Int J Syst Assur Eng Manag (February 2018) 9(1):230–244
https://doi.org/10.1007/s13198-016-0543-y

ORIGINAL ARTICLE

Software reliability prediction using machine learning techniques

Arunima Jaiswal[2] · **Ruchika Malhotra**[1]

Received: 13 May 2016 / Revised: 22 August 2016 / Published online: 24 November 2016

Abstract Software Reliability is indispensable part of software quality and is one amongst the most inevitable aspect for evaluating quality of a software product. 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 ML 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 back propagation neural network, general regression neural network, support vector machines, multilayer perceptron, Bagging, cascading forward back propagation neural network, instance based learning, linear regression, M5P, reduced error pruning tree, 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 in all the cases and thus can be used for predicting software reliability since it predicts the reliability more accurately and precisely as compared to all other above mentioned techniques. In this study, we also made comparative analysis between cumulative failure data and inter failure time's data and found that cumulative failure data gives better and more promising results as compared to inter failure time's data.

Keywords Software reliability · Assessment · Prediction · Machine learning techniques

1 Introduction

In the IEEE Standard Glossary of software engineering terminology, software reliability is defined as "The ability of the software to perform its required function under stated conditions for a stated period of time" (Standards Coordinating Committee of the IEEE Computer Society 1991). With the rapid growth and increasing complexity of the software, the software reliability is hard to achieve. Reliability is one of the important and crucial aspect and attribute 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" (Quyoum et al. 2010). Software Reliability Growth Models (SRGM) has been used for predicting and estimating number of errors in the software. The primary goal of software reliability modeling is to find out the probability of a system failing in given time interval or the expected time span between successive failures.

ML techniques have proved to be successful in predicting better results than statistical methods and can be used for prediction of software failures more accurately and precisely Malhotra and Negi (2013). These techniques envisages past failure data as input and quite less assumption is required for modeling of the software's with

✉ Arunima Jaiswal
arunimajaiswal@gmail.com

Ruchika Malhotra
ruchikamalhatra2004@yahoo.com

[1] Department of Software Engineering, Delhi Technological University, Bawana Road, Delhi 110042, India

[2] Department of Computer Science and Engineering, Amity University, Noida, Uttar Pradesh 201313, India

complex phenomena. 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 (Aggarwal et al. 2006; Goel and Singh 2009; Singh and Kumar 2010a, b, c).

In this study, we present an empirical study of various ML techniques such as ANFIS, FFBPNN, GRNN, SVM, MLP, Bagging, CFBPNN, IBK, Lin Reg, M5P, RepTree, M5Rules for predicting software reliability based on five industrial datasets. Thereafter, we investigate about the accuracy and performances of ML based models in predicting the Software Reliability when applied to past failure week data Zhou et al. (2010). We also performed a comparative analysis between cumulative failure data and inter failure time's data to investigate the type of failure data more appropriate for reliability prediction and inferred that cumulative data yields better results and is always preferred over inter failure time's data Tian and Noore (2005). The results show that correlation coefficient for cumulative data always yields positive linear relationship which shows strong correlation between the actual and the predicted values for reliability prediction.

The rest of the paper is organized as follows. Section 2 includes the objectives of the study. Section 3 summarizes the related research work conducted on software reliability prediction. Section 4 provides overview about the research background. Section 5 focuses on the various research methodologies used in predicting Software Reliability. Section 6 includes the results of study, analysis and discussion of the results. Section 7, highlights the threats to validity and finally Sect. 8 concludes the paper. Followed by references.

2 Study objective

Business applications which are critical in nature require reliable software, but developing such software's is a key challenge which our software industry faces today. With the increasing complexity of the software these days, achieving software reliability is hard to achieve. SRGM has been used for predicting and estimating number of errors remaining in the software. The primary goal of software reliability modeling is to find out the probability of a system failing in given time interval or the expected time span between successive failures. In our study, we attempt to empirically assess the use of ANFIS for predicting the software failures. Other ML techniques used for predicting software reliability are FFBPNN, GRNN, SVM, MLP, Bagging, CFBPNN, IBK, Lin Reg, M5P, RepTree, M5Rules. Although ANN, SVM etc. has been previously used in literature (Xingguo and Yanhua 2007) but for the first time ANFIS has been applied to cumulative week failure dataset. In this work, in order to make the most realistic and efficient comparison we have also analyzed the same data sets for above mentioned ML techniques. The background of using ANFIS was that if it had proven empirically to predict the software failures with least errors in comparison to other above mentioned techniques, then it may possibly be used as a sound alternative to other mentioned existing techniques for software reliability predictions. Also other above mentioned ML techniques were empirically analyzed for the first time together on five different types of data sets 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. (Malhotra et al. 2011; Xingguo and Yanhua 2007; Hua Jung 2010; Karunanithi et al. 1992; Singh and Kumar 2010d; Eduardo et al. 2010; Cai et al. 1991; Specht 1991). Karunanithi et al. (1992) carried out analysis of detailed study to explain the use of connectionist models in the reliability growth prediction for the software's. Cai et al. (1991) 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. (2003) 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 modes. Su and Huang (2006) had applied neural network for predicting software reliability. Madsen et al. (2006) focused on the application of Soft Computing techniques for software reliability prediction. Pai and Hong (2006) 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. Lou et al. (2009) discusses about the software reliability prediction using relevance vector machine. Yang et al. (2010) develops a hybrid model using model mining techniques and genetic algorithms for software reliability prediction. Lo (2011) discusses about the utilization and applicability of Auto-Regressive Integrated Moving Average technique and SVM for reliability prediction. Kumar and Singh (2012) discusses about the usage of machine learning techniques like cascade correlation neural network, decision trees and fuzzy inference system to predict the

reliability of software products. Torrado et al. (2013) described the usage of Bayesian model together with Gaussian processes to estimate and predict the number of software failures over time. Park et al. (2014) talks about the applicability of data driven methods to identify an appropriate multi-step prediction strategy for software reliability. Liu et al. (2015) discusses the applicability of hybridization of Singular Spectrum Analysis method and Auto Regressive Integrated Moving Average methodology for prediction of medium and long-term software failures. Lou et al. (2016) carried out study to estimate future occurrences of software failures to aid in maintenance and replacement using Relevance vector machines which are kernel-based learning methods.

