Software Reliability Prediction Using Machine Learning Techniques

Arunima Jaiswal and Ruchika Malhotra

Abstract Software reliability is an indispensable part of software quality. Software industry endures various challenges in developing highly reliable software. Application of machine learning (ML) techniques for software reliability prediction has shown meticulous and remarkable results. In this paper, we propose the use of machine learning techniques for software reliability prediction and evaluate them based on selected performance criteria. We have applied ML techniques including adaptive neuro fuzzy inference system (ANFIS), feed forward backpropagation neural network (FFBPNN), general regression neural network (GRNN), support vector machines (SVM), multilayer perceptron (MLP), bagging, cascading forward backpropagation neural network (CFBPNN), instance-based learning (IBK), linear regression (Lin Reg), M5P, reduced error pruning tree (reptree), and M5Rules to predict the software reliability on various datasets being chosen from industrial software. Based on the experiments conducted, it was observed that ANFIS yields better results and it predicts the reliability more accurately and precisely as compared to all the above-mentioned techniques. In this study, we also made comparative analysis between cumulative failure data and inter failure time data and found that cumulative failure data give better and more promising results as compared to inter failure time data.

Keywords Software reliability · Assessment · Prediction · Machine learning techniques

Arunima Jaiswal (✉) · Ruchika Malhotra
Delhi Technological University, Shahbad Daulatpur, Main Bawana Road,
Delhi 110042, India
e-mail: ajaiswal34@amity.edu

Ruchika Malhotra
e-mail: ruchikamalhotra@dce.edu

M. Pant et al. (eds.), *Proceedings of Fifth International Conference on Soft Computing for Problem Solving*, Advances in Intelligent Systems and Computing 436, DOI 10.1007/978-981-10-0448-3_12

1 Introduction

Software reliability is defined as "The ability of the software to perform its required function under stated conditions for a stated period of time" [1]. Reliability is one of the important attributes of the software quality. According to ANSI, software reliability is "The probability of failure-free operation of a computer program for a specified period of time in a specified environment" [2]. Software reliability growth models (SRGM) have been used for predicting and estimating the probability of a system failing in given time interval or the expected time span between successive failures.

ML is an approach which is focused on learning automatically and allows computers to evolve and predict the system behavior based on past and the present failure data. Thus, it is quite natural for software practitioners and researchers to know that which particular method tends to work well for a given failure dataset and up to what extent quantitatively [3–7].

In this study, we present an empirical study of above-mentioned ML techniques for predicting software reliability based on five industrial datasets and investigate about the accuracy and performances of these models in predicting the software reliability when applied to past failure week data. We also performed a comparative analysis between cumulative failure data and inter failure time data to investigate the type of failure data more appropriate for reliability prediction.

2 Study Objective

Business applications which are critical in nature require reliable software, but developing such reliable software is a key challenge which our software industry faces today. With the increasing complexity of the software these days achieving software reliability is hard. In our study, we attempt to empirically assess the use of ANFIS for predicting the software failures. Although ANN, SVM etc., have been previously used in the literature [8] but for the first time ANFIS has been applied to cumulative week failure dataset. The background of using ANFIS was that if it had proven empirically to predict the software failures with least errors in comparison to the above-mentioned techniques, then it may possibly be used as a sound alternative to other mentioned existing techniques for software reliability predictions. Also, above-mentioned ML techniques were empirically analyzed for the first time together on five different types of datasets taken altogether.

3 Related Work

Several ML techniques have been proposed and applied in the literature for software reliability modeling and forecasting. Some of the techniques are genetic programming, gene expression programming, artificial neural network, decision trees, support vector machines, feed forward neural network, fuzzy models, generalized neural network, etc. [8–16]. Karunanithi et al. [11] carried out analysis of detailed study to explain the use of connectionist models in the reliability growth prediction for the software. Cai et al. [15] focused on the development of fuzzy software reliability models instead of probabilistic software reliability models as he says that reliability is fuzzy in nature. Ho et al. [17] carried out a comprehensive study of connectionist models and their applicability to software reliability prediction and inferred that these are better as compared to traditional models. Su and Huang (2006) [18] had applied neural network for predicting software reliability. Madsen et al., 2006 [19] focused on the application of soft computing techniques for software reliability prediction. Pai and Hong (2006) [20] performed experiments using SVMs for forecasting software reliability. Despite of recent advances in this field, it was observed that different models have varied predictive reliability capabilities.

4 Research Background

In this section, we summarize empirical data collection and independent and dependent variables.

4.1 Empirical Data Collection

In this paper, we have used software failure data from various projects given in (Project Data) [20, 21], data collected from tandem computers software data project (Project Data) (release 1 and 2) [22]. Other datasets include telecommunication system data (Project Data) (phase 1 and 2) [23]). Also, we have used data from the project on-line data entry IBM software package (Project Data) [24].

4.2 Independent and Dependent Variables

The dependent variable which we have used in this study is failure rate and the independent variable used is time interval in terms of weeks.

5 Research Methodology

In this study, we explored the above-mentioned ML techniques for predicting failures. We have divided entire dataset into two parts: training and testing datasets. The training and testing datasets selection are being employed using k-fold cross-validation. Here, 10 cross-validation is used where nine parts are used for training and one part is used for validation taken randomly 10 times and results are recorded for each of the 10 runs. This process is applied to each of the five datasets taken into consideration. Thus, cross-validation is used so as to maximize the utilization of failure past datasets by repeatedly resampling the same dataset randomly by reordering it and then splitting it into tenfolds of equal length [25].

A number of modeling techniques, both statistical and intelligent, were used by us to predict reliability.

5.1 Machine Learning Techniques

5.1.1 Adaptive Neuro Fuzzy Inference System (ANFIS)

ANFIS is a hybrid of two intelligent system models. It combines the low-level computational power of a neural network with the high-level reasoning capability of a fuzzy inference system [26]. The basic steps of the FIS model are (i) identification of input variables (failure time) and output variables (cumulative failures) (ii) development of fuzzy profile of the input/output variables (iii) defining relationships between input and output variables using fuzzy inference system. Thus, FIS is capable of making decisions under uncertainty which can be applied for reliability prediction when applied to unknown failure datasets [27].

