Journal of The Korea Society of Computer and Information
Vol. 23 No. 9, pp. 81-86, September 2018

www.ksci.re.kr
https://doi.org/10.9708/jksci.2018.23.09.081

Defect Severity-based Defect Prediction Model using CL

Na-Young Lee*, Ki-Tae Kwon*

Abstract

Software defect severity is very important in projects with limited historical data or new projects. But general software defect prediction is very difficult to collect the label information of the training set and cross-project defect prediction must have a lot of data. In this paper, an unclassified data set with defect severity is clustered according to the distribution ratio. And defect severity-based prediction model is proposed by way of labeling. Proposed model is applied CLAMI in JM1, PC4 with the least ambiguity of defect severity-based NASA dataset. And it is evaluated the value of ACC compared to original data. In this study experiment result, proposed model is improved JM1 0.15 (15%), PC4 0.12(12%) than existing defect severity-based prediction models.

▸Keyword: Software Quality, Software Defect Severity, Label Clustering, Statistical Significance

I. Introduction

소프트웨어 결함 예측은 소프트웨어 품질을 결정하는 데 있어 중요한 요소이다.[1] 그리고 산업 현장에서는 소프트웨어 결함 예측 기술은 소프트웨어 품질을 보증하기 위해 실제로 많이 사용되고 있다.[2] 흔히 적용되는 기술로는 결함 예측 Metric은 변화/코드 Metric 측량, 변화/코드 엔트로피, 밀집도, 개발자의 상호 작용과 같은 다양한 소프트웨어 성과를 수집하여 계산하는 방법이 사용된다. 그러나 이와 같은 기존 결함 예측 방법은 미리 생성한 학습 데이터세트를 활용하여 지도적 기계학습에 의존하고 있기 때문에 학습 데이터세트를 생성할 수 없는 새로운 프로젝트에 적용하기는 어렵다. 그래서 이러한 지도적 기계학습의 제한점 때문에 연구자들은 제한된 데이터를 가지고 결함 예측을 위한 다양한 방법으로 준지도학습, 비지도학습, 강화학습 등을 적용한 모델을 제안한다. 최근에 보편적인 결함 예측 모델은 비슷한 유형의 프로젝트를 사용하여 다양한 학습기법으로 결함을 분류하고 교차검증을 통해 결함을 예측한다.[3] 그러나 분류되지 않은 데이터세트에서 결함 예측을 위한 기존 방법은 전문가의 노력이 필요하고 원본 데이터세트의 상이한 분포가 결함 예측 결과에 영향을 미친다. 이에 본 논문에서는 이러한 제한점을 보완한 클러스터링 라벨(CL)을 이용한 결함심각도 기반 결함 예측 모델을 제안한다. 이 모델은 결함심각도를 분류하는 기존 모델인 앙상블 모델과는 달리 CLA(Clustering, Labeling)와 CLAMI(Clustering, Labeling, Metric Selection, Instance Selection)를 적용해서 유사 데이터세트 없이 결함이 분류되어 있지 않은 원본 데이터세트만을 가지고 Metric의 값에 따라 Column을 자동으로 분류하고 최적의 훈련 데이터세트를 생성함으로써 결함 예측 정확도를 높일 수 있다. 그래서 본 논문에서는 이 모델의 성능을 증명하기 위해 기존 결함심각도 기반 연구에서 활용되었던 NASA의 데이터세트 중에 모호성이 가장 낮은 JM1, PC4에 CLAMI를 적용하여 5개의 그룹으로 클러스터링한다. 그리고 결함심각도에 기반한 클러스터링 평균값에 따라 Buggy와 Clean으로 라벨링하고 최소 MVS(Metric Violation Scores)값에 따라 생성된 최종 훈련 데이터세트와 기존 결함심각도에 기반하지 않은 결함 예측 모델과 ACC(Accuracy)의 값을 비교 분석한다.

