A Frame Work for Business Defect Predictions in Mobiles

N.Gayatri[1] S.Nickolas[2] A.V.Reddy[3]
Research Scholar[1], Associate Professor[2], Professor[3]
Department of Computer Applications, NIT Tiruchirappalli

ABSTRACT

Software quality is very highly inflammable area where enormous research has been done and still being continued for effective and exclusive product outcome. Identification of defects or issues in the early phase makes the system to rectify the issues earlier and helps to manage better product outcome within given budget and time which is ultimate goal of every company or project. The issues occurred while coding can be identified by the tools already developed. But if the issues occurred after delivery of the product cannot be identified before and these are called business defects, i.e., while using the system defects occurs. This paper proposes a frame work for identifying the business defects and prediction for mobiles. The data used here is from Android mobiles. This frame work identifies the information, warnings, errors occurred while using the mobile. And also it gives the percentage of faulty status of the product if these errors occur regularly, so these issues can be rectified and corrected in the future developments of the product. Identification of business defects for mobile logs is the novelty of the paper.

Keywords

Mobile logs, threshold, defect prediction, business defects

1. INTRODUCTION

Software defect prediction is one of the software quality assurance activities. The timely prediction of the defects will help managers or developers to achieve the high user satisfaction, by correcting them before actual delivery of the product. Defect prediction will be helpful in all the fields like software, hardware, middleware etc. Software defects can be predicted using the previously available data and many techniques have been developed for predicting defects in software like statistical techniques, regression techniques, classification etc. Still there is need of some techniques because all techniques are algorithm dependent. Mobiles have become very essential in day to day life communication. Without those one cannot explore the world for many things like business, health, education etc. So communication through mobiles plays an important role in today's world. And today, many companies have been established for manufacturing different types of mobiles, which offers many great comforts to the user. Also there is chance of occurrences of defects while developing the mobiles. The software defects can be identified and rectified by many methods available for defect prediction. But especially for the mobiles there is a system which records the issues after the product is delivered. This records all the warning messages, error issues like as the user is speaking suddenly if display is off, suddenly phone becomes shutdown, if it is touch screen there might not be any sense even if someone touch. For these kinds of defects, the developer cannot predict before hand, but whenever error occurs that log is recorded with the string error attached. These are not traditional defects because they might or might not happen. These kinds of defects are known as business defects, which occur after the delivery and at the time of usage of the product. Due to this kind of defects the user may become unsatisfactory and due to these the production level may go down. There is a need to identify these kinds of defects beforehand in order to achieve high user satisfaction and also with the other competitors in terms of time and money which are very important through life.

This paper presents a frame work to identify the business defects from the recorded file and also predicts the health of the device i.e., device may be faulty or not based on input data. A tool is developed to identify these defects based on input data which includes the defects from the recorded files also. and to predict the defects based on rules given and also assess the health of the device. Prediction of defects helps to manage resource allocation when similar log files will be used for development of new versions at the design phase itself. Health of the device tells about the prediction accuracy. If accuracy is high then all the needed will be done before itself and there will be no chance of occurrence of these errors again. The rules are based on the standard logs and the on the average of the each type of error and number of times the error occurred. So software defect prediction can also be used for mobile applications also.

Organization of the work:

This paper is organized as follows: section 1 gives the introduction about defect prediction and mobile devices. Section 2 gives the related work followed by the proposed work in the section 3 and section 4 deals with the experimental results and discursion and finally conclusions are presented.

2. RELATED WORK

Identification of error prone modules is a challenging as far as complex software systems are considered. Prediction of defects as early as possible based on previous data allows one to modify the design and allows the correct resource utilization. Many classification approaches has been developed [1] [2] [3] [4] for prediction of fault prone or non fault prone modules which are represented by a set of software metrics or code attributes [5]. Many software metrics have been developed for prediction of software defects. These software metrics are available within the software repository which is publically available [6]. These early metrics are based on size [7] and complexity [8].Some studies explored regression models [9] and neural networks [10]. Some have argued about the measurement of predictive performance [11] and confirmed that ROC (Receiver operating curves) is the best measure for accurate predictive performance.

Ostrand, Weyuker and Bell suggests the use of binomial regression model that exploits the knowledge of past releases of software project regarding faults to predict the number of defects in the next release of the project.