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) (Hu et al. 2006; Pai and Hong 2006). We also use the data collected from Tandem Computers Software Data Project. This set of failure data, is from two of four major releases of software products at Tandem Computers (Project Data) (release 1 and 2) (Wood 1996). Other data sets include Telecommunication System Data. This data set was reported by (Project Data) (phase 1 and 2) (Zhang et al. 2002) based on system test data for a telecommunication system. System test data consisting of two releases (Phases 1 and 2). In both tests, automated test and human-involved tests are executed on multiple test beds. Also we have used data from the project On-Line Data Entry IBM Software Package. The data reported by (Project Data) (Ohba 1984) are recorded from testing an on-line data entry software package developed at IBM. All the datasets contain weeks as testing time input and failures as outputs.

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. As the number of faults changes, the failure rate changes accordingly. In this study, we predict the dependent variable based on the number of failures to be detected using various ML techniques. For the corresponding failures, testing time in terms of weeks is chosen to be the independent variable.

5 Methodologies adopted

In this study we explored various ML techniques such as ANFIS, FFBPNN, GRNN, SVM, MLP, Bagging, CFBPNN, IBK, Lin Reg, M5P, RepTree, M5Rules for predicting failures. We have divided entire dataset into two parts: training and testing data set. The training data set is applied to the ML models for predicting software reliability. The training and testing dataset selection is being employed using k-fold cross validation procedure where the entire data set is divided randomly into k subsets ($k = 10$) and every time one of the k subsets is used as training data and the remaining ($k - 1$) subsets are being used as testing data set so as to validate the prediction model for software reliability. Here, 10 cross validation is used where nine parts are used for training and one part is used for validation taken randomly ten times and results are recorded for each of the ten 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 data sets by repeatedly resampling the same data set randomly by reordering it and then splitting it into tenfolds of equal length (Kohavi 1995).

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 fuzzy inference system which establishes input output relationship using if then else rules through back propagation method. It 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 (Aljahdali and Buragga 2008). The FIS structure was generated using genfis1 function from the Matlab Fuzzy logic Toolbox (http://www.mathworks.com). The basic steps of the FIS model are: (1) identification of input variables (failure time) and output variable (cumulative failures) (2) development of fuzzy profile of the input/output variables (3) 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 (Kumar and Singh 2012). The adaptive neuro-fuzzy learning techniques work in a similar manner to that of the neural networks. Fuzzy Logic Toolbox calculates the membership function parameters in such a manner that these parameters best permit the associated FIS (Fuzzy Inference System) to follow the input/output data.

5.1.2 Cascade-forward back propagation neural networks (CFBPNN)

These are a type of neural networks consisting of more than one layer of neurons. The weights of first layer originate from the input. The weights of each of the subsequent layer originate from the input and all layers prior to the layer in question. Each layer has biases. The last layer is called the network output. Using the neural network toolbox of MATLAB, weights and biases of each of the layers are initialized. Adaption is a process which updates weights with the stipulated learning function. First, adaption is done 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. The advantage of the additional connections is that they sometimes improve the speed at which the model is trained.

5.1.3 Feed-forward back propagation neural networks (FFBPNN)

It contains more than one layer of neurons, just like the cascade-forward back propagation neural networks. A methodology similar to the cascade forward neural networks 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) permit 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)

It is one of a kind of probabilistic neural network. A probabilistic neural network is quite beneficial because of its ability to adapt to the basic function the data follows even when only a few training data samples are available. It is used for function approximation. A generalized regression neural network consists of a radial basis layer along with a special linear layer. A special parameter called 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 back propagation 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. A more complex model would comprise of dozens of input variables. 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. To create the model, we use WEKA tool.

5.1.7 M5P

It is a technique based on Quinlan's M5 algorithm. Based on linear regression at the nodes this technique merges a schematic decision tree with a possibility of functions. M5P generates M5 model trees, combining a conventional decision tree with the incorporation of linear regression functions at the leaves. Initially in order to build a tree, a decision-tree induction algorithm is executed first. A splitting criterion is then applied. The splitting criterion, along each branch, minimizes intra-subset variation in class values. This splitting procedure is 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. If an inner node is pruned, that node is then converted into a leaf having a regression plane.

5.1.8 M5Rules

This technique is also based on M5 algorithm. For problems based on regression, 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)

This technique was being propounded as a method for classifying data, and is 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 machine learning ensemble. This method is being employed to improve the accuracy as well as stability of the machine learning algorithms which are generally used in statistical regression and/or classification. It reduces the variance and helps avoiding over fitting issue. Usually, it is used with decision tree methods, but it can be applied to other types of methods as well.

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. Missing values are dealt with using other methods such as C4.5's method. REPTree produces a suboptimal tree under the constraint that a sub-tree can only be pruned if it does not contain a sub-tree with a lower classification error than itself.

5.1.12 IBk (instance based K nearest neighbor)

It is a technique incorporated in WEKA. It employs k-nearest neighbor algorithm in order to classify data. It is one of the types of lazy learning. 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 prediction methodology adopted is being illustrated through a flow chart in Fig. 1.