본 논문의 구성은 2장 관련 연구, 3장 제안된 모델에 대한 설명, 4장 실험결과, 5장 결론 및 향후 과제로 구성되어 있다.

• First Author: Na-Young Lee, Corresponding Author: Ki-Tae Kwon
*Na-Young Lee (nylee@gwnu.ac.kr), Dept. of Computer Science and Engnieering, Gangneung-Wonju National University
*Ki-Tae Kwon (ktkwon@gwnu.ac.kr), Dept. of Computer Science and Engnieering, Gangneung-Wonju National University
• Received: 2018. 07. 31, Revised: 2018. 08. 20, Accepted: 2018. 08. 26.

II. Preliminaries

1. Existing Defect Prediction Process

소프트웨어 결함은 요구 사항이나 설계 사양에 맞지 않고 심지어는 그것의 변경이나 교체를 요구하는 것을 의미한다. 그래서 이러한 소프트웨어 결함은 프로젝트의 성공과 실패를 좌우하는 요소가 된다. 그리고 소프트웨어 결함은 결함심각도에 따라 Blocking, Critical, Major, Minor, Inconsequential의 5단계로 다음과 같이 나누고 있다.[4]

- Blocking : 테스트가 어렵거나 적절한 해결책이 없는 경우
- Critical : 필수 작업의 중단 및 안전, 보안이 위험한 경우
- Major : 필수 작업에게는 영향을 주지만 진행 가능한 경우
- Minor : 필수 작업이 아닌 작업이 중단되는 경우
- Inconsequential : 운영에 영향을 주지 않는 경우

기존 결함 예측 프로세스는 지도적 기계학습에 의해 데이터세트의 특징을 찾아 결함의 유무만을 분류한다.[1] 그리고 결함심각도는 전문가에 의해 분류하기 때문에 객관화가 어렵다.[5] 이에 그것을 보완하기 위해 기계학습에서 사용되는 전처리 기술을 이용하지만 전처리 기술은 데이터세트의 분포에 따라 결함 예측이 상이한 결과가 나오거나 이상치가 존재한다. 그래서 결함 예측 연구에서는 학습알고리즘으로 두 변수 사이의 관계를 통한 회귀 분석, 의사 결정 규칙에 따른 의사결정 트리, 확률을 통한 나이브베이즈, 의사결정트리의 확장인 랜덤포레스트 등을 사용하여 보다 나은 훈련 데이터세트를 생성하고 실험했다.[6] 그리고 결함심각도를 기반한 연구에서는 결함심각도 정도에 따른 영향을 분석하고 결함심각도 자동 평가 시스템도 연구되었다.[5][7] 또한 결함심각도에 기반한 소프트웨어 품질 예측 연구에서는 공개되는 데이터세트 중에서 결함심각도가 있고 모호성이 적은 NASA의 JM1, PC4 데이터세트를 이용하여 모델을 제시하고 평가하였다.[8]

2. CLAMI Method

CLAMI 방법은 그림 1과 같이 분류되지 않은 원본 데이터세트에 클러스터링과 라벨링을 통한 CLA방법을 적용해서 Metric의 값을 이용하여 최종 훈련 데이터세트를 생성하는 방법이다.[3]

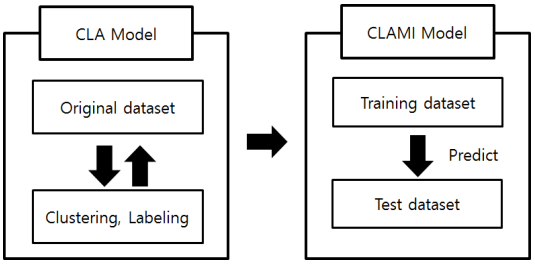

Fig. 1. The Overview of CLA and CLAMI

2.1 Clustering Instances

Column명을 Instances x1, x2, ..., xn로 각 레코드를 Instances a1, a2,.....,an 라고 하면 각 필드에 해당하는 레코드 값들의 중간값(Median)을 계산한다. 이때 중간값은 주어진 각 레코드 n개에 대해 n이 홀수일 경우는 (n+1)/2번째의 값, n이 짝수일 경우는 n/2번째 또는 (n/2)+1번째 값을 의미한다. 이렇게 구해진 값이 중간값보다 큰 경우에 표1과 같이 색깔로 표시한다.

