World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:9, No:8, 2015

Performance Evaluation of Data Mining Techniques for Predicting Software Reliability

Pradeep Kumar, Abdul Wahid

***Abstract*—**Accurate software reliability prediction not only enables developers to improve the quality of software but also provides useful information to help them for planning valuable resources. This paper examines the performance of three well-known data mining techniques (CART, TreeNet and Random Forest) for predicting software reliability. We evaluate and compare the performance of proposed models with Cascade Correlation Neural Network (CCNN) using sixteen empirical databases from the Data and Analysis Center for Software. The goal of our study is to help project managers to concentrate their testing efforts to minimize the software failures in order to improve the reliability of the software systems. Two performance measures, Normalized Root Mean Squared Error (NRMSE) and Mean Absolute Errors (MAE), illustrate that CART model is accurate than the models predicted using Random Forest, TreeNet and CCNN in all datasets used in our study. Finally, we conclude that such methods can help in reliability prediction using real-life failure datasets.

***Keywords*—**Classification, Cascade Correlation Neural Network, Random Forest, Software reliability, TreeNet.

I. Introduction

IN software reliability engineering, different models have different predictive capabilities and there is no unique model in the literature, which can be applied in all circumstances. Some statistical methods applied to software reliability estimation and prediction are maximum likelihood estimation (MLE), least square estimation (LSE), analysis of variance (ANOVA), linear regression analysis (LRA) and logistic regression [1]-[3].

Machine learning is an approach concerned with the design and development of algorithms that allow computers to evolve the system behavior from experience, training, analytical observations and other means, which results in a system that can continuously self-improve. The machine learning techniques are focused on learning automatically, recognizing complex patterns, and making decisions based on system behavior. This way, the machine is able to learn whenever it changes its structure, program, or data based on its input or in response to the external information in such a manner that it's expected future performance improves [4]-[10].

Dr Pradeep Kumar is an Associate Professor in the Department of Computer Science & Information Technology at Maulana Azad National Urdu University, Hyderabad (Telangana State), Pin- 500032, India (phone: 91-9959829128; e-mail: drpkumar1402@gmail.com).

Dr. Abdul Wahid is Professor & Head, Department of Computer Science & IT at Maulana Azad National Urdu University, Hyderabad (Telangana State), Pin- 500032, India (phone: 91-8297097786; e-mail: wahidabdul76@yahoo.com).

The data mining techniques such as Decision Trees (DTs), Classification and Regression Trees (CART), Random Forest (RF), and TreeNet have been found very effective than classical statistical techniques. Data mining is the process of extracting knowledge from large amounts of complex datasets [11]-[15].

In order to improve the quality of software systems we apply the regression techniques using CART, TreeNet, RF, and Cascade correlation neural network (CCNN) for predicting software reliability. However, the present challenge is to make prediction models even more efficient by incorporating a fairly new technique that can improve the prediction rate realistically and require less computational resources. Therefore, it would be interesting to see which particular method tends to work well and up to what extent quantitatively [16]-[20].

In this paper, we examine the comparative performance of three well-known and widely used data mining techniques for predicting software reliability that can help researchers to build an adequate body of knowledge in order to draw stronger conclusions leading to better theories. We briefly focus on two main issues: (i) how accurately and effectively data mining techniques can be utilized for predicting software reliability in real-life situations (ii) Correlate various commonly used data mining techniques for software reliability predictions since their performance varies when applied to different size failure datasets in realistic operating context.

The contribution of our paper can be summarized as follows. First, we present a comparative analysis of the CART, TreeNet, RF, and CCNN from a methodological and applied perspective. Second, we empirically compare the performance of the models predicted with the help of sixteen datasets from the DACS using two commonly used data mining tools (DTReg and Salford predictive modeling system). Third, we study various pertinent issues (availability of failure data, parameter estimation, and the type of assumptions made for modelling) of real-life projects.

The rest of the article is organized as follows: In Section II, we discuss the significant work from the literature. In Section III, we discuss the research background and in Section IV, we describe the data mining techniques applied to predicting software reliability in detail. The experimental results and observations of our study are discussed in Section V. Finally, the conclusions are drawn in Section VI.

II. Related Work

Several data mining techniques for classification and regression such as Decision Trees (DTs), Artificial Neural

Networks (ANNs), and Support Vector Machines (SVMs) have been applied to predict software reliability in practice. The major challenges of these methods do not lie in their technical soundness, but their validity and applicability in real world applications remain open issues particularly in web-based environments [7], [8], [10].

The effectiveness of neural network based prediction models depends on the behavior of the dataset which may be of fluctuating nature in some cases. Therefore, ANNs may overfit the results while dealing with real-life unknown large datasets. Overfitting occurs usually, when the parameters of a model are tuned in such a way that the model fits the training data pretty well but it has poor accuracy when applied on separate unknown dataset, which is not used for training the model.

