Empirical validation of human factors in predicting issue lead time in open source projects

Nguyen Duc Anh
IDI- NTNU
Trondheim, Norway
anhn@idi.ntnu.no

Daniela S. Cruzes
IDI- NTNU
Trondheim, Norway
dcruzes@idi.ntnu.no

Claudia Ayala
Technical University of Catalunya,
Department of Service Engineering
and Information Systems
Barcelona, Spain
cayala@essi.upc.edu

ReidarConradi
IDI- NTNU
Trondheim, Norway
conradi@idi.ntnu.no

ABSTRACT

[Context] Software developers often spend a significant portion of their resources resolving submitted evolution issue reports. Classification or prediction of issue lead time is useful for prioritizing evolution issues and supporting human resources allocation in software maintenance. However, the predictability of issue lead time is still a research gap that calls for more empirical investigation. **[Aim]** In this paper, we empirically assess different types of issue lead time prediction models using human factor measures collected from issue tracking systems. **[Method]** We conduct an empirical investigation of three active open source projects. A machine learning based classification and statistical univariate and multivariate analyses are performed. **[Results]** The accuracy of classification models in ten-fold cross-validation varies from 75.56% to 91%. The R^2 value of linear multivariate regression models ranges from 0.29 to 0.60. Correlation analysis confirms the effectiveness of collaboration measures, such as the number of stakeholders and number of comments, in prediction models. The measures of assignee past performance are also an effective indicator of issue lead time. **[Conclusions]** The results indicate that the number of stakeholders and average past issue lead time are important variables in constructing prediction models of issue lead time. However, more variables should be explored to achieve better prediction performance.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics - *process metrics*; D.4.8 **[Operating Systems]**: Performance - *Measurements, Modeling and prediction*

General Terms

Measurement, Human Factors, Experimentation, Verification

Keywords

Bug prediction, bug triage, regression model, classification model, empirical study, issue lead time, bug lead time, issue resolution time.

PROMISE '11, September 20-21, 2011, Banff, Canada

1. INTRODUCTION

Software maintenance and evolution are the most expensive activities in the software lifecycle process, consuming 60 to 80% of the total effort spent on a software system [5, 15, 22]. Dealing with software evolution issues includes fixing bugs, implementing new feature requests, or enhancing current system features. As software engineers often spend significant effort on resolving submitted evolution issue reports, suitable mechanisms to resolve these issues in a cost-effective manner are desirable. Ideally, the more important issues, for example serious bugs or highly desired requests, should be solved first, and the less important issues can be executed later. Hence, predicting issue lead time is a crucial task in supporting both project managers and developers. Issue lead time represents the amount of time between initiation and resolution of an issue. For managers, this information can enable appropriate resource allocation and project scheduling. For developers, it allows them to decide the order in which to work on issues.

Predicting issue lead time is a difficult task mainly due to the dynamics of software evolution and lack of clear understanding of the factors influencing issue lead time. The literature on prediction models reveals some studies' attempts to classify bug fixing times into predefined categories by using machine learning techniques based on historical databases of issues. For instance, classifying bug lead times into slow or fast categories could achieve a precision of 60 to 75% [1, 3, 15, 18]. Furthermore, some studies have attempted to predict the exact bug fixing time by fitting the historical data into a mathematical model, which is often a linear regression model. For instance, linear regression models that predict bug fixing time have R^2 values between 0.30 and 0.98 [20, 25]. These prediction results are heterogeneous, depending on the choices of datasets, prediction techniques, and set of input variables. Consequently, it is necessary to conduct more empirical research to confirm and enhance issue lead time prediction models.

Measures of human factors (e.g., collaboration among developers, capacity of developers, or developers' past performance) have been recognized as an important element affecting defect prediction models [4, 16, 18, 19]. A human factor is defined as *"a physical or cognitive property of an individual or social behavior, which is specific to humans and influences functioning of technological systems as well as human–environment equilibriums"* [30]. While research on predicting issue lead time is often based on the issue report characteristics, such as issue priority [1, 3, 15, 25], issue creation time [1, 3, 15], and version [1, 3, 15], integration of human factors into models is still limited.

The main objective of this empirical study is to extend previous research on issue lead time prediction models by:

- Confirming the predictability power of issue lead time prediction models using human factor measures and comparing it with results from existing classification and linear regression models. The following research questions are addressed:

  RQ1. Do human factor metrics improve classification of issue lead time?

  RQ2. Which characteristics of issues increase the predictive power of a linear regression model for predicting issue lead time?

On the one hand, we deal with collaboration metrics such as the number of comments on the issue and the number of stakeholders associated with the issue. Stakeholders are people involved in software development projects such as developers, testers, and end-users. On the other hand, we also introduce a stakeholder past-performance metric to measure the effect of the stakeholder experience. The research questions addressed are:

RQ3. Which human factor metrics contribute significantly to issue lead time prediction in the linear regression models?

To carry out this research we made use of data from three Open Source Software (OSS) projects. The bug tracking systems used in these projects provide us with thousands of issue report data.

The rest of the paper is organized as follows. Section 2 presents the relevant studies on prediction of lead time. Section 3 presents the research questions, investigated variables, and description of our case study. Section 4 provides the results of the data analysis, including the descriptive analysis of datasets, classification model, and univariate and multivariate regression analyses. Section 5 discusses the findings and Section 6 the threats to validity. The last section is devoted to the conclusions and future work.

2. RELATED WORK

In the following subsections we summarize some related work that was used as background or motivation to state our research questions.

2.1 Bug Fixing Time Prediction Models

Various authors have investigated diverse models for predicting bug fixing time.

On one hand, some researchers have investigated classification models. For instance, Panjer constructed a septenary classification model of bug fixing time using several machine learning and statistical techniques [1]. Using the Eclipse dataset, the author found that the number of comments and information about the assignee (i.e., the person(s) in charge of solving the bug) were the most important variables in his classification model. Giger et al. conducted a similar investigation using other OSS datasets. The obtained classification achieved an average predictive power of 65% for precision and 75% for recall [3]. The authors also found that post-submission data, namely the number of comments on the bug reports, improved the performance of the prediction models by 5 to 10%. Bougie et al. validated a classification model of bug lifetime using bug report features. The most suitable features for prediction were category, severity, priority, and confidentiality. The authors also found that the predictive accuracy of the classification algorithms varies between software projects [15].

