

Enhancing Random Forest Performance through Optimal Instance Subset Selection using Genetic Algorithm

Umair Rashid^{*1}, Asad Abbas², Malik Muhammad Aqib³, Syed Haider Shah⁴, Umar Khattab⁵

¹⁻⁴Department of Computer Science, Bahauddin Zakariya University, Multan, Pakistan.

⁵Department of Information Technology, Bahauddin Zakariya University, Multan, Pakistan.

Corresponding Author: Umair Rashid (umairrashid@gmail.com)

ABSTRACT

The amount of high-dimensional data grows at an exponential rate that makes it harder for standard machine learning algorithms to process data quickly and correctly. As datasets get bigger and more complicated, it's important to use good data reduction methods to make models perform better and be more accurate. The optimal subset selection is an effective solution that finds the most representative instances while getting rid of unnecessary or noisy data. The study presents a novel approach to use a Genetic Algorithm (GA) to select the best subsets and improve classification performance. The method not only selects the optimal instances, but it also makes the Random Forest (RF) classifier's hyperparameters work better. The DEAP library, which allows evolutionary computations, is used to carry out the GA-based optimization. The combination of RF model with GA-based optimization gets better classification accuracy of 85.8%, while reducing data redundancy, as shown by experiments done on KDD CUP dataset. The results show that the GA-based instance selection not only makes the RF classifier better at learning, but also makes computing faster by cutting down on duplicate data. It shows how evolutionary algorithms can be used to improve traditional machine learning methods by giving a useful and expandable way to deal with big and complicated dataset.

Keywords: Machine Learning, Instance Selection, Random Forest, Genetic Algorithm, Optimization

1. INTRODUCTION

The data is growing daily, and the handling and processing of such data is becoming a challenge. The huge data has made an appearance that is ignorant to the traditional techniques. To preprocess such data, Instance selection is the best approach. Instance selection is the data preprocessing technique used in machine learning. Instance selection aims to reduce the volume of the original dataset to a manageable length that preserves the quality of information. It helps to discard irrelevant and noisy information to speed up the learning tasks and classification accuracy. Common in real-life situations, data characteristics might contain missing or omitted values and commonly come with several scale levels[1]. Thus, several instance selection algorithms are used to select an optimal subset with zero compromise on the crucial data to achieve the specified performance same as when we consider the complete dataset without any reduction. Sometimes, the machine learning models learn the wrong patterns from the noisy data that make the wrong prediction. Namely, the model that is trained on the misleading data gives inaccurate results. Here, the instance selection makes sense and supplies sophisticated information to the model as training data[2]. Aiming to choose a subset of columns (bands) that best reflect the original data matrix, another class of unsupervised BS approaches based on subset search strategies mainly concentrate on the column subset selection problem (CSSP) for matrices [3]. Some instance selection algorithms assess the maturity of the dataset that helps to reduce the instances at maximum level. While others only consider the inferior vectors. The first instance selection algorithm (CNN) was introduced by Hart to focus on reducing the number of instances to remove errors and to reduce the computational complexity. CNN was not proved as the best instance selection algorithm, but is the base to propose other algorithms[4]. The idea behind CNN was to extract the small consistent data subset. However, the consistent data subset may not be minimal, which is the main disadvantage of the CNN [5]. CNN [4] inspired other instance reduction algorithms such as ENN, RNN, and SNN. The RNN (Reduced Nearest Neighbour) [6] was proposed by Gates after rejection of the first instance selection method (CNN). The concept of RNN is the same as CNN, but it mainly targets those instances in the training dataset that didn't affect the accuracy of the dataset. Furthermore, ENN by Dennis L. Wilson [7] has gained popularity that aims to extract the instances where the classes of these instances have no direct correlation to classes of their neighbors. Over the time, some variations have come to improve the performance of CNN. The study [8] has level-up the instance selection algorithm after adding the ranking factor, and also initiated to introduce the ranking of the instances of the training dataset for each class, and using ranking, the algorithm takes decision to select the instances. Thus [9] introduced the algorithm that calculates the scores from instances' ranking based on [8]. These all algorithms have the support for the K Nearest Neighbor. Although, we have some other learning algorithms. When SVM came

and started to be used for the machine learning classification tasks, then at that time,[10] interacted with the new approach that is able to regenerate the training data for the Support Vector Machine. With some variations mentioned in [11], the genetic algorithm was in the game to extract the optimal subset for the training data. Meanwhile, [12] considered the LSH (Locality-Sensitive Hashing) to propose the new instance selection algorithm to train the Support Vector Machine. The algorithm is also called DR.LSH, used to identify and remove the redundant information. Later, to speed up the support vector machine, [13] adopted the new approach (clustering) to enhance the process of instance selection through extracting the instances from the clusters using two main factors such as detection of boundary and the boundary of the instance selection.

