An Empirical Evaluation of the Tobit Model on Software Defect Prediction

Yukasa Murakami, Masateru Tsunoda
Department of Informatics
Kindai University
Higashiosaka, Japan
m.yukasa@gmail.com, tsunoda@info.kindai.ac.jp

Koji Toda
Department of Information and Systems Engineering
Fukuoka Institute of Technology
Fukuoka, Japan
toda@fit.ac.jp

***Abstract*—In project management, project plan is made based on the prediction results of the project. Predicting the number of defects is one of important prediction. To enhance the prediction accuracy of the number of defects, many studies proposed various prediction models. The model is built using a dataset collected in past projects, and the number of defects is predicted using the model and the data of the current project. Datasets sometimes have many data points where the dependent variable, i.e., the number of defects is zero. When a multiple linear regression model is made using the dataset, the model may not be built properly. To build proper model, we use the Tobit model as software defect prediction. The model assumes that the range of a dependent variable is limited, e.g., the minimum value of the variable is zero, and the model is built based on the assumption. In the experiment, we applied the regression model based on ordinary least squares and the Tobit model to fault prediction. Also, we evaluated models applied log-transformation. In the experiment, the Tobit model applied log-transformation was the highest accuracy in the models. Median *BRE* of the model was 14% improvement, and Pred25 was 7% improvement, compared with other models.**

Keywords—Fault prediction; censored data; log-transformed; linear regression

I. Introduction

In a large scale software development project, management is important, to avoid failure of it. In project management, planning is based on the prediction of the project. For example, the software testing plan is made based on software defect prediction. Therefore, the accuracy of the prediction is important. To enhance the prediction accuracy of the number of defects, various models are proposed [4][6]. The model is built based on a dataset of past projects, and it is predicted using the model and the data of the current project. For instance, the programming language and software size of a current project are input to the model, and using them, the number of defects is predicted. To make the prediction model, multiple linear regression model is widely used.

However, some datasets have many data points where the number of defects is zero and a major independent variable such as software size is not zero. In this case, when a multiple linear regression model is built based on ordinary least squares, it may not be built appropriately. For example, a model is built using the dataset in which the number of defects is zero when software size is smaller than 100 FP (Function Point). Then, the model may predict that the number of defects is smaller than zero, when software size of a current project is much smaller than 100 FP.

To solve the problem, we apply the Tobit model [21] to software defect prediction and evaluate the prediction performance of it. It is widely used in other fields such as quantitative sociology. It assumes that the range of a dependent variable is limited, e.g., the minimum value of the variable is zero, and the model is built based on the assumption. Although the built model is similar to the linear regression model, the Tobit model uses another model when the dependent variable is smaller than the lower bound of it. The assumption seems well fit to software defect prediction. So, applying the Tobit model is expected to enhance prediction accuracy of software defect prediction.

In the experiment, we used a dataset collected from actual software development companies, and compared the prediction accuracy of the Tobit model with the ordinary linear regression. They predicted the number of defects found after the release of software. When log-transformation is applied, the ordinary regression can describe non-linear relationships. So, in addition to the above model, we applied log-transformation to the dataset, and built other models. Although the Tobit model was not very new method, the performance of the model has not been evaluated on software defect prediction, as long as we know.

This is an extended study of our past study presented in a domestic symposium [22]. Compared with the past study, the major new contribution of this study is the comparison with an existing method. Our experiments consist of three parts. That is, 1) the application of the Tobit model, 2) the combination of the Tobit model and the regression model, and 3) the comparison with an existing method (i.e., the Poisson regression model). In the past study, the experiment 3) is not included. The Poisson regression model seems similar to the Tobit model. So, without the comparison, readers cannot judge the novelty and effectiveness of the Tobit model, and hence the experiment 3) is quite important and indispensable to enhance the reliability of the research. Additionally, to clarify the purpose of the experiment, we set research questions in this study. They are very important to explain why we highly evaluated the Tobit model.