In order to conduct this experiment, foremost requisite is the selection of independent and dependent variables. The dependent variable which we have used in this work is failure rate and the independent variable used is time interval in terms of weeks. In this study, we predict the dependent variable based on the number of failures to be

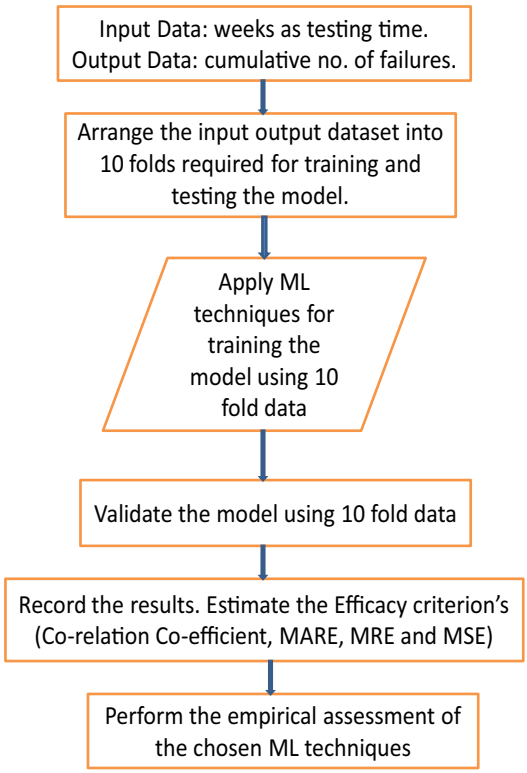

Fig. 1 Flowchart of software reliability prediction methodology adopted

Table 1 Efficacy measures used for evaluating performance criteria

Efficacy measure	Definition
Correlation coefficient	Correlation coefficient measures the intensity of relationship or agreement of predictions with the actual class. This statistics shows that how closely actual and predicted values are correlated with each other. The correlation coefficient lies between −1 and +1. The positive linear relationship grows stronger as correlation coefficient nears 1, and the negative linear relationship grows stronger as correlation coefficient nears −1. No linear relationship is there if the correlation coefficient is 0
MARE	Mean absolute relative error is a measure of the relative differences between the actual values and the values of the output the model is predicting and 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 another measure of the relative differences between the actual values and the values of the output the model is predicting and 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

detected using various ML techniques. For the corresponding failures, testing time in terms of weeks is chosen to be the independent variable. Next step envisages the application of tenfold cross validation method which divides the data set into tenfolds (nine parts are used for training and one part is used for validation taken randomly ten times) required for training and testing the model. Then ML techniques are applied for training the model using training data. After this, we validate the models trained above 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. Based on the comparative analysis, we found that ANFIS produces better results as compared to all other mentioned techniques in terms of predicting software failures for various chosen datasets.

6 Result analysis

In this section, we present the summary of results obtained for predicting reliability using five data sets which we have taken for comparison using ML techniques in terms of efficacy measures. Efficacy criterions which we have used for model evaluation for software reliability prediction are correlation coefficient between actual and predicted values, MARE, MRE and MSE. Software reliability prediction is phenomena that predict the future failure trends based on the past failure data. These results are then collected and used for analyzing and deciding the appropriate time to release the software.

We obtained the following results for various datasets which we have taken for analysis in our work. The results are being summarized in the following Tables 2, 3, 4, 5, 6, 7, 8, 9 and 10.

Sometimes, MARE, MRE, MSE are expressed in %. However, this paper follows the definition given in Table 1 and does not express MARE in % (van Koten and Gray 2005).

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. The results show that the correlation coefficient is above 0.99 in most of the predictions for ANFIS shows that the actual and the predicted values are very close.

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. The results show that the error is between the ranges of 0.5–16.0 in most of the predictions for ANFIS is quite lowest in comparison to the MSE's is being calculated by other mentioned techniques.

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. The results show that the MARE is between the ranges of 0.025–1.5 in most of the predictions for ANFIS is quite lowest in comparison to the MARE's is being calculated by other mentioned techniques.

Table 5 shows the values of MRE on five datasets taken for reliability analysis in this work using 11 ML techniques and and one statistical method (Lin Reg) method. The results show that the error found in most of the predictions for ANFIS is quite lowest in comparison to the errors is being calculated by other mentioned techniques.

These performance criteria were also being observed for the Cumulative as well as Inter Failure times' data sets. Table 6 shows the results of correlation coefficients on dataset given in Project Data (Pai and Hong 2006) 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 yields high correlation coefficients within the ranges of 0.8–0.9 in comparison to the

Table 2 Summary of correlation coefficient predictions for different datasets

Project data	ANFIS	GRNN	FFBPNN	CFBPNN	Bagging	IBK	Linear regression	M5P	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	0.937	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	M5P	M5RULES	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	M5RULES	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 dat a(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	M5P	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's data

ML techniques	Interfailure data	Cummulative 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	0.087

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

MARE	Interfailure data	Cummulative data
ANFIS	91.963	0.285
GRNN	104.924	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's data

ML Techniques	Interfailure data	Cummulative 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

correlation coefficient's being calculated for interfailure time's data.

Table 7 shows the results of MARE on dataset given in Project Data (Pai and Hong 2006) taken for reliability analysis in this work using 9 ML techniques and one

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

MARE	Interfailure data	Cummulative 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 perceptron	0.175	0.123
REPTree	0.393	0.35
SVM reg	0.289	0.188

statistical method (Lin Reg) method. The results show that the value of MARE is 0.28 for ANFIS prediction. It also depicts that the cumulative failures yields low MARE within the ranges of 0.28–1.38 in comparison to the MARE's being calculated for interfailure time's data.

Table 8 shows the results of correlation coefficients on dataset given in Project Data (Ohba 1984) 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 yields high correlation coefficients within the ranges of 0.74–0.99 in comparison to the correlation coefficient's being calculated for interfailure time's data.

