Journal of The Korea Society of Computer and Information
Vol. 20, No. 5, May 2015
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.5.073

결함 심각도에 기반한 소프트웨어 품질 예측

홍 의 석*

Software Quality Prediction based on Defect Severity

Euy-Seok Hong*

요 약

소프트웨어 결함 예측 연구들의 대부분은 입력 개체의 결함 유무를 예측하는 이진 분류 모델들에 관한 것들이다. 하지만 모든 결함들이 같은 심각도를 갖지는 않으므로 예측 모델이 입력 개체의 결함경향성을 몇 개의 심각도 범주로 분류할 수 있다면 훨씬 유용하게 사용될 수 있다. 본 논문에서는 전통적인 복잡도와 크기 메트릭들을 입력으로 하는 심각도 기반 결함 예측 모델을 제안하였다. 학습 알고리즘은 많이 사용되는 네 개의 기계학습 기법들을 사용하였으며, 모델 구조는 삼진 분류 모델로 하였다. 모델 성능 평가를 위해 실험 데이터는 두 개의 NASA 공개 데이터 집합을 사용하였고, 평가 측정치는 Accuracy를 이용하였다. 평가 실험 결과는 역전파 신경망 모델이 두 데이터 집합에 대해 각각 81%와 88% 정도의 Accuracy 값으로 가장 좋은 성능을 보였다.

▸ Keywords : 소프트웨어 품질, 메트릭, 결함 심각도, 결함 예측

Abstract

Most of the software fault prediction studies focused on the binary classification model that predicts whether an input entity has faults or not. However the ability to predict entity fault-proneness in various severity categories is more useful because not all faults have the same severity. In this paper, we propose fault prediction models at different severity levels of faults using traditional size and complexity metrics. They are ternary classification models and use four machine learning algorithms for their training. Empirical analysis is performed using two NASA public data sets and a performance measure, accuracy. The evaluation results show that backpropagation neural network model outperforms other models on both data sets, with about 81% and 88% in terms of accuracy score respectively.

▸ Keywords : software quality, metrics, defect severity, fault prediction

•제1저자 : 홍의석
•투고일 : 2015. 4. 4, 심사일 : 2015. 4. 22, 게재확정일 : 2015. 5. 7.
* 성신여자대학교 IT학부(School of Information Technology, Sungshin Women's University)
※이 논문은 2013년도 성신여자대학교 학술연구조성비 지원에 의하여 연구되었음.

I. 서 론

메트릭 기반 소프트웨어 결함 예측 모델은 소프트웨어를 구성하는 개체들을 소프트웨어 메트릭들로 정량화한 후 이들의 결함 정보를 예측하는 모델이다. 소프트웨어 개체는 모듈, 함수, 클래스 등이며 이들의 정량화에는 전통적인 복잡도 메트릭들이나 객체지향 메트릭들 또는 가끔 프로세스 메트릭이 사용된다. 모델은 구현이 완료된 후에 또는 다음 릴리즈에 나타날 결함 정보들을 미리 찾아냄으로써 적절한 자원 할당, 테스트 프로세스 개선, 최적의 리팩토링 후보 결정, 테스트 프로세스 개선을 통한 품질 개선, 높은 신뢰성을 갖춘 시스템 구축을 가능케 한다[1].

소프트웨어 결함은 소프트웨어 요구사항이나 명세를 만족시키지 못하게 만드는 소프트웨어 산물의 문제나 결핍이다[2]. 대부분의 기존 결함 예측 모델들이 예측한 결함 정보는 입력 개체의 결함수가 아니라 개체가 결함을 가지고 있는지 여부를 나타내는 결함경향성이었다[2,3,4]. 이는 결함경향성 예측을 위한 분류 모델이 결함수 예측 모델보다 구축이 용이하고, 정확도가 높다는 점과 활용 면에서도 크게 뒤지지 않는다는 점에 기인한다[2]. 하지만 단순히 결함 유무 정보만을 갖는 결함경향성은 한계점을 가지고 있다. 왜냐하면 각 결함이 가지고 있는 심각도를 고려하지 않았기 때문이다. 결함 심각도는 소프트웨어 시스템과 사용자들에게 결함이 미치는 충격의 정도를 나타내는 척도이다[5]. 매우 심각한 결함은 시스템 개발 작업을 중단시킬 정도이고, 가벼운 결함은 수행 상 문제가 되지 않을 정도이다. 따라서 심각도에 기반한 결함경향성을 예측하는 모델은 시스템 개발과 테스트 시 더욱 집중하여야하는 고심각 결함경향 부분에 대한 정보를 제공함으로써 결함 유무 예측 모델보다 훨씬 효율성이 높다[6,7,8].

소프트웨어 결함 예측에 관한 연구들은 2000년대 초에 NASA IV&V 메트릭스 데이터 프로그램에 의한 NASA 데이터 집합이 공개된 후, 대표적인 소프트웨어 품질 관련 공개 데이터 집합인 PROMISE 레포지토리[1] 운영이 시작된 2005년부터 크게 늘어났다[3,4]. 하지만 심각도에 기반한 결함 예측에 관한 연구들은 매우 극소수에 불과하다. 이는 결함 심각도에 대한 데이터 수집이 어렵고, 정확한 결함 심각도를 판단할 지식을 가진 전문가를 찾기 어렵기 때문이다[7]. 따라서 심각도 평가 시 텍스트 마이닝, 기계학습 등을 이용한 자동 평가 기법들도 등장하였다[9].

