Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121

DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

Research Article

Develop Approach to Predicate Software Reliability Growth Model Parameters Based on Machine learning

[1] *Anfal A. Fadhil*
Software
Computer Science and Mathematics
University of Mosul, Mosul, Iraq
anfalaaf@uomosul.edu.iq

[2] *Asmaa' H. AL_Bayati*
Software
Computer Science and Mathematics
University of Mosul, Mosul, Iraq
asmashade77@uomosul.edu.iq

[3] *Ibrahim Ahmed Saleh*
Software
Computer Science and Mathematics
University of Mosul, Mosul, Iraq
i.hadedi@uomosul.edu.iq

A R T I C L E I N F O
Article History
Received: 16/07/2024
Accepted: 24/09/2024
Published: 31/12/2024

ABSTRACT
One of the most important aspects in determining the quality of a software product before placing it on the market is its reliability. The main problem in creating effective software that satisfies the user preferences is that it must be highly reliable. One important factor that has a remarkable influence on the overall reliability of a system is its software. Reliability is a critical aspect of software quality, and the software industry faces many challenges in its quest to produce reliable software at scale. Reliability models are a basic method for quantitatively calculating software reliability. Thus, this paper inspects the reliability of software applications as a substantial feature of this application and helps determine the extent of software reliability in performing specialized functions. This goal is accomplished by calculating the parameters of software reliability growth models (SRGMs). The parameters are evaluated using three algorithms: machine learning decision tree (DT), support vector machine (SVM), and K-nearest neighbors (K-NN). Results show that the SVM model achieves the best mean square error.

Keywords: software reliability growth models (SRGMs,) machine learning, decision tree (DT), K-nearest neighbors (K-NN), support vector machine (SVM)

1. Introduction

One of the most important fields of engineering practice and theory for software reliability is software reliability growth models (SRGMs), so all types of software systems have dominated our current environment as a result of the advancement of computer technology, including watches, phones, and appliances for the home, and even more such as buildings, cars, planes, and more important locations where they play an increasingly greater part [1]. Software reliability is an important feature that affects the overall reliability of a system. Reliability models are a typical method for quantitatively evaluating software reliability [2]. Producing highly effective software that meets end-user expectations is an essential challenge in designing such software systems. As a result, academics are increasingly concerned about software reliability. The software system's reliability can be defined as "the possibility that software will operate without fault for a specified period in a given environment" [1][2]. Hence, software reliability is a major concern. The software development team and management can use the reliability value to schedule software milestones. System engineering, product management, reengineering, and software development evaluation are all associated with software dependability measurements [2] [3]. To assess software reliability, several SRGMs are applied. For every SRGM, calculating attributes is a routine activity as part of reliability estimation. The connections between the parameters of an SRGM determine the optimal calculation of such attributes. Consequently, parameter

DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

estimation approaches are used in SRGMs [3]. SRGMs have been extensively researched in the literature, and their construction has been explored by many writers. Many models were considered, however, those that such as the power (POW) model, the S-shaped model (Yamada S-shaped model), the exponential model (Goel–Okumoto [G-O] model), the logarithmic model, and the inverse polynomial model all have two parameters and were the most widely utilized in the literature [4]. Several models are introduced throughout this article. The technique of the decision tree (DT) algorithm is introduced to indicate its role in parameter estimation to creating a model that predicts the value of a target variable by learning simple decision rules inferred from data features. The second algorithm is support vector machine (SVM) that is also used in classification and regression to find a hyperplane that predicts the target variable. The third algorithm is K-nearest neighbor (K-NN) that uses closeness for predictions or classifications about the cluster of an individual data point.

The contribution of these paper is calculating the parameters of SRGMs using the three algorithms, namely, machine learning (ML) DT, SVM, and K-NN by using three datasets (software for monitoring and real-time control).

The remainder of this article is arranged as follows: Section 2 reviews relevant literature. Section 3 focuses on the SRGMs. Section 4 describes the data set used in this paper. Section 5 presents the ML algorithm applied. Section 6 explains the estimate evaluation and results. Section 7 presents the conclusion and future works.

2. Review of Literature

Software reliability models refer to a typical method for quantitatively evaluating software reliability, and much research has been conducted about this subject. They include different algorithms from ML and deep learning techniques that have been used throughout time. Several of these algorithms are described in the section's first paragraphs below.

In 2016, Begum Momotaz and Tadashi Dohi [5] focused on a common multilayer perceptron neural network prediction problem to determine the prediction interval for the overall number of software failures during sequential testing. They built prediction intervals using the delta approach and well-known feed-forward neural network back propagation algorithm. Regarding coverage rate, prediction interval, and average relative error, four real data sets from software development projects were subjected to this technique, and the results could cover the point prediction and the actual data in their regions.