Table 1. Clustering Result

Inst.	x1	x2	x3	x4	x5	x6	x7
a1	***3***	1	***3***	0	5	1	***9***
a2	1	1	2	0	***7***	***3***	8
a3	***2***	***3***	2	***5***	5	***2***	1
a4	0	0	***8***	1	0	1	***9***
a5	1	0	2	***5***	***6***	***10***	8
a6	1	***4***	1	1	***7***	1	1
a7	1	0	1	0	0	1	7
Median	1	1	2	1	5	1	8

2.2 Labeling

각 레코드별로 중간값보다 큰 경우에 개수를 세어서 K=0, 1, 2,....,n으로 나타낸 후 K값에 따라 그룹을 나누어 클러스터링한다. 이때 K값은 크면 클수록 Buggy가 나타날 확률이 커진다. 그리고 클러스터링의 기준값인 K값의 평균값을 구한 뒤 평균값보다 이상이면 Buggy, 이하이면 Clean이라 라벨링한다.

Table 2. Labeling Result

Cluster(k)	Inst.	Label
0	a7	Clean
1	-	Clean
2	a2,a4,a6	Clean
3	a1,a5	Buggy
4	a3	Buggy

2.3 Metric Selection

Metric Selection은 가장 최적의 Column을 선택하는 작업이다. 2번째 단계에서 Buggy와 Clean으로 라벨링된 색깔 표시를 Buggy일 경우에는 색깔이 없는 Column으로 색깔을 변경하고 Clean일 경우에는 표시된 Column을 그대로 색깔을 표시한다. 이때, MVS는 식1과 같이 각 Column별로 색깔이 표시된 레코드 수를 전체레코드수로 나눈 값으로 계산한다. 그리고 Metric의 MVS 값 중에서 가장 작은 값을 가진 Column을 선택한다.

$$MVS_i = \frac{C_i}{F_i} \quad (1)$$

(C_i는 색깔이 표시된 레코드의 수, F_i는 전체 레코드의 수)

Table 3. MVS's Result

Inst.	x1	x2	x3	x4	x5	x6	x7
a1	3	***1***	3	***0***	***5***	***1***	9
a2	1	1	2	0	***7***	***3***	8
a3	2	3	***2***	5	***5***	2	***1***
a4	0	0	***8***	1	0	1	***9***
a5	***1***	***0***	***2***	5	6	10	***8***
a6	1	***4***	1	1	***7***	1	1
a7	1	0	1	0	0	1	7
MVS	***1/7***	3/7	3/7	***1/7***	4/7	2/7	3/7

2.4 Instance Selection

Instance Selection은 최소 MVS에 의해 선택된 필드를 가지고 훈련 데이터세트를 생성하는 과정이다. 최소 MVS에 의해 선택된 필드에서 XAND연산을 통해 1인 경우만을 남기고 데이터를 삭제한다. 이렇게 생성되어 남겨진 데이터세트를 최종 훈련 데이터세트로 선정한다.

Table 4. Final Training Dataset's Selection

Inst.	x1	x4	XAND
a1	3	***0***	***0***
a2	1	0	1
a3	2	5	1
a4	0	1	1
a5	***1***	5	***0***
a6	1	1	1
a7	1	0	1

III. The Proposed Scheme

1. Data Set

공개되는 NASA의 13개의 데이터세트 중에서 결함심각도가 있는 데이터세트는 5개이다. 그중에서 결함심각도를 기반한 선행 연구 논문에서 먼저 사용되었고 데이터의 모호성이 가장 적으면서 코드의 유형이 비슷한 JM1, PC4 데이터세트를 선정하였다.[8]