기존의 심각도 기반 모델 연구들은 모두 객체지향 메트릭들을 사용하였고, 일부 한정된 데이터 집합을 사용하였으며, 모델들의 성능 비교보다는 주로 입력 메트릭들이 결함경향성에 영향을 미치는가에 연구의 초점을 두었다. 본 논문의 목적은 전통적인 복잡도와 크기 메트릭들을 사용하는 심각도 기반 결함경향성 예측 모델들을 제작하여 가장 좋은 성능을 갖는 감독형 모델을 찾는 것이다. 제안 모델은 심각도 범주들을 표현하는 다중 클래스 분류 모델 형태로 하였으며, 모델 제작 및 검증 데이터는 NASA 데이터 집합을 사용하였고, 모델 학습 알고리즘은 기계학습 알고리즘들을 이용하였다.

2장에서는 결함 예측 관련 연구들을 살펴보고, 3장에서는 제작 모델의 구조와 입출력 메트릭들 대해 설명한다. 4장에서는 모델 성능 실험 결과를 언급하고, 5장에서는 결론을 기술한다.

II. 관련 연구

1. 결함 심각도

IEEE Std. 1044-2009에서 심각도 레벨의 예를 명시하였지만[1] 결함 심각도 값을 정의하여 규정하는 합의된 표준은 없다. 여러 개발 또는 연구 집단에서 사용하는 심각도 값들은 심각한 단계를 나타내는 서수 스케일(ordinal scale) 형태의 심각도 레벨 형태이다.

IEEE Std. 1044-2009에서 명시한 심각도 레벨값은 Blocking에서 Inconsequential 까지 5단계이며, 결함 추적 시스템으로 Bugzilla[2]를 사용한 Eclipse 등의 오픈 소스 프로젝트들이 정의한 레벨값은 Blocking에서 Enhancement 까지 7단계이다. Blocking은 시스템 개발과 테스트가 불가능한 심각한 결함을 의미하고 Inconsequential은 수행에 중요한 영향이 없는 결함, Enhancement는 결함이라 보기 어려운 개선 요구를 의미한다. NASA 데이터 집합을 구성하는 프로젝트들 중에서 몇몇 집합은 5단계의 심각도 레벨값을 포함하고 있으며 심각도 1이 가장 심각한 결함을, 심각도 5가 가장 가벼운 결함을 나타낸다. [10]은 모듈이 가진 결함수와 각 결함의 심각도 정보를 이용하여 모듈 심각도를 측정하는 메트릭 집합을 정의하고 NASA 데이터 집합을 이용하여 유용성을 실험하였다.

2. 결함 예측 모델

기존에 제안된 수많은 결함 예측 모델들의 대부분은 훈련

1) http://openscience.us/repo/

2) http://www.bugzilla.org/

데이터를 사용하여 학습하는 감독형 모델들이다. 훈련 데이터란 입력 데이터와 그에 대한 결함 정보 출력값이 함께 있는 데이터 집합을 의미하며, 감독형 모델은 입력 데이터에 대해 해당 답이 나오도록 학습한 모델을 의미한다. 학습 알고리즘으로는 로지스틱 회귀분석 같은 통계 기법들과 인공 신경망 같은 기계학습 기법들이 사용되었다. [3]은 1991년부터 2013년 말까지 발표된 결함 예측에 관한 중요 연구 64개를 분석하여 기계학습 모델들의 성능이 통계 모델들보다 월등히 좋다는 결론을 내렸다. 기계학습 기법들 중 가장 많이 사용된 것들은 판단트리, 베이지안 분류기, SVM(Support Vector Machine), 인공 신경망이었다. 따라서 본 연구에서도 이들 기법들을 사용한다.

감독형 모델의 문제점은 많은 개발 집단이 훈련 데이터 집합을 보유하고 있지 않으므로 해당 모델들을 사용하기 어렵다는 것이다. 따라서 훈련 데이터를 사용하지 않는 비감독형 모델의 필요성이 대두되었으며[1] 매우 극소수의 연구들이 수행되었다. 모델 구현에 주로 사용된 기법은 클러스터링 기법들이며 K-means, X-means, EM, DBSCAN 등의 알고리즘들이 사용되었다[11, 12]. 비감독형 모델은 학습할 데이터가 없으므로 예측 성능이 감독형 모델에 비해 떨어지고, 전문가의 클러스터 분석 과정이 필요하다는 문제점이 있다.

심각도에 기반한 결함 예측에 관한 연구들은 모두 객체지향 시스템을 대상으로 한 감독형 모델들이다. [7]과 [8]은 객체지향 시스템인 NASA의 KC1 프로젝트를 이용해 심각도 기반 예측 모델을 제작하였으며, 사용한 C&K 메트릭들[13] 중 DIT를 제외한 대부분이 심각도에 기반한 클래스 결함경향성과 밀접한 관계가 있음을 보였다.

