PERFORMANCE EVALUATION OF SOME SELECTED LEARNING ALGORITHMS FOR SOFTWARE DEFECT PREDICTION USING HARMONY SEARCH ALGORITHM

FOLORUNSHO Bamisaye[1], V.B. Oyekunle[2], O.J. Olabode[3]
[1]Thomas Adewumi University, Nigeria, [2]Lead City University, Nigeria, [3]Thomas Adewumi University, Nigeria
[1]folorunsho.bamisaye@tau.edu.ng, [2]bola.oyekunle@lcu.edu.ng,
[3]omosola.olabode@tau.edu.ng

Keywords: Dimensionality reduction, Feature Selection, Defect, classification

Abstract: *Software defect is a significant area of software development that has called for the attention of stakeholders in the last few decades. The more software evolves, the more its reliability is of paramount. The quest to predict defects or better still produce error-free software is commendable, however, other factors such as identifying the best LM for prediction and time taken for prediction called for imperative attention. The prolonged processing of dataset in prediction can lead to misclassification. The long processing of dataset is inevitable when the large dataset is used in prediction. That is why this study has applied a meta-heuristic optimization algorithm for feature selection, five classifiers- Support Vector Machine (SVM), Artificial Neural Network (ANN), K-Nearest Neigbour (KNN), Naïve Baiye (NB), and C4.5, four datasets; Mylyn, Eclipse, Lucene, and Equinox and five evaluation metrics- precision, accuracy, recall, classification time, and F1 score. The operational output of the model for prediction was developed and achieved with all the aforementioned tools. The recorded results with HSA revealed that the ANN algorithm achieved the lowest classification time of 24.09s in the Eclipse Dataset which shows that the predictive rate of ANN outperformed other classifiers used for defect classification*

1. INTRODUCTION

Software defects in software engineering are a major concern to developers, researchers, and end-users. Defects emanate from the misconception of the end-users requirements by the software developer [1]. The variance between the expected and actual software has caused colossal losses to businesses and society at large. Software defects are the mistakes of the programmer that twists the expected performance of the software to generate different or unexpected results [2]. Software defect and bug are used interchangeably. However, Bugs are faults in a source code [3]. It causes the software to produce inappropriate or undesired outputs [4]. When bugs are being found and fixed is termed debugging [5]. Errors identified after the application has been released are characterized as software defects [6]. It is an aberration of the customer requirement. With the rapid increase of software evolution in our day, software security has been a major nightmare of stakeholders. The vulnerability of the software solutions calls for imperative attention to the source of the problem (source code and programmer), the effect of bugs on the software (Software Defect), and the troubleshooting mechanism. Software Defect Prediction has been a core area of concern for researchers in the discipline of software development.

Defect prediction reduces the time and cost of development while customer satisfaction is increasing. Therefore, defect classification practices are imperative to achieving software quality. Defect prediction is identifying defective components in the source code. Classification of defective software datasets using the classifiers; Support Vector Machine (SVM), Artificial Neural Network (ANN), K-Nearest Neigbour (KNN), Naïve Baiye (NB), and C4.5 was executed in this study. The predicted results will assist developers to locate and fix potential defects, thereby improving software stability and reliability.

Software stability is a critical issue as contemporary software continues to grow. The effectiveness of MLA depends majorly on the features used for the classification phase. If features are not adequately selected, it can result in misclassification. Feature reduction is a significant phase in the ML algorithm,

this technique consists of feature extraction and selection [7]. Dimensionality reduction basically means the process of eliminating redundancy attributes from the dataset while trying as much as possible to keep the variation in the original dataset unaltered. The transformation of the dataset from a high-dimensional state to a low-dimensional state so that the subset representation holds approximately the significant aspects of the original dataset, perfectly represents the intrinsic dimension, is known as dimensionality reduction [8]. Feature Selection (FS) is imperative, and it is the picking of the discriminant feature for model construction, thus reducing training times, classification time, and misclassification. This study developed a SD predictive model using the aforementioned 5 MLAs, HSA (as feature selection), and 5 evaluation metrics

2. RELATED WORKS

Olatunji B. L. *et al.* [9] applied an optimization algorithm called GA to pick the most important features in order to reduce the classification time and the possibility of misclassification. One Classifier, ANN was employed for defect classification. The system was implemented in MATLAB (R2018a) simulation environment with the execution of a numerical toolkit and the operating model was measured using four evaluation metrics. The system used Eclipse, Lucene, Equinox, and ECLIPSE JDT CORE to evaluate the performance. The result is represented in a tabular form below:

MLA	Datasets	Evaluation Metrics			
Classifier	Dataset	Accuracy	Precision	Recall	F-Score
ANN	ECLIPSE JDTCORE	86.93	53.49	79.31	63.89%
ANN	ECLIPSE PDE UI,	83.26	31.91	45.45	37.50%
ANN	EQUINOX FRAMEWORK	84.43	57.69	45.45	50.80%
ANN	LUCENE	91.30	33.33	50.00	40.00%

Table 2.1

According to the literature survey, some researchers have examined the performance and comparison on some selected MLAs for the prediction of software defects. The study considered three different versions of the eclipse version control system. Data were split into training and tested sets. For the training purpose of the model Support Vector Machine (SVM) and ELM were used. The operational classification model was achieved using some selected evaluation metrics. The result recorded the SVM to be the best fit for the cross-version defect prediction [10].

Furthermore, Rao and Patra developed a SD predictive system in the year 2020 using a genetic algorithm with a shuffled frog leap algorithm. The study assessed the classification accuracy, precision and recall, and F measure for various classifiers used. The ANN optimizations assumed that more than two algorithms for one optimization have been implemented. The optimization used a meta-heuristic to select the best features for classification. The hybrid optimization technique for creating the linkages method was applied for the dimensional synthesis of the mechanism. The results, 5.94% of the NN-hybrid classifier showed that it outperformed the fuzzy algorithm by 3.59% and the 1.42% value of the NN-Lm training respectively.

