristics techniques, namely Genetic Algorithm [44], [45], Particle Swarm Optimization [46], simulated annealing [47] etc. However, in most of the studies involving decision trees and random forest applications to disease diagnosis, a single criterion like cross-validation error is optimized through tuning of hyperparameters. However, in the case of highly imbalanced disease datasets, mere optimization of the accuracy of a classifier would not suffice. Hence, there is a need for optimization of multiple conflicting objectives simultaneously, viz. specificity and sensitivity [48], [49].

2.3. *Multi-objective parameter optimization*

When we need to simultaneously optimize several contradictory objectives, we cannot arrive at a unique optimal solution. In such situations, there exists a trade-off between the objectives. This means that optimizing one of the objectives to its maximum (minimum) value may deteriorate the values for one or more of the other objectives. Hence, we depend upon a set of non-dominated solutions. The two most common approaches to address multi-objective problems are i) weighted sum and ii) Pareto dominance. The first approach is employed to adapt a multi-objective problem into a single objective through adjusting numeric weights to the different objectives and subsequently merging values of these objectives into a corresponding single value by using some arithmetic formula. This is easy to implement and also computationally less expensive. However, determining weights for various objectives is an uphill task and one needs the expertise of the highest order to determine weights. Most often, these weights are subjectively assigned by the user based on intuition which often leads to a

sub-optimal solution [23]. This draws our attention to the question of how to optimize the classifier for more than one objective. This requires determining optimal classifier hyperparameter configurations according to non-dominance Pareto optimal methods. To meet this requirement, we need to depend upon the Pareto dominance approach, which provides several diverse solutions, each of the solutions is optimal with respect to one or the other objective. The set of objective values of the non-dominated solutions is termed as Pareto front. Some researchers have applied multi-objective optimization to tune the hyperparameters of classifiers [18], [49].

3. Problem Statements and Objective

To solve the multi-objective hyperparameter tunning problem, we need to include all the different hyperparameter configurations of the classifier in the solution space. These configurations build classifier models that result in diverse trade-offs between sensitivity and specificity. The Pareto optimal solutions are those solutions that cannot be improved in one objective without getting worse in another objective. The formal description of the multi-objective hyperparameter optimization problem is as follows.

Assume that we have a set of n decision tree-based classifiers. Let us represent this set by $T = (t_1, t_2 \dots t_n)$. Each classifier in the set T has a unique set of values for k hyperparameters $S(s_1, s_2, \dots s_k)$. In a multi-objective framework, the classifiers in the set T are evaluated on M objective functions, i.e., $F(t_i) = (f_1(t_i),\ f_2(t_i), \dots,\ f_m(t_i)$.

A classifier $t_i \in T$ is said to dominate a classifier $t_j \in T$, if for all objective functions f_m, $f_m(t_i) \preccurlyeq f_m(t_j)$, $m \epsilon\ (1, \dots, M)$ and if there is at least one objective $f_{m^*} m^* \in\ (1, \dots, M$ for which $f_m(t_i) \prec f_m(t_j)$.

The symbols $\preccurlyeq$ and $\prec$ stand for better and better or equal, respectively. A higher (lesser) value for an objective function is considered better for the maximization (minimization) problem. A classifier t_i is known as Pareto-optimal, if there exists no classifier t_j in T that dominates it. We call the corresponding parameter configuration $s_i \in S$ of the tree as the Pareto optimal parameter configuration.

The objectives of this research are to locate the optimal set of all the Pareto optimal hyperparameter configurations regarding the two objectives, i.e., sensitivity and specificity. Subsequently, the corresponding set of non-dominated Pareto optimal hyperparameter configurations is chosen from the input parameter space P(S). From the set of Pareto optimal solutions, an expert can select any one hyperparameter setting that produces the required trade-off of sensitivity and specificity for disease diagnosis.

4. The Proposed Method (MOHPT)

Inspired by the problem identified in the preceding section, we suggest the MOHPT algorithm to capture diverse optimal configurations of hyperparameters in the form of non-dominated optimal solutions based on the objective functions provided by the user. The MOHPT method uses a decision tree (CART) and random forest as the two classifiers. To begin with, the proposed technique finds the best sampling method for each dataset individually for balancing the ratio of the class labels using a validation dataset. Next, it samples a population of hyperparameter configurations from the search space using the Niederreiter technique [50]. The Niederreiter technique is a well-tested technique that picks up distinct combinations of values uniformly from all over the search space. The technique makes sure that no area of the search space is over or under-represented in the sample. Thereafter, the performance of various hyperparameter configurations in the population is evaluated based on the objective function. Subsequently, the MOHPT algorithm forms the non-dominated Pareto optimal fronts of hyperparameter configurations by ranking the solutions using the crowding technique envisaged by NSGA-II, a well-known multi-objective genetic algorithm [51]. In the end, we get Pareto optimal non-dominated solutions with various levels of the trade-off between the objective functions. The same process is repeated 20 times. The best Pareto-optimal front out of all such fronts over the generations is selected. A user can select any hyperparameter configuration according to his/her preference from all the non-dominated solutions. In this study, we select the hyperparameter configuration that results in maximized G-mean. This section describes the overall design of the multi-objective MOHPT method in a stepwise manner.

4.1 *Partitioning datasets*

The proposed methodology initially divides the data into training data (D_{Train}), and validation data ($D_{Validation}$). The tuning of the hyperparameters is carried out on the $D_{Validatio}$ dataset. The best sampling technique is applied to D_{Train} data to address the class imbalance problem. The sampled data is further split into training-train ($D_{training\text{-}train}$) and training-test ($D_{training\text{-}test}$) datasets. Finally, the classifier is trained on $D_{training\text{-}train}$ using the optimal hyperparameters configuration resulting from MOHPT algorithm. The classifiers are evaluated on the $D_{training\text{-}test}$.

4.2 *Addressing the class imbalance*

The disease diagnosis datasets usually suffer from the class imbalance problem. To overcome this problem, we implement the following seven sampling techniques.

Random Undersampling (RU)

Random Oversampling (RO),

A combination of both over and undersampling (BOTH)

Random Over-Sampling Examples (ROSE),

Synthetic Minority Oversampling Technique (SMOTE)

Majority Weighted Minority Over Sampling Technique (MWMOTE)

Random Walk Oversampling (RWO).

No single sampling technique works well across all the datasets. The performance of sampling techniques may vary from one dataset to another and it depends upon the distribution of values of various attributes, their relationship with the class attribute and the amount of the class imbalance present in the data. We apply the grid search in conjunction with the multi-objective approach to select the best sampling technique and the corresponding set of optimal non-dominated solutions for each dataset. The MOHPT method uses the $D_{Validation}$ dataset for tuning the hyperparameters.

4.3 *Hyperparameters of decision tree and random forest classifiers*

This section describes the hyperparameters of the CART and random forest classifiers that are considered for optimization in this study.

4.3.1 Hyperparameters for classification and regression tree algorithm (CART)

Node splitting criteria: We use deviance and Gini index as the node splitting criteria. Both of these are measures of the amount of impurity in the dispersal of class labels of the instances in the resulting nodes. The deviance of a decision tree is stated as

$$D = \sum D_i\,,\ D_i = -2\sum_k n_{ik}\ log p_{ik} \tag{1}$$

where n_{ik} represent the number of instances of the kth class in the ith node; p_{ik} is the probability of an instance belonging to the kth class in the ith leaf. Further, assume that a node splits into two nodes t and u on an attribute. This changes the probability model and reduces the deviance given by the formula as below.

$$D_s - D_t - D_u = 2\sum_k [n_{tk}\ log\frac{P_{tk}}{P_{sk}} + n_{uk} log\frac{P_{uk}}{P_{sk}} \tag{2}$$

The CART algorithm picks the splitting attribute that produces the maximum decrease in the deviance of the resulting decision tree. The other criterion for CART for splitting the nodes is the Gini index which is defined as below.

$$Gini(s) = 1 - \sum_{k=1}^{m} p_k^2, \tag{3}$$

The symbol p_k is the probability of an instance from the kth class and it is estimated from training data at the root node. Considering that the root node is subdivided into two nodes t and u by producing binary splits based on the values of attribute A, the Gini index is computed as below.

$$Gini_A(s) = \frac{|n_t|}{|s|} Gini(n_t) + \frac{|n_u|}{|s|} Gini(n_u) \tag{4}$$

where $|n_t|, |n_u|$ and |s| are the total number of cases in the t, u and s nodes respectively.

$$\Delta Gini(A) = Gini(s) - Gini_A(s) \tag{5}$$

Attribute A that maximizes the $\Delta Gini(A)$ is preferred for inducing a node split.

Mincut: This hyperparameter determines the minimum number of observations to be included in any child node. This controls the smallest allowed node size, which is double the amount of the value of the *mincut*. The larger values of *mincut* grow smaller trees.