On the other hand, other researchers have focused on linear regression models. Anbalagan et al. built a linear regression model using the number of unique participants [20]. The resulting goodness-of-fit statistic (R^2) of the model was very high: from 0.80 to 0.98. Bhattacharya et al. also investigated a multivariate linear regression model using the number of developers and some bug report features [25]. Their reported R^2 ranged from 0.3 to 0.49.

Given that we detected a considerable diversity in the results related to prediction of bug fixing time, it is important to confirm the predictability of these proposed models by addressing RQ1 and RQ2. In addition, since the maintenance and evolution task comprises not only bug fixing but also dealing with other types of evolution issues, such as feature requests or any other task request (e.g., updating documentation), we consider the prediction models of general evolution issues.

2.2 Human Related Metrics in Predicting Software Quality and Productivity

Recently, there has been an increasing interest in research exploring the impact of human factor metrics on software quality and productivity. It is argued that the capability of developers and how they collaborate in developing a software module can affect how likely they are to introduce bugs in the module [16, 17, 24, 26, 27].

Some studies have explored diverse human related metrics. Herbsleb et al. formed and validated an empirical theory of coordination in software engineering [26]. The authors concluded that developers who have more work assigned to them are less productive. Hooimeijer et al. investigated bug report features to predict whether a bug report is classified within a certain amount of time [17]. The authors proposed a submitter reputation metric, which is the ratio of the number of resolved issues to that of opened issues. Their results show that a higher severity rate correlates weakly with shorter bug report turnarounds. The number of comments on the issue was the most important feature.

Guo et al. empirically studied the relationship between people involved in handling the bug and their influence on which bugs get fixed in Windows Vista and Windows 7 [16]. They concluded that developers who have successfully fixed bugs in the past are more likely to successfully fix their assigned bugs in the future. In addition, their results show that the greater the number of people editing a bug report, the more likely it is that the bug will be fixed. Abreu et al. investigated a subproject in Eclipse and found a statistically significant positive correlation between communication frequency among developers and number of injected bugs in the software [24].

Ostrand et al. investigated the usefulness of information about the programmers in predicting software faults [27]. The authors augmented the fault prediction model with a variable that counted the cumulative number of different developers who had modified the bug file; in addition, they assessed the influence of individual developers on the introduction of faults into the files. In contrast to some other research [16, 24, 26], their results showed that the past performance of individual developers was an ineffective variable for improving the prediction models.

While these studies confirm that the integration of human factors in prediction models is worth studying, success in the use of some metrics was inconsistent among these studies. Therefore, we decided to investigate some of these inconsistent metrics and also to propose other metrics that have not been used in the context of issue lead time prediction models (i.e., developers' past performances) to confirm (or not) their effectiveness. This led us to state RQ3.

3. RESEARCH DESIGN

This section provides details about the research techniques and variables used. The dependent variable in this study is issue lead time, which is the period of time (in days) from when the issue is created until when it is resolved and closed. This variable is computed based on the *creation time* and the *issue resolved time*.

In order to analyse the predictive power of classification and linear regression models, we decided to replicate some previous studies (as those cited in section 2.1) using different sets of issue report features as input variables. The accuracy of the models is then compared to assess the effectiveness of each set of input variables. This led us to answer RQ1 and RQ2. To validate the effectiveness of some human factor metrics, we integrated such metrics as variables into the models and assessed them via univariate and multivariate analyses. In addition, we also introduced a new variable (that had not been used before in the context of predicting issue lead time), namely stakeholder past performance, in order to assess whether it might improve the predictive power of issue lead time prediction models. In this way we answered RQ3. The following subsection details the set of variables used in the study.

3.1 Input Variables

We assume that the issues are collected in a time frame $[t_0, t_{end}]$. As described in Figure 1, for each issue, we consider some specific milestones:

- t_0: the time when the project started (past)
- t_1: the time when the issue is created (present)
- t_2: some time in the future after the creation of the issue (near future) and a period of time $\Delta t = t_2 - t_1$
- $t_{resolved}$: the time when the issue is resolved (future)

Figure 1 shows the time frame used as the backbone of the study. Please note that some variables are gathered and used in different time periods. During the period $[t_0, t_1]$, data about *stakeholder past performance* is collected. t_1 marks the point when the issue is created, and therefore the *issue description features* are collected. During the period $[t_1, t_2]$, data about *submission cooperation* are collected. Thus, stakeholder past performance, issue description features, and submission cooperation represent the three input variable dimensions used in this study. Below there is a description of the individual metrics belonging to each input variable dimension.

Issue description features consist of issue report features that characterize the description, location, and time of occurrence of the issue:

- **Description length**: This is the length of the issue descriptions measured in number of words. The variable is selected as a conventional indicator of the quality of the issue report [5]. We assume that an issue description of better quality could allow the issue to be resolved in a shorter period of time.
- **Issue priority**: uses an ordinal scale, including: blocker, critical, major, minor, and trivial. In this study, we convert it into a numerical scale (5 to 1). The intuitive reasoning is that the higher the priority of an issue, the faster it should be resolved.
- **Issue type**: This indicates whether an issue is a bug, feature-request, or other evolution task. The variable is included in the classification model to investigate whether there is a difference between bug issues and other types of evolution issues.

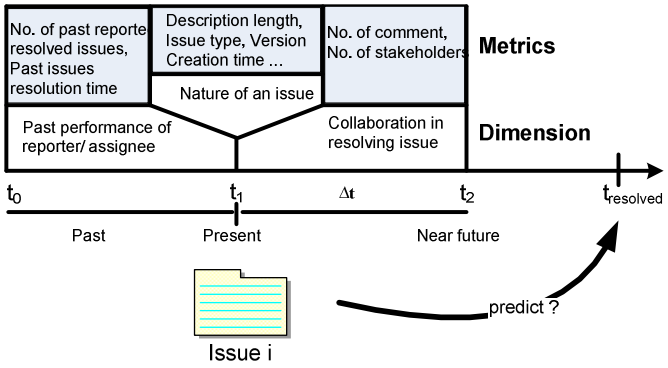

Figure 1: Three input variable dimensions

- **Version**: The software version in which an issue occurs.
- **Creation time**: The month and year when the issue is created [3, 15]. The version and creation time provide the temporal context of an issue since the data are collected over a long period of time and across multiple versions.

