IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-5, Issue-5 E-ISSN: 2347-2693

A Neural Network Approach for Anticipating Maintenance Effort using Back Propagation Algorithm

N. Chaudhary[1*], A. Kumar[2*]

[1]Department of Computer Science engineering, Baddi University, Baddi, India
[2]Department of Computer Science engineering, Baddi University, Baddi, India

Corresponding Author: neenachoudhary13@gmail.com Tel.: +91 9736718517

Available online at: www.ijcseonline.org

Received: 12/Apr/2017, Revised: 26/Apr/2017, Accepted: 20/ay/2017, Published: 30/May/2017

***Abstract*—** Software maintenance is an important phase of life cycle process. It is a transforming of a software process after receiver receives and if fault occurs then to modify software products and remove extra bugs. The phase is much important phase which starts with customer end. Therefore predicting the efforts like-cost, size has become one of an important issues which is to be analyzed for effective resource allocation. In view of these issues ,we have developed text mining techniques using machine learning method name BPA(Back Propagation Algorithm).The intended model ratified using 'browser 'application pack of android operated system. ROC (Receiver Operating Characteristics) curve is a graphical representation that describes the working of a binary classified system. The performance of model rely on the words count taken for classification which shows best result as the word number increases which describes its accuracy. More the words count more the accuracy.

***Keywords*—**Software maintenance,Machine learning,BPA,Software Prediction,Neural network

I. INTRODUCTION

Maintenance is an important phase of life cycle of software process. It is a modification of software process after receiver receives and if fault occurs modify software products and remove extra bugs. The phase is very decisive phase start with customer end. Some bugs are removing concurrently software process and some are removing after delivery of products like customer ends. In this software model define defects, bugs collecting reports of defects which is an computerized tool that are generated various bugs, faults and provide useful information according to bug which present in a version of GIT control system based upon platform and fix version. Scope of support maintenance of software process and operation is to maintain and develop by the using application in a production system environment and also to defend its end users who ceaseless use it. It is basically about Fault fixing, bug fixing, development and adapt functionalities, data, performance perfection, etc. Research carried by Lehman is needed for these changes to occur, which are known as so called Lehman's laws. A program which is useful in real world environment, Application mandatorily must transform or become less or no use in that environment system. Remarkable changes are derived from the way to procure software to interact with external circumstances, including people, organizations, and artificial systems. In fact, software is unlimitedly tensile and, therefore, it is often recognized as the easiest and crucial part to change in a system. [1,2,3,4,5]. Organization IEEE, Institute of Electrical and Electronics Engineers, defines support and maintenance as following: "Software maintenance is the process of modifying a software system life cycle process or component after delivery to correct faults, improve performances or other data set attributes, or adapt to a changed software environment [5,6,7,8,9,10,11,12,13]

Several authors disagree by this view and assure that software maintenance should start quite before a system becomes operational. Pigoski discovers the needs to start maintenance when development starts, in his new definition: "Software maintenance is the totality of activities required to provide cost-effective support in a software system. Activities are performed before-delivery stage as well as the after-delivery stage in an effective way. Before-delivery activities includes planning for after delivery operations, supportability, and logistics determination.aftert-delivery activities include software modification, training, and operating activities [1,2,6,13,14,15,16,17].

"This definition is consistent with the approach to software maintenance taken by ISO in its standard on software life cycle processes It definitively dispels the image that software maintenance is all about fixing bugs or mistakes.", (Canfora, 2000).Author of this work fully agrees with the view that operation, maintenance and software support starts deeply in

© 2017, IJCSE All Rights Reserved

the delivery solution phase. One of the finest practices in IT industry is participation of future support and operation staff in the project delivery, as it supports knowledge transfer and also smoother transfer of solution from project delivery in the production environment. Software delivery methods specify various software supporting and maintaining outputs and tasks, which must be created in making of the project.