[7]은 학습 기법으로 로지스틱 회귀분석과 Naïve Bayes, Random Forest, NNge를 사용하였으며 심각도 1을 고심각 결함, 2~5를 저심각 결함으로 분류하였다. [8]은 학습 기법으로 로지스틱 회귀분석과 판단 트리, 인공 신경망을 사용하였으며, 심각도 1을 고심각, 2를 중심각, 3~5를 저심각 결함으로 분류하였다. 이들은 성능 평가에서 다중 클래스 분류 모델을 사용하지 않고 평가의 용이성을 위해 이진 분류 모델 형태를 사용하였다. 즉, 고심각 결함 예측 모델은 입력 클래스가 고심각 결함 보유 클래스냐 아니냐를 출력하는 이진 분류 모델이다. 두 연구 모두 저심각 예측이 고심각 예측보다 좋은 성능을 보였고, [8]은 중심각 예측의 성능이 가장 좋았다. [7]은 각 메트릭이 출력과 유의미한 관계가 있는지를 알아내는데 초점을 맞춘 반면 [8]은 각 알고리즘들의 성능 비교도 ROC 분석을 통해 수행하였다. 그 결과 [3]의 관찰과 유사하게 기계학습 기법이 통계 기법보다 유의미하게 좋은 성능을 보였다.

[6]은 예측 모델이 다음 릴리즈의 심각도 기반 결함경향 클래스들을 판별해낼 수 있는지에 대한 연구를 Eclipse 프로젝트의 세 개의 릴리즈 데이터를 사용하여 수행하였다. 모델 구축을 위해 다항 로지스틱 회귀분석을 사용하였으며, Bugzilla의 7개의 심각도 레벨들 중 가장 가벼운 두 개의 레벨들을 제외하고 나머지 레벨들을 고심각, 중심각, 저심각의 세 개의 범주로 분류하였다. 해당 결함을 가진 클래스들을 각각 고심각, 중심각, 저심각 클래스들로 분류하였으며, 두가지 종류의 심각도 결함들을 가진 클래스는 각 분류에 한 번씩 삽입되었다. 실험 결과 릴리즈를 반복할수록 메트릭 및 모델의 예측 정확도가 떨어져 소수의 메트릭들을 제외하고는 메트릭들 및 예측 모델을 사용하기 어렵다는 결론을 내었다.

III. 모델 제작

1. 데이터 집합

1.1 데이터 집합 선정

NASA 데이터 집합의 두 버전들 중 PROMISE 데이터 집합은 심각도 정보가 없으므로 모델 제작 및 실험을 위해 초기 NASA 원본 데이터 집합을 사용하였다. 이를 구성하는 13개의 프로젝트들을 표 1과 같이 분석하여 심각도 정보가 없거나 모두 0인 것들을 실험 대상에서 제외하였다. 모델의 입력 메트릭 값이 매우 부족하여 결측값이 많은 KC4를 제외하고 실험 고려 대상을 JM1, KC1, KC3, PC4로 축소하였다.

표 1. 심각도 정보 분석
Table 1. Analysis of NASA data sets with severity

프로젝트	심각도 정보
CM1, PC1, PC5, MC1, MC2	없음
PC2, PC3, MW1	모두 0
JM1, KC1, KC3, KC4, PC4	있음, KC4 입력 메트릭 부족

NASA 원본 데이터 집합의 프로젝트는 여러개의 파일들로 구성되며 중요한 두 개의 파일은 product_module_metrics와 defect_product_relations이다. 전자는 모듈 id와 모듈에 대한 여러 메트릭값들과 모듈이 가진 결함수를 의미하는 error_count를 가지고 있다. 후자는 각 모듈이 가진 결함들과 그 결함들의 심각도 정보들을 가지고 있다. 2장의 그림 2의 상단에 있는 두 개의 테이블이 KC1의 두 파일을 나타낸

것이다. [14]는 NASA 원본 데이터 집합의 프로젝트들을 구성하는 이들 파일들 사이에 심각한 모호성이 있음을 밝혔다. 이는 두 파일로부터 얻어지는 결함 정보가 다르기 때문이다. 예를 들면, 그림 2의 KC1의 두 파일에서 왼쪽 파일 정보는 모듈 22903의 결함수가 1이라 되어있지만 오른쪽 파일을 보면 두 개의 결함(2283, 2284)을 갖고 있으므로 데이터 집합에 모호성이 존재한다. [14]의 실험 결과, 모호성이 가장 적은 프로젝트는 JM1과 PC4이므로 두 프로젝트들을 모델 제작에 사용하였다.

1.2 데이터 집합의 결함 분포

JM1은 C 언어로 구현된 실시간 시스템 프로젝트이며, 315 KLOC, 10,878개의 모듈들로 구성된다. 결함 보유 모듈은 2,102개가 있으며 결함은 3,161개 존재한다. 결함 심각도 값은 1~5 범위의 값을 갖는다. PC4는 C 언어로 구현된 지구 궤도 위성 비행 시스템 프로젝트이며, 36 KLOC, 1,458개의 모듈들로 구성된다. 결함 보유 모듈은 178개가 있으며 결함은 370개 존재한다. 결함 심각도 값은 4, 5가 빠진 1~3 범위의 값을 갖는다.

