The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 3, pp.151-157, Jun. 30, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.3.151
JIIBC 2018-3-20

비감독형 학습 기법을 사용한 심각도 기반 결함 예측

Severity-based Fault Prediction using Unsupervised Learning

홍의석*

Euyseok Hong*

요 약 소프트웨어 결함 예측에 관한 기존의 연구들은 대부분 모델의 입력 모듈이 결함을 가지고 있는지 여부를 판단하는 이진 감독형 분류 모델들에 관한 것들이었다. 하지만 이진 분류 모델은 결함의 복잡한 특성들을 고려하지 않고 단순히 입력 모듈의 결함 유무만을 판단한다는 문제점이 있고, 감독형 모델은 대부분의 개발 집단이 보유하고 있지 않은 훈련 데이터 집합을 필요로 한다는 한계점이 있다. 본 논문은 이러한 두 가지 문제점을 해결하기 위해 비감독형 알고리즘을 사용한 심각도 기반 삼진 분류 모델을 제안하였으며, 평가 실험 결과 제안 모델이 감독형 모델들에 필적하는 예측 성능을 보였다.

Abstract Most previous studies of software fault prediction have focused on supervised learning models for binary classification that determines whether an input module has faults or not. However, binary classification model determines only the presence or absence of faults in the module without considering the complex characteristics of the fault, and supervised model has the limitation that it requires a training data set that most development groups do not have. To solve these two problems, this paper proposes severity-based ternary classification model using unsupervised learning algorithms, and experimental results show that the proposed model has comparable performance to the supervised models.

Key Words : Fault prediction, Defect severity, Unsupervised learning

Ⅰ. 서 론

소프트웨어 시스템 개발 초기 단계에서 구성 모듈들의 결함 정보들을 예측하는 결함 예측 모델은 최적의 자원할당, 적절한 리팩토링 후보 결정 및 테스트 프로세스 개선을 가능케 함으로써 신뢰성 있는 시스템 구축에 큰 도움이 된다. 지난 수십년간 수행된 결함 예측 모델들에 관한 연구들은 대부분 모듈의 결함경향성을 모듈이 보유한 결함 유무만으로 판단하는 이진 분류 모델이었으며, 모델 구축을 위해 훈련 데이터 집합이 필요한 감독형 학습 모델이었다[1]. 결함 유무만으로 모듈의 결함관련 특성을 규정하는 것은 적절치 않다. 왜냐하면 모듈이 지닌 결함의 수와 결함 심각도나 우선도와 같은 결함 특성들을 전혀 고려하지 않기 때문이다[2]. 결함의 특성들 중 가장 중요한 것은 결함 심각도이다. 결함 심각도는 소프트웨어 시스템과 사용자들에게 결함이 미치는 충격의 정도를 나타내는 척도이며[3], 척도를 규정하는 합의된 표준은 없지만 심각한 단계를 나타내는 서수 스케일의 레벨값으로 표현된다. 예를 들면, 결함 예측 모델 연구에 가장 많이 사용된 NASA MDP 공개 데이터 집합은 5단계의 심각도

*정회원, 성신여자대학교 정보시스템공학과
접수일자: 2018년 4월 9일, 수정완료: 2018년 5월 9일
게재확정일자: 2018년 6월 8일

Received: 9 April, 2018 / Revised: 9 May, 2018
Accepted: 8 June, 2018
*Corresponding Author: hes@sungshin.ac.kr
Dept. of Information Systems Engineering,
Sungshin Women's University, Korea

레벨값을 포함하며 심각도 1은 시스템 개발을 중단시킬 정도의 가장 심각한 결함을, 심각도 5는 수행 상 문제가 되지 않을 매우 가벼운 결함을 나타낸다. 매우 심각한 결함을 보유한 모듈과 아주 가벼운 결함만을 보유한 모듈을 구분 없이 모두 결함경향 모듈로 보는 이진 분류 모델보다는 모듈이 지닌 결함의 심각도에 따라 모듈의 세분화된 결함경향성을 예측하는 다중 분류 모델이 성공적인 소프트웨어 프로세스 관리에 훨씬 효율적으로 사용될 수 있다[2]. 기존 연구들의 또 다른 문제점은 대부분 모델들이 모델 구축에 훈련 데이터 집합이 필요한 감독형 학습 알고리즘들을 사용하였다는 것이다. 예측 모델이 필요한 많은 개발 집단들은 이러한 데이터 집합을 보유하고 있지 않다. 따라서 이들을 대체할 수 있는 비감독형 학습 알고리즘들을 사용한 모델들이 제안되었지만 모델 제작의 어려움과 낮은 성능의 문제 때문에 매우 극소수에 불과하다. 본 논문의 목적은 기존 예측 모델들의 두 문제점들을 해결하기 위하여, 비감독형 학습 알고리즘을 사용한 심각도 기반 결함 예측 모델을 제안하고 제안 모델의 성능을 감독형 모델과 비교 평가하여 그 효용성을 증명하는 것이다.

