THE ASIAN BULLETIN OF BIG DATA MANAGMENT
Vol. 6. Issue 1 (2026)
https://doi.org/10.62019/jeg1vv20

Survey of Machine Learning Classification Techniques for Software Defects Prediction

Quratulain

Chronicle

Article history
Received: Feb 12, 2026
Received in the **revised format:** Feb 28, 2026
Accepted: March 15, 2026
Available online: March 30 2026

Quratulain is currently affiliated with the Department of Computer Science, ILMA University Karachi, Pakistan.
Email: engineerqirat@gmail.com

Corresponding Author*

Abstract

This review paper critically assesses the range of machine learning (ML) approaches used in software defect prediction, with an emphasis on advances in Mathematical Modeling and Simulation. Software defect prediction (SDP) continues to be an important component of software project planning, having a major influence on time to delivery estimates, maintenance activities, and overall performance standards. Before software deployment, some defects need to be forecasted by organizations so during this phase the challenges that occur are investigated in the study to emphasize SDP significance in domains such as robotics, healthcare, aviation, and manufacturing. This review explores the SDP models in terms of evolution specifically highlighting ML classifier integration with static program metrics. Recognizing the possible limits of human feature engineering in missing vital information, the research examines previous SDP outcomes and provides adaptive strategies for future systems to identify anomalies. Machine learning approaches that include Random forest, Multi-layer Perceptron (MLP), K-Means, and support vector machines are reviewed. Further SDP model's current state is discussed and their effect on software quality is improved in the phase of maintenance.SDP models include the evaluation of metrics such as detection rate, false alarm rate, and precision. The review findings highlight the high accuracy achieved by K-Means and SVM (99.19%), K-Means and RF (97.76%), and K-Means and MLP (99.67%) in predicting defects and provide insights into ML technique's effectiveness about SDP.

Keywords: Software Defects Prediction (SDP), Machine Learning (ML), K-Means, Support Vector Machines Linear (SVML), K-Means, Random Forest (RF), Multi-layer Perceptron (MPL) Algorithm

INTRODUCTION

Software defects prediction (SDP) is an important area of study that focuses on finding problems in software programs and suggesting novel solutions to them. As software systems get more complicated, the requirement for easily maintained, top-notch and inexpensive software grows [1-3]. Prompt rectification can be facilitated by detecting the flaws early which will lead to better performance and improved reliability of software [4]. For large code bases manual code reviews are impractical and consume more time which makes automatic SDP algorithms critical in terms of managing finite resources effectively [5-6]. Software defect prediction has advanced significantly over the last three decades, with several methodologies categorizing software components as non-defect-prone or defect-prone, relationships, and predicting residual flaws in software systems. This study focuses on constructing software prediction models for defects based on previous failing data and software attributes in order to categorize modules and classes [7-8]. Error-prone areas are concentrated for testing due to which high product quality can be achieved within the budget and timelines [9-11]. Organizational performance can be improved by

analyzing, reducing, and identifying defects [12-14]. It helps to improve organizational excellence [15Customer can retain in organization by reducing defects [16]. Knowledge management and information retrieval can be enhanced by having software with zero defects [17]. The work stress is more on the employees of software development organization in pakistan [18]. Updated ICT and modern application also lead to reducing software defects [19-21]. Learning businesses have a track record of improving organizational operations via the use of high-quality programs, machine learning, and artificial intelligence, approaches [22-28]. Several prior research on SDP dealt with the vulnerability of software parts by analyzing code metrics [29]. Although several attempts to employ machine learning techniques, few have exhibited consistent dependability. Many firms in Pakistan recognize the possibilities of AI and ML technologies in operational improvement, yet they lag behind [30-31]. Procurement report [32] is one of the most current applicable case studies of Pakistani firms in the field of optimization via improved quality software applications and there are many other like acquisition reports [32], purchase orders [34], material delivery time analysis [37], routine report making [33], order costing analysis [39], Price Evaluation Report [36], planning report [35], procurement report [34], product mix & profit maximization [38], production plan [40], and comparative analysis of material and cost [42]. In the framework of optimization by improved quality software applications, modern applications of Pakistani institutions include hospital outpatient departments [43-48] Health Care Units of Pakistan and emergency units [49-50]. For software defect prediction, this work utilizes supervised and unsupervised learning approaches, including K-Means clustering and Support Vector Machines Linear, RF, and MLP algorithms for clustering, LR, and classification. These strategies have increased recalled, precision, f1-score, forecasting, preciseness, groupings, and classification algorithms, indicating that they will boost defect prediction accuracy.

LITERATURE REVIEW

Performance Analysis of Software Defects Prediction is the area of concern for cyber security professionals due to security threats and increasing phishing attacks [51-54]. Mathematical modelling, simulations, IoT, AI and ML are being used effectively to evaluate the performance of SDP [55-59]. DL and Industry 4.0 are also the recent developments in the techniques to improve the Cyber security and to safeguard the organizations' critical systems from phishing attacks [60-65]. Performance Analysis of software has been performed by many experts with various Mathematical modelling &simulations techniques [66-69]. Medical field is getting the remarkable results by using the machine learning techniques for the more accurate diagnosis & prediction of diseases at the individual and public level [70-75]. The systematic review of SDP models was performed by many researchers and the results of various models were compared [76-79]. The SDP models with ML & empirical assessment were critically evaluated by the researchers and proposed frameworks were developed by them for better results of Software Defects Prediction[9], [80]–[82]. Simulation can be used as an effective for SDP [83-85]. Numerous projects have been successfully implemented SDP by simulation tools & techniques [86-88]. Propagation neural network model, poisson regression, spiderhunt-based deep convolutional neural network classifier and discrete mycorrhiza optimization nature-inspired algorithm are used effectively researchers for SDP [89-92]. Hassan et al. achieved more than 99% accuracy on the dataset with an integrated approach for sentiment classification and information retrieval techniques [93]. Mathematical Modelling & Simulation is getting popularity for the prediction of software defects. The ROCUS, Ayesian networks, Petri nets, AHP and boosting approach are amongst the effective

Mathematical Modelling & Simulation techniques for SDP[94], [97-98]. Machine Learning is also getting popularity for predicting software defects and researchers consider it as effective techniques [99-102]. The recently completed software prediction projects are the quite evident of the fact that machine learning also proved its worth in the field of SDP [103-107].Deep Learning is an effective AI based tool for predicting software defects [108-110]. There are very few recently completed projects of software defects prediction projects by using deep learning technique but they have shown the remarkable results [111-115]. SVM is a type of supervised learning algorithm which is comparatively new machine learning tool in the field of SDP to solve classification problems [116-119]. Though there are very few recently completed projects of software defects prediction projects by using support vector machine technique but they have proved the effectiveness in SDP [120-121].K-means clustering can be used effectively to increase software defect prediction [122-124]. Researchers quoted the benefits & applications of K-means in the various fields to predict the software defects [125-128].Practitioners used Random Forest in SDP projects and mentioned its benefits [129-135]. A multilayer perceptron (MLP) is a misnomer for a feedforward artificial neural network, consisting of fully connected neurons with a nonlinear activation [136-138]. The recently completed software prediction projects are the quite evident of the fact that MLP also proved its effectiveness in the field of SDP [139-142].

There is the growing need of more accurate Software defects prediction (SDP) from modern complex systems to daily routine systems. SDP is also used in the critical systems of aviation, healthcare, manufacturing, and robotics where the prediction of accurate defect before software deployment is actually very crucial for estimating delivery time, maintenance efforts, and ensuring quality expectations. Despite many developments still many organizations face difficulty in forecasting the accurate defect before software deployment. SDP enhances software quality by spotting potential defects in the upkeep phase.The topic of discussion for SDP are machine learning and artificial intelligence and several mathematical modelling simulation . The current models of SDP rely on static program metrics for machine learning classifiers, but manual feature engineering may miss vital information impacting defect prediction accuracy. The objective of this review is to compare the previous models of SDP and their results then aims to develop methods by adapting to future anomaly detection techniques. To achieve this, it is crucial to explore various prediction models and ML methods that can accurately predict software defects outcomes using the available dataset. The performance of these models needs to be evaluated and measured. This research aims to address these challenges by utilizing the selected dataset and analyzes machine learning algorithms's performance in developing prediction models.Division of this review is done in four parts.Part one provides an introduction to the research study. Part two is based on the discussion to the related work in this area of research. Part three focuses on the results and discussion of various algorithm combinations used for predicting the software defects. Lastly, the concluding section presents a statement on the most efficient algorithm combination.

BACKGROUND

Classification, Regression, And Clustering In Machine Learning

Based on simiilarity federation mechanisms are categorized into subclasses and discrete groups involving classification in machine learning. Classification is the

methodical method of breaking systems into recognized groupings and subcategories based on their similarities. Many researchers employed Machine Learning techniques such as classification, , and clustering, and regression to analyze and examine illnesses [143-147]. Three common methods for categorization are linear classification, naive Bayes classification, and linear regression. labeled and organized data is where classification is applied. Various classification methods are utilized in several operations as shown in Figure 1.

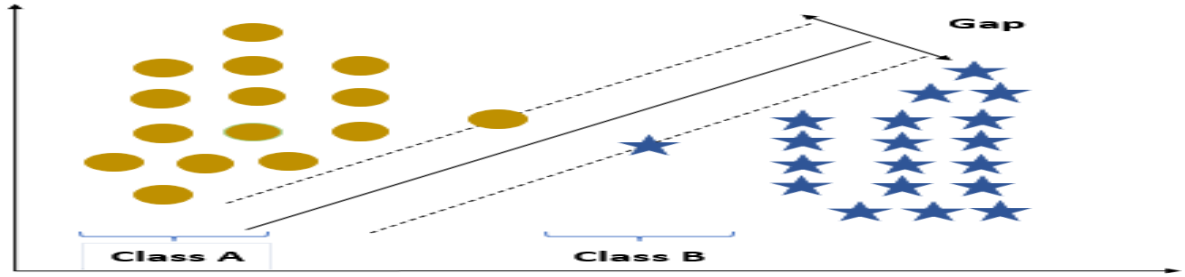

Figure 1.
Overview of Classification [148]

Unsupervised and supervised methods are required differently in nonlinear and linear regression because dependent and independent variables in every model have diverse natures for interaction. To perform regression tasks these methods are used.

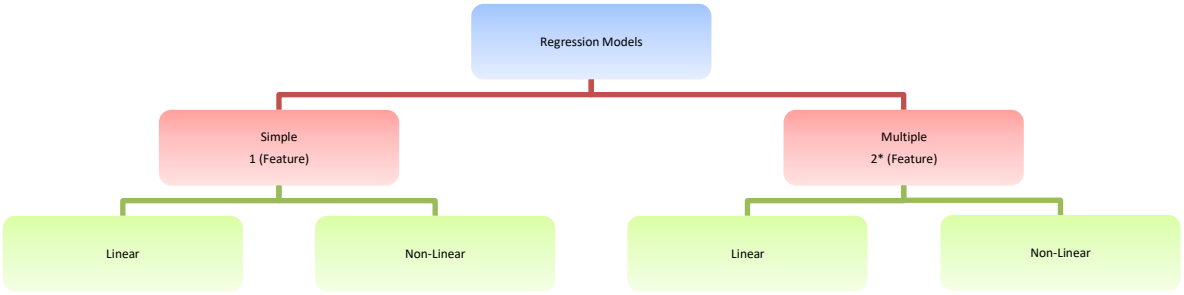

Figure 2.
Regression Models [161]

Figure 2 demonstrates how algorithms for machine learning use a variety of regression qualities, as well as unorganized and organized information. Methods for machine learning make use of data that is both unstructured and structured, as well as a wide range of regression characteristics. The regression model's first and second property includes nonlinear and linear regression. Unsupervised learning is clustering that consists of many uses in different sectors. Based on the data machine when realated pieces of information go through processing and isolation in groups is called a cluster. Several clusters are observed in Figure 3.