- **Effort prediction and software in support**

In respect to the literature as studied (for elaborated literature and review visit chapter References in this article and author's thesis (Marounek, 2012), work considers to mention following five core key works about effort prediction in software in support and its maintenance area. Moreover, few of them deeply discuss why historical complex models and their approaches are poorly used (more precisely they are not used). Ease of Use. Basically seven main crucial phases in maintenance process are-

- ***Change Management***

In this phase the user modifies, a customer, a programmer and a manager is allocated a maintenance category task, like that procedures operations related to requirement precedence and an exclusive identifier. The change management phase includes activity which establishes whether to accept or reject the request and to assign it to a set of modifications scheduled for implementation maintaining the set of Integrity and Specifications functions formal and unformal methods [1,6,18,19,20,21,22].

- ***Analysis***

In this phase analysis phase without analyse you don't plan for design, execution test, and delivery. The main motive of analysis is to cease the possibility of the requested change for arrangement and implementation by the change. Analysis is two levels: feasibility analysis and detailed analysis. Feasibility analysis identifies solutions and assesses their aspect and costs, whereas detailed analysis defines the necessities for the modification, test policy, and develop an implementation plan [18,19,20,21,22].

- ***Design***

The system is actually designed in this phase. This brings about all present system and documentation of projects, database and existing software and output of the analysis phase [8]. It aims to expand a revised logical, physical design for the change and to design the changes for all of the Categories of maintenance [18,19,20,21,22].

- ***Implementation***

This phase includes the activities of coding and unit testing, understand of the mutate code, integration and analysis, regression testing, and risk. The phase also includes a test-readiness review to asses' sensibility for the system and regression testing [18,19,20,21,22].

- ***Regression/System Testing***

In this phase the complete system is tested to make certain conformity to the new necessities with addition to the alterations. Additionally in this phase functional and interface test is tested through regression testing, which Authenticate that no new faults is been added to it. Finally, this phase is liable for verifying familiarity for Acceptance testing [18,19,20,21,22].

- ***Acceptance Testing***

This level of testing is sensible with the fully assimilate system and involves users, customers, or a third party installation by the customer. Acceptance testing composes regression tests, association tests and Functional tests [18,19,20,21,22].

- ***Delivery***

This is the phase in which the accommodated systems is unlimited for both operation and installation [18,19,20,21,22].

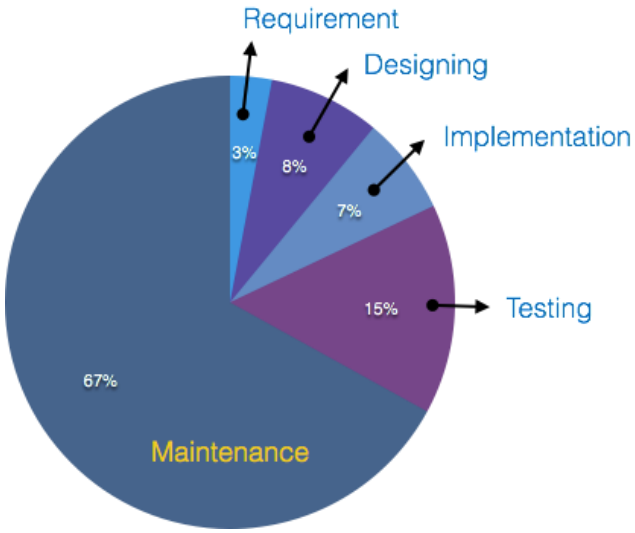

Figure. 1 Maintenance software designing process in pie chart

II. RESEARCH BACKGROUND

In this section, we will be focus on the research background which is required for the accomplishment of the purposed methodology. Firstly we will explain the brief overview of machine learning method used for classification. Finally we highlight on the performance evaluation measures used to validate the result.

