International Journal of Software Engineering and Its Applications
Vol. 9, No. 2 (2015), pp. 211-218
http://dx.doi.org/10.14257/ijseia.2015.9.2.18

A Robustness Analysis of Imputation Method for Software Development Project Data: Missing Value Treatment for Software Quality Prediction

Takayuki Morita† and Mitsuhiro Kimura‡*

†*Graduate School of Science Engineering, Hosei University*
3-7-2 Kajinocho, Koganei-shi, Tokyo,184-8584, Japan
E-mail: takayuki.morita.9j@stu.hosei.ac.jp

‡*Faculty of Science Engineering, Hosei University*
3-7-2 Kajinocho, Koganei-shi, Tokyo,184-8584, Japan
E-mail: kim@hosei.ac.jp

Abstract

As our goal, we are interested in estimating the degree of software reliability based on software development project data. It is widely-known that several software development attributes which are measured can be used to evaluate and predict software reliability/quality via multi-variable analyses. In this article, we focus on the data treatment method which is needed prior to the software reliability assessment, since the software development data sets often include missing data. This paper discusses the method of data preparation against missing data and their effectiveness by using the Random Forest as a multi-variable analysis.

Keywords: *Software Quality, Software Project Data, Missing Value Imputation, EMB Algorithm, Random Forest*

Introduction

It is known that software quality is influenced by how the software was developed. Several past studies have proposed the methods to estimate the software reliability from the software development attributes data sets (*e.g.*, [1-4]). Such software project data sets, i.e. multi-variable software development data sets consist of many attributes of software development activities and records, *e.g.*, SLOC (source lines of code), the number of pages of design review reports, and so on. Among those variables, several ones are chosen as the explanatory variables for multi-variable analysis in order to predict the objective variable. As an objective variable, in general, it is often the case that the number of detected software failures or the software failure occurrence in the testing/operational phase is selected. By choosing the effective explanatory variables (in the sense of statistics, for example), the prediction performance of the software quality is improved.

However, in the real situation, the data sets collected from the software development companies contain many missing values. One reason of yielding missing values may be the busyness of the development companies' workers. That is, in the software development companies, measuring and recording many attributes of the current development projects are difficult to be perfectly executed because all workers are so busy and the goal of the companies is not to record the data but to deliver the quality software product to the users at the strict delivery due date. Therefore it is considered to be natural that these data sets contain missing values.

* Corresponding Author

ISSN: 1738-9984 IJSEIA

Hence we focus on the performance of the treatment methods of missing values included in the software development project data sets in this study. In particular, the EMB (expectation-maximization with bootstrapping)-based multiple imputation method is used to fill the blanks of data sets. On the performance evaluation of this imputation method, we compare the prediction performances of multi-variable analysis by using the complete data and the virtual incomplete data which are made by masking some values of the complete data. Furthermore, the multi-variable analysis is executed by the Random Forest.

Under such a set-up, we investigate the robustness of the imputation method. That is, if the prediction performances of the objective variable of each data set (complete data/virtual incomplete data) are considered the same, we can judge that our imputation method is acceptable and robust for the missing value treatment.

Software Project Data with Missing Values

The original data sets [5] to be analyzed here, were disclosed to us by the IPA/SEC[1] under the written contract. The data set consist of 3089 cases of the real companies' software development projects. The IPA/SEC has been continuously collecting such software development data sets from system integrators and software vendors in Japan in order to grasp the degree of the quality of the developed software systems and the goodness of the companies' development activities in our country. In other words, such multi-dimensional data sets may indicate the abilities of the software development companies and quality of their software products.

In terms of our data sets to be analyzed, the data have in total 611 attributes concerning software development activities. Among these attributes, one attribute shows the number of detected software faults in the operational phase after the release of the developed software system. This value directly reflects the software quality. Therefore one of our interests is to find the relationship among several development attributes and the number of detected software faults. More concretely, we would like to predict the number of software faults when the project attributes data are available before the software release or before the system test phase. If it is possible, the information on the number of failure occurrence will be able to help the software testing/development management.

Looking into the actual datasets, however, we see that the datasets contain a lot of missing values due to each company's information management scheme and some other reasons (one of them was mentioned in the previous section). Therefore we firstly need to fill the blanks of missing values in the data sheets with some substitute values.

In the next 2 sections, we explain a missing-data processing method and discuss the method of the missing pattern adjustment.