In 2017, Ramasamy and Lakshmanan [6] suggested a traditional nonhomogeneous Poisson process (NHPP) model in combination with an infinite test effort function. They utilized data on software failures to train a suggested model using artificial neural networks (ANNs). To clarify historical failure data fully, they obtained a large set of weights for the same model and applied the ML approach to determine the proper weights for the model that will precisely describe past and future data. The results showed the proposed logpower testing effort function (TEF) that depends on SRGM, which uses the ML technique, enhances the accuracy of the goodness of fit performance

In 2018, Jaiswal and Malhotra [7] proposed a method to predict software reliability by using several methods ML ("neuro fuzzy inference," "general regression neural network," "feed forward backpropagation neural network," "multilayer perceptron," "support vector machines," "linear regression," "instance-based learning," and "cascading forward backpropagation neural network"). They evaluated these techniques using specific performance criteria, and "neuro fuzzy inference" produced the best result.

In 2019, Hanagal and Bhalerao [8] used an SRGM for overall extended inverse Weibull finite failures. They explained the failure occurrence rate fault's increasing and decreasing behavior by using the hazard of the generalized extended inverse Weibull distribution, which considers the hazard function's increasing and decreasing features. They suggested a finite failure NHPP for SRGMs and obtained unknown model parameters for interval domain data using the maximum likelihood method. They evaluated the suggested model using mean square error (MSE), predictive ratio risk, and Akaike information criteria, Considering the sum of squares attributable to error, compared with earlier techniques, this dependability estimating method produced satisfactory results.

In 2020, Da Hye Lee et al. [9] proposed a new SRGM that assumes related software failures. They executed experiments on real-world datasets and used several evaluation criteria. They evaluated the model's recommended goodness-of-fit to the results of previous NHPP SRGMs using these criteria. Furthermore, Wald's test of sequential probability ratio (SPRT) was used to assess the reliability of the software. The results illustrated the excellent performance of the suggested model.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

In 2021, Liya Li [10] produced a new method of endless evaluation execution with the recommended data to apply ANN to previous models of programming model disappointment using NHPP. SRGM employed common-sense information programming to create a disappointed pointer that displayed the suggested number of TEF and SRGM. Additionally, this research effectively demonstrated disappointment in extensive information, ML, and ANN.

In 2022, Shoichiro Miyamoto et al. [11] proposed a method for a deep learning-based reliability assessment Open Source Software (OSS) that people with no previous experience with deep learning may use. The method combines general and specialized data to estimate OSS reliability. This method illustrated that certain types of data that have different features from general data are valuable as explanatory variables.

In 2023, Lee et al. [12] suggested a model that depends on dependent breakdowns as well as unpredictable operating conditions for evaluating software reliability by applying SPRT. This paper realized the goal of a premature reliability. In the same year, Youn Su Kim et al. [13] used similarity in shape between deep learning (sigmoid function) and software reliability model to produce a deep learning model for software dependability that substitutes software reliability function for activation function and sigmoid function. They used 10 criteria to compare and analyze two datasets, and the results verified the superiority of the suggested deep learning software reliability model.

3. Software Reliability Growth Models

Software for reliability is one of the most important customer-oriented qualities of software quality [14] [15]. Effective ways to developing reliable software, as well as objectively estimating software reliability, are critical [15]. Thus, the primary goal of the software industry is to create superior quality, error-free software. By providing mathematical models based on simulated testing oriented environments, software reliability growth models assist industry in producing required quality goods [16].

Over the past three decades, SRGMs have been developed and can offer a great deal of information on how to increase reliability. Through SRGM, essential parameters facilitate determining the duration, quantity of unrepaired defects, mean time between failures, and mean time to failure [17]. Two main kinds of viewpoints focus on how to organize reliability models [4] [18]:

Defect Density Models (DDMs) estimate reliability based on design variables. To assess the number of flaws in software, DDMs use code features such as input/output, external references, code lines, and loop nesting. SRGMs assess software reliability using test results. These models establish a statistical connection between defect detection data and well-known functions, such as an exponential function. For example. as soon correlation is sufficient, the known function can be utilized to predict future behavior. In this paper, the models listed below are examined:

1) Exponential Model (G-O Model) [4] [19-22]:

In the topic of software reliability, the G-O model is a well-known basic model. The exponential distribution is assumed as the lifetime distribution per defect in this model. As a result, the rate at which problems occur is constant. Goel and Okumoto presented a nonhomogeneous Poisson-based stochastic model for software failure, which uses an exponential curve to represent failure observation phenomena. Following the introduction of software systems, the software systems employed in development-testing settings are identical to or comparable with those used in field environments. NHPP can be used to represent the finite failure G-O model. The expectation of the model can be written as Eq. (1,2):

$$\mu(t) = a \times (1 - e^{-b\times t}), \tag{1}$$

$$\mu(t) = a \times b \times e^{-b\times t}, \tag{2}$$

where (a) is the initial prediction of the total failure, (b) is the fault detection rate, and (t) is the time of failure. The number of flaws that need to be found (a) treats the observed value as a random variable that depends on the test and other environmental factors. This model differed fundamentally from earlier ones that treated the quantity of errors as a constant variable [4].