II. Prediction Model

A. *Multiple Linear Regression Model*

The multiple linear regression model is widely used when predicting the number of software defects mathematically. The model is built based on ordinary least squares. When the number of defects is denoted as *y*, and independent variables such as software size are denoted as x_1, x_2, … , x_k (*k* is the number of independent variables), *y* is explained as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \varepsilon \tag{1}$$

In the equation, β_0 is an intercept, β_1, β_2, … , β_k are partial regression coefficients, and ε is an error term. As a rule of thumb, to build a proper model using linear regression analysis, it is needed that the number of data points is five to ten times larger than the number of independent variables.

When building a regression model which predicts software development effort or the number of defects, log-transformation is sometimes applied, to enhance the accuracy of the model [8]. This is because the distributions of some variables are log-normal distribution, and predicted values are larger than zero, when log-transformation is applied to the dependent variable (i.e., the predicted value of development effort and the number of defects is larger than zero).

B. *Tobit Model*

The Tobit model [21] focuses the bias of the distribution of the dependent variable, when building a model. The Tobit model classifies the bias into the followings [15].

- Censored
- Truncated
- Incidental truncation

Censored means the range of the dependent variable is limited (e.g., the minimum value of the variable is zero). For instance, the number of software defects is censored data. Truncated means some data points (e.g., data points whose number of defects are zero) are excluded from the dataset for some reason. Incidental truncation means the values of some data points are zero, although the original values are not zero. For example, the number of found defects in the code review is zero when the review is skipped. When data is regarded as censored or truncated, the type I Tobit model is applied, and when it is regarded as incidental truncation, the type II Tobit model is applied. To predict the number of defects, the type I Tobit model is applied.

Type I Tobit model includes the censored regression model and the truncated regression model. The censored regression model is denoted as:

$$y^* = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \varepsilon \tag{2}$$

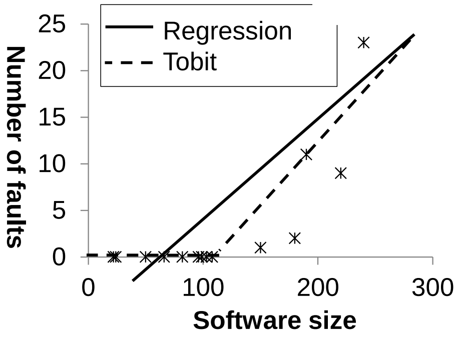

Fig. 1. The Tobit model and the regression model

$$y = \left\{ \begin{array}{l} y^*, y^* > 0 \\ 0, y^* \le 0 \end{array} \right\} \tag{3}$$

In the equation, β_0 is an intercept, β_1, β_2, … , β_k are partial regression coefficients, and ε is an error term. We use the censored regression model to predict software defects. Although the Tobit model was not very new method, the performance has not been evaluated on software defect prediction, as long as we know.

Fig. 1 illustrates the Tobit model and the linear regression model based on ordinary least squares. In the figure, the independent variable (x-axis) is software size, and the dependent variable (y-axis) is the number of software defects. Points in the graph are data points of projects. As illustrated in the figure, the Tobit model properly treats data points whose number of defects are zero.

III. Experiment

A. *Overview*

To evaluate performance of the Tobit model for software defect prediction, we made four prediction models, and compared the prediction accuracy of them.

- OLS method (ordinary least squares regression)
- Tobit method (Tobit model)
- Log-OLS method (OLS method with log-transformation)
- Log-Tobit method (Tobit method with log-transformation)

On the Log-OLS method and Log-Tobit method, log-transformation was applied to ratio scale variables, since it is expected to improve prediction accuracy, as explained in section II.A. However, the number of defects includes zero, and log-transformation cannot be applied to it. So, we added one to it before the log-transformed. That is often applied when building a regression model.

To clarify the purpose of the experiment, we set two research questions as follows:

TABLE I. VARIABLES USED IN THE EXPERIMENT

Variable	Scale	Detail
Number of defects	ratio	Defects found after software release within one month
FP	ratio	Raw function points
Development type	nominal	Enhancement, and new development
Business area	nominal	Banking, financial, insurance, manufacturing, and others
Platform	nominal	Mainframe, and midrange

- **RQ1**: Should we consider to use the Tobit model when building software defect prediction model?
- **RQ2**: When log-transformed is applied, should we consider to use the Tobit model when building software defect prediction model?