Mindev: This hyperparameter refers to the within-node deviance and controls the complexity of the tree. A node split takes place only if its within-node deviance is at least a user-defined threshold times that of the root node.

4.3.2 Hyperparameters for random forest algorithm

Ntree: This hyperparameter decides the number of trees to be grown in the random forest. The value of the parameters should be set appropriately to ensure that every instance gets predicted at least a few times. Setting a very small value for the parameter compromises the predictive performance of the algorithm. A large value for the hyperparameter increases the computational overhead of the algorithm for no reasonable advantages.

Mtry: This hyperparameter is related to the number of attributes randomly selected as the candidate for each split. The default value for the hyperparameters is the square root of the overall attributes in the data.

Maxnodes: This hyperparameter regulates the number of terminal nodes in the trees of the random forest. By default, the trees grow to a maximum size. The appropriate value for the parameter is required to avoid the overfitting or under fitting of the model.

4.4 *Initial population*

For the purpose of optimization, all the possible hyperparameter configurations make the hyperparameter input space (*HIPS*) for optimization. The number of configurations is often exhaustively large and evaluating each of these individually is computationally expensive. Therefore, we employ a sampling technique to pick up a limited number of diverse hyperparameter configurations from the search space. Most often, the candidate solutions in the initial population are created uniformly randomly. However, this does not guarantee an even distribution of the parameter configurations from *HIPS*. Therefore, we apply the Niederreiter technique [50], which is based on the use of quasi-random low discrepancy sequences. The sequence is designed to ensure the distribution of the sampled points evenly in an interval or a hypercube. The Niederreiter sequence of dimension d and base b is defined as:

$$X^n = \left(x_1(n), \dots x_d(n)\right) \tag{6}$$

with

$$x_1(n) = \sum_{j=1}^{\infty} c_{jr}^{i} \psi_r(a_r(n-1)) \tag{7}$$

Here, the computations are done in a commutative ring R with identity and cardinality b. The values of $\psi_r: \{0,1, \dots, b-1\} \to R$ and $\lambda_{ij}: R \to \{0, 1, \dots, b-1\}$ are appropriately chosen bijections and c_{jr}^{i} are suitable preferred elements of R. The a_r denote the digits of the base b representation of n. The reader can refer [50] for more details about this technique. We sample 20 hyperparameter configurations from the search space in each generation.

4.5 Objective functions

In a multi-objective perspective, the fitness of an individual *I* is a vector of *M* objective function values, $F(I_i) = (f_1(I_i), f_2(I_i), \dots, f_M(I_i))$. For medical diagnosis, the most popular objective is to find the solutions with various amounts of trade-offs between sensitivity and specificity. Therefore, the MOHPT approach uses these two objectives for arriving at a set of non-dominated optimal solutions.

$$fitness(I) = \left(sensitivity(I_i), specificity(I_i)\right) \tag{8}$$

Where I_i denotes a particular hyperparameter configuration

4.6 Multi-objective optimization

The MOHPT applies a tune-pareto approach inspired by NSGA-II, which was devised by [51]. The NSGA-II is a promising multi-objective genetic algorithm for evolving solutions in a multi-objective viewpoint. The MOHPT method employs a crowding distance function from NSGA-II to obtain Pareto optimal solutions.

4.7 Stopping criteria

The whole process is repeated 20 times. The best non-dominated solutions set of the hyperparameter configuration (*OHP*) is selected from all the generations.

4.8 Selection of hyperparameter configuration

Among all the non-dominated solutions produced by the different sampling schemes on the $D_{Validation}$, we select the best pareto-optimal front of solutions. For picking up a solution from the Pareto-optimal front, we choose the one with the highest G-mean. Subsequently, the classifier is trained on the optimal hyperparameter configuration and it is evaluated on test data. The set of non-dominated solutions and their corresponding hyperparameter configurations for a given sampling technique is shown in Table 1 for a sample dataset. The solution in the last row is the optimal solution. The overall working of the proposed approach is illustrated in Fig. 1.

Table 1. The non-dominated solutions using MOHPT algorithm for Abalone dataset.

Sno.	Hyperparameter Configurations	Sensitivity	Specificity	G-mean
1.	split = deviance, mincut = 50, mindev = 0.009	0.611	0.735	0.449
2.	split = gini, mincut = 35, mindev = 0.007	0.692	0.687	0.475
3.	split = deviance, mincut = 25, mindev = 0.01	0.580	0.770	0.447
4.	split = deviance, mincut = 50, mindev = 0.003	0.753	0.656	0.494
5.	split = deviance, mincut = 10, mindev = 0.009	0.601	0.747	0.449
6.	split = deviance, mincut = 30, mindev = 0.007	0.654	0.7145	0.467
7.	**split = deviance, mincut = 20, mindev = 0.003**	**0.740**	**0.678**	**0.500**

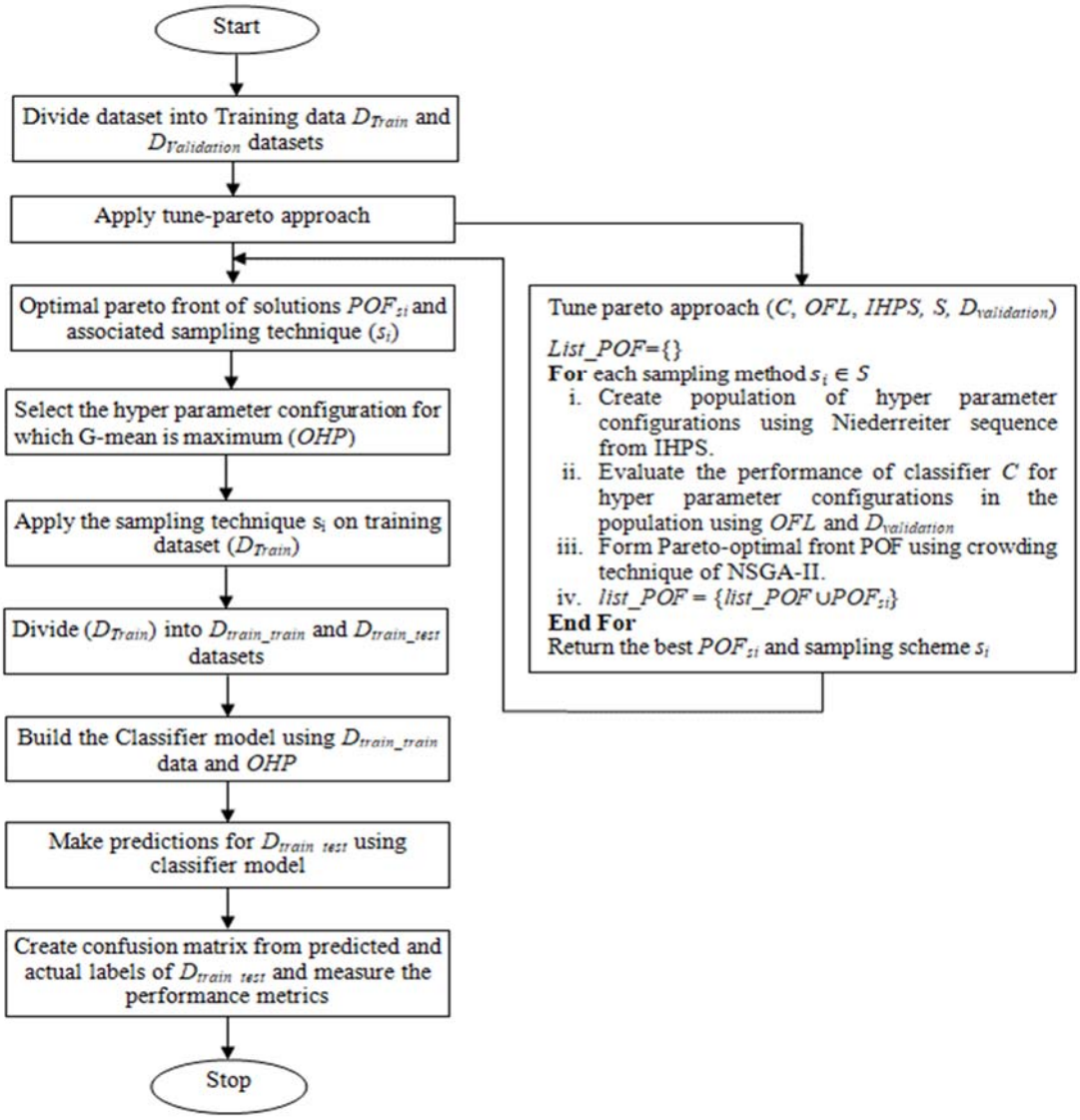


Fig. 1. The MOHPT system for optimizing hyperparameters.