Gray et al. [11] offered a study using static code metrics and a Support Vector Machine classifier for a group of modules contained inside eleven NASA data sets. Before classification, the data underwent a comprehensive pre-processing stage that included balancing the two classes (defective or anything else) and eliminating numerous repetitive events. In this study, the SVM achieves a normal accuracy of 70% on previously unnoticed data. According to the evaluated relevant works, previously created software prediction systems have an overfitting limitation, which occurs when the system acquires so much detail in the training data that it severely affects the system's performance on fresh data.

More also, Sanusi, B. A. et al. [12] conducted a defect classification model using ML-based algorithms. To choose the discriminant characteristics, the Genetic Algorithm, a nature-inspired meta-heuristic algorithm, was used, using RF, DT, and ANN classification techniques, the selected features were classified into defect or non-defect. Accuracy, f-score, precision, and recall were also used to assess the strategies. The RF and other learning algorithms were evaluated in terms of accuracy, precision, and f-score, with average scores of 83.40 percent, 53.18 percent, and 52.04 percent, respectively, among all the algorithms used. In a summary, the ANN outperformed other MLA as it recorded the best recall value result with an average score of 60% among the algorithms.

Bayesian networks are used by Fenton and Neil [13] to forecast software quality and defects. It considers the restrictions of traditional software obstacles by utilizing random process elements as well as qualitative and quantitative metrics. The use of a sophisticated discretization algorithm results in a more accurate software fault prediction system.

Furthermore, Rao, S. V. A. *et al.* [14] developed a SD predictive system in the year 2020 using a genetic algorithm with a shuffled frog leap algorithm. The study assessed the accuracy, precision, classification, recall, and F measure for various classifiers used. The ANN optimizations assumed that more than two algorithms for one optimization have been implemented. The optimization used a meta-heuristic to select the best algorithm for classification. The hybrid optimization technique for creating the linkages method was applied for the dimensional synthesis of the mechanism. The results, 5.94% of the NN-hybrid

classifier showed that it outperformed the fuzzy algorithm by 3.59% and the 1.42% value of the NN-Lm training respectively.
Lastly, in the year 2014, Ren, Qin, Ma, and Luo [26] also conducted how the kernel techniques can be applied to classify defects in software development since the class discrepancy can influence negatively, the correct operation of defect classification. There are two classifiers that were applied which are, asymmetric kernel partial least squares (AKPLS) and asymmetric kernel principal component analysis (AKPCAC) classifier. The study aimed at solving the class inconsistency or imbalance issue. Finally, the study achieved its aim by using a kernel function for the asymmetric partial least square classifier and asymmetric PCA classifier, respectively. The kernel function used for the two classifiers is the Gaussian function. The experiment was conducted in NASA and SOFTLAB datasets using the *F*-measure, Friedman's test, and Tukey's test to confirm the validity of the methods.

Feature Selection Technique for Software Defect Classification

To reduce prolonged processing of datasets which results in misclassification, this study has come up with a model developed using a meta-heuristic optimization algorithm, 5 MLAs, and 5 metrics. The significance of HSA in this study is to pick a relevant feature for defect prediction. The 5 MLAs were applied for classification and 5 metrics were used to determine the level of defect.
In the first step, the study used HSA subset selection method to select pertinent feature subsets.
The output of the FS method retains the meaningful representation or intrinsic dimension of the original dataset
In the second step, the subset was used to predict defects using 5 MLAs. The results of these experiments are described in detail in the results and discussions section
Lastly, 5 metrics were applied to determine the genuineness and the level of defect

3. METHODOLOGY

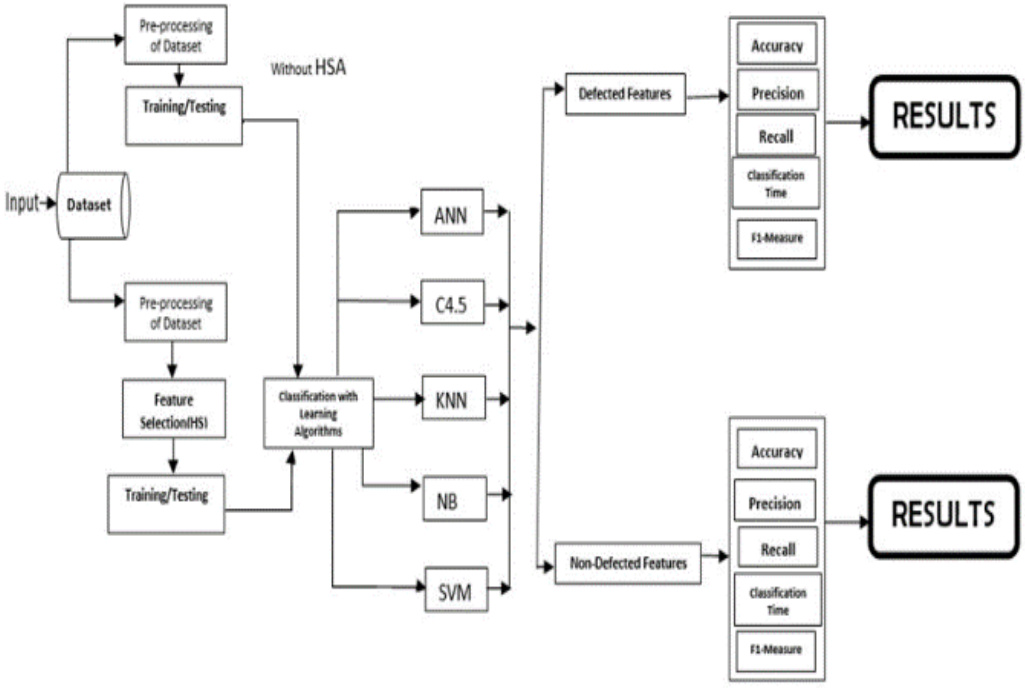

Figure 3.1: System Model. Block diagram of software defect predictive model

3.1 Requirements specification

The device with which this study generated outputs was a laptop computer with specifications: 3.4 GHz Intel Core i7 8GB RAM, running Windows OS 64-bit (a laptop or Desktop computer with much lower specifications can run this simulation). The simulation took place in python 3.10 to evaluate the efficiency of the classifiers which consist of naïve Bayes, C4.5, SVM, K-nearest neighbour, and Artificial Neural Network. The study applied an optimization algorithm to obtain relevant features.
Four datasets; Eclipse PDE UI, Lucence, Mylyn, and Equinox were processed and classified. Finally, the operational effectiveness of the classifiers was achieved using precision, accuracy, recall, classification time, and F1-score evaluation metrics.

