machine learning &
knowledge extraction

Article

Large Scale Fault Data Analysis and OSS Reliability Assessment Based on Quantification Method of the First Type

Yoshinobu Tamura [1,*,†] and Shigeru Yamada [2,†]

[1] Department of Intelligent Systems, Faculty of Information Technology, Tokyo City University, Tokyo 158-8557, Japan
[2] Graduate School of Engineering, Tottori University, Tottori 680-8552, Japan; yamada@tottori-u.ac.jp
* Correspondence: tamuray@tcu.ac.jp; Tel.: +81-3-5707-0104
† These authors contributed equally to this work.

Received: 19 July 2020; Accepted: 12 October 2020 ; Published: 20 October 2020

Abstract: Various big data sets are recorded on the server side of computer system. The big data are well defined as a volume, variety, and velocity (3V) model. The 3V model has been proposed by Gartner, Inc. as a first press release. 3V model means the volume, variety, and velocity in terms of data. The big data have 3V in well balance. Then, there are various categories in terms of the big data, e.g., sensor data, log data, customer data, financial data, weather data, picture data, movie data, and so on. In particular, the fault big data are well-known as the characteristic log data in software engineering. In this paper, we analyze the fault big data considering the unique features that arise from big data under the operation of open source software. In addition, we analyze actual data to show numerical examples of reliability assessment based on the results of multiple regression analysis well-known as the quantification method of the first type.

Keywords: fault big data; reliability analysis; multiple regression analysis; quantification method; open source project

1. Introduction

A waterfall development model is well-known as the traditional software development style. At present, the software development style has been changed to various development paradigms. In particular, the development style of open source software (OSS) has the unique style such as the OSS project. The OSS project has the development cycle in the flow such as development, version release, usage of users, reporting of bug, checking and modifying of OSS, and release of new version. Recently, the OSS with network connection service is increasing more and more. Thus, the numbers of network-oriented OSS such as cloud service, server, IoT (Internet of Things) device software have been increasing as well as the standalone software.

In the past, various methods based on software reliability growth models have been proposed by several research groups [1,2]. On the other hand, several research papers for OSS reliability assessment have been published [3].

There are many OSS reliability assessment methods based on the stochastic models. In addition, there are several methods based on empirical data analysis [4,5]. In particular, it is very useful for the OSS developers to understand the trend of fault big data recorded on the OSS bug tracking system from the standpoint of bird's-eye view. The organization of this paper is as follows: Section 2: discusses the relationship between the raw data and categorical one. Section 3: proposes the multiple regression analysis in order to solve the problem in terms of the degree of freedom for the large scale fault data. Section 4: describes and analyzes the forward-backward stepwise selection method by analyzing the fault big data. Section 5: discusses the upper and lower confidence limits based on the typical hazard rate model. Section 6: discusses the characteristics of the proposed method.

Many software reliability assessment methods based on the stochastic model have been proposed by several researchers [6–8]. Recently, it is difficult to assess the software reliability because there are various software development styles. Historically, the fault data sets are used for the software reliability assessment. In addition, the software reliability assessment methods based on the measurements of software metrics have been proposed in the past [9,10]. At present, many kinds of fault data have been recorded on the bug tracking system in the case of OSS. In particular, it will be useful for the reliability assessment by using various categorized fault data sets in the case of OSS. We will be able to propose the high accuracy method by integrating the stochastic models and the statistical analysis if we can assess the fault big data from the standpoint of statistical analysis.

As the related works, several research papers have proposed the methods in terms of the upper and lower limits based on software reliability growth models [1,2], and the empirical approach for OSS [3]. However, it is difficult to understand the upper and lower bounds of the stochastic model for the big data because of the problem for the degree of freedom. Generally, the degree of freedom is given by the number of data. However, it is difficult to obtain the degree of freedom from the big data, because the data set is in the large scale. Then, we will be able to use the number of explanatory variables in place of the number of data. In this paper, we propose the data analysis method based on a quantification method of the first type. Then, we focus on the fault big data analysis with a more simple method, because the analyses of fault big data are required to take a lot of time for calculation and analysis. The multiple regression-based model to analyze financial data has been proposed in the financial research area [11,12]. Moreover, the multiple regression analysis is used in the research area of network [13]. In this way, the statistical methods such as the multiple regression analysis have been applied to various research areas. This paper proposes the method based on the statistical analysis and typical hazard rate model for the large scale fault data analysis and OSS reliability assessment. Furthermore, we show several analysis examples based on the proposed method by using the actual fault big data.

2. Fault Data Analysis

Table 1 presents the part of raw data in terms of fault big data. We can use the data in terms of the time and categories as shown in Table 1. However, it is difficult to analyze the categorical data for the reliability assessment. Historically, the data sets in terms of the number of faults and the time between software failures have been well-used for the software reliability assessment. Therefore, we convert the categorical data sets to the number of software faults. For example, Table 1 can be converted to Table 2. For example, each line in Table 1 means one fault, e.g., Table 1 contains 5 fault. In addition, the unit of "Opened" is "day". Many software reliability growth models have been proposed by several researchers as follows:

- Non-homogeneous Poisson process (NHPP) model (Fault Count Type).
- Hazard rate model (Time Interval of Fault Detection).
- Stochastic differential equation model (Fault Count Type).
- Logistic curve model (Fault Count Type).

Therefore, it will stand to reason that the categorical data sets are converted to the number of faults and the time between software failures from the stand point of the software reliability engineering. We define the data sets in Table 2 as the dummy variables considering the multiple regression analysis.

Table 1. A part of the raw fault big data.

Opened	Product	Component	Version	Reporter	Assignee	Severity	Status	Resolution	Hardware	OS
0.83895	Apache httpd-1.3	Documentation	1.3.23	rineau+apachebugzilla	docs	normal	CLOSED	FIXED	Other	other
1.12118	Apache httpd-1.3	Other mods	1.3.24	siegfried.delwiche	bugs	blocker	CLOSED	FIXED	PC	Linux
0.17191	Apache httpd-1.3	Documentation	1.3.23	dard	bugs	normal	CLOSED	FIXED	All	FreeBSD
0.40766	Apache httpd-1.3	Other	1.3.23	bernard.l.dubreuil	docs	minor	CLOSED	FIXED	All	All
0.51352	Apache httpd-1.3	Other	1.3.23	george	bugs	normal	CLOSED	WORKSFORME	PC	Linux

Table 2. A part of the numerical value converted from the raw fault big data.

