Journal of Engineering and Applied Sciences 13 (Special Issue 5): 4687-4693, 2018
ISSN: 1816-949X

TreeNet Model Based Prediction of Testing Effort Class

Safa Saad A. AL-Murieb and Fryal Jassim Abd Al-Razaq
Information Technology College, Babylon University, Hillah, Iraq

Abstract: This strudy presents the use of the regression capability of the TreeNet regression tool to predict software class testing effort using a dataset obtained from Apache Ant 1.7.0. A total of twelve models with different loss regression criterion, were built with each set of predictor variables to predict the code lines (DLOC) and the Test Cases Number (NTC) of a test class. Six of the models were built using all the variables and the remaining six with subsets of the variables that were found important to their respective models. The subsets of variables were selected using a stepwise regression tool. The comparison between the TreeNet regression models results and Multi-Linear Regression (MLR) showed that the TreeNet regression models are better and further confirmed the prediction capability of the TreeNet regression tool.

Key words: Subsets, built, models, tool, regression, prediction

INTRODUCTION

Testing a software class is an important thing to determine the amount of effort required to build that class. The aggregation of the testing efforts for all the classes has a major impact on the overall effort required to build a software system (McGregor and Srinivas, 1996). Testing is an important software activity to ensure quality upon delivery.

Most related section of total cost of life-cycle of software is occupied by the activities of testing and Maintenance. Testing the software is one of the characteristics of the quality, it is done under a particular input/output (I/O). When a program tends to detect the faults and errors during the testing step, this means that a it has high testing and vice versa.

Testability is defined as a measure of testing the software, many techniques are used for measuring the testability, like using Domain Range Ratio (DRR) analysis, Neural Network (NN) analysis, Observability and Controllability (OC) analysis and Propagation, Infection and Execution (PIE) analysis. The two analysis techniques of NN and PIE give better testing's measure with comparing to others.

Obviously, by the usage of a process of software management effectively and with concentration on product, persons, process and project; a standard productiveness and quality level can be fulfill. Software projects require a planning stage with score that are uphold by a set of activities, among which the fundamental is guesses of resources, test effort, time, etc., whereas a guide and aid of the other activities they supply (Barcelos *et al*, 2007). From a very extensive review of various literatures, most of the works done in the metrics of software field have been used in the general effort prediction required in developing applications, especially Web applications. Various methods have been proposed in literature for measuring a set of software attributes that are important, like: complexity, cohesion, coupling, reliability and defects density, size, those methods were proposed for describing the applications to predict a set of factors that are related to the software quality like: reliability, effort's test, exposure of error, etc., mostly with relatively fair accuracy.

Previous researches have focused on the estimation of software testing effort in general and class testing effort in particular, at the early stages of software development, even at the risk of reduced accuracy. This led to the problem of low prediction accuracy being widely used. Our main objective in this study is to present a model to predict such software class testing effort with better accuracy than the widely used tools such as MLR. Similarly, the TreeNet prediction model has been studied and used in literature to make predictions of various software parameters mostly in comparison with other models with relatively better accuracy. In view of the bright performance of the TreeNet model reported in literature, our confidence has been confirmed by this presentation of the good features of TreeNet as a prediction model (McGregor and Srinivas, 1996).

Background: Jerome Friedman of Salford Systems Inc. developed a technology of data mining, it is TreeNet. In the case of dirty data and incomplete data, accuracy, fault tolerance with high degree and speed of burning are offered by TreeNet. The problems of regression and classification can be treated using a TreeNet in addition

Corresponding Author: Safa Saad A. AL-Murieb, Information Technology College, Babylon University, Hillah, Iraq

to its effectiveness in mining a text and numeric data. The purpose of TreeNet is designing a model of predicting with high accuracy. There is a difficultness of understanding models in detail because TreeNet tends as much as possible to achieve that aim. In a model of TreeNet, the models of classification and regression are gradually built up by collection of small trees. In normal state, the model of TreeNet composed of many dozens to many thousands of small trees, typically everyone isn't larger than (2-8) terminal nodes. The concept of the model is similar to the expansion of a long series like: Fourier or Taylor when the expansion continues a set of factors are in progress become more accurate.