5.1.2 Cascade Forward Backpropagation Neural Networks (CFBPNN)

It has more than one layer of neurons. The weights of each of the subsequent layer originate from the input and all the layers prior to the layer are in question. Each layer has biases. First, adaption is performed to create the model. Then, the model is trained using the stipulated number of epochs. Performance is computed according to the stipulated performance function. These networks have a weight connection from input to every successive layer and from every layer to all the following layers which sometimes improve the speed at which the model is trained and they only require past failure data as input for prediction analysis and no assumptions.

5.1.3 Feed Forward Backpropagation Neural Networks (FFBPNN)

It contains more than one layer of neurons and is carried out in order to build the model and train it. Single-layered feed forward neural networks have one layer of sigmoid neurons which are then followed by an output layer of linear neurons. The input layer with transfer functions that may be sigmoid (or of any other type except linear) permits the network model to learn both nonlinear as well as linear relationships between input variable vectors and output variable vectors.

5.1.4 Generalized Regression Neural Networks (GRNN)

A generalized regression neural network (probabilistic neural network) consists of a radial basis layer along with a special linear layer. Spread is associated with it whose value generally lies close to 1. A large spread leads to a bigger area from the input vector where the input layer will respond with a number of significant outputs.

5.1.5 Multilayered Perceptron (MLP)

It is a type of FFBPNN, where the backpropagation learning algorithm is in the form of gradient descent. It maps vectors of input data to a vector of appropriate outputs and contains more than one layer of nodes where each of the layers is completely connected to the next layer. In this type of networks, except for the input nodes, each node has a nonlinear activation function like sigmoid function associated with it.

5.1.6 Linear Regression (Lin Reg)

This technique envisages creation of the simplest model using a single input variable and a single output variable. In this model, there are some independent variables between which a linear relationship is found out to yield result in the form of a dependent variable.

5.1.7 M5P

It is a technique based on Quinlan's M5 algorithm and generates M5 model trees. Initially in order to build a tree, a decision tree induction algorithm is executed first. A splitting criterion is then applied till the class values of each of the instance that reaches a node vary slightly or when just a few instances are left. Then, the backward pruning from each of the leaf is being applied.

5.1.8 M5Rules

This technique uses divide and conquer method to generate a decision list. With every iteration, it builds a model tree and converts the best leaf into a separate rule.

5.1.9 Support Vector Machine (SVM)

It is also called as support vector regression (SVR). Application of this technique yields a model which is usually affected only by a subset of the data used while training the model. This is made possible since the cost function used for creating the model has no impact of the training points lying beyond the margin.

5.1.10 Bagging (Bootstrap Aggregating)

It is a ML ensemble which is employed to improve the accuracy as well as stability of the ML algorithms which are generally used in classification. It reduces the variance and helps avoiding over fitting issue.

5.1.11 Reduced Error Pruning Tree (REPTree)

It is a fast decision tree learner which builds a decision or a regression tree and performs the pruning with back over fitting. It sorts values for numeric attributes only. REPTree produces a suboptimal tree under the constraint that a subtree can only be pruned if it does not contain a subtree with a lower classification error than itself.

5.1.12 IBk (Instance-Based k-Nearest Neighbor)

It employs k-nearest neighbor algorithm in order to classify data. It stores the instances in memory while performing training and then compares new instances with these stored instances. It predicts the failure by looking at the k-nearest neighbors of a test instance.

5.2 Statistical Efficacy Measures

In order to validate the proposed reliability prediction models, we have used several efficacy measures which are summarized in Table 1.

The precise view of overall training and prediction process is being illustrated through a flow diagram in Fig. 1.

Table 1 Efficacy measures used for evaluating performance criteria

Efficacy measure	Definition
Correlation coefficient	Correlation coefficient measures how closely actual and predicted values are correlated with each other. The correlation coefficient lies between −1 and +1. No linear relationship is there if the correlation coefficient is 0
MARE	Mean absolute relative error is computed as the mean of the absolute difference of corresponding actual value and the corresponding predicted value divided by the corresponding actual value
MRE	Mean relative error is computed as the mean of the difference of corresponding actual value and the corresponding predicted value divided by the corresponding actual value
MSE	Mean squared error is the mean of the square of the difference of corresponding actual value and the corresponding predicted value, i.e., mean of the square of errors

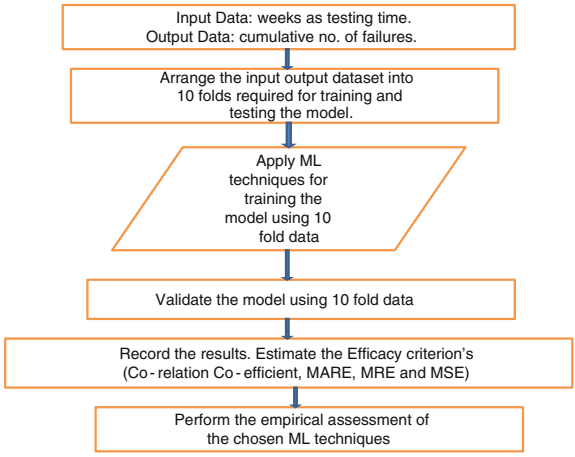

Fig. 1 Overview of software reliability prediction process

In this study, we predict the dependent variable (failure rate) based on the number of failures to be detected using various ML techniques and envisaging the application of tenfold cross-validation method required for training (training the model using training data) and testing the model (using the testing data). Next step includes recording of failures predicted by applying above-mentioned ML techniques. Then we estimate the statistical efficacy measures for all the chosen datasets. Finally, we perform the empirical assessment of the chosen ML techniques.

6 Result Analysis

In this section, we present the summary of results obtained for predicting reliability using five datasets which we have taken for comparison using ML techniques in terms of efficacy measures which are correlation coefficient, MARE, MRE, and

MSE. The results are being summarized in the following Tables 2, 3, 4, 5, 6, 7, 8, 9, and 10. Sometimes, MARE, MRE, and MSE are expressed in percentage (%). However, this paper follows the definition given in Table 1 and does not express MARE in percentage (%) [28].