SVM is a new methodology based on statistical learning theory, which has been successfully applied to solve nonlinear regression and time series problems. SVM can be represented as a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The applications of SVM in place of traditional techniques have shown remarkable improvement in the prediction of software reliability in recent years. The design of SVM is based on the extraction of a subset of the training data that serves as support vectors and therefore represents a stable characteristic of the data. On the other hand, Decision Trees have been applied in various data mining applications for classification purposes and other related disciplines [8], [12], [14], [15].

The RF is an ensemble of decision trees, with improved performance developed by Leo Breiman at the University of California Berkeley [2]. The RF is comprised of many individual trees designed to operate quickly over large complex datasets. The main advantage of using DTs is their descriptive nature, which allows practitioners to interpret the model's decision easily in comparison to other machine learning techniques like ANNs and SVMs. Although, RF and CCNN have shown their strengths in various real-life applications, these methods have rarely been used for predicting software reliability to the best of our knowledge. Thus it is worthwhile to include these techniques for the predictions of software reliability in our study [15], [24]-[26].

The CCNN is a self-organizing network that deals with the input and output neurons only. The neurons are selected from a pool of candidates and added to the hidden layer during the training process. Moreover, CCNN was developed in an attempt to overcome certain limitations of back-propagation learning algorithm. One of the main limitations of back propagation neural network (BPNN) is the slow learning rate from the training set of input-output samples due to step size and moving target. In the CCNN, input nodes and hidden nodes are connected directly to the output with adjustable weighted connections leading to improved performance of the network [17]-[19].

The CART is a commonly used algorithm that can handle various type of input & output data (nominal, ordinal or continuous) in the prediction model. In comparison to other methods used in our study, TreeNet is insensitive to data errors and takes very little time for pre-processing the data to handle the missing values. This way, TreeNet is more useful to resist the overtraining and therefore it is faster than neural network models [14]-[17]. However, in order to make the generalization of such techniques, more similar data-based empirical studies that are capable of being verified by observations and experiments must be carried out.

III. Research Background

The primary objective of developing a software reliability prediction model is to apply for making decisions about the software such as whether the product can be released in its present state or we require further testing to improve the quality of software systems. The data mining techniques (CART, TreeNet and RF) are found very useful and can be utilized as powerful tools for making accurate decision than statistical techniques like LRA on our test data.

A. Dependent and Independent Variables

Failure rate is the dependent variable used in our study applied to assess and predict the reliability of software systems. As the number of remaining faults change, the failure rate of the program changes accordingly. The dependent variable was predicted based on the number of failures detected during the testing phase. Days of testing time is the independent variable taken in terms of calendar time notations (number of weeks/days/Hrs/min./seconds).

B. Empirical Data Collection

The sanctity of collected failure data depends on how accurately we observe the failure data in a realistic environment of modern computing systems. The failure datasets used of various projects are extracted from the Software Life Cycle Empirical/Experience Database (SLED).

C. Evaluation Criteria for Model

In order to analyze and compare the performance of the data mining methods presented in our study, we apply various statistics such as R-sq. MAE, RMSE and NRMSE [19]-[23] computed as follows:

1. R-square (R-sq.)

R-square is the correlation between the output and target values. This statistical measure shows how well the predicted values from a prediction model fit with the observed value of real-life data.

2. Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|\Pr edicted - Actual\right| \quad (1)$$

MAE is the quantity used to measure how close predictions are to the actual outcomes.

3. Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\Pr edicted - Actual)^2}{n}} \quad (2)$$

where n is the number of observations for corresponding predicted and observed values of the model.

4. Normalized Root Mean Square Error (NRMSE)

$$NRMSE = \sqrt{\frac{\sum_{i=1}^{n}(\Pr edicted - Actual)^2}{\sum_{i=1}^{n}\Pr edicted^2}} \quad (3)$$

NRMSE is a non-dimensional form of the RMSE, which is normalized form of RMSE to the range of the observed data.

D. Experimental Setup

This section presents the experimental set up to illustrate how data mining techniques are used as an approximation tool for predicting software reliability quantitatively. The number of failures is defined as the target variable and the days of failure is taken as predictor to assess the reliability of given software systems. Two commonly used data mining tools namely DTReg and Salford-system predictive tool are applied for training and testing the capability of each prediction model presented in our study [16]-[18].

IV. Research Methodology

In this section, we discuss in brief the regression techniques using CART, TreeNet, RF, and CCNN for predicting software reliability. These data mining methods can be utilized to deal with the issues of how to build and design computer programs that improve their performance for some specific task such as reliability prediction based on statistical observations. The performance analysis of various data mining methods are discussed and illustrated in Tables I-VIII summarizing how accurately the data mining techniques are useful and able to predict the reliability of software systems based on past failures.