표 2. JM1과 PC4의 심각도에 기반한 결함 분포
Table 2. Fault distribution based on severity of JM1, PC4

프로젝트	결함 심각도	결함 모듈수	결함 모듈비율	결함수	결함비율
JM1	1	343	3.15%	372	11.77%
	2	163	1.5%	186	5.88%
	3	1144	-	1749	55.33%
	4	64	-	116	3.67%
	5	388	-	738	23.35%
PC4	1	58	3.98%	85	22.97%
	2	40	2.74%	92	24.86%
	3	80	-	193	52.16%
	4	0	-	0	0
	5	0	-	0	0

표 2는 JM1과 PC4의 심각도를 고려한 결함 분포를 나타낸 것이다. PC4는 심각도 값이 1~3까지만 존재한다는 문제가 있으나 심각도 3부터는 중요성이 낮은 심각도라 볼 수 있으므로 큰 문제가 되지 않는다. PC4에서 심각도 2인 모듈이 40개란 의미는 심각도 1인 결함은 없고, 심각도 2인 결함들이 있는 모듈의 수가 40개란 것이다. 단, 심각도 2인 결함을 갖는다는 것은 심각도 3~5인 결함도 가질 수 있다는 의미이다. JM1의 심각도 5인 모듈수 388은 심각도 1~4인 결함이 없으며, 심각도 5인 결함만 갖는 모듈 수란 의미이다. 즉 심각도 레벨에 따른 결함 모듈수는 모두 독립적인 의미이므로 이들의 합은 결함 보유 모듈수가 된다. 두 프로젝트는 가장 심각한 모듈인 심각도 1인 모듈의 비율이 3% 대로 비슷한 분포를 보인다.

2. 예측 모델 제작

2.1 모델 구조

본 논문에서 제안하는 모델은 그림 1과 같이 n개의 모듈들 $\{M_1, M_2, \ldots, M_n\}$을 입력 받아 고심각 결함경향 그룹, 저심각 결함경향 그룹, 비결함경향 그룹으로 나누는 삼진 분류 모델이다.

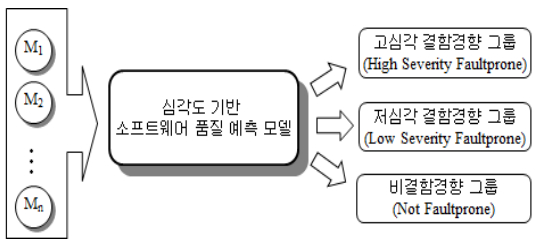

그림 1. 심각도 기반 분류 모델 구조
Fig. 1. Structure of the severity-based classification model

시스템을 구성하는 모듈이 d개의 메트릭들로 정량화될 때, 입력 메트릭 벡터 $\mathbf{x_i}$와 출력값 y_i의 쌍인 ($\mathbf{x_i}$, y_i)가 n개 있는 훈련 데이터 집합은 다음과 같이 표현된다. 출력은 고심각 결함경향, 저심각 결함경향, 비결함경향을 나타내는 상수값인 HSF(High Severity Faultprone), LSF(Low Severity Faultprone), NF(Not Faultprone)가 된다.

$$(\boldsymbol{x_i}, y_i),\ i=1,\ldots,n \quad where\ \boldsymbol{x_i} \in R^d, y_i \in \{HSF, LSF, NF\} \tag{1}$$

기계 학습 또는 통계 기법을 이용하여 분류 문제를 해결하는 것은 (1)에 기술한 훈련 데이터 집합을 가장 잘 학습하여 다음과 같은 최적화된 모델 f를 구하는 것이다. (1) 데이터를 학습한 모델 f는 (2)와 같이 하나의 모듈, 즉 모듈을 정량화한 입력 메트릭 벡터를 입력으로 받아 그 모듈의 결함경향성 값을 출력한다.

$$f : R^d \mapsto \{HSF, LSF, NF\} \tag{2}$$

2.2 입출력 메트릭 정의

NASA 프로젝트가 가진 파일들을 합성하여 모델 제작에

사용할 입출력이 포함된 하나의 파일로 만들었다. 그림 2는 KC1의 예를 든 것으로, KC1의 product_module_metrics와 defect_product_relations를 합성하여 두 파일의 밑의 그림과 같은 데이터 집합을 제작한 것을 나타낸 것이다. 모델의 입력은 product_module_metrics의 모듈의 성질을 나타내는 메트릭들을 사용하며, 출력은 해당 모듈의 결함과 심각도 정보를 defect_product_relations에서 찾아 NF, LSF, HSF 중 하나로 결정한다. 그림 2에서 모듈 22952는 심각도 1인 결함이 존재하므로 출력값이 HSF가 된다.