3.3 Data Acquisition

The implementation section of this study required the accessibility of the preceding defective software dataset. The dataset for this study was acquired from the dataset bank repository that is available for public use. The software datasets that were used for defect classification are Eclipse, Equinox. Lucene, and Mylyn. Each of these datasets held a unique piece of information, but the one that provided the necessary parameter for the research activity was used during the implementation process.

3.4 Dataset Pre-processing

Dataset pre-processing in ML is a sensitive and significant area that helps enhance the selection of the discriminant feature of the dataset for classification. The technique of cleaning and organizing the dataset make it appropriate for training, constructing, and testing the MLAs. Dataset pre-processing is a data mining method of transforming a dataset into a readable and more understandable format.
The initiation of dataset pre-processing is required to clean, format, and organize the raw dataset, thereby making it useable for ML models. Dataset Acquisition is primarily, the fundamental prerequisite in data pre-processing as it precedes and is required for the development of any ML model. To build and develop ML models is to first acquire the relevant dataset. Thereafter, the dataset was split into two separate sets – training and testing. Follow by importing the dataset that has been tested and trained into the ML model for classification.

Table 3.1 Information on Datasets

Dataset	Languages/Description	Instances
Lucene	Free Apache search software	691
Mylyn	Eclipse's task, application, lifecycle management framework	1862
Equinox	OSGi framework implementation applied for all Eclipse	324
Eclipse	Tools of a user interface for developing and creating Eclipse plug-ins and features.	997

Figure 3.2 below shows the simulation interface for the experimental setup.

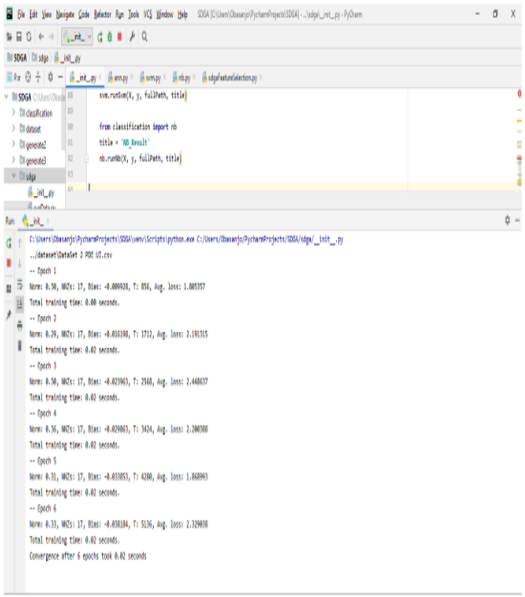

Figure 3.2: Interface for Experimental Setup

3.5 Feature Selection

Feature selection (FS) can be described as a preprocessing method applied to pick relevant features. It is a vital component of pattern recognition and ML since it can lower computation costs while improving classification performance. Dimensionality reduction occurs when only the significant or relevant features are selected resulting to the precise classification. When compared to the original datasets, a smaller feature set improves classification accuracy.

In the MLA, several FS methods have been introduced. The basic goal of this method is to eliminate the unnecessary or redundant features from the dataset. Wrapper and filter are the two types of FS techniques. The wrapper examines and picks attributes based on the target learning algorithm's accuracy estimates. Wrapper simply searches the feature space by omitting various features and analyzing the impact of feature absence on the prediction metrics using a specific learning method [15], [16].

3.6 Machine Learning Algorithms Discussed

Machine learning studies automatic methods for learning to make accurate predictions or useful decisions that is based on previous experience observation and experience. It has grown in popularity, with applications in a variety of fields including natural language processing, speech recognition, computer vision, and gene discovery. An ML technique can build models based on labeled

historical websites and then these models can be integrated into the browser to detect phishing activities. When a user surfs a new web page, ML models guess the type of the website in real-time and then communicate the outcome to the end-user.

CLASSIFICATION ALGORITHMS

3.6.1 Artificial Neural Network (ANN)

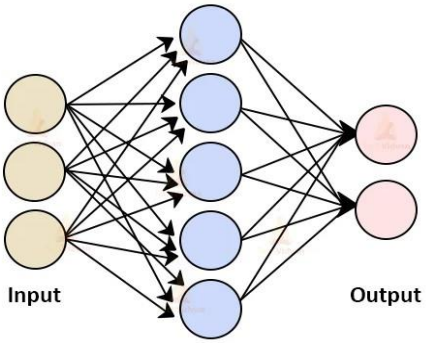

Figure 3.3: ANN [17].

ANN is fundamentally based on biological principles of neural networks, which is an emulation of the biological neural system. It contains a connected set of artificial neurons and processes information using a method of connectionist for computation. ANN are models which draw their inspiration from biological nervous systems which comprise neural networks. Artificial Neural Networks (ANN) are mathematical models that can be applied to model complex which its connection between inputs and outputs or to find patterns in the data. This technique learns by examples, they are trained with known examples of the problem that knowledge is to be acquired from when trained well, it can be used effectively to provide a solution to similar problems of unknown instances [17].