5. Experimental Design and Results

This section evaluates the performance of the proposed MOHPT for hyperparameter tuning of CART and RF classifiers. The section compares the performance of the suggested approach with the related approaches and presents the analysis of the results.

5.1 Datasets

We use 17 datasets related to disease diagnosis which are taken from UCI and Kaggle repositories. These datasets have been frequently used in the prevailing research works on disease diagnosis. A brief description of the datasets is given in Table 2. Each row of the table includes the number of instances, number of attributes, frequencies and percentage instances in the majority and minority classes of the dataset. The ratio of instances in the majority and minority classes varies from 65:35 to 99:1.

Data pre-processing is essential to get meaningful results from any machine learning algorithm. At the pre-processing stage, we have removed the examples with missing values and unnecessary attributes such as sequence number, patient ID, address, etc., from the datasets. The first eight datasets contain the continuous type of attributes, and the remaining nine datasets comprise discrete as well as continuous attributes

Table 2. Details description of diseases datasets.

S. No.	Datasets	#Instances	#Attribute	Class-1	Freq-1	Class-2	Freq-2	%Majority	%Minority
1	Diabetes	768	9	negative	500	positive	268	65	35
2	Ecoli	281	8	negative	274	positive	7	98	2
3	Haberman	306	4	1	225	2	81	74	26
4	Parkinson	195	23	1	147	0	48	75	25
5	Vertebral Column	310	7	ab	210	no	100	68	32
6	Liver ILPD	583	11	1	416	2	167	71	29
7	WPBC	198	34	N	151	R	47	76	24
8	Yeast5	1484	9	negative	1440	positive	44	97	3
9	Abalone19	4174	9	negative	4142	positive	32	99	1
10	Cervical cancer rf	668	34	0	623	1	45	93	7
11	Diabetic Mellitus	279	98	0	99	1	180	35	65
12	Hepatitis	80	20	live	67	die	13	84	16
13	Saheart	462	10	1	302	2	160	65	35
14	Sick	3103	28	negative	2888	sick	215	93	7
15	Spect heart	267	23	0	55	1	212	79	21
16	Thoraric	470	17	false	400	true	70	85	15
17	Thyroid	7200	22	2	534	3	6666	93	7

5.2 Performance metrics

Since accuracy cannot be the only criteria for assessing the performance of classifiers in disease diagnosis, we have evaluated the proposed and related approaches on geometric-mean (G-mean) and area under receiver operating curve (AUC) in addition to accuracy. G-means gives an overall performance of classifiers by combining their accuracy on the majority and minority classes. A high value for this measure means that the classifier does well on the majority as well as a minority class. Since AUC is not sensitive to the class imbalance ratio, it is considered a promising metric for evaluating and comparing classifiers on imbalanced datasets, like in the case of disease diagnosis. The AUC metric computes the area under the ROC. The formulations for the various performance metrics are given below.

$$Accuracy = \frac{TP+TN}{N} \tag{9}$$

Where N represents the number of examples in the test data.

$$\text{Sensitivity} = \frac{TP}{TP+FN} \tag{10}$$

$$Specificity = \frac{TN}{TN+FP} \tag{11}$$

$$Gmean = Sensitivity \times Specificity \tag{12}$$

5.3 Parameter settings

This study tunes the respective hyperparameters of CART and RF classifiers in a multi-objective perspective. Table 3 gives the names of the hyperparameters, their significance and the range of their input values. The rest of the parameters of the classifier algorithms were kept at their default values. The sample size for extracting the combinations of values for different parameters is set to 20. The Niederreiter sampling is repeated over 20 generations.

Table 3. Range of values for MOHPT hyperparameters for decision tree and random forest classifier algorithms.

CART tree		**Random Forest**	
Attribute	**Values**	**Attribute**	**Values**
split(splitting criteria)	gini, deviance	ntree	10:200
mincut	5:50	mtry	2:sqrt(#attrib.)
mindev	seq(0.001,0.01,0.001)	maxnode	3:50
number of combination	20	Number of combinations	20
number of iterations	20	number of iterations	20
Objective functions	Specificity,Sensitivity	objective functions	Specificity,Sensitivity

5.4 Implementation tools

The proposed hyperparameter tuning multi-objective algorithm is implemented in R, an open source platform for statistical analysis and data analytics. We have used many packages from the CRAN repository of R for implementing the whole approach. The packages named 'tree' and 'randomforest' are used to implement CART and random forest classifiers. We have used 'DMwR', 'ROSE' and 'imbalance' packages for employing the various sampling techniques for tackling the class imbalance. The 'caret' package is used for assessing the classifiers' performance. Lastly, for carrying out multi-objective optimization of hyperparameters, we have used the 'tunepareto' package [50]. The package includes predefined interfaces to frequently used classifiers, i.e, KNN, SVM, CART, Random Forest and Naïve Bayes for optimizing their hyperparameters. It provides generic methods for hyperparameter tuning of classification algorithms using multiple objective functions such as error rate, sensitivity, specificity, etc.

5.5 Results and analysis

We perform four sets of experiments for each classifier. We run each of the two classifiers (Decision tree (CART) and Random Forest) with and without applying the sampling techniques. Further, the tune-pareto versions of the classifiers are run with and without sampling. Overall, we evaluate eight variations of the two classifiers. The four variations of the decision tree algorithm are abbreviated as ST- Simple Tree, STS – Simple Tree using Sampling, TTP - Tree with Tune-Pareto and TTPS-Tree with Tune-Pareto using sampling. Similarly, the four variations of the random forest algorithm are named RF – Random Forest, RFS – RF using Sampling, RFTP – RF with Tune-Pareto approach and RFTPS - RFTP using sampling. The results reported are the average over the ten runs of each experiment.

The simulation results of the various classifiers are given in tables 4, 5, and 6. The first two columns of the tables list the optimal sampling methods for decision trees and random forest classifiers, respectively, for addressing the class imbalance in the various disease diagnosis datasets. The results show that no single sampling technique is suitable across all the datasets. The Mwmote and Rwo prove to be optimal methods for the datasets that have continuous attributes. Likewise, for the datasets that have continuous as well as discrete attributes, oversampling or a combination of oversampling and undersampling techniques corroborate to be ideal techniques for resolving the class imbalance issue.

Tables 4, 5 and 6 list the results of accuracy, G-mean and AUC of the various versions of the two classifiers. The best values of the performance metrics are highlighted in boldface. The performance of classifiers is ranked for analysis of the results. The performance metric values are ranked from 1 to 4. The lower is the rank, better the performance of the classifier. If the two classifiers have the same values for a performance metric, then these are assigned the average of the two consecutive ranks. The last row in the tables contains the average ranks of the classifiers over all the datasets. The least average rank signifies the best performance. We notice from tables 4, 5 and 6 that the CART and RF classifiers with sampling and Tune-Pareto approach (TTPS and RFTPS) are the winner algorithms with the lowest ranks- for accuracy (1.79, 1.91), for G-mean (1.30, 1.65), AUC (1.32, 1.89) respectively. The tables also show that classifiers with sampling (STS, TTPS, RFS, RFTPS) consistently outperforms the ones without sampling (ST, TTP, RF, RFTP). Further, various random forest classifiers (RF, RFS, RFTP, RFTPS) achieve better accuracy, G-mean and AUC than their decision tree counterparts (ST, STS, TTP, TTPS).

Table 4. Classification accuracies of the variants of Decision Tree and Random Forest algorithms.