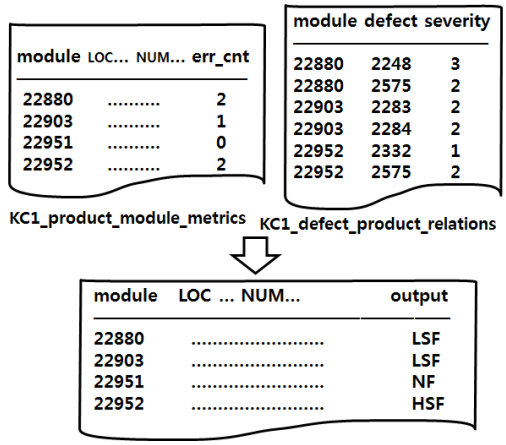

그림 2. 실험 데이터 제작
Fig. 2. Construction of experimental data

표 3. 품질 예측 모델의 입력 메트릭들
Table 3.Input metrics of the quality prediction models

프로젝트		입력 메트릭	설명
PC4 (36)	JM1 (21)	LOC_BLANK	The number of blank lines in a module.
		BRANCH_COUNT	Branch count metrics.
		LOC_CODE_AND_COMMENT	Number of lines which contain both code & comment in a module.
		LOC_COMMENTS	The number of lines of comments in a module.
		CYCLOMATIC_COMPLEXITY	The cyclomatic complexity of a module.
		DESIGN_COMPLEXITY	The design complexity of a module.
		ESSENTIAL_COMPLEXITY	The essential complexity of a module.
		LOC_EXECUTABLE	The number of lines of executable code for a module (not blank or comment)
		HALSTEAD_CONTENT	The Halstead length content of a module.
		HALSTEAD_DIFFICULTY	The Halstead difficulty metric of a module.
		HALSTEAD_EFFORT	The Halstead effort metric of a module.
		HALSTEAD_ERROR_EST	The Halstead error estimate metric of a module.
		HALSTEAD_LENGTH	The Halstead length metric of a module.
		HALSTEAD_LEVEL	The Halstead level metric of a module.
		HALSTEAD_PROG_TIME	The Halstead programming time metric of a module.
		HALSTEAD_VOLUME	The Halstead volume metric of a module.
		NUM_OPERANDS	The number of operands contained in a module.
		NUM_OPERATORS	The number of operators contained in a module.
		NUM_UNIQUE_OPERANDS	The number of unique operands contained in a module.
		NUM_UNIQUE_OPERATORS	The number of unique operators contained in a module.
		LOC_TOTAL	The total number of lines for a given module.
		CALL_PAIRS	Number of calls to other functions in a module.
		CONDITION_COUNT	Number of conditionals in a given module.
		CYCLOMATIC_DENSITY	Ratio of the module's cyclomatic complexity to its length in NCSLOC.
		DECISION_COUNT	Number of decision points in a given module.
		DECISION_DENSITY	Calculated as: Cond / Decision.
		DESIGN_DENSITY	Design density is calculated as: iv(G)/v(G).
		EDGE_COUNT	Number of edges in a given module. Transfer of control from one module to another.
		ESSENTIAL_DENSITY	Essential density is calculated as: (ev(G)-1)/(v(G)-1).
		PARAMETER_COUNT	Number of parameters to a given module.
		MAINTENANCE_SEVERITY	Maintenance Severity is calculated as: ev(G)/v(G).
		MODIFIED_CONDITION_COUNT	
		MULTIPLE_CONDITION_COUNT	Number of multiple conditions that exist within a module.
		NODE_COUNT	Number of nodes found in a given module.
		NUMBER_OF_LINES	Number of lines in a module. Pure, simple count from open bracket to close bracket.
		PERCENT_COMMENTS	Percentage of comments. Calculated:((CLOC+C&SLOC)/(SLOC+CLOC+C&SLOC))*100
	생략 메트릭	GLOBAL_DATA_COMPLEXITY	cyclomatic complexity of module's structure as it relates to global/parameter data.(모두 0)
		GLOBAL_DATA_DENSITY	Global Data density is calculated as: gdv(G) / v(G). (모두 0)
		PATHOLOGICAL_COMPLEXITY	measure of degree to which a module contains extremely unstructured constructs.(모두 1)
		NORMALIZED_CYLOMATIC_COMPLEXITY	CYCLOMATIC_COMPLEXITY 와 중복 메트릭

① 입력 메트릭

사용할 프로젝트인 JM1과 PC4를 분석하여 얻은 모델의 입력 메트릭들을 표 3에 나타내었다. 결함 결과 관련 메트릭인 error_count, error_density를 제외하면 JM1의 입력 메트릭은 21개이다. 이들은 Halstead 계열, McCabe 계열, LOC 계열의 복잡도 관련 메트릭들이며, 다른 메트릭들의 수식으로 얻어진 파생 메트릭들도 입력에서 제외하지 않았다. PC4의 입력 메트릭은 JM1의 입력 메트릭에 15개 메트릭들이 첨가되었다. PC4 메트릭들 중 global_data_complexity, global_data_density는 모든 모듈이 같은 값을 가지므로 입력에서 제외하였고, normalized_cyclomatic_complexity도 입력 집합에 있는 cyclomatic_complexity와 의미상 중복되므로 제외하였다.