3.6.2 Support Vector Machine (SVM)

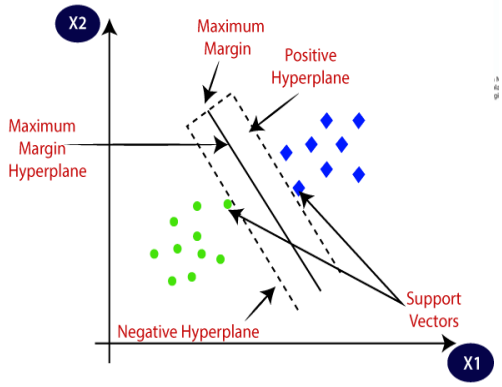

Figure 3.4: SVM [18]

The SVM belongs to a generalized family of linear classifiers, which is mostly used in classification, regression, and prediction tools. The SVM algorithms apply models linearly to introduce class boundaries non-linearly by converting the instance space applying a nonlinear matching into a new space, a linear model constructed in the new space can then represent a nonlinear decision boundary in the original space. [18]

3.6.3 Naïve Bayes (NB)

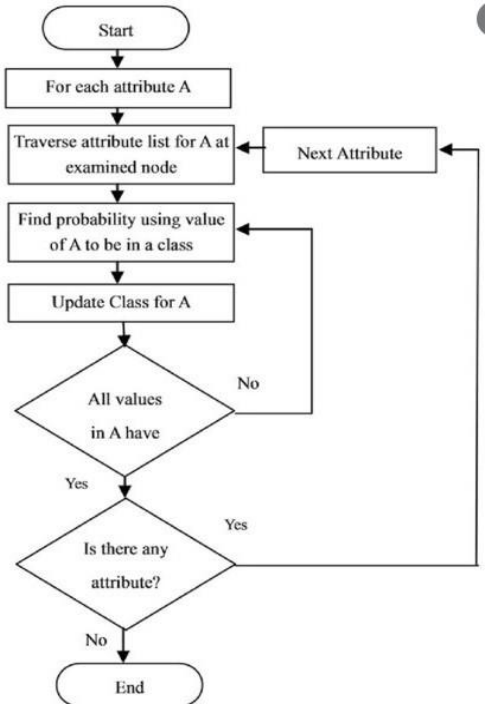

Figure 3.5: NB [19]

The Nave Bayes Classifier is a simple and effective classification method that aids in the development of fast machine learning models capable of making quick predictions. [19]. It is a supervised learning technique for addressing classification issues that are based on the Bayes theorem [20]. The naive theorem is combined with an attribute condition where it is assumed to be independent. The system scrutinizes the connection between each attribute and the class for each instance in order to calculate a conditional probability for the relationship between attribute values and class. This technique has been declared to be very effective with actual datasets and when combined with feature selectors eliminate redundant and unimportant features. The Bayes theorem is given as follows:

$$P(H/X) = \frac{P(H)P(\frac{X}{H})}{P(X)} \qquad (2.1)$$ [21].

Where P(H/X) represents the likelihood that activity H occurred given that test X was positive) P(X/H) signifies the likelihood that activity X occurred given that test H was positive. Also, P(H) interprets the possibility that event H occurred and P(X) characterizes the probability that event x took place.

3.6.4 K-Nearest Neighbour (KNN)

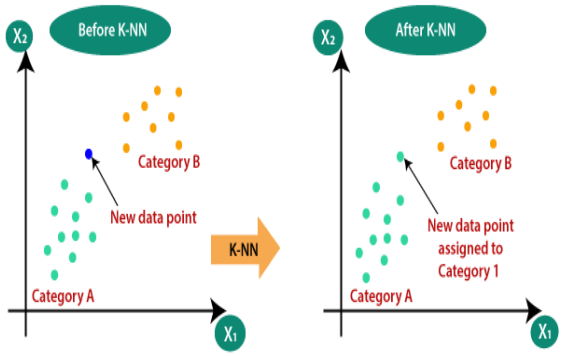

Figure 3.6: KNN [21].

The algorithm applied to predict objects and data based on the closest training samples in the feature space. It represents one of the atomic techniques in ML. It is basically an analogy that has its basis on the fact that objects that are near each other will also have similar characteristics. This approach is also known as classification based on memory as the samples of training must be memory during execution. This classification technique is the basic classification technique with little or no prior knowledge about the data distribution [21]

3.6.5 C4.5 Algorithm

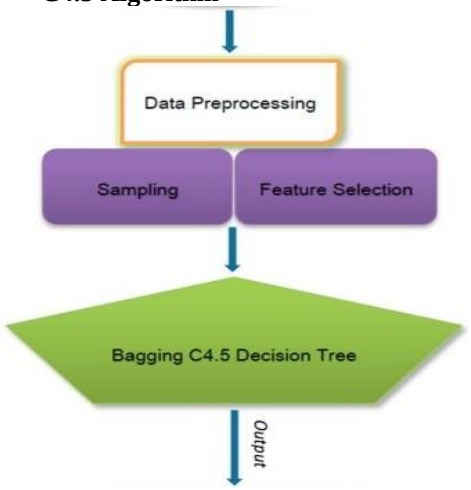

Figure 3.7: C4.5 [22]

Ross Quinlan created the C4.5 algorithm, which is used to produce a decision tree. C4.5 generates decision trees that can be employed for defect prediction, which is why it's commonly referred to as a statistical classifier. The C4.5 algorithm was defined as a momentous DT program that is possibly the most commonly used ML by the Weka application software [21].

4 EVALUATION METRICS

The operational evaluation metrics employed are Accuracy, Recall, classification time, Precision, and F-score

i.Sensitivity = $\frac{TP}{TP+FN}\ X\ 100\%$ (3.1)

ii.Accuracy = $\frac{TN+TP}{TP+FP+TN+FN}\ X\ 100\%$ (3.2)

iii.Precision = $\frac{TP}{TP+FP}$ (3.3)

iv.F-measure = $\frac{Precision \times \frac{TP}{TP+FN}}{Precision + \frac{TP}{TP+FN}}$ (3.4)

where TP, FP, TN, and FN are defined as follows:

True positive (TP): The outcome is the correctly predicted positive, indicating that the actual and projected results are both "yes."

True negative (TN): The outcome is the correctly predicted negative, indicating that both the actual and expected result values are "No."

False-Positive (FP): This indicates that the actual result is no, but the projected outcome is yes.

False-negative (FN): This indicates that the actual result is yes, but the projected outcome is no. [23]

4.1 Accuracy

When using asymmetric datasets, where the false positive and false negatives have the same value, accuracy is defined as the percentage of properly predicted values to the total. It's the proportion of subjects who have been correctly identified to the total number of subjects. Accuracy is the most intuitive one. The following question is answered by accuracy:

How many nations of Africa did we correctly label corrupt out of all the nations?