Several studies compared different methods such as regression techniques and classification techniques [12] .However most accurate method varied according to the context of the study. Principal Component Analysis, Discriminant Analysis layered neural networks and Holographic Networks are applied in Lanuabile et al Regression via Classification has been proposed for modeling uncertainty for software defect prediction [13]. Soft Computing techniques such as Genetic algorithm, Fuzzy logic [14] also have been used for defect prediction.

The main idea behind the prediction model is to provide a clue of which tools, languages and application domains are connected with existence of faults and prepare the ground of the appropriate managerial decisions. Fenton and Neil [15] clarified that all of the problems described cannot be solved easily , however modeling the complexities of the software development using new probabilistic techniques present a positive way forward.

In this paper a prediction model is developed based on the threshold of the input data. Rules are formed based on this threshold and the application used here is the mobile applications. The threshold based is taken because the errors are distributed uncertainly over the log files and with the threshold the least occurrence and the highest. occurrence of defects can be checked. The issues of the mobile logs are recorded and these are used for prediction of the possible defects for the new developing versions. The mobile data is collected from Motorola Company. This model gives the percentage of prediction i.e., how much % the device is healthy and how much% it is faulty based on the rules. It is observed that single rule cannot cover all the situations where as if more number of rules are formed than it covers many situations and prediction accuracy will be high. The performance measure used here is accuracy and it is included within the code itself. This model is exclusively used for mobile defect predictions.

3. PROPOSED WORK

This paper develops tool for monitoring the recorded issues in the mobile devices. Before this tool has been developed all the issues has to be monitored and checked manually only and it was a tedious task for the programmers or developers to monitor and check them manually. Now this tool will give information about the number of errors occurred, number of warnings and any information if, once the recorded data is given as the input. To develop this tool a C++ library called QT is used GUI. QT(cue - tee) is a cross platform frame work that is widely used for developing applications software with a graphical user interface and also used for developing a non GUI programs and console for servers. QT works on android platforms. Android is a Linux based operating system used especially for touch screen mobile devices like smart phones and tablets. The below figure gives the frame work or methodology of the prediction model. In this paper, a tool is developed in which, errors are found and a diagram called pie chart is shown for showing the distribution of errors, information and warnings from the given input.

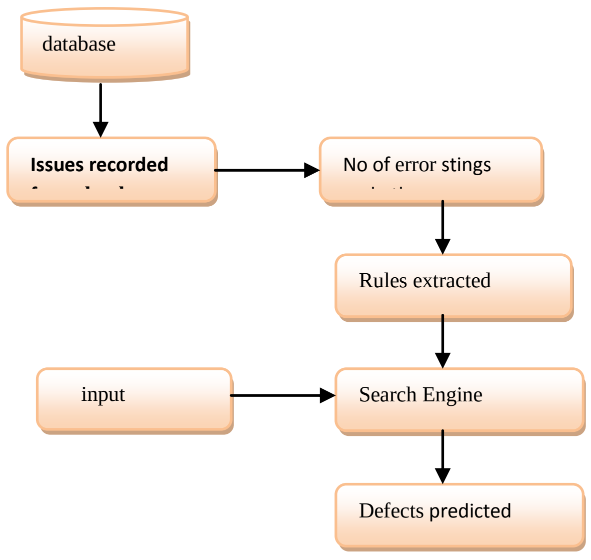


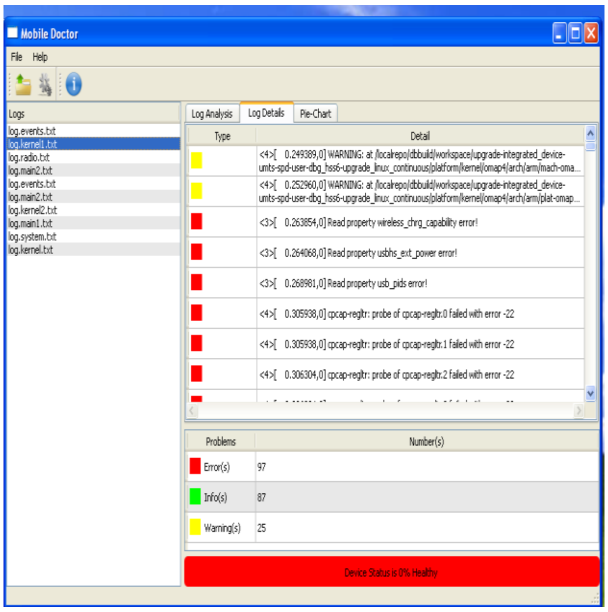
Fig 1.Frame work for mobile defect prediction

The dataset can be in the form of text also. The input will be in the form of logs recorded. These logs contain date on which log has been recorded, some reference number, port number, some text and the information, or error or warning and attached with the message.