2) The Power Model [4] [22]:

The model calculates the hardware system's reliability during testing. The NHPP model was used to build the model. The following equations describe the relationship between time t and μ(t) and λ(t), as shown in Eq. (3,4):

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

$$\mu(t) = a \times t^b \tag{3}$$

$$\mu(t) = a \times b \times t \times e^{b-1} \tag{4}$$

3) The Yamada Delayed S-Shaped Model [4] [23]:

The gamma distribution class includes the delayed S-shaped model. However, the model uses Musa and Okumoto's categorization scheme and several Poisson-type failures per period rather than a binomial type. Because the exponential model has been improved, the model represents learning based on the project team's developing experience. This paradigm of failure is also finite. The system equations for μ (t) and λ (t) are shown as Eq. (5,6):

$$\mu(t) = a(1 - (1 + bt) \times e^{-b\times t} \tag{5}$$

$$\mu(t) = a \times b^2 \times t^{-b\times t} \tag{6}$$

4. Description of Dataset

In this paper, three datasets were used (software for monitoring and real-time control). The sample in Table 1 comprises the measured fault (xk) reading, the total number of test workers (twk), and the accumulated faults (yk) shared in the procedures for evaluation [24]:

TABLE I. Part of datasets (software for monitoring and real-time control)

Days	Xk	yk	Twk
1	4	4	1
2	0	4	1
3	7	11	1
4	10	21	1
5	13	34	1
6	8	42	1
7	13	55	1
8	4	59	1
9	7	66	1
10	8	74	1

Dataset 1 includes 109 measurements. The program is used in a middle-level language to assess and debug the data of an application for real-time control utilizing 870 kilo lines of code (KLOC) in size [24].

Dataset 2 includes 111 measurements. The program uses report data to assess or troubleshoot a real-time program. The program system has 200 modules, and each module has 1 KLOC [24].

Dataset 3 includes 46 measurements. The data gathered through the testing are limited. This small number of data makes correct parameter estimation difficult at times [24].

For each program, the dataset was divided into training data and testing data to construct a reliability paradigm. To predict the model's parameters, 80% of the measurements were used assess the developed model, 20% of the dataset was used.

5. Machine Learning

Because ML can examine large, complicated data sets and detect patterns that are difficult to find with traditional statistical methods, it has been used to forecast dependability growth patterns. Furthermore, ML models can predict future reliability growth more accurately because they can anticipate complicated data and predict accuracy by learning from previous reliability data. They can forecast how a system's dependability will increase over time by considering several variables, including historical failures, solutions that have been deployed, and usage trends. ML handles complicated data, captures nonlinear correlations, and learns continually from fresh data to enhance the forecasting power of reliability growth models. ML results in models that are more precise, adaptable, and flexible, which eventually improves software.

5.1 Decision Tree

DT is a preferred supervised learning technique that is utilized in machine learning and data mining for tasks including regression and classification. This structure is similar to a flowchart, where each internal node represents a "test" on an attribute, each branch displays the result of the test, and each leaf node represents a class label (in classification) or a numerical value (in regression). DT typically works for the following:

1) Feature Selection: The algorithm selects the best feature from the dataset that best splits the data into subsets. Various criteria such as Gini impurity, entropy, or information gain are commonly used to measure the quality of a split.

2) Splitting: The dataset is divided into smaller subsets according to the values of the selected characteristic.

3) Recurve: This procedure for deciding which trait is the best, and separating the dataset is recursively employed for each subset until the criteria for stopping is satisfied. A minimum improvement in impurity, a minimum number of samples needed to divide a node, or a maximum depth for the tree could all be examples of stopping criteria.

Once the stopping criteria are met, the final subsets become the leaf nodes of the tree, and each leaf node is assigned a class label or a regression value, as shown in Figure 1 [25].

5.2 Support Vector Machine

A supervised learning algorithm used for regression and classification tasks is called SVM. SVM is widely utilized in ML for categorization and is especially successful in high-dimensional spaces. The key idea behind SVM is to find the hyperplane that best separates the data points into different classes, as shown in Figure 2, optimizing the margin, it is the distance among hyperplane and the nearest data points from each class, is called support vectors, it works for the following:

1) Given a set of labeled training data, where each data point belongs to one of two classes, SVM finds the optimal hyperplane that separates these classes.

2) SVM uses the kernel function to points has maximum values features of input data to find linear boundary which nearest features from points.

3) The algorithm searches for the hyperbolic level that is chosen to separate the data points with the maximum possible margin between the two classes, which expressed mathematically in Eq. (7).

$$H_0 = b_0 + w_0 . x = 0, \tag{7}$$

where x is the input vector, w is the weight, and b is the bias value.

4) Once the boundary and hyperplane are defined, any new point can be classified by calculating on which side of the hyperplane the data points closest to the hyperplane are called support vectors [25].