The discussed instance selection algorithms are effective for the standard size dataset such as small and medium length of data. When we go forward towards big data, the above algorithms didn't response because, to extract the instances, we need to first load the complete dataset in the main memory. However, for the big data, we should have an ample memory space. The memory is the main reason for the infeasibility of the discussed algorithms in [14] and introduced the most valuable variation or extension to the instance selection algorithm with the functionality of MapReduce and Democratic Distance. Moreover, [15] tested the prototype algorithms that gave zero result for the big data. Afterward, they introduced the instances reduction algorithm based on MapReduce. Next, to power-up the classification and reduction accuracy, [16] adopted the MapReduce framework for RMHC (algorithm). Influenced by the Mutation and Crossover techniques along with the concept of optimization, we based our new technique to extract optimal subset from the dataset. For this case, we have used Genetic Algorithm and Random Forest. Our technique works to calculate the entropies with the support of the several classification algorithms, and to assess the optimality of the instances' subset. In addition, the implementation of GA on Random Forest Algorithm illustrates the new beginning of accuracy and performance.

2. LITERATURE REVIEW

Huge data requires large memory for the processing. Different researchers consider the instance selection for small dataset but not for the big data. Of course, handling the big data is a challenge, we need to extract the meaningful information and to reduce the computational complexity. However, some researchers have given some of their contributions for the optimal instance selection. The complete description is in[15] that adopted several factors with the aim to improve the classification accuracy of the big data. They implemented the several algorithmic techniques into one approach to divide the functionality using clusters. Nevertheless, the single prototype reduction method was not enough. Therefore, they added MapReduce to support the prototype reduction algorithm. Additionally, in [17] highlighted the same issues that now the machine learning algorithms individually cannot handle the data. Therefore, they decided to use the two algorithms based on the LSH (Locality Sensitive Hashing) to check the similarities and correlation between the instances of the training dataset. The proposed approach outperformed the conventional technique that scored quadratic time, large computational time. Furthermore, to get remarkable sureties, they cut-off the computational complexity with the bulk (millions) of data instances. In the study [18], methods for feature selection and removal in best subset selection are enhanced with simpler computation. Our suggested technique improves the accuracy of predictions without adding more computational burden. The approach offers a combination of analysis and practical results on many data sets. Application of these methods to Matching Pursuit and ABESS improves their results in sparse regression and compressed sensing. GAIS is introduced, a way to use Graph Attention Networks (GATs) to discover informative data instances [19]. GAIS represents data on graphs and adds attention to notice the connections among different instances. On the 13 datasets used in the experiment, GAIS manages to remove over 96 percent of data with no drop or better than traditional model performance. Various ways of reducing graphs for choosing the best instances are compared in the study[20]. The methods assess performance in different ways by considering accuracy, redundancy and model complexity. The results revealed that using graphs can greatly reduce the amount of data needed, without losing performance. This study [21] introduces an optimal method for selecting and including features into logistic regression by using mixed-integer conic optimization. Because the method includes acquiring features, it is valuable for clinical uses where costs matter a lot. Through looking at generated synthetic data and carefully evaluating approaches, the study shows the weaknesses of today's methods in situations with little data and noisy labels, suggesting new methods for upcoming work. A suboptimal method for selecting the best subset is examined in [22], and also proposes a fast algorithm for selecting the best variables from a large set that gives acceptable answers using less effort. Results from straight experiments on multiple synthetic and real data sets demonstrate that the suggested method equals existing suboptimal approaches and acts as a good balance between effectiveness and efficiency. The research study [23] presents the EFG-m method, a reliable and effective way to choose optimal subsets in large sample. This method reduces the number of samples needed, still providing reliable

results, so it is well suited to uses where a limited or expensive data collection is necessary. The effectiveness and efficiency of EFG-m are confirmed by both theory and practical experiments.

The approach in [24] presents the problem of selecting the best subset in linear regression as a continuous optimization task to provide a more manageable convex version of discrete models. With this method, it is possible to do efficient calculations for thousands of features. Components of Machine Based Scientific Studies demonstrate that COMBSS is both more accurate and faster at computing than Lasso and similar feature selectors. The study [25] scaled up the instance selection algorithm to extract the instances from the large dataset. The idea is to keep computations light and the data scalable, so that the final dataset promises results similar to those using the full one for classification. From the experiments, we find that LSIS can substantially reduce the size of data without reducing accuracy, so it is a useful method for preprocessing big data. Then, the MapReduction functionality came into notice, and the researchers added it to improve the performance of the instance selection method [15]. Three instance selection techniques that rely on Locality Sensitive Hashing (LSH) for clustering are suggested in this study[26]. These methods which are implemented in Apache Spark, can process large datasets that are not evenly balanced. The findings demonstrate that LSH-based instance selection can increase the geometric mean performance measurement by 5% to 19%. LSH considered the effective approach at that time. Inspired by LHS, [27] again targeted the Locality Sensitive Hashing, but they also included the Double Voting Mechanism for the instance selection of the large dataset. It was an iterative approach that completed the k round to remove the biasness of the optimal subset of instances.