The main features of TreeNet are: automatic selection from thousands of candidate predictors, the ability of data handling with needless to the pre-processing step, high speed and resistance to over-training. In addition to providing easy interpretation of resource selection functions, TreeNet also can be used with model selection and works well with complex and correlated data sets.

In terms of hardware operability, TreeNet takes advantage of Windows pre-emptive multi-tasking ability, so it can be run while switching easily between other applications. However, some stages of the TreeNet model building process may share the CPU only intermittently with other applications (Friedman, 2005).

Literature review: The TreeNet Model has been used in various areas with brilliant performance. Such areas of study include ecology, insurance (Derrig and Francis, 2008), finance (Mukkamala *et al*., 2008), networking (Beutel, 2005) and software reliability prediction (Kiran and Ravi, 2008).

The early research on metrics for class testing effort prediction lays emphasis on increasing the quality of a software product and reducing the amount of effort guesses that were required for product testing as this is considered as a necessary part of any complete project prediction (McGregor and Srinivas, 1996).

Rela (2004) presented an approach for making an experimental study for evaluating a set of metrics of object-oriented source code regarding to their possibilities to predict the effort that were needed for test. The metrics that have impact on the checking effort were evaluated. The result opened ways for further studies.

By Mookerji (2005) building a model of analysis a multiple regression for predicting the Testing Time Effort (TTE) of software development, this was done depending on many independent or predictor variables. With considering that for each model the total test cases that were created are independent variables. From a large software development project, the data were collected, normalized and analyzed. Based on our research, the TreeNet regression model has not been used in the prediction of class testing effort. The result of this study will therefore be a significant contribution to knowledge in the regression model of the TreeNet application and predict the testing effort class in particular and software engineering field in general.

MATERIALS AND METHODS

Experimental design: In order to achieve our aim of predicting software class testing effort using the TreeNet regression model, the following hypotheses were defined:

- Null hypothesis, H: the TreeNet Model's performance will not be better than that of MLR when compared
- Alternative hypothesis, H_1: the TreeNet Model's performance will be better than that of MLR when compared
- Null hypothesis, H_2: the TreeNet Models with a subset of the predictor variables will not perform better than the ones built using all the variables
- Alternative hypothesis, H_3: the TreeNet Models with a subset of the predictor variables will perform better than the ones built using all the variables

To this end, a dataset obtained from Apache Ant 1.7.0 was used. It is described in the next section.

The dataset: The data used in this research was obtained from the Apache Ant 1.7.0 project. This is an open-source Java-based building tool with the full portability of pure Java code. It was initially part of the Tomcat code base, when it was donated to the Apache software foundation.

Variables: The dataset comprises of two dependent variables and eleven independent variables.

Dependent variables: The Ant dataset consists of two dependent variables; DLOC and NTC (Table 1).

- DLOC: this is obtained by counting the code lines in a testing class
- NTC: it is the count of the number of test cases, it was

Table 1: Descriptive statistics table for the actual DLOC and NTC

Statistics	Actual DLOC	Actual NTC
Mean	52.21649485	8.597938144
Median	33	2.5
Mode	23	0
Std	62.66867011	17.83264736
Var	3927.362214	318.0033118

Table 2: Descriptive statistics table of the predictor variables

Variables	NoMeths	LCOM	iLOC	INST	RFC	MPC	FIN	FOUT	NSUP	NSUB	MI
Average	16.90	0.83	197.72	7.93	66.16	72.41	6.79	11.96	0.55	0.25	90.01
Median	10.00	0.89	69.00	5.00	33.00	27.50	0.00	8.00	0.00	0.00	79.32
Mode	3.00	0.00	44.00	0.00	26.00	12.00	0.00	8.00	0.00	0.00	104.62
Std	26.59	0.80	1172.43	9.26	214.25	305.54	26.79	13.07	0.85	0.96	61.22
Var.	707.08	0.65	1374594.00	85.79	45902.02	93356.39	717.89	170.86	0.72	0.93	3747.96

calculated by counting the calling number of J unit 'assert' approaches which happen in the test class code such as: 'assert True', 'assert False' or 'assert Equal' A descriptive statistics of the distribution of their values is summarized.