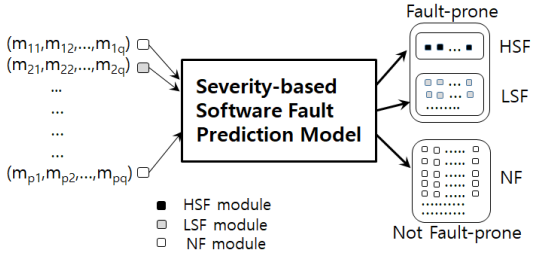

그림 1. 심각도 기반 결함 예측 모델
Fig. 1. Severity-based Fault Prediction Model

그림 1은 제안 모델을 나타낸다. 모델의 입력은 결함경향성을 예측하고자하는 대상인 소프트웨어 시스템의 모듈들의 집합이며, 모듈들은 그들의 특성을 나타내는 소프트웨어 메트릭 값들로 정량화된다. 그림에서 입력 데이터 집합은 q차원 메트릭 벡터 형태로 정량화된 p개 모듈들의 집합이다. 기존의 이진 분류 모델은 이들을 단순히 결함경향 모듈인지 비결함경향 모듈인지로 판단하는 모델이지만, 제안 모델은 이들을 고심각(HSF: High Severity Faultprone) 모듈, 저심각(LSF: Low Severity Faultprone) 모듈, 비결함경향(NF: Not Faultprone)로 판단하는 삼중 분류 모델이다.

2장에서는 기존의 결함 예측 연구들과 심각도 기반 연구들을 살펴보고, 3장에서는 제안 모델의 구조에 대해 설명한다. 4장에서는 모델 성능 실험 결과를 감독형 모델과 비교하여 언급하고, 5장에서는 결론을 기술한다.

II. 관련 연구

감독형 모델의 학습에 필요한 훈련 데이터란 모델의 입력 데이터와 같은 형태의 입력에 결함 정보 출력값이 함께 있는 데이터이며 과거의 유사 프로젝트나 현재 프로젝트의 이전 릴리즈 개발에서 얻어진다. 감독형 예측 모델들은 훈련 데이터 학습을 위해 통계 기법들과 기계학습 기법들을 사용하였다. 가장 많이 사용된 알고리즘은 통계 기법에서는 로지스틱 회귀분석, 기계학습 기법에서는 인공 신경망, 베이지안 분류기, 판단 트리, SVM이었다. 1991년부터 2013년 말까지 발표된 감독형 모델에 관한 64개의 대표 연구들을 분석한 결과는 기계학습 모델들의 성능이 통계 모델들을 압도하였다[1]. 수많은 모델들이 제안되었지만 어떤 상황에 어떤 모델이 가장 우수하다는 결론은 내려지지 않았다. 합의된 일반화 모델이 없다는 것은 예측 모델이 데이터 집합들의 특성에 매우 의존적이라는 것을 뜻한다.

훈련 데이터 집합이 없는 대다수의 개발 집단에서 사용가능하다는 큰 장점이 있는 비감독형 학습 모델에 관한 연구들은 매우 극소수에 불과하다. 그 이유는 입력 데이터 분석에 전문가가 개입하는 반자동 모델이라 사용하기 어려우며, 감독형 모델 대비 예측 성능이 떨어지기 때문이다. 모델들은 주로 클러스터링 기법을 사용하였으며 클러스터 수를 결정하는 방법에 따라 결과 클러스터 수를 수동으로 결정하는 모델들과[4, 5] 자동으로 최적화된 클러스터링을 사용하는 모델들로[6] 분류할 수 있다. 사용한 알고리즘들은 K-means, X-means, EM, DBSCAN 등이었다.