If the accuracy of the OLS model and the Tobit model is almost the same, we think that the Tobit model should not be discarded. Since the Tobit model is more suitable when the dependent variable is censored, and the built model may be more proper than the OLS model, to clarify the causes of the defects (The causes are clarified by referring the partial coefficients of the model).

So, we set the answer of RQ1 "yes," when the accuracy of the OLS method and the Tobit method is almost same. Similarly, we set the answer of RQ2 "yes," when the accuracy of the Log-OLS method and the Log-Tobit method is almost same. Note that even when the answers of the questions are "yes," we do not assert the Tobit model should be used, but suggest that the Tobit model is better to use as one of the candidates of the prediction model.

B. Dataset

To build the prediction models, we used the dataset provided by ISBSG (International Software Benchmarking Standards Group) [3]. Included projects were collected from software development companies in 20 countries. It is widely used to evaluate prediction models [13]. Version of the dataset is Release 9, and it includes the projects which were carried out between 1989 and 2004.

The dataset has 3026 projects and 99 variables, and there are many missing values. To uniform the data points used to build models, we selected projects whose data quality is A or B, and FP measurement method is IFPUG (International Function Point Users Group) method. The selection criteria is often applied by many studies [12]. Additionally, we eliminated data points which have missing values (i.e., listwise deletion was applied). As a result, 221 data points were selected, and the number of data points whose number of defects was zero was 86. Fig. 2 shows the distributions of the number of defects.

Fig. 3 shows the relationship between FP and the number of defects. The Spearman rank correlation between them was 0.38, and when we removed the data points whose defect was zero, the correlation was 0.43. This result suggests that it is not easy to predict the number of faults based on FP.

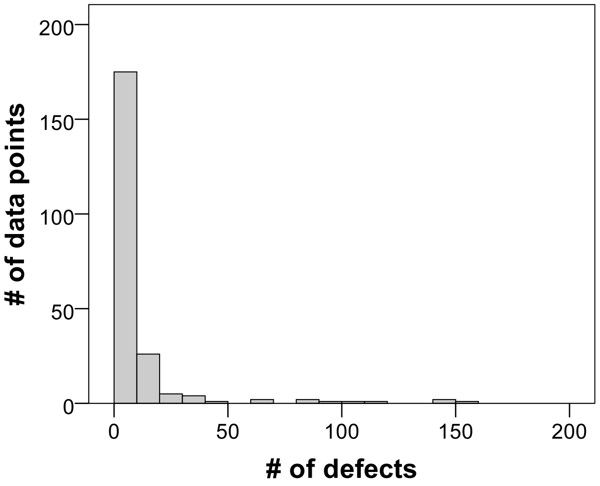

Fig. 2. The distribution of the number of the defects

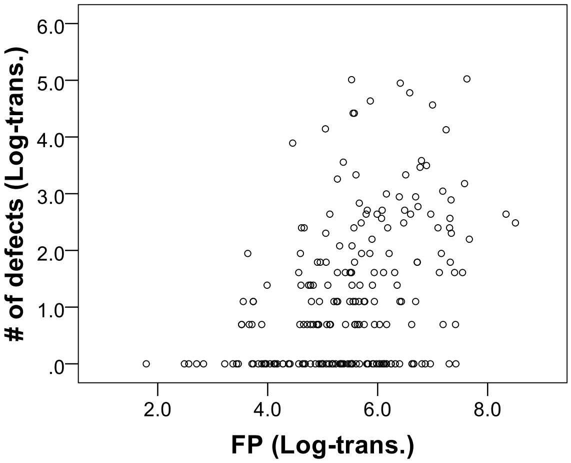

Fig. 3. The relationship between FP and the number of defects

The dependent variable is the number of defects, and candidates of independent variables are shown in Table I (except for the number of defects). We use variables such as platform and business sector, since we do not predict fault-prone module but predict fault-prone project. In the dataset, code metrics are not recorded. However, business sector relates to the quality of the software to some extent. For example, if the business sector is banking, the quality of the software will be high. So, variables such as business sector are useful to predict faults.