A. CART

The classification and regression tree generates a binary decision tree. Even if the input variable is nominal and has more than two categories then it groups different categories in one branch. However, when the input variable is continuous (testing time in our study), it still generates two branches associating a set of values limited by the relational operators to each one of them such as less than or equal to or greater than a certain value. Thus, CART is a non-parametric statistical method used for analyzing classification tasks either from categorical or continuous dependent variables. If the dependent variable is categorical then CART produces a classification tree otherwise a regression tree in case of when the dependent variable is continuous [24]-[27].

In data mining, decision trees are used as the predictive model mapping observations of an item to its target values. In a tree structure, the leaves represent classification and branches represent conjunctions of features that lead to the classification. Therefore, a decision tree can be utilized as a data mining model for predicting software reliability. Thus, CART is a decision-tree tool in data mining, which can be applied for pre-processing the data and predictive modeling. The CART has the capability of searching important patterns and relationships in highly complex datasets also. Thus, CART is capable to generate more reliable prediction results from past failure datasets of real-life projects. The CART can handle missing values also by substituting surrogate splitters [2]. We employed CART 6.0, Windows based decision tree tool for data mining & predictive modeling, and data pre-processing which can be found at http://www.salford-systems.com [16]. Tables I and II illustrate the architecture and performance measurements of CART for the assessment and prediction of software reliability using sixteen failure datasets in terms of R-sq., MAE, RMSE, and normalized RMSE.

TABLE I
Architecture of CART

Parameters	CART Structure
Initial value of the complexity parameters	0
Construction rule	Least square method
Estimation method	10-fold cross-validation
Number of predictor variable	1
Minimum size below which node will not be split	10
Minimum size for a child node	1
Maximum no. of cases allowed in the learning sample	Maximum rows
Maximum number of surrogates used for missing values	1

TABLE II
Summary of Predictions for Different Data Sets using CART

Data Set	R-sq.	MAE	RMSE	Normalized RMSE
1	0.9968	4.9117	2.2162	0.0164
2	0.9837	1.7037	1.9860	0.0375
3	0.9673	1.6315	1.9823	0.0536
4	0.9836	1.6792	1.9546	0.0376
5	0.9999	1.7689	2.1319	0.0026
6	0.9898	1.7808	2.1261	0.0295
14C	0.9382	2.2222	2.5819	0.0738
17	0.9376	2.3684	2.7386	0.0740
27	0.9755	1.5365	1.8511	0.0463
40	0.9959	1.5940	1.8628	0.0186
SS1A	0.9964	1.6607	1.9387	0.0175
SS1B	0.9997	1.5600	1.8421	0.0049
SS1C	0.9990	2.1660	2.5142	0.0091
SS2	0.9989	1.5104	1.7603	0.0092
SS3	0.9991	2.0323	2.3798	0.0086
SS4	0.9989	1.5510	1.7999	0.0093

B. TreeNet

A TreeNet model consists of large number of small trees. Each tree usually contains up to six terminal nodes and contributes to the overall model [2]-[5]. Moreover, each tree is responsible to contribute some portion of the overall model and final prediction result of the model is the sum up of all individual tree prediction outcomes. The TreeNet models are capable of handling data quality issues such as tracking missing values, ignorance of suspicious data values, selection of variables (predictors) and detecting interactions among them.

The best part of TreeNet model is, not overly sensitive to the data errors and requires much less time for data preparation and pre-processing of missing values. Furthermore, TreeNet adaptively deals with errors while selecting target variables. The TreeNet structure is implemented using CART 6.0, a Salford Predictive Modeling tool [16], [18].

Tables III and IV illustrate the architecture and performance measurements of TreeNet for various datasets in terms of R-sq., MAE, RMSE, and normalized RMSE.

TABLE III
ARCHITECTURE OF TREE-NET

Parameters	TreeNet structure
Maximum trees in the forest	200
Maximum splitting level	50
Maximum node size to split	2
Maximum categories for continuous predictors	200
Maximum depth of tree in forest	variable (10-20)
Variable weight	Equal
Number of predictor variable	1