Preparation of the data sets

One of our research purposes is to estimate whether at least one software failure will be occurred in the operation phase or not. In the past research of us[1], we discussed how we could precisely predict the number of faults latent in the released software system. However, the market and users of the software systems strongly require the zero-bug software. From such a view point, occurring at least one failure during the first one month after the release is considered fatal. Therefore we use the number of failures occurred within a month after the release as the objective variable for the multi-variable analysis. Each value of this objective variables are categorized by two values ‘zero’ and ‘$\geqq 1$’.

On the selection of the explanatory variables for the multi-variable analysis, by considering the results of our past research[1] and other investigation, we found the

[1] Information-Technology Promotion Agency, Japan/Software Reliability Enhancement Center

influential variables are SLOC, the number of planned months, and the average manpower for the development. Hence we decided to use four variables (one objective and three explanatory variables) for the analysis, however, we need to impute the missing values as mentioned before.

Table 1 summarizes the missing patterns of the observed three explanatory variables and one objective one. There are 8 data which have some missing values out of 57 data. In the table, the value 0 means "missing" and the value 1 means "observed". We found 3 patterns of the existence of the missing values (6 data belong to the missing pattern 0-1-1-1, and other data do 0-0-1-1 and 1-0-0-1 in the table). We call this the missing pattern (7, 2, 1). On the other hand, 49 (= 57-8) data have no missing values. The objective variable i.e. the number of failures (in the right most column) were completely observed in the all 57 data sets.

Table 1. Missing Patterns Seen in the Original Data Sets

No		SLOC	The number of planned months	Average manpower	The number of failures within a month after the release
Missing pattern	1	0	1	1	1
	2	0	1	1	1
	3	0	1	1	1
	4	0	1	1	1
	5	0	1	1	1
	6	0	1	1	1
	7	0	0	1	1
	8	1	0	0	1
The number of missing values		7	2	1	0
The number of the data sets		57	57	57	57

0:Missing values. 1:Observed values.

In addition, in order to check the appropriateness of the data imputation method which is provided in the following, we prepare a complete data set from the original data set. The data set consists of 49 data. Therefore our intention in this paper can be listed below.

1. We prepare the virtual incomplete data set which contains missing values from the original complete data set.
2. The occurrence tendency of the missing pattern seen in Table 1 is introduced into the virtual data set.
3. The provided virtual data set is used to test the imputation method.
4. Both of the virtual and complete data sets are executed to estimate the objective variable by using the Random Forest.

If the prediction performances of the objective variable of each data set are considered the same, we can judge that our imputation method is acceptable for the missing value treatment.

Consideration of missing patterns

To make the virtual data sets, we insert missing values (by masking some values) based on the missing pattern (7, 2, 1) into the complete data sets. Upon this data preparation, in order to make the virtual data sets look as much like the original complete data sets as possible (cf. Table 1), we equalize the missing ratios between two data sets. The missing ratio is denoted by r = n, where r is the number of data sets holding missing values and n is the number of data sets. Therefore we set the following relationship for both data sets.

$$\frac{r_1}{n_1} = \frac{r_2}{n_2} \quad (1)$$

where

r_1: the number of the original data set having missing values,

n_1: the number of the original data set,

r_2: the number of the missing values in the complete data set to be estimated,

n_2: the number of the complete data set.

We solved the Eq. (1) with respect to r_2, and the value was r_2 = 6. 871. We rounded 6.871 to 7 as the number of masked data. We think that the missing ratio of the original

data sets and virtual ones must be as same as possible. Thus we decided three missing patterns (6, 2, 1) (*i.e.*, we discard one data set from No. 1 to No. 6 of Table 1), (6, 1, 1) (discard No. 7) and (7, 1, 0) (discard No. 8) by considering the size of data set. Consequently, we made three types of the virtual incomplete data sets. The number of data becomes 48 in all data sets.

After this procedure, we additionally applied the Box-Cox transformation to the data sets. Each of three virtual data sets consists of three variables (each takes continuous value) and one categorical variable (objective variable denoted by 'zero' and '≧1'). Among them, the continuous variables are desirable to obey a normal distribution, since the data imputation method which we employ later assumes normality. Therefore we conducted the Box-Cox transformation to the three continuous variables for each data set. Hence the virtual incomplete data sets denoted by (6, 2, 1), (6, 1, 1) and (7, 1, 0) are respectively imputed. Upon the imputation of the data, we employ the multiple imputations with EMB (expectation-maximization with bootstrapping) algorithm [6].

