KIPS Tr. Software and Data Eng.
Vol.3, No.2 pp.57~64 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2014.3.2.57

Unsupervised Learning Model for Fault Prediction Using Representative Clustering Algorithms

Euyseok Hong† · Mikyeong Park††

ABSTRACT

Most previous studies of software fault prediction model which determines the fault-proneness of input modules have focused on supervised learning model using training data set. However, Unsupervised learning model is needed in case supervised learning model cannot be applied: either past training data set is not present or even though there exists data set, current project type is changed. Building an unsupervised learning model is extremely difficult that is why only a few studies exist. In this paper, we build unsupervised models using representative clustering algorithms, EM and DBSCAN, that have not been used in prior studies and compare these models with the previous model using K-means algorithm. The results of our study show that the EM model performs slightly better than the K-means model in terms of error rate and these two models significantly outperform the DBSCAN model.

Keywords : Fault-proneness, Fault Prediction Model, Unsupervised Learning, Clustering

대표적인 클러스터링 알고리즘을 사용한 비감독형 결함 예측 모델

홍 의 석† · 박 미 경††

요 약

입력 모듈의 결함경향성을 결정하는 결함 예측 모델 연구들은 대부분 훈련 데이터 집합을 사용하는 감독형 모델에 관련된 것들이었다. 하지만 과거 데이터 집합이 없거나 데이터 집합이 있더라도 현재 프로젝트와 성격이 다른 경우는 비감독형 모델이 필요하며, 이들에 관한 연구들은 모델 구축의 어려움 때문에 극소수 존재한다. 본 논문에서는 기존 비감독형 모델 연구들에서 사용하지 않은 대표적인 클러스터링 알고리즘인 EM, DBSCAN을 사용한 비감독형 모델들을 제작하여, 기존 연구들에서 사용한 K-means 모델과 성능을 비교하였다. 그 결과 오류율 면에서 EM이 K-means보다 약간 나은 성능을 보였으며, DBSCAN은 두 모델에 떨어지는 성능을 보였다.

키워드 : 결함경향성, 결함 예측 모델, 비감독형 학습, 클러스터링

1. 서 론

일반적인 결함 예측 모델은 메트릭 벡터로 정량화한 입력 모듈을 사용하여 모듈의 결함경향성을 결정하는 분류 모델(classification model)을 의미한다. 이들은 소프트웨어 프로세스에서 구현이 완료된 후에 나타날 문제 부분을 미리 찾아냄으로써 적절한 자원 할당, 테스트 프로세스 개선, 최적의 리팩토링 후보 결정, 테스트 프로세스 개선을 통한 품질 개선, 높은 신뢰성을 갖춘 시스템 구축 등을 가능케 한다[1]. 본 논문에서의 결함 예측 모델도 입력 모듈의 결함경향성을 예측하는 모델을 의미한다.

메트릭 기반 결함 예측 모델에 관한 연구들은 소프트웨어 복잡도 연구가 한창이던 1980년대부터 계속 되어 왔으며, 2000년대 중반부터 관련 연구 결과 발표들이 급속히 많아졌다[1]. 이와 같은 현상의 주된 이유는 연구에 사용할 수 있는 공개 데이터 집합의 등장 때문이다. 이들의 대표적인 예로, 2000년대 초에 NASA IV&V 메트릭스 데이터 프로그램[1]의 데이터가 공개되고 2005년에 소프트웨어 예측 모델에 관련된 공개 데이터 집합들을 관리하는 PROMISE 레포지토리[2] 운영이 시작되었다.