Stakeholder past performance includes measures of the experience of the reporter (i.e., the person who reports the issue) and the assignee (i.e., the developer assigned to solve the issue) in reporting and resolving issues using historical data from the issue tracking systems. The category includes the following metrics:

- Reporter experience (ExpR) of a reporter rep_j at time t_1 is measured by the number of previous issues reported by the reporter prior to time t_1:

$$\exp r(rep_j, t_1) = count(iss) : t_{created\_iss} < t_1$$

- Assignee experience (ExpA) of an assignee ass_j at time t_1 is measured by the number of previous issues resolved by the assignee prior to time t_1:

$$\exp a(dev_j, t_1) = count(iss) : t_{resolved\_iss} < t_1$$

- Average past issue lead time (Apit) of a developer ass_j at time t_k is calculated as an average of the lead time of k – 1 previous issues resolved by a developer:

$$apit(dev_j, t_k) = \frac{\sum_{i=1}^{k-1} t_i}{k-1} \begin{cases} t_i = t_{resolved\_i} - t_{created\_i} : t_{resolved\_i} < t_1 \\ t_i = t_1 - t_{created\_i} : t_{resolved\_i} \geq t_1 \end{cases}$$

Post submission collaboration includes stakeholders' comments on other's works. These data are collected during the period of time $[t_1, t_2]$ after the issue is created. The selection of the time point t_2 is of practical importance for the prediction model since the model can be used only after this time. Therefore, the model is not meaningful if t_2 is near or equal to $t_{resolved}$. Giger et al. found that the number of comments made within 14 days or after 31 days is useful for improving the predictive power of a classification model [3]. In this study, we set $\Delta t = 14$. We investigate the following variables:

- The number of comments (NoC) of an issue iss_i is the number of comments made on the issue within 14 days after the issue creation time.

$$noc(iss_i) = count(c) : t_{comment\_c} \in [t_1, t_2]$$

- The number of involved stakeholders (NoS) of an issue iss_i is the number of unique stakeholders who comment on the issue within 14 days after the issue creation time.

$$nos(iss_i) = count(s) : \exists c_j(s) : t_{comment\_c_j} \in [t_1, t_2]$$

Table 1 summarizes the research techniques applied and the metrics used in each of these three techniques. Please note that the metrics used for the first time in this study are marked by an asterisk.

Table 1: Mapping of research questions, methodology, and input variables

Research technique/input variables	Classification (R1)	Correlation analysis (R3)	Linear regression (R2, R3)
Description length*	X	X	X
Issue priority [1, 3, 15, 25]	X		
Version [1, 3, 15]	X		
Issue type [1, 3, 15]	X		
Creation time [3, 15]	X		
ExpR*	X	X	X
ExpA*	X	X	X
NoC [1, 3]	X	X	X
NoS [20, 25, 27]	X	X	X
Apit*	X	X	X

3.2 Data Collection

Three OSS projects were selected for our study, namely Qt, Qpid, and Geronimo. These projects were selected for the following reasons:

(1) They are medium-size, active, and ongoing OSS projects which have been taking place for at least the last four years, which ensures the scale of the datasets.
(2) The issue tracking systems used in these projects are similar, which facilitates the data collection.
(3) The projects are similar at the business domain and technical levels, which reduces error due to heterogeneous data and makes the prediction results comparable among projects.

Issue report features, stakeholder past performance, and post-submission collaboration data were collected from the issue tracking system and mailing list. The data extraction process was executed using a small Perl script. The stakeholder name and nickname are checked to remove redundancy in the number of stakeholders.

As stated before, issue lead time is the period of time from when the issue is created until the time it is resolved and closed and is computed based on the *creation time* and the *issue resolution time*. Therefore, we only select issues with resolution type "RESOLVED" and status "CLOSED". Issues with resolution types such as "DUPLICATE", "WONTFIX", and "INVALID" were removed from our analysis. Pending issues were not used since they did not have actual resolution times that could be used for model evaluation. The software issues with invalid creation dates such as "do not report creation date", "invalid resolution date", or "do not report resolution date" were also filtered out from the datasets. After the filtering process, 16986 out of 25531 issues were selected for further analysis. Table 2 shows a summary of datasets describing the main firm owning each project, the time frame of the issues collected for analysis, the total number of issues, the number of stakeholders and reporters of issues (people who collaborated on the project during this period), the total number of issues in the repository, and the total number of issues that we used for our analysis. In our dataset, the earliest issue was created in September 2003 and latest in December 2009.

Table 2: Case studies description

Info.\Projects	Qt	Qpid	Geronimo
Main organization involved	Qt (Nokia)	Red Hat, JP Morgan	IBM
Collection time frame	85 months	51 months	87 months
Number of stakeholders	133	39	60
Number of issues	16818	3016	5697
Number of selected issues	9921	2278	4787

The priority values of 1586 issues in three projects were missing. For classification models, we left the missing values blank since the decision tree algorithm J48 is able to handle missing values using the distribution of priority values in training datasets [14]. For linear regression models, since we converted the priority values into a numerical scale, the missing values were replaced by the most frequent priority value in the project.

4. DATA ANALYSIS

4.1 Descriptive Analysis

Table 3 describes the distribution of numeric input variables. Overall we observe the occurrence of skewness and outliers in the datasets. The standard deviations of description length, reporter experience, assignee experience, average past issue lead time, and issue lead time are high. The standard deviations, medians, and mean values of number of comments and number of stakeholders are low.

There are many issues which have empty parts of descriptions while the main informative descriptions lie in the issue titles and are further explored in comments on the issue. The long issue descriptions often include a stack trace or source code. The most extreme case is the description with 62489 words consisting of the stack trace, source code, and description of the bug.