In this paper, we used "Amelia", the package in R, to impute by the multiple imputations with EMB algorithm [7]. We set the number of repeats of nonparametric bootstrapping 5 based on the past works [8]. Figure 1 shows a schematic diagram of the multiple imputations with the EMB algorithm.

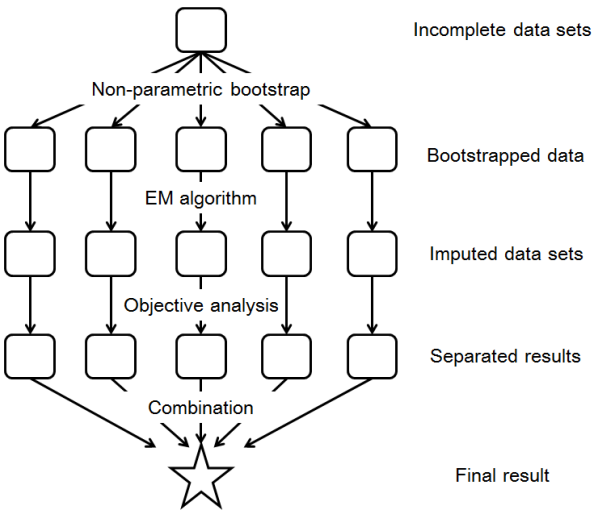

Figure 1. Schematic Diagram of Multiple Imputation with the EMB Algorithm

Multi-variable Analysis by Random Forest

Now we have two kinds of the data sets. One is the original complete data set, and the other is the masked complete data set (we called this the virtual incomplete data set[2]). The latter was imputed by the multiple imputation method with EMB. By using these data sets, we compare each estimation result of the two in terms of the degree of failure occurrence, in order to investigate the influence of the missing values treatment method and the estimation accuracy. The degree of failure occurrence is the objective variable, which is considered to be described by three explanatory variables. As mentioned before, this objective variable is a categorical one, which is shown by 'zero' and '≧1'.

We adopted the Random Forest to estimate the degree of failure occurrence[9, 10]. In this section, we explain the analysis method with Figure 2. Figure 2 illustrates the analysis procedure for the imputed data sets. All 48 data sets are separated 38 data sets for learning and 10 data sets for test which were chosen at random. We estimate by the Random Forest with learning data sets for the first step. After the learning, the parameters of the Random

[2] N.B.: we have three missing patterns of the virtual incomplete data set.

Forest are tuned. Therefore by applying the test data sets to the tuned Random Forest, we finally obtain the estimation result by taking a majority from five estimation results with five bootstrapped data sets. For the comparison of the estimation accuracy, we analyze the complete data set by the same fashion.

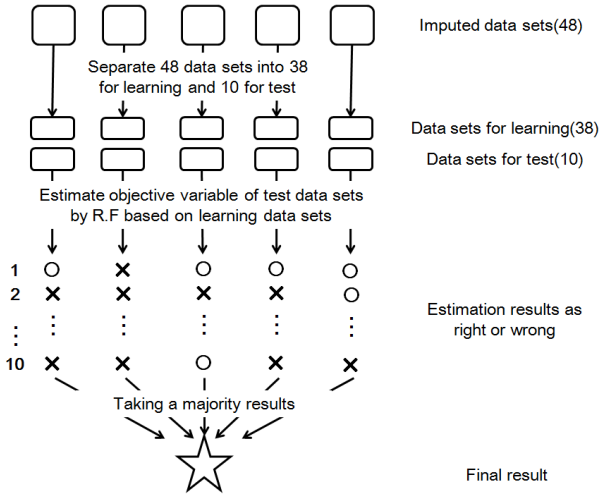

Figure 2. Analysis Processes of the Imputed Data Sets

Random Forest is one of the machine learning by means of ensemble learning [9] and builds decision trees which are combined as weak identifiers and conducts classification and regression as the multi-variable analysis. In this paper, we built classification trees consist of two categories in order to estimate failures by the three explanatory variables, *i.e.*, SLOC, the number of planned months and the average manpower.

Estimation Results by Random Forest

Table 2-4 show the results of estimated categories and true categories of the test data. We compare the two results, the estimated categories from the imputed data sets and that from the complete data sets considering missing pattern. According to the two results comparison on the missing pattern (6, 2, 1), there is no difference between two results from the view point of the total number of the correctly-estimated categories and each estimated result from No. 1 to No. 10. Thus we can regard the two estimation accuracy as nearly equivalent in the case of missing pattern (6, 2, 1). On the missing pattern (6, 1, 1), there is one difference of the estimation of test data No. 5 shown in Table 3. Moreover in Table 4, the difference is occurred at No. 1 and No. 2 for the (7, 1, 0).