TABLE IV
SUMMARY OF PREDICTIONS FOR DIFFERENT DATA SETS USING TREE-NET

Data Set	R-sq.	MAE	RMSE	Normalized RMSE
1	0.9698	5.0861	7.1330	0.0528
2	0.8708	4.9546	6.2564	0.1180
3	-3.0737	9.6661	11.1057	0.3002
4	0.8632	5.1075	6.4152	0.1234
5	0.9776	29.0297	35.8190	0.0432
6	0.9264	4.6885	6.6804	0.0928
14C	-1.6153	9.1141	10.4418	0.2983
17	-3.0737	9.6382	11.0988	0.3000
27	-3.1180	10.3195	11.8811	0.2970
40	0.9611	4.1551	6.3891	0.0639
SS1A	0.9649	4.3469	6.5697	0.0592
SS1B	0.9774	13.2538	16.3981	0.0438
SS1C	0.9773	9.8252	12.1157	0.0439
SS2	0.9766	6.7597	8.5669	0.0449
SS3	0.9772	9.7786	12.0563	0.0435
SS4	0.9763	6.8680	8.7606	0.0452

C. Random Forest

The RF consists of tree predictors that are dependent on the values of random vectors sampled independently with same distribution for all trees in the forest [2]-[8]. The generalization error of the forest tree classifiers depends on the strength of individual tree in the forest and the correlation between them. The random forests are designed to operate quickly over large datasets by using random samples to build each tree in the forest. RF is composed of unpruned regression trees created by using bootstrap samples of the training data and random feature selection in tree induction. The prediction accuracy of RF is achieved by averaging the predictions of the ensembles.

Thus, random forest tree is the collection of decision trees whose predictions are combined to make the overall prediction for the forest. Therefore, decision trees can be applied for predicting software reliability to decide the predictor like mean time to failure (MTTF) in terms of the target value from the set of input values. The internal nodes of a decision tree denote different attributes and the branches between nodes represent the possible values these attributes can have in the observed samples [2], [14], [16], [18], [26].

In RF, a large number of trees are grown in parallel independently and they do not interact until all the trees are built up completely. The RF models produce high accuracy over single decision tree model. We implement the random forest method using DTReg [16]. Tables V and VI show the architecture and performance measurements of RF for various datasets in terms of R-sq, MAE, RMSE, and normalized RMSE.

TABLE V
ARCHITECTURE OF RANDOM FOREST

Parameters	Random Forest structure
Maximum Number of trees	200
Maximum splitting level	100
Maximum node size to split	2
Maximum categories for continuous predictors	200
Maximum depth of tree in forest	variable (10-20)
Variable weight	Equal
Number of predictor variable	1

TABLE VI
SUMMARY OF PREDICTIONS FOR DIFFERENT DATA SETS USING RANDOM FOREST

Data Set	R-sq.	MAE	RMSE	Normalized RMSE
1	0.9910	625.4292	5.9677	0.0442
2	0.9826	141.0810	3.1494	0.0594
3	0.9593	106.7763	3.2872	0.0888
4	0.9711	157.0853	3.9327	0.0756
5	0.9978	1158.960	18.837	0.0227
6	0.9804	264.9911	4.8375	0.0672
14C	0.9417	114.9851	3.9966	0.1142
17	0.9576	111.9452	3.7959	0.1026
27	0.9586	125.7674	3.8419	0.0960
40	0.9865	414.5744	5.6581	0.0566
SS1A	0.9911	493.2970	5.4423	0.0490
SS1B	0.9964	364.7817	11.118	0.0297
SS1C	0.9947	294.4492	10.1142	0.0366
SS2	0.9931	151.6904	7.5547	0.0396
SS3	0.9947	2048.107	9.8371	0.0355
SS4	0.9953	1009.332	6.7349	0.0347

D. Cascade Correlation Neural Network

Artificial neural networks consist of several processing nodes analogous to neurons in the brain. Each node has a function associated with it and a set of local parameters, determining output of the node for a given set of input. Cascade correlation neural networks (CCNN) are based on supervising learning algorithms [17], [27].

The CCNN is a robust network model capable of producing accurate results with small variation in the adjustment of parameters. Although, Back-propagation neural network is widely used multi-layer feed forward network. However, the disadvantage of multi-layer feed forward networks using back propagation is that number of hidden layers and neurons in the

network are problem specific. This varies from task to task and hence cannot be utilized for generalization. Thus, if large number of hidden neurons is used then the network will learn irrelevant details during the training and once trained it does not generalize well. Alternatively, if the size of the network is very small then it will not be able to learn from the training set accurately. Therefore, we require such a network which could determine the size for a network dynamically starting with a minimal network and then adding hidden neurons and connections as required. Thus, CCNN is an alternative viable solution, which helps in overcoming the shortcomings of BPNN by adjusting the number of hidden layers dynamically during the learning phase. The CCNN model is implemented using DTReg [16]-[18]. Tables VII and VIII show the architecture and performance measure of CCNN for various datasets in terms of R-sq., MAE, RMSE, and normalized RMSE.