Table 2 shows the results of correlation coefficients on five datasets taken for reliability analysis in this work using 11 ML techniques and one statistical method (Lin Reg) method.

Table 3 shows the values of MSE on five datasets taken for reliability analysis in this work using 11 ML techniques and one statistical method (Lin Reg) method.

Table 4 shows the values of MARE on five datasets taken for reliability analysis in this work using 11 ML techniques and one statistical method (Lin Reg) method.

Table 5 shows the values of MRE on five datasets taken for reliability analysis in this work using 11 ML techniques and one statistical method (Lin Reg) method.

These performance criteria were also being observed for the cumulative as well as inter failure time datasets. Table 6 shows the results of correlation coefficients on dataset given in project data [20] taken for reliability analysis in this work using 9 ML techniques and one statistical method (Lin Reg) method.

Table 7 shows the results of MARE on dataset given in project data [20] taken for reliability analysis in this work using 9 ML techniques and one statistical method (Lin Reg) method.

Table 8 shows the results of correlation coefficients on dataset given in project data [24] taken for reliability analysis in this work using 9 ML techniques and one statistical method (Lin Reg) method. The results show that the correlation coefficient is above 0.9 for ANFIS prediction which shows that the actual and the predicted values are very close. It also depicts that the cumulative failures yield high correlation coefficients within the ranges of 0.74–0.99 in comparison to the correlation coefficients being calculated for inter failure time data.

Table 9 shows the results of MARE on dataset given in project data [24] taken for reliability analysis in this work using 9 ML techniques and one statistical method (Lin Reg). The results show that the value of MARE is 0.04 for ANFIS prediction. It also depicts that the cumulative failures yield low MARE within the ranges of 0.04–0.48 in comparison to the MAREs being calculated for inter failure time data.

Table 10 shows the pred(0.25) for different datasets. It also depicts that most of the values of MRE lies below 0.25. Hence, the results are very high.

Figure 2 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on MARE for the industrial datasets taken into consideration.

Figure 3 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on correlation coefficient for the industrial datasets taken into consideration.

Figure 4 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on MRE for the industrial datasets taken into consideration.

Table 2 Summary of correlation coefficient predictions for different datasets

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Linear regression	MSP	M5Rules	Multilayered perceptron	REPTree	SVM
Project data	0.999	0.998	0.978	0.99	0.969	0.992	0.891	0.967	0.984	0.994	0.967	0.866
Project data (release 2)	0.999	0.998	0.997	0.984	0.976	0.986	0.966	0.982	0.988	0.998	0.945	0.955
Project data (release 1)	0.999	0.998	0.984	0.987	0.965	0.99	0.961	0.982	0.994	0.998	0.951	0.96
Project data (phase 1)	0.997	0.986	0.978	0.986	0.98	0.978	0.982	0.985	0.984	0.995	0.957	0.984
Project data (phase 2)	0.998	0.997	0.987	0.993	0.98	0.983	0.985	0.986	0.994	0.997	0.94	0.983

Table 3 Summary of MSE for different datasets

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Linear regression	MSP	M5 Rules	Multilayered perceptron	REPTree	SVM
Project data	15.838	22.077	231.55	116.534	438.628	85.902	1061.5	352.199	178.3	70.021	352.199	1636.6
Project data (release 2)	1.176	2.717	11.107	67.484	111.949	38	88.726	47.438	12.774	4.069	151.623	118.156
Project data (release 1)	1.658	4.857	43.193	39.131	61.866	15.875	61.834	28.861	12.774	3.286	91.562	64.69
Project data (phase 1)	0.39	2.792	2.607	1.649	3.401	3.321	2.681	2.277	2.381	0.852	10.663	2.781
Project data (phase 2)	0.548	0.618	4.431	3.043	9.001	6.81	5.912	5.516	4.784	1.109	24.313	7.557

Table 4 Summary of MARE for different datasets

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Linear regression	M5P	M5Rule5	Multilayered perceptron	REPTree	SVM
Project data	0.145	0.293	0.442	0.151	1.031	0.194	1.34	0.832	0.188	0.155	0.832	2.166
Project data (release 2)	0.027	0.053	0.108	0.21	0.266	0.105	0.161	0.109	0.117	0.031	0.256	0.223
Project data (release 1)	0.031	0.059	0.194	0.11	0.162	0.075	0.125	0.083	0.067	0.032	0.158	0.131
Project data (phase 1)	0.14	0.55	0.638	0.219	0.397	0.147	0.246	0.222	0.213	0.145	0.449	0.214
Project data (phase 2)	0.073	0.09	0.249	0.142	0.268	0.141	0.177	0.161	0.141	0.074	0.318	0.194

Table 5 Summary of MRE for different datasets

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Linear regression	MSP	M5RULES	Multilayered perceptron	REPTree	SVM
Project data	0.025	−0.249	−0.077	−0.005	−0.974	−0.115	−1.191	−0.755	−0.114	−0.074	−0.755	−2.112
Project data (release 2)	−0.012	−0.035	0.066	0.107	−0.22	−0.012	−0.074	−0.04	−0.011	0.006	−0.154	−0.116
Project data (release 1)	−0.015	−0.045	0.036	−0.064	−0.093	−0.015	−0.046	−0.028	−0.023	−0.084	−0.103	−0.058
Project data (phase 1)	0.019	−0.458	0.239	−0.121	−0.307	−0.045	0.077	0.076	0.062	0	−0.304	0.056
Project data (phase 2)	−0.008	−0.044	0.158	0.048	−0.201	−0.036	0.029	0.027	0.008	−0.002	−0.169	−0.013

Table 6 Summary of correlation coefficient predictions for cumulative versus inter failure time data