최근까지 제안된 대부분의 예측 모델들은 훈련 데이터를

※ 이 논문은 2013년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2010-0021902).
※ 이 논문은 제40회 한국정보처리학회 추계학술발표대회에서 '비감독형 학습 알고리즘을 사용한 결함예측모델'의 제목으로 발표된 논문을 확장한 것임.
† 종신회원 : 성신여자대학교 IT학부 교수
†† 준 회 원 : 성신여자대학교 IT학부 학사과정
논문접수 : 2013년 11월 25일
수 정 일 : 1차 2014년 1월 8일
심사완료 : 2014년 1월 26일
* Corresponding Author : Euyseok Hong(hes@sungshin.ac.kr)

1) http://mdp.ivv.nasa.gov
2) http://promise.site.uottawa.ca/SERepository

사용하여 학습하는 감독형 모델들이었다. 훈련 데이터란 입력 모듈 데이터와 그에 대한 답, 즉 결함경향성 데이터가 함께 있는 데이터 집합을 의미하며, 감독형 모델은 입력 데이터에 대해 해당 답이 나오도록 학습한 모델을 의미한다. 이들은 학습에 판별분석, 회귀 분석 등과 같은 통계 방법이나 신경망 등과 같은 인공지능 알고리즘들을 사용하였다. 하지만 많은 개발 집단들은 훈련 데이터를 보유하고 있지 않으며, 보유하고 있다 하더라도 훈련 데이터를 얻은 프로젝트와 현재 프로젝트의 특성이 다르면 데이터를 사용할 수 없다. 따라서 훈련 데이터가 없는 경우에 사용할 수 있는 비감독형 모델이 필요하며, [1]은 결함 예측 모델 분야의 중요한 향후 연구 주제로 비감독형 모델에 관한 연구를 제시하였다. 하지만 모델 제작의 어려움 등의 이유로 인하여, 감독형 모델에 관한 연구들이 매우 많은 데 비해 비감독형 모델에 관한 연구들은 극소수에 불과하다.

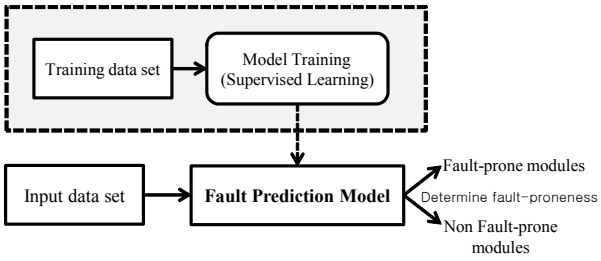

Fig. 1. Fault prediction model

Fig. 1은 입력 모듈들로 구성된 데이터 집합을 입력 받아 각 모듈의 결함경향성을 결정하는 결함 예측 모델의 기능을 표현한 것이다. 그림 윗부분은 감독형 모델에만 적용되는 부분으로 훈련 데이터 집합이 있는 경우 감독형 학습 알고리즘을 이용하여 훈련된 예측 모델을 사용하는 것을 나타내었다. 비감독형 모델은 학습할 훈련 데이터가 없으므로 현재 입력 데이터만을 모델 구축에 사용하여야 한다. 따라서 입력 데이터의 분포를 이용하여야 하며 이를 위해 사용된 기법은 클러스터링 알고리즘이다. 본 논문의 목적은 많지 않은 기존의 비감독형 모델 연구들에서 사용하지 않았던 EM(Expectation Maximization), DBSCAN(Density-based spatial clustering of applications with noise) 등과 같은 대표적인 클러스터링 방법들을 사용한 모델들을 제작하고 기존 모델들과 성능을 비교하여 제작 모델들의 효용성을 알아보는 것이다.

2장에서는 기존 비감독형 모델 연구들을 살펴보고, 3장에서는 모델 제작 방법 및 사용 시나리오에 대해 설명한다. 4장에서는 모델 성능 실험에 사용될 데이터와 평가 척도 및 실험 결과에 대해 언급하고, 5장에서는 결론을 기술한다.