Opened	Product	Component	Version	Reporter	Assignee	Severity	Status	Resolution	Hardware	OS
0.83895	898	815	95	1	815	4946	2426	2910	1460	912
1.12118	898	91	62	1	8378	392	2426	2910	4755	3347
0.17191	898	815	95	1	8378	4946	2426	2910	2188	278
0.40766	898	141	95	2	815	658	2426	2910	2188	2812
0.51352	898	141	95	1	8378	4946	2426	335	4755	3347

3. Multiple Regression Analysis

Generally, the number of data are used as the degree of freedom in the statistics. In the case of the big data, it is very difficult to estimate the upper and lower limits from the number of big data in the stochastic models, because the volume of data is huge. Many methods of OSS empirical assessment have been proposed [4,5,14,15]. However, the size of fault data in OSS is large. Therefore, it is difficult to assess the fault big data. Then, we focus on the number of explanatory variables. We will be able to estimate the upper and lower bounds by using the number of explanatory variables as the degree of freedom. The multiple regression analysis is well-known as the analysis method understanding the relationship between the objective variable and explanatory one. The analysis step in this paper is shown as follows:

Step 1: The pairplots for each factor are used in order to overlooking the fault big data.
Step 2: We apply the heatmap to the decision of the objective variable
Step 3: The explanatory variables are narrowed by using the forward-backward stepwise selection method. Then, the degree of freedom is decided by the number of explanatory variables.
Step 4: The upper and lower bounds are estimated from the stochastic model and the degree of freedom in place of the number of explanatory variables.

We show analysis examples by using the Apache HTTP Server Project [16] as the OSS. At first, we show the pairplot for the OSS fault big data in Figures 1–4. We show the explanatory variables as follows:

Opened: The date and time recorded on the bug tracking system,
Changed: The modified date and time.
Product: The name of product included in OSS.
Component: The name of component included in OSS.
Version: The version number of OSS.
Reporter: The nickname of fault reporter.
Assignee: The nickname of fault assignee.
Severity: The level of fault.
Status: The fixing status of fault.
Resolution: The status of resolution of fault.
Hardware: The name of hardware under fault occurrence.
OS: The name of operating system under fault occurrence.
Summary: The brief contents of fault.

The set of 10,000 lines data are plotted in Figures 1–4, respectively. Figures 1–4 simply visualize the whole data. We can understand the whole trend of data from Figures 1–4. Then, the number of whole data is about 130,000 data sets. In addition, all categories are simply shown by using three figures every three categories because of the convenience for the paper size.

Generally, the equation of multiple regression is given as follows:

$$F = \beta + \alpha_1 x_1 + \alpha_2 x_2 \cdots + \alpha_n x_n, \tag{1}$$

where F is the objective variable, α_i i-th partial regression coefficient, and x_i $(i = 1, 2, \cdots, n)$ is i-th explanatory variable. β is the intercept.

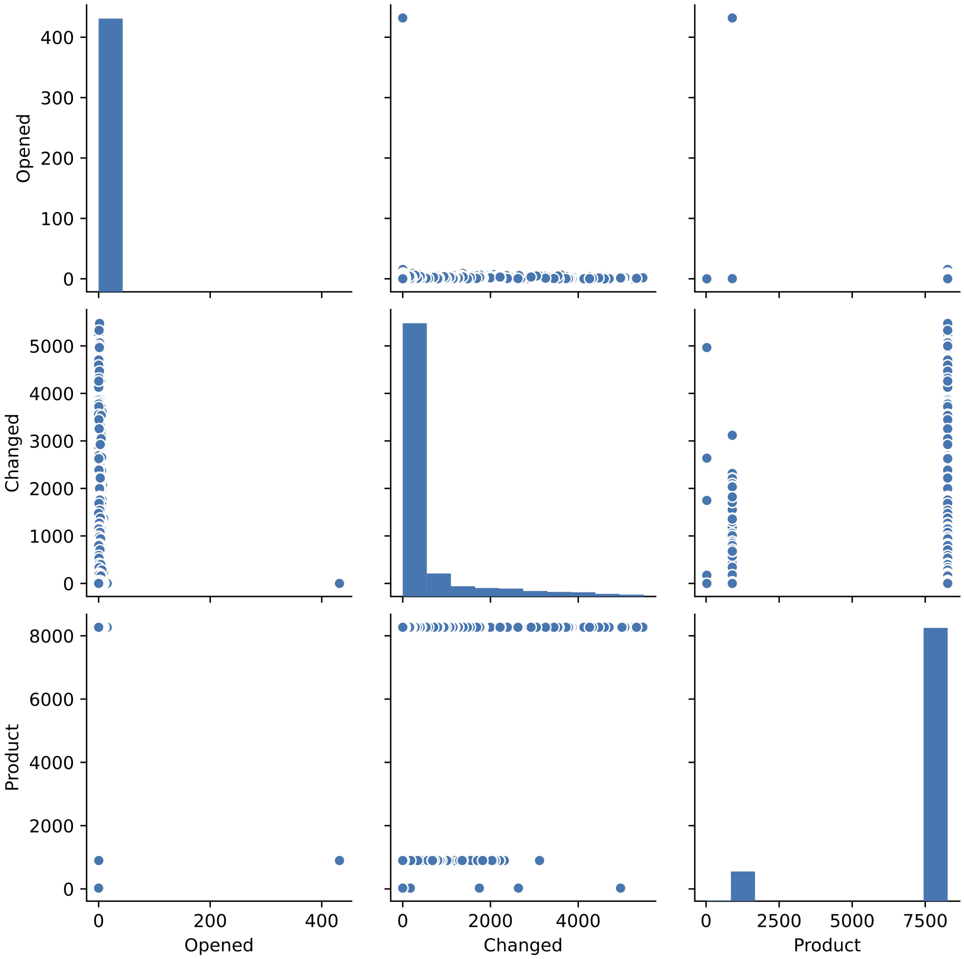

Figure 1. The pairplot for actual fault big data (1).