Table 9 shows the results of MARE on dataset given in Project Data (Ohba 1984) taken for reliability analysis in this work using 9 ML techniques and one statistical method (Lin Reg) method. The results show that the value of MARE is 0.04 for ANFIS prediction. It also depicts that the cumulative failures yields low MARE within the ranges of 0.04–0.48 in comparison to the MARE's being calculated for interfailure time's 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.

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.

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 (Pai and Hong 2006) 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 (Pai and Hong 2006) 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 (Ohba 1984) 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 (Ohba 1984) taken into consideration.

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

1. ANFIS yields better results as compared to other techniques in predicting failures in terms of statistical efficacy measures undertaken.
2. ANFIS produces correlation coefficient nearest to +1 which depicts that it shows positive correlation

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	0.895	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

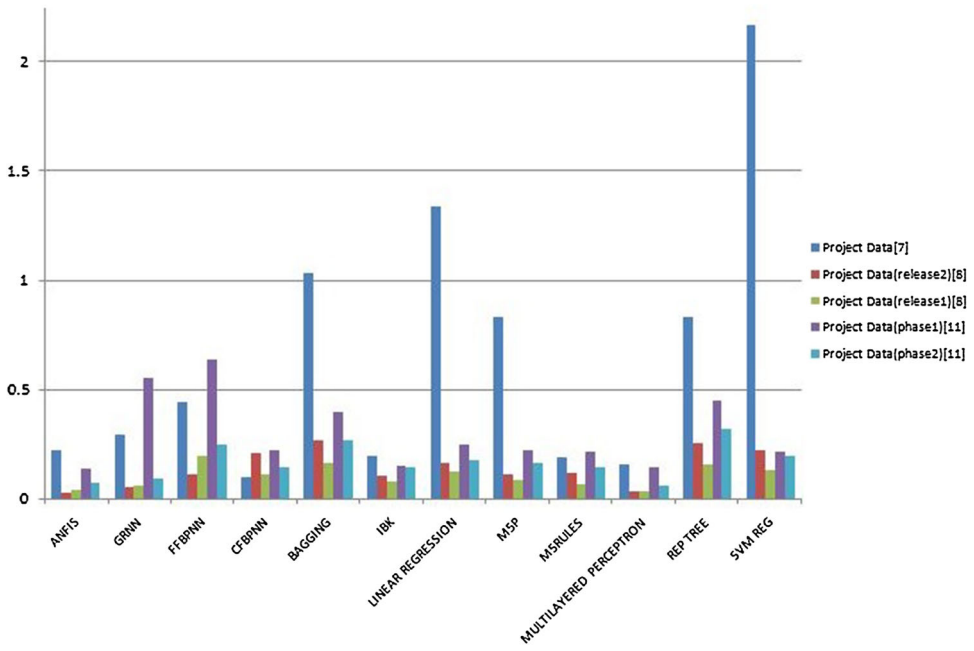

Fig. 2 Comparison of MARE for five different datasets

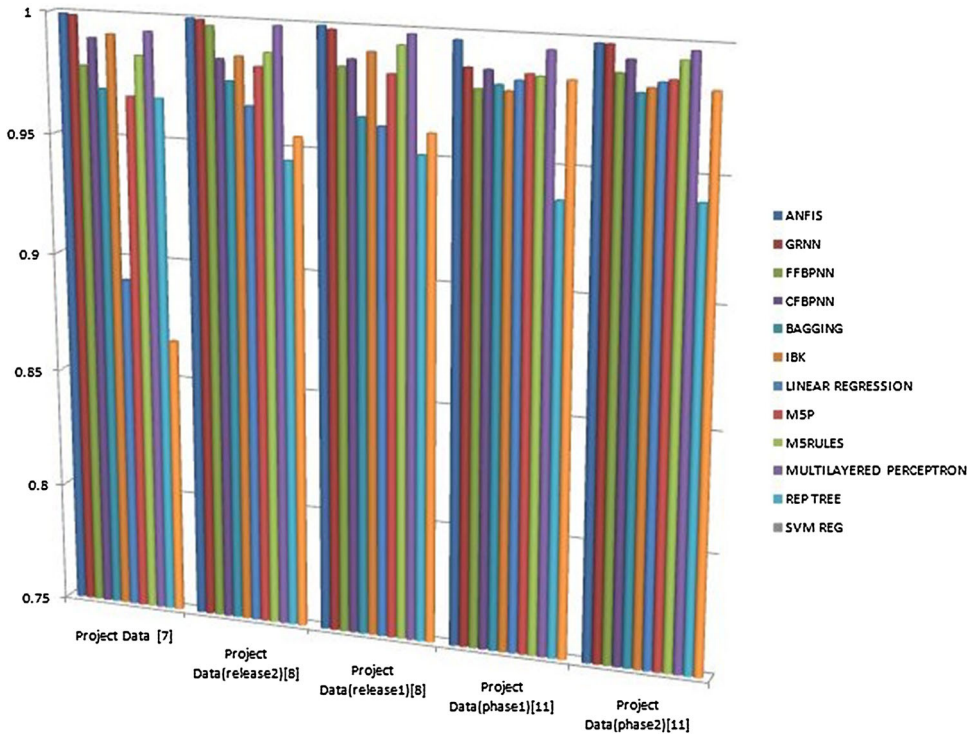

Fig. 3 Comparison of correlation coefficient for five different datasets

coefficient as compared to all other techniques. It also clearly demonstrate that how closely actual and predicted values are correlated with each other. The higher the correlation the better the reliability. Hence we can say that ANFIS predicts failure more accurately than other mentioned techniques.