ML techniques	Inter failure data	Cumulative data
ANFIS	0.307	0.995
GRNN	0.36	0.979
FFBPNN	0.324	0.939
CFBPNN	0.158	0.975
Bagging	0.126	0.946
IBK	−0.147	0.951
Linear regression	0.264	0.92
Multilayered perceptron	0.333	0.975
REPTree	−0.093	0.856
SVM REG	0.392	0.87

Table 7 Summary of MARE predictions for cumulative versus inter failure time data

Mare	Inter failure data	Cumulative data
ANFIS	91.963	0.285
GRNN	104.925	0.314
FFBPNN	88.761	1.192
CFBPNN	94.627	0.504
Bagging	73.151	1.123
IBK	152.986	0.318
Linear regression	61.497	1.504
Multilayered perceptron	91.745	1.384
REPTree	84.267	2.255
SVM REG	32.054	0.798

Table 8 Summary of correlation coefficient predictions for cumulative versus inter failure time data

ML techniques	Inter failure data	Cumulative data
ANFIS	0.709	0.999
GRNN	0.897	0.996
FFBPNN	0.645	0.983
CFBPNN	0.635	0.984
Bagging	0.772	0.918
IBK	0.653	0.971
Linear regression	0.868	0.976
Multilayered perceptron	0.807	0.997
REPTree	0.609	0.743
SVM REG	0.817	0.979

Figure 5 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on MSE for the industrial datasets taken into consideration.

Table 9 Summary of MARE predictions for cumulative versus inter failure time data

Mare	Inter failure data	Cumulative data
ANFIS	0.798	0.046
GRNN	0.216	0.122
FFBPNN	0.657	0.45
CFBPNN	0.672	0.489
Bagging	0.506	0.225
IBK	0.288	0.186
Linear regression	0.386	0.188
Multilayered perception	0.175	0.123
REPTree	0.393	0.35
SVM REG	0.289	0.188

Figure 6 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on MARE for the project data [20] taken into consideration.

Figure 7 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on correlation coefficient for the project data [20] taken into consideration.

Figure 8 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on MARE for the project data [24] taken into consideration.

Figure 9 depicts the graphical representation of the analysis of the performances of above-mentioned ML techniques for predicting software reliability based on correlation coefficient for the project data [24] taken into consideration.

Based on the results obtained from rigorous experiments being conducted, we have made following observations regarding this study:

i. ANFIS yields better results as compared to other techniques in predicting failures in terms of statistical efficacy measures undertaken.
ii. ANFIS produces correlation coefficient nearest to +1 which depicts that it shows positive correlation coefficient as compared to all other techniques. The higher the correlation the better the reliability. Hence, we can say that ANFIS predicts failure more accurately than other mentioned techniques.
iii. The techniques GRNN and MLP follow ANFIS in predicting reliability. GRNN also showed to be encouraging and sound. GRNN and MLP produce positive correlation coefficient nearer to 1 and thus can also be used for predicting reliability efficiently and accurately after ANFIS.
iv. From the above results, we also found that ANFIS produces lowest MARE, MRE, and MSE scores as compared to rest of the techniques which again proves it to be better in terms of predicting failures. It shows that ANFIS yields least failure discrepancy between the actual and the predicted failures.

Table 10 Summary of prediction values pred(0.25) for MRE

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Lin Reg	MLP	M5P	REPTree	M5Rules	SVM
Project data	0.964	0.964	0.929	0.964	0.964	0.964	1	0.964	1	1	0.964	1
Project data (release 2)	1	1	0.947	0.947	1	0.947	1	1	1	O.895	0.947	1
Project data (release 1)	1	1	0.95	1	1	1	1	1	1	0.95	1	1
Project data (phase 1)	0.905	1	0.905	1	1	0.952	0.905	0.905	0.905	0.952	0.952	0.905
Project data (phase 2)	1	1	0.81	0.952	1	0.952	0.905	0.952	0.905	0.905	0.905	0.905