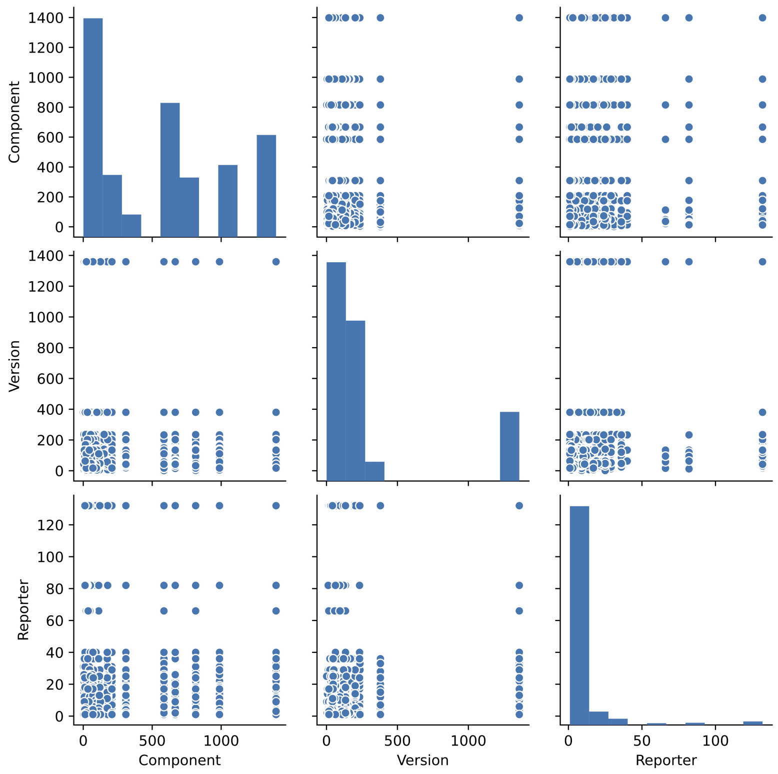

Figure 2. The pairplot for actual fault big data (2).

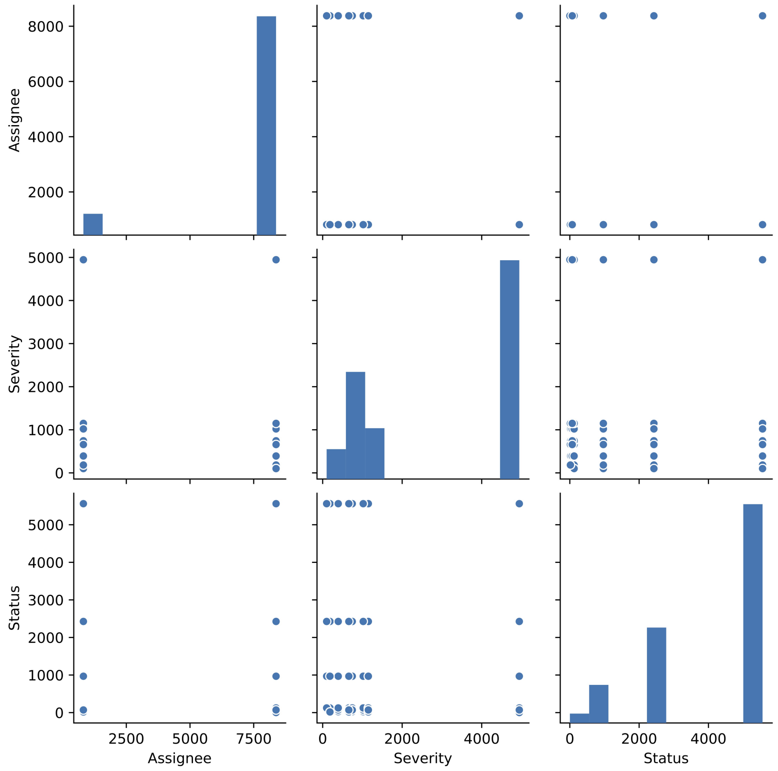

Figure 3. The pairplot for actual fault big data (3).

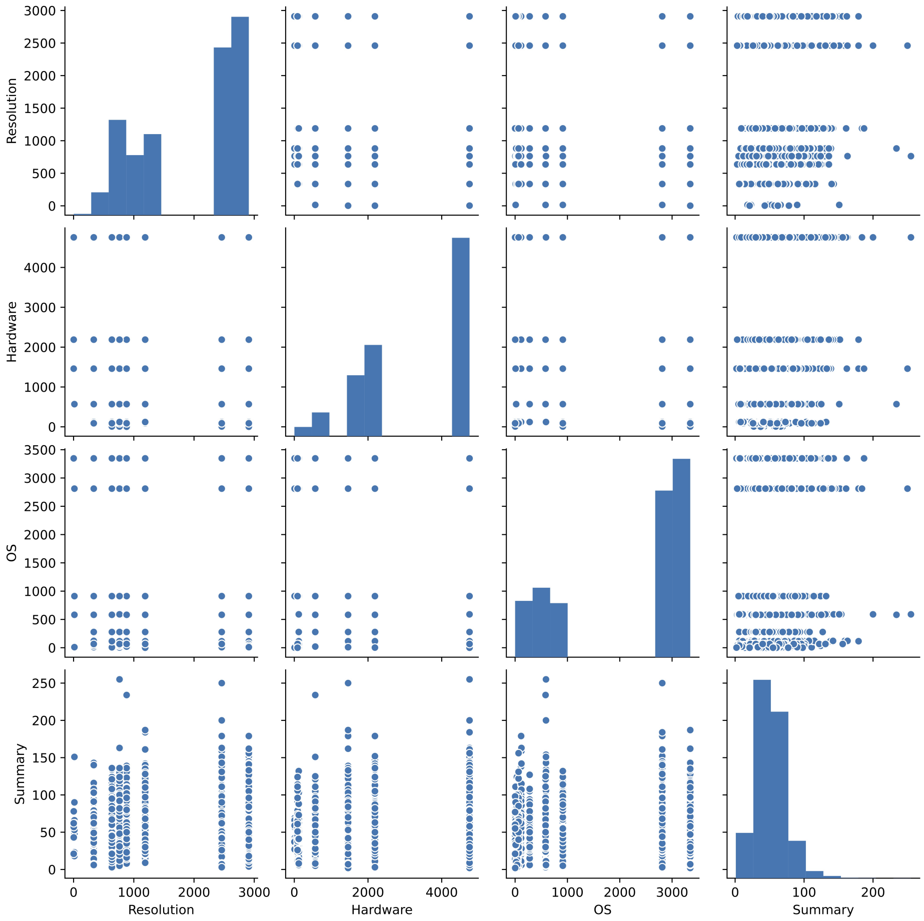

Figure 4. The pairplot for actual fault big data (4).

Deciding the objective variable, we discuss the estimation results by the heatmap analysis. Then, the heatmap for actual fault big data is shown in Figure 5. From Figure 5, we find that the weight parameters of “Hardware”, “OS”, “Changed”, and “Status”are large. Therefore, we focus on 4 factors of “Hardware”, “OS”, “Changed”, and “Status” - as the objective variable, respectively. We analyze the data in the period from January 2001 to May 2020. The x and y axes of Figures 1–5 are analyzed by using the values of Table 2.

