International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 21 (2017) pp. 10546-10550

Efficient analysis of Defects using Naïve Bayes Classification and Impact on Effort and Cost of the Project

Dr. S.Veni
Professor and Head: Department of Computer Science, Karpagam Academy of Higher Education
Karpagam University, Coimbatore, Tamil Nadu, India.

Aparna Srinivasan
Research Scholar: Department of Computer Science, Karpagam Academy of Higher Eduction,
Karpagam University, Coimbatore, Tamil Nadu, India.

Abstract

Research work focuses on impact of defects identified in various stages of Software Development Life Cycle (SDLC) on overall project effort and cost. Naïve Bayes classification technique is used to identify the bug tracking, discovery and maintenance of risky modules. Statistical analysis is performed for analyzing the impact on effort and cost of a project. In this paper, bug tracking is represented as a variable of severity which can be analyzed based on classification and can be displayed high posterior in plot on defect stage in each phase.

Keywords: SLDC Defects, Predict Cost Variance, Naïve Bayes classification, Statistical analysis, Impact, Severity

INTRODUCTION

Naïve Bayes classification technique can be used to identify the defect of project management stage to discover and maintenance of risky modules. In this paper, defect tracking represents as the variable of severity class label which can be analysed based on classification can be displayed high posterior in plot on defect stage in each phases. Also statistical analysis can be performed for analysed impact on effort and cost of a project.

This research work focuses on classifying defects based on phase detected and severity. Effort required for fixing the defects, is predicted using pre-defined values. Cost impact is calculated based on the defect detected Phase. Implementation of probability theory, Naïve Bayes classifier can check the condition rule, decision making deals with probability inference which is used to gather the knowledge of prior events by predicting events through rule base and classifies using learning phase and testing phase.

Data for Research- Defect Collection from various stages of Software Development Life Cycle (SDLC)

In this research work, Defects from Requirements, Design, Build, System Test, User Acceptance Test (UAT) and Implementation stages are listed for analyzing. The defects classified are based on Phase Detected and Phase Attributed which are assigned a weight based on Phase Detected and Severity. The collection of data can be represented as in table.1 and table.2

Table 1: Collection of Requirements and design defects

Defect	Phase Detected	Phase Attributed	Impact	Weight	Severity
Ambiguous Requirements	Requirements	Requirements	10%	1	Minor
Inadequate Requirements	Requirements	Requirements	10%	1	Minor
Incorrect Requirements	Requirements	Requirements	10%	1	Minor
Missing Requirements	Requirements	Requirements	10%	1	Minor
Ambiguous Design	Design Review	Design	10%	1	Minor
Boundary Conditions Neglected in Design	Design Review	Design	10%	1	Minor
Data error	Design Review	Design	10%	1	Minor
Database Error	Design Review	Design	10%	1	Minor
Incorrect Design	Design Review	Design	10%	1	Minor
Inadequate Design	Design Review	Design	10%	1	Minor
Sub optimal Design	Design Review	Design	10%	1	Minor
Message Error	Design Review	Design	10%	1	Minor
Missing Design	Design Review	Design	10%	1	Minor
Test Plan / Cases Error	Design Review	Design	10%	1	Minor
Ambiguous Requirements	Design Review	Requirements	25%	2.5	Moderate
Inadequate Requirements	Design Review	Requirements	25%	2.5	Moderate
Incorrect Requirements	Design Review	Requirements	25%	2.5	Moderate
Missing Requirements	Design Review	Requirements	25%	2.5	Moderate