Accuracy = (TP+TN)/(TP+FP+FN+TN)

numerator: refers to all correctly labeled subjects (All trues)

denominator: refers to all subjects

4.2 Precision

Precision is the function of pertinent instances among the recovered instances. This is the proportion of the correctly and positively categorized by the program to all positive categorized.

Thus, the precision answers the following: How many presidents of Africa nations that we labeled as competent are actually competent?

Precision = TP/(TP+FP)

numerator: labeled competent president.

denominator: all labeled by the program (whether they're competent or not in reality).

4.3 Recall

Sensitivity is another name for recall. The percentage of accurately forecasting positive for everyone in the actual result is referred to as recall. It is also the percentage of the correctly positive identified by the program to all who are competent in reality.
The following questions are answered by recall: of all the presidents who are corrupt, how many of those do we correctly predict?
Recall = TP/(TP+FN)
numerator: positively labeled corrupt Presidents.
denominator: all presidents who are corrupt (whether detected by the program or not)
denominator: all labeled by the program (whether they're competent or not in reality).

4.4 F1-Score

F1-score is also known as F-Score or F-Measure. The F1-Score is the sum of Precision and Recall, adjusted for false positives and negatives.
When the data distribution is imbalanced, F1-Score is more effective than accuracy. Precision and recall are both taken into consideration while calculating the F1 Score. It's the harmonic mean of memory and precision. F1-measure is excellent if a balance can be accomplished between precision (p) and recall (r) in the system. Oppositely F1-Score is not so high if one measure is improved at the expense of the other. For example, if P is 1 and R is 0, F1 score is 0. F1-Score = 2*(Recall*Precision) / (Recall + Precision) [23].

4.5 Classification Time

Classification time is the time taken to complete software defect classification using an MLA.

5. RESULTS AND IMPLEMENTATION

This section explains in tabular forms, the result as well as the display of the simulation environment where the outputs of defect classification were generated. It mentions the result and the objectives. listed for the study as well as the workability of the methodology used to achieve these objectives.

5.1 Discussion of Findings

The Feature Selection (FS), defects classification as well as performance evaluation of the results (defect or non-defect) were conducted for different datasets used. The results were provided using the following tables.

Table 5.1: Performance Metrics for Lucene Dataset (without HSA)

Classifier	**Evaluation Metrics**				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	92.04	0.9200	1.000	0.9600	26.31
C4.5	84.13	0.9000	0.9200	0.9100	27.45
KNN	93.26	0.9300	1.000	0.9700	27.52
NB	86.54	0.9100	0.9500	0.9300	27.61
SVM	89.39	0.8900	1.000	0.9400	27.80

From Table 5.1, ANN has the maximum accuracy value of 92.04%, KNN has the best precision value of 0.9300, the best Recall value of 1.000 was obtained in KNN, ANN, and SVM. KNN has the best F1-score value of 0.9700 and ANN has the lowest classification time of 26.31s was obtained in Lucene.

Table 5.2: Performance Metrics for Lucene Dataset (with HSA)

Classifier	**Evaluation Metrics**				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	90.25	0.9000	0.9500	0.9500	27.20
C4.5	84.13	0.9000	0.9200	0.9100	27.30
KNN	93.27	0.9300	1.000	0.9700	27.40
NB	86.54	0.9100	0.9500	0.9300	27.30
SVM	89.39	0.8900	1.000	0.9400	27.35

From Table 5.2, KNN recorded a value of 93.27% at its highest degree, precision at the best degree value of 0.9300 was recorded in KNN. SVM and KNN both have the best Recall value of 1.000, ANN has the best F1-score value of 0.9500 as well as lowest classification time of 27.20s, all obtained in Lucene dataset.

Table 5.3: Performance Metrics for Mylyn Dataset (without HSA)

Classifier	**Evaluation Metrics**				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	63.08	0.6300	0.9900	0.7700	26.01
C4.5	66.33	0.7500	0.6900	0.7200	26.05
KNN	75.51	0.7700	0.8200	0.8000	26.10
NB	69.39	0.6800	0.9500	0.7900	26.15
SVM	61.22	0.6400	0.8500	0.7300	26.35

From Table 5.3, SVM has the most correct value of 86.31%, NB has the highest precision degree of 0.9000, SVM has the best Recall value of 1.000, and SVM has the best F1-score value of 0.9800, all obtained Mylyn Dataset.

Table 5.4: Performance Metrics for Mylyn Dataset (with HSA)

Classifier	Evaluation Metrics				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	85.64	0.8600	1.000	0.9200	29.07
C4.5	78.18	0.8800	0.8600	0.8700	29.19
KNN	86.94	0.8800	0.9900	0.9300	29.29
NB	85.15	0.9000	0.9400	0.9200	29.46
SVM	87.30	0.8700	1.000	0.9300	29.8

From Table 5.4, KNN has the most accurate value of 86.94%, NB recorded the most precision value of 0.9000, and the best Recall value of 1.000 was obtained in both SVM and ANN, SVM has the best F1-score value of 0.9300 all obtained in Mylyn Dataset.

Table 5.5: Performance Metrics for Equinox Dataset (without HSA)

Classifier	Evaluation Metrics				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	48.03	0.4800	1.000	0.6500	25.61
C4.5	66.33	0.7500	0.6900	0.7200	26.08
KNN	75.51	0.7700	0.8200	0.8000	26.12
NB	69.38	0.6800	0.9100	0.9500	26.21
SVM	61.22	0.6400	0.8500	0.7300	26.50

From Table 5.5, KNN has the leading accuracy of 75.51%. KNN has the peak precision of 0.7700, ANN obtained the topmost Recall of 1.000, and the maximum F1-score value of 0.9500 was obtained in NB and the classification time of 26.08s was recorded in C4.5 all obtained Equinox Framework Dataset.