Table 3: Descriptive statistics of datasets

Variables	Mean			S.D.			Min.			Median			Max.		
	Qt	Qpid	Ger.	Qt	Qpid	Ger.	Qt	Qpid	Ger.	Qt	Qpid	Ger.	Qt	Qpid	Ger.
Description length	87.74	73.12	66.91	757.00	116.13	116.19	0	0	0	51	44	31	62489	2522	2498
ExpR	75.91	80.66	70.25	131.40	116.46	85.55	1	1	1	18	34	41	670	587	403
ExpA	73.74	199.90	101.7	79.93	238.55	101.31	1	1	1	46	88.	70	421	871	444
NoC	1.64	2.95	2.137	2.61	3.542	2.64	0	0	0	1	2	1	60	44	46
NoS	1.62	1.57	1.742	1.13	1.017	0.82	0	0	0	2	2	2	8	10	7
Apit	33.25	125.15	128.99	130.70	150.02	107.33	0	0	0	2	93	108	2267	1300	1357
Resol. time	316.15	88.61	92.07	375.67	193.95	160.51	0	0	0	127	11	20	2290	1965	1357

In all projects, most of the issues are commented on by zero to two stakeholders. In the extreme case, there are up to 10 stakeholders involved in an issue (as in Qpid). The number of comments ranges from 0 to 44 (in Qpid), 46 (in Geronimo), and 60 (in Qt) in three projects. The most commonly observed cases are:

- the issue does not have any comments;
- the issue has one or two comments from the reporter to clarify the description or from the assignee to announce the resolution or discussion;
- the issue has more than two comments which have a ping-pong pattern between reporter and assignee.

There are many stakeholders who participate in both reporting and resolving issues. Many issues are reported and resolved by the same stakeholder.

4.2 Classification Model

4.2.1 Method

Each dataset was divided into two equally sized categories based on the value of issue lead time. We constructed a classification model which classifies a new issue lead time into one of these categories, namely slow and fast. The binary classification is used to make the model results comparable with previous studies [5, 34]. Without loss of generality, we treated fast issues as a positive class. Since both categories are equal in size, the prior probability for each experiment is 0.5, which corresponds to random classification. The decision tree algorithm J48, which was implemented using the WEKA toolkit [8], was used to provide the baseline for classification performance using our datasets. For the validation of the classification model, we used tenfold cross-validation [14]. The results of the cross-validation were then averaged to produce the performance measure. The value represents the percentage of classifications of incoming issues into the fast category which are accurate.

We used accuracy, precision, recall, and the area under the receiver operating characteristic (ROC) curve statistic to evaluate the classification. Precision and recall are defined by the formulae below:

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

where TP is true positive, FP is false positive, and FN is false negative.

The area under the ROC curve (AUC) can be interpreted as the probability that when randomly selecting a positive and a negative example the model will assign a higher score to the positive example [15].

4.2.2 Issue report feature model

The issue report feature model uses only issue description features, which are: priority, version, open month, open year, issue type, and description length. The classification performance of this model is used as a baseline model to evaluate the effectiveness of human factor metrics as complementary variables. The result is shown as Model 1 in Table 4. Among the three projects, the best prediction performance was achieved for project Qt. The model can classify 84.59% of fast issues accurately. The ROC value is 0.870. The classification accuracy of Geronimo models is 59.56%. The Qpid project model performs least accurately, with a prediction accuracy of 58.52%. This result is only slightly better than the results of random classification (50%).

4.2.3 Integration of human factor metrics

We investigated the complementary effects of human factor metrics on improving the classification accuracy by integrating these metrics into the baseline model. Table 4 shows the accuracy of classification models with a different set of input variables and increases in accuracy compared to the baseline model. Model 2 to Model 6 in Table 4 integrate different individual human factor metrics. Model 7 integrates metrics from the stakeholder past performance dimension and Model 8 integrates metrics from the post-submission collaboration dimension. Model 9 integrates all five investigated metrics to assess the comprehensive influence of the metric set on the model performance.

Table 4: Accuracy of classification models

#	Model	Qt	Qpid	Geronimo
1	Issue feature	84.59%	58.52%	59.56 %
2	Issue feature + ExpR	85.53% (+0.94%)	60.18% (+1.66%)	61.77% (+2.21%)
3	Issue feature + ExpA	85.78% (+1.19%)	60.72% (+2.2%)	62.00% (+2.44%)
4	Issue feature + Apit	87.46% (+2.87%)	70.59% (+12.07%)	62.90% (+3.34%)
5	Issue + NoC	86.56% (+1.97%)	59.83% (+1.31%)	72.72% (+13.16%)
6	Issue + NoS	86.77% (+2.18%)	62.20% (+3.68%)	66.13% (+6.57%)
7	Issue feature + ExpR + ExpA + Apit	89.96% (+5.37%)	73.35% (+14.83%)	64.07% (+4.51%)
8	Issue + NoC + NoS	88.14% (+3.55%)	64.74% (+6.22%)	71.69% (+12.13%)
9	All	90.58% (+5.99%)	72.78% (+14.26%)	73.22% (13.66%)

We observe that ExpR and ExpA do not improve the predictive power of the models significantly: in all projects, they increase the accuracy by only 0.94–2.44%. Apit (average past issue lead time) is the best metric among the human factor metrics. It improves the accuracy by 2.87% in Qt, 12.07% in Qpid, and 3.34% in Geronimo. NoC improves the accuracy by 1.97% in Qt, 1.31% in Qpid, and 13.16% in Geronimo. NoS improves the accuracy by 2.18% in Qt, 3.68% in Qpid, and 6.57% in Geronimo.

The stakeholder past-performance based model performs better than the collaboration based model in Qt and Qpid, while the collaboration based model is the best one in Geronimo. The integration of all stakeholder past-performance and collaboration metrics increases the accuracy by 6% in Qt, 14.26% in Qpid, and 13.66% in Geronimo.

4.2.4 Top node analysis

The analysis of the decision tree structure constructed by the J48 classifier [8] provides information about:

(1) which input variable is important in the classification model. The node at the top of the tree is more important as a classifier [14];

(2) special rules that lead to a subset of fast or slow issues. To be able to extract a meaningful and general rule, the model needs to be adjusted with regard to its confidence factor and pruning parameters.

In Geronimo, the topmost node is number of comments. The nodes on the second and third levels are number of stakeholders and average past issue lead time, respectively. In Qt, the topmost node is average past issue lead time and the second level node is

assignee experience. In Qpid, the topmost node is average past issue lead time and the node on the second level is number of stakeholders.

Exploring the decision trees reveals some interesting rules. In Geronimo, all issues that have no comments are quickly resolved. In Qt, all issues that are resolved by an assignee whose average past issue lead time is zero and whose experience has a value greater than two are quickly resolved.