Independent variables: The dataset consists of eleven independent variables, briefly explained as follows:

- No methods: it is methods number that were in a test class
- LCOM: lack of cohesion of methods
- iLOC: it represents the intermediate code lines. This is the lines count of code in assembly-level language (Rela, 2004)
- INST: number of instance variables (Arisholm and Briand, 2006; Arisholm *et al*., 2007)
- RFC: it is the class response, it represents methods count that were executed in a class in addition to the methods number that are reachable to an object class as a result of inheritance
- MPC-depth of Message Passing Coupling (Tahvildari and Singh, 2000)
- FAN-IN: it represents a count of classes number which based a class
- F-OUT: this is classes count number which is based on a class
- NSUP: it represents super classes count number (Fioravanti and Nesi, 2000)
- NSUB: this is a count of the number of subsystems (Michael *et al*., 2002)
- MI-A measure of the Maintainability Index of a test class (Sassenburg, 2003)

A descriptive statistics of the distribution of their values is summarized in Table 2.

Building the models: In TreeNet Version 2.0 Model terminologies, the variables that are independent were denoted as predictors, dependent variables were denoted as targets. The dataset containing both the dependent and independent variables were entered into the TreeNet in a tab-delimited ASCII text format. Then, all the eleven independent variables were indicated as predictor

Table 3: p-values of selected variables for DLOC

Step	1	2	3	4
Constant	21.58	20.86	18.13	17.09
iLOC	0.258	0.233	0.119	0.124
t-value	8.540	7.460	2.090	2.200
p-value	0.000	0.000	0.038	0.029
FIN		0.590	0.730	0.730
t-value		2.620	3.170	
p-value		0.101	0.002	
INST			1.980	
t-value			2.400	
p-value			0.017	
NSUB				
t-value				
p-value				
S	53.50	52.70	52.00	51.70
R^2	27.54	30.05	32.12	33.23
R^2 (adj.)	27.16	29.32	31.04	31.82
Mallows C_p	12.00	7.000	7.000	2.200

variables and one of the dependent variables was chosen as the target variable. A model was built for each of the target variables, since, TreeNet allows only one target variable to be chosen at a time. The method that was used is leave-one-out cross-validation by indicating all the 194 observations in the V-cross validation option. This method used 193 records for training stage while the remained record is for testing. In a similar fashion, a model each was built for each of the target variables with different regression loss criterion: Huber-M, Least Absolute Deviation (LAD) and Least Squares (LS). Another similar set of models were built with a subset of the predictor variables.

The subsets were obtained using the combined forward and backward stepwise regression function of minitab 15.0 statistical software with a default alpha (p) value of 0.15. Thus, any of the predictor variables with alpha value more than 0.15 is rejected while those that are less than or equal to 0.15 are accepted to enter. During this process, out of the 11 predictor variables entered for the target variables DLOC and NTC, 4 (iLOC, FIN, INST and NSUB) were found to be important to DLOC while only 2 (FIN and MPC) were found for NTC.

The same procedure as described above was then used to build a model each for the target variables with different regression loss criterion. A total of 12 different models were built using this approach. For DLOC, no variable was entered or rejected after 4 steps. The p-values of the selected variables with their corresponding statistics are shown in Table 3. For NTC, no variable was entered or rejected after 2 steps. The p-values of the selected variables with their corresponding statistics are shown in Table 4.