Table 5.6: Performance Metrics for Equinox Dataset (with HSA)

Classifier	Evaluation Metrics				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	84.52	0.8500	0.9900	0.9100	27.74
C4.5	79.36	0.8800	0.8800	0.8800	28.27
KNN	85.51	0.8700	0.9800	0.9200	28.43
NB	83.33	0.9000	0.9100	0.9000	28.72
SVM	86.31	0.8600	1.000	0.9800	28.90

From Table 5.6, KNN has the leading accuracy of 75.51%, KNN has the uppermost precision of 0.7700, ANN has the best Recall value of 0.9900, NB has the best F1-score value of 0.8000 and ANN has the classification time of 26.01s all obtained Equinox Framework Dataset

Table 5.7: Performance Metrics for Eclipse PDE UI Dataset (without HSA)

Classifier	Evaluation Metrics				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	89.16	0.8900	1.000	0.9400	24.14
C4.5	78.09	0.8600	0.8800	0.8700	24.44
KNN	84.85	0.8600	0.9800	0.9200	24.56
NB	81.35	0.8600	0.9300	0.8900	24.68
SVM	83.22	0.8500	1.000	0.9200	24.70

From Table 5.7, ANN has the best accuracy of 89.16%. ANN has the best precision of 0.8900. Also, ANN has the best Recall of 1.000. ANN and SVM both have the best F1-score value of 0.9400was. The lowest classification time of 24.14s was recorded in ANN all obtained Eclipse Dataset.

Table 5.8: Performance Metrics for Eclipse PDE UI Dataset (with HSA)

Classifier	Evaluation Metrics				
	Accuracy (%)	Precision	recall	F1-Score	Classification Time (s)
ANN	85.97	0.8600	1.000	0.9200	24.09
C4.5	77.78	0.8800	0.8700	0.9000	25.04
KNN	83.11	0.8400	0.9900	0.9100	25.18
NB	81.56	0.8800	0.9100	0.8900	25.48
SVM	86.44	0.8600	1.000	0.9300	25.52

From Table 5.8, SVM has the highest accuracy value of 86.44%, ANN has the topmost precision value of 0.8900, ANN has the best Recall value of 1.000 while SVM also has the best F1-score value of 0.9400was in ANN, and finally, the lowest classification time of 24.14s was recorded in ANN all obtained Eclipse Dataset.

6. COMPARATIVE ANALYSIS RESULTS OF DEFECT PREDICTION

Comparative analysis of defect prediction was conducted for the different datasets with respect to different datasets used. The results were provided using the following Figures.

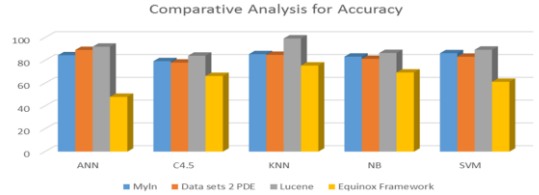

Figure 6.1: Comparative Analysis for Accuracy (%) (Without HSA)

From Figure 6.1, the highest accuracy of 99.26% was obtained in KNN for Lucene dataset, while the lowest accuracy of 48.03% was recorded in ANN for

Equinox framework dataset all obtained when the HSA was not applied for FS

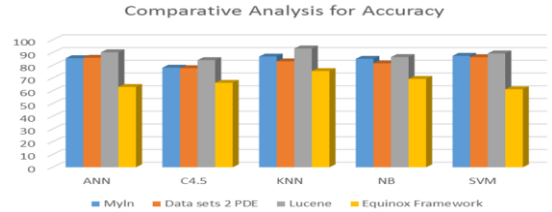

Figure 6.2: Comparative Analysis for Accuracy (%) (With HSA)

From Figure 6.2, the highest accuracy of 93.27% was obtained in KNN for Lucene dataset, while the lowest accuracy of 61.22% was recorded in SVM for Equinox framework dataset all obtained when the HSA was applied for FS

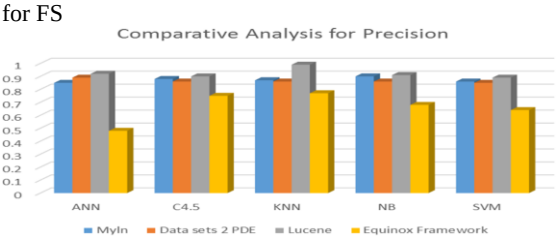

Figure 6.3: Comparative Analysis for Precision (without HSA)

From Figure 6.2, KNN has the top precision value of 0.9900 in KNN for Lucene dataset, while the lowest accuracy of 0.4800 was recorded in ANN for Equinox framework dataset all obtained when the HSA was not applied for feature selection

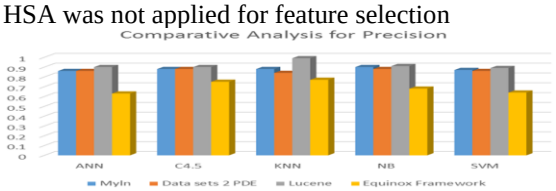

Figure 6.4: Comparative Analysis for Precision (with HSA)

From Figure 6.4, KNN has the peak precision value of 0.9900 in KNN for Lucene dataset, while the lowest accuracy of 0.6300 was recorded in ANN for Equinox framework dataset all obtained when the HSA was applied for FS

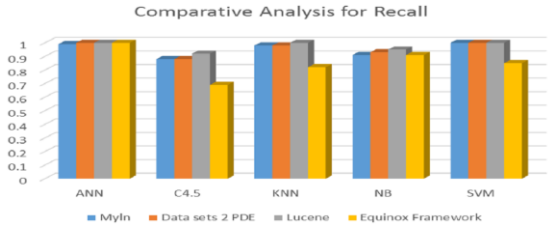

Figure 6.5: Comparative Analysis for Recall (without HSA)

From Figure 6.5, the highest recall value of 1.000 was obtained in ANN for Eclipse, Lucene dataset, also 1.000 was recorded in KNN for Lucene, 1.000 was recorded in SVM, for Mylyn, Eclipse Dataset, Lucene and Equinox Framework, the lowest recall value of 0.6900 was obtained in C4.5 for Equinox Framework when no FS was applied

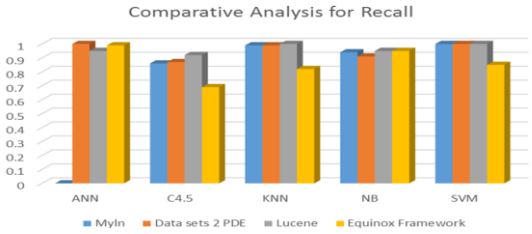

Figure 6.6: Comparative Analysis for Recall (with HSA)

From Figure 6.6, the highest recall value of 1.000 was obtained in ANN for Mylyn, Lucene, 1.000 was obtained in KNN for Lucene, 1.000 was obtained in SVM for Mylyn, Eclipse, and Lucene, and the lowest recall value of 0.6900 was recorded in C4.5 for Equinox Framework when HSA was applied to pick feature.