3. The techniques GRNN and MLP follow ANFIS in predicting reliability. GRNN also showed 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.
4. From the above results, we also found that ANFIS produces lowest MARE, MRE, MSE scores as compared to rest of the techniques which again proves it to be better in terms of predicting failures. It shows that

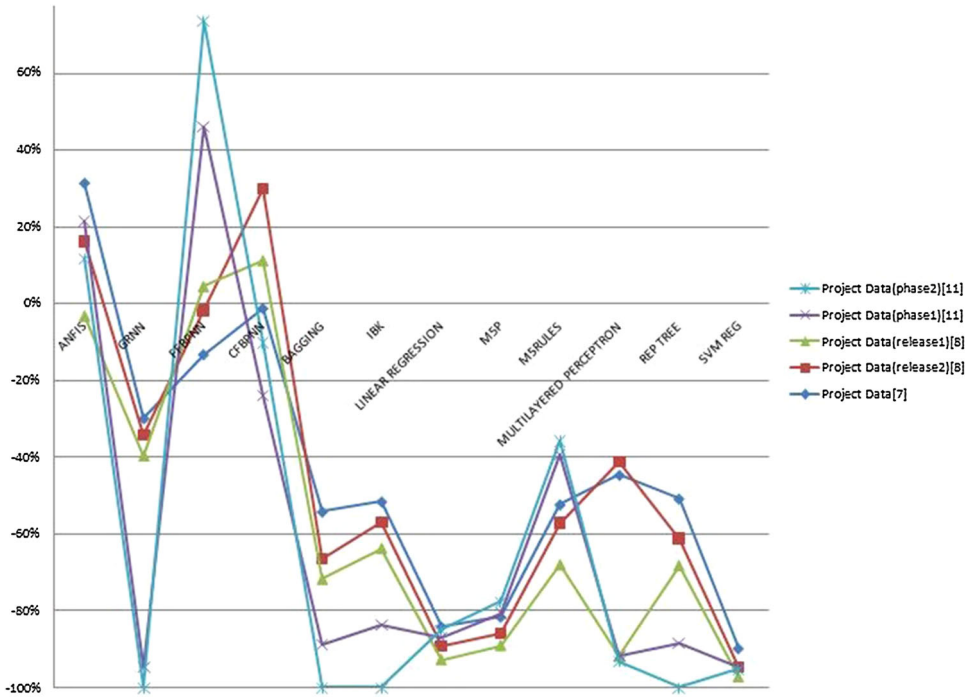

Fig. 4 Comparison of MRE for five different datasets

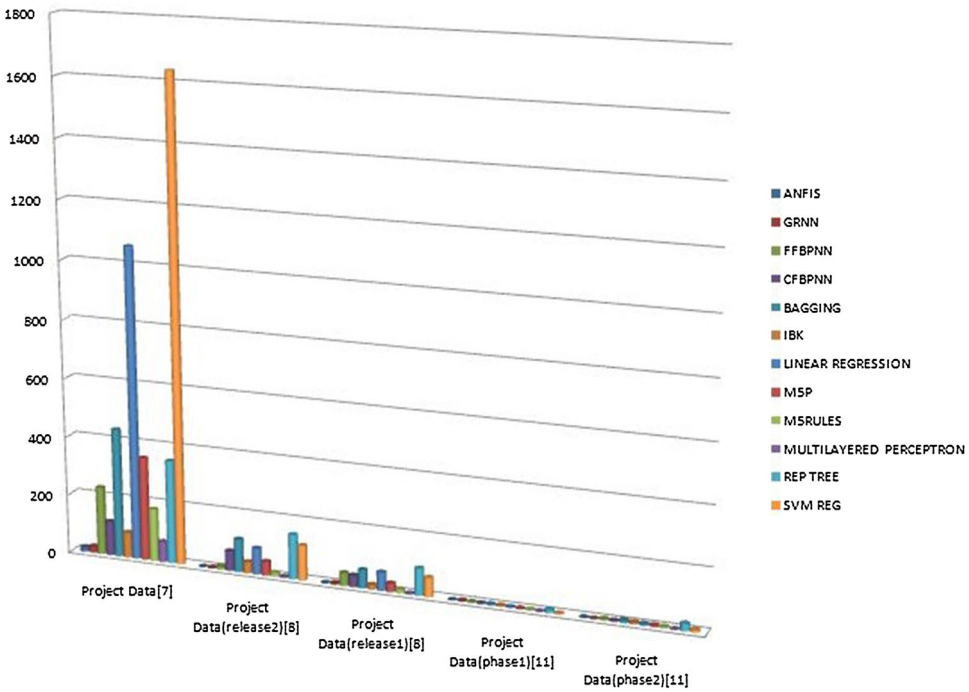

Fig. 5 Comparison of MSE for different datasets

ANFIS yields least failure discrepancy between the actual and the predicted failures. Hence we can conclude that ANFIS predicts reliability more precisely and accurately.

5. From the results obtained, we also infer that cumulative failures produce more accurate and precise results as compared to inter failure time's data. Cumulative failures yields correlation coefficient nearer to 1 unlike inters failure time's failures whose correlation coefficient lie more close to -1. The positive linear relationship grows stronger as correlation coefficient nears 1, and the negative linear relationship grows stronger as correlation coefficient nears -1. It shows that how closely actual and