Dataset	Tree Classifier					Random Forest Classifier				
	Sampling Method	Accuracy				Sampling Method	Accuracy			
		ST	STS	TTP	TTPS		RF	RFS	RFTP	RFTPS
Diabetes	Mwmote	0.81(2)	0.79(3)	0.77(4)	***0.82 (1)***	Rwo	0.9(3)	0.94(1)	0.88(4)	***0.92(2)***
Ecoli	Mwmote	***0.97(3.5)***	0.98(2)	0.97(3.5)	***0.99(1)***	Mwmote	0.99 (3)	0.99 (3)	0.99 (3)	***1.00 (1)***
Haberman	Rwo	0.70(4)	0.73(3)	***0.80 (1)***	0.76 (2)	Mwmote	0.89(2)	0.86(3.5)	0.86(3.5)	***0.9(1)***
Parkinsons	Over	0.87(4)	0.94(2.5)	0.94(2.5)	***0.95 (1)***	Over	0.95(4)	0.97(3)	0.96(2)	***0.99(1)***
Vertebral	Rwo	***0.8 (3.5)***	0.88(2)	0.84(3.5)	0.91 (1)	Rwo	0.93(3.5)	0.96(2)	0.93(3.5)	***0.98(1)***
Liver	Over	0.80(3)	***0.83(1)***	0.79(4)	0.82 (2)	Over	0.89(3)	0.92(2)	0.86(4)	***0.93(1)***
WPBC	Rwo	0.76(4)	***0.90(1.5)***	0.84(3)	***0.90(1.5)***	Rwo	0.90(4)	0.92(2)	0.91(3)	***0.95(1)***
Yeast5	Mwmote	***0.99(2)***	***0.99(2)***	0.98(4)	***0.99(2)***	Rwo	***1.00(3)***	***1.00(3)***	0.99(4)	***1.00(3)***
Abalone	Over	0.99(2.5)	0.93(4)	0.99(2.5)	***1.00(1)***	Over	***1.00(2.5)***	***1.00(2.5)***	***1.00(2.5)***	***1.00(2.5)***
Cervical C	Over	***0.97(1.5)***	***0.97(1.5)***	0.96(3.5)	0.96(3.5)	Both	0.97(3)	***0.99(1.5)***	0.96(4)	***0.99(1.5)***
Diabetic_m	Both	0.98(4)	***0.99(2)***	***0.99 (2)***	***0.99(2)***	Over	***1.00(3)***	***1.00(3)***	0.99(4)	***1.00(3)***
Hepatitis	Both	0.95(4)	***0.98(1.5)***	***0.98(1.5)***	0.96(3)	Both	0.94(4)	***0.98(1.5)***	0.95(3)	***0.98(1.5)***
Saheart	Over	0.77(4)	0.80(2)	0.79(3)	***0.81(1)***	Over	0.87(2)	***0.9(1)***	0.83(4)	0.85(3)
Sick	Over	0.93(4)	0.94(2.5)	0.94(2.5)	***0.95(1)***	Over	***0.96(1.5)***	***0.96(1.5)***	0.94(3)	0.92(4)
Spheart	Rose	0.82(3.5)	0.82(3.5)	***0.87(1)***	0.84(2)	Rose	0.88(3)	***0.93(1.5)***	0.86(4)	***0.93(1.5)***
Thoracic	Over	0.81(4)	***0.88(1)***	0.85(3)	0.87(2)	Both	0.93(3)	***0.97(1)***	0.92(4)	0.95(2)
thyroid	Over	***1.00(2)***	***1.00(2)***	0.99(4)	***1.00(2)***	Over	***1.00(2.5)***	***1.00(2.5)***	***1.00(2.5)***	***1.00(2.5)***
	-	2.91	2.17	2.85	**1.79**	-	2.94	2.09	3.41	**1.91**

Table 5. Geometric mean on the variants of Decision Tree and Random Forest algorithms.

Dataset	Tree Classifier					Random Forest Classifier				
	Sampling Method	G-mean				Sampling Method	G-mean			
		ST	STS	TTP	TTPS		RF	RFS	RFTP	RFTPS
Diabetes	Mwmote	0.77(3)	0.79(2)	0.74(4)	***0.82(1)***	Rwo	0.89(3)	0.94(1)	0.85(4)	**0.92(2)**
Ecoli	Mwmote	0.7(3)	0.98(2)	0.69(4)	***0.99(1)***	Mwmote	0.80(4)	0.99(3)	**1.00(1.5)**	**1.00(1.5)**
Haberman	Rwo	0.51(4)	0.72(2)	0.68(3)	0.76(1)	Mwmote	0.84(3)	0.86(2)	0.78(4)	**0.90(1)**
Parkinsons	Over	0.81(4)	0.93(3)	0.94(2)	***0.95(1)***	Over	0.92(3)	0.97(2)	0.91(4)	**0.99(1)**
Vertebral	Rwo	0.91(1.5)	0.88(3)	0.84(4)	0.91(1.5)	Rwo	0.91(3.5)	0.96(2)	0.91(3.5)	**0.98(1)**
Liver	Over	0.79(3)	**0.83(1)**	0.67(4)	0.82(2)	Over	0.88(3)	0.92(1.5)	0.74(4)	**0.92(1.5)**
WPBC	Rwo	0.64(4)	**0.90(1.5)**	0.77(3)	***0.90(1.5)***	Rwo	0.81(4)	0.94(2)	0.82(3)	**0.95(1)**
Yeast5	Mwmote	0.92(3)	**0.99(1.5)**	0.83(4)	***0.99(1.5)***	Rwo	0.97(3)	**1.00(1.5)**	0.88(4)	**1.00(1.5)**
Abalone	Over	0.64(3)	0.93(2)	0.48(4)	***1.00(1)***	Over	0.66(3.5)	**1.00(1)**	0.66(3.5)	0.99(2)
Cervical c	Over	0.88(3)	0.93(2)	0.76(4)	0.96(1)	Both	0.87(3)	**0.99(1.5)**	0.72(4)	**0.99(1.5)**
Diabetic_m	Both	0.97(4)	**0.99(2)**	**0.99(2)**	***0.99(2)***	Over	0.99(3.5)	**1.00(1.5)**	0.99(3.5)	**1.00(1.5)**
Hepatitis	Both	0.79(4)	**0.97(1)**	0.89(3)	0.96(2)	Both	0.59(4)	**0.98(1)**	0.7(3)	0.97(2)
Saheart	Over	0.73(4)	0.80(2)	0.74(3)	***0.81(1)***	Over	0.85(2.5)	**0.9(1)**	0.77(4)	0.85(2.5)
Sick	Over	0.47(3)	0.81(2)	0.43(4)	***0.95(1)***	Over	0.66(3)	**0.96(1)**	0.49(4)	0.92(2)
Spheart	Rose	0.68(4)	0.81(2)	0.72(3)	0.84(1)	Rose	0.78(3)	**0.93(1.5)**	0.73(4)	**0.93(1.5)**
Thoracic	Over	0.48(4)	0.86(2)	0.62(3)	0.87(1)	Both	0.73(3.5)	**0.97(1)**	0.73(3.5)	0.94(2)
thyroid	Over	0.99(3)	**1.00(1.5)**	0.97(4)	***1.00(1.5)***	Over	**1.00(2.5)**	**1.00(2.5)**	**1.00(2.5)**	**1.00(2.5)**
	-	3.38	1.91	3.41	**1.30**	-	3.24	1.59	3.53	**1.65**

Table 6. Area under curve on the variants of Decision Tree and Random Forest Algorithms.

Dataset	Tree Classifier					Random Forest Classifier				
	Sampling Method	AUC				Sampling Method	AUC			
		ST	STS	TTP	TTPS		RF	RFS	RFTP	RFTPS
Diabetes	Mwmote	0.78(3)	0.79(2)	0.75(4)	**0.82(1)**	Rwo	0.89(3)	**0.94(1)**	0.86(4)	0.92(2)
Ecoli	Mwmote	0.70(3)	0.98(2)	0.69(4)	**0.99(1)**	Mwmote	0.80(4)	0.99(3)	**1.00(1.5)**	**1.00(1.5)**
Haberman	Rwo	0.59(4)	0.73(2)	0.71(3)	**0.74(1)**	Mwmote	0.85(4)	0.87(2)	0.80(3)	**0.90(1)**
Parkinsons	Over	0.84(4)	0.94(2.5)	0.94(2.5)	**0.95(1)**	Over	0.93(3)	0.97(2)	0.92(4)	**0.99(1)**
Vertebral	Rwo	**0.93(1)**	0.88(3)	0.84(4)	0.91(2)	Rwo	0.91(3.5)	0.96(2)	0.91(3.5)	**0.98(1)**
Liver	Over	0.79(3)	**0.83(1.5)**	0.68(4)	**0.83(1.5)**	Over	0.88(3)	0.92(2)	0.77(4)	**0.93(1)**
WPBC	Rwo	0.70(4)	**0.90(1.5)**	0.79(3)	**0.90(1.5)**	Rwo	0.83(4)	**0.99(1)**	0.85(3)	0.95(2)
Yeast5	Mwmote	0.93(3)	**0.99(1.5)**	0.86(4)	**0.99(1.5)**	Rwo	0.97(3)	**1.00(4)**	0.90(4)	**1.00(4)**
Abalone	Over	0.68(3)	0.93(2)	0.60(4)	**1.00(1)**	Over	0.70(3.5)	**1.00(1.5)**	0.70(3.5)	**1.00(1.5)**
Cervical C	Over	0.89(3)	0.94(2)	0.80(4)	**0.96(1)**	Both	0.88(3)	**0.99(1.5)**	0.77(4)	**0.99(1.5)**
Diabetic_m	Both	0.98(4)	**0.99(2)**	**0.99(2)**	**0.99(2)**	Over	0.99(3.5)	**1.00(1.5)**	0.99(3.5)	**1.00(1.5)**
Hepatitis	Both	0.79(4)	**0.97(1)**	0.89(3)	0.96(2)	Both	0.59(4)	**0.98(1)**	0.74(3)	0.97(2)
Saheart	Over	0.74(4)	0.80(2)	0.75(3)	**0.81(1)**	Over	0.85(2.5)	**0.90(1)**	0.79(4)	0.85(2.5)
Sick	Over	0.61(3)	0.83(2)	0.60(4)	**0.95(1)**	Over	0.72(3)	**0.96(1)**	0.62(4)	0.92(2)
Spheart	Rose	0.71(4)	0.82(2)	0.75(3)	**0.85(1)**	Rose	0.81(3)	**0.93(1.5)**	0.76(4)	**0.93(1.5)**
Thoracic	Over	0.56(4)	**0.87(1.5)**	0.68(3)	**0.87(1.5)**	Both	0.78(3)	**0.97(1)**	0.77(4)	0.95(2)
thyroid	Over	0.99(3)	**1.00(1.5)**	0.97(4)	**1.00(1.5)**	Over	**1.00(2.5)**	**1.00(2.5)**	**1.00(2.5)**	**1.00(2.5)**
	-	3.35	1.88	3.44	**1.32**	-	3.27	1.74	3.50	**1.79**