TABLE VII
ARCHITECTURE OF CCNN

Parameters	CCNN Structure
Maximum neurons in hidden layer	50
Hidden neuron kernel function	Sigmoid and Gaussian
Output neuron kernel function	Sigmoid
Validation method	Cross validation
Number of cross-validation folds	10
Number of predictor variables	1
Number of neurons in input layer	1
Number of neurons in output layer	1

TABLE VIII
SUMMARY OF PREDICTIONS FOR DIFFERENT DATA SETS USING CCNN

Data Set	R-sq.	MAE	RMSE	Normalized RMSE
1	0.9978	1.9188	2.5728	0.0191
2	0.9979	0.8082	0.9862	0.0186
3	0.9895	1.0356	1.5825	0.0428
4	0.9972	0.9172	1.1327	0.0218
5	0.9996	5.0218	6.3647	0.0077
6	0.9977	1.1608	1.4255	0.0198
14C	0.9971	0.4664	0.7782	0.0222
17	0.9974	0.6367	0.7900	0.0214
27	0.9968	0.7656	0.9344	0.0234
40	0.9981	1.4842	1.7825	0.0178
SS1A	0.9986	1.3032	1.6736	0.0151
SS1B	0.9794	14.3782	22.8095	0.0610
SS1C	0.9993	2.3560	2.9762	0.0108
SS2	0.9983	2.5025	3.1829	0.0167
SS3	0.9956	5.5567	7.4658	0.0270
SS4	0.9988	2.1166	2.7746	0.0143

E. Training and Validation Method

The data mining techniques applied for predicting software reliability predictions have been validated using sixteen failure datasets taken from the DACS. The cumulative number of failures x_i detected during the testing at time t_i, is taken as the target variable where t_i is the predictor. Each dataset is divided into two parts: training and testing data. The training data is then applied to the prediction model for predicting software reliability.

Separate training and validation dataset are desired for testing the accuracy of predicting models. However, when the database used for modeling is small such as the datasets 2, 3, 4, 14C, 17 and 27 used in our study, we are not able to spare a large portion of data for the testing, thus testing is performed on relatively small samples. Therefore, to maximize the utilization of each dataset we apply k-cross validation, an alternative procedures that allows more of the data to be used for fitting and testing. Using k-cross validation, the entire dataset is randomly divided into k subsets (here k=10) and for each iteration one of the k subsets is used as the training data and the remaining subsets are used to validate the model for predicting software reliability [8], [12], [14], [25], [26].

V. ANALYSIS RESULTS

In this section, we present summary of the results for all sixteen data sets using CART, TreeNet, RF and CCNN in terms of R-sq, MAE, and RMSE.

A. Observations

Some specific observations of our study for predicting software reliability using the data mining techniques are discussed as follows:

1. Based on the results achieved through experiments we conducted, it is observed that designing the models for reliability growth of varying complexity for a given data set using CCNN is more easy. However, the effectiveness of the CCNN model depends on behavior of the dataset, which is basically of fluctuating nature. The CCNN suffers from overfitting the results while dealing with real-life previously unseen large failure datasets. Thus, Cascade correlation neural network based models face the problem of learning from the dynamic environment, which requires a larger sample size for training and testing the model together with large no. of independent variables.
2. It can be easily shown from Tables I-VIII that the CART and RF fit very well in terms of MAE and RMSE. Thus, CART and RF can approximate the continuous function accurately, which implies that such methods may be employed effectively for estimation and prediction of cumulative failures observed by time t in software reliability modeling. On other hand, the TreeNet method does not make accurate predictions in terms of R-sq., MAE, RMSE and therefore such methods are not found suitable for current datasets applied in our study. However, it would be interesting to see the impact of other similar studies on updated large failure dataset of real-life projects before making such generalization.
3. The robustness and validity of the CART, RF, and CCNN models make it easier for real-world applications to model complex failure phenomena of predicting software reliability accurately. Moreover, the CART and CCNN models are more adaptive to the modeling of nonlinear functional relationships, which are difficult to model with other classical techniques (LR) of predicting software reliability in practice.

4. The performance measurement of the CART and CCNN generalize well even in high dimensional spaces under small training datasets. Therefore, software reliability prediction models can be built much earlier with these methods than other conventional techniques with relatively good performance achieved. However, the TreeNet model makes relatively accurate predictions for the training data resulting in better accuracy on the training data but poor performance on previously unseen failure datasets. Here, we have applied DTReg and Salford predictive modeling tools that include an over fitting control facility to prevent over fitting.
5. From the observed results we conclude that CART provides more reliable performance and accurate results than other data mining algorithms presented in our study. We observed that the CCNN is another viable alternative model which performs better in comparison to other data mining techniques. Since, the supervised learning algorithms are used to create and install hidden neurons for maximizing the magnitude of correlation between existing and new neurons. This way, CCNN is capable of learning very quickly, determines the size and topology dynamically to minimize the residual error signals leading to the more accurate prediction result of the model.

