A Replication Study of the Growth of Eclipse Defects

Deokate Sayali
Department of Computer Science
Northern Kentucky University
Louie B Nunn Dr, Highland Heights,
Kentucky, 41099.
deokates1@nku.edu

***Abstract*— Collected the Eclipse bug data from October 2001 to December 2007 from MSR 2008 Challenge website and analyzed. The data from October 2001 to December 2006 is used to train the machine. After the analysis, I found the number of defects in the Eclipse software and the growth rate of these defects. Then I used polynomial regression and an ensemble regression algorithm, called random forest algorithm to predict the future Eclipse defects. Finally, the predicted set for the year of 2007 was compared with the collected data set of the same year and a Mean Relation Error for each regression algorithm was derived. The MRE generated here should be less than 25% to assure good results. The errors were compared between the 2 prediction models to check which one produced better result.**

Keywords— Machine learning, polynomial regression, ensemble regression, random forest, Eclipse defects

1. INTRODUCTION

Eclipse is a widely used Java IDE with added plug-ins including other major programming languages such as C, C++, Python, PHP, R, Scala, etc. With such a rich client base, the smooth working of the software is very important. The software must work bug-free to satisfy the customer's need. In order to make the software more efficient and fix the issues by removing the defects, one needs to detect the bugs. Once a definite growth rate of the quantity of the defects is derived, a proper prediction model can be designed to find the number of defects in future versions of the software.

In this study, I followed the work of H. Zhang from Tsinghua University, On "An Initial study of the growth of eclipse defects" [1].The reference paper shows data analysis for period June 2004 to November 2006 and its growth pattern. After training the machine with this growth rate data, it compares the 2007 data set he collected from the same website with the 2007 Prediction data set and Mean Relative Error is calculated using below formula

$$MRE = |N - N\hat{}|/N$$

where N and N^ are the actual and estimated values, respectively.

MRE results are used to define the degree of certainty the predicted value being correct. For MRE value less than 25% then the prediction results are acceptable otherwise it widely differs from the actual value; and the prediction is not successful.

2. RESEARCH SCOPE

To achieve accurate prediction results, I considered larger period for training dataset (data from October 2001 to December 2006). With the larger set of training elements, I could train my prediction model for better performance. I performed the polynomial regression modeling on selected dataset.

As part of my extension I have used an ensemble regression algorithm, which unlike the polynomial

regression uses multiple decision-making models to tune its algorithm to find the best possible prediction curve. The random forest algorithm is a prevailing machine learning algorithm due to its efficient training method and accurate results I used random forest regression algorithm. This algorithm requires less resources while giving comparatively better results. Since the data to be fed into the training machine is purely numerical and can be formatted into a structured tabular manner, this study perfectly aligned with the requirements of the random forest regression algorithm.

The prediction models used gave different prediction curves. On comparing the Mean Relative Errors for each algorithm, one giving accurate predictions is selected

3. ACTUAL WORK

Over the years, there has been much research work has been done on defect analysis and prediction. For example, the work described in [9, 11, 12] focuses on the construction of defect prediction models based on the measurement of static code attributes, and the work reported in [2, 7, 13] analyzes the distribution of defects over modules in a large software system.

4. DATA COLLECTION

A. Generating the Data Set

In this study, I have used the public Eclipse defect data set acquired from October 2001 to December 2006. This date set is a public information and is available at the MSR 2008 Challenge website [6].

This data is provided in the form of XML format. In order to process the data, I used python script for converting the data set in CSV file. Once the CSV file is available, data fields such as different bug IDs, their Creation time stamps, Product Name and Component Name are used. Using this information, I have calculated number of bugs observed for combination of different platform and component names.

B. Finding the Growth Rate

I found two growth rates using the data, i.e. monthly growth rate and weekly growth rate.

Here I calculated monthly and weekly growth rate of the bug and a cumulative growth rate.

The training dataset comprises of 63 months of data i.e. 273 weeks in total. Using the number of months, I could create 63 data points for month and 273 for weekly data.