A. BPA (Back Propagation Algorithm)

The most popular neural network algorithms are back propagation algorithm. In this Back propagation Algorithm could be divided in to four main types. After selecting the weights of the network randomly, the back propagation

© 2017, IJCSE All Rights Reserved

algorithm is used to compute the necessary corrections. The algorithm is been elaborated into the four steps mentioned below:

i) Computation of Feed-forward
ii) Up to Output layer from back propagation
iii) Up to hidden layer from back propagation
iv) Updates of weight

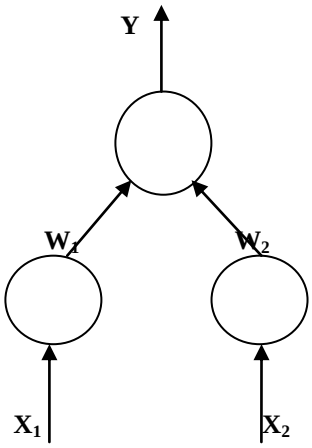

Figure .2 Back propagation algorithm

III. MODEL EVOLUTION

The performance measured used in the following parameters is

A. Sensitivity

It is defined as the ratio of correctly define as defect reports like defect fixing error and others documents prone.it is also called recall.

Formula

Sensitivity=TP/TP+FN*100

B. Receiver Operating Characteristics (ROC)

ROC is an analysis techniques which is used to evaluate the performance of predicted model.it is a plot of value is depend upon the x-axis and y-axis at the different cut-off point.

Table. 1: The receiver operating characteristics

TP	FP
FN	TN
1	1

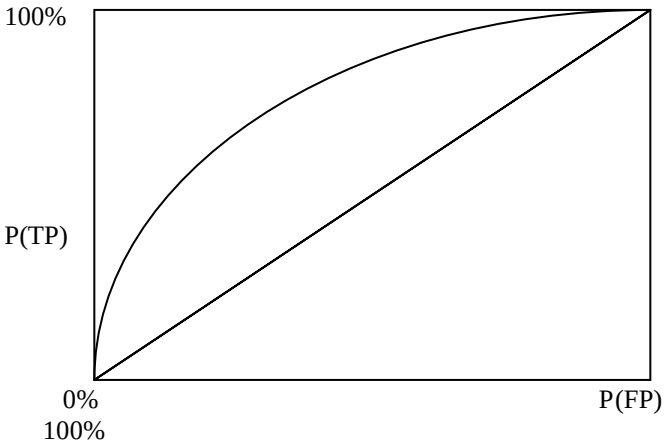

Figure. 3 Plot between true positive &false positive

IV. PRECISION RECALL

Precision recall is a most popular technique used to a number of all events you recall is known as precision recall.

PRECISION=TP/(TP+FP)
RECALL=TP/(TP+FN)

V. VALIDATION METHOD USED

Validation method is used two steps like that Training data set and testing data set.

Training data set is divided into 70% train data and the rest 30% testing data. This validation method is repeated for 10 different partitioning variables in order to get more generalized and accurate result.

VI. PROPOSED METHODOLOGY

In this section we will be discussing the methodology that has been used in our work for effort prediction with some following step. In fig 1 that can be show two basic modules to achieve the prediction maintainability. Defect collection and Reporting system (DCRS) module and text mining module. Each module will be described between sequences like that-