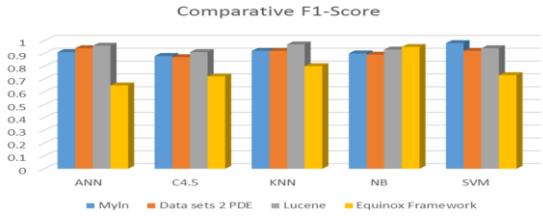

Figure 6.7: Comparative Analysis for F1-Score (without HSA)

From Figure 6.7, F1-score has the best value of 0.9800 in SVM for Mylyn, and Recall has the lowest value of 0.6500 in ANN for Equinox Framework when HAS was not was applied.

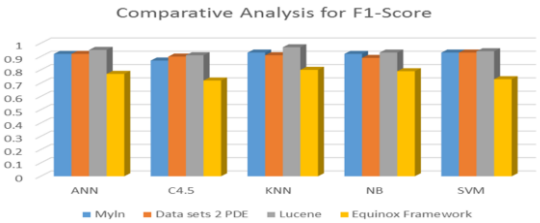

Figure 6.8: Comparative Analysis for F1-Score (with HSA)

From Figure 6.8, the best F1-score value of 0.9700 was recorded in KNN for Lucene and the lowest recall value of 0.7200 was recorded in C4.5 for Equinox Framework when the HSA was applied.

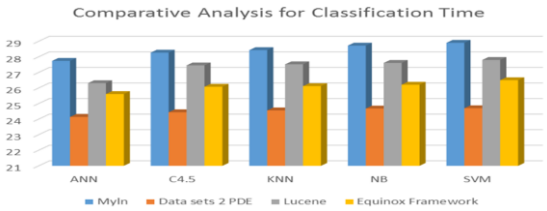

Figure 6.9: Comparative Analysis for Classification Time (s) (without HSA)

From Figure 6.9, the lowest classification time of 24.14s was obtained in ANN for Eclipse Dataset while the highest classification time of 28.90 was recorded in SVM for Mylyn when HSA was not applied.

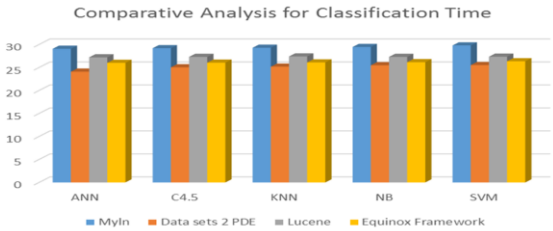

Figure 6.10: Comparative Analysis for Classification Time (s) (with HSA)

From Figure 6.10, the lowest classification time of 24.09s was obtained in ANN for Eclipse Dataset while the highest classification time of 29.80 was recorded in SVM for Mylyn when HSA was applied.

7. CONCLUSION SUMMARY OF FINDINGS

Feature selection has been considered by researchers to be one of the predominant stages in the SDP system. It involves the techniques of obtaining the discriminant feature of the original dataset in which an initial set of raw datasets is reduced to a more manageable form for classification purposes. Representation of feature produces an approximation to the original feature in fewer dimensions, while still maintaining the same structure of original features. This study designed a software defect classifying system using some selected MLAs. An optimization algorithm known as HSA was introduced for

The experimental result showed the maximum accuracy of 99.26% was obtained in KNN in the Lucene dataset when the HSA was not used, the highest precision of 0.9900 which was also the best precision was obtained in KNN recorded in Lucence when the HSA was not applied. The highest recall of 1.0000 which is also the best recall was obtained in ANN and KNN in Lucene, also 1.000 was obtained in SVM for Mylyn, Dataset 2 PDE UI, Lucence, and Equinox framework dataset when harmony search was not applied. When the HSA was not applied, the Recall has the best record value of 1.000 in ANN for Mylyn while in Lucence, 1.000 was recorded in KNN for Lucence, and 1.000 was obtained in SVM for Dataset2PDE UI and Lucence. dataset and the highest F1-score of 0.9800 which is also the best F1-score were obtained in SVM for Mylyn and the lowest classification time of 24.14s was recorded in ANN for Eclipse. The study concluded that the application of the HSA resulted in a slight improvement in some aspects of the SDP system.

7.1 Recommendation

There are tendencies for prolonged processing of the dataset and misclassification using the whole dataset discriminant feature. defect prediction. Hence, applying a FS technique reduces the risk of prolonged data processing and misclassification. The defect classification model enhances the achievement of quality and reliable software as well as improves customer satisfaction. For an efficient defect prediction, it is strongly recommended to apply FS to pick the subset of the discriminant feature.

7.2 Contribution to Knowledge

The impact or contribution of this study includes:

1. To begin, this study identified HSA to be an effective Feature Selection.
2. After that, this study also presented an active model for SDP using an optimization algorithm (HSA), classifiers (SVM, ANN, KNN, C4.5, & NB), and metrics (precision, recall, classification time, F1-score, and accuracy)
3. Artificial Neural Network has the lowest classification time of 24.04s
4. HSA was found to be very efficient in dimensionality reduction through feature selection which reduced prolonged data processing, classification time, and misclassification.
5. Finally, the results of our experiments show that selecting relevant features for classification reduces prediction time

7.3 Suggestions for further studies

The following recommendations are listed to draw the attention of future studies in SDP systems:

1. Developing a single model that can be used to select features and predict defects of text, images, and others
2. In the future, to further enrich and broaden the scope of the research, we may include more optimization algorithms, MLAs, and diversified software defect datasets in the testing and experimentation process.
3. Furthermore, we may compare the performance of the new ensemble techniques results with this study.