The monthly defect number can be represented as f(x) where x is the number of months, i.e. the first month will have f(1) defect numbers, the second month will have f(2) and so on until the 63rd month which will have f(63) number of defects. The monthly cumulative defect number can be derived by the summation of the defect numbers for each month,

$$f(m) = f(1) + f(2) + f(3) + \cdots + f(63)$$

$$f(m) = \sum_{i=0}^{63} f(i)$$

Similarly, the equation for weekly cumulative defect number can be calculated by adding all the defect numbers for each week as,

$$f(w) = \sum_{i=0}^{273} f(i)$$

I have analyzed the 14 major components of the Eclipse IDE and plot the graphs for monthly defect data and weekly defect data.

5. METHOD USED

A. Polynomial Regression Model

Using this regression technique on the growth rate of the chosen components, I derive the following formula,

$$y = \beta_2 x^2 + \beta_1 x + \beta_0$$

Where y is the number of bugs throughout the total time and x is the time. (for monthly data x=1, 2, ..., 63, for weekly data x=1, 2, ..., 273 and β2, β1, and β0 are coefficients.)

B. Random forest Regression Model

Random forests for regression are formed by growing trees depending on a random vector Θ such that the tree predictor h(x,Θ) takes on numerical values as opposed to class labels. The output values are numerical, and we assume that the training set is independently drawn from the distribution of the random vector Y, X. The mean-squared generalization error for any numerical predictor h(x) is

$$E_{X,Y}(Y-h(X))^2$$

The random forest predictor is formed by taking the average over k of the trees $\{ h(x,\Theta_k)\}$. [14]

6. REPLICATION RESULT

The paper to be replicated I have used polynomial regression so since it is a very commonly used regression method and produces accurate curves for trained model. Because of the insufficient of data set, the replication result for actual data differ than original paper [1].

And figure 1 and 2 show the growth of data for JDT.Core component.

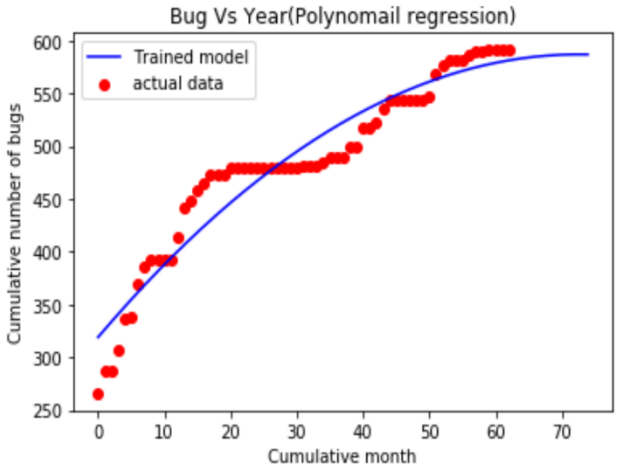

Figure 1. Graph showing defect number for JDT.Core for each Month

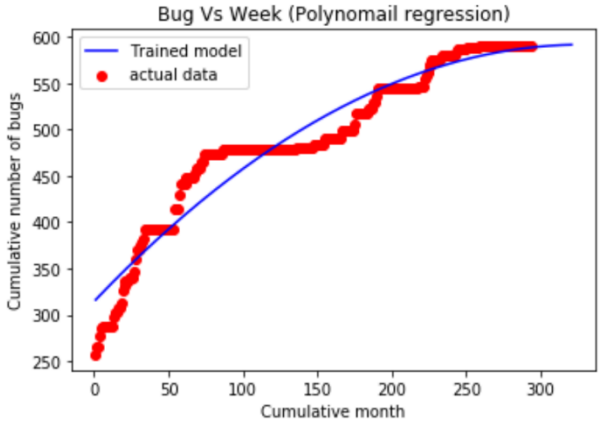

Figure 2. Graph showing defect number for JDT.Core for each Week

And figure 3 and 4 show the growth of data for 'Platform.team' component.