© 2017, IJCSE All Rights Reserved

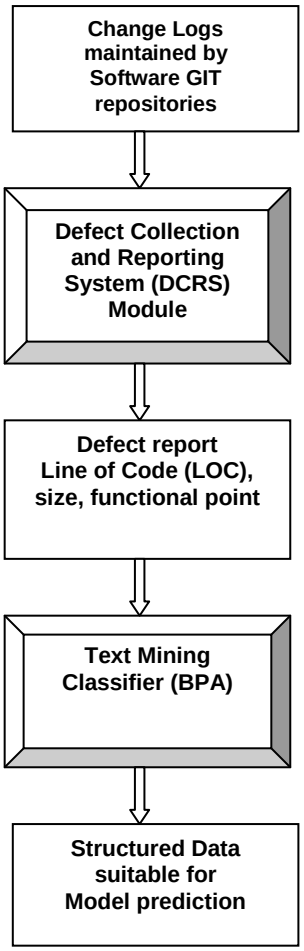

Figure. 4 Framework used for software maintenance framework

A. DEFECT COLLECTION AND REPORTING SYSTEM MODULES

Defect Collection and Reporting System module that generate defect ID, defect description and total number of changes like line of code (LOC), Size and functional point which may be added and deleting for one version to next version. In our work total number of changes effort prediction. Such like that of a system is expected to perform the following operations: First, occupy the 'it change logs' for a given software and process them to obtain the defects which were reported in a specific version of that software and were fixed in the subsequent version. Then, the system should extract vital defect information such as unique bug or defect identifier and the defect description, if provided. Next, we require the association of defects to their corresponding source files (matlab code files, or class files contained in the source code). Then the system should perform the computation of total number of defects fixed corresponding to each class, i.e., the number of defects associated with that class. Finally, the corresponding values of different metrics should be calculated by the system for each source code file for previous version of the open source software. In this work, we have developed tool named Defect Collection and Reporting System (here onwards, referred to as DCRS), which incorporates each and every functionality stated above and consequently generates various reports that present the collected data in a more processed, meaningful and use.

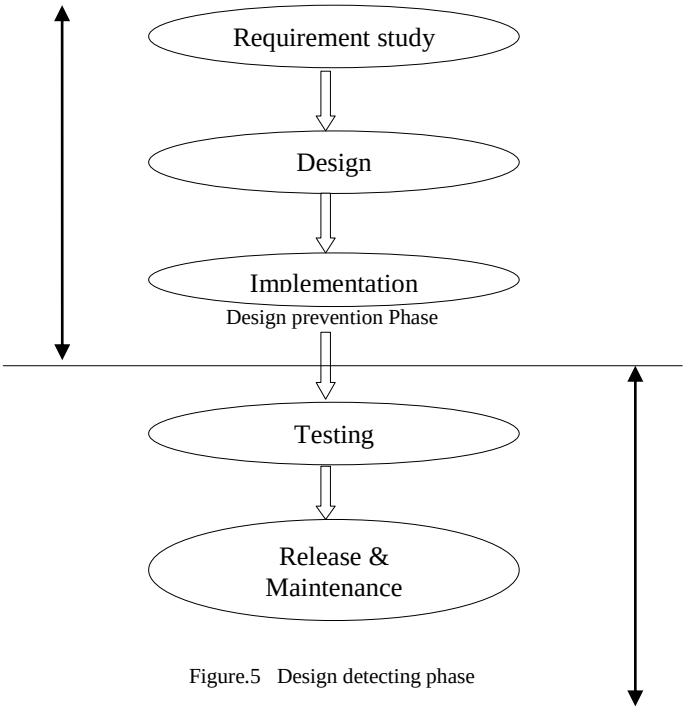

Figure.5 Design detecting phase

Table .2

No. of Defects	0	1-5	6-10	11-15	>15
No. of Classes	224	96	4	4	3

B. Text Mining Module

Text mining classifier is transform numbers of data unstructured form into structure form. In this classifier used to develop prediction model. If this kind of classification is performed manually it have certain disadvantages –

i. It needs domain experts in the field of predefined categories.

ii. It is time-consuming, lengthy, leads to frustration

iii. It leads to certain errors and can be employee biased (subject biased).

iv. Human decision among two experts may vary.

© 2017, IJCSE All Rights Reserved

v. For new documents we need to repeat the process (possibly of another domain).