4.2.5 Comparison among prediction techniques

In order to assess the impact of the choice of prediction techniques on the accuracy of the classification models, we compare the classification result using J48 with results obtained using Bayesian Network, Random Forest, and Logistic Regression [32, 33]. A detailed investigation of these techniques is beyond the scope of this paper. In this study, we compare the prediction accuracy and ROC value for each technique on three datasets, as shown in Table 5. The results show that the performance of Random Forest is slightly worse than that of Logistic Regression in Qt but is the best in Qpid and Geronimo. The prediction performance of Naïve Bayes is the worst among prediction techniques for all projects. Overall the choices of classification algorithms increase the accuracy of models slightly, by 2–3%.

Table 5: Accuracy of different classification techniques

Techniques	Project Qt	Project Qpid	Project Geronimo
J48	90.583%	72.78%	73.22%
Bayes Net.	87.401%	67.91%	72.74%
Logistic Reg.	91.422%	69.34%	68.31%
Random Forest	91.060%	75.86%	75.60%

4.3 Univariate Analysis

Univariate analysis was performed for each individual measure (independent variable) against the dependent variable, issue lead time (counted in days), in order to determine whether the measure is a useful predictor of issue lead time. Since we observe a skewed distribution of the independent variables (except for priority value), a nonparametric technique, Spearman's Rho, was selected to measure the monotonic correlation between the investigated variables. The correlation results are shown in Table 6. Two asterisks denote a significance of 0.01 and a single asterisk denotes a significance of 0.05. We adopted Hopkins' correlation coefficient classification in which values of 0.3–0.5 are classified as medium, 0.5–0.7 as large, 0.7–0.9 as very large, and 0.9–1.0 as almost perfect [6]. Coefficient values larger than 0.3 are shown in bold typeface.

Table 6: Spearman correlation with issue lead time

Variables	Project Qt	Project Qpid	Project Geronimo
Description length	–0.123**	0.065**	0.118**
Priority value	–0.157**	0.021	–0.021
ExpR	**0.372****	0.222**	–0.113**
ExpA	–0.186**	–0.021	–0.168**
NoC	0.008*	0.243**	**0.416****
NoS	0.123**	**0.309****	**0.303****
Apit	**0.799****	0.284**	0.222**

Table 6 shows that the correlation coefficients of NoS and Apit are significantly positive in all projects. Especially in Qt, Apit shows a very high correlation with issue lead time. Therefore, these two variables are potential predictors of issue lead time and should be analysed in the multivariate models.

The coefficients of description length, ExpR, and NoC are also statistically significant in all projects. But the directions of the effects of these variables are inconsistent among the projects. The coefficient of ExpR is 0.372 and 0.222 in Qt and Qpid, respectively, while it is –0.113 in Geronimo. The coefficient of NoC is 0.243 and 0.416 in Qpid and Geronimo, respectively, while it is 0.008 in Qt. ExpA is statistically correlated with issue lead time in Qt and Geronimo. Besides, the coefficients are all negative, which suggests an inverse relationship with issue lead time. Referring to Hopkins' interpretation [6], the coefficients of description length, priority value, and ExpA are small. Therefore, these variables are probably not useful for predicting issue lead time.

A pairwise correlation among independent variables is performed to discover the potential correlated independent variables. The observation that many project members are active in both reporting and resolving issues raises the question of correlation between ExpR and ExpA. However, the test shows insignificant correlation between these variables. The Spearman correlation between NoC and NoS is statistically significant and has a value of 0.413 in Qt, 0.569 in Qpid, and 0.491 in Geronimo. This significant correlation is expected because the larger number of stakeholders involved in an issue (i.e. editing the reports or commenting on the issue) clearly leads to an increase in the number of comments or report edits. Since the correlation is medium to large, the relationship between these variables should be further analysed when integrating them in multivariate models.

4.4 Multivariate Linear Regression Models

4.4.1 Methods

While univariate analysis shows which independent variables are potential indicators of the dependent variable, multivariate regression analysis determines how well we can predict the issue lead time when the measures are used in combination [7]. Multivariate linear regression models attempt to model the relationship between independent variables and a dependent variable by fitting a linear equation to observed data:

$$y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + ... + \beta_n x_n$$

where y is the dependent variable and β_i is the regression coefficient of the independent variable x_i.

In the analysis performed here, we assume that the relationships between independent variables and the dependent variable can be modelled by a linear formula. Firstly, we construct a model with all input variables to observe the best R^2 value. Then we use stepwise selection to choose the subset of variables which contribute significantly to explaining the dependent variable [34]. All models are implemented using the R statistical package [9].

Variable coefficients: The regression coefficient indicates the relative effect of an increase in the independent variable on the value of the dependent variable [7]. In addition, each independent variable has an associated t-value that indicates its statistical significance. The significance of a coefficient and the direction of the coefficient value indicate the influence of the coefficient in combination with other variables.

Model coefficient: To evaluate the goodness-of-fit of the model, we use the R^2 statistic. This is the proportion of the variance in the dependent variable that is explained by the variance in the independent variables [7]. The *adjusted R^2 measure* is also used to explain any bias in the R^2 measure by taking into account the degrees of freedom of the independent variables and the sample population.

A linear regression model should be tested for influential observations and multicollinearity [7]. To identify influential observations, we calculate Cook's distance for each data point [21]. An observation with a Cook's distance greater than 4/n (where n is the number of observations) is regarded as an influential observation and is thus particularly worth checking [21]. Multicollinearity refers to the degree to which any variable's effect can be predicted by the other variables in the analysis. As multicollinearity rises, the ability to define any variable's effect is diminished [12]. Since the univariate analysis reported moderate correlation between NoC and NoS, multicollinearity analysis is necessary to explore whether only one of these variables should be used. To investigate this potential problem, we compute the variance inflation factors (VIFs), which are widely used to measure the degree of multicollinearity between variables in regression models. Kutner et al. [5] suggest that in OLS regression a VIF value of a variable greater than 10 may indicate possible multicollinearity problems and should be investigated further.

4.4.2 Model building and validation

The VIFs in the three projects are shown in Table 7. The results show that VIF values of all variables are well below the critical threshold of 10, which indicates the absence of multicollinearity.

Table 7: Multicollinearity analysis

Variables	Project Qt	Project Qpid	Project Geronimo
Description length	1.001	1.019	1.027
Priority	1.077	1.033	1.045
ExpR	3.136	1.003	1.543
ExpA	2.317	1.019	1.482
NoC	1.048	1.497	1.464
NoS	1.113	1.545	1.484
Apit	1.118	1.051	1.309

The multivariate linear regression model is shown in Table 8, Table 9, and Table 10. For each model variable, we report the variable coefficients, standard error (S.E.), and significance level. The model statistics are reported as R^2, adjusted R^2, and p value.