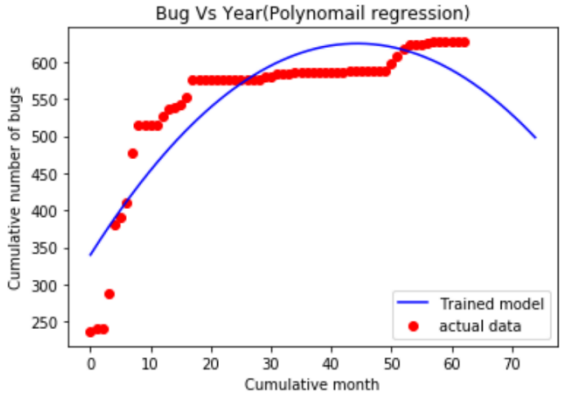

Figure 3. Graph showing defect number for 'Platform.team' for each month

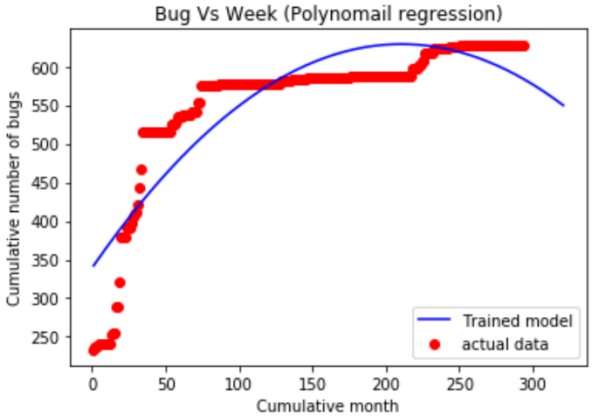

Figure 4. Graph showing defect number for 'Platform.team' for each week

The only similarity in original result graphs and replicated result graphs is the nature of the growth of defect number is increasing manner in all 14 major components.

7. EXTENSION RESULT

As part of my extension I have used ensemble regression technique to try and reduce the Mean Relative Error, since ensemble regression algorithms combine different estimators to produce a more powerful estimator which can improve the generalizability of any single estimators.

I have used the random forest regression. Upon performing the regression on the components, I can generate the generalization formula for the respective component.

Figure 5 and 6 shows ensemble Regression of Monthly defects and weekly defects for JTD.Core.

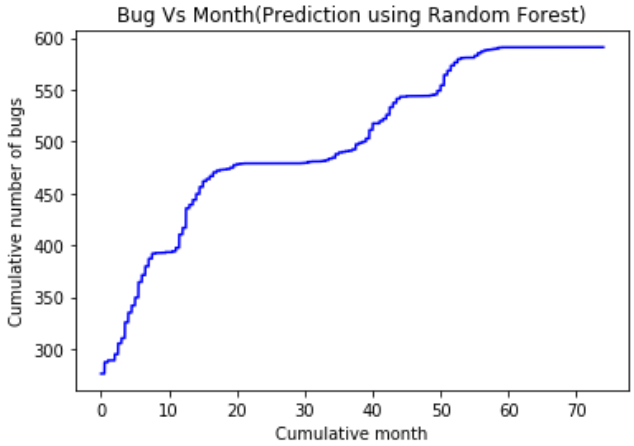

Figure 5. Ensemble Regression of Monthly defects for JDT.Core component.

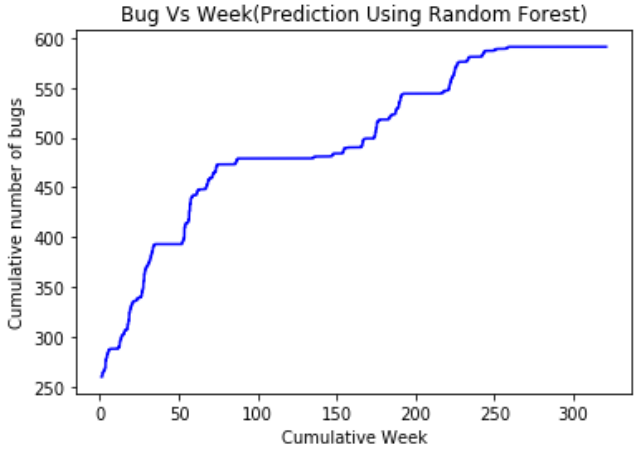

Figure 6. Ensemble Regression of Weekly defects for JDT.Core component.