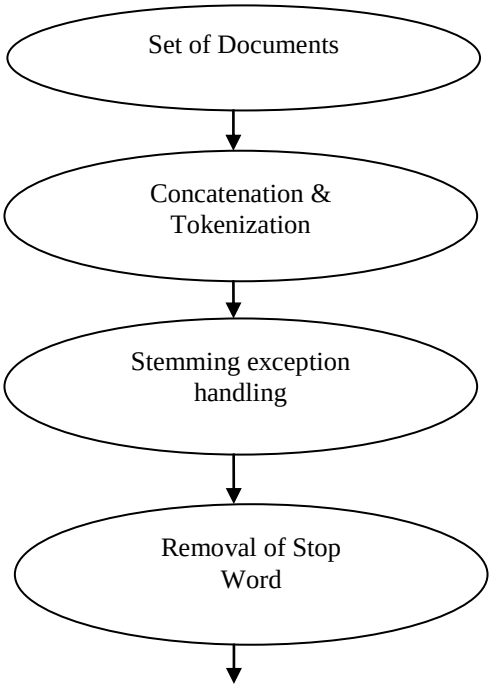

Figure. 6 Text mining modules

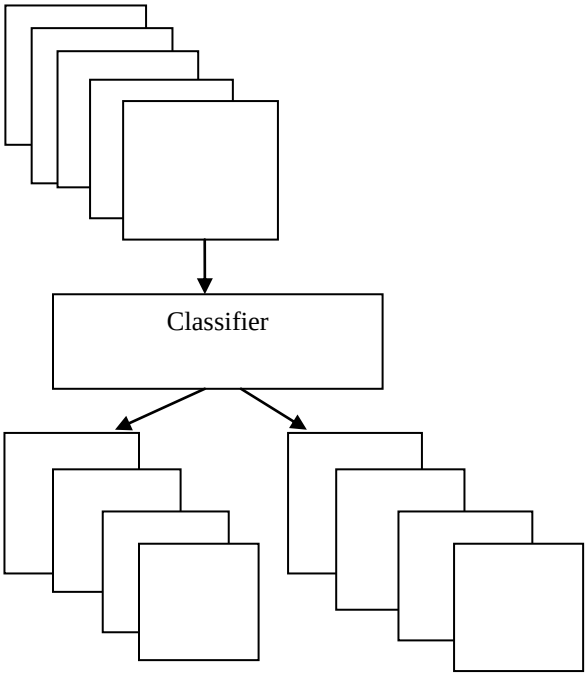

Figure. 7 Multiclass single label classifications

VII. RESULT AND DISCUSSION

Figure. 8 denotes the result obtained by running the particular application which describes Epoch, Time, Performance, Gradient, Mu and validation Checks.

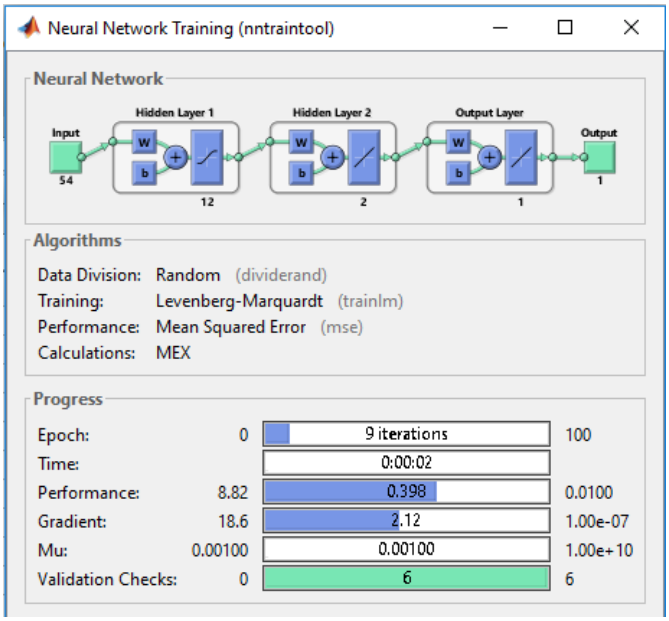

Figure .8 Trains to neural network

Figure.9 Denoting the graph plotted between Mean Squared Error (mse) versus 9 Epochs therefore denotes the best validation Performance as 0.84321 at epoch 3.