심각도 기반 결함 예측 모델 연구들은 모두 감독형 모델들이었다. 이들은 결함을 보유한 모듈들을 해당 결함의 심각도에 따라 몇가지 레벨로 분류한 데이터를 학습한 모델로 입력 모듈의 결함 레벨을 예측하는 모델들이다. 결함 레벨은 고-저[2, 7] 또는 고-중-저[8]로 나뉘었으며 다중 분류 모델 형태이지만 성능 평가에서는 예측하

려는 특정 심각도 레벨과 나머지를 분류하는 이진 분류 형태를 사용하였다.

본 논문은 심각도 기반 모델에 사용된 적 없는, 비감독형 알고리즘을 사용한 예측 모델을 제작한다. 클러스터링 방법은 수동과 자동 방법을 모두 사용한다.

Ⅲ. 모델 제작

1. 모델링 프로세스

제안 모델의 제작 프로세스는 비감독형 이진 분류 모델[4] 제작과 유사하다. 단, 후자는 결과 클러스터들이 두 형태를 가지는데 비해, 전자는 그림 2와 같이 결과 클러스터들이 세 형태를 가진다.

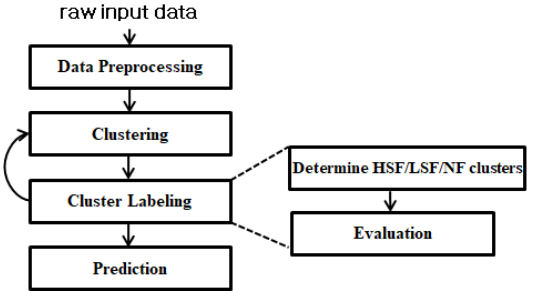

그림 2. 모델 제작 및 예측 프로세스
Fig. 2. Model Construction and Prediction Process

본 논문에서 사용하는 NASA 데이터 집합은 여러 결측값과 중복값 및 일관되지 않는 비정상적인 값들을 지니고 있다. 이들을 삭제하거나 수정하는 데이터 정제 작업이 데이터 전처리 과정에서 수행된다. 또한 원본 데이터에는 모듈의 정보를 나타내는 메트릭 집합과 해당 모듈의 심각도 정보가 다른 파일에 저장되어있으므로 이 정보들을 조인하는 작업도 이 과정에 포함된다. 결함 심각도 정보는 1~5 범위의 레벨값이다. 심각도 1을 고심각, 심각도 2~5를 저심각 결함으로 분류하고, 이들을 보유한 여부에 따라 고심각, 저심각 모듈을 정의한 [7]의 SEVERITY1 데이터를 사용한다. 또한 입력 메트릭값들을 [0, 1] 범위의 값으로 바꾸는 정규화 작업과 입력 속성 집합의 차원을 줄이는 속성 선정 작업도 수행된다.

클러스터링 알고리즘은 분할 클러스터링 기법들 중 모델 기반 알고리즘으로 가장 대표적인 EM 기법을 사용한다. 선정 이유는 EM 기법이 수동 및 자동 클러스터링이 모두 가능하다는 장점이 있고, [4]의 실험 결과 EM 기법이 K-means와 DBSCAN보다 나은 성능을 보였기 때문이다. 클러스터링 구현은 예측 모델 연구들에 많이 사용된 WEKA[9]를 사용하였다.

클러스터링 후 결과 클러스터들을 분석하여 각 클러스터에 고심각 결함경향(HSF), 저심각 결함경향(LSF), 비결함경향(NF) 라벨을 붙인다. 이 작업은 소프트웨어 메트릭 전문가들이 수행하며 라벨링 결과에 대한 평가를 통해 재 클러스터링이 필요한지를 판단해 만족한 결과가 나올 때까지 클러스터링-클러스터 라벨링 과정을 반복한다. 라벨링 결과 HSF 클러스터에 속한 모듈은 HSF 모듈, LSF 클러스터에 속한 모듈은 LSF 모듈, 그 외의 모듈은 NF 모듈로 예측되는 것이다. 클러스터링에 사용되지 않은 새로운 입력 모듈에 대한 예측이 필요한 경우는 입력 모듈과 가장 가까운 클러스터의 라벨이 입력 모듈의 예측 결과가 된다.