B. Discussions

The performance measurement of the data mining techniques (CART, TreeNet, RF, and CCNN) in terms of R-sq, MAE, RMSE and NRMSE values corresponding to all sixteen failure data-sets are summarized as follows. The correlation coefficient values vary from 86% to 99% for the majority of the datasets except dataset 3, dataset 14C, 17, and dataset 27 using the TreeNet model. This way, the relationship between predicted values and actual values are correlated strongly, suggesting high degree of relationship between predicted and actual values. Experimentally, we found that a statistic is biased if, it consistently over or underestimates the parameter it is estimating. A statistic is positively biased if it tends to overestimate the parameter and a statistic is negatively biased if it tends to underestimate the parameter.

The prediction results of CART is fairly good in terms of normalized RMSE for all datasets lying between 0.0026 (minimum for dataset 5 having maximum failure records) and 0.0740 (maximum for dataset 17) suggesting that the model predicts well for large size failure datasets shown in Fig. 1. However, the performance measurement of the TreeNet and RF in terms of NRMSE is relatively good but very inconsistently and over fits in terms of MAE and RMSE shown in Figs. 2 and 3.

Another positive outcome of our study is the performance measurement of the CCNN in terms of NRMSE which lies between 0.0191 and 0.0610 suggesting that the model predicts well but very inconsistently in terms of MAE and RMSE. That is, CCNN shows very encouraging results and makes it an alternative choice, which outperformed the model, predicted using the TreeNet and random forest methods shown in Fig. 4.

The CART model applied to predicting software reliability was found to be quite close to the target values. This illustrates that the model will not collapse when applied to unknown failure datasets of a realistic environment. Finally, based on overall performance we observe that the CART method is more appropriate and capable of yielding more accurate results and therefore it can be utilized as a tool for predicting software reliability in real-life applications. Moreover, CCNN was found to be alternative choice, which can be used in modeling complex non-linear relationships more effectively such as in predicting software reliability. Fig. 5 is the graphical representation of overall performance analysis of the data mining techniques presented in our study for predicting software reliability.

C. Threats to Validity

The present study for software reliability prediction using the data mining techniques has certain limitations, which need to be addressed thoroughly before deployment in practice. Some technical/legal/social and engineering constraints of our study are as follows. First, the measures could not be evaluated over updated and current failure datasets due to the lack of empirical software failure data of modern computing systems. Therefore, the prediction and assessment capability of our approach across different organizations remains an open issue for the acceptance of such models. Second, the performance measures of various data mining algorithms applied to software reliability prediction depends on the representation of software failure data. Further, the failure datasets available in literature is not updated frequently probably due to the competitive nature of business, fear of losing customers and other legal issues of software industry. Therefore, more similar studies need to be carried out with different data sets to give generalized results.

VI. Conclusion

In this paper, we have examined the performance of three data mining techniques namely CART, TreeNet and Random Forest for predicting software reliability based on past failure data of software systems. Their effectiveness is demonstrated through sixteen failure datasets taken from the Data and Analysis Center for Software. The performance measurement is compared in terms of MAE and NRMSE obtained in the test set. From experiments conducted, we conclude that the CART model outperformed the model predicted using the RF, TreeNet and CCNN models in all datasets. The results obtained through CCNN are also very encouraging, which outperformed the model predicted using the TreeNet and RF methods and therefore it may be utilized as an alternative choice for making predictions.

Further, we plan to replicate our study of software reliability prediction models by introducing some more intelligent machine learning algorithms applied to a large category of failure datasets of real life industrial software projects in a realistic operating context.

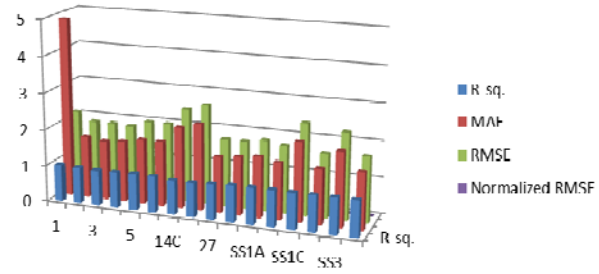

Fig. 1 The performance measurement of the CART for predicting Software reliability