Moreover, we analyze all categories of OSS fault big data by the multiple regression. The estimation results based on multiple regression analysis in cases of Hardware, OS, Changed, and Status as objective variables is shown in Table 3. For example, the top of Table 3 denotes that the objective variable is Hardware. Then, the other categories denote the explanatory variables. From Table 3, the multivariate regression models are obtained as follows:

$$\begin{aligned}
F_{hardware} &= 1645.7 + 2.791757x_1 - 0.067763x_2 + 0.051574x_3 - 0.008983x_4 \\
&- 0.131288x_5 - 0.146627x_6 + 0.058349x_7 + 0.063834x_8 - 0.02625x_9 \\
&+ 0.003068x_{10} + 0.342187x_{11} - 0.452142x_{12}, \qquad (2) \\
F_{os} &= 1765.5 + 0.546647x_1 - 0.042945x_2 + 0.020329x_3 - 0.072043x_4 \\
&+ 0.162994x_5 + 0.040314x_6 - 0.035459x_7 - 0.010353x_8 - 0.038089x_9 \\
&- 0.028639x_{10} + 0.199227x_{11} + 3.442676x_{12}, \qquad (3) \\
F_{changed} &= 159.3 + 0.43598x_1 + 0.027367x_2 - 0.093938x_3 - 0.054772x_4 \\
&+ 0.53278x_5 + 0.019791x_6 - 0.003285x_7 + 0.120565x_8 - 0.282455x_9 \\
&- 0.024331x_{10} + 0.026484x_{11} + 1.344123x_{12}, \qquad (4) \\
F_{status} &= 3191.5 - 1.334526x_1 + 0.444458x_2 + 0.030289x_3 + 0.069858x_4 \\
&- 0.197055x_5 + 0.366868x_6 - 0.018191x_7 + 0.025926x_8 + 0.423487x_9 \\
&- 0.034746x_{10} - 0.086595x_{11} + 1.250057x_{12}. \qquad (5)
\end{aligned}$$

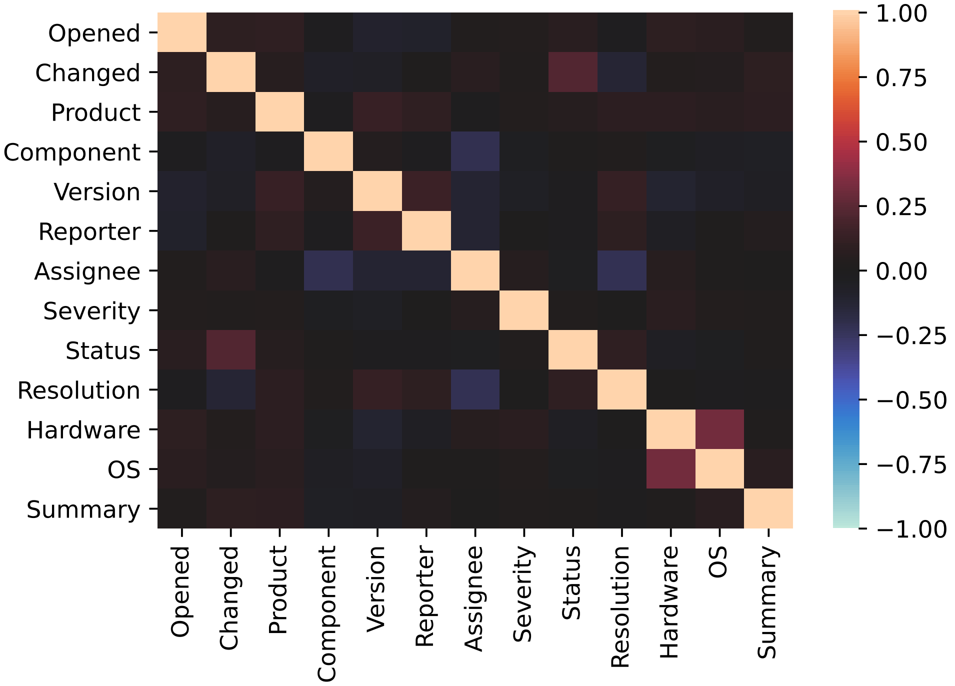

Figure 5. The heatmap for actual fault big data.

Table 3. The estimation results in cases Hardware, OS, Changed, and Status as objective variables.

Hardware	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	1645.720058	114.700127	14.348	0
Opened	2.791757	3.498686	0.7979	0.424923
Changed	−0.067763	0.017142	−3.9531	0.000078
Product	0.051574	0.005491	9.3933	0
Component	−0.008983	0.032654	−0.2751	0.783246
Version	−0.131288	0.037497	−3.5013	0.000465
Reporter	−0.146627	0.873514	−0.1679	0.866698
Assignee	0.058349	0.005035	11.5888	0
Severity	0.063834	0.00753	8.4776	0
Status	−0.02625	0.008096	−3.2424	0.00119
Resolution	0.003068	0.016816	0.1824	0.855249
OS	0.342187	0.012378	27.6439	0
Summary	−0.452142	0.712837	−0.6343	0.525911
OS	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	1765.506001	86.557174	20.397	0
Opened	0.546647	2.669727	0.2048	0.837766
Changed	0.042945	0.013078	3.2837	0.001028
Product	0.020329	0.004211	4.8277	0.000001
Component	−0.072043	0.024922	−2.8908	0.003852
Version	0.162994	0.028514	5.7162	0
Reporter	0.040314	0.666462	0.0605	0.951767
Assignee	−0.035459	0.003937	−9.0061	0
Severity	−0.010353	0.005781	−1.7908	0.073361
Status	−0.038089	0.006183	−6.1603	0
Resolution	−0.028639	0.012835	−2.2314	0.02568
Hardware	0.199227	0.007261	27.4381	0
Summary	3.442676	0.539452	6.3818	0
Changed	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	159.274647	69.47537	2.2925	0.021897
Opened	0.43598	2.096636	0.2079	0.835278
Product	0.027367	0.003366	8.1295	0
Component	−0.093938	0.019572	−4.7996	0.000002
Version	−0.054772	0.022467	−2.4379	0.014792
Reporter	0.53278	0.523398	1.0179	0.30874
Assignee	0.019791	0.003121	6.3411	0
Severity	−0.003285	0.004545	−0.7227	0.469904
Status	0.120565	0.004782	25.2141	0
Resolution	−0.282455	0.009813	−28.7829	0
Hardware	−0.024331	0.005928	−4.1044	0.000041
OS	0.026484	0.007756	3.4146	0.000642
Summary	1.344123	0.427139	3.1468	0.001656
Status	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	3191.460936	129.213128	24.6992	0
Opened	−1.334526	4.025544	−0.3315	0.740264
Changed	0.444458	0.019224	23.1196	0
Product	0.030289	0.006215	4.8737	0.000001
Component	0.069858	0.037401	1.8678	0.061817
Version	−0.197055	0.043141	−4.5677	0.000005
Reporter	0.366868	1.004787	0.3651	0.715031
Assignee	−0.018191	0.005768	−3.154	0.001616
Severity	0.025926	0.008674	2.9889	0.002807
Resolution	0.423487	0.018391	23.0264	0
Hardware	−0.034746	0.011364	−3.0576	0.002238
OS	−0.086595	0.014885	−5.8176	0
Summary	1.250057	0.816914	1.5302	0.125997