In this study, a thorough review of the existing literature has been carried out and identified that Genetic Algorithms have been widely used for features selection and fine-tuning parameters. However limited literature has delved into the integration with instance subset selection to make the RF classifier performance better. Notably, no prior work has utilized GA and RF together in this way to improve both the quality of the data and the model parameters simultaneously. This study bridges that gap by utilizing a Genetic Algorithm to find the training cases that give the most information and increase the Random Forest model's ability to classify things at the same time.

3. METHODS

The method is based on the implementation to get optimal subset instances selection using genetic algorithm. As, we provided the reduced dataset as an input to the Random Forest algorithm to classify the labels. The genetic algorithm has optimized the Random Forest with guaranteed improved accuracy. The research design starts with some pre-processing techniques to clean and normalize the data. Afterward, the process begins to extract the useless instances from the dataset to preserve the quality of data. In the second process, the Random Forest algorithm classifies to check the best fit.

3.1 Data Preprocessing

The research study performs the experimentation on KDD dataset which is available at UCI Machine Learning Repository [28]. KDD CUP 1999 [29] has 4000000 instances and 42 attributes. approximately 5 million connection records are included in dataset, each labeled as either normal or as one of 22 different types of attacks which are divided into four categories including DoS, U2L, R2L, and Probe. The data preprocessing methods used in this study include data cleaning and normalization which have already been described in our previous work[30]. It is essential to do data preprocessing to guarantee the accuracy and integrity of the used data. Data quality can be lowered by incomplete, inconsistent, and irrelevant special characters in the gathered data. Since these problems must be addressed in order to improve the overall data quality, substantial data pretreatment is required.

3.2 Random Forest

The foundation of Random Forest is the idea of ensemble learning, which involves training and combining several models to produce predictions. Decision trees serve as the individual models in Random Forest. Each decision tree uses a separate subset of characteristics and is trained on a random portion of the data. The final prediction is then created by combining the predictions of all the trees. Random Forest uses two methods to introduce unpredictability. In the beginning, it builds random subsets of the original dataset using bootstrap sampling (sampling with replacement). This enables variation in the individual models by training each decision tree on a distinct subset of the data. Second, rather than examining all features at each split in the decision tree, only a random sample of features is taken into account. This randomization contributes to the model's increased robustness and decreased overfitting. Once every decision tree has been trained, they all work together to forecast the future using a voting method. The feature

importance is measured via Random Forest. It determines the average impurity reduction brought on by each feature across all of the decision trees, such as the Gini impurity [31].

3.3 Genetic Algorithm

The Genetic Algorithm is an optimization method that draws inspiration from genetics and natural evolution. It is frequently used to find the optimum solution within a broad search space in order to solve complex optimization challenges.

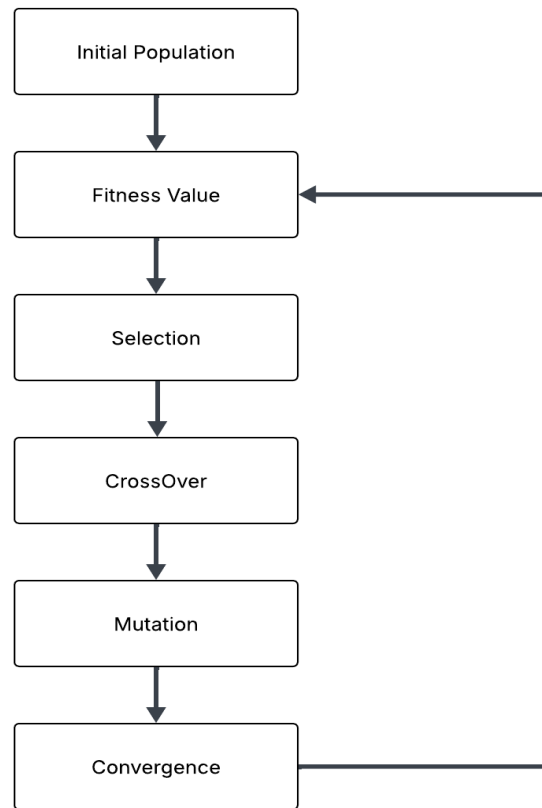


Figure 1. Working Diagram of Genetic Algorithm

The initial population of potential solutions used by the Genetic Algorithm is represented as chromosomes. Genes that encode the variables or parameters of the problem are found on each chromosome. A wide range of potential solutions are represented by the population. Each chromosome's fitness is assessed by measuring how well it uses a fitness function to solve the problem. The optimization problem's objective or goal is represented by the fitness function. Higher fitness values are better and have a better probability of being chosen for the following generation of chromosomes.