2. 클러스터 라벨링

클러스터 라벨링은 출력 클러스터들을 분석하여 HSF, LSF, NF 클러스터를 결정하는 작업이다. 일반적인 경우는 전문가의 분석 과정이 필요하지만 NASA 데이터를 사용하는 본 연구에서는 각 모듈의 입력 데이터에 대한 출력 심각도 값을 알고 있으므로 실험의 용이성을 위해 두 가지 방법을 사용하였다. 첫 번째 방법은 가장 많은 모듈을 가진 클러스터를 NF 클러스터로 가장 적은 모듈을 가진 클러스터를 HSF 클러스터로 정하는 것이다. 이는 시스템을 구성하는 HSF 모듈들이 매우 소수이며, 대다수의 모듈들은 NF 모듈들이라는 사실에 기반을 둔 방법이다. 두 번째 방법은 단순히 모듈의 수가 아닌 각 클러스터에 속한 결함 모듈들의 비율을 고려하는 방법이다. HSF 클러스터의 결정이 가장 중요하므로 HSF 모듈 비율이 가장 큰 클러스터를 HSF 클러스터로 선정하고, 나머지 클러스터들 중 NF 모듈 비율이 큰 클러스터를 NF 클러스터로, 나머지 클러스터를 LSF 클러스터로 선정한다.

표 1은 NASA 데이터 집합들 중 하나인 JM1에 전체 메트릭 속성들을 사용하고 출력 클러스터 수를 세 개로 고정한 경우의 클러스터링 실험 결과이다. 첫 번째 방법은 모듈 수가 가장 작은 클러스터 2를 HSF 클러스터로, 가장 큰 클러스터 0을 NF 클러스터로, 나머지 클러스터

1을 LSF 클러스터로 선정한다. 하지만 두 번째 방법은 HSF 모듈 비율이 가장 큰 클러스터 2를 HSF 클러스터로, NF 비율이 가장 큰 클러스터 1을 NF 클러스터로, 나머지 클러스터 0을 LSF 클러스터로 선정한다.

표 1. 클러스터 수가 세 개로 고정된 경우의 라벨링
Table 1. Labeling when the number of clusters is fixed to 3

클러스터	HSF		LSF		NF		SUM
0	153	2%	1025	17%	4972	81%	6150
1	31	1%	257	8%	2988	91%	3276
2	159	11%	477	33%	816	56%	1452

최적의 클러스터 수로 자동 클러스터링 되는 경우는 비율을 사용한 라벨링 방법을 사용한다. 표 2는 JM1에 자동 클러스터링을 사용한 결과이다. 결과는 9개의 클러스터로 나뉘며 HSF 모듈을 33%, 29% 보유한 클러스터 1과 3을 HSF 클러스터로, NF 모듈을 94%, 90%, 87%, 80% 보유한 클러스터 8, 6, 4, 2를 NF 클러스터로, 나머지 클러스터들을 LSF 클러스터로 설정한다.

표 2. 자동 클러스터링이 사용된 경우의 라벨링
Table 2. Labeling when the automatic clustering is used

클러스터	HSF		LSF		NF		SUM
0	23	13%	70	40%	82	47%	175
1	5	33%	7	47%	3	20%	15
2	79	3%	540	18%	2427	80%	3046
3	28	29%	28	29%	41	42%	97
4	38	1%	326	12%	2446	87%	2810
5	97	8%	364	30%	743	62%	1204
6	13	1%	107	8%	1141	90%	1261
7	57	4%	264	20%	1021	76%	1342
8	3	0%	53	6%	872	94%	928

Ⅳ. 실험 및 결과

1. 데이터 집합 및 평가 척도

NASA 데이터 집합들은 여러 프로젝트들로 구성되며 각 프로젝트는 여러개의 데이터 파일들로 구성된다. [10]은 모듈의 특정 정보를 서로 다른 파일들의 조합으로 얻어냈을 때 다른 결과가 나오는 것을 보고 이 데이터 파일들 사이에 심각한 모호성이 있으며, 가장 모호성이 가장 적은 프로젝트는 JM1과 PC4임을 밝혔다. JM1은 10878개의 모듈로 구성된 대형 시스템으로 PC4보다 훨씬 큰 프로젝트이고, PC4는 심각도 레벨값 정보가 상대적으로 부족하기 때문에 본 연구에서는 JM1을 실험 데이터 집합을 사용하였다. [7]에서도 JM1을 사용하였으므로 [7]의 감독형 모델 성능과 비교가 가능하다는 것도 중요한 선정 이유이다. JM1의 입력 메트릭은 21개이며 Halstead 계열, McCabe 계열, LOC 계열의 복잡도 관련 메트릭들이다. CFS(Correlation based Feature Selection)를 사용하여 21개 속성을 8개로 축소하였으며, 모델 평가 실험은 전체 속성을 사용한 경우와 차원 축소를 한 경우 두가지 형태로 시행하였다.