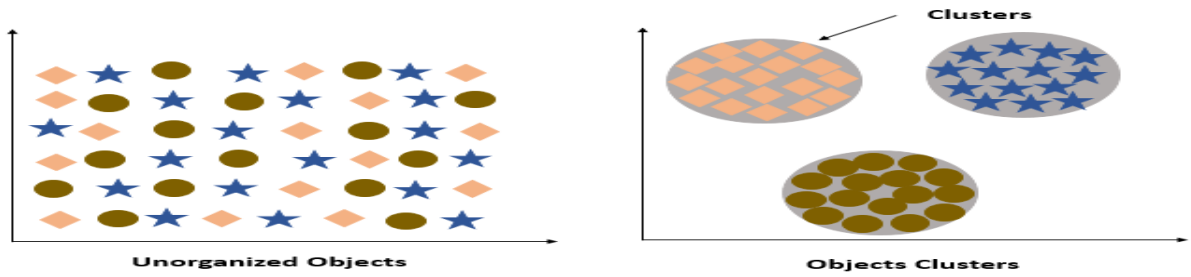

Figure3.

Clustering [149]

PROPOSED METHODOLOGY

This section of the review paper provides an analysis of the process of developing a work structure to predict software defects (SDP) [161].

- Google drive is used to retrieve dataset from it in the first step
- Cleaning, standardization using (MinMaxScaler), feature extraction methods like (CountVectorizerandTfidfTransformer), and standardization using (MinMaxScaler) are the procedures through which data is passed
- • Standardisation necessitates the development of a framework for converting changing amplitude and frequency, like (0.98671539), followed by a standardisation assessment to obtain the result.
- Accuracy of the model, recall, f1 score, and precision can be improved by employing K-Means clustering which is machine learning technique called unsupervised
- Testing data and training data is produced by the splitting of data , the size of the test data is set at 0.25%, and of training data is 0.75% for the implementation of this method
- Lastly, the finalized model is built using the MPL, SVML, and RF algorithms.

Figure 4 depicts the framework of software defects prediction (SDP), which provides a clear view on the study's scope as well as a quick explanation of the structure for the work breakdown.

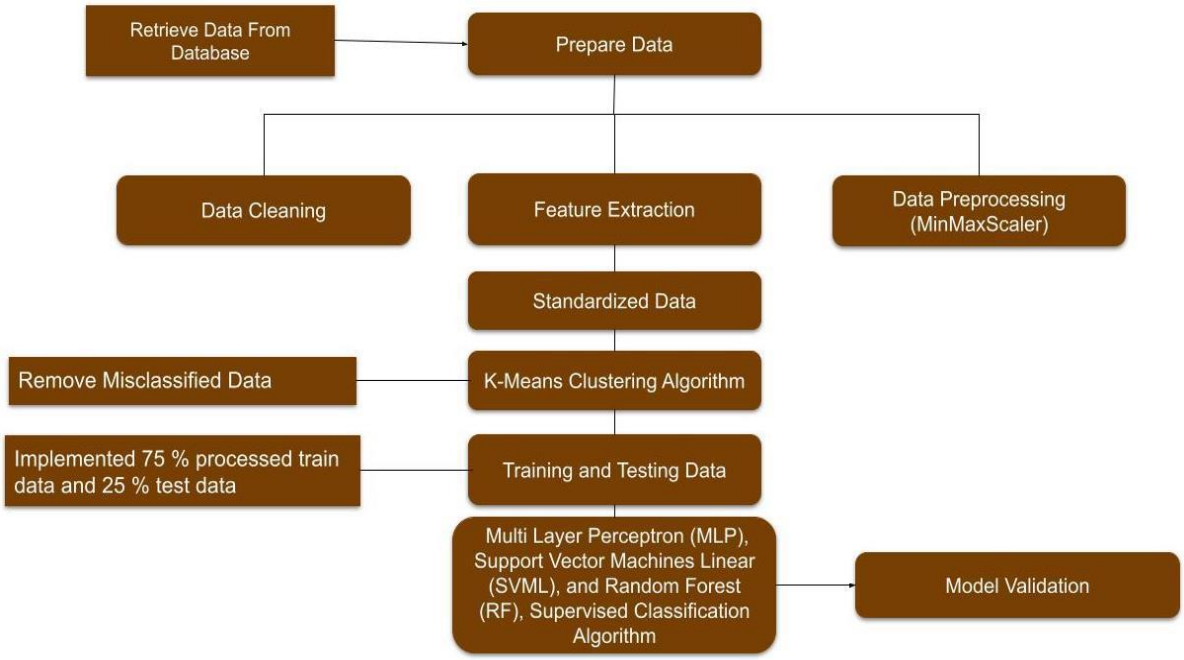

Figure 4.
Proposed algorithm for Software Defects Prediction (SDP) [161]

The flowchart shows that data is retrieved from the base, by the usage of data cleaning methods preprocessing of the data is done through normalization, and finally to validate the model seventy-five percent of the data is trained and twenty-

five percent is tested using classification, and clustering. Data processing and classification are two different sections of the algorithm which are used.

Preprocessing

In the initial stage, data is prepared for cleaning by applying clustering techniques to extract important information. For this purpose two known methods are utilized first is Kmean clustering which is defined below and classification is the stage where processed data is further manipulated.

Performance Analysis

The language mainly used for scripting purposes is called Python, a range of domains including databases, programming, web development, and machine learning consists of its applications. This review analysis studies that the Anaconda Navigator ->Jupyter Notebook GUI framework is used along with Python is utilized as well to implement several algorithms such as SVM, RF,MLP, and k-means and linking of datasets is done through it. This set of data is about software defects prediction (SDP), which involves determining the likelihood that a program has flaws based on software bugs. 10855 observations(rows) and 22 attributes (columns) are in the dataset. Three distinct programs are run on the same dataset. Multi-layer Perceptron (MPL) and K-Means are utilized first, Support Vector Machines Linear (SVML) and means are utilized second, and Random Forest (RF) and k means are utilized in the third program. The following configuration is used to execute the programs on the personal computer:

- 2nd Generation Intel Core (TM) i5-2520M CPU operating at 2.50 Gigahertz is installed in the computer
- 4GB RAM is utilized
- The 64-bit operating system particularly window 10 is running on PC
- The hard disk is 500 GB

DATA COLLECTION

The Software Defects Prediction (SDP) dataset was collected from Kaggle, a website that hosts numerous machine learning datasets. **Shadab & Naseem** previously used this particular dataset in their research [161]. It comprises 10,885 instances or observations, each with 22 attributes representing the specifications of software applications and their measures related to SDP. The target class in this dataset represents the status of each outcome, with a total of 5,427 not-defects software bugs and 5,458 defects software bugs The specifications of software applications are represented by 10,855 observations and 22 attributes. The status of each result is represented by the target class, with a total of defective software bugs up to 5458 and no defect software bugs up to 5427 [151]. SDP used the dataset in the research to predict software defects so it consists of features and parameters having a concise summary presented in Table 1.

Table 1.
presents the Original Dataset Used for Predicting Software Defects [161]

Parameters of the dataset	Characteristics of SDP
loc	No of statements of program
v(g)	Complexity of cyclomatic
ev(g)	Intrinsic complexity

iv(g)	Design complexity
n	No of operators and operands
v	Space amount
l	Length of the program
d	adversity
i	Intellect
e	Exertion
b	Count of errora
t	Prediction of time
lOCode	Lines counting
lOComment	Sum of somment lines
lOBlank	Whitespace lines totaling
lOCodeAndComment	comments and code lines counting
Uniq_Op Unique	operators that are different
Uniq_Opnd Unique	Operands that are different
Total_Op	total count of operator
Total_Opnd	total count of operands
branchCount	flowchart branch counting
defects	report of defects

The goal of this study is to determine the procedures required for anticipating software issues. It is analyze collecting the proper information from the data we have, preparing it, and making any required changes to make it perform better. We're also scrutinizing crucial areas, dealing with complicated challenges, and assessing the system's effectiveness after all of these phases. The initial step is to collect details gathered from the data. Following that,data is sanitize and make a few modifications, such as standardizing and normalizing it. Table 2 shows the cleaned-up and modified data we ended up with. In addition, Figure 5 shows a visual depiction of complex information that does not use the K-Means approach..Purple value representation is of X value and yellow circle representation is of Y

Table 2.
shows the dataset for predicting software defects, which has been processed [161]

22-Dimension
array ([[0.36223789, 0.60325949, 0.25972736, ..., 0.04290384, 0.99847326, 0.79664566],
[0.20296517, 0.47553557, 0.51124005, ..., 0.01224384, 0.39541578, 0.66811618],
[0.17949324, 0.12738392, 0.65493002, ..., 0.35573798, 0.03057093, 0.34464949], ...,
[0.9456746, 0.98671539, 0.38383904, ..., 0.52999682, 0.31716936, 0.70528904],
[0.13678812, 0.82731781, 0.71771077, ..., 0.02882109, 0.29340566, 0.69901713],
[0.69547178, 0.63604136, 0.42970602, ..., 0.64185376, 0.03466157, 0.37666046]])

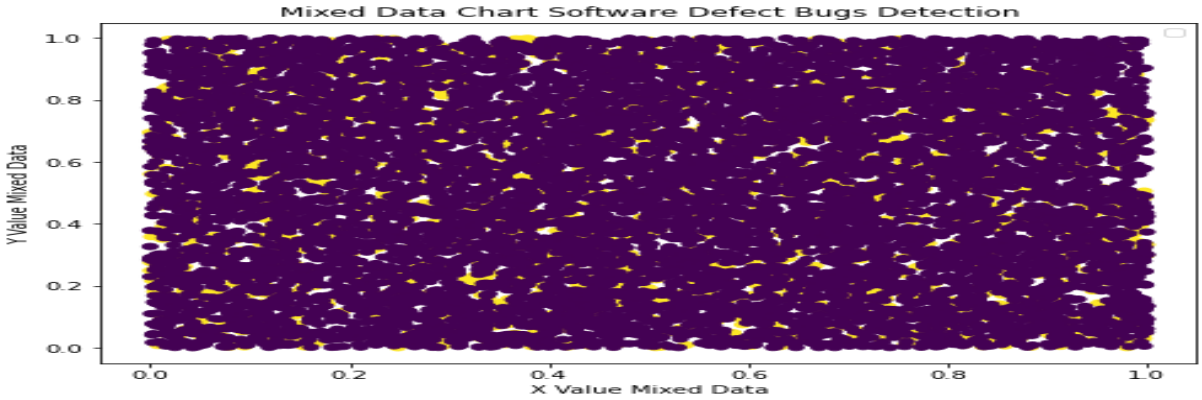

Figure 5.
Mixed Data Chat Software Defects Prediction (SDP) [161]

K-Means Clustering Algorithm

Clusering is a technique commonly known as unsupervised learning which can be used and adopted in many fields. To identify a cluster each cluster is assigned a distinct number for identification purposes.The unsupervised machine learning technique called Kmeans classifies the data into two categories such as mixed and structured. Randomly average values are selected from the dataset which serves as an initial point for groups. The value at the intermediate location is calculated to enhance clustering [152]. Kmeans fundamental principles of the algorithm are as follows:

- In clustering process suitable clusters (k) will be identified
- Before the calculation dataset is sorted and k values are randomly selected to be the centroid
- Identify clusters after there is no change in centroid still the approach to clustering the datais identical
- Number of patterned lengths between the data points and each centroid is calcualted
- The points closest to cluster will be allocated
- Obtaining cluster centroid all the points assigned to every cluster will be calculated
- The clustering process is completed

Each program was classified in the dataset by using scientific methods and metrics such as Hamming measures and Euclidean, Manhattan

Euclidean	$\sqrt{\sum_{i=1}^{k}(xi - yi)2}$ **(1)**
Manhattan	$\sum_{i=1}^{k} \lvert xi - yi \rvert$ **(2)**
Minkowski	$\left(\sum_{i=1}^{k}(\lvert xi - yi \rvert)q\right)1/q$ **(3)**

The standard collection is employed in this processing to generate mixed data representations using a pre-processing approach. K-Means has been employed to sort and analyze enormous datasets, making the results easier to grasp and removing duplicate information. By employing the technique of clustering, we managed to identify two separate groups and provide a probability score to every piece of data so as to ascertain its inclusion in a certain cluster. This approach produced an element of the matrix that depicts the relationship among each of the samples and its associated cluster. The approach used involves the use of clustering techniques, notably the use of the K-Means algorithm alongside centroid clustering values. This approach is used on a dataset with 22 dimensions and binary-class data. Each point of data connects to a centroid based on the inter-point distance, with a stronger relationship indicating closer closeness to the data centroid. The Software Defect Prediction (SDP) dataset has 22 dimensions with features important to forecasting software problems and a property cluster number identification. A simple summary of the K-Means Clustering centroid value has been provided. Both Figures 6 and 7 have also been included to graphically display the clusters and line charts showing the sum total squared error values for the 22-dimensional binary-class datasets. The graphical representation is displayed after unstructured material has been converted into an organized format.

K-Means Clustering Centroid Value [161].

array ([[0.49914726, 0.49098853, 0.5040306, 0.48515271, 0.51490666, 0.51906228, 0.47665096, 0.4984902, 0.49849351, 0.51083994, 0.50353293, 0.50095931, 0.50579481, 0.5030984, 0.49457925, 0.750223, 0.50110969, 0.49787315, 0.50347232, 0.49546553, 0.48247945, 0.48096822],

[0.50335415, 0.50101048, 0.48892057, 0.51358704, 0.48948397,0.48769329, 0.51316378, 0.50301923, 0.50445169, 0.49481033, 0.48963746, 0.49711861, 0.49109309, 0.49119229, 0.51433448, 0.25323482, 0.50181001, 0.50683171, 0.49787394, 0.50327462, 0.51617455, 0.51853753]])

Table 3.
K-Means Two Clusters Pre-processed Software Defects Prediction (SDP) Dataset [161]

array ([[0.36223789, 0.60325949, 0.25972736, ..., 0.04290384, 0.99847326, 0.79664566],
[0.20296517, 0.47553557, 0.51124005, ..., 0.01224384, 0.39541578, 0.66811618],
[0.17949324, 0.12738392, 0.65493002, ..., 0.35573798, 0.03057093, 0.34464949] ...,
[0.9456746, 0.98671539, 0.38383904, ..., 0.52999682, 0.31716936, 0.70528904],
[0.13678812, 0.82731781, 0.71771077, ..., 0.02882109, 0.29340566, 0.69901713],
[0.69547178, 0.63604136, 0.42970602, ..., 0.64185376, 0.03466157, 0.37666046]])

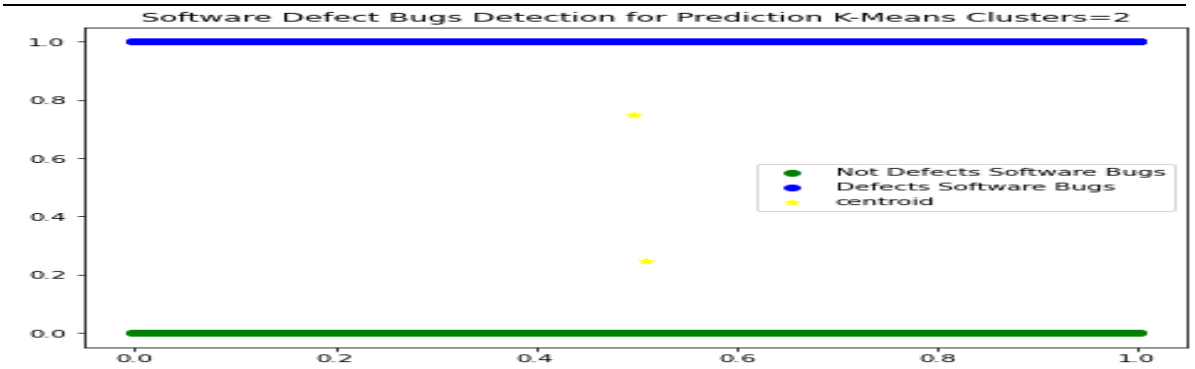

Figure 6.
K-Means Two Clusters Software Defects Prediction (SDP) [161]

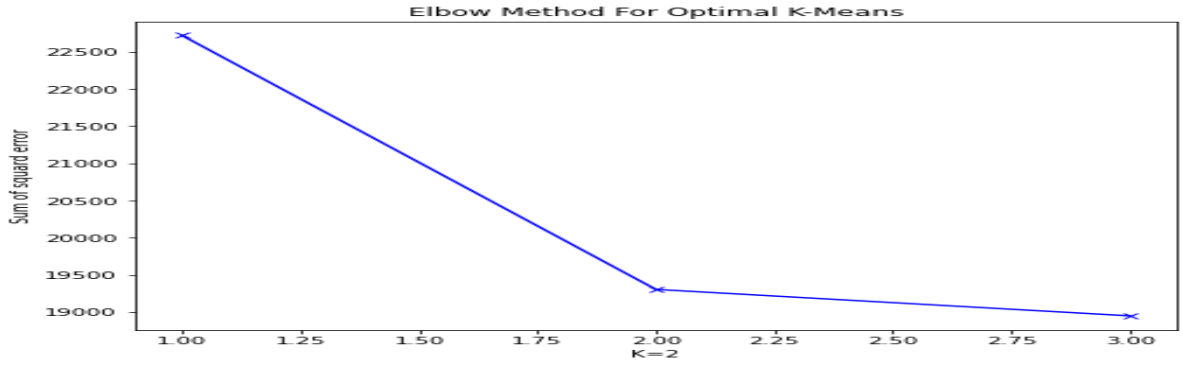

Figure 7.
K-Means Sum of Squared Error Line Chart [161]

To evaluate the effectiveness of CFD using both clustering methods, precision, recall, and f-measure are used. A concern score is determined by measuring how much the system deviates from the standard, and the result is classified as valid, suspicious, or illegal.

CLASSIFICATION

Observed data is categorized by the use of training data in the classification algorithm which is a type of supervised learning. Grouping observed data into sections and categories is a process called classification. To evaluate which classifier performs well with the dataset for this reason many classifiers are tested.

MULTI-LAYER PERCEPTRON'S (MLP) ALGORITHM

The Multilayer Perceptron (MLP) is a sophisticated optimization technique containing many perceptrons. Three layers are there in MLP : an input layer that takes input data, an output layer that makes verdicts or estimations on the basis of the input, and a random assortment of hidden layers that provide MLP processing power. The MLP may approximate any continuous function by adjusting the amount of hidden layers. [153], [154]. In the cases in which datasets are not conditionally independent, to overcome this challenge MLP employs participants to develop prediction and MI models with complex and flexible framework. This strategy, which is commonly employed in supervised learning, overcomes problems associated with challenging data patterns and promotes scientific improvements in a variety of domains. Based on the principles of clasificaation different approaches are constructed such as sigmoid, linear, cost linear, non linear regression.

Sigmoid	$\boldsymbol{S\,(z) = \frac{1}{(1\ +\ e^{-z})}}$	**(4)**
Linear Regression	$\boldsymbol{y\ =\ e\text{^}(b0\ +\ b1 * x)\ /\ (1\ +\ e\text{^}(b0\ +\ b1 * x))}$	**(5)**
Cost Linear Regression	$\boldsymbol{(Cost\ (h\theta(x), y))\ = -log\ (h\theta(x)), if\ y = 1\ and}$ $\boldsymbol{(Cost\ (h\theta(x), y)) = -\ log\ (1 - h\theta(x)), if\ y = 0}$	**(6)**
Nonlinear Regression	$\boldsymbol{Y\ =\ f(X, \beta) + \varepsilon}$	**(7)**

The MLP algorithm is followed as:

- Identical to perceptron, the processes of MLP input parameters and data between the hidden layer and input that undergoes partial derivatives giving the result as a hidden layer that is not elevating
- To transfer the computed output to the visible layer activation functions are used such as rectified linear units, tanh, and sigmoid
- After the activation function generates the anticipated output in the visible layer, the corresponding partial derivatives are extracted and transmitted to another layer within MLP.
- until the final output is achievd Step two and step three are repeated iteratively
- The acquired estimations function as the outcome for producing results using either a feed-forward approach employing the selected activation methods for Multilayer Perceptron (MLP) in the case of working with training data, or a choice based on the outcomes in the case of working with testing data.

During the training phase, the Multilayer Perceptron (MLP) anticipates labels for past data and attempts to match these forecasts with the labels that are used to forecast values in fresh data. Figure 8 depicts the outcome of the MLP-generated confusion matrix.

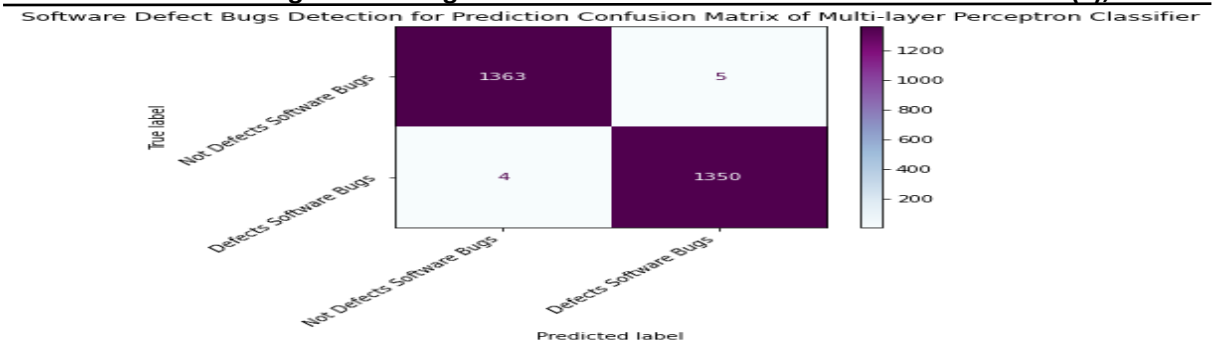

Figure 8.
Confusion Matrix Multi-Layer Perceptron's (MLP) Algorithm [161]

At the time of review the confusion matrix was defined as [[A B] [C D]], where

- A: shows that negative instances count is predicted correctly
- B: shows positive instances count were incorrectly forcasted,
- C: Representation of incorrectly predicted instances are called negative
- D: representation correctly predicted instances are called as positive.

If believe that Perceptron's Multilayer (MLP) model is applicable for the circumstance, the confusion matrix let us determine the expected labels for the identification and forecast.