8. REFERENCES

[1] Anonymous "SOFTWARE DEFECT DEFINITION", web accessed 20.11.2022: https://www.lawinsider.com/dictionary/software-defect/

[2] Lakshay Sharma, "ERROR, DEFECT, AND FAILURE" web accessed 05.02.2022:

https://www.toolsqa.com/software-testing/istqb/error-defect-failure/

[3] Nadya Bakhur. "WHAT CAUSES SOFTWARE BUGS? Types of Defects in Software Testing" web accessed 09.01.2022: https://neklo.com/what-causes-software-bugs/

[4] Hussain, N., & Bixin, L. "PERFORMANCE COMPARISON OF ML CLASSIFIERS IN SOFTWARE DEFECTS PREDICTION ABSTRACT:" IOSR Journal of Computer Engineering (IOSR-JCE), https://doi.org/10.9790/0661-2205034858 (2020).

[5] Anonymous. "SOFTWARE-DEVELOPMENT" web accessed 15.12.2021: https://economictimes.indiatimes.com/definition/debugging/

[6] Anonymous. "DIFFERENCE BETWEEN DEFECT, ERROR, BUG, FAILURE AND FAULT!" web accessed 25.10.2021: https://www.360logica.com/blog/difference-between-defect-error-bug-failure-and-fault/

[7] Telgaonkar, A. H & Deshmukh, S. "Dimensionality Reduction and Classification through PCA and LDA." International Journal of Computer Applications, 122(17), 4–8. 2015.

[8] Wikipedia, the free encyclopedia, "DIMENSIONALITY REDUCTION" web accessed 28.12.2021: https://en.wikipedia.org/wiki/Dimensionality_reduction/

[9] Olatunji B. L.[1] , Olabiyisi S. O.[2] , Oyeleye C. A.[3] , Sanusi B. A.[4] , Olowoye A. O.[5] , Ofem O. A.[6] "Development of software defect prediction system using artificial neural network", web accessed 15.2021.2022: https://pdfs.semanticscholar.org/6047/c62f7d169b133e6389d6840eaca0201e140c.pdf/

[10] Hussain, N., & Bixin, L. "PERFORMANCE COMPARISON OF ML CLASSIFIERS IN SOFTWARE DEFECTS PREDICTION ABSTRACT:" IOSR Journal of Computer Engineering (IOSR-JCE), https://doi.org/10.9790/0661-2205034858 (2020).

[11] Gray, D., Bowes, D., Davey, N. and Sun, Y., "Bruce Christianson, Using the Support Vector Machine as a Classification Method for Software Defect Prediction with Static Code Metrics," 11th International Conference, EANN 2019, pp. 21-25, 2019.

[12] Sanusi, B. A., Olabiyisi, S. O., Olowoye, A. O., & Olatunji, B. L. "SDF SYSTEM USING ML-BASED ALGORITHMs." Journal of Advances

[13] Fenton, N. and Neil, M., "Using Bayesian networks to predict software defects and reliability," Proc. IMechE vol. 222 Part O: J. Risk and Reliability, pp. 702-703, 2008.

[14] Rao, S. V. A. et al. "AN ARTIFICIAL NEURAL NETWORK GENETIC ALGORITHM WITH SHUFFLED FROG LEAP ALGORITHM FOR SOFTWARE DEFECT PREDICTION.", 831–836. 2020

[15] M. Ashraf, G. Chetty, and D. Tran, "Feature selection techniques on thyroid, hepatitis, and breast cancer datasets," International Journal on Data Mining and Intelligent Information Technology Applications (IJMIA), vol. 3, no. 1, pp. 1-8, 2013.

[16] M. Leach, "Parallelising feature selection algorithms," University of Manchester, Manchester, 2012.

[17]. Wikipedia, the free encyclopedia "Artificial neural network", web accessed 20.12.2021: https://en.wikipedia.org/wiki/Artificial_neural_network

[18] Ernest Yeboah Boateng[1], Joseph Otoo[2], Daniel A. Abaye1* "Basic Tenets of Classification Algorithms K-Nearest-Neighbor, Support Vector Machine, Random Forest and Neural Network: A Review". web accessed 28.10.2021: https://www.scirp.org/journal/paperinformation.aspx?paperid=104256/.

[19] Anonymous., "Naïve Bayes Classifier Algorithm" web accessed 08.01.2022: https://www.javatpoint.com/machine learning-naive-bayes-classifier/

[20] Khan, R. U., Albahli, S., Albattah, W., Nazrul, M., & Khan, I., "Software Defect Prediction Via Deep Learning." International Journal of Innovative Technology and Exploring Engineering (IJITEE), 9(5), 343–349, 2020. https://doi.org/10.35940/ijitee.D1858.039520

[21] Anonymous., "K-Nearest Neighbor (KNN) Algorithm for Machine Learning", web accessed 08.01.2022: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning/

[22] Wikipedia, the free encyclopedia, "C4.5 algorithm" web accessed 28.12.2021: https://en.wikipedia.org/wiki/

[23] Salma Ghoneim "Accuracy, Recall, Precision, F-Score & Specificity, which to optimize on?", web accessed 26.04.2022: https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124/

[24] Patra, B., & Dash, S. (2016). A FRGSNN Hybrid FS Combining FRGS filter and GSNN wrapper. International Journal of Latest Trends in Engineering and Technology, 7(2), 8–15.

[25] Youm, K. C., Ahn, J., & Lee, E. IMPROVED BUG LOCALIZATION BASED ON CODE CHANGE HISTORIES AND BUG REPORTS. Information and Software Technology, 82, 177–192. 2017. https://doi.org/10.1016/j.infsof.2016.11.002

[26] Mahmoud A. E. Marwa S. F. & Mono G. G. "Spotted hyena, a novel meta-heuristic optimization algorithm were used for defect prediction," web accessed 09.10.2021: https://www.researchgate.net/publication/353808322_Meta-heuristic_optimization_algorithm_for_predicting_software_defects/

[27] Ren, J., Qin, K., Ma, Y., & Luo, G. (2014). On Software Defect Prediction Using Machine Learning. *Journal of Applied Mathematics*, 1–9.