② 출력 메트릭

식 (1), (2)에 의하면 제안 모델은 출력값으로 세가지 값을 갖는다. 실험 데이터 집합에 있는 각 모듈들의 출력값을 정하려면 이들 세가지 값에 관련된 정의를 해야 한다.

앞에서 기술한 바와 같이 심각도는 비율 스케일이 아니며 심각도 1은 심각도 2보다 훨씬 심각한 결함이다. 따라서 모듈의 심각도를 정의할 때 결함수를 고려하는 것은 큰 의미가 없다. 왜냐하면 심각도 1인 결함 하나가 심각도 2인 결함 10개보다 심각한 경우가 많기 때 문이다. 따라서 [7]과 같이 심각도 1인 결함을 고심각 결함, 나머지 심각도 2~5인 결함을 저심각 결함이라 볼 수 있다. 하지만 심각도 1인 결함이 매우 극소수인 데이터 집합의 경우 또는 심각도 2 역시 심각도 3~5보다는 훨씬 심각한 결함이라는 관점에서는 심각도 2까지를 고심각 결함이라 볼 수도 있다. 본 논문에서는 실험에 두개의 관점을 모두 사용하였으며, 전자를 사용한 데이터를 SEVERITY1, 후자를 사용한 데이터를 SEVERITY2라 하였다.

식 (2)와 [정의 2]를 이용하면 고심각 결함경향 모듈의 모델 출력값은 HSF이고, 저심각 결함경향 모듈은 LSF, 비결함경향 모듈의 출력값은 NF임을 알 수 있다. 출력값은 [정의 1]에 의해 결정된다. 그림 2의 실험 데이터 집합은 [정의 1]의 SEVERITY1의 결과이므로 모듈 22880, 22903의 출력값은 LSF가 된다.

[정의 1] 고심각 결함, 저심각 결함
SEVERITY1: 고심각 결함은 심각도 값이 1인 결함이며, 저심각 결함은 2, 3, 4, 5 중 하나인 결함이다.
SEVERITY2: 고심각 결함은 심각도 값이 1 또는 2인 결함이며, 저심각 결함은 3, 4, 5 중 하나인 결함이다.

[정의 2] 고심각 결함경향 모듈, 저심각 결함경향 모듈, 비결함경향 모듈
고심각 결함경향 모듈은 고심각 결함이 하나 이상 있는 모듈이고, 저심각 결함경향 모듈은 고심각 결함이 없고 저심각 결함이 하나 이상 있는 모듈이며, 비결함경향 모듈은 결함이 없는 모듈이다.

IV. 실험 및 평가

1. 속성 선정

입력 메트릭의 차원이 21 이상이므로 작지 않고 각 메트릭이 세가지 부류에 속하는 메트릭들이며 파생 메트릭들도 없애지 않았으므로 차원 축소 작업을 수행하였다. 축소 기법으로 CFS(Correlation based Feature Selection)를 사용하였으며, 이는 입력 메트릭들의 모든 가능한 부분집합들을 찾아 메트릭들의 예측 성능과 그들 사이의 상관도를 평가해 가장 좋은 성능의 부분집합을 찾아내는 방법이다[3]. 표 4는 두 프로젝트에 대한 차원 축소 결과이다.

표 4. 속성 선정 결과
Table 4. Results of attribute selection

프로젝트	선정된 입력 메트릭
JM1	LOC_BLANK, LOC_CODE_AND_COMMENT, LOC_COMMENTS, CYCLOMATIC_COMPLEXITY, DESIGN_COMPLEXITY, ESSENTIAL_COMPLEXITY, HALSTEAD_CONTENT, LOC_TOTAL
PC4	LOC_CODE_AND_COMMENT, PARAMETER_COUNT, PERCENT_COMMENTS, MODIFIED_CONDITION_COUNT

2. 평가 실험

실험은 WEKA[3]의 Experimenter를 이용하였으며 모델의 입력 메트릭은 전체와 차원 축소한 두 경우, 출력은 [정의 1]의 SEVERITY1, SEVERITY2의 두 경우를 모두 실험하였다. 사용한 알고리즘들은 예측 모델 연구에 가장 많이 사용되어온 네가지 기법들을 사용하였다[3]. 이들은 대표적인 인공 신경망 기법인 역전파 신경망(BPN: BackPropagation Neual Net)과 SVM, 베이지안 모델의 Naïve Bayes(NB), 판단 트리 모델인 J48이다. 각 모델의 구조와 초기 설정값은

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

WEKA의 기본값을 이용하였다. 교차 검증은 일반적인 10 폴드 교차 검증을 사용하였으며 실험의 정확도를 높이기 위해 이를 10회 반복하였다. 즉, 한 모델을 훈련 데이터 집합과 검증 데이터 집합을 바꾸면서 10회씩 10회 반복하므로 100회 실행시킨다. 실험 결과 SVM은 모든 입력에 대해 NF를 출력하였으므로 SVM은 실험 결과에서 제외하였다.