Next, we apply statistical tests to validate if the observations made from tables 4, 5, and 6 are statistically significant. The statistical tests rule out the chance of drawing a conclusion based on random factors. We test the statistical significance of the results by applying Friedman's Rank Sum test and Nemeny's test. Friedman's Rank Sum test is a well-known statistical test for comparing multiple algorithms on multiple datasets [52]. The NULL hypothesis for Friedman's Rank Sum test is that none of the algorithm's performance is significantly superior to the others. A p-value lower than 0.05 leads to the rejection of the NULL hypothesis. The Nemeny's test is used to make a pairwise comparison of the algorithms once the result for Friedman's Rank Sum test is affirmative.

Table 7. Friedman test results on three metrics.

Classifiers	Accuracy	Geometric Mean	Area Under Curve
Tree	Friedman chi-squared = 9.906 (16.926) p-value = 0.01938 (0.00073)	Friedman chi-squared = 36.5 p-value = 5.87e-08	Friedman chi-squared = 36.619 p-value = 5.54e-08
Random Forest	Friedman chi-squared = 22.523 p-value =5.08E-05	Friedman chi-squared = 36.765 p-value = 5.159e-08	Friedman chi-squared = 30.286 p-value = 1.202e-06

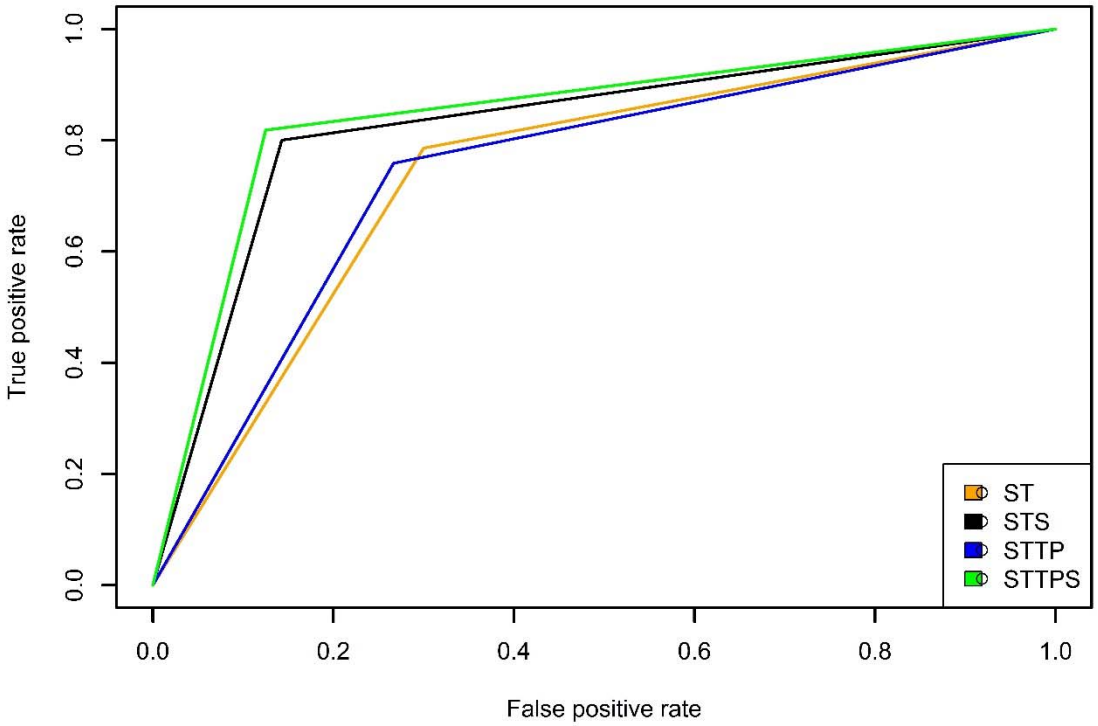
Table 8. Pair-wise comparison of the performance of decision tree classifiers.

Eval. Metric	Algorithm		Nemeny's Test P-values			Conclusions (Significance level=0.05)
Accuracy	Decision Tree		ST	STS	TTP	STS is significantly better than ST TTPS is significantly better than ST and TTP
		STS	**0.067**	-	-	
		TTP	0.79	0.42	-	
		TTPS	**0.024**	0.71	**0.0472**	
	Random Forest		RF	RFS	RFTP	RFS is significantly better than RFTP *RFTPS is significantly better than RF (significance level=0.1). RFTPS is significantly better than RFTP.
		RFS	0.2170	-	-	
		RFTP	0.5447	**0.0062**	-	
		RFTPS	0.0924	0.9786	**0.0015**	
Geometric Mean	Decision Tree		ST	STS	TTP	STS is significantly better than ST. STS is significantly better than TTP. TTPS is significantly better than ST and TTP.
		STS	**0.005**	-	-	
		TTP	0.9999	**0.0039**	-	
		TTPS	**1.40E-05**	0.5025	**1.00E-05**	
	Random Forest		RF	RFS	RFTP	RFS is significantly better than RF. RFS is significantly better than RFTP. RFTPS is significantly better than RF and RFTP.
		RFS	**0.00114**	-	-	
		RFTP	0.91054	**6.9e-05**	-	
		RFTPS	**0.00190**	0.99916	**0.00013**	
Area under curve	Decision Tree		ST	STS	TTP	STS is significantly better than ST. STS is significantly better than TTP. TTPS is significantly better than ST and TTP.
		STS	**0.005**	-	-	
		TTP	0.9972	**0.0024**	-	
		TTPS	**2.70E-05**	0.5872	**1.00E-05**	
	Random forest		RF	RFS	RFTP	RFS is significantly better than RF. RFS is significantly better than RFTP. RFTPS is significantly better than RF and RFTP.
		RFS	**0.00497**	-	-	
		RFTP	0.91054	**0.00039**	-	
		RFTPS	**0.00781**	0.99916	**0.00068**	

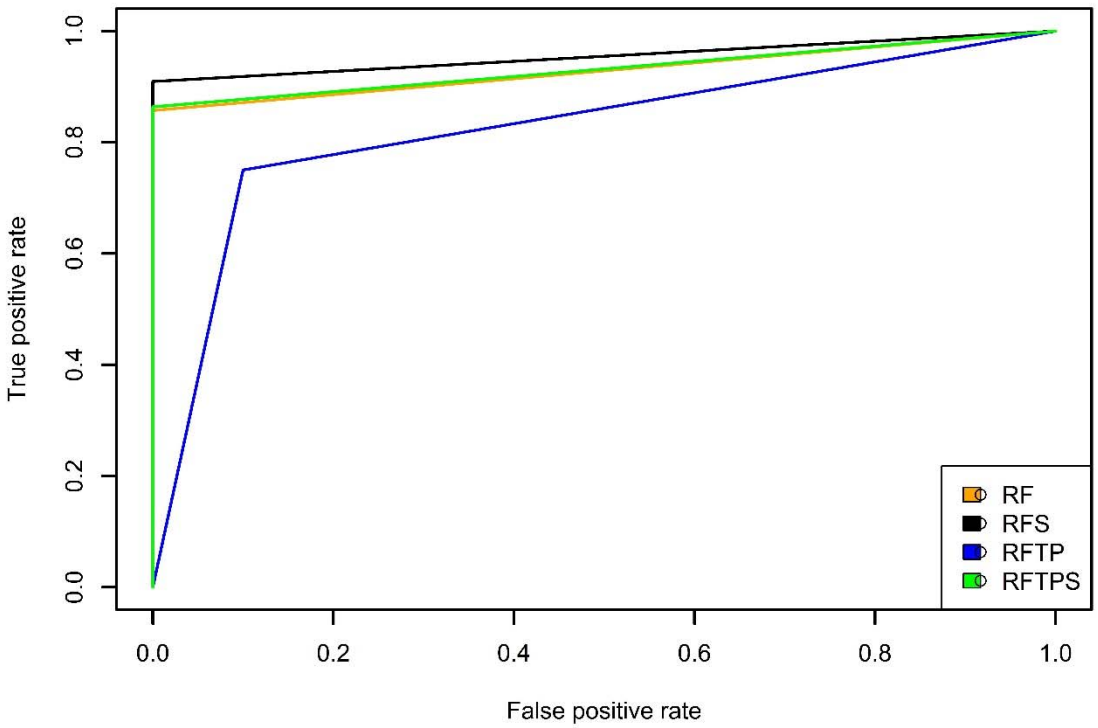
Table 7 summarizes the Friedman rank sum test results for the decision tree and random forest classifiers for accuracy, G-mean and AUC. Since all the resulting p-values are extremely smaller than 0.05. Furthermore, for pairwise, it is safe to reject the NULL hypothesis for both the classifiers for all the performance metrics. Comparison of all classifiers, we apply Nemeny's test. The test results are given in Table 8. The conclusions drawn on the basis of the resulting P-values are recapped in the last column of the table.