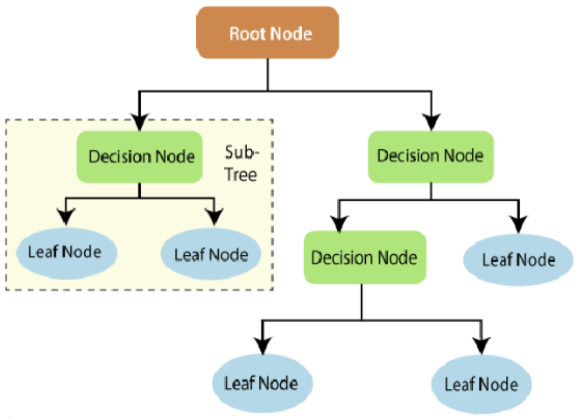

Fig. 1. Decision tree

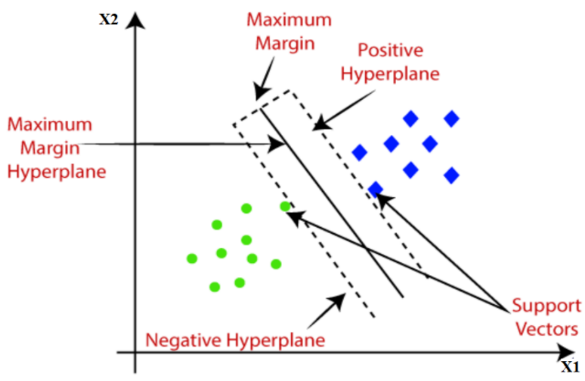

Fig. 2. Support vector machine

5.40 K_Nearest Neighbor

K-NN is a supervised learning ML algorithm that has been widely used in classification problems. K-NN is most commonly used for classification and prediction due to its many interesting features, including its efficiency and easy implementation. The algorithm steps are listed as follows:

Step 1: The algorithm stores a training dataset.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

Step 2: K that represents the quantity of nearby neighbors is determined.

Step 3: The Euclidean distance between the test points and the trained points is computed using Euclid's law through Eq. (8), as shown below:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (xi - yi)^2}, \qquad (8)$$

where d is the distance between any two points; xi and yi are data points in the search space.

Step 4: The nearest neighbors of the K points are taken according to the Euclidean distance computed in Eq. (8).

Step 4: The number of data points for each class among these K neighbors is calculated.

Step 5: The class with the most neighbors is given the maximum data points. Figure 3 illustrates the K-NN algorithm [26].

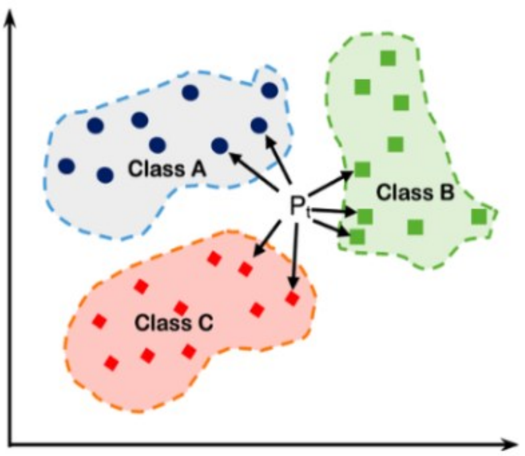

Fig. 3. K-nearest neighbor boundaries

6. Estimation of the Software Reliability Growth Model Parameters Using Machine Learning

The proposed work applied ML algorithms to describe the time-dependent nature of N (t). The algorithm was applied to three SRGMs and three databases. Figure 4 shows an activity diagram for ML algorithms to determine the best parameters for SRGMs. When applying a DT for SRGM, initially, the initial tree is created with the following properties: training size=0.8, testing size=0.2, random state=42, max_depth=23, and criterion=entropy. The tree's root node, which contains the integral data set, is formed, then the best feature in the data set is searched, and the information entropy of samples is measured. If all samples belong to the same class, its value is 0. If the samples are evenly divided into different classes, its value is 1. New DTs are recursively created using subsets, and the discovered values are applied to the SRGMs. This process is continued until a point when the nodes cannot be sorted further, and the final node is called the leaf node to determine the best parameters for SRGM. When applying the second K-NN algorithm, its properties are as follows: training size=0.8, testing size=0.2, random state=42, and K=5. The number of neighbors that equals (5) is determined, the Euclidean distance of the number K between the points is calculated, and then the nearest neighbors are determined according to the Euclidean distance that represents the data points for each category to obtain best parameters for the SRGMs.

The third algorithm, namely, SVM applied to the SRGM with properties of training size=0.8, testing size=0.2, random state=42, kernel=rbf, and c=165. SVM iteratively creates the hyperplane to best separate the classes and then selects the hyperplane that correctly separates the classes.

6.1 Evaluation and Results

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

The efficiency of DT, SVM, and K-NN algorithms that are used to solve the problem of parameter estimation in SRGMs, which have been applied in MATLAB 2018 to find the best parameters, is computed by using following two metrics on three datasets:

1- Root Mean Square Error (RMSE) as shown in Eq. (9) [24]:

$$\text{RMSE}=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - Y_i)^2}\ , \tag{9}$$

2- Mean Squared Error (MSE) as shown in Eq. (10) [27]:

$$\text{MSE}=\frac{1}{N}\sum_{i=1}^{N}(y_i - Y_i)^2\ , \tag{10}$$

where y_i the actual failure number in time i, and Y_i is the failure rate calculated using the ML algorithm in SRGM in time i.