예측 모델의 성능 평가를 위해 이진 분류 모델에서 사용할 수 있는 평가 척도는 여러 가지가 있으나 심각도 기반 모델은 삼진 분류 모델이므로 Accuracy 외에는 사용하기가 어렵다[7]. 하지만 심각도 모델 성능에서 가장 중요한 것은 고심각 모듈인 HSF 예측 성능이므로 HSF 여부 관점에서 삼진 분류를 HSF와 Not HSF인 이진 분류로 변환할 수 있다. 표 3은 삼진 분류 모델의 결과인 삼진 오류행렬을 HSF에 기반한 이진 오류행렬로 변환한 내용과 평가 척도 수식을 보여준다. FNR(False Negative Rate)은 HSF 모듈을 Not HSF 모듈로 잘못 예측한 비율로 Type I/II 오류에서 더 중요한 오류인 Type II 오류를 의미한다. 평가 실험에서는 모델의 평가 척도로 전체 예측 결과들 중 맞는 예측 결과들의 비율인 Accuracy와 HSF 기반 FNR을 사용한다. Accuracy는 백분율 형태로 FNR은 비율 형태로 나타낸다.

표 3. 삼진 오류행렬
Table 3. Ternary confusion matrix

Predicted Class / Actual Class	HSF	LSF	NF
HSF	a_1	a_2	a_3
LSF	b_1	b_2	b_3
NF	c_1	c_2	c_3

⇒

Predicted Class / Actual Class	HSF	Not HSF
HSF	a_1	a_2+a_3
Not HSF	b_1+c_1	$b_2+b_3+c_2+c_3$

$$Accuracy = \frac{a_1+b_2+c_3}{\sum_i a_i + \sum_i b_i + \sum_i c_i} \qquad FNR = \frac{a_2+a_3}{\sum_i a_i}$$

2. 평가 실험

본 연구의 주요 목적은 심각도 기반 결함 예측 모델 영역에서 비감독형 모델들의 성능이 감독형 모델들과 얼마나 차이를 보이는가를 알아내는 것이다. 같은 데이터 집합과 전처리 방법 및 유사 평가 척도를 사용한 [7]의 성능 평가 결과는 표 4와 같다. All과 CFS는 각각 전체 속성과 CFS로 차원 축소를 한 경우를 나타낸다. 사용한 감독형 알고리즘들은 기존 예측 모델 연구들에서 가장 많이 사용되고 좋은 성능을 보였던 역전파 인공 신경망(BPN), 나이브 베이지안(NB), 판단 트리(J48)이며[1], Accuracy만으로 평가한 t-test 결과 BPN이 나머지 모델들보다 유의미하게 좋은 성능을 보였다[7]. 실험 결과를 평가하면 Accuracy는 80% 정도이므로 적당하나 FNR 값이 매우 높다. 이는 전체 모듈에서 HSF 모듈의 비율이 매우 작기 때문으로 표 3의 삼진 오류행렬에서 b_2, c_3의 값은 크나 a_1이 작다는 것을 의미한다. 즉, NF와 LSF 모듈을 옳게 예측해서 Accuracy 값은 높아졌지만 HSF 모듈을 잘못 예측해서 FNR 값도 높아진 것이다. 모델의 평가는 이 두 평가치의 적절한 조합에 대한 평가로 행해져야한다.

표 4. 감독형 모델의 실험 결과 [7]
Table 4. Experimental Results of Supervised Models [7]

평가 척도	속성 선정	NB	BPN	J48
Accuracy	All	78.70	80.75	78.51
	CFS	78.64	80.73	79.98
FNR	All	0.99	0.86	0.90
	CFS	0.99	0.86	0.92