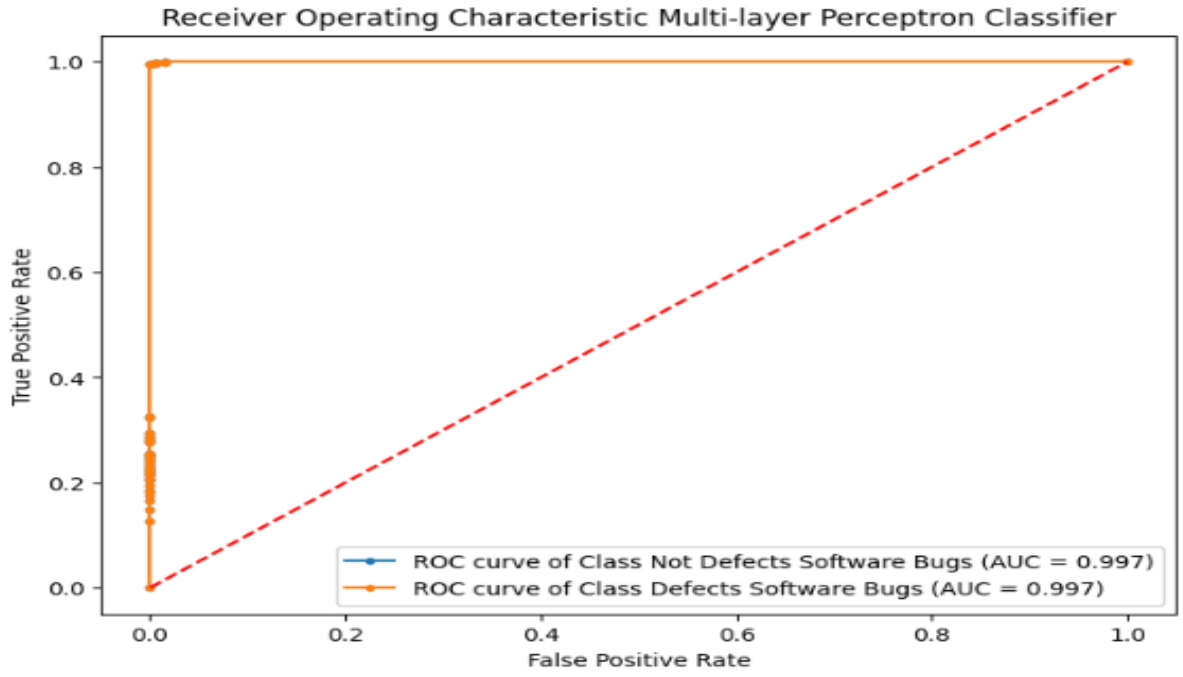

Figure 9.
presents the Receiver Operating Characteristic (ROC) Curve for Multi-Layer Perceptron's (MLP) [161]

The results of using the Multi-Layer Perceptron's (MLP) algorithm on a synthetic dataset can be visualised through the Receiver Operating Characteristic (ROC) Curve, as shown in Figure 9. In this study, we utilised the concept of ROC curves to evaluate the accuracy of our model's predictions for user reviews ratings. This analysis allows us to better understand prediction patterns and improve the overall precision of our estimation method.

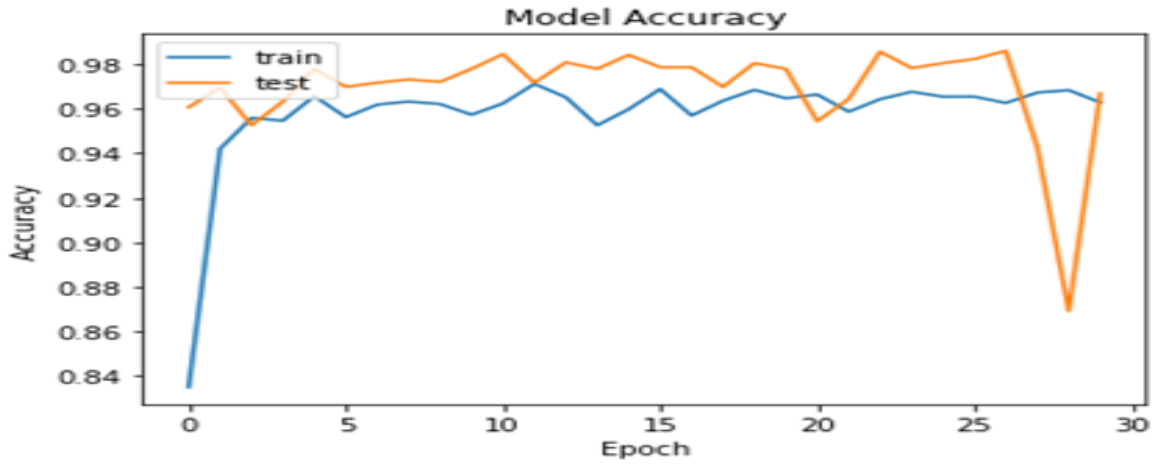

Figure 10.
Model Accuracy Multi-Layer Perceptron's (MLP) Algorithm [161]

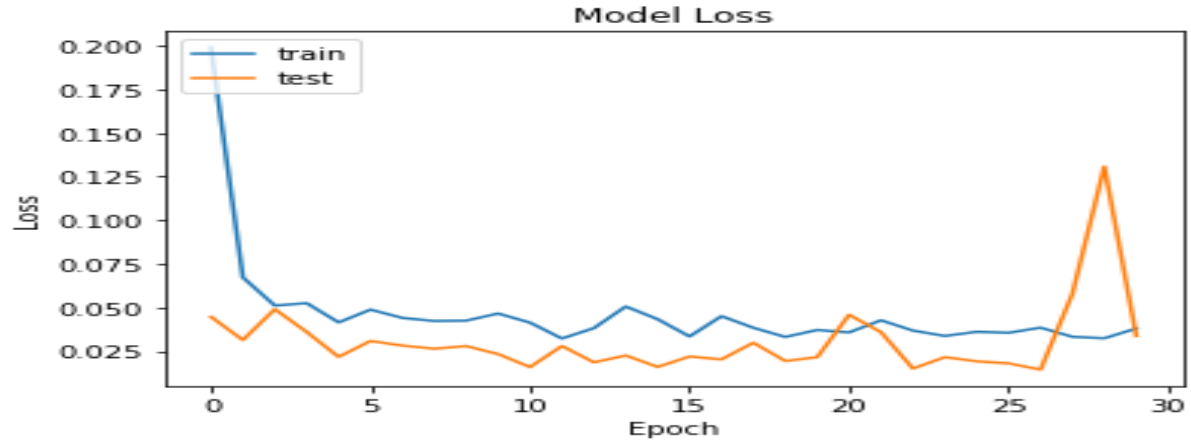

Figure 11.
Model Loss Multi-Layer Perceptron's (MLP) Algorithm [161]

The performance of the model is evaluated to forcast software defects the Multi-Layer Perceptron's (MLP) Algorithm is employed to assessed loss metrics and accuracy. Due to this approachthe aim is to enhance the accuracy for predicitions while the pattern of predicting software defects can be fulfilled consisitently. The model's loss metric and accuracy is shown in figure 9 and 10 which were the important indicators of the and analysis.Particularly train accuracy of MLP model was 0.97 while test accuracy was 0.97 (Figure 9), along with the train loss was 0.040 and the test loss was 0.40 (Figure 10).

Support Vector Machine Linear (Svml) Algorithm

The regression and classifiacation tasks used by the supervised learning approach called as SVML.The working of this algorithm is followed in away where mixed clases are partitioned on a graph into separate groups called as Maximum Margin Higher dimensional space.Among two categories the identification of smallest piece of data is done by the SVML and different mathematical techniques are applied such as sigmoid,RBF, and polynomial radial to achieve separation. For different classes seaparte data points are utilized to separate decision bounary support vectos specifically,and two nearest points will be called as support vectors [155-156]. The SVM technique utilizes mathematical classification and regression functions such as Linear SVM, Non-linear SVM, and Kernel function.

Table 5.
SVM Mathematical Equations

Linear SVM Model	***xi.xj***	**(8)**
SVM Non-Linear	***ϕ (xi).ϕ (xj)***	**(9)**
Function of Kernel	***k(xi.xj)***	**(10)**

Set of critical steps are followed in SVML algorithm which are :

- Initially suitable identification of hyperplanes is done that increases margin among different classes and separate data
- Misinterpretation is prevented and non linear separate data is handled employing different techniques.
- Immediate selections identification of surface areas become easy when the data is change into dimensional space.

After training, the system can predict classifications for new as well as old values. The objective is to get these forecasts as near to the real labels as feasible. The ensuing matrix of confusion for the SVML forecasts is shown in Figure 12.

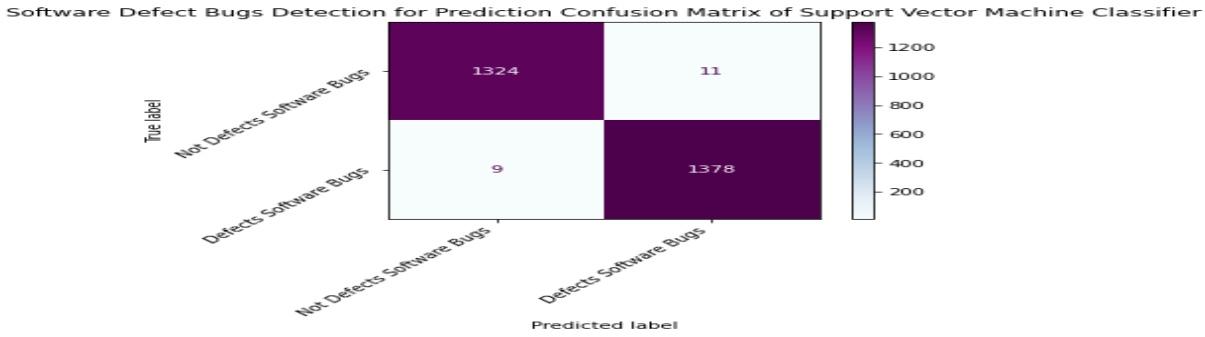

Figure 12.
Confusion Matrix Support Vector Machine Linear (SVML) Algorithm [161]

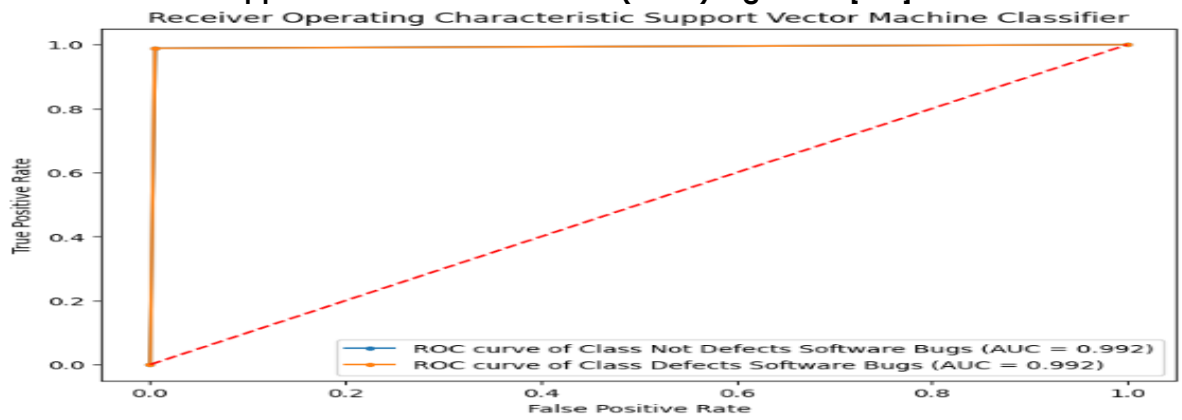

Figure 13.
Confusion Matrix Support Vector Machine Linear (SVML)Algorithm [161]

ROC analysis is a technique for determining the manner in which a model of a classifier works when the classification threshold is modified. This study is closely connected to cost/benefit research, which considers the costs and benefits of choices. The SVML ROC curve, shown in Figure 13, depicts the results of the support vector machine with a kernel that is linear at various threshold settings.