Input data will be in the following form.

```
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932] Error in
src/config/omx_config/params/omx_params_custom.c,
line:252
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932]
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932] [ERR=181]
src/config/omx_sup/cam_super_cfg_plugin.c:[104]:
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932] Error in src/config/omx_sup/cam_super_cfg_plugin.c,
line:104
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932]
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932] [ERR=182]
src/config/omx_sup/cam_super_cfg_plugin.c:[108]:[Error]
Get Index: 0x7f00005b
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.932] [ERR=183] src/conf_mng.c:[159]:
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.933] Error in src/conf_mng.c, line:159
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.933]
02-24 10:18:38.281   186   191 D DUCATISS: [CORE1]: [
3.933] [ERR=184]
omx_rpc/src/omx_rpc_skel.c:[294]:Component returned
error: 0x8000100e
02-24 10:18:38.281    186    191 D DUCATISS: [CORE1]: [
3.936] [ERR=185] src/tools_memory.c:[111]
```

Here errors are being considered to build the new model for prediction. In the input data the error string is separated from all other information based on the key word error. The message attached with the keyword error is considered as error occurred for eg error cam_super_cfg_plugin error and with the keyword information considered as information. All the errors, warnings, information are separated using respective keywords. The errors may be of different types like cam_super_cfg_plugin, src/sensor_detect error, Component returned error: 0x8000100e, kernel error etc. These different kinds of error have been considered for building prediction model. The tool developed will give information about how many errors occurred , the frequency of particular error in particular log file , the number of information strings and number of warnings for a given input log file. The working of the tool is presented below by taking the example.



log.events.txt
log.kernel1.txt
log.radio.txt
log.main2.txt
log.events.txt
log.main2.txt
log.kernel2.txt
log.main1.txt
log.system.txt
log.kernel.txt

Type	Detail
🟨	<4>[0.249389,0] WARNING: at /localrepo/dbbuild/workspace/upgrade-integrated_device-umts-spd-user-dbg_hss6-upgrade_linux_continuous/platform/kernel/omap4/arch/arm/mach-oma...
🟨	<4>[0.252960,0] WARNING: at /localrepo/dbbuild/workspace/upgrade-integrated_device-umts-spd-user-dbg_hss6-upgrade_linux_continuous/platform/kernel/omap4/arch/arm/plat-omap...
🟥	<3>[0.263854,0] Read property wireless_chrg_capability error!
🟥	<3>[0.264068,0] Read property usbhs_ext_power error!
🟥	<3>[0.268981,0] Read property usb_pids error!
🟥	<4>[0.305938,0] cpcap-regltr: probe of cpcap-regltr.0 failed with error -22
🟥	<4>[0.305938,0] cpcap-regltr: probe of cpcap-regltr.1 failed with error -22
🟥	<4>[0.306304,0] cpcap-regltr: probe of cpcap-regltr.2 failed with error -22

Problems	Number(s)
🟥 Error(s)	97
🟩 Info(s)	87
🟨 Warning(s)	25

Device Status is 0% Healthy

Fig 2:Screen shot for description of log details

Let us consider, for example some errors present in the input data are cam_super_cfg_plugin. src/sensor_detect error, Component returned error: 0x8000100e etc. Consider these three errors are occurred indifferent log files. Now based on these three rules to be formed which will be used for prediction. For the rule formation the method goes like this: Initially all the errors are separated and are placed in a file named E.

Let us represent E as error file which consist of different errors like e_1,e_2,e_3……en where e_1- cam_super_cfg_plugin,e_2- src/sensor_detect error, e_3- Component returned error: 0x8000100e and en- some other different error.
So E= <e_1,e_2,e_3>

In general this can be denoted as E={$e_1,e_{,2},e_3,e_4$………en}where n denotes the number of errors.

There may be many log files in which each of them may contain different errors.