Nominal scale variables were transformed into dummy variables (The value of the variable is 0 or 1). As a preliminary analysis, we performed variable selection using AIC (Akaike's Information Criterion) on the regression model based on ordinary least squares. Based on the result, we selected FP, new development (development type), banking (business sector), mainframe (platform), and midrange (platform) as the independent variables.

TABLE II. PREDICTION ACCURACY OF THE MODELS

	Average *AE*	**Median *AE***	**Average *BRE***	**Median *BRE***	**Pred25**
OLS method	10.74	4.11	2.84	1.65	0.12
Tobit method	9.66	2.04	3.78	1.15	0.08
Log-OLS method	8.24	1.86	2.26	1.07	0.21
Log-Tobit method	8.45	1.53	2.6	0.93	0.28
Log-Merge method	8.19	1.85	2.32	1.1	0.33

C. *Evaluation Criteria*

To evaluate prediction accuracy of the models, we used Pred25 [2], and average and median of *AE* (Absolute Error), and *BRE* (Balanced Relative Error) [14]. Note that in this paper, Pred25 indicates the ratio of data points whoso *BRE* is smaller than 25%. When *AE* and *BRE* is low and Pred25 is high, prediction accuracy is regarded as high.

When x denotes actual effort, and $\hat{x}$ denotes estimated effort, each criterion is calculated by the following equations:

$$AE = |x - \hat{x}| \tag{4}$$

$$BRE = \begin{cases} \dfrac{|x - \hat{x}|}{x}, x - \hat{x} \geq 0 \\ \dfrac{|x - \hat{x}|}{\hat{x}}, x - \hat{x} < 0 \end{cases} \tag{5}$$

MRE (Magnitude of Relative Error) [2] is widely used to evaluate a prediction model, and *MER* (Magnitude of Error Relative to the estimate) [10] is sometimes used. However, *MRE* and *MER* are imbalanced for underestimation and overestimation [1][11]. The maximum *MRE* is 1 even if an extreme underestimate occurs (For instance, when the actual effort is 1000 person-hour, and the estimated effort is 0 person-hour, *MRE* is 1). Similarly, maximum *MER* is smaller than 1 when an overestimate occurs. So, instead of *MRE*, we adopted *BRE* whose evaluation is not biased [16].

To build the models and calculate the evaluation criteria, we applied 5-fold cross validation.

IV. RESULTS AND DISCUSSION

A. *Models without log-transformation*

Table II shows the prediction accuracy of the models without log-transformation. On the Tobit method, three criteria showed higher accuracy than the OLS method. Especially, median *BRE* showed about 50% improvement from the OLS method. On the contrary, average *BRE* and Pred25 got worse. The difference was about 100% and 8%. From the result, it is not easy to conclude that which model is appropriate for the defect prediction, if log-transformation is not applied. So, the answer of RQ1 is “No.”

B. *Models with log-transformation*

The prediction accuracy of the models with log-transformation are shown in Table II. Compared with the models without log-transformation, the accuracy of them were improved very much. On the Log-Tobit method, three criteria showed higher accuracy than the Log-OLS method. Although average *AE* was larger than the Log-OLS method, the difference was very small. Average *BRE* was 34% larger than the Log-OLS method, but the difference was smaller than the models without log-transformation.

Median *BRE* was 14% improvement, and Pred25 was 7% improvement from the Log-OLS method. The difference is not ignorable, and hence the result suggests log-transformed is also effective to the Tobit method, and applying the Log-Tobit method should be considered when the defect prediction model is built. That is, the answer of RQ2 is “Yes.”

C. *Discussion*

We focused on data points whose actual value of the number of defects was zero and the predicted value was smaller than one (i.e., it was almost zero). Recall [20] can be calculated, when the number of the data points is used as the numerator, and the number of data points whose actual values are zero is used as the denominator. In the same way, precision and F-measure [20] can be calculated.