• JM1 : 실시간 C 프로젝트 데이터세트, 315 KLOC, 10,878 module

• PC4 : 지구궤도 위성 비행 소프트웨어 데이터세트, 36 KLOC, 1,458 module

2. Proposed Defect Prediction Model Structure

기존 결함심각도에 기반한 연구에서 사용되었던 NASA의 데이터세트 중 JM1, PC4 데이터세트의 핵심 입력 Column을 선정한다. 그리고 결함심각도가 분류되어 있지 않는 JM1, PC4에 2장에서 설명한 CLAMI 방법을 결함심각도에 기반하여 적용한다. 그 다음에 최종 훈련 데이터세트를 생성하고 제안한 모델과 기존 모델의 성능을 비교 분석한다.

2.1 Applications of Clustering Instances

CFS(Co-relation Based Feature Selection)는 특징추출 기법으로 노이즈가 많고 불필요한 요소를 제거하는 방법이다. 가장 최적의 조합을 찾아내는 CFS방법에 의해 JM1과 PC4의 입력 Column을 8개로 선정한다.[9] 그리고 그 입력 Column에 CLA과정을 결함심각도에 기반하여 적용한다. 이때, 선정된 8개의 입력 Column은 LOC_BLANK, LOC_CODE_AND_COMMENT, LOC_COMMENTS, CYCLOMATIC_COMPLEXITY, DESIGN_COMPLEXITY, ESSENTIAL_COMPLEXITY, HALSTEAD_CONTENT, LOC_TOTAL이다.

2.2 Applications of Labeling

JM1과 PC4의 데이터세트는 입력 Column이 8개이므로 K=0, 1, 2, ..., 8로 클러스터링 되는데 이렇게 클러스터링한 결과값이 평균값보다 이상이면 Buggy, 이하이면 Clean이라 라벨링한다. 이때, 결함심각도에 기반하여 K값에 따라 결함심각도를 고심각(K=4, 5, 6, 7, 8), 저심각(K=2, 3), 비결함(K=0, 1)으로 분류한다.

2.3 Applications of Metric Selection and Instance Selection

Metric Selection을 통해 8개의 Column 중에서 가장 최적의 Column을 선택한다. 이 때 Buggy일 경우 색깔을 반전하고 Clean일 경우에는 그대로 둔다. 그리고 MVS를 계산하여 최소 MVS 값을 가진 Column을 훈련 데이터세트의 Column으로 선택한다. 선택된 Column을 가지고 XAND연산을 통해 데이터를 삭제하고 최종 훈련 데이터세트로 생성한다.

IV. Experiment Results

1. Experiment Results using CL Model

최적의 조합으로 선정된 8개의 Column의 중간값을 계산하면 JM1은 LOC_BLANK=3, LOC_CODE_AND_COMMENT=0, LOC_COMMENTS=0, CYCLOMATIC_COMPLEXITY=4, DESIGN_COMPLEXITY=2, ESSENTIAL_COMPLEXITY=1, HALSTEAD_CONTENT=27.61, LOC_TOTAL=26이였고 PC4는 LOC_BLANK=3, LOC_CODE_AND_COMMENT=0, LOC_COMMENTS=0, CYCLOMATIC_COMPLEXITY=3, DESIGN_COMPLEXITY=2, ESSENTIAL_COMPLEXITY=1, HALSTEAD_CONTENT=19.02, LOC_TOTAL=12였다. 이렇게 계산된 중간값보다 큰 경우에는 그림 2와 그림 3처럼 색깔로 표시하고 각 레코드마다 표시된 색깔의 개수를 세어 K=0, 1, 2, ..., 8로 그 값에 따라 클러스터링하였다. JM1을 클러스터링한 결과는 그림 2와 같고 PC4를 클러스터링한 결과는 그림 3과 같다.