Let it be represented as L_1, L_2, L_3--------------L_n

Now logfile 1 i.e., L1 has different errors like e_1,e_2,e_3 etc.

i.e, L_1⟶ <$e_1,e_2,e_3,e_2,e_1,e_1,e_3$,----------$e_n$>

logfile L2 also may have same errors or different errors L2⟶<e1,e2,e4,e6,e1,e2,e3,e3,e4,e1--------en>

So in general from this it can be understood that a logfile contains different types of errors and each error occurring many times, but without any specific pattern. The same error may occur at any place in the log file.

For the rule formation, frequency of each error occurring inside the each log file has to be found. Then all the same error from all the log files should be summed and total occurrences has to be found out. From that total , a rule is formed based on the threshold . Threshold is formed from the formula Min(avg(r)/2 , max(r)/3).

Now to represent mathematically, one can consider by taking example

Let us consider there are 3 log files and in each log file there are three errors. So let E={e1,e2,e3} where e1-cam_super_cfg_plugin error, e2- src/sensor_detect error, e3-Component returned error: 0x8000100e. And there are three log files present say L1,L2,L3.

Therefore L⟶<L1,L2,L3>

E⟶<e1,e2,e3>

Now log file L1 = {e11,e12,e13,e21,e31,e14,e22,e32----------enm}

i.e., e11⟶ represent 1st error i.e. e1 in log file 1

e12⟶ represents 1st error i.e., in log file 1,whuch has occurred for 2nd time

e13⟶ represents 1st error i.e., in log file l1 occurred for 3rd time

e21⟶ represents 2nd error i.e., e2 occurred for first time in L1

e22⟶ represents e2 occurring for 2nd time

.

.

.

.

In general it can be represented as $Xjk = \sum_{i=1}^{n} eij$; 1<j<n
n varies according to number of errors in repective logs.

Here i- represents the number of occurrence of error e1 i.e first time and j represents the first error i.e e1

So now using the above representation one finds the frequency of each error occurred in each logfile and store them in variable called X for further usage.

So $fen=Xjk=\sum_{i=1}^{n} eij$ where 1<j<n where n varies according to no of errors in respecive logs.

In Xjk ---- j represents the error type i.e.,e1 or e2-----------

k represents log file number i.e., 1, 2--------------- and also k increases only after the completion of j i.e., only after the completion of finding the frequency of each error in each log file.

So from the above discursion, countof frequencies of errors occurred for each log file are ready.

Next it is needed to find out the total count of each error by considering all log files i.e.,

T_{ce1} = $f_{e11}+f_{e12}+f_{e13}$-----

Where Tc represents total count

e_1 represents the error 1

f_{e11} – represents frequency of e1 in logfile1

f_{e12} – frequency of e1 in log file 2

f_{e13} – frequency of e1 in log file 3

Similarly e_{21} – frequency of e2 in log file 2

e_{22} – frequency of e2 in log file 2

e_{23} - frequency of e2 in log file 3

.

.

.

$e_{nm} \rightarrow f_{en}$

$\sum_{n=1}^{10} Tcen = \sum_{k=1}^{l} fenk$ $1<n<$ number of errors,Here Tc_{en} is the total count of e_n , f_{enk} represents the frequency of error 1 1 to l logfiles. K represents the log file number.

n gets incremented only after k completes

Now one need to find out the threshold to form the rules. The Threshold may be found out by using the formula

$$\mathrm{Min}(\mathrm{avg}(e_1)/2 \ , \ \max(X_{jk})/3)$$

For finding average the general formula can be given as $(\mathrm{Avg}(T_{cen)/}\mathrm{logfile}(l_n))/2$.

So after finding the average of each error and maximum occurrence of error in particular file, one finds a threshold based on the above formula. From this one can get threshold and rules formed according to for each input data and used for prediction for future purpose and also it tells how much percentage the device is faulty or healthy. Healthy or faulty is that about the prediction accuracy.
The algorithmic approach is given below

```
Algorithm

1 .Take different log files like $lf_1$, $lf_2$,…..$lf_n$

2. Error strings are identified based on the keyword error like $e_1$, $e_2$,$e_3$……..$e_n$
for each log file

3 .Find the frequency of each error occurred from the log file.

Frequency can be calculated as $f_{en}=X_{jk}=\sum_{i=1}^{n} eij$ where i= no of time e
occurs in one log file and m is some fixed value.

4.Repeat the steps for all log files given to form the rules

5   Then rules are formed as below

      .$r_1$= Min(avg(Tce1)/2 or max(r1)/3)

       $r_2$ = Min(avg(Tce2)/2 or max(r2)/3)

        $R_n$ = Min(avg($f_n$)/3 or max(rn)/3)

6. For new input data if the en <=Threshold then prediction is high and status
of the device is healthy else prediction is low and the status of the device is
faulty.

7. The percentage of faulty or healthy are also given
```

If the prediction accuracy is high the device is healthy else not. Based on this , the prediction for unseen issues can be done and accordingly the changes and modifications may be done.