Table 2: Collection of code review and unit testing defects

Defect	Phase Detected	Phase Attributed	Impact	Weight	Severity
Computational Error	Code Review & Unit Testing	Coding	10%	1	Minor
Boundary Conditions Neglected in Code	Code Review & Unit Testing	Coding	10%	1	Minor
Interface Error	Code Review & Unit Testing	Coding	10%	1	Minor
Logic Error	Code Review & Unit Testing	Coding	10%	1	Moderate
Navigation Error	Code Review & Unit Testing	Coding	10%	1	Minor
Performance Error	Code Review & Unit Testing	Coding	10%	1	Moderate
Sequencing / Timing Error	Code Review & Unit Testing	Coding	10%	1	Moderate
Missing / Inadequate Standards in Code	Code Review & Unit Testing	Coding	10%	1	Minor
Typographical Error	Code Review & Unit Testing	Coding	10%	1	Minor
Variable Declaration Error	Code Review & Unit Testing	Coding	10%	1	Minor
Ambiguous Design	Code Review & Unit Testing	Design	25%	2	Moderate
Boundary Conditions Neglected in Design	Code Review & Unit Testing	Design	25%	2	Moderate
Data error	Code Review & Unit Testing	Design	25%	2	Moderate
Database Error	Code Review & Unit Testing	Design	25%	2	Moderate
Incorrect Design	Code Review & Unit Testing	Design	25%	2	Moderate
Inadequate Design	Code Review & Unit Testing	Design	25%	2	Moderate
Sub optimal Design	Code Review & Unit Testing	Design	25%	2	Moderate
Message Error	Code Review & Unit Testing	Design	25%	2	Moderate
Missing Design	Code Review & Unit Testing	Design	25%	2	Moderate
Test Plan / Cases Error	Code Review & Unit Testing	Design	25%	2	Moderate
Ambiguous Requirements	Code Review & Unit Testing	Requirements	50%	5	Moderate
Inadequate Requirements	Code Review & Unit Testing	Requirements	50%	5	Moderate
Incorrect Requirements	Code Review & Unit Testing	Requirements	50%	5	Moderate
Missing Requirements	Code Review & Unit Testing	Requirements	50%	5	Moderate

Similarly, defects detected in the System Testing, User Acceptance Testing, Implementation and Post Implementation stages are assigned accordingly as Weight and Severity.

METHODOLOGY

Naïve Bayes Classifier

The attributes are conditionally independent to one another by the given class label of the tuple which predicts the data in tuple where X belongs to the class Ci.

$$P(C_i \mid X) > P(C_j \mid X) \quad \text{for } 1 \le j \le m, j \ne i \tag{1}$$

By Bayes' theorem, the classic for which P (Ci |X) represents maximum posterior hypothesis

$$P(C_i \mid X) = P(X \mid C_i)\, P(C_i) / P(X) \tag{2}$$

The probability can easily estimate by $P(x_1|C_i) \times P(x_2|C_i) \times \cdots P(xn|C_i)$ from the training tuples by the relationship as given below,

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) \tag{3}$$

In this paper, to classify the target value of defect on each stage of SDLC by using the rule theorem of equation (1) and found the maximum posteriori of defect in each phase.

An input of numerical variables can be transformed to their categorical counterparts before constructing the frequency tables. In this paper, an implementation is worked out by WEKA tool. Probability of each phase is analyzed based on the severity variables such as Major, Moderate and Minor.

STATISTICAL ANALYSIS

From Table.3, represents the estimation matrix of effort to fix the Minor, Moderate and Major Defects detected in various stages.

Table 3: Estimation Matrix of Effort to fix defects in various stages of SDLC

Defect	Phase Detected	Min Defects	Effort / FP	Change in Effort	Mod Defects	Effort / FP	Change in Effort	Maj Defects	Effort / FP	Change in Effort
Requirements Defe	Requirements	1	4	4	1	8	8	1	16	16
Requirements Defe	Design Review	1	10	10	1	20	20	1	40	40
Requirements Defe	Code Review & Unit Testing	1	20	20	1	40	40	1	80	80
Requirements Defe	System Testing	1	24	24	1	48	48	1	96	96
Requirements Defe	User Acceptance Testing	1	32	32	1	64	64	1	128	128
Requirements Defe	Implementation & Post Implementation	1	40	40	1	80	80	1	160	160
Design Defect	Design Review	1	4	4	1	8	8	1	16	16
Design Defect	Code Review & Unit Testing	1	8	8	1	16	16	1	32	32
Design Defect	System Testing	1	20	20	1	40	40	1	80	80
Design Defect	User Acceptance Testing	1	30	30	1	60	60	1	120	120
Design Defect	Implementation & Post Implementation	1	40	40	1	80	80	1	160	160
Coding Defect	Code Review & Unit Testing	1	4	4	1	8	8	1	16	16
Coding Defect	System Testing	1	10	10	1	20	20	1	40	40
Coding Defect	User Acceptance Testing	1	20	20	1	40	40	1	80	80
Coding Defect	Implementation & Post Implementation	1	30	30	1	60	60	1	120	120
Implementation Def	Implementation & Post Implementation	1	20	20	1	40	40	1	80	80

From fig.1 to fig. 4 represents the effort required to fix the defects based on severity from each Phase of SDLC.