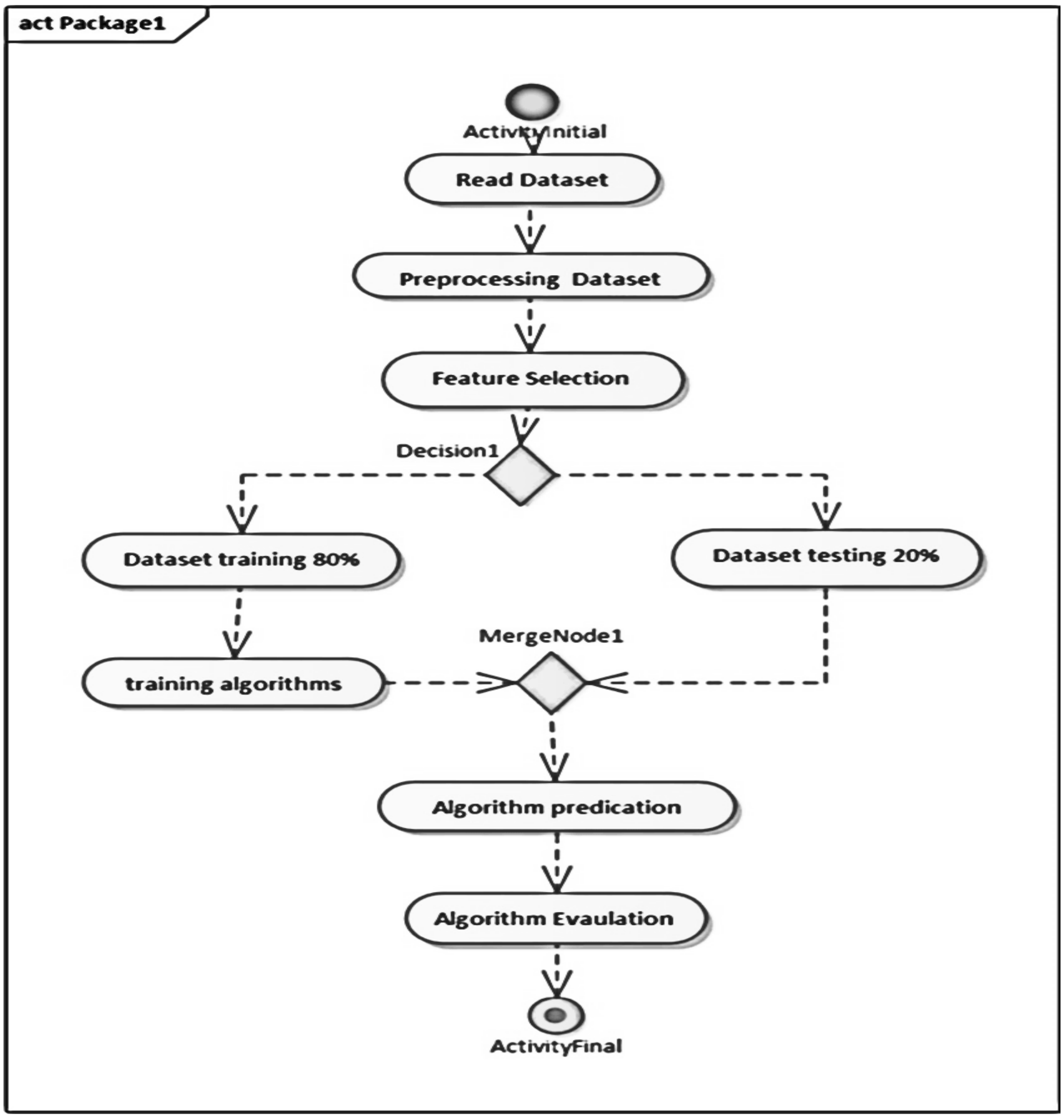

Fig. 4. Activity diagram for machine learning algorithms to find the best parameters for the software reliability growth models

The proposed algorithms were compared using three datasets taken corresponding with those that other academics have referred to as follows:

1) The first case study (Dataset 1) [24] includes 109 measurements with daily collected data. The predicted model parameters (a and b) that depend on the DT, SVM, and KNN algorithms in the search space in these dataset cases were [0, 750] and b [0, 1]. Table 2 displays the SRGMs' parameters: RMSE and MSE. Figure

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

5 presents the actual and estimated accumulated failures for the best result (SVN algorithm) using the SRGMs.

TABLE II. SRGM parameters (RMSE and MSE) using Dataset 1

Model	Algorithm	A	B	RMSE	MSE
G-O	DT	1670.9987	0.0051	67.9757	4620.69
	K-NN	717.098	0.01495539	28.7056	824.01147
	SVM	705.046	0.015353	28.704	823.919616
POW	DT	10.1655	0.92338	113.478	12877.25
	K-NN	24.541	0.684974	71.2465	5076.06
	SVM	25.750	0.6681	70.5187	4972.88
DSS	DT	596.729	0.045927	19.7213	388.929
	K-NN	562.3995	0.04947323	14.9117	222.358
	SVM	565.75	0.0490	14.9105	222.323