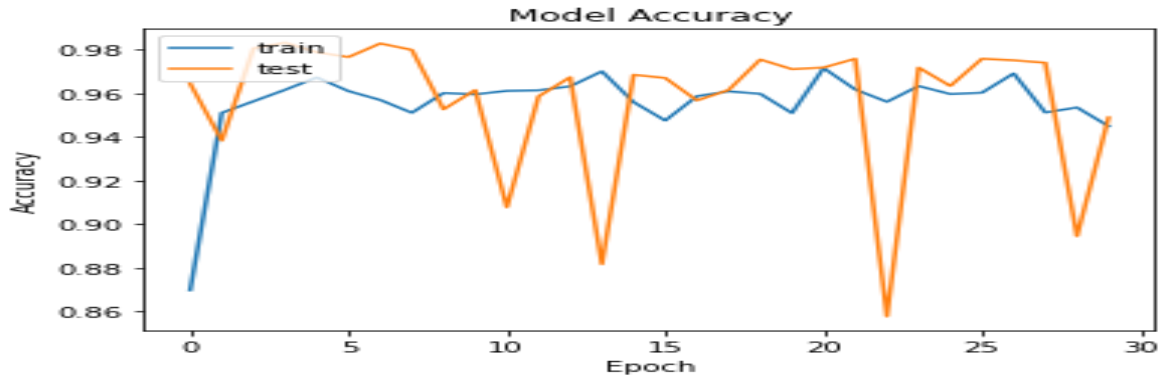

Figure 14.
Model Accuracy Support Vector Machine Linear (SVML) Algorithm [161]

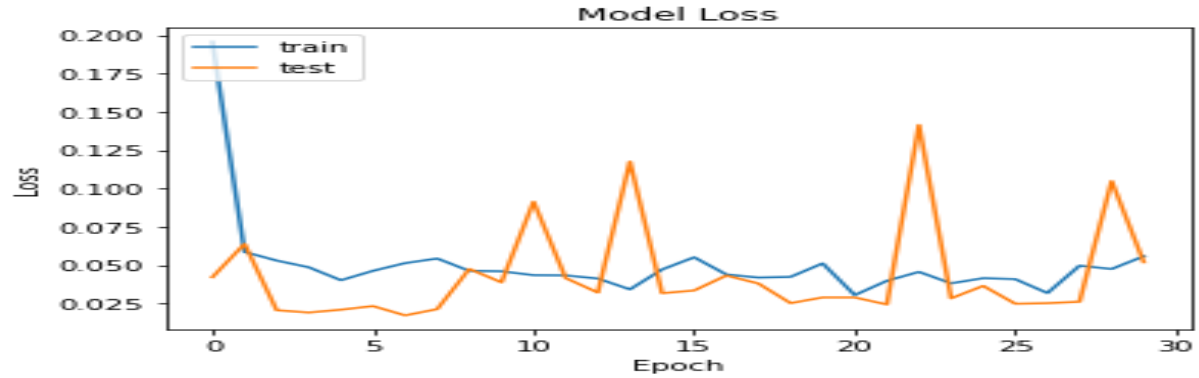

Figure 15.
Model Loss Support Vector Machine Linear (SVML) Algorithm [161]

The description outlines the performance evaluation of the Support Vector Machine with Linear (SVML) algorithm on a specific dataset, as illustrated in Figures 14 and 15. As per the description, Figure 14 indicates that the SVML algorithm achieved an accuracy of 0.96 on both the training and testing data. This implies that the algorithm successfully classified 96% of the data points in both sets. In Figure 15, the model loss for the SVML algorithm was noted as 0.050 on both the training and testing data. Model loss serves as an indicator of the algorithm's ability to predict the correct class for each input, with lower model loss values signifying better performance. Consequently, the statement suggests commendable performance of the SVML algorithm related to accuracy and model loss measures for the dataset given.

RANDOM FOREST (RF) ALGORITHM

The Random Forest (RF) technique stands as a machine learning method designed to address classifier issues, employing a combination of classifiers to form a sophisticated problem-solving system with diverse classification approaches. By amalgamating multiple categories, RF adeptly deals with intricate problems, enhancing system efficiency. Built on predictions derived from classification trees, RF assesses their effectiveness through assumptions and aggregates outcomes from numerous trees. As the number of nodes increases, the output improves, mitigating the constraints of a Decision Tree (DT) [157-158].

The RF process initiates by randomly selecting observations from the available data. Subsequently, a tree structure is created for each instance, and outcomes for each tree structure are generated. During this phase, each result undergoes determination.

Ultimately, the prediction outcome with the highest probability is chosen as the preferred result.

Additionally, the RF Algorithm incorporates various mathematical functions or formulas, including Gini (Coefficient, Index, or Ratio), Entropy, and Mean Squared Error (MSE) [159]. These procedures serve as illustrative examples for evaluating the approach.

Table 6.
Random Forest Mathematical Equations

Mean Squared Error (MSE)	$\frac{1}{N} = \sqrt{\sum_{i=1}^{N} (xi - yi)^2}$	**(11)**
Gini Coefficient	$Gini = 1 \sum_{i=1}^{C} (p_i)^2$	**(12)**
Entropy	$Entropy = \sum_{i=1}^{c} -p_i * log_2(p_i)$	**(13)**

We applied the RF technique to our dataset and assigned labels to the previous data values. This helped us predict the value of the data. When we utilise the RF approach to ensure that our predictions align with the categories during preparation, the results are shown in the matrix in Figure 16.

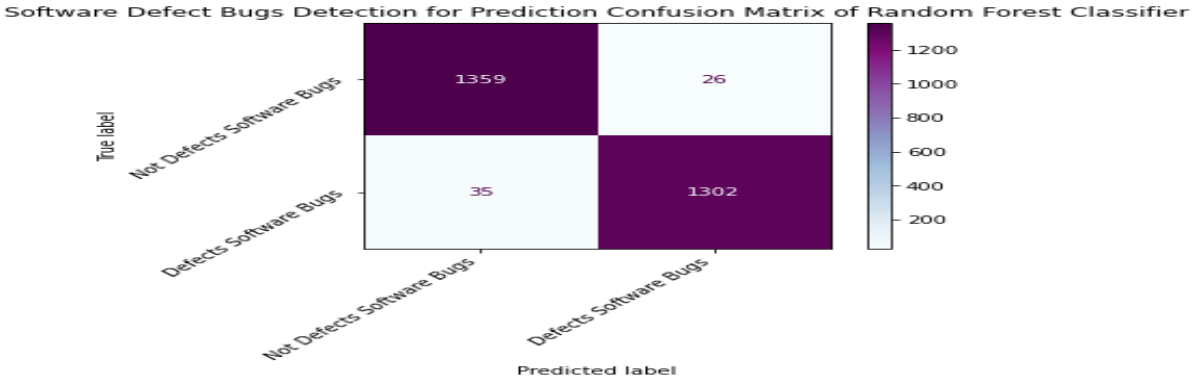

Figure 16.
Confusion Matrix Random Forest (RF) Algorithm [161]

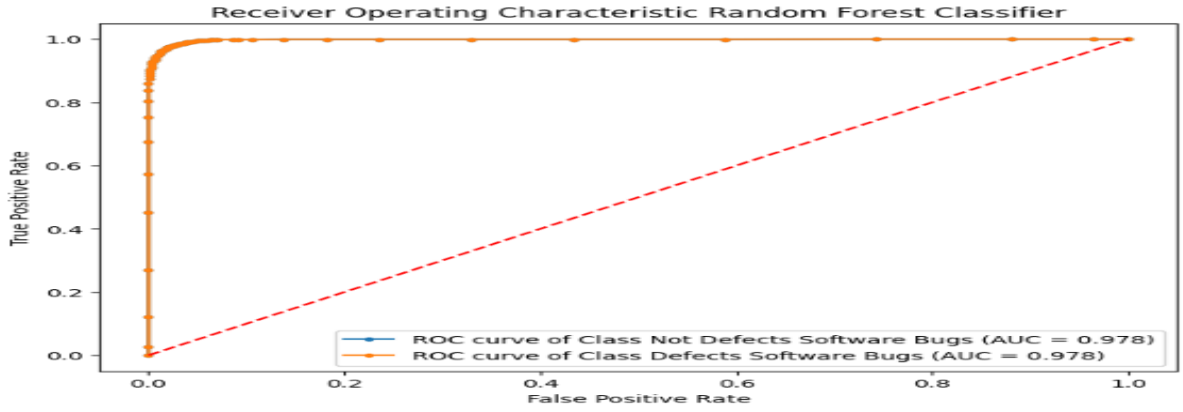

Figure 17.

Random Forest (RF)Receiver Operating Characteristic (ROC Curve) [161]

The ROC evaluation serves as a tool for evaluating a classifier model's systematic performance while its discriminating threshold is changed. This approach is strongly related to cost-benefit studies on rational choice making. Figure 17 depicts the curve's outcome.

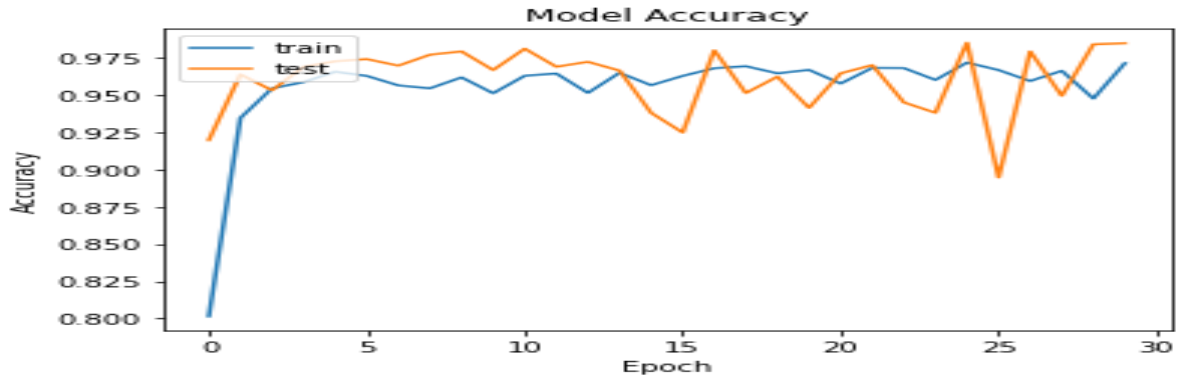

Figure 18.
Model Accuracy Random Forest (RF) Algorithm [161]

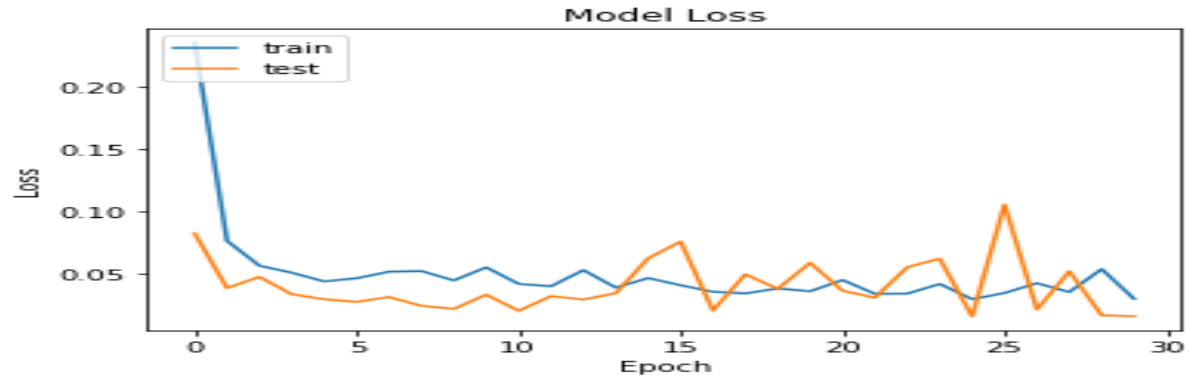

Figure 19.
Model Loss Random Forest (RF) Algorithm [161]

Random forest achieved the accuracy depicted in figure 18,which shows that testing data was 0.976 and training data was 0.975.The model loss resulted from RF indicated that the loss for testing data was 0.03 and for training data it was 0.04

RESULTS AND DISCUSSION

In this review article, we look at the practical uses of machine learning, a flexible method that allows algorithms to solve problems without explicit programming. While the use of deep learning is the present apex of machine learning, classical machine learning approaches remain vital in a range of applications related to industry due to advances in procedures, computing capacity, and accessibility to large datasets.