2. 관련 연구

기존에 제안된 수많은 결함 예측 모델들의 대부분은 감독형 모델들이다. 사용된 훈련 알고리즘들은 판별분석, 선형 회귀분석, 로지스틱 회귀 분석 등과 같은 통계 기법이나 분류트리, 랜덤포리스트, 역전파 신경망, CBR(Case Based Reasoning), SVM(Support Vector Machine), 베이지안 분류 기법 등과 같은 인공지능 기법들이다[1, 2]. 이 연구들이 언급하지 않은 중요한 감독형 모델의 문제점은 대부분의 개발 집단이 훈련 데이터 집합을 보유하고 있지 않거나 제한된 형태로 보유하여 실제 개발 집단에서는 해당 모델들을 사용하기 어렵다는 것이다. 설사 훈련 데이터를 보유하고 있더라도 과거에 얻은 훈련 데이터가 현재 프로젝트의 데이터와 비슷한 분포를 가져야 모델을 사용할 수 있다는 제약이 있다. 이는 예측 모델이 훈련 데이터에 대한 의존성 때문에 다른 시스템에서 사용되기 어렵다는 모델의 범용성 실험 결과와 일치한다[3].

매우 많은 감독형 모델들이 제안되었지만 어느 모델이 가장 우수한 모델이라는 결론은 내려지지 않았다. 즉, 일반화된 모델은 존재하지 않는다. 왜냐하면 각 훈련 알고리즘들의 성능은 훈련 데이터 집합의 특성에 크게 의존하기 때문이다. 따라서 수많은 모델들 중 어떤 것을 선택하는가도 매우 어려운 문제이다. 그러므로 여러 모델들에 대한 선택부터 결함 예측, 평가를 지원하는 결함 예측 프레임워크에 대한 연구들도 등장하였다[4, 5].

비감독형 모델은 입력 데이터의 분석 과정 때문에 모델 구축이 어렵고, 예측 성능이 감독형 모델에 비해 떨어지기 때문에 매우 높은 필요성에도 불구하고 매우 극소수의 연구들만이 수행되었다. [6, 7]은 전문가가 각 클러스터의 대표 속성과 통계 정보를 토대로 결함경향성을 결정하여 클러스터링 알고리즘들의 성능을 비교하였다. 클러스터링 알고리즘으로 Neural-Gas, K-means를 사용하였는데 Neural-Gas 알고리즘은 K-means보다 전체 오류율은 좋지만 속도 면에서 좋지 않았다. [8]은 비감독형 학습 신경망인 SOM(Self Organizing Map)을 이용한 모델을 제시하고, 모델의 예측 성능을 많이 사용되는 감독형 모델인 오류역전파 신경망 모델과 비교하였으나 실제 데이터를 사용하여 검증을 하지 못했다는 문제점이 있다. 이들은 비감독형 모델의 시초가 되는 연구인 점에서 의미가 있지만 가장 좋은 클러스터링 결과를 얻기 위해 전문가에게 의존해야하고 이를 통하여 클러스터 수를 인위적으로 설정해야 한다는 문제점이 있다.

[9]와 [10]은 클러스터 수가 자동으로 결정되는 클러스터링 알고리즘을 사용하였다. [9]는 X-means 알고리즘을 사용한 모델이 Pure Metrics Thresholds methods, Fuzzy C-means, K-means를 사용한 모델들보다 나은 성능을 보임을 알아내었다. [10]은 K-means의 변형된 형태인 QDK(Quad Tree-based K-means) 알고리즘을 이용하여 K-means의 문제들 중 하나였던 불안정한 초기 클러스터의 중심 할당 문제를 안정화시킨 모델을 제안하였으며, K-means, GM(Global K-means), DD(SAS2004) 초기화 알고리즘 보다 성능이 좋음을 밝혀내었다. 그러나 이 두 연구들은 PROMISE에서 연구에 많이 사용되는 NASA 데이터 집합을 사용하지

않고, 입력 모듈의 수가 상대적으로 적은 AR3, AR4, AR5 데이터 집합을 이용했다는 문제점이 있다.