Recall, precision and F-measure of the Log-OLS method and the Log-Tobit method are shown in Table III. F-measure of the Log-Tobit method was higher than the Log-OLS method, i.e., that suggests the Log-Tobit method was higher accuracy. However, the Log-Tobit method may overfit to data points whose the actual value was zero, since precision was lower than the Log-OLS method. If predicted values of the Log-Tobit method are not used when the values are not zero, the prediction accuracy may be improved. So, we combined prediction results of the Log-OLS method and the Log-Tobit method, and analyzed the prediction accuracy of it. Note that the combination is rather rough statistically, considering the assumption of the Tobit model. We call the method the Log-Merge method, and combined them as the followings:

- When the predicted value of the Log-Tobit method is zero, the value is selected.
- When the predicted value of the Log-Tobit method is not zero, the predicted value of the Log-OLS method is selected.

TABLE III. RECALL, PRECISION AND F-MEASURE OF THE MODELS

	Recall	Precision	F-measure
Log-OLS method	44.2%	64.4%	52.4%
Log-Tobit method	74.4%	52.5%	61.5%
Log-Merge method	46.5%	56.3%	51.0%

TABLE IV. THE DISTRIBUTION OF THE DEPENDENT VARIABLE

	Average	Variance	p-value
Non-log-transformed	8.8	530.6	0.00
Log-transformed	1.2	1.7	0.00

We predicted the number of defects using the model, and the prediction accuracy is shown in Table II. Compared with the Log-OLS method, although average *BRE* and median *BRE* were 6% and 3% lower, Pred25 showed 12% improvement. Compared with the Log-Tobit method, average *AE*, average *BRE*, and Pred25 were improved. Especially, average *BRE* showed 28% improvement. However, median *AE* and median *BRE* were lower. Table III shows recall, precision and F-measure of the Log-Merge method. F-measure is lower than the Log-OLS method and the Log-Tobit method. It was small that the difference of F-measure between the Log-Merge method and Log-OLS method.

From the result, the Log-Merge method seems a bit better than the Log-OLS method, when focusing on Pred25. However, statistically speaking, the combination is rather rough, and hence it is not clear that the Log-Merge method always shows higher accuracy than the Log-OLS method, when other datasets are used.

D. *Applicability of the Poisson regression*

Similar to the Tobit model, the Poisson regression model assumes there are many data points whose value is zero on the dependent variable. The Poisson regression model is sometimes used to build a defect prediction model [5]. In more detail, the Poisson regression model assumes the distribution of the dependent variable is the Poisson distribution.

However, the average and the variance of the variable is same in the Poisson distribution, and the assumption is not fit to some datasets. Actually, the distribution of the dependent variable in our dataset, and the statistical test (i.e., goodness of fit test) showed the distribution of the variable is not regarded as the Poisson distribution, since the p-value was smaller than 0.05. The distribution of the variable is shown in Table IV. That is, applying the Poisson regression model to our dataset is not appropriate.

V. RELATED WORK

As long as we know, the Tobit model was not applied to software defect prediction, and hence the performance of the model has not been evaluated. Shao et al. [18] used the Tobit model to analyze efficiency of investments about information technology. Sojer at al. [19] used the Tobit model to analyze developers' knowledge about internet code licenses. However, their research topics are different from software quality.

Some statistical models consider censored distribution of a variable. For example, survival analysis [9] considers censored data, and some papers [10][17] used it to analyze characteristics of software such as the duration of open source project. However, the dependent variable in survival analysis is duration, and therefore, it is not used to predict the number of defects.

VI. CONCLUSIONS

In this paper, we applied the Tobit model to the software defect prediction, and evaluated the prediction accuracy compared with an ordinary prediction model. In the experiment, we used the regression model based on ordinary least squares and the Tobit model. Additionally, we applied log-transformation before building the models. As a result, the Tobit model with log-transformation showed higher accuracy than other models. Median *BRE* was 14% improvement, and Pred25 was 7% improvement, compared with the Log-OLS method with log-transformation. As future work, we will apply the Tobit model to other datasets, and analyze prediction accuracy, focusing on data points whose number of defects are zero.

ACKNOWLEDGMENT

This research was partially supported by the Japan Ministry of Education, Science, Sports, and Culture [Grant-in-Aid for Scientific Research (C) (No. 25330090)]

REFERENCES

[1] C. Burgess, and M. Lefley, "Can genetic programming improve software effort estimation? A comparative evaluation," Journal of Information and Software Technology, Vol.43, No.14, pp.863–873, 2001.