The results of the statistical tests corroborate that the CART and RF classifiers with the MOHPT approach are statistically superior methods for disease diagnosis. Since the disease diagnosis datasets have a skewed class ratio, applying classifiers with sampling techniques improves the efficacy of diagnosis significantly. The two sample representative ROC graphs for different versions of decision trees and random forest classifiers are shown in Fig. 2. Table 9 compares the MOHPT versions of CART and RF classifiers. The performance of RF dominates the CART in accuracy, G-mean and AUC across almost all the datasets for better visualization. The sample ROCs for various experimental combinations are shown in Fig. 2.

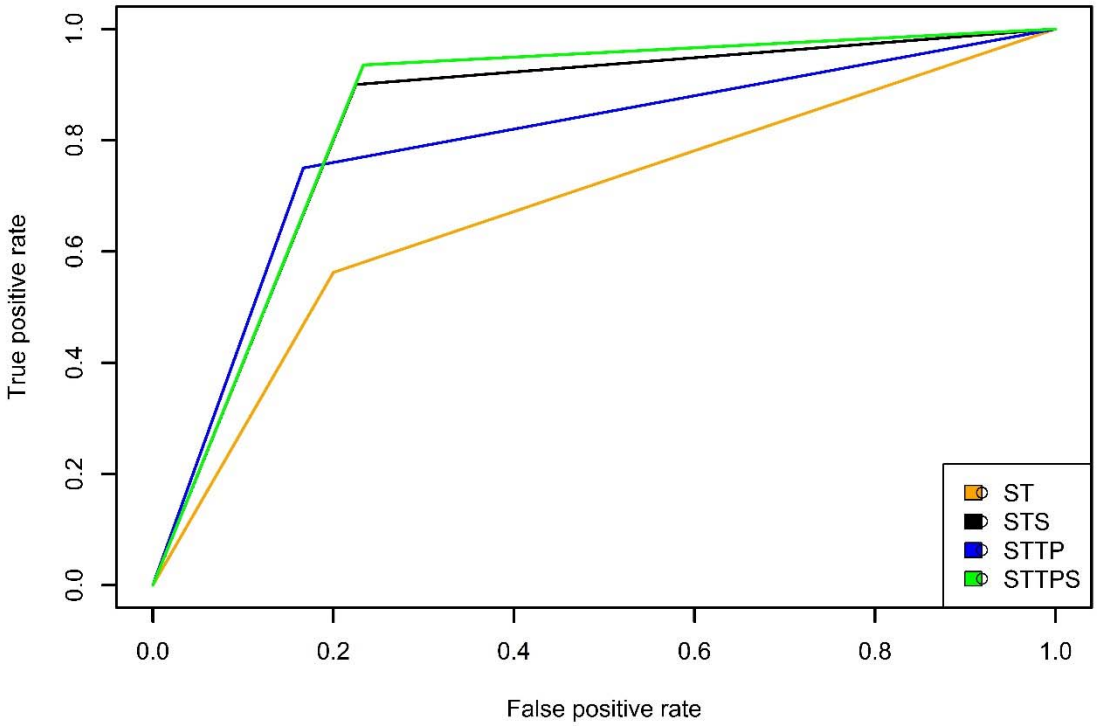
(a)


True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
ST
STS
STTP
STTPS

(b)


True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
RF
RFS
RFTP
RFTPS

(c)



(d)

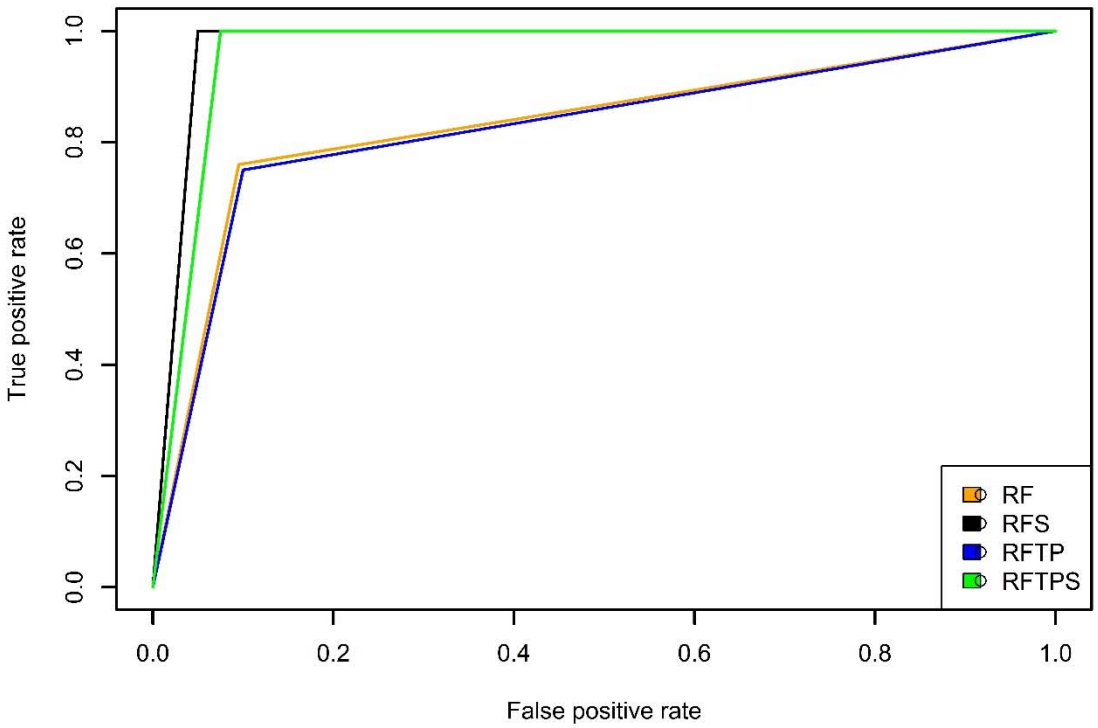


Fig. 2. Sample ROCs: (a) Decision tree classifier on Saheart dataset. (b) Random Forest classifier on Saheart dataset. (c) Decision tree classifier on Thoracic dataset. (d) Random Forest classifier on Thoracic dataset.

Table 9. Comparison of decision tree and random forest MOHPT classifiers.

Dataset	Accuracy		Gmean		AUC	
	TTPS	RFTPS	TTPS	RFTPS	TTPS	RFTPS
Diabetes	0.82	**0.92**	0.82	**0.92**	0.82	**0.92**
Ecoli	0.99	**1.00**	0.99	**1.00**	0.99	**1.00**
Haberman	0.76	**0.90**	0.75	**0.90**	0.74	**0.90**
Parkinsons	0.95	**0.99**	0.95	**0.99**	0.95	**0.99**
Vertebral	0.91	**0.98**	0.92	**0.98**	0.91	**0.98**
Liver	0.82	**0.93**	0.83	**0.92**	0.83	**0.93**
WPBC	0.90	**0.95**	0.90	**0.95**	0.90	**0.95**
Yeast5	0.99	**1.00**	0.99	**1.00**	0.99	**1.00**
Abalone_19	**1.00**	**1.00**	**0.99**	**0.99**	**1.00**	**1.00**
Cervical_cancer	0.96	**0.99**	0.96	**0.99**	0.96	**0.99**
Diabetic_m	0.99	**1.00**	0.99	**1.00**	0.99	**1.00**
Hepatitis	0.96	**0.98**	0.96	**0.97**	0.96	**0.97**
Saheart	0.81	**0.85**	0.81	**0.85**	0.81	**0.85**
Sick	**0.95**	0.92	**0.94**	0.92	**0.95**	0.92
Spectheart	0.84	**0.93**	0.84	**0.93**	0.85	**0.93**
Thoracic	0.87	**0.95**	0.87	**0.94**	0.87	**0.95**
Thyroid	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**
	3	16	2	16	3	16

6. Comparative Analysis

Here, we compare the MOHPT with two related research works that adopt multi-objective framework. The first one is [26], and the second is [49]. The first paper presents a multi-objective evolutionary algorithm (MOEA) to automatically construct decision tree classifiers (MOHEAD-DT). In this work, the authors have adopted two methods, Pareto Dominance and Lexicographic Analysis, for producing non-dominated pareto optimal solutions. The two versions of MOHEAD-DT are named MOHEAD-L and MOHEAD-P, respectively. This research is not specifically dedicated to disease diagnosis. We could only find three datasets common between our work and Basgalupp et al. [26]. Moreover, the latter have used F- measure for estimating the performance of their system. The comparison based on the three datasets is shown below.