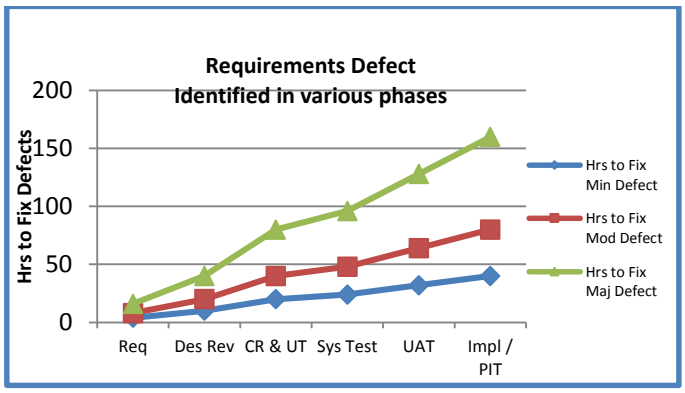

Figure 1: Effort to fix Requirements defects

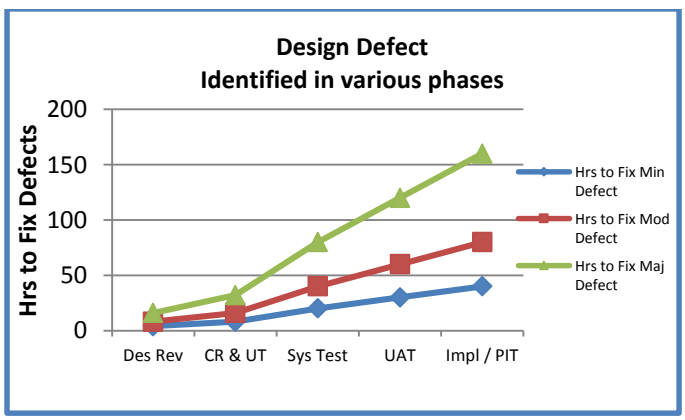

Figure 2: Effort to fix Design defects

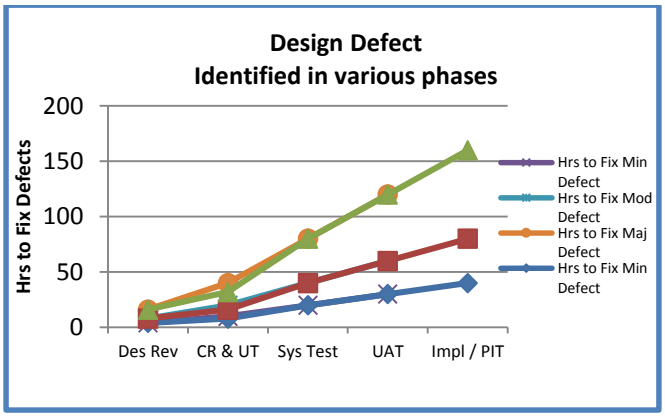

Figure 3: Effort to fix Coding defects

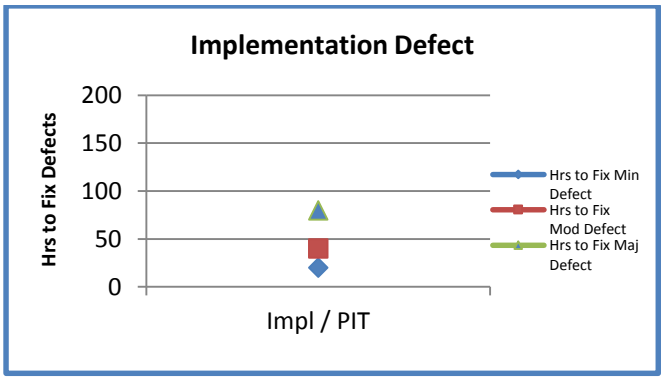

Figure 4: Effort to fix Implementation Defects

RESULT ANALYSIS

Impact of Defects identified in various stages on Cost

Tables 4 and 5 represent the impact of the change in effort and cost can be predicted based on the defects detected in various stages of SDLC cycle.