다음으로 클러스터링한 JM1의 평균값은 3.27이고 PC4의

No	LOC_BLANK	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CYCLOMATIC_COMPLEXITY	DESIGN_COMPLEXITY	ESSENTIAL_COMPLEXITY	HALSTEAD_CONTENT	LOC_TOTAL	CLUSTER
1	1	0	0	4	3	1	25.05	14	1
2	5	0	6	19	16	10	81.19	98	7
3	2	0	0	1	1	1	56.44	14	1
4	16	0	0	1	1	1	168.57	70	3
5	0	0	0	4	2	3	14.32	12	1
6	8	0	0	3	3	1	33.85	42	4
7	1	0	0	4	3	4	17.28	8	2
8	12	0	6	13	7	10	45.65	131	7
9	12	0	9	7	7	6	49.5	78	7
10	3	0	19	10	6	6	29.59	56	6

Fig. 2. JM1's Clustering Result

No	LOC_BLANK	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CYCLOMATIC_COMPLEXITY	DESIGN_COMPLEXITY	ESSENTIAL_COMPLEXITY	HALSTEAD_CONTENT	LOC_TOTAL	CLUSTER
1	17	2	8	6	4	1	37.84	24	7
2	2	0	1	5	3	4	12.5	9	4
3	2	1	1	3	2	3	15.99	18	4
4	4	0	0	3	2	1	21.06	10	2
5	7	3	0	3	1	1	21.09	20	4
6	5	0	0	6	5	1	27.56	13	5
7	2	0	0	1	1	1	8.98	3	0
8	2	0	0	3	1	1	12.72	9	0
9	36	18	43	11	3	1	15.89	69	6
10	1	0	0	5	5	1	31.67	18	4

Fig. 3. PC4's Clustering Result

평균값은 3.30이므로 K값이 평균값 이상이면 Buggy, 이하이면 Clean으로 나누어 라벨링하였다. JM1을 라벨링한 결과는 그림 4과 같고 PC4를 라벨링한 결과는 그림 5와 같다.

No	CLUSTER	CLEAN/BUGGY
1	1	CLEAN
2	7	BUGGY
3	1	CLEAN
4	3	CLEAN
5	1	CLEAN
6	4	BUGGY
7	2	CLEAN
8	7	BUGGY
9	7	BUGGY

Fig. 4. JM1's Labeling Result

No	CLUSTER	CLEAN/BUGGY
1	7	BUGGY
2	4	BUGGY
3	4	BUGGY
4	2	CLEAN
5	4	BUGGY
6	5	BUGGY
7	0	CLEAN
8	0	CLEAN
9	6	BUGGY

Fig. 5. PC4's Labeling Result

Metric Selection을 통해 각 Column의 MVS를 식1에 의해 계산하면 표5와 같다. 그리고 표 5에서 값이 가장 작은 두 개의 Column을 훈련 데이터세트의 Column으로 선택했다. 그 결과 JM1은 CYCLOMATIC_COMPLEXITY, LOC_TOTAL로, PC4는 HALSTEAD_CONTENT, LOC_TOTAL로 선택되었다.

Table 5. MVS's Value

Column	JM1	PC4
LOC_BLANK	0.1796	0.1866
LOC_CODE_AND_COMMENT	0.3484	0.1907
LOC_COMMENTS	0.2325	0.1934
CYCLOMATIC_COMPLEXITY	***0.1551***	0.2174
DESIGN_COMPLEXITY	0.1699	0.2572
ESSENTIAL_COMPLEXITY	0.2841	0.3251
HALSTEAD_CONTENT	0.2174	***0.1811***
LOC_TOTAL	***0.0810***	***0.0802***

선택된 Column을 가지고 XAND연산을 통해 데이터를 삭제하고 최종 훈련 데이터세트를 생성하게 되는 데 이때 생성된 훈련 데이터세트는 그림 6, 그림 7과 같다.