Table 4: p-values of selected variables for

Steps	1	2
Constant	5.041	3.022
FIN	0.573	0.511
t-value	9.470	7.970
p-value	0.000	0.000
MPC		0.045
t-value		2.570
p-value		0.011
S	14.800	14.500
R^2	31.850	34.120
R^2 (adj.)	31.490	33.430
Mallows C_p	3.000	-1.400

Table 5: Advanced default TreeNet settings used for model building

Parameters	Values
Learn rate	0.01 (Auto)
Subsample fraction	0.50
Influence timing factor	0.10
M-regression breakdown parameter	0.90

Table 6: Defaults TreeNet limits used for model building

Parameters	Limit values
No of trees to use	200
Maximum number of trees including restart continuations	1000
Maximum nodes per tree	6
minimum number of training observation in terminal nodes	10
Maximum number of most optimal models to save summary results for	1

In building the models, the default advanced settings of the TreeNet regression tool in Table 5 were used while the default limit values in Table 6 were used. This is in order to provide a basis for valid comparisons. Table 7 shows the cross validation summary for all models with Tree size of 6 and 193 Trees grown. Also, plots of Mean Absolute Error (MAE) and number of trees for Least Absolute Deviation (LAD) and Least Squares (LS) options of the model building training and testing using the Huber-M regression loss criterion are shown in Fig. 1 and 2. Similarly, lift chart and cum lift chart of the training and testing are plotted and shown in Fig. 3 and 4. Similar LAD and LS regression loss criteria plots for other models are not shown due to space limitation.

Using the models for prediction: After the models were built for each target variable, the same input data was supplied to the models to obtain the predicted target variables. In TreeNet terminology, the process of using the built model for prediction is called scoring data. The same data was used and a data file containing the predicted target variable was obtained. The same was

Table 7: Cross validation summary for all models with tree size of 6 and 193 trees grows

		Optimal trees		Optimal criterion		
Tree Net Model option	Number of predictors	MAD	MSE	MAD	MSE	Learn sample R^2
DLOC (All-H-M)	11	145	183	31.4414597	0.324156E+04	0.3192739
DLOC (All-LAD)	11	192	193	31.1038686	0.34385E+04	0.2325778
DLOC (All-LS)	11	121	121	33.3734057	0.295418E+04	0.4227844
DLOC (Sub-H-M)	4	145	181	31.2698204	0.324304E+04	0.2714000
DLOC (Sub-LAD)	4	193	193	31.0547239	0.340454E+04	0.2050222
DLOC (Sub-LS)	4	156	170	33.0859674	0.302601E+04	0.3557407
NTC (All-H-M)	11	45	178	7.8557256	296.4593543	0.1967611
NTC (All-LAD)	11	189	193	7.7924801	316.8319717	0.0507232
NTC (All-LS)	11	137	193	8.8858459	278.2601420	0.3158516
NTC (Sub-H-M)	2	108	192	7.7262438	287.7482820	0.1589417
NTC (Sub-LAD)	2	181	193	7.6639592	311.9187269	0.0607789
NTC (Sub-LS)	2	155	181	8.5803143	267.9349898	0.2677653