Our research presents a unique strategy for forecasting and identifying software defects that combines machine learning and deep learning approaches, employing data from prior software defect incidences. The study examines the characteristics of people who have experienced software flaws as well as the specific sorts of problems they are prone to encounter. To improve the reliability of software defect detection, we combine K-Means, Multi-layer Perceptron (MPL), Support Vector Machines Linear (SVML), and Random Forest (RF), with K-Means playing a key role in each combinations.

The most exact technique combination is attained by combining K-Means with Multi-layer Perceptron (MPL), gaining first place. The use of K-Means with Support Vector Machines Linear (SVML) and K-Means with Random Forest (RF) come in the 2nd and 3rd place, respectively. Tables 6 and 7 [161] provide a complete overview of correctness and other performance criteria for every combination.

Table 4.
Displays the Accuracy of Models that use a Combination of Algorithms for Predicting Software Defects [161]

Hybrid Algorithm	Accuracy of Algorithms
Mini-Batch K-means **[156]**	63.57%
Perceptron **[156]**	71.87%
PAC **[160]**	77.53%
GNB **[156]**	81.50%
KNN **[156]**	82.82%
QDA **[156]**	83.02%
GMM **[156]**	83.26%
LGBM **[156]**	85.99%
ET [**156]**	87.76%
XGBoost **[156]**	88.14%
RF **[156]**	88.18%
MVC **[20]**	88.27%
STC **[156]**	88.63
K-Means, Random Forest (RF) **[161]**	97.75
K-Means, Support Vector Machine (SVM) **[161]**	99.19
K-Means, Multi-layer Perceptron (MLP) **[161]**	99.66

Table 5.
Combination of Algorithms Parameter Score for Software Defects Prediction (SDP) [161]

S/No.	Parameter Score	K-Means, Random Forest Algorithm	K-Means, Support Vector Machine Algorithm	K-Means, Multi-layer Perceptron Algorithm
1	Precision	0.97765704	0.99193050	0.99669192
2	Recall	0.97752471	0.99192200	0.99669540
3	F1-Score	0.97758017	0.99191769	0.99669353
4	Sensitivity	0.98122743	0.99484915	0.99634502
5	Specificity	0.97382198	0.98899486	0.99704579

The accuracy is achieved by the combination of Multi-layer Perceptron (MLP) and K-Means is high as evident from the findings shown in figure 20 and 21.The second ranking is of SVML and K-Means and the third ranking is given to Random forest and K-Means

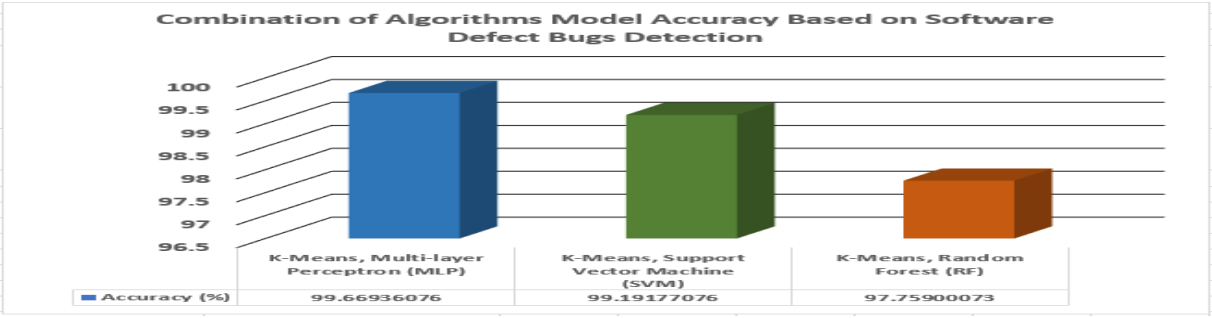

Figure 20.

Combination of Algorithms Model Accuracy Software Defects Prediction (SDP) of Prediction [161]

Based on the data, the graph demonstrating accuracy levels also demonstrates that the pairings' forecasts are at their best.

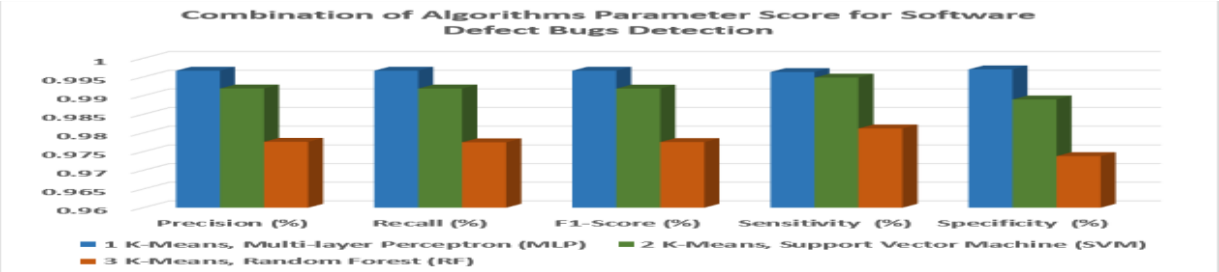

Figure 21.

Combination of Algorithms Parameter Score Software Defects Prediction (SDP) [161]

Based on our requirements, we can alter or limit the level of precision. Parameter Score Precision, Recall, F1-Score Sensitivity, and Specificity, for example, are now reaching ideal accuracy.

CONCLUSION

This comprehensive review concludes the most current development in the field of Software Defects Prediction (SDP) models using Mathematical Modelling & Simulation approaches in this thorough study. Many important industries, including aviation, healthcare, manufacturing, and robotics, rely on defect-predicting software to improve efficiency. The difficulty of precisely projecting errors prior to software deployment constitutes a major worry for many businesses. Our recommendation emphasizes SDP's ongoing importance as a promising research field. Despite several research using machine learning approaches over the last three decades, none have consistently proved dependability. SDP develops as an appealing study subject, concentrating on discovering defects in software systems and presenting novel solutions. As software becomes more integrated into business and social life, the demand for easily maintained, excellent, and inexpensive software grows.

As noted, early fault identification has a favorable influence on quick repair, resulting in improved software dependability and performance. While existing SDP models for machine learning classifiers rely on static program metrics, our work highlights the potential limits of human feature engineering in neglecting critical information impacting defect prediction performance. Exploration of previous SDP results lays the groundwork for future study, with the goal of developing approaches that respond to evolving systems for detecting anomalies.

Our research thoroughly evaluates several SDP techniques, such as the K-Means methodology, Support Vector Machines Linear (SVML), Random Forest (RF), and Multi-layer Perceptron (MLP) algorithms, as well as a comprehensive overview of existing SDP models. SDP models offered are rigorously evaluated utilizing measures such as false alarm rate, accuracy, and detection rate. The findings show that K-Means and MLP achieve 99.67% accuracy in defect prediction, K-Means and SVML get 99.19% accuracy, and K-Means and RF achieve 97.76% accuracy. This not only reinforces the need of SDP investigation, but also highlights the efficacy of particular models in tackling the difficulties of defect prediction.

DECLARATIONS

Acknowledgement: We appreciate the generous support from all the contributors to the research and their different affiliations.
Funding: No funding body in the public, private, or nonprofit sectors provided a particular grant for this research.
Availability of data and material: In the approach, the data sources for the variables are stated.
Authors' contributions: Each author participated equally in the creation of this work.
Conflicts of Interest: The authors declare no conflict of interest.
Consent to Participate: Yes
Consent for publication and Ethical approval: Because this study does not include human or animal data, ethical approval is not required for publication. All authors have given their consent.

REFERENCES

A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, and K. Kifayat, "A comprehensive survey of AI-enabled phishing attacks detection techniques," Telecommun. Syst., vol. 76, pp. 139–154, 2021.

A. G. Liu, E. Musial, and M.-H. Chen, "Progressive reliability forecasting of service-oriented software," in 2011 IEEE international conference on web services, IEEE, 2011, pp. 532–539.

A. Iqbal and S. Aftab, "A Classification Framework for Software Defect Prediction Using Multi-filter Feature Selection Technique and MLP.," Int. J. Mod. Educ. Comput. Sci., vol. 12, no. 1, 2020.

A. Iqbal, S. Aftab, and F. Matloob, "Performance analysis of resampling techniques on class imbalance issue in software defect prediction," Int. J. Inf. Technol. Comput. Sci, vol. 11, no. 11, pp. 44–53, 2019.

A. Memon, A. A. Siddiqui, and M. A. Khan, "Impact of Total Quality Management, Entrepreneurial Orientation and Organizational Excellence on Organizational Performance: Evidence from Manufacturing Firms of Kotri (S.I.T.E) Sindh Pakistan," Int. Res. J. Mod. Eng. Technol. Sci., vol. 4, no. 12, pp. 2083–2097, 2022, [Online]. Available: https://www.irjmets.com/uploadedfiles/paper//issue_12_december_2022/32250/final/fin_irjmets1676015268.pdf

A. N. Babatunde, R. O. Ogundokun, L. B. Adeoye, and S. Misra, "Software Defect Prediction Using Dagging Meta-Learner-Based Classifiers," Mathematics, vol. 11, no. 12, p. 2714, 2023.

A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques," J. King Saud Univ. Inf. Sci., 2023.

A. Shankar Mishra and S. Singh Rathore, "Implicit and explicit mixture of experts models for software defect prediction," Softw. Qual. J., pp. 1–38, 2023.

A.-T. Nguyen, S. Reiter, and P. Rigo, "A review on simulation-based optimization methods applied to building performance analysis," Appl. Energy, vol. 113, pp. 1043–1058, 2014.

B. A. Akinnuwesi, G. D. Adenaike, and O. C. Nwokoro, "A Systematic Review of Soft Computing Techniques for Software Testing.," Int. J. Comput. Sci. Manag. Stud., vol. 40, no. 4, 2019.

B. B. Gupta, K. Yadav, I. Razzak, K. Psannis, A. Castiglione, and X. Chang, "A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment," Comput. Commun., vol. 175, pp. 47–57, 2021.

C. Gupta, I. Johri, K. Srinivasan, Y.-C. Hu, S. M. Qaisar, and K.-Y. Huang, "A systematic review on machine learning and deep learning models for electronic information security in mobile networks," Sensors, vol. 22, no. 5, p. 2017, 2022.