또 다른 최근 연구 방향은 제한된 훈련 데이터 집합이 존재할 때 이를 이용할 수 있는 예측 모델들에 관한 연구들이다. 즉 정확한 모델을 만들기에는 충분치 않은 훈련 데이터가 존재하여, 출력 결과 값을 모르는 데이터와 결과 값을 아는 데이터가 함께 존재하는 경우이다. 이런 경우에는 이 두 가지 형태의 데이터들을 사용하기 위해 세미 감독형 학습 기법이 사용된다. [11]은 [6]에서 사용한 기법에 더 정확한 K-means 클러스터링을 위해 초기 클러스터 위치를 지정해주는 기법을 사용하였다. [12]는 EM을 사용하였으며, [13]은 작은 데이터 집합을 사용하는 경우에는 Naïve Bayes 알고리즘이 모델 구축에 가장 적절하고, 큰 데이터 집합을 사용하는 경우에는 두 단계를 거쳐 모듈 구축을 하는 메타 모델인 YATSI(Yet Another Two Stage Idea) 알고리즘이 기본 알고리즘의 성능을 높일 수 있다는 것을 보였다. 세미 감독형 모델은 비감독형 모델에 비해 분석 단계에서 전문가가 더 나은 라벨링을 할 수 있지만 여전히 전문가에게 의존하고, 클러스터 수 설정에서 자유롭지 않다는 한계점도 있다.

3. 모델 제작

3.1 사용 클러스터링 알고리즘

본 논문의 목적은 기존 연구에서 사용하지 않았던 클러스터링 알고리즘들 중에서 많이 사용되는 대표적인 알고리즘들을 사용한 모델을 제작해 평가하는 것이다. Fig. 2는 클러스터링 알고리즘들을 분류한 것이다.

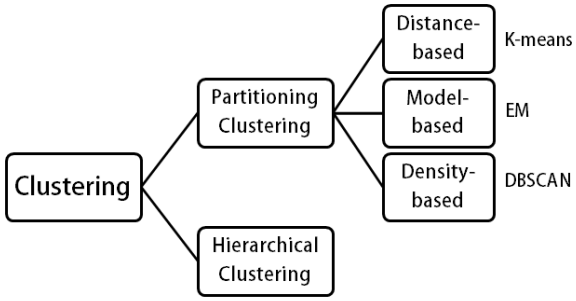

Fig. 2. Clustering algorithms categories

클러스터링 방법은 크게 분할 클러스터링(Partitioning clustering)과 계층적 클러스터링(Hierarchical clustering)의 두 가지로 구분되고, 각각의 클러스터링은 거리 기반 알고리즘, 밀도 기반 알고리즘 그리고 모델 기반 알고리즘으로 나뉜다[14]. 본 논문에서는 클러스터링을 하기 위해 WEKA[3] 데이터 마이닝 도구를 사용하였다. WEKA는 대부분의 클러스터링 알고리즘들을 제공하는데, 이들 중 각 클러스터링 분류에 속하는 대표적인 알고리즘을 분류당 하나씩 선정하였다. 그 결과 분할 클러스터링 중 거리기반 알고리즘에서는 K-means, 밀도 기반 알고리즘에서는 DBSCAN, 모델 기반 알고리즘에서는 EM 알고리즘을 선정하였다. 계층적 클러스터링은 여러 고사양 PC 플랫폼에서 실험하였지만 적절한 학습 결과를 얻지 못하여 실험에서 제외하였다. 2장에서 기술한 비감독형 모델들 중 실제 데이터를 사용한 [6, 7], [9], [10]은 모두 K-means 또는 이를 변형한 알고리즘을 사용한 모델이므로 본 논문에서는 DBSCAN과 EM을 사용한 모델이 K-means와 비교해 어떤 예측 성능을 보이는 지를 실험한다.

3.2 모델 제작 및 사용 시나리오

클러스터링을 이용한 비감독형 모델 제작 시 사용 데이터 집합은 각 모듈에 대한 출력값(결함 유무)이 없는 상태이다. Fig. 3은 정제되지 않은 초기 입력 데이터 집합을 가지고 데이터 전처리 - 클러스터링 - 분석 과정을 거쳐 비감독형 모델을 구축하고 이를 이용하여 결함경향성을 예측하는 전체 프로세스를 나타낸 것이다.