Summarizing these three comparison results, we found that the serious differences did not occur, *i.e.*, the missing data treatment worked to some extent, but the estimation accuracy has a little fluctuation depending on the missing patterns.

Table 2. Results of Missing Pattern (6, 2, 1)

Test data No	The true number of failures	Missing Pattern	
		(6, 2, 1) from imputed data	(6, 2, 1) from complete data
1	over 1	0	0
2	over 1	over 1	over 1
3	0	over 1	over 1
4	over 1	over 1	over 1
5	0	over 1	over 1
6	0	0	0
7	0	0	0
8	0	0	0
9	0	over 1	over 1
10	0	over 1	over 1
Total number of rigtht estimations		5	5

Table 3. Results of Missing Pattern (6, 1, 1)

Test data No	The true number of failures	Missing Pattern	
		(6, 1, 1) from imputed data	(6, 1, 1) from complete data
1	over 1	0	0
2	over 1	0	0
3	0	over 1	over 1
4	over 1	over 1	over 1
5	0	0	over 1
6	0	0	0
7	0	0	0
8	0	0	0
9	0	over 1	over 1
10	0	over 1	over 1
Total number of rigtht estimations		5	4

Table 4. Results of Missing Pattern (7, 1, 0)

Test data No	The true number of failures	Missing Pattern	
		(7, 1, 0) from imputed data	(7, 1, 0) from complete data
1	over 1	0	over 1
2	over 1	0	over 1
3	0	over 1	over 1
4	over 1	over 1	over 1
5	0	over 1	over 1
6	0	0	0
7	0	0	0
8	0	0	0
9	0	over 1	over 1
10	0	over 1	over 1
Total number of rigtht estimations		4	6

Concluding Remarks

We have shown that the multiple imputation method exerted the decent performance by the Random Forest estimation. Our final goal is to provide the prediction scheme for the degree of failure occurrence of software system after its release. It is also desirable that the prediction works at the early stage of software development. In this paper, we could not provide the accurate prediction scheme but show our imputation method is considerably robust for the missing data sets. In other words, if the software development attributes data sets contain missing values, the methods which were applied in this paper can estimate the objective value by the Random Forest with keeping the same

performance as if the complete data set is available. For the future study, we need to develop the more accurate prediction method for the objective variable by Random Forest.

Acknowledgments

The authors are grateful to IPA/SEC (Information-technology Promotion Agency, Japan/ Software Reliability Enhancement Center) for providing the collected data sets and participating in discussion.

References

[1]. T. Morita, K. Esaki and M. Kimura, "A Note on Modeling of Quality Evaluation Based on Large Data Sets in Software Development Projects," APIEMS 2013 Conference Proceedings, **(2013),** Cebu.
[2]. M. Kimura, T. Fujiwara and S. Yamada, "A Note on Structure Equation Modeling for Software Reliability Assessment Based on Review-related Data," Proc. of 17th ISSAT Intern. Conf. on Reliability and Quality in Design, **(2011),** pp. 389-393.
[3]. M. Kimura and T. Fujiwara, "Practical Optimal Software Release Decision Making by Bootstrap Moving-Average Quality Control Chart," IJSEIA, vol. 4, no. 1, **(2002),** pp. 29-42.
[4]. H. Okamura, T. Hirata and T. Dohi, "Semi-Parametric Approach for Software Reliability Evaluation Using Mixed Gamma Distributions", IJSEIA, vol. 7, no. 4, **(2013),** pp. 401-414.
[5]. "IPA/SEC", White paper 2012-2013 on Software Development Data in Japan, IPA/SEC, **(2012).**
[6]. M. Takahashi and T. Ito, "Comparison of Competing Algorithm of Multiple Imputation," Japanese Joint Statistical Meeting, **(2013),** Japan.
[7]. J. Honaker, G. King and M. Blackwell, "Amelia II: A Program for Missing Data," Journal of Statistical Software, **(2011)**.
[8]. A. Gelman and J. Hill, "Data Analysis Using Regression and Multilevel/Hierarchical Models," Cambridge University Press, **(2006)**.
[9]. H. Habe, "Random Forest. (in Japanese)," IPSJ SIG Technical Report. 2012-CVIM-182, vol. 31, **(2012),** pp. 1-8.
[10]. L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, **(2001),** pp. 5-32.