The procedure of selecting people from the current population to develop the next generation comes next. Following the "survival of the fittest" tenet, it is more likely that chromosomes with higher fitness values will be chosen. You can use selection methods like roulette wheel selection or tournament selection. A genetic process called crossover combines the genetic material from specific parent chromosomes to produce kids. It mimics how sexual reproduction occurs in nature. Gene exchange and recombination can create novel solutions that may acquire advantageous traits from their parents. Another genetic operator that creates arbitrary changes in the progeny's chromosomes is mutation. It keeps population variety intact and delays the emergence of unsatisfactory solutions. As a result, multiple generations of solutions are produced by iteratively applying the stages of selection, crossover, and mutation. A termination criterion, such as completing a set number of iterations or arriving at a satisfying solution, must be satisfied

for the process to end [32]. The optimal parameter values for each stage of the genetic algorithm during subset selection is outlined in Table1.

Table 1. Optimal values for genetic algorithm		
Step	Parameter	Value
Initial Population	Population Size	50–100
Fitness Value	Fitness Function	Accuracy – $\alpha \times$ Feature Ratio
Selection	Tournament Size	3–5
Crossover	Crossover Rate	0.8 (80%)
Mutation	Mutation Rate	0.01–0.05
Convergence	Max Generations	50–100
	Early Stopping	10–15 generations without progress

System design to select optimal instance subset and random forest optimization using genetic algorithm is depicted in Figure 2.