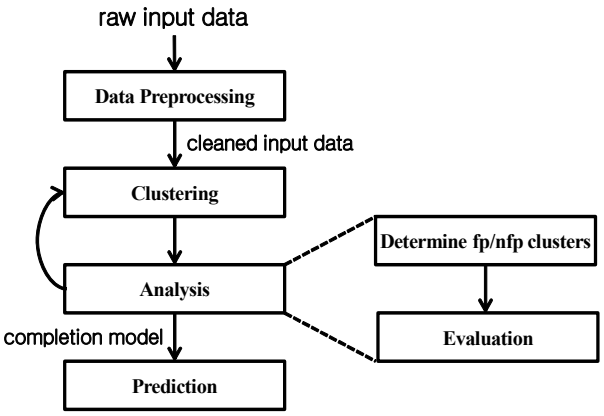

Fig. 3. Model construction process

1) 데이터 전처리 단계

개발 집단에서 예측하고자 하는 품질, 즉 결함경향성,의 대상은 설계 명세나 초기 코드이고 예측을 위해서는 이들을 소프트웨어 메트릭 벡터 형태로 정량화 하여야 한다. 정량화된 초기 데이터(raw data)는 바로 모델의 입력으로 사용되지 않고, 데이터 정제(data cleaning) 작업을 하는 데이터 전처리 과정을 거치게 된다. 데이터 정제 작업이란 데이터를 사용하기 전에 초기 데이터에 존재하는 여러 문제점들을 찾아 해결하여 문법적으로나 의미적으로 완전한 데이터 집합을 만드는 과정이다. 이는 모델의 예측 성능에 매우 큰 영향을 끼치므로 올바른 정제 작업은 매우 중요하다[15]. 이 단계는 모든 케이스가 같은 값을 갖는 상수 속성 삭제, 같은 의미를 갖는 중복 속성 삭제, 데이터 값이 존재하지 않는 결측값 처리, 속성 간의 비정상적인 관계나 비정상적인 속성값을 갖는 경우를 없게 하는 데이터 무결성 처리, 중복

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

또는 모순 케이스 처리 작업으로 구성된다. 전처리 과정에는 정제 작업 외에 모든 데이터 값을 [0, 1]로 바꾸는 정규화(normalization)와 메트릭 벡터의 차원이 큰 경우 효율적인 수행을 위한 차원 축소 과정이 포함된다.

2) 클러스터링 단계

클러스터링 단계에서는 정제된 전체 입력 데이터 인스턴스들을 유사한 인스턴스들끼리 클러스터로 묶어주는 작업이 수행된다. 모델 구축 프로세스의 예를 보여주는 Fig. 4에서 점은 하나의 데이터 인스턴스이며 하나의 모듈을 의미한다. 흰 점은 비결함경향 모듈을, 검은 점은 결함경향 모듈을 나타내며, 예에서는 클러스터링 후 모듈들이 5개의 클러스터로 클러스터링 되었다.

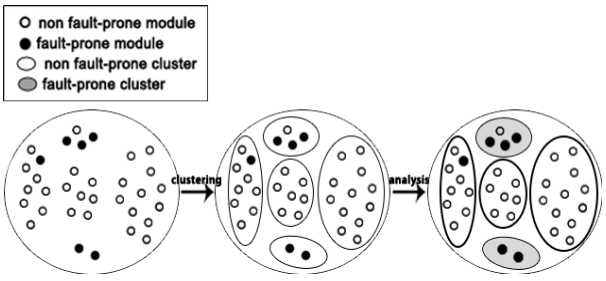

Fig. 4. Example of the model construction

3) 분석 단계

분석 단계에서는 클러스터링 단계에서 결정된 클러스터들에 대한 전문가의 분석이 이루어진다. Fig. 3에서 분석 단계를 두 단계로 나눈 것처럼 분석 과정에서 결함경향 클러스터와 비결함경향 클러스터가 결정되며, 각 클러스터의 성질과 다른 성질을 갖는 모듈의 비율을 계산함으로써[4] 제작 모델의 평가가 이루어진다. 전문가의 판단은 결정론적인 알고리즘에 의한 것이 아니므로 주관적인 결과를 나을 수 있지만 입력 데이터에 대한 출력값을 알 수 없으므로 클러스터의 해석을 자동하는 것은 불가능하다. 따라서 기존 비감독형 모델 연구들은 모두 전문가의 분석으로 이 단계를 수행했으며 이 점이 비감독형 모델의 가장 중요한 한계점이다. 이로 생기는 문제점을 줄이기 위해 전문가는 반드시 입력 데이터를 구성하는 소프트웨어 메트릭 전문가여야 하며, 두 사람 이상이 작업을 하여 서로 교차 검증을 하여야 한다. Fig. 4의 분석 단계를 살펴보면 두 개의 클러스터를 결함경향 클러스터로 결정하였으며 실제 이 클러스터들에 결함경향 모듈이 많이 있으므로 올바른 분석을 하였음을 알 수 있다.