Table 4: Requirement Defect detected in the requirement stage

Defect	Phase Detected	Min Defects	Effort / FP	Change in Effort	Mod Defects	Effort / FP	Change in Effort	Maj Defects	Effort / FP	Change in Effort	Change in Effort
Requirements Defe	Requirements	4	4	16	2	8	16	1	16	16	**48**
Requirements Defe	Design Review		10	0		20	0		40	0	**0**
Requirements Defe	Code Review & Unit Testing		20	0		40	0		80	0	**0**
Requirements Defe	System Testing		24	0		48	0		96	0	**0**
Requirements Defe	User Acceptance Testing		32	0		64	0		128	0	**0**
Requirements Defe	Implementation & Post Implementation		40	0		80	0		160	0	**0**
Design Defect	Design Review		4	0		8	0		16	0	**0**
Design Defect	Code Review & Unit Testing		8	0		16	0		32	0	**0**
Design Defect	System Testing		20	0		40	0		80	0	**0**
Design Defect	User Acceptance Testing		30	0		60	0		120	0	**0**
Design Defect	Implementation & Post Implementation		40	0		80	0		160	0	**0**
Coding Defect	Code Review & Unit Testing		4	0		8	0		16	0	**0**
Coding Defect	System Testing		10	0		20	0		40	0	**0**
Coding Defect	User Acceptance Testing		20	0		40	0		80	0	**0**
Coding Defect	Implementation & Post Implementation		30	0		60	0		120	0	**0**
Implementation	Implementation & Post Implementation		20	0		40	0		80	0	**0**
							CHANGE IN OVERALL PROJECT EFFORT				**48**
							Offshore Cost Impact				**$ 1,200**

Table 5: Requirement Defects detected in the implementation /post-implementation stages

Defect	Phase Detected	Min Defects	Effort / FP	Change in Effort	Mod Defects	Effort / FP	Change in Effort	Maj Defects	Effort / FP	Change in Effort	Change in Effort
Requirements Defe	Requirements		4	0		8	0		16	0	**0**
Requirements Defe	Design Review		10	0		20	0		40	0	**0**
Requirements Defe	Code Review & Unit Testing		20	0		40	0		80	0	**0**
Requirements Defe	System Testing		24	0		48	0		96	0	**0**
Requirements Defe	User Acceptance Testing		32	0		64	0		128	0	**0**
Requirements Defe	Implementation & Post Implementation	4	40	160	1	80	80	1	160	160	**400**
Design Defect	Design Review		4	0		8	0		16	0	**0**
Design Defect	Code Review & Unit Testing		8	0		16	0		32	0	**0**
Design Defect	System Testing		20	0		40	0		80	0	**0**
Design Defect	User Acceptance Testing		30	0		60	0		120	0	**0**
Design Defect	Implementation & Post Implementation		40	0		80	0		160	0	**0**
Coding Defect	Code Review & Unit Testing		4	0		8	0		16	0	**0**
Coding Defect	System Testing		10	0		20	0		40	0	**0**
Coding Defect	User Acceptance Testing		20	0		40	0		80	0	**0**
Coding Defect	Implementation & Post Implementation		30	0		60	0		120	0	**0**
Implementation	Implementation & Post Implementation		20	0		40	0		80	0	**0**
							CHANGE IN OVERALL PROJECT EFFORT				**400**
							Offshore Cost Impact				**$ 10,000**

With respect to severity, defects are classified as Major, Moderate and Minor. Under this classification, Major severity defects are displayed with high posterior in graphs plotted based on defects, phased detect and phase attributed as shown in the figures 5 to 7.