In the Qpid model, the value of R^2 is 0.2922. The model has 14 influential observations, which were excluded from the final model fitting. However, to retain the objectivity of the results of the analysis, influential outliers will be kept during the evaluation of this model. In the model, ExpR, ExpA, NoS, and Apit are statistically significant. The coefficient of ExpA is negative, which represents an inverse relation with issue lead time.

Table 8: Linear regression model for Qpid

Variables	Coeffic.	S.E.	t value
Intercept	–17.859	16.283	-1.097
Description length	0.004	0.027	0.164
Priority	–7.549	4.833	–1.562
ExpR	0.110	0.036	3.082**
ExpA	–0.051	0.030	-1.671*
NoC	1.617	1.425	1.135
NoS	43.038	4.606	9.344**
Apit	0.386	0.029	13.322**
Model	R^2 = 0.2922 Adjusted R^2 = 0.2809 P value = 0.000		

In the Geronimo model, the value of R^2 is 0.3226. The model has four influential observations, which were excluded from the final model fitting. In the model, Priority, ExpR, NoC, NoS, and Apit are statistically significant. The coefficient of priority is negative, indicating an inverse relation with issue lead time.

Table 9: Linear regression model for Geronimo

Variables	Coeffic.	S.E.	t value
Intercept	–6.478	11.180	–0.579**
Description length	0.003	0.021	0.151
Priority	–10.740	3.569	–3.008**
ExpR	0.045	0.026	1.762*
ExpA	–0.010	0.012	–0.832
NoC	2.710	0.824	3.290**
NoS	11.38	11.950	9.523**
Apit	0.588	0.016	35.946**
Model	R^2 = 0.3226 Adjusted R^2 = 0.3196 P value = 0.000		

In the Qt model, the value of R^2 is 0.5954. The model has 11 influential observations, which were excluded from the final model fitting. In the model, priority, ExpR, ExpA, NoS, and Apit are statistically significant. The coefficients of priority, ExpR, and ExpA are negative, indicating inverse relations with issue lead time.

Table 10: Linear regression model for Qt

Variables	Coeffic.	S.E.	t value
Intercept	–47.130	7.459	–6.318**
Description length	–0.001	0.004	–0.212
Priority	–53.090	2.589	–20.51**
ExpR	–0.892	0.039	–23.072**
ExpA	–1.432	0.055	–26.212**
NoC	1.607	1.123	1.355
NoS	20.500	20.490	9.786**
Apit	0.837	0.023	36.071**
Model	R^2 = 0.5954 Adjusted R^2 = 0.595 P value = 0.000		

The regression analysis shows how well the models fit the sample data. Using stepwise regression we remove the variables which were insignificant in the previous models and validate the new model using ten-fold cross-validation. The Spearman correlation coefficients between actual and predicted issue lead time are 0.135 in Qt, 0.535 in Geronimo, and 0.770 in Qpid.

5. DISCUSSION

5.1 Answers to Research Questions

*5.1.1 **RQ1**. Do human factor metrics improve classification of issue lead time?*

The top node analysis and integration of individual human factor metrics show that there is no single metric which outperforms the others in all projects. In Qt and Qpid, average past issue lead time is the best complementary factor. In Geronimo, number of comments is the best. This result can be explained by the moderate to high correlation between these variables and issue lead time in different projects. We also observe that ExpR and ExpA do not significantly improve the predictive power of the models since the increase in accuracy is only 0.94–2.44%. Consequently, among stakeholder past performance metrics, average past issue lead time is the most useful metric. The accuracy improvement introduced by single metrics suggests that average past issue lead time, number of comments, and number of

stakeholders are potentially useful variables in issue lead time models.

The integration of all human factor metrics and selected prediction techniques provide classification accuracy of 75.86–91%. This result is compared with results obtained by previous studies as shown in Table 11. The achieved precision is higher in our study than in other studies. The improvement in our classification model reflects the fact that rather than using characteristics of the data distribution of the dependent variable, we also use the stakeholder past-performance metrics.

Table 11: Comparison with prior classification models

Study	Dataset	Technique	Precision	Recall
Giger et al. [5]	Firefox	Decision tree	0.680	0.683
	Streamer	Decision tree	0.613	0.652
	Evolution	Decision tree	0.665	0.760
Hooimeijer [34]	Firefox	Linear reg.	0.7	0.825
This study	Qt	Logistic reg.	0.919	0.914
	Qpid	Random forest	0.759	0.759
	Geronimo	Random forest	0.761	0.757

*5.1.2 **RQ2**. Which characteristics of issues increase the predictive power of a linear regression model for predicting issue lead time?*

The multivariate analysis shows that reporter experience, number of stakeholders, and average past issue lead time are useful for building linear regression models in all three OSS projects. The goodness-of-fit statistics' adjusted R^2 values are 0.292, 0.323, and 0.595 in Qt, Geronimo, and Qpid, respectively. This result is compared with results from prior studies as shown in Table 12. The result of low R^2 values is similar to findings by Bhattacharya et al. [25]. There could be several reasons for this result. First, the metrics collected from the issue tracking system are useful but insufficient to explain a significant amount of variance in issue lead time. Therefore, the metrics collected from other data sources need to be explored in order to improve the prediction accuracy of the model. Second, the skewness of data distributions might affect the effectiveness of the linear regression model [29]. Therefore, data transformation, for example by logarithmic or log transformation, could help to increase the model performance. Last but not least, the relationship between issue resolution and some metrics might be better modelled by a nonlinear formula. Therefore, the use of some nonlinear models such as the MARS model [28] could be explored.

Table 12: Comparison with previous linear regression models

Study	Dependent var.	Dataset	R^2
Bhattacharya et al. [25]	Bug fixing time	Firefox	0.401
		Thunderbird	0.498
		Seamonkey	0.366
		Eclipse	0.301
Anbalagan et al. [20]		Ubuntu 5.10	0.98
		Ubuntu 6.04	0.81
This study	Issue lead time	Qpid	0.292
		Qt	0.595
		Geronimo	0.326

*5.1.3 **RQ3**. Which human factor metrics contribute significantly to the issue lead time prediction models?*

Description length: Univariate analysis shows that the correlation coefficient is trivial in all projects. Besides, the variable is not statistically significant in all multivariate models. Therefore the description length is unlikely to have a relationship with issue lead time.