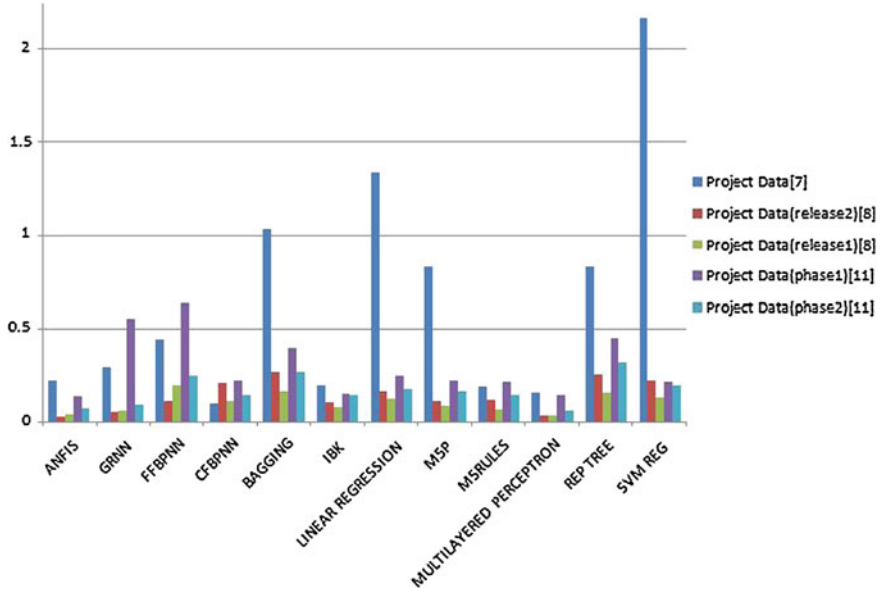

Fig. 2 Comparison of MARE for five different datasets

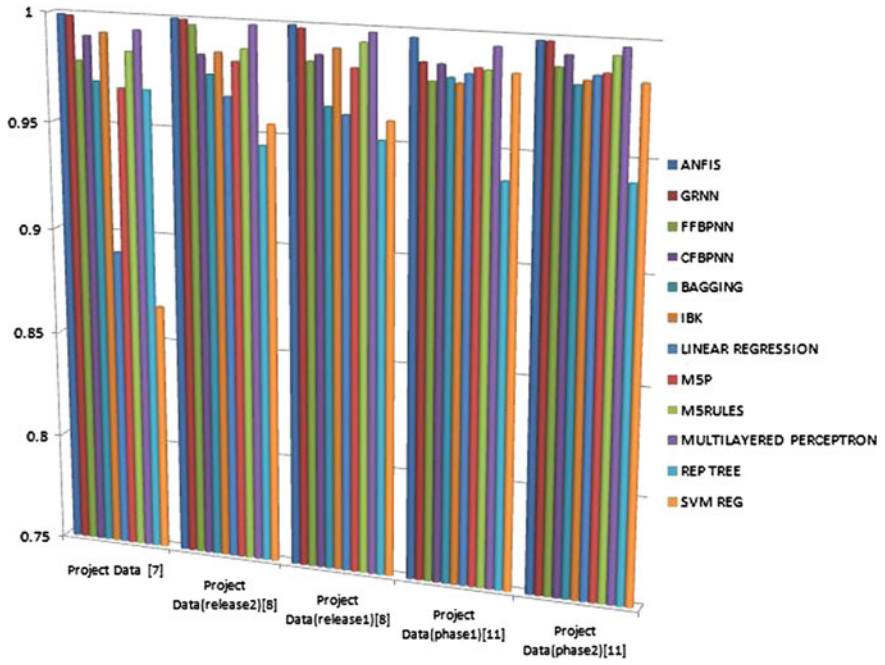

Fig. 3 Comparison of correlation coefficient for five different datasets

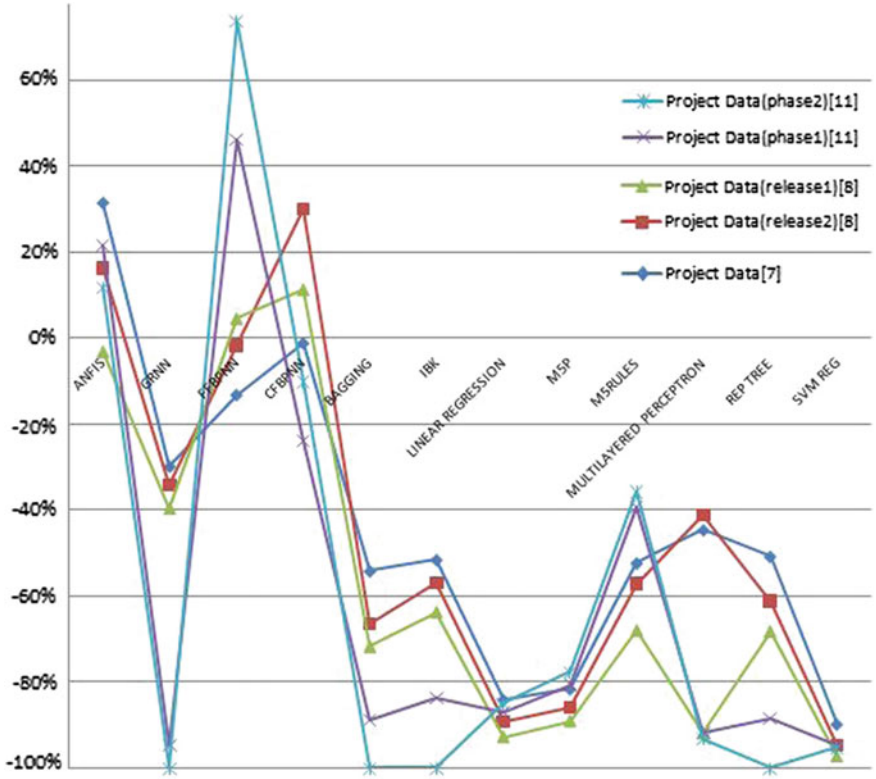

Fig. 4 Comparison of MRE for five different datasets

v. From the results obtained, we also infer that cumulative failures produce more accurate and precise results as compared to inter failure time data. Cumulative failures yield correlation coefficient nearer to 1 unlike inter failure time failures whose correlation coefficient lie more close to −1. Also, we observed that ANFIS yields correlation coefficient nearer to 1 (i.e., 0.9948 and 0.9989) for cumulative failures (project data [20, 24]) as compared to rest of the other mentioned techniques. Based on our experimental evaluation, we also observed that the predicted and the actual values are more closely related for cumulative failures in comparison to inter failure time failures. For cumulative failures, the correlation coefficient ranges from 0.743 to 0.999, which again proves that there is less deviation from actual and predicted values in case of cumulative failures compared to inter failure time failure whose correlation coefficient ranges from −0.043 to 0.897. Thus, it shows that values of correlation coefficient for cumulative failures are more closely related to +1 and hence yields better and accurate results with less deviation between actual and predicted values.