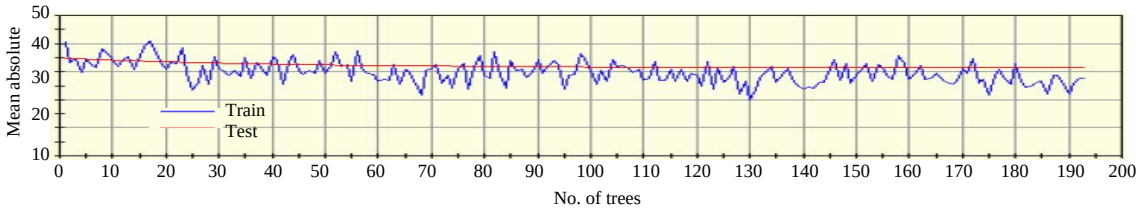

Fig. 1: A plot of MAE and number of trees for LAD option

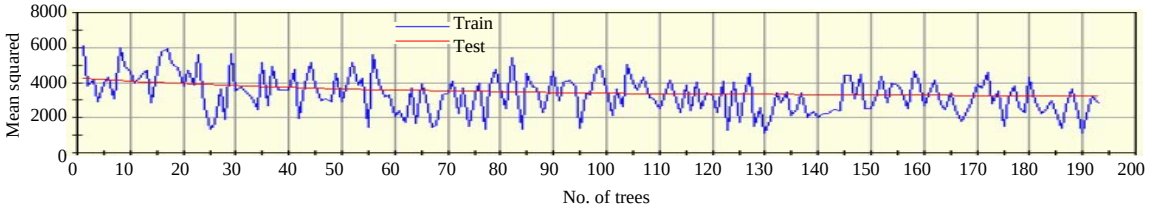

Fig. 2: A plot of MAE and number of trees for LS option

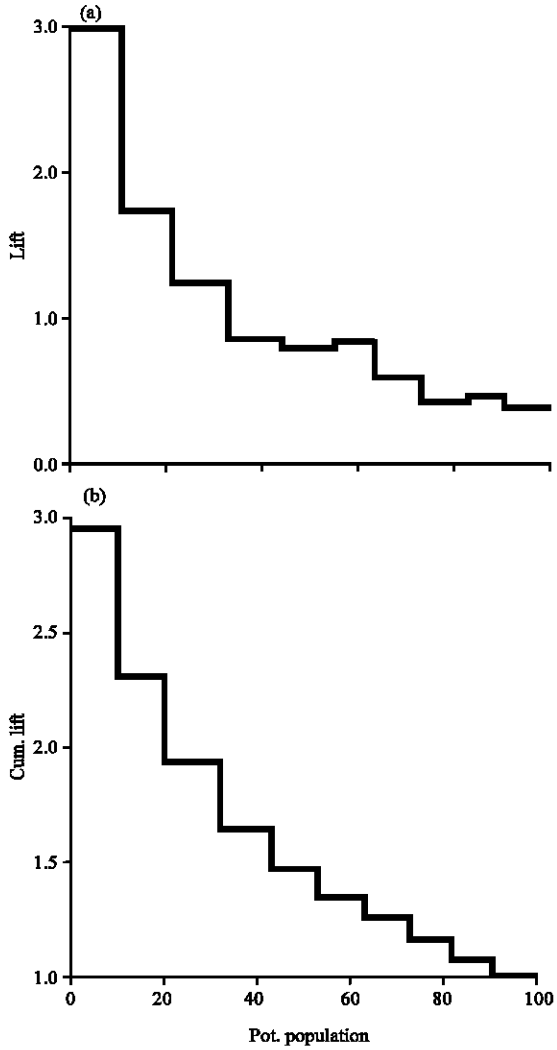

Fig. 3: a) Lift and b) Cum lift charts of the training

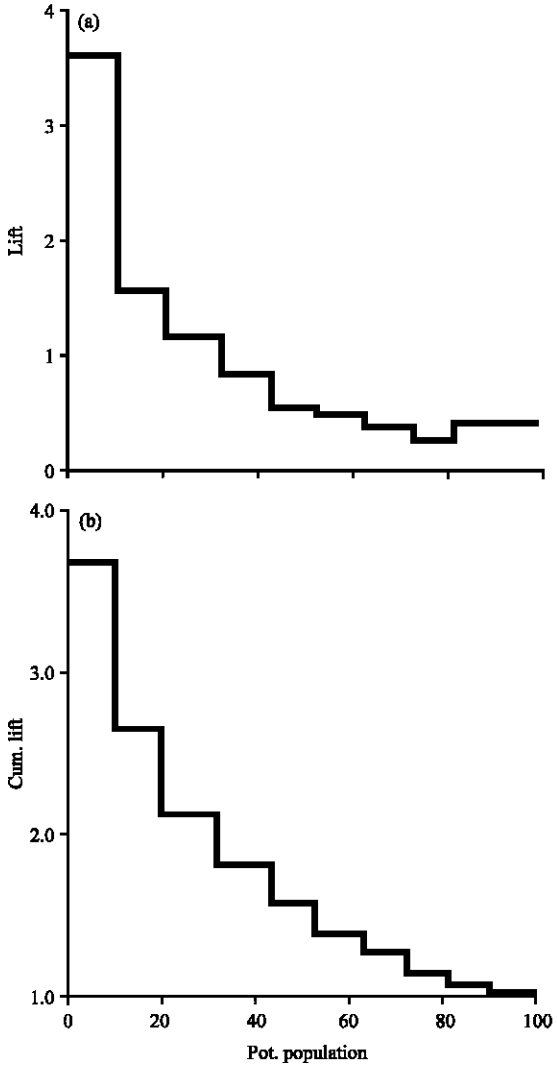

Fig. 4: a) Lift and b) Cum lift charts of the testing

done for all the 12 models built for both DLOC and NTC. Five measurements of performance evaluation: Correlation Coefficient (CC), Mean Absolute Error (MAE) Root Mean-Squared Error (RMSE), Root Relative Squared Error (RRSE) and Relative Absolute Error (RAE) were then carried out on the results and compared with the same measurements for MLR. It should be noted that the performance indices of MLR were compared with those of TreeNet models produced with all the predictor variables only and not with those built with a subset of the predictor variables. Again, this is in order to provide a basis for valid comparisons. A descriptive statistics of the distribution of the predicted values using all the 12 different models is summarized in Table 8-10. The results of the comparative analysis are discussed after the error measurements used are described.

Table 8: Descriptive statistics table for the predicted DLOC values for all variables using different TreeNet regression loss criterion and MLR (reduced to 2 decimal places)

Variables	DLOC (All-H-M)	DLOC (All-LAD)	DLOC (All-LS)	DLOC (All-MLR)
Mean	43.57	39.36	55.04	52.15
Median	35.86	32.64	44.22	37.96
Mode	33.06	24.50	35.51	20.88
Std.	19.73	18.47	26.50	38.94
Var	389.26	341.23	702.43	1516.58