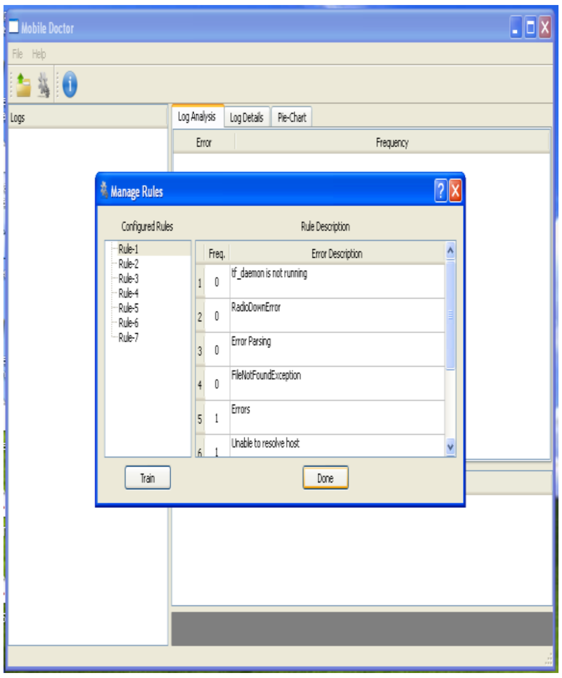
The use and advantages of this frame work is

1. Before developing this tool the recorded issues have been checked manually and now this tool gives the number of errors occurred as soon as the input is given and it has become easy for developers to find the errors occurred.
2. For the newly developed versions if the same design with slight modification is used then prediction helps an the developers can guess these many errors may occurs and accordingly resource allocation and testing can e done.

4. EXPERIMENTAL ANALYSIS

The tool has been developed for the analysis of logs , for finding the errors and information from the input data. The rules are also developed using the standard files within the tool. The screen in the tool looks like this Some Mobile logs are taken and experimented to check the working of the tool .These mobile logs are given from Motorola company related to android software. According to the given input a pie chart is given and the also the health of the mobile device is also checked. The accuracy of the device is checked based on the input strings. In the tool, the topmost bar consists of the options called File and Help. In the file options can be used to give the rules. The number of files given, that many number of rules will be formed. From all these rules one can come up with single rule based on the formula which will cover all the situations for predictions. In the tool manage rules option is present which allows to give the input for generation of rules , and in the left side frequency of the errors and rule description options are present which tells about the rules occurred and their frequency. Then tool has option called train which when pressed trains according to the formula and a single rule is formed and trained which will predict the unseen test data. The screen shot for generation of rules will look s like the below(fFig3). The right side of the screen consists of Log Analysis, Log Details and Pie-chart options where Log Analysis will analyze the number of errors, information and warnings and their frequencies(shown in fig 4). Log Details give the text logs where the errors or warnings are found. Pie chart gives the pictorial view of the Log Analysis (Fig 5). Now at the left column there is option given for loading the test data. File open loads the test data. As soon as the test data is given as the input and on the left side the Log Analysis option if pressed gives the details of the logs, i.e., the number of errors occurred and their frequencies,

Pie chart gives the pictorial view for distribution of errors, information and warnings of that particular file.
Likewise for every input file to be tested these information are given. Also according to the trained rules, at the down a status bar appears which shows the % faulty or healthy of the device if that particular file is used. This percentage depends on the rules given. Also down of the screen before % status bar a summarization is shown for each log file which gives the details of the number of errors occurred and etc. The screen shots are shown for some log file where the status of the same is shown.