Table 10. Comparison of the proposed MOHPT method with [53] based on F-measure.

Datasets	MOHEAD-L	MOHEAD-P	TTPS	RFTPS
Abalone	0.23	0.23	99	1.0
Hepatitis	0.75	0.78	98	99
Sick	0.98	0.98	97	98

For the three datasets, MOHPT outperforms both MOHEAD-L and MOHEAD-P versions. Second, we compare the MOHPT-DT method with the recent work by Rao et al. (2018). This research tunes cost, tolerance, gamma and epsilon parameters of sequential mining optimization classifier (SMO), an alternative of the classical SVM algorithm. In this paper, the optimization of the hyperparameters is carried out by applying three popular evolutionary algorithms, namely, Elephant Heard Optimization (EHO), Multi-objective evolutionary algorithm based on decomposition (MOEA/D) and non-dominated sorting algorithm (NSGA-II). The authors also implement a hybridized version on EHO, called CEHO, in which they employ a novel cuboid based initial population method for enhancing the performance of EHO. The four algorithms are tested on 17 medical datasets using tenfold cross validation. The authors have made extensive comparisons to the existing related works and they have concluded that the CEHO performs statistically better than the other comparable counterparts. Table 11 gives the comparison of TTPS, RFTPS and CEHO on the seven datasets that are found common in the two research works. It is clear from the table that the TTPS and RFTPS dominate CEHO over all the seven datasets regarding accuracy as well as G-mean. In this scenario, there is no need to apply any statistical tests.

Table 11. Comparison of random forest MOHPT classifier with CEHO.

Dataset	Accuracy			G-mean		
	TTPS	RFTPS	CEHO	TTPS	RFTPS	CEHO
Haberman	0.76	**0.90**	79.03	0.75	**0.90**	0.61
Liver	0.82	**0.93**	74.44	0.83	**0.92**	0.76
WPBC	0.90	**0.95**	86.52	0.90	**0.95**	0.83
Hepatitis	0.96	**0.98**	89.68	0.96	**0.97**	0.86
Spectheart	0.84	**0.93**	84.64	0.84	**0.93**	0.89
Thoracic	0.87	**0.95**	85.53	0.87	**0.94**	0.85
Thyroid	**1.00**	**1.00**	97.67	0.99	**1.00**	0.97

Although the two comparisons made here are on a smaller number of datasets, these are positive indicators of the competitive performance of MOHPT.

7. Conclusion and Future Work

This research presents a disease diagnosis system (MOHPT) that uses CART and Random Forest classifiers. We have addressed two challenging issues pertaining to disease diagnosis: i) tackling the class imbalance in the disease diagnosis datasets and ii) setting the optimal values of hyperparameters of the classifiers. The main contribution of the research comes from optimizing the combination of the hyperparameter configuration of the classifiers and sampling technique for creating decision tree-based classifiers for disease diagnosis. We have devised MOHPT for obtaining the best non-dominated Pareto-optimal hyperparameter configurations. Further, a single resampling technique may not work equally well across different disease data domains. Hence, the best suited sampling technique is determined individually for each of the disease diagnosis datasets. The MOHPT method makes available several hyperparameter configurations, each one corresponding to a different level of trade-off between sensitivity and specificity. Although a medical expert has the freedom to select the hyperparameter configuration that suits any prevailing specific requirements, we envisage the use of hyperparameter configuration parallel to the best G-mean. The suggested disease diagnosis system is able to reduce the false negative and false positive inferences and hence enhance the efficacy of disease diagnosis. The performance of the suggested method was found to be significantly better compared to the other related approaches. In the future, we will apply the suggested approach to Covid-19 data. We also intend to use deep learning strategies from a multi-objective perspective.

Acknowledgments

I would like to express gratitude to Prof. Saroj, my supervisor, for his numerous contributions to this study.

References

[1] I. Kononenko, “Machine learning for medical diagnosis: history, state of the art and perspective,” *Artificial Intelligence in Medicine*, vol. 23, no. 1, pp. 89–109, Aug. 2001, doi: 10.1016/S0933-3657(01)00077-X.

[2] N. Esfandiari, M. R. Babavalian, A.-M. E. Moghadam, and V. K. Tabar, “Knowledge discovery in medicine: Current issue and future trend,” *Expert Systems with Applications*, vol. 41, no. 9, Art. no. 9, Jul. 2014, doi: 10.1016/j.eswa.2014.01.011.

[3] K. Chu, “An introduction to sensitivity, specificity, predictive values and likelihood ratios,” *Emergency Medicine*, vol. 11, no. 3, pp. 175–181, 1999.

[4] A. Jain, S. Ratnoo, and D. Kumar, “Addressing class imbalance problem in medical diagnosis: A genetic algorithm approach,” in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, Aug. 2017, pp. 1–8. doi: 10.1109/ICOMICON.2017.8279150.

[5] A. Jain, S. Ratnoo, and D. Kumar, “Performance comparison of classification algorithms for medical diagnosis,” *Pertanika Journal of Science and Technology*, vol. 26, pp. 729–748, Apr. 2018.

[6] B. V. Ramana and R. S. Kumar Boddu, “Performance Comparison of Classification Algorithms on Medical Datasets,” in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Jan. 2019, pp. 0140–0145. doi: 10.1109/CCWC.2019.8666497.

[7] S. K. Jha, Z. Pan, E. Elahi, and N. Patel, “A comprehensive search for expert classification methods in disease diagnosis and prediction,” *Expert Systems*, vol. 36, no. 1, p. e12343, 2019.

[8] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, “Comparing different supervised machine learning algorithms for disease prediction,” *Medical Informatics and Decision Making*, vol. 19, no. 1, pp. 1–16, Dec. 2019.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[10] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning,” in *Advances in Intelligent Computing*, Berlin, Heidelberg, 2005, pp. 878–887. doi: 10.1007/11538059_91.

[11] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, Jun. 2008, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.

[12] S. Barua, Md. M. Islam, X. Yao, and K. Murase, “MWMOTE--Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2014, doi: 10.1109/TKDE.2012.232.