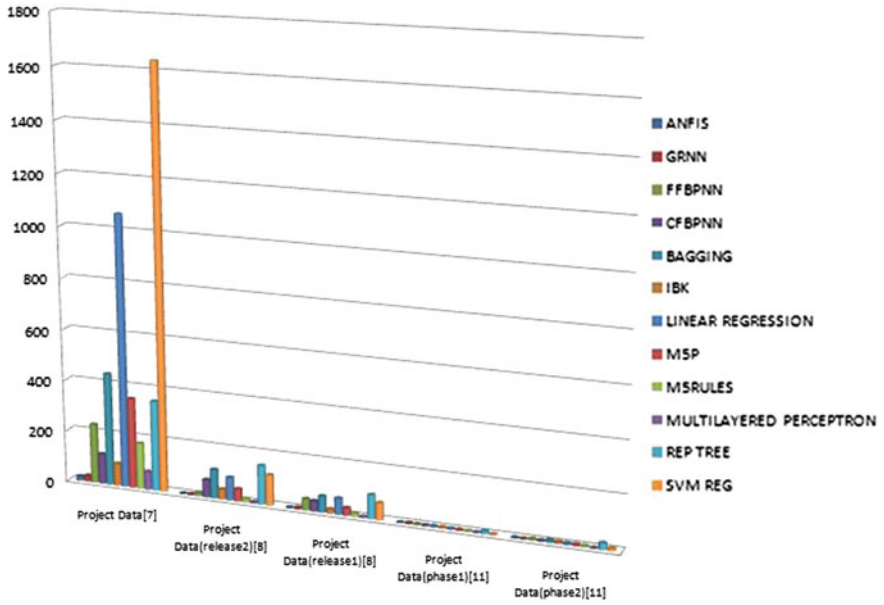

Fig. 5 Comparison of MSE for different datasets

vi. From above, we observed that cumulative failures yield lowest MARE scores as compared to inter failure time data. Out of all, ANFIS gives lowest MARE score (28.49, 4.64 %) for cumulative failures (project data [20, 24]) which again proves it to be the best method among the other mentioned techniques. Based on our experimental evaluation, we also inferred that MARE results ranges from 0.188 to 2.255 for cumulative failures in comparison to inter failure time failure which ranges from 32.054 to 152.986. We have compared MARE results for both the cumulative and inter failure time failure by analyzing it for two different datasets and comparing over ten different techniques. For each of our analysis, we observed that cumulative failures produce better and accurate results with very less percentage of deviation from the actual and the predicted values for software reliability and hence, they are being preferred over inter failure time data as per our study.

7 Threats to Validity

A few of the limitations confronted during the current study are given as follows: One is, the threats to internal validity are present due to the degree to which conclusions can be drawn between independent and dependent variables [28]. The data may not be cumulative and there may be lagging between the values which

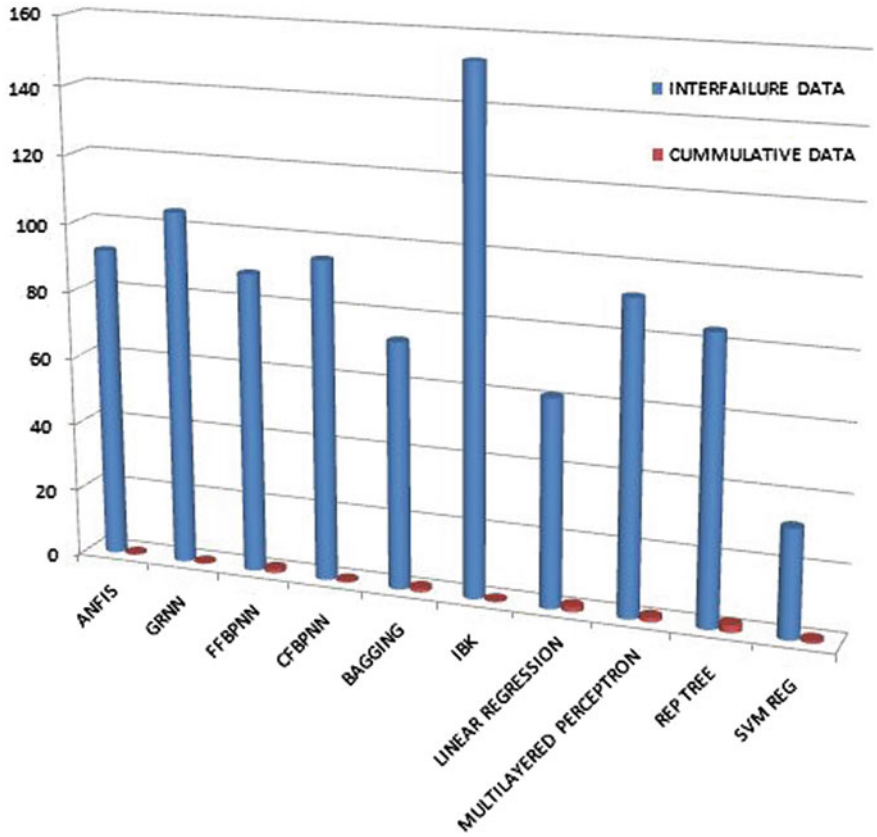

Fig. 6 Comparison of MARE for cumulative versus inter failure time data

need to be addressed. The other is the threats to external validity which are associated with the generalizability of the predicted models. The results in this paper are obtained from open source software WEKA and MATLAB tool, and hence these may not be applicable to other systems. In other terms, the effectiveness of the reliability prediction models depend on the operational environment. The size of the dataset is also not very large. These threats can be minimized by conducting more number of replicated studies across the various systems. Eventually, in spite of all these constraints and limitations, the findings of our work provide the guidance for the future research in order to assess the impact of past failure datasets for the prediction of software reliability using ML techniques.

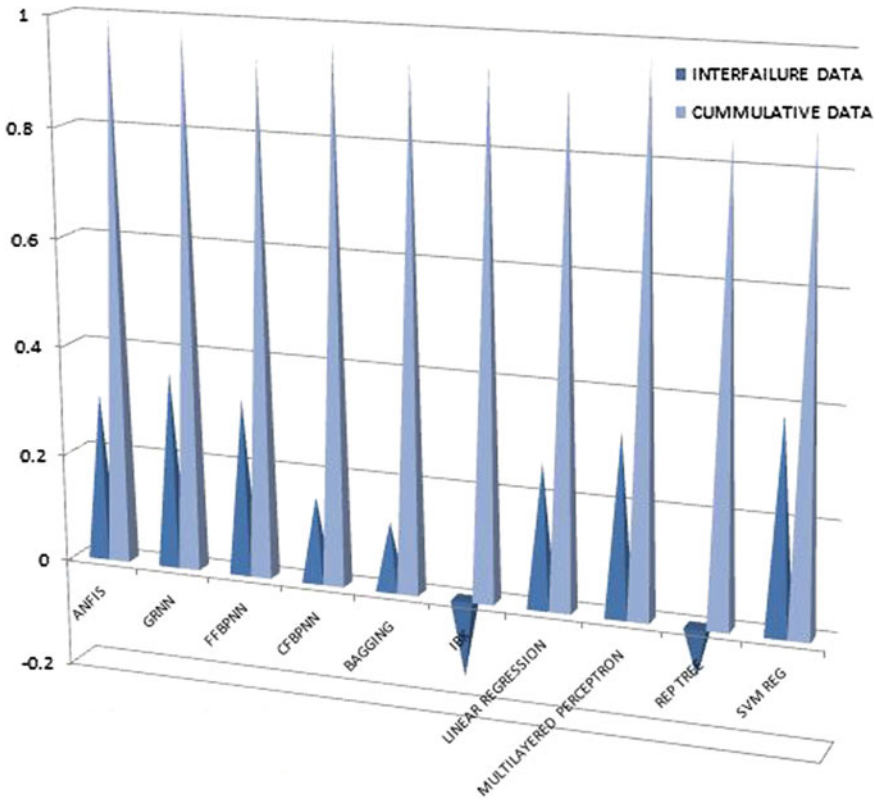

Fig. 7 Comparison of correlation coefficient for cumulative versus inter failure time data