Issue priority: In the univariate analysis, the correlation coefficients are not statistically significant. When integrating the priority value in multivariate models, the variable's coefficients are significant in two out of three projects. It is worth noticing that the coefficient values are all negative, which shows that priority has an inverse relationship with issue lead time. Alternatively, the issues with higher priority will be resolved faster. This result is consistent with intuitive reasoning and confirms the findings of earlier studies [16].

Assignee experience: Univariate and multivariate analyses show that this variable is significant in two out of three projects. In particular, the coefficients' values are negative in all cases. This result indicates an inverse relation between assignee experience and issue lead time. That is, the developer tends to resolve issues faster when he or she has resolved many issues before. This result is in agreement with the findings of other papers [16, 17].

Reporter experience: This variable is found to be significantly related to issue lead time in all univariate and multivariate analyses. However, the direction of the relationship is inconsistent among analyses and projects. Therefore this variable is useful in constructing the prediction models but its relationship with issue lead time needs further investigation.

Number of stakeholders: This variable is statistically significant in all analyses. The moderate correlation coefficient indicates that number of stakeholders is a good indicator of current issue lead time. Therefore, this result confirms the findings of other studies [16, 18, 19]. The positive coefficients indicate that the larger the number of stakeholders involved in an issue, the more slowly it will be resolved.

Number of comments: The univariate analysis shows a statistically significant correlation between number of comments and issue lead time in all projects. The positive coefficients indicate that the larger the number of comments on an issue, the more slowly it will be resolved. As observed from the content of issue reports and comments (Section 4.1), there are many issues with poor descriptions or issues that relate to other issues or software modules. Explanation and discussion of such issues require effort, which increases the time taken to resolve them. It is also noticed that when combined with other variables, the variable does not contribute significantly to explaining variance in issue lead time.

Table 13: Summary of variable influence

Project	Qpid		Qt		Geronimo	
Analysis	***Multi***	***Uni***	***Multi***	***Uni***	***Multi***	***Uni***
Desc. length	O	++	O	--	O	++
Priority	O	O	--	--	--	O
Reporter exp.	++	++	--	++	+	--
Assignee exp.	-	O	--	--	O	--
Number of comments	O	++	O	+	++	++
Number of stakeholders	++	++	++	++	++	++
Average past resolution time	++	++	++	++	++	++

Average past lead time: This variable is statistically significant in all analyses. The moderate to high correlation coefficient shows that average past lead time of an assignee could be used to predict the lead time of his or her current issue. The relationship between average past lead time and current issue lead time, together with the inverse relationship between assignee experience and issue lead time, might be evidence of a learning process. In Qt, the high correlation between average past lead time and lead time of the current issue shows a monotonic trend in issue lead time for an assignee; the assignee experience shows that this trend is monotonically decreasing. This means that for a specific developer the average time taken to resolve an issue is shortened when he or she gains experience in resolving issues. However, this trend varies between developers and projects, as indicated by the variety of correlation coefficient values among different projects.

The summary of results is given in Table 13. The regression or correlation coefficients which are positive or negative and significant at 0.01 are marked as "++" or "--", respectively. The regression or correlation coefficients which are positive/negative and significant at 0.05 are marked as "+" or "-". Insignificant coefficients are marked as "O".

6. THREATS TO VALIDITY

In this paper, we have reported our results based on three medium-size OSS projects of similar domains. We are aware that there are several threats to the validity of our study. However we consider that the results obtained provide useful insights for envisaging further studies to improve issue lead time prediction models.

First, we are aware that although we could suggest successful predictors of issue lead time from the issue history in each of the studied OSS projects, these results might not be generalizable to all projects.

Second, we cannot claim that our analysis of the relationship between human factor metrics and issue lead time indicates causal effects, as we are investigating correlations rather than conducting impact studies. For example, even though we observed a relationship between the number of stakeholders and issue lead time, the larger number of involved stakeholders could be caused by the size and complexity of the issue, which would also cause the resolution of the issue to take more time. Besides, in the multivariate model, we assume that the relationship among variables takes a linear form; however, the actual relationship can be better represented by another mathematical formula.

Third, we are aware of the quality of data contained in OSS repositories since the data can be randomly filled in and duplicated reports can occur. In order to deal with this threat, we implemented a filtering process to remove data that did not fulfill a certain level of quality (see Section 3.2)

Fourth, we used a limited set of metrics to test the model, and therefore it could be argued that the selection of metrics might not be complete: our measurement might capture only the symptomatic effects of other variables. Besides, there might be a correlation among independent variables. To deal with this, we performed an analysis of multicollinearity using variance inflation factors.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have investigated the effectiveness of some bug report based metrics in predicting issue lead time in classification and linear regression models. We also tested the effect of some metrics related to human factors on the effectiveness of these models. We draw some conclusions based on the analysis of data from three OSS projects. For classification models:

1. Number of comments, number of stakeholders, and average past issue lead time are effective complementary variables in classifying issue lead time.
2. The Random Forest technique shows slightly better performance than the other techniques. Overall, there is no significant difference in the performance of different techniques.

For linear regression models:

1. Description length cannot be used to predict issue lead time.
2. Priority has a negative relationship with issue lead time: issues with higher priority are resolved faster.
3. Assignee experience has a negative relationship with issue lead time: issues resolved by more experienced assignees are resolved faster.
4. Number of comments, number of stakeholders, and average past issue lead time have a positive relationship with issue lead time.
5. Number of stakeholders and average past issue lead time are good predictors of issue lead time.
6. Issue-report based prediction models yield far from desirable predictive power.

The results show that prediction of issue lead time is still far from straightforward. The following future work is planned:

- More variables collected from other data sources, such as mailing lists and version control systems, should be investigated to gain a more comprehensive view of factors which have an influence on issue lead time.
- Skewness of datasets may affect the results of the linear regression model. In future, data transformation methods such as log transformation will be investigated in an effort to improve model performance.
- To confirm the effectiveness of some metrics used and proposed in this study, we aim to test them on other sets of OSS projects. Besides, we plan to replicate the study on closed source projects to make comparisons.