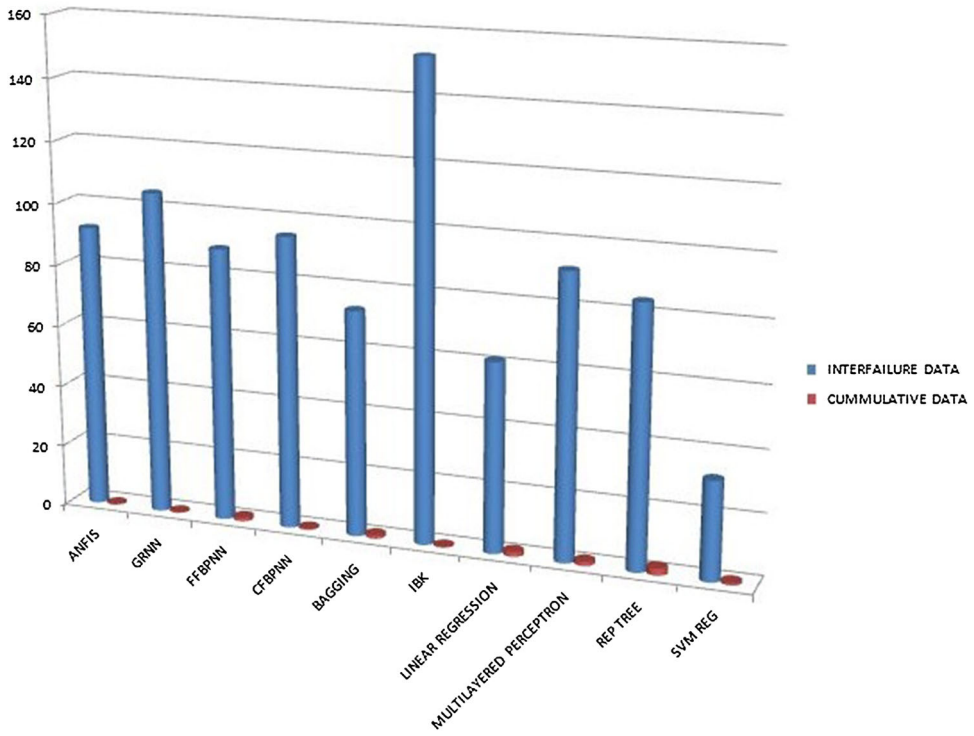

Fig. 6 Comparison of MARE for cumulative versus inter failure time's data

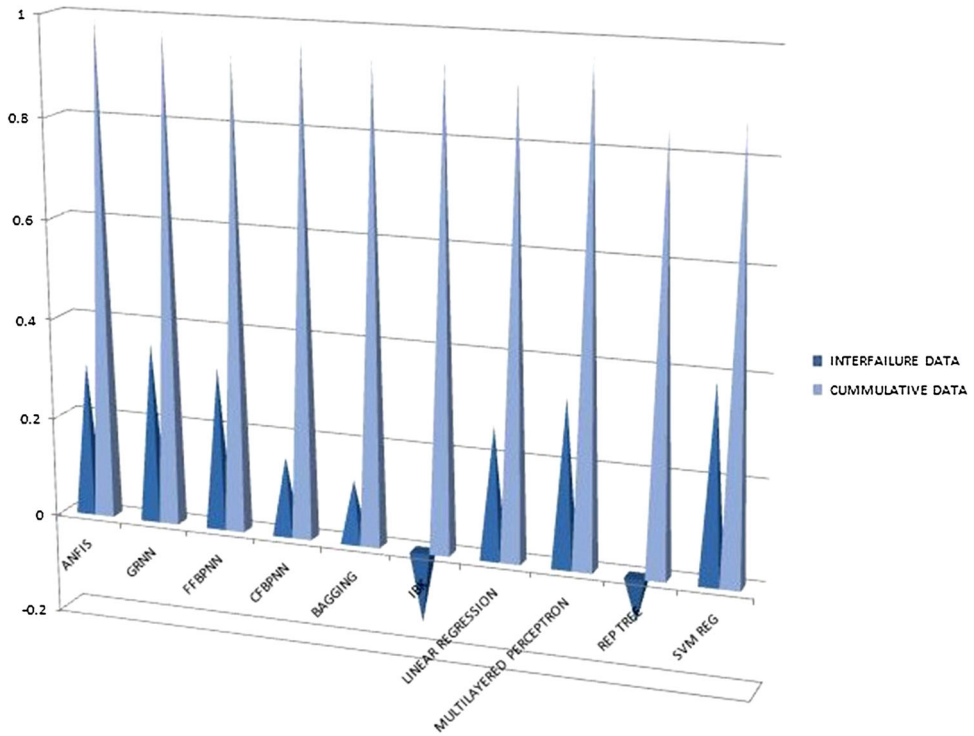

Fig. 7 Comparison of correlation coefficient for cumulative versus inter failure time's data

predicted values are correlated with each other. Also we observed that ANFIS yields correlation coefficient nearer to 1 (i.e. 0.9948 and 0.9989) for cumulative failures [Project Data (Pai and Hong 2006; Ohba 1984)] as compared to rest of the other mentioned techniques.