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

C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto, "Comments on 'researcher bias: the use of machine learning in software defect prediction,'" IEEE Trans. Softw. Eng., vol. 42, no. 11, pp. 1092–1094, 2016.

C.-Y. Huang, "Performance analysis of software reliability growth models with testing-effort and change-point," J. Syst. Softw., vol. 76, no. 2, pp. 181–194, 2005.
D. Ryu, J.-I. Jang, and J. Baik, "A transfer cost-sensitive boosting approach for cross-project defect prediction," Softw. Qual. J., vol. 25, pp. 235–272, 2017.
D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," Empir. Softw. Eng., vol. 21, pp. 43–71, 2016.
E. Erturk and E. A. Sezer, "A comparison of some soft computing methods for software fault prediction," Expert Syst. Appl., vol. 42, no. 4, pp. 1872–1879, 2015.
E. v Venkatesan and T. Velmurugan, "Performance analysis of decision tree algorithms for breast cancer classification," Indian J. Sci. Technol., vol. 8, no. 29, pp. 1–8, 2015.
F. H. Alshammari, "Software Defect Prediction and Analysis Using Enhanced Random Forest (extRF) Technique: A Business Process Management and Improvement Concept in IOT-Based Application Processing Environment.," Mob. Inf. Syst., 2022.
F. Hassan, N. A. Qureshi, M. A. Khan, Muhammad Zohaib Khan, A. S. Soomro, A. Imroz, and H. B. Marri, "An Integrated Approach for Sentiment Classification and Information Retrieval Techniques Using K-Means, Logistic Regression, Random Forest, and Decision Tree, Algorithm," J. Appl. Res. Technol. Eng., vol. 4, no. 2, 2023, [Online]. Available: https://polipapers.upv.es/index.php/JARTE/article/view/19306/15859
F. Matloob et al., "Software defect prediction using ensemble learning: A systematic literature review," IEEE Access, vol. 9, pp. 98754–98771, 2021.
H. Carreon-Ortiz, F. Valdez, and O. Castillo, "A new discrete mycorrhiza optimization nature-inspired algorithm," Axioms, vol. 11, no. 8, p. 391, 2022.
H. D. D. Meares and M. P. Jones, "When a System Breaks: A Queuing Theory Model for the Number of Intensive Intensive Intensive Care Beds Needed Dur-ing the COVID-19 Pandemic".
H. Koziolek, "Performance evaluation of component-based software systems: A survey," Perform. Eval., vol. 67, no. 8, pp. 634–658, 2010.
H. Mittal and N. Sharma, "A probabilistic model for the assessment of queuing time of coronavirus disease (COVID-19) patients using queuing model," Technology, vol. 11, no. 8, pp. 22–31, 2020.
I. Arora, V. Tetarwal, and A. Saha, "Open issues in software defect prediction," Procedia Comput. Sci., vol. 46, pp. 906–912, 2015.
I. E. Haines and M. P. Jones, "When a system breaks: a queuing theory model for the number of intensive care beds needed during the COVID-19 pandemic," Med. J. Aust, 2020.
J. Liu, J. Lei, Z. Liao, and J. He, "Software defect prediction model based on improved twin support vector machines," Soft Comput., pp. 1–10, 2023.
J. Pachouly, S. Ahirrao, K. Kotecha, G. Selvachandran, and A. Abraham, "A systematic literature review on software defect prediction using artificial intelligence: Datasets, Data Validation Methods, Approaches, and Tools," Eng. Appl. Artif. Intell., vol. 111, p. 104773, 2022.
J. Ren, K. Qin, Y. Ma, and G. Luo, "On software defect prediction using machine learning," J. Appl. Math., vol. 2014, 2014.
J. Tian and M. V Zelkowitz, "Complexity measure evaluation and selection," IEEE Trans. Softw. Eng., vol. 21, no. 8, pp. 641–650, 1995.
K. Jeet, N. Bhatia, and R. S. Minhas, "A bayesian network based approach for software defects prediction," ACM SIGSOFT Softw. Eng. Notes, vol. 36, no. 4, pp. 1–5, 2011.
K. Khan, M. A. Khan, J. A. Thebo, T. Ahmed, and L. A. Rahoo, "Examining The Human Resource Architecture Relationship With Employee Productivity Of Chemical Industries," J. Contemp. Issues Bus. Gov., vol. 27, no. 2, pp. 5847–5856, 2021, [Online]. Available: https://www.cibgp.com/article_11267_91767391154f6eee74a8fa4a1c11a1c6.pdf
K. S. Kavya and D. Y. Prasanth, "An ensemble deepboost classifier for software defect prediction," Int. J. Adv. Trends Comput. Sci. Eng., vol. 9, no. 2, pp. 2021–2028, 2020.
L. A. Rahoo, M. A. K. Nagar, and A. Bhutto, "The Use of Information Retrieval Tools by the Postgraduate Students of Higher Educational Institutes of Pakistan," Asian J. Contemp. Educ., vol. 3, no. 1, pp. 59–64, 2019, doi: 10.18488/journal.137.2019.31.59.64.

L. A. Rahoo, P. B. Channar, and M. A. Khan, "Analysis of Stress on the Employees of Software Development Industries of Pakistan," Int. Res. J. Comput. Sci. Technol., vol. 1, no. 1, pp. 6–12, 2020, [Online]. Available: http://irjcst.com/index.php/irjcst/article/view/2/1

L. A. Rahoo, P. Hasnain, A. M. Abbasi, T. Ahmed, and M. A. Khan, "The Relationship Between Information Technology and Organizational Culture in The University Libraries of Sindh, Pakistan," J. Contemp. Issues Bus. Gov. Vol, vol. 27, no. 2, 2021, [Online]. Available: https://www.cibgp.com/article_10816_ff2852c7bcdca4f3c72857a4da607bbe.pdf

L. Gong, S. Jiang, and L. Jiang, "Tackling class imbalance problem in software defect prediction through cluster-based over-sampling with filtering," IEEE Access, vol. 7, pp. 145725–145737, 2019.

L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," Mach. Learn. Knowl. Extr., vol. 3, no. 3, pp. 672–694, 2021.

M. A. Kalwar, H. B. Marri, and M. A. Khan, "Performance Improvement of Sale Order Detail Preparation by Using Visual Basic for Applications: A Case Study of Footwear Industry," Int. J. Bus. Educ. Manag. Stud., vol. 3, no. 1, pp. 1–22, 2021, [Online]. Available: https://ijbems.com/doc/IJBEMS-159.pdf

M. A. Kalwar, H. B. Marri, M. A. Khan, and S. A. Khaskheli, "Applications of Queuing Theory and Discrete Event Simulation in Health Care Units of Pakistan," Int. J. Sci. Eng. Investig., vol. 10, no. 9, pp. 6–18, 2021, [Online]. Available: www.IJSEI.com

M. A. Kalwar, M. A. Khan, M. F. Shahzad, M. H. Wadho, and H. B. Marri, "Development of linear programming model for optimization of product mix and maximization of profit: case of leather industry," J. Appl. Res. Technol. Eng., vol. 3, no. 1, pp. 67–78, 2022, doi: 10.4995/jarte.2022.16391.

M. A. Khan, A. Khatri, and H. B. Marri, "Identification of Defects in Various Processes of Spinning: A Case Study of Kotri, Sindh, Pakistan," in Proceedings of the First Central American and Caribbean International Conference on Industrial Engineering and Operations Management, Port-au-Prince, Haiti, June 15-16, 2021, 2021. [Online]. Available: http://ieomsociety.org/proceedings/2021haiti/299.pdf

M. Assim, Q. Obeidat, and M. Hammad, "Software defects prediction using machine learning algorithms," in 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI), IEEE, 2020, pp. 1–6.

M. Bukhsh et al., "Productivity Improvement in Textile Industry using Lean Manufacturing Practice of Single Minute Die Exchange (SMED)," in Proceedings of the 11th Annual International Conference on Industrial Engineering and Operations Management Singapore, March 7-11, 2021, 2021. [Online]. Available: http://www.ieomsociety.org/singapore2021/papers/1282.pdf

M. J. Hernández-Molinos, A. J. Sánchez-García, R. E. Barrientos-Martínez, J. C. Pérez-Arriaga, and J. O. Ocharán-Hernández, "Software Defect Prediction with Bayesian Approaches," Mathematics, vol. 11, no. 11, p. 2524, 2023.

M. Jorayeva, A. Akbulut, C. Catal, and A. Mishra, "Machine learning-based software defect prediction for mobile applications: A systematic literature review," Sensors, vol. 22, no. 7, p. 2551, 2022.

M. K. Albzeirat, M. I. Hussain, R. Ahmad, F. M. Al-Saraireh, A. Salahuddin, and N. Bin-Abdun, "Applications of nano-fluid in nuclear power plants within a future vision," Int. J. Appl. Eng. Res., vol. 13, no. 7, pp. 5528–5533, 2018.

M. Memon, M. A. Khan, and L. A. Rahoo, "Usage and Availability of Information and Communication Technology Applications Facilities at Central Library," Int. Res. J. Comput. Sci. Technol., vol. 1, no. 1, pp. 86–92, 2020, [Online]. Available: http://irjcst.com/index.php/irjcst/article/view/7/6

M. Nabi, A. Wahid, and P. Kumar, "Performance Analysis of Classification Algorithms in Predicting Diabetes.," Int. J. Adv. Res. Comput. Sci., vol. 8, no. 3, 2017.

M. Prashanthi and C. M. Miryala, "Defect prediction in software using spiderhunt-based deep convolutional neural network classifier," Int. J. Netw. Virtual Organ., vol. 27, no. 4, pp. 337–357, 2022.

M. S. Arain, M. A. Khan, and M. A. Kalwar, "Optimization of Target Calculation Method for Leather Skiving and Stamping: Case of Leather Footwear Industry," Int. J. Bus. Educ.

Manag. Stud., vol. 7, no. 1, pp. 15–30, 2020, [Online]. Available: https://www.ijbems.com/doc/IJBEMS-137.pdf

M. Shafiq and Z. Gu, "Deep residual learning for image recognition: A survey," Appl. Sci., vol. 12, no. 18, p. 8972, 2022.

M. Singh and D. S. Salaria, "Software defect prediction tool based on neural network," Int. J. Comput. Appl., vol. 70, no. 22, 2013.

M. Z. Khan et al., "The Performance Analysis of Machine Learning Algorithms for Credit Card Fraud Detection," Int. J. Online Biomed. Eng., vol. 19, no. 03, pp. 82–98, 2023, doi: 10.3991/ijoe.v19i03.35331.

M. Z. Khan, F. U. Zaman, M. Adnan, A. Imroz, and M. A. Rauf, "Comparative Case Study : An Evaluation of Performance Computation Between SQL And NoSQL Database," Sindh J. Headways Softw. Eng., vol. 01, no. 02, pp. 14–23, 2022.

N. Baladi, P. B. Channar, L. A. Rahoo, T. Ahmed, and M. A. Khan, "Improve Customer Retention through Service Quality Attributes in the Restaurant Industry of Pakistan," J. Contemp. Issues Bus. Gov., vol. 27, no. 6, pp. 331–340, 2021, [Online]. Available: https://www.cibgp.com/article_12147_76fd80af7f9013320f57d25d1cfccea1.pdf

N. E. Fenton and M. Neil, "A critique of software defect prediction models," IEEE Trans. Softw. Eng., vol. 25, no. 5, pp. 675–689, 1999.