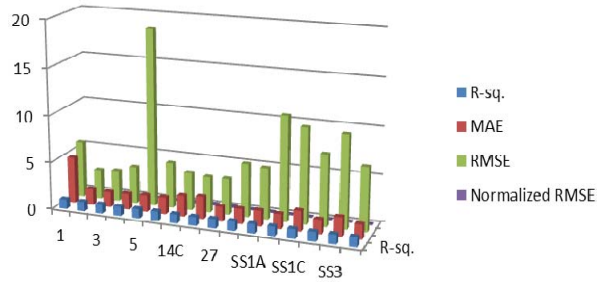

Fig. 2 The performance measurement of the TreeNet for predicting Software reliability

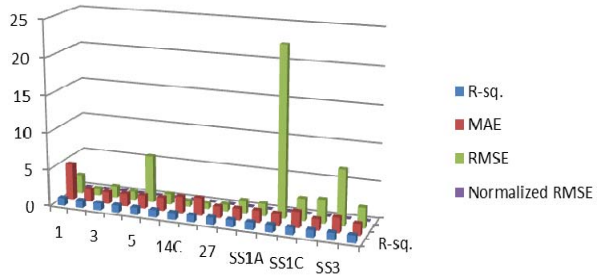

Fig. 3 The performance measurement of the RF for predicting Software reliability

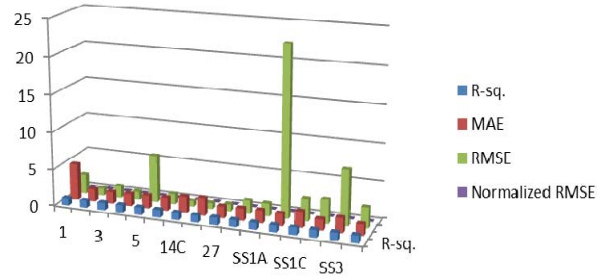

Fig. 4 The performance measurement of the CCNN for predicting Software reliability

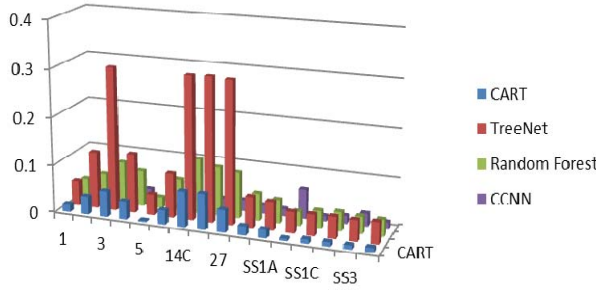

Fig. 5 Comparative Analysis of various data mining in terms of NRMSE

ACKNOWLEDGMENT

The authors wish to thank to the publisher of several research papers and the failure datasets used of various projects is extracted from the Software Life Cycle Empirical/Experience Database (SLED). These datasets were compiled by John D. Musa at Bell Telephone Laboratories and published by the Data and Analysis Center for Software, which can be found at http://www.dacs.org.

REFERENCES

[1] K.K. Aggarwal, Y. Singh, A. Kaur and R. Malhotra, "Empirical analysis for investigating the effect of object-oriented metrics on fault proneness: a replicated case study," *Software Process Improvement Practice* Vol. 14, No. 1, pp. 39–62, 2008.

[2] L. Breiman, "Random Forests, "*Machine Learning*," Vol. 35, no. 1, pp. 5-32. DOI: 10.1023/A:1010933404324, 2001.

[3] K. Funatsu, "*Knowledge-Oriented Applications in Data Mining*," In Tech., under CC BY-NC-SA, 2011.

[4] J. Han, M. Kamber, "*Data Mining: Concepts and Techniques*," Morgan Kaufmann Publishers, India, 2006.

[5] T. Hastie, R. Tibshirani and J. Friedman, "*The Elements of Statistical Learning: Data Mining, Inference, and Prediction*," New York: Springer, 2001.

[6] S. Ho, M. Xie, and T. Goh, "A study of the connectionist models for software reliability prediction," *Computers and Mathematics with Applications,* Vol. 46, pp. 1037-1045, 2003.

[7] N. Karunanithi, D. Whitley and Y. Malaiya, "Prediction of software reliability using connectionist models," *IEEE Transactions on Software Engineering*, Vol. 18, no. 7, pp. 563-574, 1992.

[8] R. Kohavi, "The power of decision tables," *The Eighth European Conference on Machine Learning (ECML-95)*, Heraclion, Greece 1995, pp. 174-189.

[9] C. Kuei, H. Yeu and L. Tzai, "A study of software reliability growth from the perspective of learning effects," *Reliability Engineering and System Safety*, Vol. 93, no. 10, pp. 1410-1421, 2008.

[10] M.R. Lyu, "*Handbook of Software Reliability Engineering,"* McGraw Hill, India, pp.131-151, 1999.

[11] R. Malhotra, Y. Singh and A. Kaur, "Comparative analysis of regression and machine learning methods for predicting fault proneness models," *International Journal of Computer Applications in Technology,* Vol. 35, no. 2, pp. 183-193, 2009.