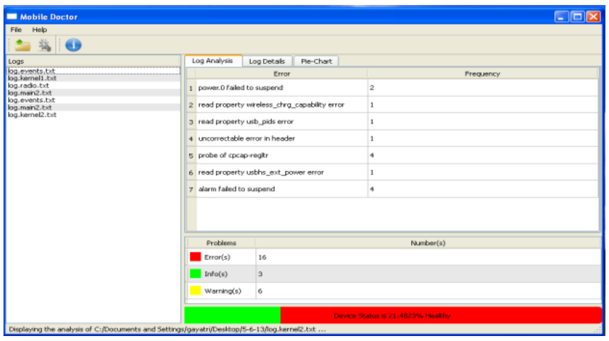
	Freq.	Error Description
1	0	tf_daemon is not running
2	0	RadioDownError
3	0	Error Parsing
4	0	FileNotFoundException
5	1	Errors
6	1	Unable to resolve host

Fig3.Screen shot for generation of rules

Below two steps are given briefly and accordingly the status of the device is decided.

Settings menu

– Hot train button and load the reference faulty device log file.
  - Rule-set will be generated based on the configured error strings and its frequency of occurrence.
– Load a device log file to be parsed.
  - A analysis view will display the list of rules and its frequency of occurrence by processing the logs and validating across the configured rule-set. If the occurrence count of a error string is more than the threshold (rule-set error occurrence), then the device is flagged as faulty



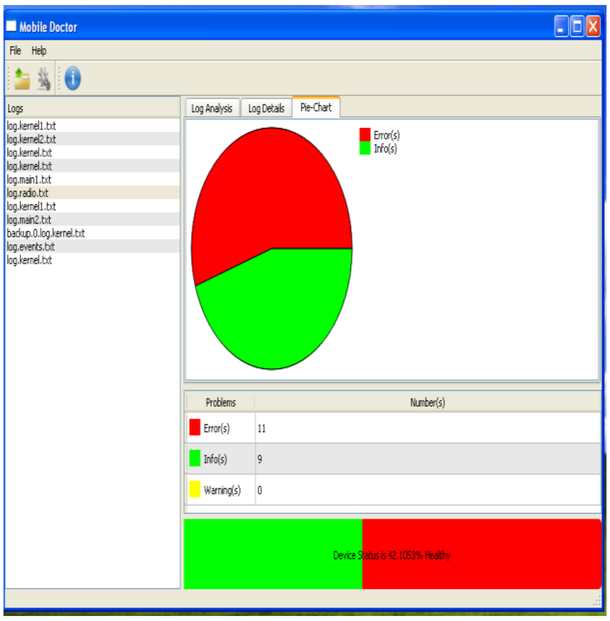
	Error	Frequency
1	power.0 failed to suspend	2
2	read property wireless_chrg_capability error	1
3	read property usb_pids error	1
4	uncorrectable error in header	1
5	probe of cpcap-regltr	4
6	read property usbhs_ext_power error	1
7	alarm failed to suspend	4

Problems	Number(s)
🟥 Error(s)	16
🟩 Info(s)	3
🟨 Warning(s)	6

Device Status is 21.4823% Healthy

Displaying the analysis of C:/Documents and Settings/gayatri/Desktop/5-6-13/log.kernel2.txt ...

Fig 4:Screen shot for description when log file is given as input



Problems	Number(s)
🟥 Error(s)	11
🟩 Info(s)	9
🟨 Warning(s)	0

Device Status is 42.1053% Healthy

Fig5:Pie chart for distribution of errors and information

The graph (Fig 6)shows the percentage predictions for different logfiles.Each logfile is given with the errors, warnings and information.All the logfile may conatin same or different errors with different patterns and different frequencies. According to the rules based on the threshold prediction performnace is seen. It is observed that for most of the log files the prediction accuracy is above 60% and it depends up on the number of rules formed. If less nuber of rules are formed then prediction accuracy may be less and it may not cover all the situations.So training should be done with more number of data and more number of rules should be formed for better prediction accuracy. This work is exclusively can be used for mobile defect prediction.According to the given algorithm rules are generated and the test data has been tested.