3장에서 기술한 제작 방법에 의해 실험한 비감독형 모델의 실험 결과를 표 5에 나타내었다. 각 실험 결과는 클러스터 수를 고정한 경우와 자동으로 최적화한 경우, 모든 속성을 사용한 경우와 CFS에 의해 차원 축소를 한 경우, 클러스터 라벨링을 단순히 모듈 수로만 한 경우와 심각도 특성 비율을 고려한 경우에 의해 달라진다. Accuracy는 자동 클러스터링에 전체 속성을 사용한 경우가 가장 높았으나 이 경우 FNR이 0.91로 너무 높았다. 표 2는 이 경우의 9개의 클러스터가 생긴 결과를 나타낸 것이다. FNR을 줄이기 위해 HSF 모듈을 13% 가지고 있는 클러스터 0을 HSF로 라벨링 해본 결과 Accuracy는 69.59로 매우 조금 낮아졌지만 FNR은 0.84로 꽤 떨어졌다. 표 5에서 비율 조정의 결과가 이에 해당한다. 이는 최적화된 클러스터링을 하는 클러스터 알고리즘 결과가 가장 좋은 예측 모델이 되지는 않는다는 것을 의미한다. 결과적으로 자동화된 클러스터링을 사용한 비감독형 모델은 감독형 모델에 비해 FNR은 약간 좋지만 Accuracy는 10% 정도 떨어지는 결과를 보였다.

표 5. 비감독형 모델의 실험 결과
Table 5. Experimental Results of Unsupervised Model

클러스터 수	속성 선정	클러스터 라벨링	Accuracy	FNR
수동 (3)	All	수	42.82	0.54
		비율	38.35	0.54
	CFS	수	48.27	0.97
		비율	23.83	0.24
자동 (최적)	All	비율	70.02	0.91
		비율조정	69.59	0.84
	CFS	비율	61.66	0.68

표 6. 클러스터 수 변화에 따른 실험 결과
Table 6. Results according to the number of clusters

클러스터 수	Accuracy	FNR
3	42.82	0.54
9	69.59	0.84
10	73.94	0.64
11	77.73	0.71
13	75.11	0.57
15	77.16	0.49
17	76.57	0.67
19	81.02	0.68
23	82.51	0.70
30	79.66	0.34

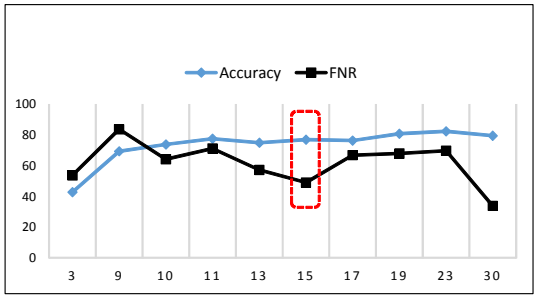

그림 3. 최적의 클러스터 수 찾기
Fig. 3. Finding the Optimal Number of Clusters

최적화된 클러스터 수가 가장 좋은 결과를 보이지 않기 때문에 더 나은 성능의 모델을 찾기 위해 클러스터 수를 늘려서 실험하였다. 클러스터 라벨링은 비율 방법에 기반하여 조정이 필요할 땐 약간 조정을 첨가하였다. Accuracy는 최적 수인 9부터 클러스터 수를 늘림에 따라 점차적으로 증가하지만 FNR은 감소 추세를 보이다 15 이후 다시 증가한다. Accuracy는 클러스터 수가 23일 때 82.51로 가장 좋은 결과를 보이나 FNR이 0.70으로 너무 높고 클러스터 수도 너무 많다. FNR은 클러스터 수가 30일 때 0.34로 가장 낮지만 역시 클러스터 수가 너무 많아 모델 제작이 어렵다. 따라서 클러스터 수가 15인 경우가 Accuracy는 77.16이고 FNR이 0.49로 매우 낮으므로 가장 좋은 결과를 보인다. 이는 감독형 모델 결과보다 Accuracy는 3% 낮지만 FNR은 거의 두배로 좋아진 결과이다. 실험을 통해 클러스터 수를 변화하며 만든 모델들 중에 성능면에서 감독형 모델에 필적하거나 능가하는 모델이 있다는 것은 알 수 있었지만 가장 좋은 결과를 보인 클러스터 수를 찾는 방법을 자동화하기는 어렵다는 문제가 있다.