Table 9: Descriptive statistics table for the predicted NTC values for all variables using different TreeNet regression loss criterion and MLR (reduced to 2 decimal places)

Variables	NTC (All-H-M)	NTC (All-LAD)	NTC (All-LS)	NTC (All-MLR)
Mean	3.68	3.76	8.79	8.72
Median	3.11	3.20	6.53	5.21
Mode	2.03	1.01	4.26	4.22
Std.	1.62	2.69	6.57	11.64
Var	2.63	7.25	43.13	135.50

Table 10: Descriptive statistics table for the predicted DLOC and NTC values for subsets of variables and different TreeNet regression loss criterion (reduced to 2 decimal places)

Variables	DLOC (Sub-H-M)	DLOC (Sub-LAD)	DLOC (Sub-LS)	NTC (Sub-H-M)	NTC (Sub-LAD)	NTC (Sub-LS)
Mean	43.23	38.38	54.11	5.07	3.75	8.83
Median	63.91	32.64	43.56	4.26	2.76	6.27
Mode	26.82	20.34	32.01	2.06	0.95	3.64
Std.	18.49	18.56	29.68	3.09	2.92	6.69
Var	341.89	344.37	880.71	9.57	8.50	44.81

Table 11: Error measurements for the predicted DLOC values for all variables with different TreeNet regression loss criteria and MLR (reduced to 2 decimal places)

Variables	DLOC (All-H-M)	DLOC (All-LAD)	DLOC (All-LS)	DLOC (All-MLR)
CC	0.64	0.61	0.65	0.37
RMSE	52.91	54.75	49.71	60.21
MAE	27.46	26.88	30.76	34.35
RRSE	0.85	0.88	0.80	0.96

Table 12: Error measurements for the predicted NTC values for all variables with different TreeNet regression loss criteria and MLR (reduced to 2 decimal places)

Variables	NTC (All-H-M)	NTC (All-LAD)	NTC (All-LS)	NTC (All-MLR)
CC	0.51	0.48	0.61	0.35
RMSE	17.71	17.35	14.71	17.33
MAE	7.49	6.98	7.60	8.96
RRSE	1.00	0.98	0.83	0.99
RAE	0.78	0.72	0.79	0.94

Accuracy measurement: Out of a number of accuracy measurements used in literature, we considered the commonly used ones in the evaluation of the excellence of the results of this study that were indicated in this study.