4. Forward-Backward Stepwise Selection Method

The forward-backward stepwise selection method is well-known as the selection method of explanatory variables for the multiple regression analysis. We use the forward-backward stepwise selection method as the multiple regression analysis for OSS fault big data. The forward-backward stepwise selection method is well-known as the selection method of explanatory variables in multiple regression analysis. In particular, we apply the backward stepwise selection method. Then, we consider the following steps:

Step 1: All explanatory variables are analyzed by the multiple regression.
Step 2: As the results of step 1, the explanatory variable is removed if p-value becomes large than 0.01.
Step 3: The selected explanatory variables are analyzed by the multiple regression again.
Step 4: The above steps 1 and 2 are continued until there is no p-value of explanatory variable larger than 0.01.

There are many methods as forward-backward stepwise selection ones. Then, it will be difficult to analyze the fault big data sets by using the other complex analysis method. Therefore, this paper is simply analyzed by above steps, because the fault big data sets have many factors and lines of bugs. In the case of the big data, it is very important to consider the calculation time and complexity in the estimation.

From Table 4, the multivariate regression models based on backward stepwise selection method are obtained as follows:

$$\begin{aligned} F_{hardware} &= 1626.4 - 0.068803x_1 + 0.05123x_2 - 0.130621x_3 + 0.058414x_4 \\ &+ 0.063954x_5 - 0.026041x_6 + 0.341771x_7, \qquad (6) \\ F_{os} &= 1743.1 + 0.043135x_1 + 0.020108x_2 - 0.071013x_3 + 0.163322x_4 \\ &- 0.035639x_5 - 0.038459x_6 - 0.028896x_7 + 0.198147x_8 + 3.431072x_9, \qquad (7) \\ F_{changed} &= 156.8 + 0.027344x_1 - 0.09417x_2 - 0.051485x_3 + 0.019413x_4 \\ &+ 0.120492x_5 - 0.282509x_6 - 0.024697x_7 + 0.026598x_8 + 1.348889x_9, \qquad (8) \\ F_{status} &= 3316.2 + 0.444006x_1 + 0.03109x_2 - 0.198328x_3 - 0.021023x_4 \\ &+ 0.025704x_5 + 0.422041x_6 - 0.034968x_7 - 0.086049x_8. \qquad (9) \end{aligned}$$

In particular, the selection results from explanatory variable is shown in Table 5 by using the method of backward stepwise selection. From Table 5, “Product”, “Version”, and “Assignee”are included as the common factors for all objective variables. This means that these three factors are very important factors to detect and fix the fault recorded on the bug tracking system. From this estimation results, we consider that the OSS developers can appropriately manage by using the information obtained from “Product”, “Version”, and “Assignee”. On the other hand, “Opened” and “Reporter” have been removed from all explanatory variables by using the backward stepwise selection method. In other words, “Opened” and “Reporter” may not be important from the standpoint of the quality control of OSS.

Table 4. The estimation results by backward stepwise selection method in cases Hardware, OS, Changed, and Status as objective variables.

Hardware	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	1626.405219	97.43963	16.6914	0
Changed	−0.068803	0.016619	−4.1401	0.000035
Product	0.05123	0.00532	9.6292	0
Version	−0.130621	0.037285	−3.5033	0.000462
Assignee	0.058414	0.004989	11.7087	0
Severity	0.063954	0.007503	8.5233	0
Status	−0.026041	0.007774	−3.3498	0.000812
OS	0.341771	0.012315	27.7524	0
OS	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	1743.085854	85.250018	20.4468	0
Changed	0.043135	0.01308	3.2978	0.000978
Product	0.020108	0.004139	4.8586	0.000001
Component	−0.071013	0.024916	−2.8501	0.00438
Version	0.163322	0.028485	5.7336	0
Assignee	−0.035639	0.00385	−9.2573	0
Status	−0.038459	0.006145	−6.2587	0
Resolution	−0.028896	0.012789	−2.2595	0.023879
Hardware	0.198147	0.007192	27.5505	0
Summary	3.431072	0.537922	6.3784	0
Changed	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	156.810893	68.42801	2.2916	0.02195
Product	0.027344	0.003314	8.2522	0
Component	−0.09417	0.019566	−4.8129	0.000002
Version	−0.051485	0.022444	−2.2939	0.021816
Assignee	0.019413	0.003056	6.353	0
Status	0.120492	0.004752	25.3539	0
Resolution	−0.282509	0.009778	−28.8923	0
Hardware	−0.024697	0.00588	−4.2001	0.000027
OS	0.026598	0.007748	3.4327	0.0006
Summary	1.348889	0.425998	3.1664	0.001548
Status	**Estimate**	**Std. Error**	**t Value**	***p* Value**
Intercept	3316.187141	116.49209	28.4671	0
Changed	0.444006	0.019226	23.0943	0
Product	0.03109	0.006181	5.0303	0
Version	−0.198328	0.043107	−4.6008	0.000004
Assignee	−0.021023	0.005759	−3.6502	0.000263
Severity	0.025704	0.008673	2.9637	0.003048
Resolution	0.422041	0.018285	23.0815	0
Hardware	−0.034968	0.011359	−3.0784	0.002087
OS	−0.086049	0.014875	−5.7846	0

Table 5. The selection results of explanatory variable.