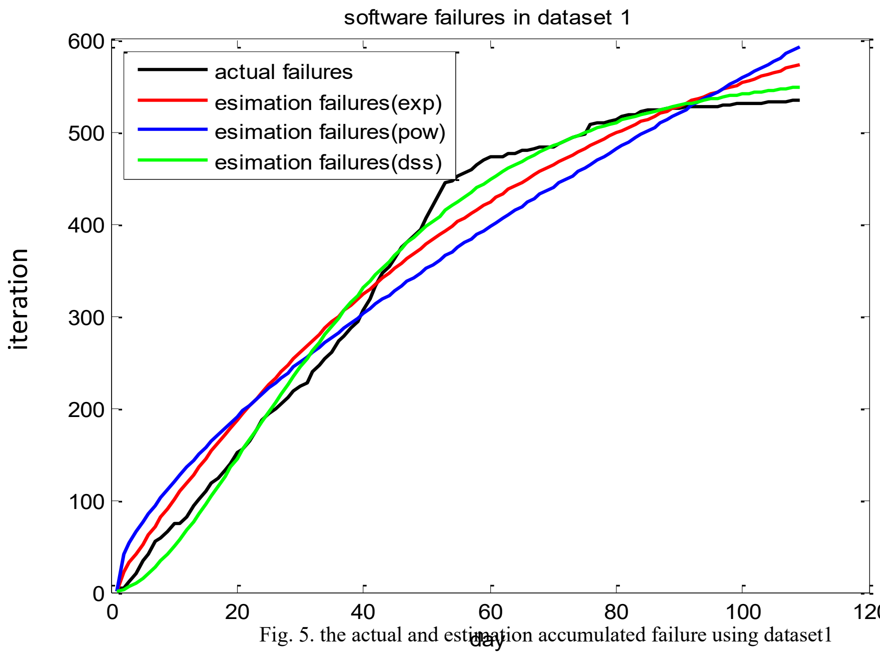

Fig. 5. the actual and estimation accumulated failure using dataset1

2. The second case study (Dataset 2) includes 111 measurements with daily collected data. The predicted model parameters (a and b) that depend on machine learning in the search space in this dataset case were a [0, 550] and b [0, 1]. Table 3 displays the SRGMs' parameters: RMSE and MSE. Figure 6 shows the actual and estimated accumulated failures for the best result (SVN algorithm) using the SRGMs.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

TABLE III SRGM parameters (RMSE and MSE) using Dataset 2

Model	Algorithm	A	b	RMSE	MSE
G-O	DT	696.7171	0.017	49.5443	2454.6
	K-NN	538.6468	0.0256	28.1024	789.744
	SVM	542.785	0.0252	28.10197	789.7207
POW	DT	21.7567	0.73627	109.237	11932.7
	K-NN	30	0.684974	74.2117	5507.37
	SVM	54.0405	0.4747	71.6299	5130.8
DSS	DT	502.571	0.0635	19.7598	390.449
	K-NN	486.3256	0.0669	18.0853	327.078
	SVM	488.487	0.0664	18.0588	326.121

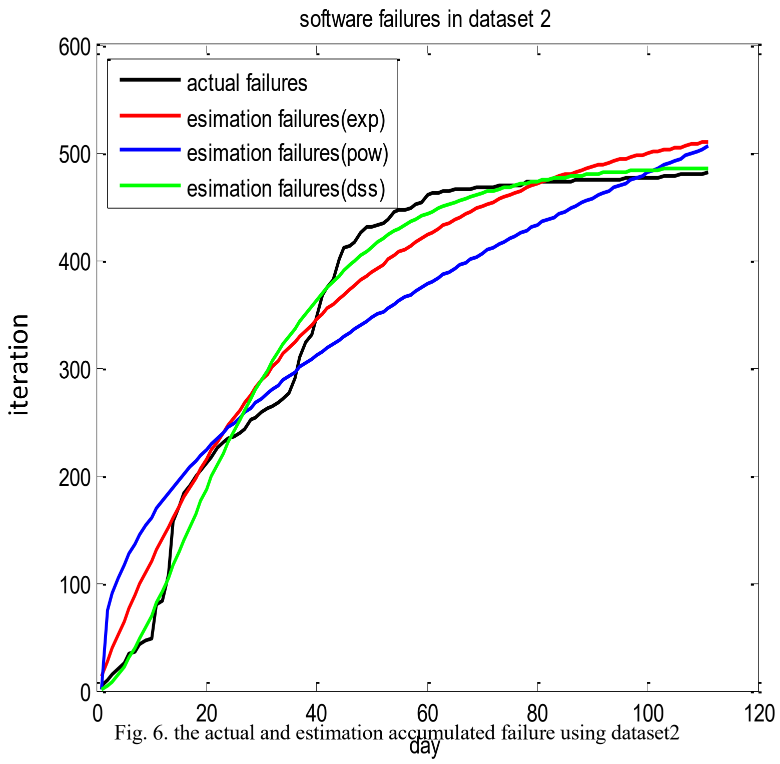

Fig. 6. the actual and estimation accumulated failure using dataset2

3. The third case study (Dataset 3) includes 46 measurements with daily collected data. The predicted a and b in the search space in these dataset cases were a [0, 450] and b [0, 1]. Table 4 displays the SRGMs' parameters: RMSE, and MSE. Figure 7 shows the actual and estimated accumulated failures for the best result (SVN algorithm) using the SRGMs.