[2] S. Conte, H. Dunsmore, and V. Shen, *Software Engineering, Metrics and Models*, Benjamin/Cummings, 1986.

[3] International Software Benchmarking Standards Group, *ISBSG Estimating, Benchmarking and Research Suite Release 9*, ISBSG, 2004.

[4] Y. Kastro, and A. Bener, "A defect prediction method for software versioning," *Software Quality Control*, vol.16, no.4, pp.543-562,2008.

[5] T. Khoshgoftaar, K. Gao, and R. Szabo, "An application of zero-inflated poisson regression for software fault prediction," *In Proc. of International Symposium on Software Reliability Engineering (ISSRE)*, pp.66-73 ,2001.

[6] T. Khoshgoftaar, and N. Seliya, "Fault Prediction Modeling for Software Quality Estimation: Comparing Commonly Used Techniques," *Empirical Software Engineering*, vol.8, no.3, pp.255-283, 2003.

[7] B. Kitchenham, S. MacDonell, L. Pickard, and M. Shepperd, "What Accuracy Statistics Really Measure," *In Proc. of IEE Software*, Vol.148, No.3, pp.81–85, 2001.

[8] B. Kitchenham, and E. Mendes, "Why comparative effort prediction studies may be invalid," *In Proc. of International Conference on Predictor Models in Software Engineering (PROMISE)*, art.4, p.5, 2009.

[9] J. Klein, and M. Moeschberger, *Survival Analysis: Techniques for Censored and Truncated Data*, Springer, 2003.

[10] A. Koru, D. Zhang, and H. Liu, "Modeling the Effect of Size on Defect Proneness for Open-Source Software," *In Proc. of International Workshop on Predictor Models in Software Engineering (PROMISE)*, p.10, 2007.

[11] C. Lokan, “What Should You Optimize When Building an Estimation Model?,” In Proc. of International Software Metrics Symposium (METRICS), pp.34, Como, Italy, Sep. 2005.

[12] C. Lokan, and E. Mendes, “Cross-company and single-company effort models using the ISBSG Database: a further replicated study,” *In Proc. of the International Symposium on Empirical Software Engineering (ISESE)*, pp.75–84, Rio de Janeiro, Brazil, Sep. 2006.

[13] E. Mendes, C. Lokan, R. Harrison, and C. Triggs, “A Replicated Comparison of Cross-company and Within-company Effort Estimation Models using the ISBSG Database,” *In Proc. of International Software Metrics Symposium (METRICS)*, p.36,2005.

[14] Y. Miyazaki, M. Terakado, K. Ozaki, and H. Nozaki, “Robust Regression for Developing Software Estimation Models,” *Journal of Systems and Software*, vol.27, no.1, pp.3-16, 1994.

[15] M. Mizuochi, “Introduction to Censored and Truncated Regression Models,” *Sociological Theory and Methods*, vol. 24, no, 1, pp.129-138, 2009 (in Japanese).

[16] K. Mølokken-Østvold, and M. Jørgensen, “A Comparison of Software Project Overruns-Flexible versus Sequential Development Models,” *IEEE Transactions on Software Engineering*, vol.31, no.9, pp.754-766, 2005.

[17] I. Samoladas, L. Angelis, and I. Stamelos, “Survival analysis on the duration of open source projects.” *Information and Software Technology*, vol.52, no.9, 2010.

[18] B. Shao, and W. Lin, “Technical efficiency analysis of information technology investments: A two-stage empirical investigation,” *Information and Management*, vol.39,no. 5, pp. 391-401, 2002.

[19] M. Sojer, and J. Henkel, “License risks from ad hoc reuse of code from the internet,” *Communications of the ACM*, vol.54, no.12, pp.74-81, 2011.

[20] M. Sokolova, and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing and Management*, vol.45, no.4, pp.427-437, 2009.

[21] J. Tobin, “Estimation of relationships for limited dependent variables,” *Econometrica*, vol.26, no.1, pp.24-36, 1956.

[22] Y. Murakami, M. Tsunoda, and K. Toda, “An Experiment of Software Defect Prediction Based on the Tobit Model,” *In Proc. of Software Engineering Symposium (SES)*, pp.77-82, 2015 (in Japanese).