Factors Selected from All Objective Variables	**Factors Removed from All Objective Variables**
Product	Opened
Version	Reporter
Assignee	———

5. Multiple Regression Analysis with Application to Reliability Assessment

Many software reliability assessment models have been proposed in the past [17–21]. In particular, the hazard rate model is well-known as the typical software reliability model. We apply the hazard rate model to the time-interval between correction faults. The distribution function of $X_k (k = 1,\ 2,\ \cdots)$ representing the time-interval between correction faults of $(k-1)$th and k-th is defined as:

$$Q_k(x) \equiv \mathrm{Pr}\{X_k \leq x\} \qquad (x \geq 0), \tag{10}$$

where $\mathrm{Pr}\{\Phi\}$ represents the occurrence probability event Φ. Therefore, the following derived function means the probability density function of X_k:

$$q_k(x) \equiv \frac{dQ_k(x)}{dx}. \tag{11}$$

From Equations (10) and (11), the hazard rate is given by the following equation:

$$z_k(x) \equiv \frac{q_k(x)}{1 - Q_k(x)}, \tag{12}$$

where the hazard rate means [1,22] the software correction rate when the software correction does not occur during the time-interval $(0,\ x]$. Therefore, the software reliability assessment measures are obtained from the typical hazard rate model in Equation (12). The probability density function can be derived as

$$z_k(x) \quad = \quad \phi(N - k + 1),$$

where N is the number of latent faults in OSS, ϕ the hazard rate per inherent fault. Then, the mean time between software failures correction ($MTBF_c$) is given as follows:

$$\begin{aligned} \mathrm{E}[X_k] &= \int_0^\infty x q_k(x) dx \\ &= \int_0^\infty 1 - Q_k(x) dx \\ &= \int_0^\infty e^{-\phi(N-k+1)x} dx \\ &= \left[\frac{-e^{-\phi(N-k+1)x}}{\phi(N-k+1)} \right]_0^\infty \\ &= \frac{1}{\phi(N-k+1)}. \end{aligned} \tag{13}$$

It is important to assess the upper and lower bounds of $MTBF_c$, $\mathrm{E}[X_k]$, because the difficulty of fault correction continuously keeps the variation state. Then, the upper and lower confidence limits for the $MTBF_c$ can be estimated from Chi-squared distribution. Then, the upper and lower confidence limits in $100(1-\alpha)$ percentage point of Chi-squared distribution for the $MTBF_c$ is given by

$$\frac{2m\ \hat{\mathrm{E}}[X_k]}{\chi^2_{2m}(\frac{\alpha}{2})} \leq \hat{\mathrm{E}}[X_k] \leq \frac{2m\ \hat{\mathrm{E}}[X_k]}{\chi^2_{2m}(1-\frac{\alpha}{2})}, \tag{14}$$

where m is the statistical degrees of freedom in objective variable of the regression equation. Then, we consider the regression equation of Equation (8). The explanatory variables are shown in Table 4. From Equation (8) and Table 4, the statistical degrees of freedom for the regression equation is 9 in the case of “Changed”. In the case of “Changed”, 90% upper and lower confidence limits for $MTBF_c$ are as follows:

$$\frac{18\,\hat{\mathrm{E}}[X_k]}{\chi^2_{18}(\frac{0.1}{2})} \le \hat{\mathrm{E}}[X_k] \le \frac{18\,\hat{\mathrm{E}}[X_k]}{\chi^2_{18}(1-\frac{0.1}{2})}. \tag{15}$$

As an example, the estimated upper and lower confidence limits for $MTBF_c$ are shown in Figure 6. As shown in Figure 6, we can assess the upper and lower confidence limits for $MTBF_c$. By using the estimation results in Equation (8) and Table 4, we can consider the influence degrees from the explanatory variables for $MTBF_c$ as the upper and lower confidence limits. The upper and lower bounds in Figure 6 mean the influences from “Product”, “Component”, “Version”, “Assignee”, “Status”, “Resolution”, “Hardware”, “OS”, and “Summary”as the main factors.

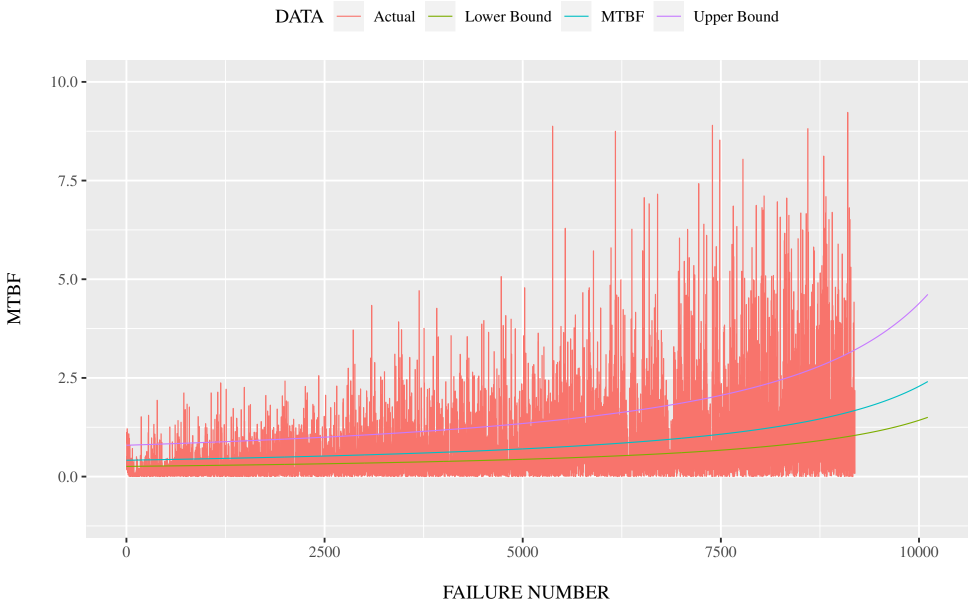

Figure 6. The estimated upper and lower confidence limits for the $MTBF_c$ in actual fault big data.

Several research papers have proposed the methods in terms of the upper and lower limits based on software reliability growth models, and the empirical approach for OSS [23,24]. As the comparison with the conventional method, we show the estimated upper and lower confidence limits based on the conventional method in Figure 7. The number of fault data is used as the degree of freedom in Figure 7. In Figure 7, we found that the conventional method can not estimate the upper and lower bounds accurately because the value of degree of freedom is large. On the other hand, the proposed method can appropriately estimate the upper and lower confidence limits for the actual fault big data, because the degree of freedom is properly given by using the proposed method.

As the comparison results of the other model, we compare the following Moranda model with the Jelinski–Moranda model.

$$\mathrm{E}[X_k] \quad = \quad \frac{1}{D(c^{k-1})}. \tag{16}$$

where c is the decreasing rate of hazard rate, D the hazard rate per inherent fault.

Similarly, we show the estimated upper and lower confidence limits for the $MTBF_c$ in actual fault big data in the case of the Moranda model in Figure 8. Moreover, we show the estimated upper and lower confidence limits based on the conventional method in the case of the Moranda model in Figure 9.