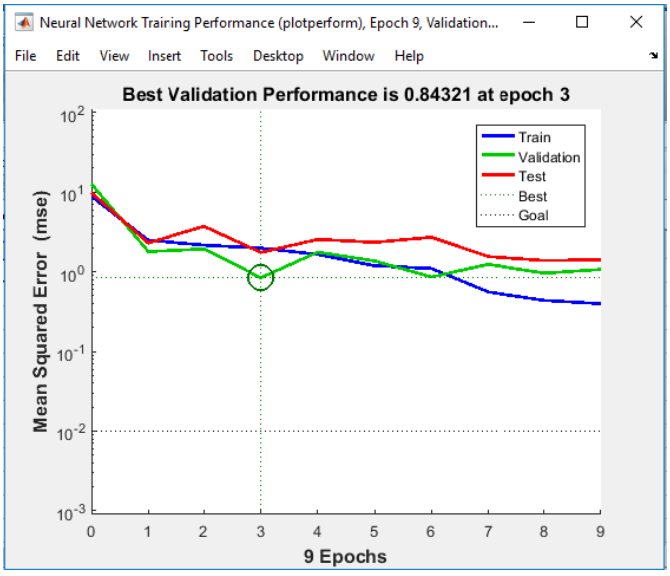

Figure .9

Figure.10 Denotes the graph of Gradient, Mu and Validation Checks versus 9 epochs. Here result came out as Gradient is

© 2017, IJCSE All Rights Reserved

2.12 at epoch 9, Mu is 0.001 at epoch 9, and Validation Checks is 6 at epoch 9.

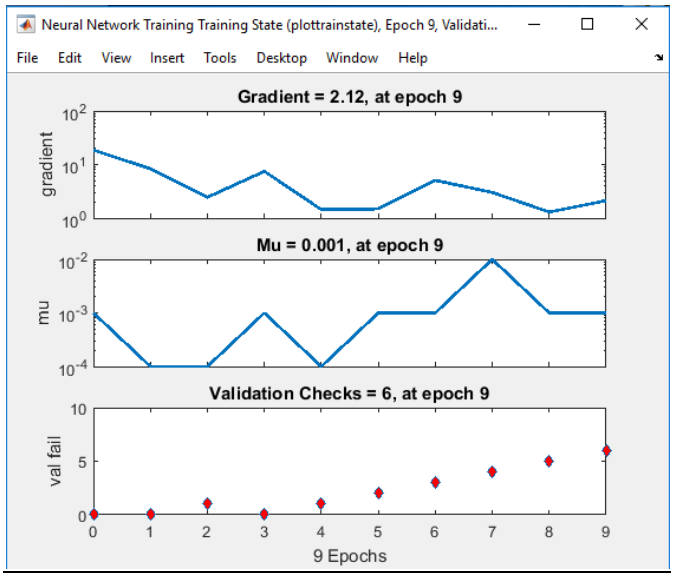

Figure .10

Figure.11 Training comes out as R=0.64306 output-=0.52 at target +2.1, Validation as R=0.78515 output-=0.59 at Target +1.9, Test as R=0.61779 output-=0.42 at Target+2.5 and All R=0.66235 output-=0.52 at Target +2.1.