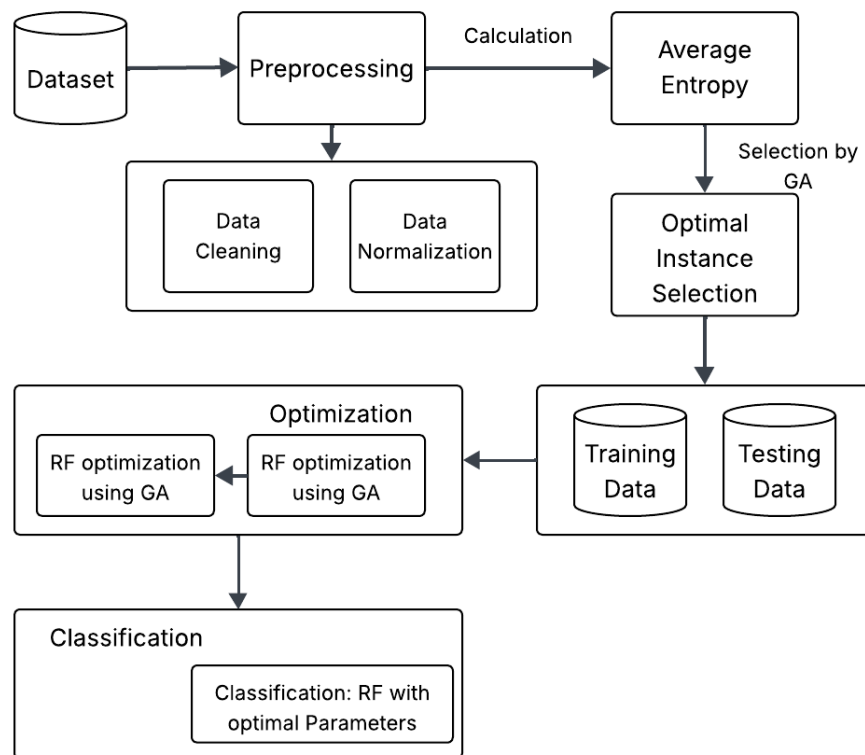


Figure 2. Evaluation Framework Process

3.4 Optimal Instance Subset Selection using Genetic Algorithm

The demonstration is started here with the genetic algorithm to extract the optimal subset of instances from the dataset. The main purpose of instance selection is to generate the quality dataset with useful data parameters. In the dataset, we have hard and soft instances, and the hard instances have the higher chances of misclassification. In this case, hard instances are considered more informative to the classification. On the other hand, the simple instances have the highest probability of correct classification. Therefore, considering hard data for the classification is the best approach because of the effective contribution to data loss. In addition, the falsely classified instances help to render maximum level of information to the classifiers, which work to set the decision boundary. So, the major task is to define the threshold to extract the hard instances, such as the criteria is to select the instances above or below the defined hard threshold value that is the fair representation of informative data instances [33]. The genetic algorithm uses the search strategy that is applied to the population to extract the optimal instance subset. In this case, we have individuals, and these individuals are considered to be the solution to your underlying problem [32]. Firstly, (Individuals) solutions encoding is done. The individual is given that is represented in the binary digit (0 and 1) through encoding process. Afterward, the fitness function is used to check the quality and power of individuals. Fitness Function is the significant measure of how much an instance has the probability of misclassified or incorrectly classified. If the instance has highest probability of incorrectly classified, then this instance is considered to be the hard instance or the important index.

The Genetic Algorithm proceeds the process with manipulation, including selection, mutation, and crossover. Roulette Wheel favors chromosomes (individuals) with the strong fitness value. Roulette Wheel generates the cumulative fitness value by summing up all the fitness values from each individual. Using the cumulative fitness value, the selection probability of each individual is calculated. The roulette wheel is then spun, and random selection is made by generating numbers at random, such as those between 0 and 1. The roulette wheel is circled beginning with the first segment during the selecting process. The selection probabilities of the persons encountered are added up until the total exceeds the number produced at random. The winner is the person whose total exceeds the random number. Next, the crossover and mutation update the population by changing (1 to 0 or to convert the 0 into 1).

3.5 Classification using Random Forest

Random Forest has used reduced dataset with informative instances in this study. In this way, the output of the Genetic Algorithm is the input of Random Forest till this stage. However, the genetic algorithm also works to optimize the Random Forest. Of course, the hyperparameters of the Random Forest help in the classification process, and the right ones boost the performance. Thus, Genetic Algorithm focuses on the optimal parameters to enhance the classification accuracy. We started with the data pre-processing to clean and normalize the data. The genetic algorithm helps to extract the optimal instance subset selection. The original population of chromosomes, which the Genetic Algorithm uses as a starting point, is initialized first. In this work, the chromosomes stand in for the Random Forest machine learning algorithm's parameters. The Random Forest parameters `max_depth`, `max_features`, `min_samples_leaf`, `min_samples_split`, and `n_estimators` are optimized and used in the Genetic Algorithm's population initialization. The fitness value of each chromosome is then assessed in order to determine its selection eligibility. The individual meets the specified criteria, then it is selected. Otherwise, the individual will not be chosen. Crossover gets two parent chromosomes and generate a new individual or a solution by swapping the genes. On the other hand, the mutation technique makes some changes to the present gene for the authentic modifications.

During this process, we get optimal individuals (solutions), also known as optimal parameters. An evolutionary algorithm, more precisely a genetic algorithm, will be run through the DEAP algorithm module. This algorithm considers a number of characteristics, including population size, generational length, mutation probability, and crossover probability. The Hall-of-Famer() module from the algorithm module is necessary to store the Genetic Algorithm process. It produces a list of the top individuals who make it through the evolutionary process, denoted as best parameters. These "best parameters" refer to random forest parameters that have been optimized using the genetic method. In the end, for better performance, the classification process uses the optimized parameters. DEAP is a cutting-edge evolutionary computation framework that makes rapid concept testing and prototyping possible. Its main goal is to simplify data structures and make algorithms intelligible. Furthermore, DEAP effortlessly interfaces with

parallelization technologies like multiprocessing, enabling effective parallel execution[34]. Random Forest Classification using Genetic Algorithm is shown in Figure 3.