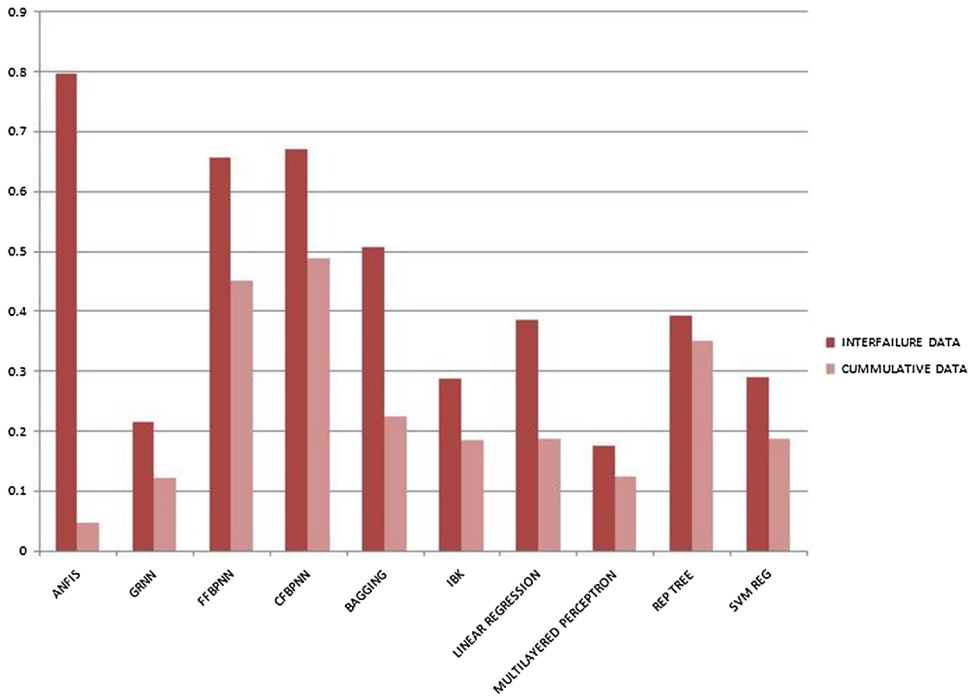

Fig. 8 Comparison of MARE for cumulative versus inter failure time's data

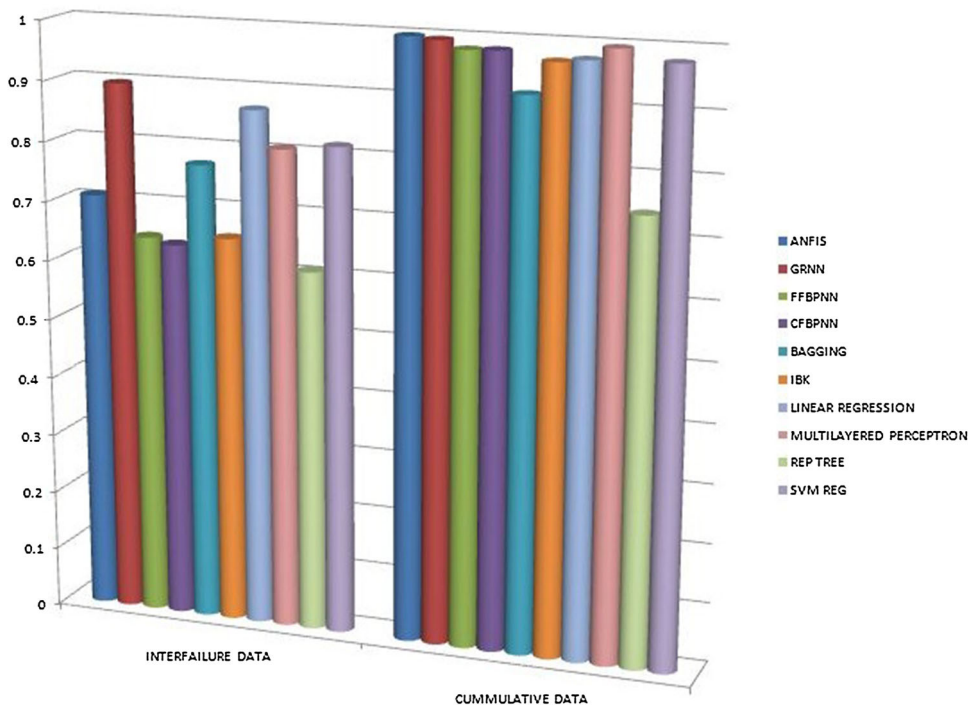

Fig. 9 Comparison of correlation coefficient for cumulative versus inter failure time's data

6. From above, we observed that cumulative failures yields better results than inter failure time's data. They produce lowest MARE scores as compared to inter failure time's data. Out of all, ANFIS gives lowest MARE score (28.49, 4.64%) for cumulative failures [Project Data (Pai and Hong 2006; Ohba 1984)] which again proves it to be the best method amongst the other mentioned techniques.

7 Threats to validity

Like other empirical studies, few of the limitations confronted during the current study are given as follows: there are two types of threats to validity. One is threat to external validity and another is threat to internal validity. The threats to external validity are believed to be more crucial as compared to the threats to internal validity. The threats to internal validity are present due to degree to which conclusions can be drawn between independent and dependent variables (van Koten and Gray 2005). The data may not be cumulative and there may be lagging between the values which need to be addressed. The threats to external validity are the threats 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 depends on the operational environment. The size of the data set 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 future research in order to assess the impact of past failure datasets for the prediction of software reliability using ML techniques.

8 Conclusions

Software reliability is one amongst the important facet of the software quality. Presence of faults/failures makes the software unreliable. 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 (Quyoum et al. 2010). In this paper, we have applied machine learning techniques namely ANFIS, FFBPNN, GRNN, SVM, MLP, Bagging, CFBPNN, IBK, Lin Reg, M5P, RepTree, M5Rules for predicting software reliability based on past failures of software products. The performance of above mentioned Machine Learning techniques have been evaluated using five different types of data sets being extracted from industrial data to predict the failure intensity of the software's in use. We have empirically proved that ANFIS outperformed the model predicted using other mentioned ML techniques for all the datasets being taken into consideration in predicting reliability in terms of the various statistical efficacy measures applied. 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 MARE's is being calculated by other mentioned techniques. The results also depicts that the MSE ranges between 0.5 and 16.0 in most of the predictions for ANFIS and MRE is quite lowest in comparison to the other mentioned techniques. Apart from this, GRNN shows very appreciating and encouraging results and can also be used for reliability prediction. We may also infer that GRNN and MLP follow ANFIS in predicting reliability. Also we observed that cumulative data produces always better results as compared to inter failure time's data. The result shows that the cumulative failures yield high correlation coefficients within the ranges of 0.74–0.99 in comparison to the correlation coefficient's being calculated for interfailure time's data which signifies that the actual and the predicted values are very close. The results also depicts that the cumulative failures yields low MARE within the ranges of 0.04–1.38 in comparison to the MARE's being calculated for interfailure time's data. This is the reason that cumulative failure data is always chosen for failure prediction experiments.

For further work, more techniques like DENFIS (Dynamic Neuro Fuzzy Inference System), GMDH (Group Method of Data Handling), and PNN (Probabilistic Neural Networks) 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.