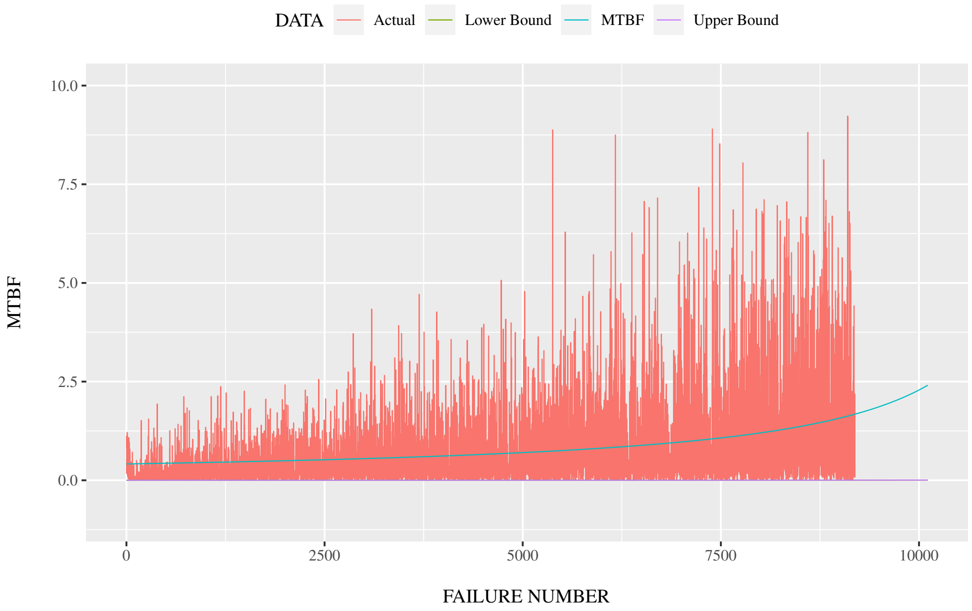

Figure 7. The estimated upper and lower confidence limits based on the conventional method.

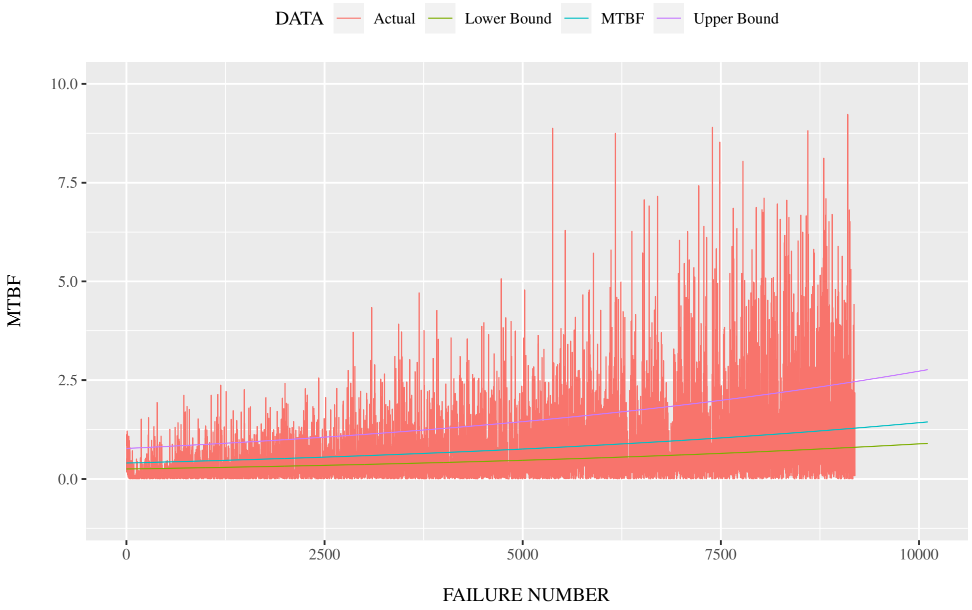

Figure 8. The estimated upper and lower confidence limits for the $MTBF_c$ in actual fault big data in the case of the Moranda model.

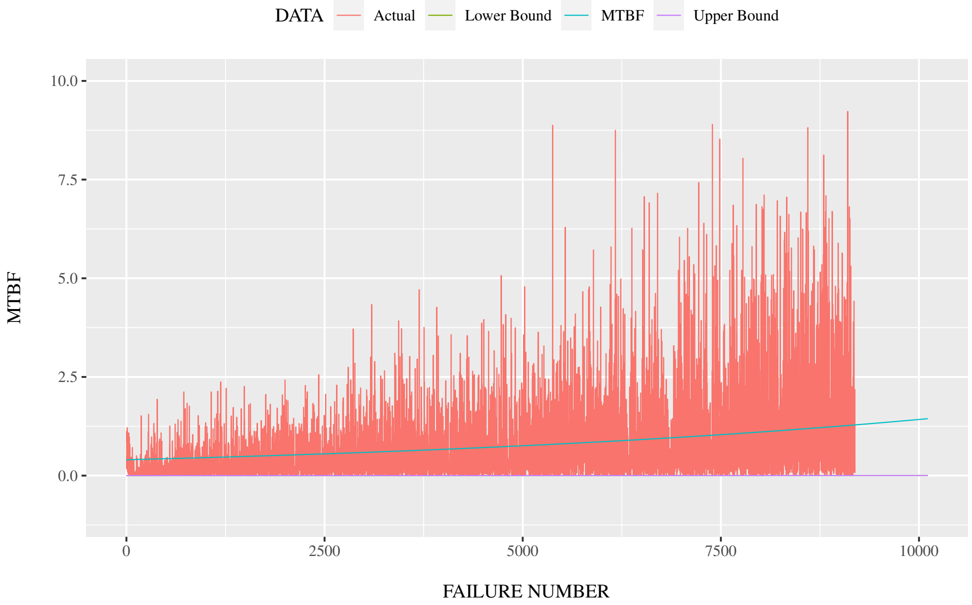

Figure 9. The estimated upper and lower confidence limits based on the conventional method in the case of the Moranda model.

As our main contribution, we have proposed the estimation method for upper and lower confidence limits based on the typical hazard rate model. The conventional models cannot estimate the upper and lower confidence limits because the degree of freedom is very large. The proposed method can estimate the upper and lower confidence limits based on the typical hazard rate model in case of large scale fault data sets by using our method.