이진 분류 모델의 성능 평가 척도로는 Accuracy, Precision, Recall, F-measure, Area Under ROC, Type I, II 오류 등이 사용된다. 하지만 삼진 분류에서는 모델 평가에 이들을 사용하기 어렵다. 왜냐하면 이들은 Accuracy를 빼고는 각 분류 클래스 별로 존재하는 척도이기 때문이다. 따라서 모델들의 전체적인 성능을 비교해보기 위해 전체 데이터에 대해 모델의 예측이 맞은 경우의 비율, 즉 Accuracy로 모델들의 예측 성능을 평가하였다. 각 모델의 성능이 다른 모델보다 유의미하게 차이가 나는지를 비교하기 위해 표준 t-test가 아니라 유의 수준 0.05인 corrected resampled t-test를 사용하였다. 표준 t-test가 값들 사이에 존재하는 의존성 때문에 중요한 차이를 생성할 수 있기 때문에 x-폴드 교차 검증을 사용하는 경우에는 corrected resampled t-test가 적당하다고 알려져 있다[15].

3. 실험 결과

표 5부터 네 개의 표는 세 개 모델의 성능 평가 결과를 나타낸다. 표 5와 표 6는 모델의 출력 중 고심각 결함을 심각도 1 결함까지로 여긴 SEVERITY1 데이터 집합으로 실험한 결과이고, 표 7와 표 8은 고심각 결함을 심각도 1, 2 결함까지로 여긴 SEVERITY2 집합으로 실험한 결과이다. 표 5, 표 7은 NB를 테스트 베이스로 하여 나머지 두 모델과의 성능을 비교한 결과이다. 이 결과로 알기 힘든 BPN과 J48의 성능 비교 결과는 BPN을 테스트 베이스로 하여 표 6, 표 8에 나타내었다. 표에서 측정치 우측의 'v'는 t-test 결과 해당 모델이 테스트 베이스 모델보다 유의미하게 좋은 성능을 지녔다는 의미이고, '*'는 나쁜 성능을 지녔다는 의미이다. 각 성능치 옆의 괄호 안의 값은 성능치의 표준편차 값이다. JM1_as와 PC4_as는 각 데이터 집합에 속성 선정(as: attribute selection)을 수행한 데이터 집합을 의미한다.

JM1의 경우에는 차원 축소 유무가 모델의 성능에 큰 차이를 만들지 않는다. 하지만 PC4를 사용한 경우는 실험의 모든 경우들에서 차원 축소를 한 PC4_as를 사용한 결과가 PC4를 사용한 결과보다 더 좋았다. 특히 NB 모델에서는 매우 큰 성능 차이가 났다. 또한 PC4를 사용한 결과들은 JM1 결과들보다 표준 편차가 컸다. 이와 같은 결과는 PC4가 36으로 JM1에 비해 매우 큰 입력 차원을 가지므로 입력 메트릭들 중 상호 연관된 것이 많음을 의미하며, NB 모델이 다른 모델들보다 중복된 데이터 처리에 약점을 보인다는 것을 의미한다.

표 5. SEVERITY1에서 NB가 테스트베이스 경우 실험 결과
Table 5. Result of SEVERITY1 data when NB is a test base

데이터집합	NB	BPN	J48
JM1	78.70(0.87)	80.75(0.51) v	78.51(0.93)
JM1_as	78.64(0.92)	80.73(0.36) v	79.98(0.83) v
PC4	64.80(6.65)	87.97(2.17) v	86.85(2.03) v
PC4_as	86.21(2.03)	88.05(1.34) v	88.07(1.67) v

표 6. SEVERITY1에서 BPN이 테스트베이스 경우 실험 결과
Table 6. Result of SEVERITY1 data when BPN is a test base

데이터집합	BPN	NB	J48
JM1	80.75(0.51)	78.70(0.87) *	78.51(0.93) *
JM1_as	80.73(0.36)	78.64(0.92) *	79.98(0.83) *
PC4	87.97(2.17)	64.80(6.65) *	86.85(2.03)
PC4_as	88.05(1.34)	86.21(2.03) *	88.07(1.67)

표 7. SEVERITY2에서 NB가 테스트베이스 경우 실험 결과
Table 7. Result of SEVERITY2 data when NB is a test base

데이터집합	NB	BPN	J48
JM1	78.65(0.94)	80.75(0.34) v	78.29(0.83)
JM1_as	78.44(0.93)	80.77(0.29) v	79.39(0.79) v
PC4	49.95(5.38)	88.43(2.02) v	86.40(2.48) v
PC4_as	88.55(1.57)	88.95(1.34) v	88.61(1.31)

표 8. SEVERITY2에서 BPN이 테스트베이스 경우 실험 결과
Table 8. Result of SEVERITY2 data when BPN is a test base

데이터집합	BPN	NB	J48
JM1	80.75(0.34)	78.65(0.94) *	78.29(0.83) *
JM1_as	80.77(0.29)	78.44(0.93) *	79.39(0.79) *
PC4	88.43(2.02)	49.95(5.38) *	86.40(2.48)
PC4_as	88.95(1.34)	88.55(1.57)	88.61(1.31)