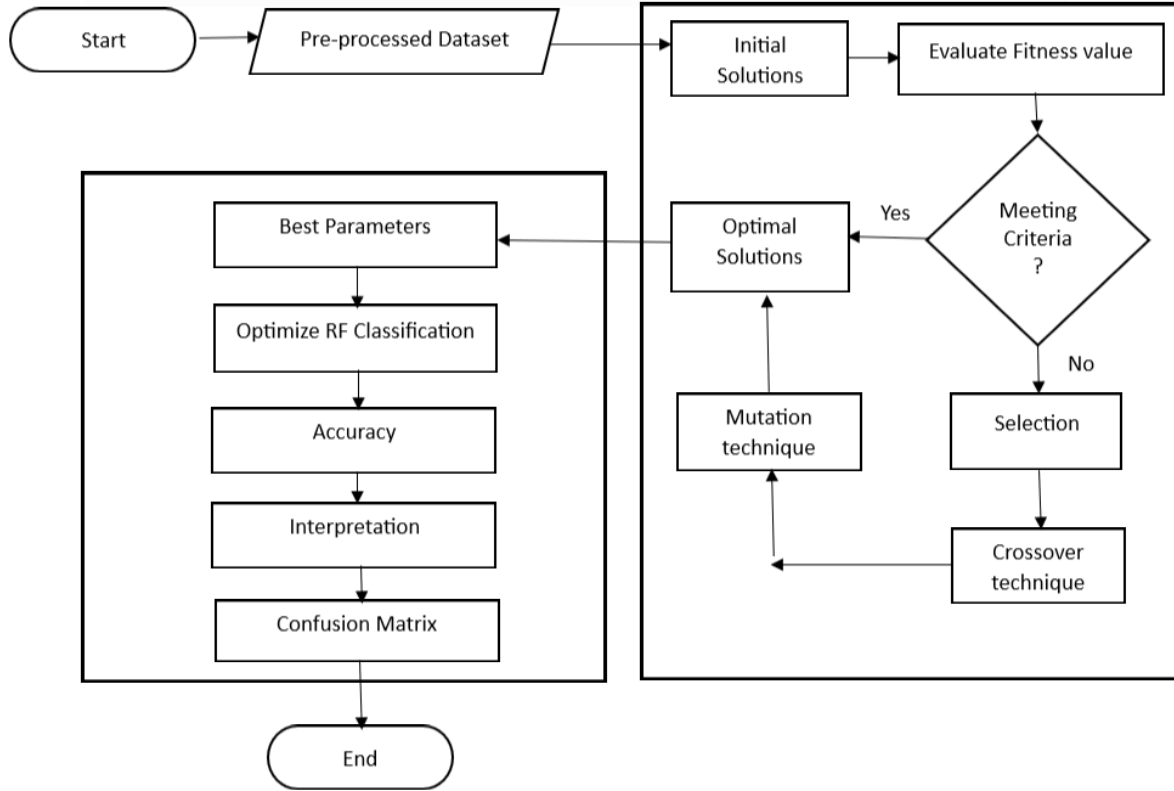


Figure 3. Optimized RF Workflow

4. EVALUATION

The Confusion Matrix is one of the evaluation tools used to rate the effectiveness of the classification model. The objective of this matrix is to assess the classification accuracy of the algorithm. One can determine the classification's accuracy value by looking at the confusion matrix. The four main terms True Positive (TP), True Negative (TN), False Negative (FN), and False Positive (FP) are used to evaluate performance using the confusion matrix. Performance metrics, also called evaluation matrices, are produced by the terms in the confusion matrix. Classification accuracy, classification error, precision, and recall are some of the evaluation matrices used. Classification accuracy is the accuracy inferred from the results of the evaluation. The number of errors or inaccuracies discovered in the evaluated data is quantified as classification error. While recall assesses the efficacy of the accuracy attained, precision measures the accuracy of the evaluation[35].

- **Accuracy** = $(TP + TN) / (TP + TN + PF + FN)$
- **Precision** = $TP / (TP + FP)$
- **Recall** = $TP / (TP + FN)$
- **Error rate** = $(FP + FN) / (TP + TN + FP + FN)$

4.1 Experimentation Results

The results of several performance measures, including as accuracy, error, precision, and recall, are shown in the below Table 2.

Table 2. Evaluation Results Analysis

Experimental Analysis	Accuracy	Recall	Precision	Error
Default	0.81	0.89	0.85	0.18
Best Parameters	0.86	0.90	0.89	0.14

The Random Forest algorithm, when optimized, outperforms other algorithms in terms of attaining correct classification, according to these metrics. The graphical representation of default and improved results are shown below in Figure 4 to compare the improved values visually.