No	CYCLOMATIC_COMPLEXITY	LOC_TOTAL	CLUSTER	CLEAN/BUGGY
2	19	98	7	BUGGY
6	3	42	4	BUGGY
8	13	131	7	BUGGY
9	7	78	7	BUGGY
10	10	56	6	BUGGY
11	10	54	7	BUGGY
12	19	113	6	BUGGY
14	6	195	7	BUGGY
16	5	30	4	BUGGY
21	14	94	8	BUGGY
22	5	36	5	BUGGY
25	11	64	7	BUGGY
33	1	58	4	BUGGY
34	13	70	7	BUGGY
37	5	38	5	BUGGY
40	10	41	5	BUGGY
42	14	109	8	BUGGY

Fig. 6. JM1's Final Training Dataset

No	HALSTEAD_CONTENT	LOC_TOTAL	CLUSTER	CLEAN/BUGGY
1	37.84	24	7	BUGGY
2	12.5	9	4	BUGGY
3	15.99	18	4	BUGGY
5	21.09	20	4	BUGGY
6	27.56	13	5	BUGGY
9	15.89	69	6	BUGGY
10	31.67	18	4	BUGGY
13	27.56	22	4	BUGGY
14	13.96	29	7	BUGGY
15	35.24	19	4	BUGGY
20	28.84	18	4	BUGGY
29	29.53	48	5	BUGGY
31	30.33	36	6	BUGGY
32	46.18	30	7	BUGGY
34	14.8	12	5	BUGGY
38	10.03	20	5	BUGGY
40	95.83	47	5	BUGGY
42	27.48	23	7	BUGGY
43	51.72	24	7	BUGGY

Fig. 7. PC4's Final Training Dataset

본 논문에서 제시된 그림 2, 그림 3, 그림 4, 그림 5, 그림 6, 그림 7의 일련의 과정은 R패키지 또는 VBA(Visual Basic for Application)로 작성하여 생성된 결과이다.

2. Performance Comparison with Existing Models

False Negative는 결함이 있는 모듈을 결함이 없는 모듈로 잘못 예측한 비율로 오류의 유형 중에서는 Type II를 의미한다. Type II의 오류는 개발 프로세스에서 Type I의 오류보다 많은 비용과 노력을 요구하기 때문에 기존 모델과 클러스터링 라벨의 성능비교를 위해서 결함심각도 기반 결과표를 이용하여 ACC로 모델을 평가했다. 이 때, 결함심각도에 기반한 혼동행렬(Confusion Matrix) 표 6을 보면 HSF(High Severity Faultprone)는 고심각, LSF(Low Severity Faultprone)는 저심각, NF(Not Faultprone)는 비결함, Predicted Class는 True(Clean)와 False(Buggy)의 분류 모델에서 구한 분류 예측값, Actual Class는 실제 데이터의 분류값을 나타낸다.[10] 그리고 식 2와 같이 실제값과 예측값이 모두 Ture(Clean)인 경우(TP)의 수, 실제값이 True이고 예측값이 False인 경우(FN)의 수, 실제값이 False이고 예측값이 True인 경우(FP)의 수, 실제값이 False이고 예측값이 False인 경우(TN)의 수로 ACC을 계산한다. 계산된 ACC의 값은 전체 예측에서 True나 False에 관계없이 옳은 예측의 비율로 성능을 알 수 있는 평가척도이다.

Table 6. Defect Severity-based Result Table

		Predicted Class	
		True(HSF)	False(LSF/NF)
Actual Class	True (HSF)	True Positive(TP)	False Negative(FN)
	False (LSF/NF)	False Positive(FP)	True Negative(TN)

$$ACC(Accuracy) = \frac{TP+TN}{TP+FP+FN+TN} \quad (2)$$

Table 7. Performance comparison table by model

		ACC
Existing Model	JM1	0.542791
	PC4	0.628938
CL Model	JM1	0.691211
	PC4	0.749657

표 7에서 보는 것과 같이 기존 결함심각도에 기반하지 않은 모델과 클러스터링 라벨을 적용한 결함심각도에 기반한 모델의 ACC의 값을 비교했을 때 성능이 JM1 0.15(15%), PC4 0.12(12%)가 더 향상됨을 보였다.