[13] H. Zhang and M. Li, "RWO-Sampling: A random walk over-sampling approach to imbalanced data classification," *Information Fusion*, vol. 20, pp. 99–116, Nov. 2014, doi: 10.1016/j.inffus.2013.12.003.
[14] P. P. Ippolito, "Hyperparameters Optimization," *Medium*, Sep. 26, 2019. https://towardsdatascience.com/hyperparameters-optimization-526348bb8e2d (accessed Jul. 06, 2020).
[15] P. Probst, M. N. Wright, and A.-L. Boulesteix, "Hyperparameters and tuning strategies for random forest," *WIREs Data Mining and Knowledge Discovery*, vol. 9, no. 3, p. e1301, 2019, doi: 10.1002/widm.1301.
[16] W. Alawad, M. Zohdy, and D. Debnath, "Tuning Hyperparameters of Decision Tree Classifiers Using Computationally Efficient Schemes," in *2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, Sep. 2018, pp. 168–169. doi: 10.1109/AIKE.2018.00038.
[17] J. Pan, P. Wei, Q. Guo, C. Zhang, and A.-L. Luo, "GA-optimized random forest classification for high dimensional data," *ICIC Express Letters*, vol. 5, pp. 1529–1534, May 2011.
[18] R. C. Barros, M. P. Basgalupp, A. C. P. L. F. de Carvalho, and A. A. Freitas, "A hyper-heuristic evolutionary algorithm for automatically designing decision-tree algorithms," in *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference - GECCO '12*, Philadelphia, Pennsylvania, USA, 2012, pp. 1237–1244. doi: 10.1145/2330163.2330335.
[19] J. Ahuja and S. Ratnoo, "Optimizing Feature Subset and Parameters for Support Vector Machine Using Multiobjective Genetic Algorithm," *Journal of Intelligent Systems*, vol. 0, Jan. 2014, doi: 10.1515/jisys-2014-0107.
[20] M. Camilleri and F. Neri, "Parameter Optimization in Decision Tree Learning by using Simple Genetic Algorithms," vol. 13, p. 10, 2014.
[21] R. G. Mantovani, T. Horvath, R. Cerri, J. Vanschoren, and A. C. P. L. F. de Carvalho, "Hyper-Parameter Tuning of a Decision Tree Induction Algorithm," in *2016 5th Brazilian Conference on Intelligent Systems (BRACIS)*, Recife, Oct. 2016, pp. 37–42. doi: 10.1109/BRACIS.2016.018.
[22] S. George C and B. S. -, "Grid Search Tuning of Hyperparameters in Random Forest Classifier for Customer Feedback Sentiment Prediction," *IJACSA*, vol. 11, no. 9, 2020, doi: 10.14569/IJACSA.2020.0110920.
[23] M. T. M. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," *Nat Comput*, vol. 17, no. 3, pp. 585–609, Sep. 2018, doi: 10.1007/s11047-018-9685-y.
[24] C. Igel, "Multi-objective Model Selection for Support Vector Machines," in *Evolutionary Multi-Criterion Optimization*, Berlin, Heidelberg, 2005, pp. 534–546. doi: 10.1007/978-3-540-31880-4_37.
[25] H. Zhao, "A multi-objective genetic programming approach to developing Pareto optimal decision trees," *Decision Support Systems*, vol. 43, no. 3, pp. 809–826, Apr. 2007, doi: 10.1016/j.dss.2006.12.011.
[26] M. P. Basgalupp, R. C. Barros, and V. Podgorelec, "Evolving decision-tree induction algorithms with a multi-objective hyper-heuristic," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing - SAC '15*, Salamanca, Spain, 2015, pp. 110–117. doi: 10.1145/2695664.2695828.
[27] A. Jain, S. Ratnoo, and D. Kumar, "A novel multi-objective genetic algorithm approach to address class imbalance for disease diagnosis," *Int. j. inf. tecnol.*, May 2020, doi: 10.1007/s41870-020-00471-3.
[28] R. Schmucker, M. Donini, V. Perrone, M. B. Zafar, and C. Archambeau, "Multi-Objective Multi-Fidelity Hyperparameter Optimization with Application to Fairness," p. 14, 2020.
[29] R. Bellazzi and B. Zupan, "Predictive data mining in clinical medicine: current issues and guidelines," *Int J Med Inform*, vol. 77, no. 2, Art. no. 2, Feb. 2008, doi: 10.1016/j.ijmedinf.2006.11.006.
[30] R. D. Sah and Dr. J. Sheetalani, "Review of Medical Disease Symptoms Prediction Using Data Mining Technique," *IOSR Journal of Computer Engineering*, vol. 19, no. 03, pp. 59–70, May 2017.
[31] V. Podgorelec, P. Kokol, B. Stiglic, and I. Rozman, "Decision Trees: An Overview and Their Use in Medicine," *Journal of Medical Systems*, vol. 26, no. 5, pp. 445–463, Oct. 2002.
[32] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.
[33] Md. Z. Alam, M. S. Rahman, and M. S. Rahman, "A Random Forest based predictor for medical data classification using feature ranking," *Informatics in Medicine Unlocked*, vol. 15, p. 100180, Jan. 2019.
[34] A. Subasi, E. Alickovic, and J. Kevric, "Diagnosis of Chronic Kidney Disease by Using Random Forest," in *CMBEBIH 2017*, Singapore, 2017, pp. 589–594. doi: 10.1007/978-981-10-4166-2_89.
[35] M. Khalilia, S. Chakraborty, and M. Popescu, "Predicting disease risks from highly imbalanced data using random forest," *BMC Med. Inf. & Decision Making*, 2011, doi: 10.1186/1472-6947-11-51.
[36] A. Ozcift, "Random forests ensemble classifier trained with data resampling strategy to improve cardiac arrhythmia diagnosis," *Comput. Biol. Med.*, vol. 41, no. 5, Art. no. 5, May 2011, doi: 10.1016/j.compbiomed.2011.03.001.
[37] D. J. Dittman, T. M. Khoshgoftaar, R. Wald, and A. Napolitano, "Comparison of Data Sampling Approaches for Imbalanced Bioinformatics Data," p. 4, 2014.
[38] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, pp. 429–449, 2002.
[39] T. Khoshgoftaar, M. Golawala, and J. Van Hulse, "An Empirical Study of Learning from Imbalanced Data Using Random Forest," Oct. 2007, vol. 2, pp. 310–317. doi: 10.1109/ICTAI.2007.46.
[40] L. Yang, Y. Guo, and J. Cheng, "Manifold Distance-Based Over-Sampling Technique for Class Imbalance Learning," *AAAI*, vol. 33, pp. 10071–10072, Jul. 2019, doi: 10.1609/aaai.v33i01.330110071.
[41] A. Jain, S. Ratnoo, and D. Kumar, "Addressing class imbalance problem in medical diagnosis: A genetic algorithm approach," in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, Aug. 2017, pp. 1–8. doi: 10.1109/ICOMICON.2017.8279150.
[42] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE–Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 2, pp. 405–425, Feb. 2014, doi: 10.1109/TKDE.2012.232.
[43] B. F. F. Huang and P. C. Boutros, "The parameter sensitivity of random forests," *BMC Bioinformatics*, vol. 17, no. 1, p. 331, Sep. 2016, doi: 10.1186/s12859-016-1228-x.
[44] M. Camilleri and F. Neri, "Parameter Optimization in Decision Tree Learning by using Simple Genetic Algorithms," vol. 13, p. 10, 2014.
[45] J. Pan, P. Wei, Q. Guo, C. Zhang, and A.-L. Luo, "GA-optimized random forest classification for high dimensional data," *ICIC Express Letters*, vol. 5, pp. 1529–1534, May 2011.
[46] C.-L. Huang and J.-F. Dun, "A distributed PSO–SVM hybrid system with feature selection and parameter optimization," *Applied Soft Computing*, vol. 8, no. 4, pp. 1381–1391, Sep. 2008.
[47] S.-W. Lin, T.-Y. Tseng, S.-C. Chen, and J.-F. Huang, "A SA-Based Feature Selection and Parameter Optimization Approach for Support Vector Machine," in *IEEE International Conference on Systems, Man and Cybernetics*, Oct. 2006, vol. 4, pp. 3144–3145.

[48] M. R. Nalluri, K. K., M. M., and D. S. Roy, "Hybrid Disease Diagnosis Using Multiobjective Optimization with Evolutionary Parameter Optimization," *Journal of Healthcare Engineering*, vol. 2017, pp. 1–27, 2017, doi: 10.1155/2017/5907264.
[49] N. MadhuSudana Rao, K. Kannan, X. Gao, and D. S. Roy, "Novel classifiers for intelligent disease diagnosis with multi-objective parameter evolution," *Computers & Electrical Engineering*, vol. 67, pp. 483–496, Apr. 2018, doi: 10.1016/j.compeleceng.2018.01.039.
[50] C. Müssel, L. Lausser, M. Maucher, and H. A. Kestler, "Multi-Objective Parameter Selection for Classifiers," *J. Stat. Soft.*, vol. 46, no. 5, 2012, doi: 10.18637/jss.v046.i05.
[51] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002, doi: 10.1109/4235.996017.
[52] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.
[53] M. P. Basgalupp, R. C. Barros, and V. Podgorelec, "Evolving decision-tree induction algorithms with a multi-objective hyper-heuristic," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing - SAC '15*, Salamanca, Spain, 2015, pp. 110–117. doi: 10.1145/2695664.2695828.

Author's Profile

Sunil Kumar is working as an Assistant Professor at the Department of Computer Science & Engineering (CSE) in Guru Jambheshwar University of Science and Technology (GJUS&T), Hisar, India. He is pursuing a Ph.D. in Computer Science and Engineering from the Department of CSE, GJUS&T, Hisar. Data Mining Machine Learning, and Soft Computing are research areas. He has published five papers.

Saroj Ratnoo received M.Sc. in Computing Science from Birkbeck College, University of London, UK, in 1994. She joined the Department of Computer Science and Engineering (GJUS&T), Hisar, in 1996 as an Assistant Professor. She completed her Doctorate from the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India, in 2010. She continues to work in the Department of Computer Science and Engineering (GJUS&T), Hisar, India, as a Professor. Her research interests include Nature-Inspired Algorithms, Data Mining and Machine Learning. She has published many papers.