TABLE IV. SRGM parameters (RMSE and MSE) using Dataset 3

Model	Algorithm	A	B	RMSE	MSE
G-O	DT	382.4057	0.0261	13.1212	172.1658
	K-NN	422.5453	0.02324815	12.1613	147.897
	SVM	399.75	0.0251	12.1493	147.60549
POW	DT	14.9701	0.776	44.556	1985.2
	K-NN	16.4506	0.74628	43.6507	1905.4
	SVM	25.75	0.5970	41.8778	1753.75
DSS	DT	216.3715	0.136	30.3951	923.862
	K-NN	280.2617	0.09711093	18.5828	345.32
	SVM	300.75	0.0879	19.2699	371.329

DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

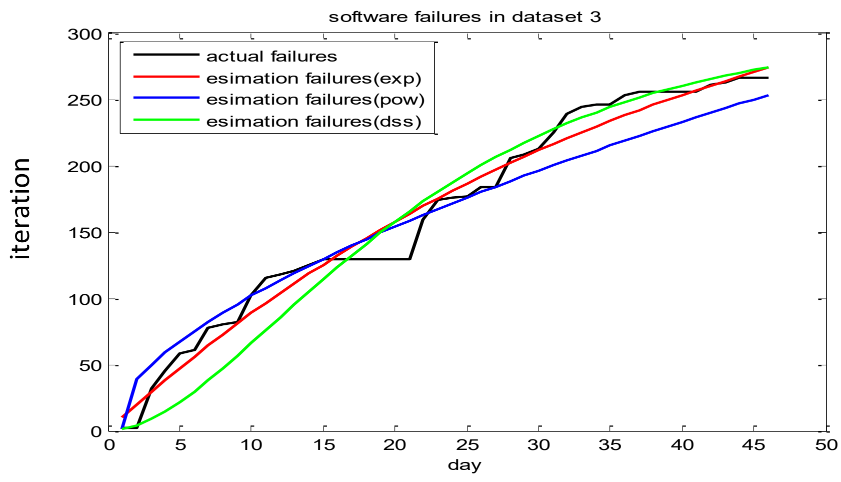

Fig. 7. Actual and estimated accumulated failure using Dataset 3

7. Conclusions and Further Work

This paper proposes a methodology to predict the parameters of SRGMs depending on three ML algorithms, namely, DT, SVM, and K-NN. The development of the software for the proposed work aims to reduce the variation between the number of actual failures and the total number of predicted failures using three basic SRGMs: the S-shaped model, the POW model, and the exponential model. All models were applied to the three datasets.

The algorithms used to find the best-parameterized SRGMs are based on the failures accumulated during testing activity in a software system. After execution for the G-O model on Dataset 1 using SVM, the results of RMSE=28.704 and MSE=823.919616 were the best for DT and K-NN. After execution for the POW model on Dataset 1 using SVM, the results based on RMSE=70.5187 and MSE=4972.88 were the best for DT and K-NN. After execution for the DSS model on Dataset 1 using SVM, the results based on RMSE=14.9105 and MSE=222.323 were the best for DT and K-NN.

When applying the algorithm to Datasets 2 and 3, SVM achieved the best performance based on two measures (MSE and RMSE) of the results, where the proposed SVM technique was the most successful in predicting the parameters of SRGMs.

In further works, ensemble learning methods can be applied to find the best parameters of SRGMs.

Acknowledgment

The authors express their thanks and appreciation to the University of Mosul/ College of Computer Sciences and Mathematics for their supplied facilities, which contributed to improving the accuracy of this paper.

References

[1] K. Lu and Z.Ma, “A modified whale optimization algorithm for parameter estimation of software reliability growth models “,Journal of Algorithms & Computational Technology, Volume 15: 1–14, 2021, https://doi.org/10.1177/17483026211034442

[2] Al-Mutairi, N. N., Al-Turk, L., Al-Rajhi, sh., “A New Reliability Model Based on Lindley Distribution with Application to Failure Data”, Mathematical Problems in Engineering, Volume 2020, Article ID 4915812, 11 pages https://doi.org/10.1155/2020/4915812

[3] Sreedhar Y., Mohan K. , “ Software Reliability Growth Models: A Critical Research with Experimentation for Parameter Estimation “,International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume 8, Issue 4, November 2019.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121

DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

[4] AL-Saati, N., Abd-AlKareem M., " The Use of Cuckoo Search in Estimating the Parameters of Software Reliability Growth Models", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 11, No. 6, June 2013.