V. 결 론

수십 년간 수행되었던 많은 소프트웨어 결함 예측 분야 연구들은 주로 감독형 학습 알고리즘들을 사용한 것들로 실제로 개발 프로젝트에 사용하기엔 훈련 데이터 집합의 부재라는 어려움이 있었다. 따라서 훈련 데이터 집합이 필요 없는 비감독형 모델에 대한 연구들이 시작되었지만, 모델 제작의 어려움과 예측 성능의 한계 때문에 수행된 연구들이 매우 극소수에 불과하며, 이들 조차도 결함 유무만을 결정하는 이진 분류 모델들이었다. 본 논문은 단순한 결함 유무 예측이 아닌 해당 소프트웨어 모듈이 고심각 결함 모듈, 저심각 결함 모듈, 비결함 모듈인지를 예측하는 삼중 품질 분류 모델을 EM 알고리즘을 사용하여 비감독형 모델로 제작하였다. 제안 모델을 감독형 모델과 비교한 성능 평가 실험 결과 사용된 평가 척도들 중 Accuracy는 떨어지지만 FNR은 더 나은 결과를 보였으며 클러스터 수를 늘려가며 수동으로 찾은 최적 모델은 FNR을 현저히 줄여 감독형 모델을 압도하는 성능을 보였다.

References

[1] R. Malhotra, “A systematic review of machine learning techniques for software fault prediction,” Applied Soft. Computing Vol.27, pp.504-518, 2015. DOI: https://doi.org/10.1016/j.asoc.2014.11.023

[2] Y. Zhou and H. Leung, “Empirical analysis of object-oriented design metrics for predicting high and low severity faults,” IEEE Trans. Software Eng., Vol.32, No.10, pp.771-789, Oct. 2006. DOI: https://doi.org/10.1109/tse.2006.102

[3] D. E. Harter, C. F. Kemerer and S. A. Slaughter, “Does Software Process Improvement Reduce the Severity of Defects? A Longitudinal Field Study,” IEEE Trans. Software Eng., Vol.38, No.4, pp. 810-827, July 2012. DOI: https://doi.org/10.1109/tse.2011.63

[4] E. Hong and M. Park, “Unsupervised learning model for fault prediction using representative clustering algorithms,” KIPS Trans. Software and Data Engineering, Vol.3, No.2, pp.57-64, Feb. 2014. DOI: https://doi.org/10.3745/ktsde.2014.3.2.57

[5] S. Zhong, T. M. Khoshgoftaar, and N. Seliya, “Analyzing Software Measurement Data with Clustering Techniques,” IEEE Intelligent Systems, Vol.19, No.2 pp.20-27, 2004. DOI: https://doi.org/10.1109/mis.2004.1274907

[6] P. S. Bishnu and V. Bhattacherjee, “Software fault prediction using quad tree-based k-means clustering algorithm,” IEEE Trans. Knowledge and Data Eng., Vol.24, No.6, pp.1146-1150, 2012. DOI: https://doi.org/10.1109/tkde.2011.163

[7] E. Hong, “Software Quality Prediction based on Defect Severity,” Journal of the Korea Society of Computer and Information, Vol.20, No.5, pp.73-81, May 2015.

[8] Y. Singh, A. Kaur and R. Malhotra, “Empirical validation of object oriented metrics for predicting fault proneness models,” Soft. Quality Journal, Vol.18, pp.3-35, March 2010. DOI: https://doi.org/10.1007/s11219-009-9079-6

[9] WEKA(Waikato Environment for Knowledge Analysis) http://www.cs.waikato.ac.nz/~ml/weka/

[10] E. Hong, "Ambiguity Analysis of Defectiveness in NASA MDP data sets," Journal of the Korea Society of IT Services, Vol.12, No.2, pp.361-371, June 2013.
DOI: https://doi.org/10.9716/kits.2013.12.2.361

[11] S. U. Lee and M. B. Choi, "A Definition and Evaluation Criteria for Software Development Success," Journal of the Institute of Internet, Broadcasting and Communication, Vol.12, No.2, pp.233-241, April 2012.
DOI: http://dx.doi.org/10.7236/JIWIT.2012.12.2.233

저자 소개

홍 의 석(정회원)

- 1992년 : 서울대학교 계산통계학과 전산과학전공 학사
- 1994년 : 서울대학교 계산통계학과 전산과학전공 석사
- 1999년 : 서울대학교 계산통계학과 전산과학전공 박사
- 현재 : 성신여자대학교 정보시스템공학과 교수

- 관심 분야 : 소프트웨어 품질 예측 모델, 소프트웨어 메트릭, MSR 등

※ 이 논문은 2016년도 성신여자대학교 학술연구조성비 지원에 의하여 연구되었음.