References

Aggarwal KK, Singh Y, Kaur A, Malhotra R (2006) Investigating the effect of coupling metrics on fault proneness in object-oriented systems. Softw Qual Prof 8(4):4–16

Aljahdali SH, Buragga KA (2008) Employing four ANNs paradigms for software reliability prediction: an analytical study. ICGST AIML J 8(2):1687–4846

Cai YK, Wen YC, Zhang LM (1991) A critical review on software reliability modeling. Reliab Eng Syst Saf 32(3):357–371

Eduardo OC, Aurora TR, Silvia RV (2010) A genetic programming approach for software reliability modeling. IEEE Trans Reliab 59(1):222–230

Goel B, Singh Y (2009) An empirical analysis of metrics. Softw Qual Prof 11(3):35–45

Ho SL, Xie M, Goh TN (2003) A study of connectionist models for software reliability prediction. Comput Math Appl 46(7):1037–1045

Hu QP, Dai YS, Xie M, Ng SH (2006) 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

Hua Jung L (2010) Predicting software reliability with support vector machines. In: Proceedings of 2nd international conference on computer research and development (ICCRD'10), Kuala Lumpur, pp 765–769

Jun-gang L, Jian-hui J, Chun-yan S, Rui Z, Ang J (2009) Software reliability prediction model based on relevance vector machine. IEEE international conference on intelligent computing and intelligent systems, pp 229–233

Karunanithi N, Whitley D, Malaiya Y (1992) Prediction of software reliability using connectionist models. IEEE Trans Softw Eng 18(7):563–574

Kohavi R (1995) The power of decision tables. In: Kohavi R (ed) The eighth European conference on machine learning (ECML-95), Heraklion, pp 174–189

Kumar P, Singh Y (2012) 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

Liu G, Zhang D, Zhang T (2015) Software reliability forecasting: singular spectrum analysis and ARIMA hybrid model. International symposium on theoretical aspects of software engineering, pp 111–118

Lo J-H (2011) A study of applying ARIMA and SVM model to software reliability prediction. International conference on uncertainty reasoning and knowledge engineering, pp 141–144

Lou J, Jiang Y, Shen Q, Shen Z, Wang Z, Wang R (2016) Software reliability prediction via relevance vector regression. Neurocomputing 186:66–73

Madsen H, Thyregod P, Burtschy B, Albeanu G, Popentiu F (2006) On using soft computing techniques in software reliability engineering. Int J Reliab Qual Saf Eng 13(1):61–72

Malhotra R, Negi A (2013) Reliability modeling using particle swarm optimization. The society for reliability engineering, quality and operations management (SREQOM), India and The Division of Operation and Maintenance, Lulea University of Technology, Sweden. Int J Syst Assur Eng Manag. doi:10.1007/s13198-012-0139-0

Malhotra R, Kaur A, Singh Y (2011) 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

Matlab fuzzy logic toolbox: tutorials on fuzzy inference system and ANFIS using MatLab. http://www.mathworks.com. Accessed 14 Feb 2011

Ohba M (1984) Software reliability analysis models. IBM J Res Dev 21(4):428–443

Pai PF, Hong WC (2006a) Software reliability forecasting by support vector machines with simulated annealing algorithms. J Syst Softw 79(6):747–755

Pai FP, Hong CW (2006b) Software reliability forecasting by support vector machines with simulated vector machines with simulated annealing algorithms. J Syst Softw 79:747–755

Park J, Lee N, Baik J (2014) On the long-term predictive capability of data-driven software reliability model: an empirical evaluation. IEEE 25th international symposium on software reliability engineering, pp 45–54

Quyoum A, Dar MD, Quadr SMK (2010) Improving software reliability using software engineering approach—a review. Int J Comput Appl 10(5):0975–8887

Singh Y, Kumar P (2010a) A software reliability growth model for three-tier client–server system. Int J Comp Appl 1(13):9–16. doi:10.5120/289-451

Singh Y, Kumar P (2010b) Determination of software release instant of three-tier client server software system. Int J Softw Eng 1(3):51–62

Singh Y, Kumar P (2010c) Application of feed-forward networks for software reliability prediction. ACM SIGSOFT, Softw Eng Notes 35(5):1–6

Singh Y, Kumar P (2010d) Prediction of software reliability using feed forward neural networks. In: Proceedings of computational intelligence and software engineering (CiSE'10), Wuhan, pp 1–5. doi:10.1109/CISE.2010.5677251

Specht FD (1991) A general regression neural network. IEEE Trans Neural Netw 2(6):568–576

Standards Coordinating Committee of the IEEE Computer Society (1991) IEEE Standard Glossary of Software Engineering Terminology, IEEE-STD-610.12-1990. IEEE, New York

Su SY, Huang YC (2006) Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models. J Syst Softw 80(4):606–615

Tian L, Noore A (2005) Evolutionary neural network modeling for software cumulative failure time prediction. Reliab Eng Syst Saf 87:45–51

Torrado N, Wiper MP, Lillo RE (2013) Software reliability modeling with software metrics data via gaussian processes. IEEE Trans Softw Eng 39(8):1179–1186

van Koten C, Gray AR (2005) An application of Bayesian network for predicting object-oriented software maintainability. The information science discussion paper, series number 2005/02, pp 1172–6024

Wood A (1996) Predicting software reliability. Tandem Comput IEEE 29(11):69–77

Xingguo L, Yanhua S (2007) 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

Yang B, Li X, Xie M, Tan F (2010) A generic data-driven software reliability model with model mining technique. Reliab Eng Syst Saf 95(6):671–678

Zhang X, Jeske DR, Pham H (2002) Calibrating software reliability models when the test environment does not match the user environment. Appl Stoch Models Bus Ind 18:87–99

Zhou Y, Xu B, Leung H (2010) On the ability of complexity metrics to predict fault-prone classes in object-oriented systems. J Syst Softw 83:660–674