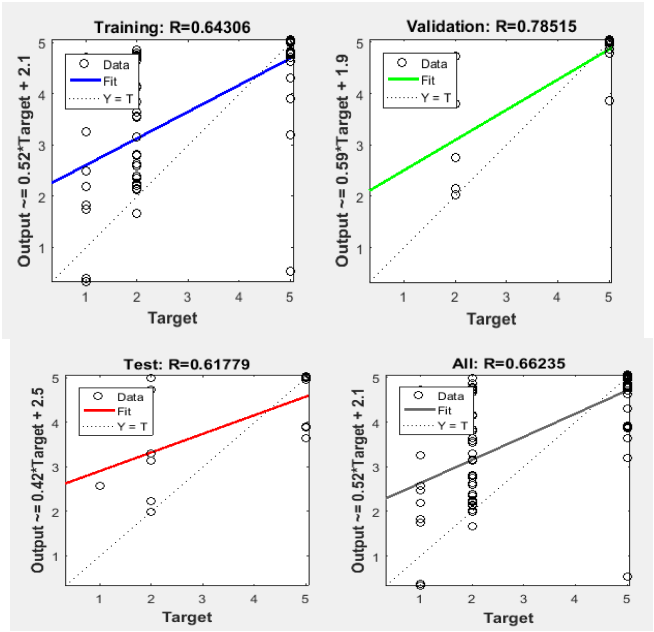

Figure .11

Similarly taking another case as described in the figure below high accuracy is been obsevered.Therefore, from the result analysis it is seen that the MLP (Multi Layer Perception) model should be preffered for effort prediction with higher number of words. These observations make it evident that MLP is the best suited model for prediction of software maintainability.

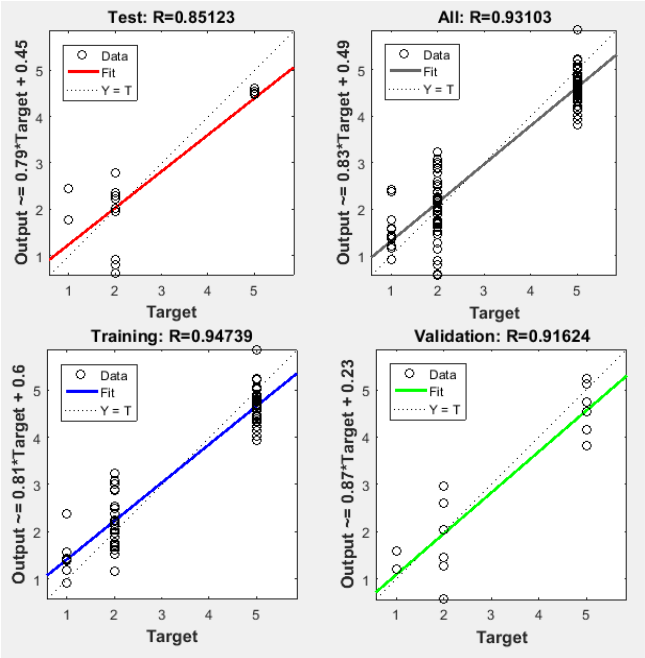

Figure .12

VIII. CONCLUSION

Prediction of software maintenance effort has become very crucial because lots of defects are generated after its operation to delivery at customer ends. Software maintenance efforts prediction is important parts to remove bugs and enhancement of software model. We had incorporated the required text mining using base propagation algorithm (BPA) of neural network to predict the efforts and result validated using android operating system. The result analysis performance of the model depends upon increased number of words and therefore the best performance was denoted by MLP model which must be preferred for effort prediction as the word strength increases.

REFERENCES

[1]. JS. Yadav, S. Dhariwa, "*Enhanced the Performance of Digital Image Compression Using Wavelet Transform Function and BP Neural Network Model*" International Journal of Computer Sciences and Engineering, Vol. 5, issue 4, pp.29-33, 2017.

[2]. C Jin , A.L. Jin , "*Applications of Support Vector Machine and Unsupervised Learning for Predicting Maintainability using Object-Oriented Metrics*", Second International Conference on Multi Media and Information Technology ,Vol.1, No.1, pp.24-27, 2010.