Figure 7 and 8 shows ensemble Regression of Monthly defects and weekly defects for 'Platform.team'

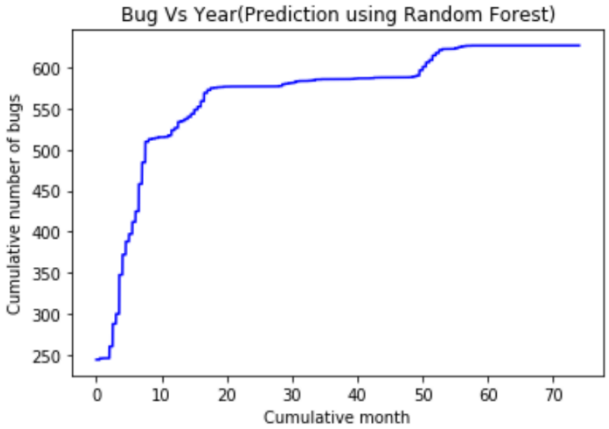

Figure 7. Ensemble Regression of Monthly defects for 'Platform.Team' component.

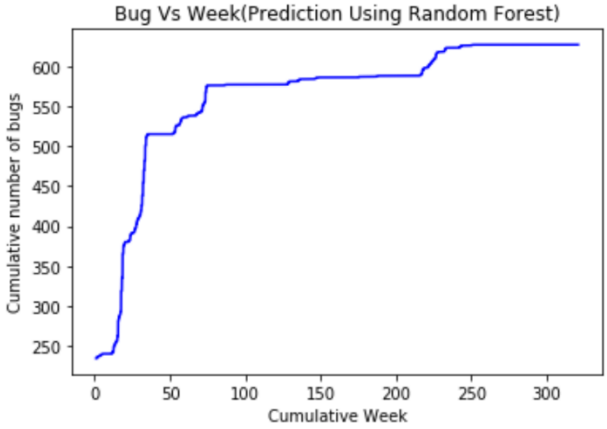

Figure 8. Ensemble Regression of Monthly defects for 'Platform.Team' component.

On comparing the actual data curve from figures 1 and 2 with the curve from figure 5 and 6, we can notice the actual data set curve is like ensemble regression curves respectively.

We can notice the same phenomenon for figures 3 and 4 on comparing with figures 7 and 8 for 'Platform.Team' component.

8. PREDICTING THE NUMBER OF FUTURE DEFECTS

Now that we have trained the machine with the growth rates of the quantity of Eclipse defects from October 2001 to December 2006, we should try and predict the number of defects that may occur in future versions of the Eclipse SDK. Upon calculating the predicted value, we need to find the Mean Relative Error using MRE = |N – N^|/N (and multiply with 100 to find the percentage error),

which will show us the accuracy of the prediction for each component. The MRE must be less than or equal to 0.25 (or 25%, if calculated in percentage) for the prediction to be accurate.

i. Polynomial Regression:

Like the original study, we performed the polynomial regression algorithm on our chosen components from the Eclipse defect data collected. The constraints given in this project aligned closely with the requirements of the algorithm.

Here I will be using the 63 data points from October 2001 to December 2006 to predict the defect values of the future 12 data points from January 2007 to December 2007. To calculate the weekly MRE values I will be using the 273 weekly Eclipse defect data points we collected from October 2001 to December 2006 and predict the future 52 weeks defect numbers.

ii. Ensemble Regression:

Ensemble regression algorithm uses a variety of machine learning algorithms to train the machine

The input it requires is much less compared to regular polynomial regression but gives a more accurate result.

Random forest algorithm uses multiple decision trees to make the prediction. Decision trees use a tree structure to enter each decision into a node and uses believability values decide which direction to take.

We are using the same 63 data point training set to train the machine and the same 12 data point testing set to predict the future number of bugs.