[5] Momotaz, B., Dohi, T.,"Prediction interval of cumulative number of software faults using multilayer perceptron", Applied Computing & Information Technology. Studies in Computational Intelligence, Volume 619. Springer, Cham. https://doi.org/10.1007/978-3-319-26396-0_4

[6] Ramasamy, S., Lakshmanan, I.,"Machine Learning Approach for Software Reliability Growth Modeling with Infinite Testing Effort Function", Mathematical Problems in Engineering Volume 2017, Article ID 8040346, 6 pages 2017. https://doi.org/10.1155/2017/8040346

[7] Jaiswal ,A., Ruchika, M. , "Software Reliability Prediction Using Machine Learning Techniques", Proceedings of Fifth International Conference on Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing, Volume 436. Springer, Singapore. (2016). https://doi.org/10.1007/978-981-10-0448-3_12

[8] Hanagal, D. D., Bhalerao, N.N.," Modeling On Generalized Extended Inverse Weibull Software Reliability Growth Model", Journal of Data Science,17(3). P. 575 - 592,2019, DOI:10.6339/JDS.201907_17(3).0007

[9] Lee, D.H., Chang, I.H., Pham, H. ,"Software Reliability Model with Dependent Failures and SPRT", Mathematics 2020, 8, 1366. https://doi.org/10.3390/math8081366

[10] Liya Li," Software Reliability Growth Fault Correction Model Based on Machine Learning and Neural Network Algorithm", Microprocessors and Microsystems, Volume 80, February 2021

[11] Miyamoto, SH., Tamura, Y., Yamada, sh., "Reliability Assessment Tool Based on Deep Learning and Data Preprocessing for OSS", American Journal of Operations Research, Volume 12, No 3, 2022, https://www.scirp.org/journal/ajor

[12] Lee, D., Chang, I., Pham, H.," Study of a New Software Reliability Growth Model under Uncertain Operating Environments and Dependent Failures", *Mathematics*, Volume 11, issue 18, 2023. https://doi.org/10.3390/math11183810

[13] Kim, Y.S., Pham, H., Chang, I.H.,"Deep-Learning Software Reliability Model Using SRGM as Activation Function", Applied Sciences , Volume 13 ,issue 19,2023, https://doi.org/10.3390/app131910836

[14] Yamad,SH. , " Software Reliability Modeling: Fundamentals and Applications ", Tokyo, Japan: Springer, pp. 1_20, 2014, https://api.semanticscholar.org/CorpusID:196591311

[15] Qiuying Li, Q., Pham, H.," A Generalized Software Reliability Growth Model With Consideration of the Uncertainty of Operating Environments ", IEEE Access, Volume 7,2019

[16] Rafi, SH.,Rao, B.,Akthar, SH.," Software Reliability Growth Model with New Dynamic Learning Effects and Release Time Determination ", International Journal of Innovative Science and Research Technology, Volume 3, Issue 6, June 2018.

[17] Kapur, P.K. , Pham, H.,Gupta, A., Jha, P.C., "Software Reliability Assessment with OR Applications", Springer Science & Business Media,2011.

[18] Quadri, S. M. K. , Ahmad, N., " Software Reliability Growth Modeling with New Modified Weibull Testing–effort and Optimal Release Policy ",International Journal of Computer Applications,Volume 6, No.12, September 2010.

[19] Al-Rahamneh, Z., Reyalat,M., Sheta, A. F., Bani-Ahmad, S., Al-Oqeili, S., "A New Software Reliability Growth Model: Genetic-Programming-Based Approach", In Journal of Software Engineering and Applications, volume 4, 2011.

[20] Albert Orwa Akuno, A.,Ndonye, T., Nthiwa, J., Orawo, L.," Regression Approach to Parameter Estimation of an Exponential Software Reliability Model ",American Journal of Theoretical and Applied Statistics, volume 5, 2016 .

[21] Song, K.Y., Chang,I. H., "Improved Exponential Software Reliability Model Based on NHPP with the Uncertainty of Operating Environments ", Journal of the Chosun Natural Science, Volume 10 Issue 4, Pages.249-257, 2017.

[22] Song, K.Y., Chang,I. H.,H. Pham, H., "A software reliability model with a weibull fault detection rate function subject to operating environments",Applied science(Switzerland), Volume 7, 2017.

[23] Hudaib, A., Moshref,M., " Survey in software Reliability Growth Models: Parameter Estimation and Models Ranking ",International Journal of Computer Systems, Volume 05, Issue 05, May 2018.

Iraqi Journal for Computers and Informatics
Information Technology and Communications University
Vol. 50, No. 2, 2024, pp. 110-121
DOI: https://doi.org/10.25195/ijci.v49i2.509
Print ISSN: 2313-190X, Online ISSN: 2520-4912

[24] Sheta, A., "Reliability Growth Modeling for Software Fault Detection Using Particle Swarm Optimization", IEEE Congress on Evolutionary Computation Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada. pp: 3071- 3078, 2006.
[25] Hasan, O. Sh., Saleh I.A., "Development of heart attack prediction model based on ensemble learning", Eastern-European Journal of Enterprise Technologies, 2021.
[26] Mirkes,E.," KNN and Potential Energy (Applet)", University of Leicester,2011, http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html
[27] BOKHAFU, M. U. , AHMAD, N.,"Software Reliability Growth Modeling for Exponentiated Weibull Function with Actual Software Failures Data", Innovative Applications of Information Technology for the Developing World,2007