[12] J. Mueller, F. Lemke, "Self-Organizing Data Mining: An Intelligent Approach to Extract Knowledge from Data," Dresden, Berlin, 1999.

[13] D. Musa, "*Software Reliability Engineering: More Reliable Software Faster and Cheaper",* Second Edition, McGraw-Hill: India, 2009.

[14] K. Raj, V. Ravi, "Software reliability prediction by using soft computing techniques," *The Journal of Systems and Software,* pp. 576-583. DOI: 10.1016/jss.2007.05.005, 2008.

[15] Q. Ross, "*C4.5: Programs for Machine Learning,"* Morgan Kaufman Publishers: San Mateo, CA., 1993

[16] Salford predictive modelling system, http//www.salford-systems.com. (Accessed 1 July 2011).

[17] E. Scott, L. Christian, "*The Cascade-Correlation Learning Architecture*," CMU-CS-90-100, School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213, 1991.
[18] P.H. Sherrod, DTReg predictive modeling software, 2003 available at http://www.dtreg.com, (Accessed 8 January 2011).
[19] Y. Singh, P. Kumar, "A software reliability growth model for three-tier client-server system," *International Journal of Computer Applications,* Vol. 1, no. 13, pp. 9-16. DOI: 10.5120/289-451, 2010.
[20] Y. Singh, P. Kumar, "Determination of software release instant of three-tier client server software system," *International Journal of Software Engineering,* Vol. 1, no. 3, pp. 51-62, 2010.
[21] Y. Singh, P. Kumar, "Application of feed-forward networks for software reliability prediction," ACM *SIGSOFT Software Engineering Notes,* Vol. 35, no. 5, September 2010, pp. 1-6. DOI: 10.1145/1838687.1838709, 2010.
[22] Y. Singh, P. Kumar, "Prediction of Software Reliability using Feed Forward Neural Networks," Proceedings of Computational Intelligence and Software Engineering (CiSE), 2010 International Conference, Wuhan, China, DOI: 10.1109/CISE.2010.5677251, 2010.
[23] Y. Singh, A. Kaur and R. Malhotra, "Application of support vector machine to predict fault prone classes," *ACM SIGSOFT Software Engineering Notes*, Vol. 34, No. 1, DOI= http://doi.acm.org/10.1145/1457516.1457529, 2009.
[24] R. Sitte, "Comparison of software reliability growth predictions: Neural Networks vs. Parametric Recalibration," *IEEE Transactions on Reliability,* Vol. 48, no. 3, pp. 285-291, 1999.
[25] Software Life Cycle Empirical/Experience Database (SLED) compiled by Musa and published by Data & Analysis Center for Software (DACS). http://www.dacs.org (Accessed 14 February 2009).
[26] I. Witten, E. Frank, "*Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*," Third Edition, Morgan Kaufman, Addison-Wesley, San Francisco CA, 2011.
[27] J. Zheng, "Predicting software reliability with neural network ensembles," *Expert Systems with Applications*, Vol. 36, no. 2, pp. 216-222. DOI: 10.1016/j.eswa.2007.12.029, 2009.

Dr Pradeep Kumar is an Associate Professor in the Department of Computer Science & Information technology at Maulana Azad National Urdu University, Hyderabad (Telangana State). He received his Master's degree in Computer Technology and Applications from Delhi Technological University, formerly Delhi College of Engineering, Delhi University. He completed his Ph.D. from the University School of Information & Communication Technology (USICT), Guru Gobind Singh Indraprastha University (GGSIPU), Delhi. His research interests include software reliability engineering, models for software metrics, machine learning, neural network modeling and soft computing. He has more than 25 publications in journals of international repute including national journals, conferences and proceedings of the international conferences. He is a Member of Association for Computing Machines (ACM), India, Member of Computer Science Teachers Association (CSTA), USA, Senior Member of International Association of Engineers (IAENG), Member of International Association of Computer Science and Information Technology (IACSIT), Singapore and Senior member of Universal Association of Computer and Electronics Engineers (UACEE). He is a member of editorial board for various national and international journals in the field of software engineering and program committee member/reviewer for several international conferences.

Prof. Abdul Wahid is Dean of School of Computer Science & IT at Maulana Azad National Urdu University, Hyderabad (Telangana State), Pin- 500032, India. He is a Member of Association for Computing Machines (ACM), India, Member of Computer Science Teachers Association (CSTA), USA, Senior Member of International Association of Engineers (IAENG), Member of International Association of Computer Science and Information Technology (IACSIT), Singapore and Senior member of Universal Association of Computer and Electronics Engineers. He is a member of editorial board for various national and international journals in the field of Web software engineering.