6. Conclusions

We have discussed the quantification method of the first type for the fault recorded on the bug tracking system of OSS. Then, we apply the multiple regression analysis. We have found that the proposed method can assess the important factors in terms of the OSS quality control by using the multiple regression analysis.

It is difficult for the OSS developers to assess from the bug tracking system because the fault big data are large scale. The proposed method is simple structure by using the traditional stepwise selection method. Therefore, our method can be simply use for the other OSS. The proposed method can find the main factors as explanatory variables affecting the quality management. Thereby, the OSS developer will be able to easily assess the quality from the standpoint of the condition recorded from actual fault big data.

In particular, we have applied the estimation results of multiple regression analysis to the reliability assessment. Under the situation of big data, the objective variable will depend on various explanatory variables. We have proposed the reliability assessment method based on the multiple regression analysis and stochastic model for the OSS fault big data. As the study, the OSS managers can assess the upper and lower limits of the software reliability models for the fault big data. Thereby, the OSS managers can comprehend the stability of OSS development and operation.

Author Contributions: Conceptualization, Y.T.; Methodology, Y.T.; Validation, Y.T. and S.Y.; Formal Analysis, Y.T.; Data Curation, Y.T.; Visualization, Y.T.; Project Administration, Y.T. and S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the JSPS KAKENHI Grant No.20K11799 in Japan. Thank JSPS KAKEN HI Grant No. 20K11799 in Japan for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yamada, S. *Software Reliability Modeling: Fundamentals and Applications*; Springer: Tokyo, Japan; Heidelberg, Germany, 2014.
2. Kapur, P.K.; Pham, H.; Gupta, A.; Jha, P.C. *Software Reliability Assessment with OR Applications*; Springer: London, UK, 2011.
3. Yamada, S.; Tamura, Y. *OSS Reliability Measurement and Assessment*; Springer International Publishing: Basel, Switzerland, 2016.
4. Zhou, Y.; Davis, J. Open source software reliability model: An empirical approach. In Proceedings of the Fifth Workshop on Open Source Software Engineering (5-WOSSE), St Louis, MO, USA, 17 May 2005; pp. 1–6. [CrossRef]
5. Norris, J. Mission-critical development with open source software. *IEEE Softw. Mag.* **2004**, *21*, 42–49. [CrossRef]
6. Janczarek, P.; Sosnowski, J. Investigating software testing and maintenance reports: Case study. *Inf. Softw. Technol.* **2015**, *58*, 272–288. [CrossRef]
7. Li, Q.; Pham, H. A Generalized Software Reliability Growth Model With Consideration of the Uncertainty of Operating Environments. *IEEE Access* **2019**, *7*, 84253–84267. [CrossRef]
8. Tariq, I.; Maqsood, T.B.; Hayat, B.; Hameed, K.; Nasir, M.; Jahangir, M. The comprehensive study on software reliability. In Proceedings of the 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 3–4 March 2018; pp. 1–7.
9. Korpalski, M.; Sosnowski, J. Correlating software metrics with software defects. In Proceedings of the Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments, Wilga, Poland, 27 May–5 June 2018. [CrossRef]
10. Madeyski, L.; Jureczko, M. Which process metrics can significantly improve defect prediction models? An empirical study. *Softw. Qual. J.* **2015**, *23*, 393–422. [CrossRef]
11. Park, N.J.; George, K.M.; Park, N. A multiple regression model for trend change prediction. In Proceedings of the 2010 International Conference on Financial Theory and Engineering, Dubai, UAE, 18–20 June 2010; pp. 22–26. [CrossRef]
12. Aiyin, W.; Yanmei, X. Multiple Linear Regression Analysis of Real Estate Price. In Proceedings of the 2018 International Conference on Robots & Intelligent System (ICRIS), Changsha, China, 26–27 May 2018; pp. 564–568. [CrossRef]
13. Rahil, A.; Mbarek, N.; Togni, O.; Atieh, M.; Fouladkar, A. Statistical learning and multiple linear regression model for network selection using MIH. In Proceedings of the Third International Conference on e-Technologies and Networks for Development (ICeND2014), Beirut, Lebanon, 29 April–1 May 2014; pp. 189–194. [CrossRef]
14. Singh, V.B.; Sharma, M.; Pham, H. Entropy based software reliability analysis of multi-version open source software. *IEEE Trans. Softw. Eng.* **2017**. [CrossRef]
15. Lavazza, L.; Morasca, S.; Taibi, D.; Tosi, D. An empirical investigation of perceived reliability of open source Java programs. In Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC ’12), Trento, Italy, 26–30 March 2012; pp. 1109–1114. [CrossRef]
16. The Apache Software Foundation, The Apache HTTP Server Project. Available online: http://httpd.apache.org/ (accessed on 14 October 2020).
17. Tamura, Y.; Yamada, S. Dependability analysis tool based on multi-dimensional stochastic noisy model for cloud computing with big data. *Int. J. Math. Eng. Manag. Sci.* **2017**, *2*, 273–287. [CrossRef]
18. Tamura, Y.; Yamada, S. Open source software cost analysis with fault severity levels based on stochastic differential equation models. *J. Life Cycle Reliab. Saf. Eng.* **2017**, *6*, 31–35. [CrossRef]

19. Tamura, Y.; Yamada, S. Dependability analysis tool considering the optimal data partitioning in a mobile cloud. In *Reliability Modeling with Computer and Maintenance Applications*; World Scientific: Singapore, 2017; pp. 45–60.
20. Tamura, Y.; Yamada, S. Multi-dimensional software tool for OSS project management considering cloud with big data. *Int. J. Reliab. Qual. Saf. Eng.* **2018**, *25*, 1850014-1–1850014-16. [CrossRef]
21. Tamura, Y.; Yamada, S. Maintenance effort management based on double jump diffusion model for OSS project. *Ann. Oper. Res.* **2019**, 1–16. [CrossRef]
22. Jelinski, Z.; Moranda, P.B. *Software Reliability Research, in Statistical Computer Performance Evaluation*; Freiberger, W., Ed.; Academic Press: New York, NY, USA, 1972; pp. 465–484.
23. Yin, L.; Trivedi, K.S. Confidence interval estimation of NHPP-based software reliability models. In Proceedings of the 10th International Symposium on Software Reliability Engineering (Cat. No.PR00443), Boca Raton, FL, USA, 1–4 November 1999; pp. 6–11. [CrossRef]
24. Okamura, H.; Grottke, M.; Dohi, T.; Trivedi, K.S. Variational Bayesian Approach for Interval Estimation of NHPP-Based Software Reliability Models. In Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07), Edinburgh, UK, 25–28 June 2007; pp. 698–707. [CrossRef]

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.