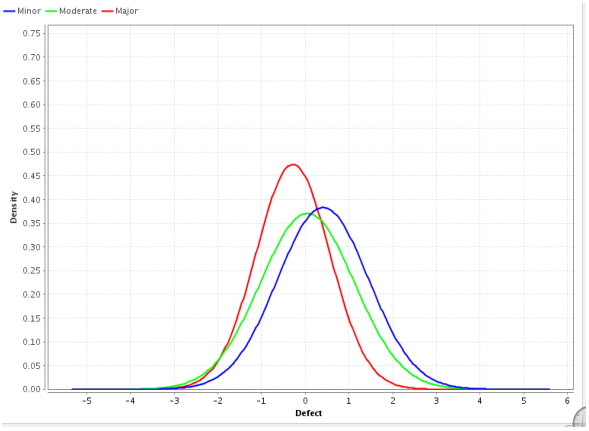

Figure 5: Classifier of defect severity based on the defect

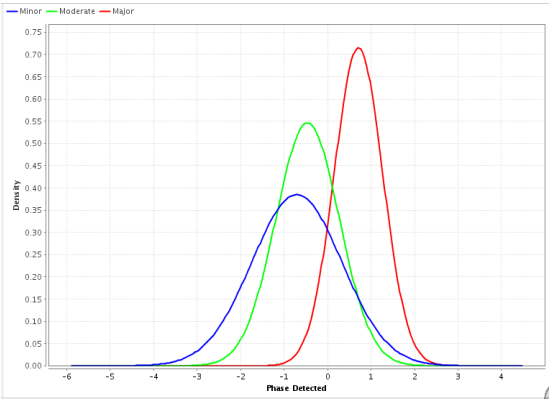

Figure 6: Classifier of defect severity based on phase detected

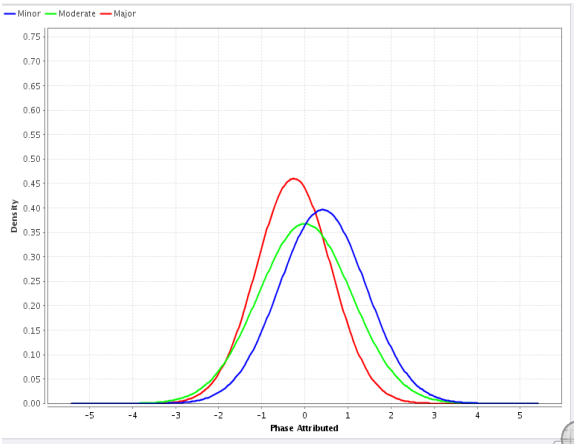

Figure 7: Classifier of defect severity based on phase attributed

Confusion Matrix

a	b	c	Classified as
28	1	1	a = Minor
0	**29**	2	b = Moderate
0	3	**49**	c = Major

From the confusion matrix, 106 instances of data can be classified out of 115 instances. 92.1739% of data can be classified as accurate. 28 instances of defects are minor, 29 instances of defects are moderate and 49 instances of defects are major.

CONCLUSION

In this research, it can be concluded that the impact of defects detected at earlier stages of software development life cycle has less effort, schedule and cost impact at the later stages. With respect to severity, defects can be classified as major, moderate and minor based on the set of predefined attributes. Under the Naïve Bayes classification, major severity defects display high posterior.

REFERENCES

[1] Dr. S.P Gupta "Statistical Methods", Sultan Chand and Sons, 28th Edition-2014, ISBN No. 978-93-5161-028-1.

[2] P.V.Praveen Sundar, "A Comparative Study for Predicting Student's Academic Performance Using Bayesian Network Classifiers", IOSR Journal of Engineering (IOSRJEN) e-ISSN: 2250-3021, p-ISSN: 2278-8719 Vol. 3, Issue 2 (Feb. 2013), ||V1|| PP 37-42.

[3] Dr. N. Geethanjali, M. Surendra Naidu, Optimal Rule Selection Based Defect Classification System using Naive Bayes Classifier, ISSN : 0975-5462, Vol. 5 No.08 August 2013.

[4] WEKA, University of Waikato, New Zealand, http://www.cs.waikato.ac.nz/ml/weka/