© 2017, IJCSE All Rights Reserved

[3]. A Kaur, K Kaur, R Malhotra, "*Soft Computing Approaches for Prediction of Software Maintenance Effort*", International Journal of Computer Applications), Vol. 1, No.16, pp.0975-8887, 2010.
[4]. L Ping, "*A Quantitative Approach to Software Maintainability Prediction*", International Forum on Information Technology and Applications, Vol .1, No.1, pp.105-108, 2010.
[5]. S.J. Farlow, "*The American Statistician*", Vol. 35, No.4, pp.210-215, 1981.
[6]. MO. Elish, KO. Elish, "*Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study*", European Conference on Software Maintenance and Reengineering, Kaiserslautern, pp. 69-78, 2009.
[7]. R. Mohanty, V. Ravi, M.R. Patra, "*Software Reliability Prediction using Group Method of Data Handling*", Proceeding of 12th International conference on RSFDGrC, Rough Sets Fuzzy SetsData Mining and Granular Computing, LNAI, pp.344- 351, 2009.
[8]. M.M. Ibrahiem, E.l. Emary, S. Ramakrishnan, "*On the Application of Various Probabilistic Neural Networks in Solving Different Pattern Classification Problems*", World Applied Sciences Journal, Vol.4, No.6, pp.772-780, 2008.
[9].Y. Zhou, H. Leung, "*Predicting object-oriented software maintainability using multivariate adaptive regression splines*", Journal of Systems and Software, Vol. 80, No.8, pp.1349-1361, 2007.
[10]. C.V. Koten, A.R. Gray, "*An application of Bayesian network for predicting object-oriented software maintainability*", Information and Software Technology Journal, Vol.48, No.1, pp 59-67, 2006.
[11]. S. Misra, "*Modelling design/coding factors that drive maintainability of software systems*", Software Quality Journal, Vol.13, No.3, pp.297-320, 2005
[12]. A.D. Lucia, E. Pompella, S. Stefanucci, "*Assessing effort estimation models for corrective maintenance through empirical studies*", Information and Software Technology, Vol.47, No.1, pp.3-5, 2005.
[13]. K.K. Aggarwal, Y. Singh, P. Chandra, M. Puri, " *Measurement of Software Maintainability Using a Fuzzy Model*", Journal of Computer Sciences, Vol.1, no.4, pp.538-542, 2005 .
[14].M. Thwin, T. Quah, "*Application of neural networks for software quality prediction using object oriented metrics*", Journal of Systems and Software, Vol.76, No.2, pp.147-156, 2005.
[15]. M. Dagpinar, J.H. Jahnke, "*Predicting Maintainability with Object-Oriented Metrics - An Empirical Comparison*", Proceedings of the 10th Working Conference on Reverse Engineering, pp 155-164, Nov 2003.
[16] K. Fujimoto, S. Nakabayashi, "*Applying GMDH algorithm to extract rules from examples*", Systems Analysis Modelling Simulation, Vol. 43, No.10, pp.1311-1319, 2003.
[17]. S.S. Patil, V.M. Gaikwad , "*Developing New Software Metric Pattern Discovery for Text Mining*", International Journal of Computer Sciences and Engineering, Vol.2, Issue.4, pp.119-125, 2014.
[18]. L Briand, C Bunse, J Daly, "*A controlled experiment for evaluating quality guidelines on the maintainability of object oriented design*", IEEE Transaction on software Engineering, Vol.27, No.6, pp 513-530, 2001.
[19]. F. Fioravanti, P. Nesi, "*Estimation and prediction metrics for adaptive maintenance effort of object oriented systems*", IEEE Transactions on Software Engineering, Vol. 27, No.12, pp.1062-1084, 2001.
[20]. W. Li, "*Another Metric Suite for Object-oriented Programming*", The Journal of System and Software, Vol.44, No.2, pp.155-162, December 1998.
[21]. S.G. Mac Donell, "*Establishing relationships between specification size and software process effort in case environment*," Information and Software Technology, Vol. 39, No.1, pp. 35-45, 1997.
[22]. W. Li, S. Henry, "*Object-Oriented Metrics that Predict Maintainability*," Journal of Systems and Software, Vol. 23, No.2, pp. 111-122, 1993.
[23]. D.F. Specht , "*Probabilistic Neural Networks*", Journal of Neural Networks, Elsevier, Vol. 3, No1, pp.109-118, 1990.
[24]. A. G. Ivakhnenko, Y. U. Koppa, "*Regularization of decision functions in the group method of data handling*", Soviet Automatic Control, Vol.15 No.2, pp.28-37, 1970.
[25]. A. G. Ivakhnenko, "*Group Method of Data Handling- A Rival of the method of Stochastic Approximation*," Soviet Automatic Control, Vol.13, No.3, pp.43-71, 1966.

© 2017, IJCSE All Rights Reserved