To calculate the weekly MRE values I will be using the 273 weekly Eclipse defect data points we collected from October 2001 to December 2006 and predict the future 52 weeks defect numbers.

Table 1 and 2 show the Mean Relative Error derived from the data set using polynomial regression method for monthly and weekly data respectively.

Table 3 and 4 show the Mean Relative Error deduced from the data set using our extension, i.e., the ensemble regression technique Random Forest for monthly and weekly data points respectively.

Component	No. of Actual Defects	No. of Predicted Defects	Mean Relative Error in Percent	MRE <= 25%?
Platform.UI	20017	19560	2.28	Yes
JDT.UI	14647	13426	8.33	Yes
Platform.SWT	11636	11615	0.17	Yes
Platform.Team	7539	6461	14.29	Yes
JDT.Core	7115	7025	1.25	Yes
JDT.Debug	5582	4823	13.57	Yes
PDE.UI	3913	3742	4.36	Yes
Platform.Debug	11636	11615	0.17	Yes
Platform.Resources	2108	1175	15.77	Yes
Platform.Update	2042	1628	2.24	Yes
JDT.Text	1878	1996	6.3	Yes
Platform.User Assitance	1836	1681	8.4	Yes
Platform.Ant	1512	1443	4.53	Yes
Platform.Compare	1171	1034	8.4	Yes

Table 1: Monthly MRE Calculated using Polynomial Regression

Component	No. of Actual Defects	No. of Predicted Defects	Mean Relative Error in Percent	MRE <= 25%?
Platform.UI	46940	45976	2.05	Yes
JDT.UI	34207	32172	5.94	Yes
Platform.SWT	27227	27164	0.22	Yes
Platform.Team	17614	15849	10.01	Yes
JDT.Core	16642	16541	0.61	Yes
JDT.Debug	13144	1176	10.41	Yes
PDE.UI	9151	8921	2.51	Yes
Platform.Debug	27227	27164	0.23	Yes
Platform.Resources	4930	4355	11.65	Yes
Platform.Update	4788	4110	14.15	Yes
JDT.Text	4396	4558	3.7	Yes
Platform.User Assitance	4318	4097	5.1	Yes
Platform.Ant	3528	3411	3.29	Yes
Platform.Compare	2743	2053	5.1	Yes

Table 2 : Weekly MRE Calculated using Polynomial Regression

Component	No. of Actual Defects	No. of Predicted Defects	Mean Relative Error in Percent	MRE <= 25%?
Platform.UI	20017	19872	.724	Yes
JDT.UI	14647	14628	.129	Yes
Platform.SWT	11636	11568	.584	Yes
Platform.Team	7539	7524	.198	Yes
JDT.Core	7115	7092	.323	Yes
JDT.Debug	5582	5520	1.11	Yes
PDE.UI	3913	3900	.332	Yes
Platform.Debug	2611	2604	.268	Yes
Platform.Resources	2108	2100	.379	Yes
Platform.Update	2042	2028	.685	Yes
JDT.Text	1878	1872	.319	Yes
Platform.User Assitance	1836	1812	1.307	Yes
Platform.Ant	1512	1512	0	Yes
Platform.Compare	1171	1164	.547	Yes

Table 3 : Monthly MRE Calculated using Ensemble Regression

Component	No. of Actual Defects	No. of Predicted Defects	Mean Relative Error in Percent	MRE <= 25%?
Platform.UI	46940	46368	1.218	Yes
JDT.UI	34207	34132	.219	Yes
Platform.SWT	27227	26992	.863	Yes
Platform.Team	17614	17556	.329	Yes
JDT.Core	16642	16548	.564	Yes
JDT.Debug	13144	12880	2.00	Yes
PDE.UI	9151	9100	.557	Yes
Platform.Debug	6106	6076	.491	Yes
Platform.Resources	4930	4900	.608	Yes
Platform.Update	4788	4732	1.168	Yes
JDT.Text	4396	4368	.636	Yes
Platform.User Assitance	4318	4228	2.084	Yes
Platform.Ant	3528	3528	0	Yes
Platform.Compare	2743	2716	.984	Yes

Table 4:. Weekly MRE Calculated using Ensemble Regression

9.COMPARISON OF REPLICATION AND EXTENSION

As we can see from the tables below, our prediction algorithm, the ensemble regression algorithm gave us a significantly smaller MRE than the MRE produced by the formula used in the original study, i.e., the polynomial regression method for both month and week. The results that we produced were as predicted.