RESULTS AND DISCUSSION

We used the TreeNet regression tool to build 12 different models to predict the dependent variables (DLOC and NTC) using all and a subset of predictor variables as explained earlier, using the same input data. Performance evaluation was performed and compared. The tabulated results of the comparisons are shown in Table 11-13.

Generally, the Correlation Coefficient (CC) for the predicted DLOC and NTC values using all variables in Table 11-13 showed the superior performance of the TreeNet models over the conventional MLR. In particular, the models built with the Least Squares (LS) regression loss criterion performed better than all the other ones built with Huber-M (HM) and Least Absolute Deviation (LAD) criteria as shown in Table 13. Comparatively, LS is better than HM which is in turn, better than LAD. The performance of MLR did not match that of others. Also, the RMSE and RRSE performance measures lead to the same conclusion.

Table 13: Error measurements for the predicted DLOC and NTC values for subsets of variables with different TreeNet regression loss criteria (reduced to 2 decimal places)

Variables	DLOC (Sub-H-M)	DLOC (Sub-LAD)	DLOC (Sub-LS)	NTC (Sub-H-M)	NTC (Sub-LAD)	NTC (Sub-LS)
CC	0.59	0.58	0.61	0.50	0.49	0.54
RMSE	54.44	55.73	50.48	16.82	17.26	15.24
MAE	28.73	28.15	30.58	7.30	7.13	8.03
RRSE	0.87	0.89	0.81	0.95	0.97	0.86
RAE	0.74	0.73	0.79	0.76	0.74	0.83

From the result in Table 11 and 12, LS regression loss criterion proved to be better in performance than others in similar manner described for DLOC and NTC. For the subset of predictor variables in Table 13, the LS regression loss criterion also showed better performance than others (including MLR) within each target variable and the LS version of the dLOC model performed best. The RMSE and RRSE measures lead to the same conclusion.

Threats to validity

Internal: The default parameters and limit values in the model setup module of the TreeNet regression tool were used. These may not be the same for other regression tools. This is however, reasonable, since, based on our research, the TreeNet regression model has not been used in the prediction of class testing effort. We therefore suggest the use of default values for other researchers who may want to use the same tool for similar or related works. This is therefore not a valid threat but rather a way to provide a basis for valid comparisons. The same goes for the use of the default alpha (p) value in the forward and backward stepwise regression (with Leave-One-Out) function of Minitab 15.0 statistical software. Different values such as 0.01 and 0.05 have been used in several literatures, one of which is (Gui and Scott, 2007). This is also reasonable since most researchers do not explain the reason for choosing different p values. This will also provide a basis for valid comparisons.

The popular forward and backward stepwise regression used in most papers to down-select predictor variables was used rather than the variable importance result of the TreeNet regression model. So, it was not a case of MLR versus TreeNet entirely. Here, we used the stepwise regression to down-select predictor variables for use in the TreeNet Model. This was done for the same reason of establishing equal premises for valid comparisons. However, the use of the TreeNet variable important result will be a subject in our plan for future research.

External: Data collection is always a very difficult task. Ensuring the integrity of such data is even more difficult. This is because some of the data collectors may be subjective in their recordings and such recordings

may be prone to noise, repetitions or even missing values. The Apache Ant dataset we used may not be immune from all these.

RECOMMENDATIONS

As mentioned in the last section as a plan for future research we intend to use the variable importance result of the TreeNet Model to down-select predictor variables instead of using stepwise regression. We hope the use of this will be a major breakthrough in the variable subset selection process and hence lead to better prediction capabilities.

CONCLUSION

The use of TreeNet regression tool to predict the testing effort class was presented. The computed results showed the superior performance of least squares regression loss criterion of the TreeNet Model over other models, including the popular Multiple Linear Regression (MLR). This is the first research done, so far in the use of TreeNet model to predict software class testing effort. Our research has been a major contribution to knowledge in this regard as we have been able to discover the prediction capability of TreeNet regression tool using the least squares regress loss criterion.