V. Conclusions

최근 연구되는 결함심각도에 기반한 예측 모델은 전문가에 의해 분류를 하기에 객관화되기 어렵다. 그리고 결함 예측 모델은 비슷한 유형의 프로젝트를 가지고 학습알고리즘으로 훈련데이터세트를 생성하여 교차검증하는 방법으로 원본 데이터세트의 상이한 분포에 따라서 결함예측 결과가 달라지거나 이상치가 발생한다. 그러나 본 논문에서 제안하는 클러스터링 라벨(CL)을 이용한 결함심각도 기반 결함 예측 모델은 전문가나 유사 프로젝트 없이 결함이 분류되어 있지 않은 원본 데이터세트만을 가지고 Column에 따라 자동으로 분류하고 최적의 훈련데이터세트를 생성하는 것이 가능하다. 그리고 결함심각도에 기반하지 않은 기존 모델과 비교한 결과 결함심각도에 기반한 본 논문에서 제안한 모델의 성능이 JM1 0.15(15%), PC4 0.12(12%)가 더 향상되었음을 보였다. 향후에는 본 논문에서 다루지 못했던 CFS 방법 외에 최적의 핵심입력 Column의 선정에 대한 방법과 결함심각도에 기반한 새로운 모델을 제안하고 NASA 데이터세트에 적용하여 기존 방법론과 비교함으로써 모델의 성능을 평가할 것이다.

REFERENCES

[1] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," IEEE Transaction on vol. 33, pp 2-13, 2007.

[2] E. Engstrom, P. Runeson, and G. Wikstrand, "An empirical evaluation of regression testing based on fix-cache recommendations," in Software Testing, Verification and Validation (ICST), 2010 Third International Conference on, pp. 75-78, April 2010.

[3] Jaechang Nam, Sunghun Kim, "CLAMI: Defect Prediction

on Unlabeled Datasets", IEEE/ACM International Conference on Automated Software Engineering, 2015.
[4] IEEE Computer Society, "IEEE Standard Classification for Software Anomalies", IEEE Std. 1044, 2009.
[5] T. Menzies and A. Marcus, "Automated Severity Assessment of Software Defect Reports", Proc. of ICSM 2008, pp. 346-355, 2008.
[6] Tracy Halla, Sarah Beechamb, David Bowesc, David Grayc, Steve Counsella, "A systematic literature review on fault prediction performance in software engineering," Software Engineering, IEEE Transactions on 38.6, pp. 1276-1304, 2012.
[7] Y. Singh, A. Kaur and R. Malhotra, "Empirical validation of object-oriented metrics for predicting fault proneness models," Software Quality Journal, Vol.18, pp.3-35, 2010.
[8] E. S. Hong, "Sortware Quality Prediction based on Defect Severity," Journal of the Korea Society of Computer and Information, Vol. 20, No. 5, pp. 73-81, 2015.
[9] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction.", Applied Soft Computing Vol.27, pp. 504-518, Feb. 2015.
[10] E. S. Hong, M. K. Park, "Severity-based Software Quality Prediction using Class Imbalanced Data", Journal of the Korea Society of Computer and Information, Vol. 21, No. 4, pp. 77-80, 2016.

Authors

Na-Young Lee received the B.S., M.S. degrees in Computer Engineering and Education from Gangneung-Wonju National University, Korea, in 2003 and 2005, respectively. She finished doctoral course in 2016. She is interested in Software Quality, Data Mining.

Ki-Tae Kwon received the B.S., M.S. and Ph.D. degrees in Computer Science from Seoul National University, Korea, in 1986, 1988 and 1993, respectively.Dr. Kwon joined the faculty of the Department of Computer Science at Gangneung National University, Gangneung, Korea, in 1990. He is currently a Professor in the Department of Computer Science and Engineering, Gangneung-Wonju National University. He is interested in Software Engineering, Data Mining, and Statistical Learning.