Table 5 and 6 shows the comparison between the MREs produced using polynomial regression and those produced using the ensemble regression algorithm, random tree using the monthly bug defect data points and weekly bug defect data points respectively.

Component	Polynomial Regression MRE	Ensemble Regression MRE	Better/Worse	Difference
Platform.UI	2.28	.724	Better	1.556
JDT.UI	8.33	.129	Better	8.201
Platform.SWT	0.17	.584	Worse	-0.414
Platform.Team	14.29	.198	Better	14.092
JDT.Core	1.25	.323	Better	0.927
JDT.Debug	13.57	1.11	Better	12.46

PDE.UI	4.36	.332	Better	4.028
Platform.Debug	0.17	.268	Worse	-0.098
Platform.Resources	15.77	.379	Better	15.391
Platform.Update	2.24	.685	Better	1.555
JDT.Text	6.3	.319	Better	5.981
Platform.User Assitance	8.4	1.307	Better	7.093
Platform.Ant	4.53	0	Better	4.53
Platform.Compare	8.4	.547	Better	7.853

Table 5: Monthly Comparison of MREs

Component	Polynomial Regression MRE	Ensemble Regression MRE	Better/Worse	Difference
Platform.UI	2.05	1.218	Better	0.832
JDT.UI	5.94	.219	Better	5.721
Platform.SWT	0.22	.863	Worse	-0.643
Platform.Team	10.01	.329	Better	9.681
JDT.Core	0.61	.564	Better	0.046
JDT.Debug	10.41	2.00	Better	8.41
PDE.UI	2.51	.557	Better	1.953
Platform.Debug	0.23	.491	Worse	-0.261
Platform.Resources	11.65	.608	Better	11.042
Platform.Update	14.15	1.168	Better	12.982
JDT.Text	3.7	.636	Better	3.064
Platform.User Assitance	5.1	2.084	Better	3.016
Platform.Ant	3.29	0	Better	3.29
Platform.Compare	5.1	.984	Better	4.116

Table 6 : Weekly Comparison of MREs

9. CONCLUSION

Using the data set provided, we were able to create a training set and a testing set by separating the large data set (consisting of the bug report of Eclipse bugs from October 2001 to December 2007) into the training set (consisting of the defect numbers from October 2001 to December 2006) and the testing data set (consisting of the defect numbers from January 2007 to December 2007). Although parsing of some files was observed, a sufficient data set was created from the xml files that were passable.

The training data set is used to generate the growth rate of the defects of each component over time and finally used to make the prediction algorithm using polynomial regression algorithm (like in the replication paper) and random forest regression (the extension that I added). The polynomial regression was used since it is a simple algorithm which can fit a large variety of curves and provides an accurate result. Random forest was chosen since the input for this model is laid out in a structured tabular fashion and has numerical values, which the algorithm can accurately compute. The ensemble regression method used is one of the most effective algorithms in machine learning to make predictive methods. This algorithm makes predictions based on the decisions from a series of base models. The

algorithm combines these decisions and produces a very accurate predictive analysis. In random forest algorithm, each decision is affected by a previous decision made by the tree-like structure with each node being a single decision. Each of these tree-like structures is called a decision tree and random forest algorithm uses multiple decision trees to make its final decision.

As we see from Table 5 and table 6, the MRE values produced by the ensemble regression technique is much lower than that produced by the original study, which used polynomial regression model. The reduced MRE is also evident from the prediction graphs produced upon training the data sets with the different algorithms. This shows the competence and superiority of the random forest regression algorithm over the polynomial regression technique.