8 Conclusions

Software reliability is dynamic and stochastic in nature so we may say that reliability is a probabilistic measure that assumes that the occurrence of failure of software is a random phenomenon [2]. In this paper, we have applied various machine learning techniques for predicting software reliability based on past failures of software products. The performance of these ML techniques has been evaluated using five different types of datasets being extracted from industrial data to predict the failure intensity of the softwares in use. For each of the five datasets taken into consideration, results show that the correlation coefficient is above 0.99 in most of the predictions for ANFIS which signifies that the actual and the predicted values are very close. Also, we found that the MARE is between the ranges of 0.025–1.5 in most of the predictions for ANFIS and is quite lowest in comparison to the MAREs and is being calculated by other mentioned techniques. The results also depict that the MSE ranges between 0.5 and 16.0 in most of the predictions for ANFIS and MRE and is quite lowest in comparison to the other

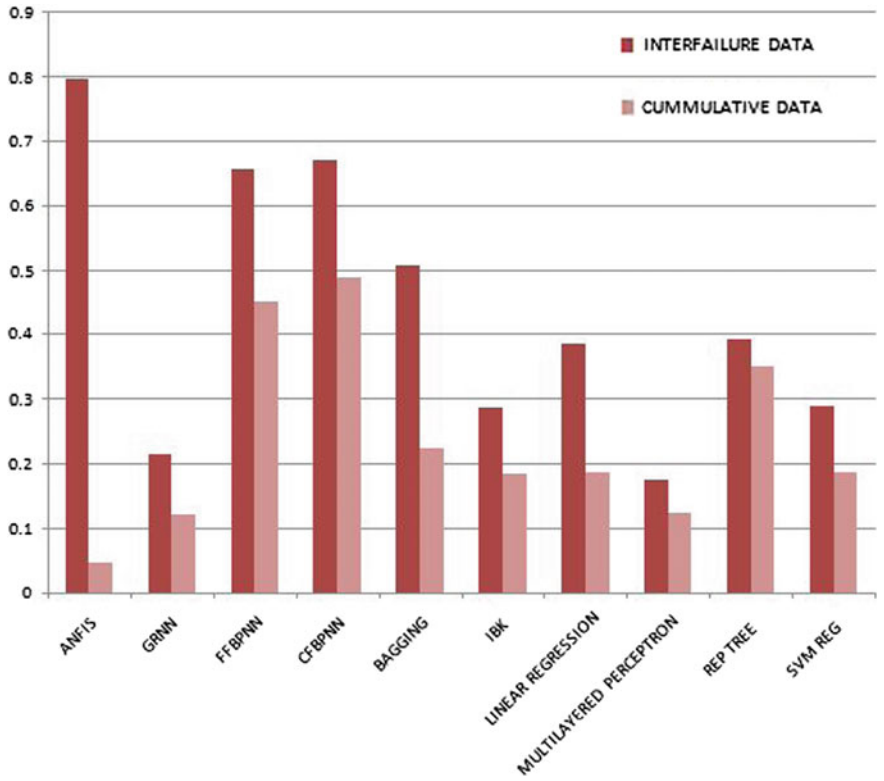

Fig. 8 Comparison of MARE for cumulative versus inter failure time data

mentioned techniques. The result shows that the cumulative failures yield high correlation coefficients within the ranges of 0.74–0.99 in comparison to the correlation coefficients being calculated for inter failure time data. The results also depict that the cumulative failures yield low MARE within the ranges of 0.04 to 1.38 in comparison to the MAREs being calculated for inter failure time data. This is the reason that cumulative failure data is always chosen for failure prediction experiments.

For the further work, more techniques like dynamic neuro fuzzy inference system (DENFIS), group method of data handling (GMDH), and probabilistic neural networks (PNN) can be applied to the data. More datasets can be collected and validated by applying various other machine learning techniques for failure predictions. Further research is planned in an attempt to combine above-mentioned models with other machine learning techniques so as to develop prediction models which can predict the reliability of software more accurately with least precision errors.

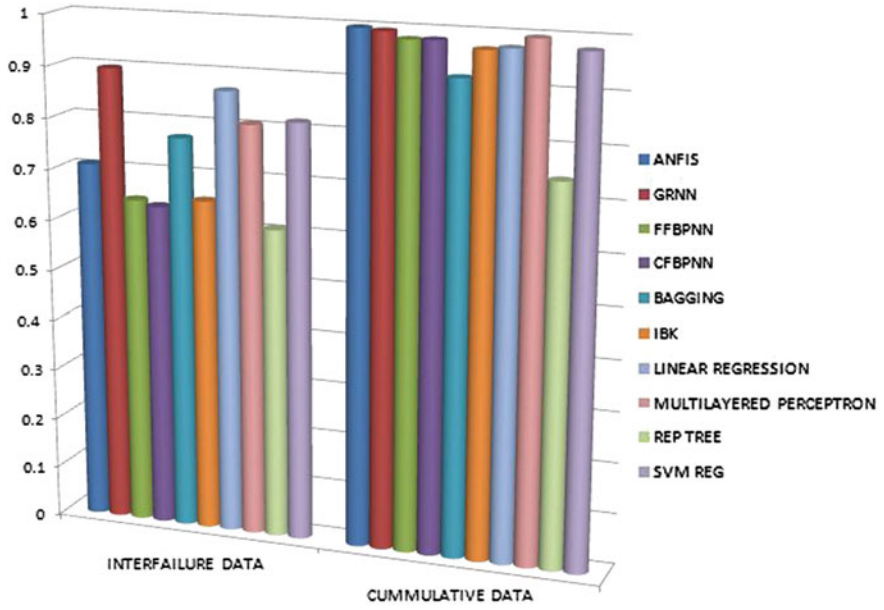

Fig. 9 Comparison of correlation coefficient for cumulative versus inter failure time data