REFERENCES

Arisholm, E. and L.C. Briand, 2006. Predicting fault-prone components in a java legacy system. Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering (ISESE'06), September 21-22, 2006, ACM, Rio de Janeiro, Brazil, ISBN:1-59593-218-6, pp: 8-17.

Arisholm, E., L.C. Briand and M. Fuglerud, 2007. Data mining techniques for building fault-proneness models in telecom Java software. Proceedings of the 18th IEEE International Symposium on Software Reliability (ISSRE'07), November 5-9, 2007, IEEE, Trollhattan, Sweden, ISBN:0-7695-3024-9, pp: 215-224.

Barcelos, D.T.I.F., D.J.D.S. Silva and N. Sant'Anna, 2007. Comparison of artificial neural network and regression models in software effort estimation. Proceedings of the International Joint Conference on Neural Networks (IJCNN 2007), August 12-17, 2007, IEEE, Orlando, Florida, USA., ISBN:978-1-4244-1379-9, pp: 771-776.

Beutel, J., 2005. Robust topology formation using BTnodes. Comput. Commun., 28: 1523-1530.

Derrig, R. and L. Francis, 2008. Distinguishing the forest from the TREES: A comparison of tree-based data mining methods. Variance, 2: 184-208.

Fioravanti, F. and P. Nesi, 2000. A method and tool for assessing object-oriented projects and metrics management. J. Syst. Software, 53: 111-136.

Friedman, J., 2005. TreeNet: An exclusive implementation of Jerome Friedman's mart methodology. Salford Systems, San Diego, California.

Grottke, M. and K. Dussa-Zieger, 2001. Prediction of software failures based on systematic testing. Proceedings of the 9th European Conference on Software Testing Analysis and Review (EuroSTAR) Vol. 280, November 19-23, 2001, Eurostar Magazine Publisher, Stockholm, Sweden, pp: 1-12.

Gui, G. and P.D. Scott, 2007. Ranking reusability of software components using coupling metrics. J. Syst. Software, 80: 1450-1459.

Kiran, N.R. and V. Ravi, 2008. Software reliability prediction by soft computing techniques. J. Syst. Software, 81: 576-583.

McGregor, J.D. and S. Srinivas, 1996. A measure of testing effort. Proceedings of the 2nd USENIX Conference on Object-Oriented Technologies (COOTS) Volume 2, June 17-21, 1996, USENIX Association, Berkeley, California, USA., pp: 10-10.

Michael, J.B., B.J. Bossuyt and B.B. Snyder, 2002. Metrics for measuring the effectiveness of software-testing tools. Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE 2002), November 12-15, 2002, IEEE, Annapolis, Maryland, USA, ISBN:0-7695-1763-3, pp: 117-128.

Mookerji, S., 2005. Prediction of testing effort in enterprise level software development using multiple regression analysis. Depaul University, Chicago, Illinois, USA. http://students.depaul.edu/~smookerj/thefinalreport.pdf

Mukkamala, S., A. Vieira and A. Sung, 2008. Model selection and feature ranking for financial distress classification. Proceedings of 8th International Conference on Enterprise Information Systems (ICEIS 2006), May 23-27, 2006, ICEIS, Paphos, Cyprus, pp: 46-53.

Rela, L., 2004. Evolutionary computing in search-based software engineering. Master's Thesis, Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland.

Sassenburg, H., 2003. SLCM: A software lifecycle model to support SPI and software metrics. Ph.D Thesis, University of Groningen, Groningen, Netherlands.

Tahvildari, L. and A. Singh, 2000. Categorization of object-oriented software metrics. Proceedings of the Canadian Conference on Electrical and Computer Engineering Vol. 1, May 7-10, 2000, IEEE, Halifax, Nova Scotia, Canada, ISBN:0-7803-5957-7, pp: 235-239.