10. REFERENCE

[1] H. Zhang, On "An Initial study of the growth of eclipse defects". *5th Working Conference on Mining Software Repositories (MSR 2008).* Leipzig, Germany, 2008.

[2] C. Andersson and P. Runeson, A Replicated Quantitative Analysis of Fault Distributions in Complex Software Systems, *IEEE Trans. Software Eng.*, 33 (5), pp. 273-286, May 2007.

[3] L.C. Briand and I. Wieczorek, Resource estimation in software engineering, *Encyclopedia of Software Engineering*, (ed. J. Marcinak), Wiley, 2002. 1160-1196.

[4] J. R. Busemeyer and Y. Wang, Model comparisons and model selections based on generalization test methodology, *Journal of Mathematical Psychology*, June 1999.

[5] J. Cohen, *Applied Multiple Regression/ Correlation Analysis for the Behavioral Sciences*, Lawrence Erlbaum Associates, 2003.

[6] Eclipse bug dataset, available at the MSR 2008 Mining Challenge website http://msr.uwaterloo.ca/msr2008/.

[7] N. Fenton and N. Ohlsson, Quantitative Analysis of Faults and Failures in a Complex Software System, *IEEE Trans. Software Eng.*, 26 (8), Aug 2000. pp. 797-814.

[8] H. Joshi, C. Zhang, S. Ramaswamy, C. Bayrak, Local and Global Recency Weighting Approach to Bug Prediction, *Proc. 4th Int'l workshop on Mining Software Repositories (MSR 2007)*, Minneapolis, USA, 2007.

[9] T. Menzies, J. Greenwald and A. Frank, Data Mining Static Code Attributes to Learn Defect Predictors, *IEEE Trans. Software Eng.*, vol. 32, no. 11, 2007.

[10] Rao, C. R., Prediction of Future Observations in Growth Curve Models, *Statistical Science*, vol. 2(4), 434-447, 1987.

[11] H. Zhang and X. Zhang, Comments on "Data Mining Static Code Attributes to Learn Defect Predictors", *IEEE Trans. On Software Eng.*, vol. 33(9), Sep 2007.

[12] H. Zhang, X. Zhang, M. Gu, Predicting Defective Software Components from Code Complexity Measures, *Proc. 13th IEEE Pacific Rim Int'l Symp. on Dependable Computing Conference (PRDC 2007)*, Melbourne, Australia, Dec 2007, IEEE Press, pp. 93-96.

[13] H. Zhang, On the Distribution of Software Faults, *IEEE Trans. on Software Eng.*, vol. 34(2), 2008.

[14] Leipzig, Germany, 2008. L. Breiman, A. Cutler, Random forests, https://www.stat.berkeley.edu/breiman/RandomForests (2001).

[15] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, 1984.

[16] Holger Schackmann, Henning Schaefer, Horst Lichter, Evaluating Process Quality Based on Change Request Data --- An Empirical Study of the Eclipse Project, Proceedings of the International Conferences on Software Process and Product Measurement, November 04-06, 2009, Amsterdam, The Netherlands

[17] Xiaoyan Zhu, Qinbao Song, Zhongbin Sun, An Empirical Analysis of Software Changes on Statement Entity in Java Open Source Projects, International Journal of Open Source Software and Processes, v.4 n.2, p.16-31, April 2012

[18] Jue Wang, Hongyu Zhang, Predicting defect numbers based on defect state transition models, Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement, September 19-20, 2012, Lund, Sweden

[19] Sandeep Krishnan, Chris Strasburg, Robyn R. Lutz, Katerina Goševa-Popstojanova, Are change metrics good predictors for an evolving software product line? Proceedings of the 7th International

Conference on Predictive Models in Software Engineering, p.1-10, September 20-21, 2011, Banff, Alberta, Canada

[20] Sandeep Krishnan, Chris Strasburg, Robyn R. Lutz, Katerina Goseva-Popstojanova, Karin S. Dorman, Predicting failure-proneness in an evolving software product line, Information and Software Technology, v.55 n.8, p.1479-1495, August 2013