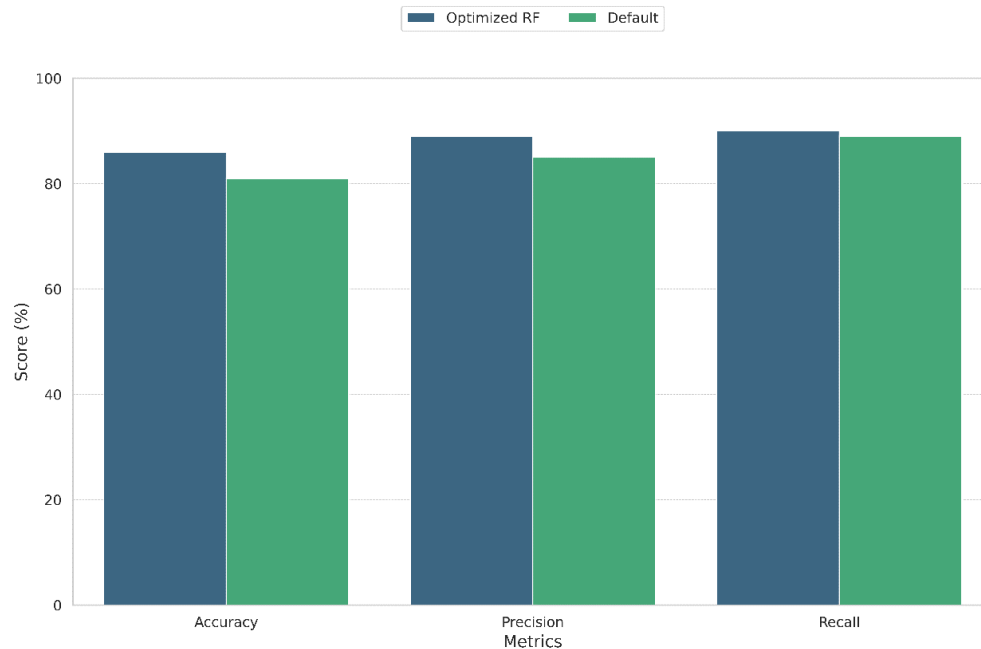


Figure 4. Comparative Evaluation Results

5. CONCLUSION

In our study, a precise standard for calculating the importance of a subset of cases was established. In addition, we developed a genetic algorithm-based method to choose the best subset from a big dataset. The defined criterion was used as the fitness function within the genetic algorithm framework, and the genetic algorithm was used for the purpose of subset selection. A classification technique known as Random Forest performs admirably in classification tasks. It includes a number of factors that are important for building an efficient categorization model. This study focuses specifically on genetic algorithms (GA) to improve five Random Forest (RF) parameter values. The Random Forest parameters are used in the Genetic Algorithm's initial population as input during the optimization process. The Genetic Algorithm performs a series of operations on the Random Forest parameters, including selection, crossover rate, and mutation rate. The classification accuracy is impressively increased to 85.8% by combining Random Forest and the Genetic Algorithm (RF + GA). Notably, the Genetic Algorithm's variables, including Generation, Population, Crossover Rate, and Mutation Rate, have an impact on this accuracy. Therefore, it can be said that Genetic Algorithms efficiently optimize the Random Forest parameters and greatly improve the accuracy of the final Random Forest model.

REFERENCES

- [1] S. Xiao, L. Zhu, X. Zhu, X. J. I. J. o. S. T. i. A. E. O. Geng, and R. Sensing, "Optimized Subset Selection for Matrices and Its Application in Hyperspectral Band Selection," 2025.
- [2] M. Moran, T. Cohen, Y. Ben-Zion, and G. J. I. S. Gordon, "Curious instance selection," vol. 608, pp. 794-808, 2022.