8. REFERENCES

[1] Panjer, L. 2007. Predicting Eclipse bug lifetimes. In *ICSE Workshops on Mining Software Repositories (MSR)* (Minneapolis, USA, May 2007). 29–35.

[2] Lamkanfi, A., Demeyer, S., Giger, E., and Goethals, B. 2010. Predicting the severity of a reported bug. In *7th IEEE Workshop on Mining Software Repositories (MSR)* (Hawaii, USA, May 2010). 1–10.

[3] Giger, E. Pinzger, M., and Gall, H. 2010. Predicting the fix time of bugs. In *2nd International Workshop on Recommendation Systems for Software Engineering* (Cape Town, South Africa, 2010). 52–56.

[4] Cataldo, M., Herbsleb, J.D., and Carley, K.M. 2008. Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity. In *2nd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Kaiserslautern, Germany, 2008). 2–11.

[5] Bettenburg, N., Just, S., Schroter, A., Weiss, C., Premraj, R., and Zimmermann, T. 2008. What makes a good bug report?

In *16th ACM SIGSOFT International Symposium on Foundations of Software Engineering* (Atlanta, Georgia, USA, 2008). 308–318.

[6] Hopkins, W.G. 2002. A scale of magnitudes for the effect statistics. *New View Stat.* (June 2002).

[7] Devore, J. L. *Probability and Statistics for Engineering and the Sciences,* 6th edition. *Technometrics*, 46, 4, 497.

[8] *Weka 3 – Data Mining with Open Source Machine Learning Software in Java* [Available online] – http://www.cs.waikato.ac.nz/ml/weka/

[9] *The R Project for Statistical Computing* [Available online] – http://www.r-project.org/

[10] *JIRA – Bug, Issue and Project Tracking for Software Development* [Available online] – http://www.atlassian.com/software/jira/

[11] Qt project [Available online] – http://qt.nokia.com/

[12] Qpid project [Available online] – http://qpid.apache.org/

[13] Geronimo project [Available online] – http://geronimo.apache.org/

[14] Liu, B. 2007. Supervised learning. In *Web Data Mining*. Springer, Berlin, Heidelberg, 55–116.

[15] Bougie, G., Treude, C., German, D., and Storey, M. 2010. A comparative exploration of FreeBSD bug lifetimes. In *7th IEEE Workshops on Mining Software Repositories (MSR)* (Hawaii, USA, May 2010). 106–109.

[16] Guo, P. J., Zimmermann, T., Nagappan, N., and Murphy, B. 2010. Characterizing and predicting which bugs get fixed: an empirical study of Microsoft Windows. In *32nd ACM/IEEE International Conference on Software Engineering (ICSE)* (Cape Town, South Africa, 2010). Vol. 1, 495–504.

[17] Hooimeijer, P. and Weimer, W. 2007. Modeling bug report quality. In *22nd IEEE/ACM International Conference on Automated Software Engineering* (Atlanta, Georgia, USA, 2007). 34–43.

[18] Pinzger, M., Nagappan, N., and Murphy, B. 2008. Can developer-module networks predict failures? In *16th ACM SIGSOFT International Symposium on Foundations of software engineering (FSE)* (Atlanta, Georgia, USA, 2008). 2–12.

[19] Meneely, A., Williams, L., Snipes, W., and Osborne, J. 2008. Predicting failures with developer networks and social network analysis. In *16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE)* (Atlanta, Georgia, USA, 2008). 13–23.

[20] Anbalagan, P. and Vouk, M. 2009. On predicting the time taken to correct bug reports in open source projects. In *IEEE International Conference on Software Maintenance (ICSM)* (Alberta, Canada, Sept. 2009). 523–526.

[21] Belsley, D., Kuhm, E., and Welsch, R. 1980. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity.* John Wiley and Sons, New York.

[22] Hanna, M. 1993. Maintenance burden begging for a remedy. *Datamation* (Apr. 1993) 53–63.

[23] Pfleeger, L. 1998. *Software Engineering: Theory and Practice*. Prentice Hall.

[24] Abreu, R. and Premraj, R. 2009. How developer communication frequency relates to bug introducing changes. In *Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops* (Amsterdam, The Netherlands, 2009). 153–158.

[25] Bhattacharya, P. and Neamtiu, I. 2011. Bug-fix time prediction models: Can we do better? In *8th IEEE Working Conference on Mining Software Repositories (MSR)* (Hawaii, USA, May 2011).

[26] Herbsleb, J. D. and Mockus, A. 2003. Formulation and preliminary test of an empirical theory of coordination in software engineering, SIGSOFT Softw. Eng. Notes, 28, 5. (Sept. 2003) 138–137.

[27] Ostrand, T. J., Weyuker, E. J., and Bell, R. M. Programmer-based fault prediction. In *6th International Conference on Predictive Models in Software Engineering* (Timisoara, Romania, Sept. 2010). 1–10.

[28] Briand, L., Melo, W., and Wust, J. 2002. Assessing the applicability of fault-proneness models across object-oriented software projects. *IEEE Trans. Softw. Eng.* 28 (2002) 706–720

[29] Monden, A. and Kobayashi, K. The effect of log transformation in multivariate liner regression models for software effort prediction. *Comp. Softw*. 27 (2010).

[30] Stanton, N., Salmon, P., Walker, G., Baber, C., and Jenkins, D. 2005. *Human Factors Methods; A Practical Guide For Engineering and Design.* Ashgate Publishing, Aldershot, Hampshire.

[31] Ahsan, S. N., Ferzund, J., and Wotawa, F. 2009. Program file bug fix effort estimation using machine learning methods for OSS. In *21st International Conference on Software Engineering and Knowledge Engineering (SEKE)* (Boston, USA, July 2009).

[32] Lessmann, S., Baesens, B., Mues, C., and Pietsch, S. 2008. Benchmarking classification models for software defect prediction: a proposed framework and novel findings. *IEEE Trans. Softw. Eng.* 34 (2008) 485–496.

[33] Challagulla, V., Bastani, F., Yen, I.-L., and Paul, R. 2005. Empirical assessment of machine learning based software defect prediction techniques. In *10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems* (Arizona, USA, 2005) 263–270.

[34] Tripepi, G., Jager, K. J., Dekker, F. W., and Zoccali, C. 2008. Linear and logistic regression analysis. *Kidney Int*. 73 (Jan. 2008) 806–810.