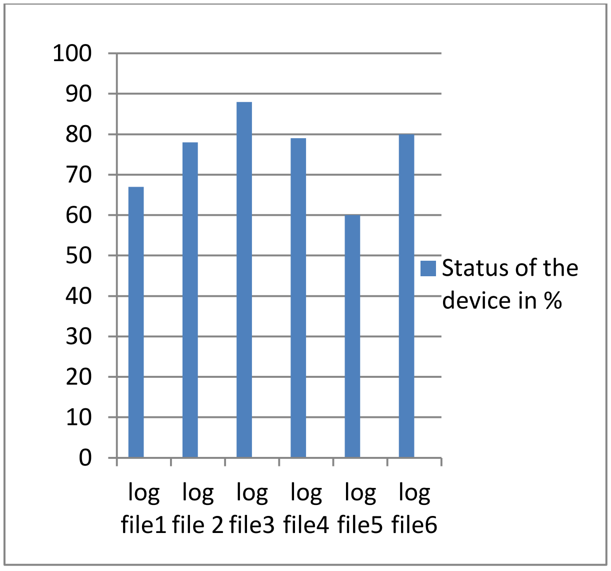


Fig 6:Prediction accuracy for different logfile when more number of rules are given.

5. CONCLUSIONS

Mobiles have become very important nowadays. Prediction of defects in mobile issues may help developers to rectify those defects and helps in development of an improved version which overcome those issues.

This paper proposes a defect prediction model for mobile data obtained from mobile logs based on the threshold of the input data. Here one don't have any prior information about the device as the errors occurred here are not related to traditional coding but errors found during usage of the device called business defects. These are identified using our proposed method .Our method also gave status about up to what extent the device will be faulty or healthy. Good results are obtained when the other mobile logs are fed from the company. So this frame work may be used for business defect prediction of the mobile devices. The advantage of the prediction model is that based on the predicted errors in design phase , the resource allocation is done so that these errors could be modified early before actual outcome of the product.

In future different errors have to be analyzed and understand which errors leads to high defects .Different log files have to be analyzed for generalization of rules.

6. REFERENCES

[1]. K.Ganesan, T.M. Khoshgoftaar, and E.B.Allen, "Case Based Software Quality Prediction," Inter National Journal of Software Engineering and Knowledge Engineering, vol. 10, no.2,pp. 139-152, 2000.

[2]. L Guo,Y. Ma, B. Cukic, and H.Singh,"Robust prediction of fault proneness by Randome Forests", Proceedinds of international sysmposium , Software Reliability Engineering, 2004.

[3]. T.M. Khoshgoftaar, and N Seliya , "Comparative assessment of software quality classification Techniques:An empirical case study", Emperical software engineering, vol.9,no. 3,pp. 229-257,2004.

[4]. Iker Gondra,"Applying machine learning to software fault proneness prediction," the journal of system software , vol. 81,pp. 186-195,2008.

[5]. N.F.Schneidwind,"Methodology for validating software metrics",IEEE Transactions.Software Engineering, vol.18,no. 5,pp.410-422',May 19992.

[6] NASA promise repository http://promise.site.uottawa.ca/SERepository.

[7]. Akiyama, F.,"An example of software system debugging", Information processing, no. 71,pp.353-379,1971

[8]. Halstead ,M. ,"Elements of Software science",Elsiveir,1977.

[9]. P.Bellini,I.Bruno,P. Nesi, and D.Rogai,"Comparing fault proneness estimation models",ICECCS,pp. 205-215,2005.

[10]. Quah, T. S., & Thwin, M. (2004). Prediction of software development faults in PL/SQL files using neural network models.Information and Software Technology, 46(8), 519–523.

[11]. Stefan Lessmann,B Baesens,Christophe Mues and Swantje Pietsch," Benchmarkinh Classification Models for software defect prediction:A proposed Frame work and Novel Findings", IEEE transactions on Software Engineering, Vol 34, No 4 ,pp 485-496 ,July/August (2008)

[12]. Ostrand, T., Weyuker, E., & Bell, R. (2005). Predicting the location and number of faults in large software systems. IEEE Transactions on Software Engineering, 31(4), 340–355.

[13]. Lanubile, F., Lonigro, A., & Visaggio, G. (1995). Comparing models for identifying fault-prone software components. In: Proceedings of the 7th inter international conference on software engineering and knowledge engineering (pp. 312–319), Washington, DC, June.

[14]. Ajeet Kumar Pandey and N. K. Goyal , " Predicting Fault-prone Software Module Using Data Mining Technique and Fuzzy Logic", International Journal of Computer and Communication Technology (Special Issue) Vol. 2, Issue 2-4, pp. 56-63 (2010)

[15]. Fenton, N., & Neil, M. (1999). A critique of software defect prediction models. IEEE Transactions on Software Engineering, 25(5), 675–689