References

1. Standards Coordinating Committee of the IEEE Computer Society, IEEE Standard Glossary of Software Engineering Terminology, IEEE-STD-610.12-1990, IEEE, New York (1991)
2. Quyoum, A., Din Dar, UdM., Quadr, S.M.K.: Improving software reliability using software engineering approach—a review. Int. J. Comput. Appl. **10**(5), 0975– 8887 (2010)
3. Aggarwal, K.K., Singh, Y., Kaur, A., Malhotra R.: Investigating the effect of coupling metrics on fault proneness in object-oriented systems. Softw. Qual. Prof. **8**(4), 4–16 (2006)
4. Goel, B., Singh, Y.: An empirical analysis of metrics. Softw. Qual. Prof. **11**(3), 35–45 (2009)
5. Singh, Y., Kumar, P.: A software reliability growth model for three-tier client–server system. Int. J. Comp. Appl. **1**(13), 9–16, doi:10.5120/289-451 (2010a)
6. Singh, Y., Kumar, P.: Determination of software release instant of three-tier client server software system. Int. J. Comp. Appl. **1**(3), 51–62 (2010b)
7. Singh, Y., Kumar, P.: Application of feed-forward networks for software reliability prediction. ACM SIGSOFT, Softw. Eng. Notes 35(5), 1–6 (2010c)
8. Xingguo, L., Yanhua, S.: An early prediction method of software reliability based on support vector machine. In: Proceedings international conference on wireless communications, networking and mobile computing (WiCom'07), pp. 6075–6078 (2007)
9. Malhotra, R., Kaur, A., Singh, Y.: Empirical validation of object-oriented metrics for predicting fault proneness at different severity levels using support vector machines. Int. J. Syst. Assur. Eng. Manag. **1**(3), 269–281. doi:10.1007/s13198-011-0048-7 (2011)
10. Hua Jung, L.: Predicting software reliability with support vector machines. In: Proceedings of 2nd International Conference on Computer Research and Development (ICCRD'10), Kuala Lumpur, Malaysia, pp. 765–769 (2010)
11. Karunanithi, N., Whitley, D., Malaiya, Y.: Prediction of software reliability using connectionist models. IEEE Trans. Softw. Eng. **18**(7), 563–574 (1992)

12. Singh, Y., Kumar, P.: Prediction of software reliability using feed forward neural networks. In: Proceedings of Computational Intelligence and Software Engineering (CiSE'10), Wuhan, China, pp. 1–5. doi:10.1109/CISE.2010.5677251 (2010d)
13. Singh, Y., Kumar, P.: Application of feed-forward networks for software reliability prediction. ACM SIGSOFT, Softw. Eng. Notes **35**(5), 1–6 (2010c)
14. Eduardo, OC., Aurora, TR., Silvia, RV.: A genetic programming approach for software reliability modeling. IEEE Trans. Reliab. **59**(1), 222–230 (2010)
15. Cai, Y.K., Wen, Y.C., Zhang, L.M.: A critical review on software reliability modeling. Reliab. Eng. Syst. Saf. **32**(3), 357–371 (1991)
16. Specht, F.D.: A general regression neural network. IEEE Trans. Neural Netw. **2**(6), 568–576 (1991)
17. Ho, S.L., Xie, M., Goh, T.N.: A study of connectionist models for software reliability prediction. Comput. Math. Appl. **46**(7), 1037–1045 (2003)
18. Su, S.Y., Huang, Y.C.: Neural network-based approaches for software reliability estimation using dynamic weighted combinational models. J. Syst. Softw. **80**(4), 606–615 (2006)
19. Madsen, H., Thyregod, P., Burtschy, B., Albeanu, G., Popentiu, F.: On using soft computing techniques in software reliability engineering. Int. J. Reliab. Qual. Saf. Eng. **13**(1), 61–72 (2006)
20. Pai, F.P., Hong, C.W.: Software reliability forecasting by support vector machines with simulated vector machines with simulated annealing algorithms. J. Syst. Softw. **79**, 747–755 (2006)
21. Hu, Q.P., Dai, Y.S., Xie, M., Ng, S.H.: Early software reliability prediction with extended ANN model. In: Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC '06), vol. 2, pp. 234–239, Sept 2006
22. Wood, A.: Predicting software reliability. IEEE Tandem Comput. **29**(11), 69–77 (1996)
23. Zhang, X., Jeske, D.R., Pham, H.: Calibrating software reliability models when the test environment does not match the user environment. Appl. Stoch. Models Bus. Ind. **18**, 87–99 (2002)
24. Ohba, M.: Software reliability analysis models. IBM J. Res. Dev. **21**(4) (1984a)
25. Kohavi, R.: The power of decision tables. In: The Eighth European Conference on Machine Learning (ECML-95), Heraklion, Greece, pp. 174–189 (1995)
26. Aljahdali, S.H., Buragga, K.A.: Employing four ANNs paradigms for software reliability prediction: an analytical study. ICGST-AIML J. **8**(2), 1687–4846 (2008)
27. Kumar, P., Singh, Y.: An empirical study of software reliability prediction using machine learning techniques. Int. J. Syst. Assur. Eng. Manag. **3**(3), 194–208. doi:10.1007/s13198-012-0123-8 (2012)
28. van Koten, C., Gray, A.R.: An application of Bayesian network for predicting object-oriented software maintainability. In: The Information Science Discussion Paper, Series Number 2005/02, pp. 1172–6024 (2005)