SEVERITY1 데이터를 사용한 실험 결과는 BPN이 JM1, PC4에서 모두 NB 모델보다 유의미하게 좋은 성능을 보이며, J48 모델보다는 JM1에서 유의미하게 좋은 성능을 보였다. PC4인 경우는 차원 축소를 하지 않은 경우에만 J48 모델보다 약간 나은 결과를 보였다. J48 모델은 JM1 경우를 제외하고는 나머지 세 데이터 집합에서 NB 모델보다 유의미하게 좋은 성능을 보였다. SEVERITY2를 사용한 실험도 SEVERITY1을 사용한 결과와 유사하게 BPN이 나머지 두 모델들보다 유의미하게 좋은 결과를 보였다. J48 모델은 JM1_as, PC4에서 NB 모델보다 좋은 결과를 보였으며, 특

히 PC4인 경우는 매우 큰 차이가 났다. t-test 결과 나머지 두 모델들보다 좋은 성능을 보인 BPN은 JM1의 경우는 약 81%, PC4의 경우는 약 88%의 Accuracy 값을 보였다. 실험 결과에서는 삼진 분류 모델이라는 특성 때문에 모델의 전체 성능 기준인 Accuracy만을 사용하였으나, 결함 데이터 집합은 결함 데이터가 비결함 데이터보다 매우 적고 또 중요도가 더 큰 불균형 데이터(imbalanced data) 형태이므로 삼진 분류 모델에 적합한 예측 오류 관련 성능 척도들을 정의하여 모델 성능을 평가할 필요가 있다.

V. 결 론

기존의 결함 예측 모델들은 대부분 입력 개체의 결함 여부만을 판단하는 이진 분류 모델이었다. 하지만 결함의 특성이 결함들마다 틀리므로 단순히 결함 유무만을 판단하는 모델의 유용성은 떨어진다. 결함 특성들 중 가장 중요한 것은 결함 심각도로, 심각도에 따라 결함들은 시스템을 멈추게 하는 치명적인 결함으로부터 매우 가벼운 결함까지로 나눠진다. 따라서 이러한 심각도 정보에 기반한 예측 모델은 결함 유무만을 판단하는 모델보다 프로젝트 관리 부분에 엄청나게 유용하게 사용될 수 있다.

본 논문에서는 심각도에 기반한 결함 예측 모델을 삼진 분류 모델 형태로 제안하였다. 출력 클래스는 고심각 결함, 저심각 결함, 비결함을 의미하며 모델 구축 및 실험에 사용한 NASA 데이터 집합의 결함 심각도 값에 따라 이들 출력의 정의를 두가지 방법으로 하였다. NASA 데이터 집합들 중 가장 모호성이 적은 JM1, PC4를 이용하여 모델을 구축하고 실험한 결과 Accuracy 평가 결과에서 역전파 신경망 모델이 가장 좋은 결과를 보였고 그 다음으로 판단 트리, Naïve Bayes 모델 순으로 좋은 예측 성능을 보였다. 향후 연구로 Accuracy 외에 삼진 분류 모델에 적합한 예측 오류 관련 성능 척도들을 정의하여 모델 성능을 평가할 것이다.

REFERENCES

[1] IEEE Standard Classification for Software Anomalies, IEEE Std. 1044-2009.

[2] C. Catal, "Software fault prediction:A literature review and current trends," Expert Systems with Applications, Vol.38, No.4, pp.4626-4636, 2011.

[3] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," Applied Soft. Computing Vol.27, pp.504-518, 2015.

[4] D. Radjenovic, M. Hericko, R. Torkar, and A. Zivkovic, "Software fault prediction metrics: A systematic literature review," Information Soft. Technology, Vol.55, pp.1397-1418, 2013.

[5] 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, 2012.

[6] R. Shatnawi and W. Li, "The effectiveness of software metrics in identifying error-prone classes in post-release software evolution process," Journal of Systems and Software, Vol.81, No.11, pp.1868-1882, 2008.

[7] 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, 2006.

[8] 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.

[9] T. Menzies and A. Marcus, "Automated Severity Assessment of Software Defect Reports," Proc. of ICSM'2008, pp.346-355.

[10] E. S. Hong, "A Metrics Set for Measuring Software Module Severity," Journal of The Korea Society of Computer and Information, Vol.20, No.1, pp.197-206, 2015.

[11] 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.

[12] E. S. Hong and M. K. Park, "Unsupervised learning model for fault prediction using representative clustering algorithms," KIPS

Trans. Software and Data Engineering, Vol.3, No.2, pp.57-64, 2014.

[13] S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design," IEEE Trans. Software Eng., Vol.20, No.6, pp.476-493, 1994.

[14] E. S. Hong, "Ambiguity Analysis of Defective-ness in NASA MDP data sets," Journal of the Korea Society of IT Services, Vol.12, No.2, pp.361-371, 2013.

[15] T. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," Neural Computation Vol.10, pp.1895-1924, 1998.

저 자 소 개

홍 의 석

1992: 서울대학교
계산통계학과 전산과학전공 학사.

1994: 서울대학교
계산통계학과 전산과학전공 석사.

1999: 서울대학교
계산통계학과 전산과학전공 박사

현 재: 성신여자대학교 IT학부 교수

관심분야: 소프트웨어 품질 예측,
소프트웨어 메트릭, MSR 등

Email : hes@sungshin.ac.kr