N. Jaleel, M. A. Khan, M. Jamal, M. Safeeruddin, M. M. Shajee, and U. Mughal, "Productivity Improvement by Lean Methodologies at Dyeing & Printing Plant," in Proceedings (Abstract) of the International Conference on Industrial & Mechanical Engineering and Operations Management Dhaka, Bangladesh, December 26-27, 2021., 2021, p. 905. [Online]. Available: https://ieomsociety.org/proceedings/2021dhaka/495.pdf

N. Li, M. Shepperd, and Y. Guo, "A systematic review of unsupervised learning techniques for software defect prediction," Inf. Softw. Technol., vol. 122, p. 106287, 2020.

P. D. Singh and A. Chug, "Software defect prediction analysis using machine learning algorithms," in 2017 7th international conference on cloud computing, data science & engineering-confluence, IEEE, 2017, pp. 775–781.

P. K. Sadhu, V. P. Yanambaka, and A. Abdelgawad, "Internet of things: Security and solutions survey," Sensors, vol. 22, no. 19, p. 7433, 2022.

P. Kumar, M. A. Khan, U. K. Mughal, and S. Kumar, "Exploring the Potential of Six Sigma (DMAIC) in Minimizing the Production Defects," in Proceedings of the 3rd International Conference on Industrial & Mechanical Engineering and Operations Management Dhaka, Bangladesh, December 26-27, 2020, 2020. [Online]. Available: http://www.ieomsociety.org/imeom/260.pdf

P. Pahwa, M. Papreja, and R. Miglani, "Performance analysis of classification algorithms," Int J Comput Sci Mob Comput, vol. 3, no. 4, pp. 50–58, 2014.

Q. Wang, S. Wu, and M.-S. Li, "Software defect prediction," J. Softw., vol. 19, no. 7, pp. 1565–1580, 2008.

Q. Zhang and J. Ren, "Software-defect prediction within and across projects based on improved self-organizing data mining," J. Supercomput., vol. 78, no. 5, pp. 6147–6173, 2022.

R. Annisa, D. Rosiyadi, and D. Riana, "Improved point center algorithm for k-means clustering to increase software defect prediction," Int. J. Adv. Intell. Informatics, vol. 6, no. 3, pp. 328–339, 2020.

R. B. Jadhav, S. D. Joshi, U. G. Thorat, and A. S. Joshi, "A software defect learning and analysis utilizing regression method for quality software development," Int. J. Adv. Trends Comput. Sci. Eng., vol. 8, no. 4, pp. 1275–1282, 2019.

R. Garg, K. Sharma, R. Kumar, and R. K. Garg, "Performance analysis of software reliability models using matrix method," Int. J. Comput. Inf. Eng., vol. 4, no. 11, pp. 1646–1653, 2010.

R. S. Wahono, "A systematic literature review of software defect prediction," J. Softw. Eng., vol. 1, no. 1, pp. 1–16, 2015.

Ruide, L. Juelong, Y. Qiliang, and X. Liqiang, "A new model for software defect prediction using particle swarm optimization and support vector machine," in 2013 25th Chinese Control and Decision Conference (CCDC), IEEE, 2013, pp. 4106–4110.

S. A. Khaskheli, H. A. Kalwar, M. A. Kalwar, H. B. Marri, M. A. Khan, and M. Nebhwani, "Application of Multi-Server Queuing Model to Analyze The Queuing System of OPD During COVID-19 Pandemic: A Case Study," J. Contemp. Issues Bus. Gov., vol. 27, no. 05, pp. 1351–1367, 2021, doi: 10.47750/cibg.2021.27.05.094.

S. A. Khaskheli, H. B. Marri, M. Nebhwani, M. A. Khan, and M. Ahmed, "Compartive study of queuing systems of medical out patient departments of two public hospitals," Proc. Int. Conf. Ind. Eng. Oper. Manag., vol. 0, no. March, pp. 2702–2720, 2020.

S. Arshad, H. A. Rehman, L. A. Rahoo, and M. A. K. Nagar, "Information Communication Technology Applications used to Enhance Knowledge Management in the University Libraries of Pakistan," in Proceedings of IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS), 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8629133/

S. Balsamo, A. Di Marco, P. Inverardi, and M. Simeoni, "Model-based performance prediction in software development: A survey," IEEE Trans. Softw. Eng., vol. 30, no. 5, pp. 295–310, 2004.

S. G. Jacob, "Improved random forest algorithm for software defect prediction through data mining techniques," Int. J. Comput. Appl., vol. 117, no. 23, 2015.

S. Goyal, "Effective software defect prediction using support vector machines (SVMs)," Int. J. Syst. Assur. Eng. Manag., vol. 13, no. 2, pp. 681–696, 2022.

S. K. Punia, M. Kumar, T. Stephan, G. G. Deverajan, and R. Patan, "Performance analysis of machine learning algorithms for big data classification: Ml and ai-based algorithms for big data analysis," Int. J. E-Health Med. Commun., vol. 12, no. 4, pp. 60–75, 2021.

S. Kabir and Y. Papadopoulos, "Applications of Bayesian networks and Petri nets in safety, reliability, and risk assessments: A review," Saf. Sci., vol. 115, pp. 154–175, 2019.

S. L. Zimmerman, A. R. Rutherford, A. van der Waall, M. Norena, and P. Dodek, "A queuing model for ventilator capacity management during the COVID-19 pandemic," Health Care Manag. Sci., pp. 1–17, 2023.

S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," IEEE Trans. Softw. Eng., vol. 34, no. 4, pp. 485–496, 2008.

S. P. Chatzis and A. S. Andreou, "Maximum entropy discrimination poisson regression for software reliability modeling," IEEE Trans. neural networks Learn. Syst., vol. 26, no. 11, pp. 2689–2701, 2015.

S. Rajput, M. A. Khan, S. Samejo, G. Murtaza, and R. A. Ali, "Productivity Improvement by the Implementation of lean manufacturing practice (takt time) in an automobile assembling plant," in Proceedings of the International Conference on Industrial Engineering and Operations Management Dubai, UAE, March 10-12, 2020, 2020, pp. 1618–1619. [Online]. Available: http://www.ieomsociety.org/ieom2020/papers/190.pdf

S. Vanaja and K. Rameshkumar, "Performance analysis of classification algorithms on medical diagnoses-a survey," J. Comput. Sci., vol. 11, no. 1, p. 31, 2015.

S. Zhang, S. Jiang, and Y. Yan, "A Software Defect Prediction Approach Based on Hybrid Feature Dimensionality Reduction," Sci. Program., vol. 2023, 2023.

T. Bergander, Y. Luo, and A. Ben Hamza, "Software defects prediction using operating characteristic curves," in 2007 IEEE International Conference on Information Reuse and Integration, IEEE, 2007, pp. 713–718.

T. Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener, "Defect prediction from static code features: current results, limitations, new approaches," Autom. Softw. Eng., vol. 17, pp. 375–407, 2010.

T. Sharma, A. Jatain, S. Bhaskar, and K. Pabreja, "Ensemble Machine Learning Paradigms in Software Defect Prediction," Procedia Comput. Sci., vol. 218, pp. 199–209, 2023.

U. K. Mughal, M. A. Khan, P. Kumar, and S. Kumar, "Identification and Analysis of Stitching Defects at the Stitching Unit: A Case Study," in Proceedings of the First Central American and Caribbean International Conference on Industrial Engineering and Operations Management, Port-au-Prince, Haiti, June 15-16, 2021, 2021. [Online]. Available: http://ieomsociety.org/proceedings/2021haiti/298.pdf

V. A. Phan, "Learning Stretch-Shrink Latent Representations With Autoencoder and K-Means for Software Defect Prediction," IEEE Access, vol. 10, pp. 117827–117835, 2022.
V. Gaur and R. Kumar, "Analysis of machine learning classifiers for early detection of DDoS attacks on IoT devices," Arab. J. Sci. Eng., vol. 47, no. 2, pp. 1353–1374, 2022.
V. U. B. Challagulla, F. B. Bastani, I.-L. Yen, and R. A. Paul, "Empirical assessment of machine learning based software defect prediction techniques," Int. J. Artif. Intell. Tools, vol. 17, no. 02, pp. 389–400, 2008.
W. Ahmad, A. Rasool, A. R. Javed, T. Baker, and Z. Jalil, "Cyber security in IoT-based cloud computing: A comprehensive survey," Electronics, vol. 11, no. 1, p. 16, 2021.
W.-D. Zhao, S.-D. Zhang, and M. Wang, "Software Defect Prediction Method Based on Cost-Sensitive Random Forest," in International Conference on Intelligent Information Processing, Springer, 2022, pp. 369–381.
X. Cai, S. Geng, D. Wu, and J. Chen, "Unified integration of many-objective optimization algorithm based on temporary offspring for software defects prediction," Swarm Evol. Comput., vol. 63, p. 100871, 2021.
X. Tan, X. Peng, S. Pan, and W. Zhao, "Assessing software quality by program clustering and defect prediction," in 2011 18th working conference on Reverse Engineering, IEEE, 2011, pp. 244–248.
X. Yu, J. Li, and F. Kang, "SSA optimized back propagation neural network model for dam displacement monitoring based on long-term temperature data," Eur. J. Environ. Civ. Eng., vol. 27, no. 4, pp. 1617–1643, 2023.
Y. A. Alsariera, V. E. Adeyemo, A. O. Balogun, and A. K. Alazzawi, "Ai meta-learners and extra-trees algorithm for the detection of phishing websites," IEEE access, vol. 8, pp. 142532–142542, 2020.
Y. Dutil, D. R. Rousse, N. Ben Salah, S. Lassue, and L. Zalewski, "A review on phase-change materials: Mathematical modeling and simulations," Renew. Sustain. Energy Rev., vol. 15, no. 1, pp. 112–130, 2011.M. Z. Khan and R. Alluhaibi, "Performance Analysis of Software Defects Prediction using Over-Sampling (SMOTE) and Resampling," Int. J. Comput. Sci. Netw. Secur., vol. 19, no. 11, pp. 202–215, 2019.
Y. Jiang, M. Li, and Z.-H. Zhou, "Software defect detection with ROCUS," J. Comput. Sci. Technol., vol. 26, no. 2, pp. 328–342, 2011.
Y. Peng, G. Kou, G. Wang, W. Wu, and Y. Shi, "Ensemble of software defect predictors: an AHP-based evaluation method," Int. J. Inf. Technol. Decis. Mak., vol. 10, no. 01, pp. 187–206, 2011.
Y. Zhang, D. Lo, X. Xia, and J. Sun, "An empirical study of classifier combination for cross-project defect prediction," in 2015 IEEE 39th Annual computer software and applications conference, IEEE, 2015, pp. 264–269.
Y.-T. Li and S. Malik, "Performance analysis of embedded software using implicit path enumeration," IEEE Trans. Comput. Des. Integr. circuits Syst., vol. 16, no. 12, pp. 1477–1487, 1997.
Z. Hu and Y. Zhu, "Cross-project defect prediction method based on genetic algorithm feature selection," Eng. Reports, p. e12670, 2023.
Z. Iftikhar et al., "Lean Manufacturing Tools and Techniques for the Productivity Improvement in Assembly Lines Operations of Industries," Int. Res. J. Mod. Eng. Technol. Sci., vol. 4, no. 7, pp. 4554–4562, 2022, [Online]. Available: https://www.irjmets.com/uploadedfiles/paper//issue_7_july_2022/28986/final/fin_irjmets1663258443.pdf
Z. Li, X.-Y. Jing, and X. Zhu, "Progress on approaches to software defect prediction," Iet Softw., vol. 12, no. 3, pp. 161–175, 2018.