- [3] Y. Wang, J. Zhou, and J. J. N. Guan, "A lightweight video anomaly detection model with weak supervision and adaptive instance selection," vol. 613, p. 128698, 2025.
- [4] N. Jankowski and M. Grochowski, "Comparison of instances selection algorithms i. algorithms survey," in *International conference on artificial intelligence and soft computing*, 2004, pp. 598-603: Springer.
- [5] M. García-Borroto, Y. Villuendas-Rey, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "Finding Small Consistent Subset for the Nearest Neighbor Classifier Based on Support Graphs," in *Iberoamerican Congress on Pattern Recognition*, 2009, pp. 465-472: Springer.
- [6] G. J. I. t. o. i. t. Gates, "The reduced nearest neighbor rule (corresp.)," vol. 18, no. 3, pp. 431-433, 1972.
- [7] D. L. J. I. T. o. S. Wilson, Man, and Cybernetics, "Asymptotic properties of nearest neighbor rules using edited data," no. 3, pp. 408-421, 1972.
- [8] P. Hernandez-Leal, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, and J. A. J. P. R. Olvera-Lopez, "InstanceRank based on borders for instance selection," vol. 46, no. 1, pp. 365-375, 2013.
- [9] G. D. Cavalcanti and R. J. J. E. S. w. A. Soares, "Ranking-based instance selection for pattern classification," vol. 150, p. 113269, 2020.
- [10] C. Liu, W. Wang, M. Wang, F. Lv, and M. J. K.-B. S. Konan, "An efficient instance selection algorithm to reconstruct training set for support vector machine," vol. 116, pp. 58-73, 2017.
- [11] M. Kawulok and J. Nalepa, "Dynamically adaptive genetic algorithm to select training data for SVMs," in *Ibero-American conference on artificial intelligence*, 2014, pp. 242-254: Springer.
- [12] M. Aslani and S. J. A. S. C. Seipel, "A fast instance selection method for support vector machines in building extraction," vol. 97, p. 106716, 2020.
- [13] J. Chen, C. Zhang, X. Xue, and C.-L. J. K.-B. S. Liu, "Fast instance selection for speeding up support vector machines," vol. 45, pp. 1-7, 2013.
- [14] Á. Arnaiz-González, A. González-Rogel, J.-F. Díez-Pastor, and C. J. P. i. A. I. López-Nozal, "MR-DIS: democratic instance selection for big data by MapReduce," vol. 6, pp. 211-219, 2017.
- [15] I. Triguero, D. Peralta, J. Bacardit, S. García, and F. J. n. Herrera, "MRPR: A MapReduce solution for prototype reduction in big data classification," vol. 150, pp. 331-345, 2015.
- [16] L. Si, J. Yu, W. Wu, J. Ma, Q. Wu, and S. J. P. c. s. Li, "RMHC-MR: Instance selection by random mutation hill climbing algorithm with MapReduce in big data," vol. 111, pp. 252-259, 2017.
- [17] Á. Arnaiz-González, J.-F. Díez-Pastor, J. J. Rodríguez, and C. J. K.-B. S. García-Osorio, "Instance selection of linear complexity for big data," vol. 107, pp. 83-95, 2016.
- [18] Z. Zhu, Y. Zhang, and Y. J. a. p. a. Xia, "Optimal Criteria for Best Subset Selection," 2025.
- [19] Z. Rustamov, A. Zaitouny, R. Damseh, and N. J. a. p. a. Zaki, "GAIS: A Novel Approach to Instance Selection with Graph Attention Networks," 2024.
- [20] Z. Rustamov, N. Zaki, J. Rustamov, A. Zaitouny, and R. J. A. I. R. Damseh, "Graph reduction techniques for instance selection: comparative and empirical study," vol. 58, no. 2, p. 62, 2024.
- [21] R. Knauer and E. Rodner, "Cost-sensitive best subset selection for logistic regression: a mixed-integer conic optimization perspective," in *German Conference on Artificial Intelligence (Künstliche Intelligenz)*, 2023, pp. 114-129: Springer.
- [22] V. Singh and M. J. a. p. a. Sun, "Solving the Best Subset Selection Problem via Suboptimal Algorithms," 2025.
- [23] Z. Li, W. Fan, and L. J. J. a. p. a. Hong, "Sample-Optimal Large-Scale Optimal Subset Selection," 2024.
- [24] S. Moka, B. Lique, H. Zhu, S. J. S. Muller, and Computing, "COMBSS: best subset selection via continuous optimization," vol. 34, no. 2, p. 75, 2024.
- [25] R. M. Marone, F. Camara, and S. Ndiaye, "LSIS: Large scale instance selection algorithm for big data," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, 2017, pp. 2353-2356: IEEE.
- [26] G. E. Melo-Acosta, F. Duitama-Muñoz, and J. D. J. a. p. a. Arias-Londoño, "An Instance Selection Algorithm for Big Data in High imbalanced datasets based on LSH," 2022.
- [27] J. Zhai and Y. J. A. i. C. I. Huang, "Instance selection for big data based on locally sensitive hashing and double-voting mechanism," vol. 2, no. 2, p. 20, 2022.
- [28] (2025, 28/05/2025). *UCI Machine Learning Repository*. Available: <https://archive.ics.uci.edu/>
- [29] (2025, 28/05/2025). *UCI Machine Learning Repository: KDD Cup 1999 Data Data Set*. Available: <https://archive.ics.uci.edu/dataset/130/kdd+cup+1999+data>
- [30] U. Rashid, M. Qadir, M. Alam, and S. J. T. J. Farid, "A Hybrid Machine Learning Model to Enhance Cybersecurity: An Integration of KNN, RF and XGBoost," vol. 29, no. 04, pp. 25-32, 2024.
- [31] G. Louppe, "Understanding random forests: From theory to practice," Universite de Liege (Belgium), 2014.

- [32] J. J. S. P. Carr, "An introduction to genetic algorithms," vol. 1, no. 40, p. 7, 2014.
- [33] Y. Ge, B. Xin, L. Zhou, X. J. I. J. o. M. L. Li, and Cybernetics, "Selecting park locations using a genetic algorithm and comprehensive satisfaction," vol. 11, pp. 1331-1338, 2020.
- [34] T. Lindner and D.-. collaboration, "DEAP-3600 data acquisition system," in *Journal of Physics: Conference Series*, 2015, vol. 664, no. 8, p. 082026: IOP Publishing.
- [35] U. Rashid *et al.*, "Anomaly Detection using Clustering (K-Means with DBSCAN) and SMO," vol. 7, no. 02, 2024.