클러스터링 알고리즘은 대부분 클러스터 수가 자동으로 결정되지 않는다. 클러스터 수가 너무 작으면 패턴 분류가 되지 않아 결함경향 모듈들과 비결함경향 모듈들이 같은 클러스터에 속할 가능성이 있고, 클러스터 수가 너무 크면 사람의 분석 노력이 많이 든다. 따라서 Fig. 3에 그려진 사이클과 같이 여러 번의 클러스터링-분석 단계(보통은 클러스터 수를 변화해 보는 것)를 거쳐 가장 좋은 평가를 얻은 결과가 최종 완성된 모델이 된다.

4) 예측 단계

모델 제작 프로세스는 현재 입력 데이터 집합을 가지고 이루어지므로 완성 모델이 결정되면 입력 모듈의 예측도 끝나게 된다. 결함경향 클러스터에 속하는 모듈들은 결함경향 모듈로 비결함경향 클러스터에 속하는 모듈들은 비결함경향 모듈로 예측된 것이다. Fig. 4의 예를 보면 두 개의 결함경향 클러스터들 중 위에 있는 클러스터에는 세 개의 결함경향 모듈과 한 개의 비결함경향 모듈이 존재하는데 비결함경향 모듈은 결함경향 모듈로 예측되므로 모델의 예측 오류가 된다.

가끔 완성된 모델이 새로운 입력 모듈의 결함 예측에 사용되어야 하는 경우가 생긴다. 매우 큰 입력 데이터 집합의 부분 집합을 가지고 모델을 제작한 경우나 소프트웨어 진화 프로세스에서 이전 릴리즈를 사용하여 완성된 모델에 다음 릴리즈의 입력 데이터를 사용해야 하는 경우 등이 이에 해당한다. 이와 같은 경우 완성된 모델에 새로운 입력 모듈이 들어오면 가장 가까운 클러스터의 성질에 따라 입력 모듈의 결함경향성이 결정된다. Fig. 5에서 각 입력 모듈은 가장 가까운 클러스터의 결함경향성으로 결정된다. 즉, 세 개의 입력 모듈 중 위, 아래 두 개의 모듈은 결함경향 모듈, 다른 하나는 비결함경향 모듈로 예측된다.

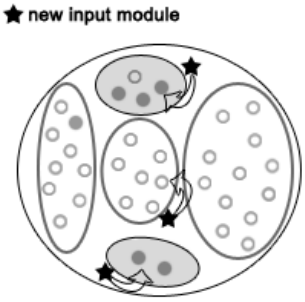

Fig. 5. Prediction of the new input modules

4. 모델 성능 실험

4.1 데이터 집합 및 속성 선정

실험에는 공개 데이터 집합들 중 결함 관련 연구에 가장 많이 사용된 NASA MDP 데이터 집합을 사용하였다. NASA 데이터 집합은 NASA에서 제공한 원본과 이들 중 일부를 정제하여 PROMISE에 넣은 두 가지 버전이 존재하며, PROMISE 버전 역시 초기 버전과 데이터 정제 작업을 많이 거쳐 사이즈가 축소된 최근 버전이 존재한다. 본 논문에서는 PROMISE 초기 버전을 사용하였으며, 데이터 집합

4) 모듈의 성질을 사람이 판단하므로 입력 데이터 집합이 큰 경우 정확한 계산은 어렵지만 개략적인 결과로 평가가 가능하였다.

을 구성하는 여러 프로젝트들 중에서는 JM1을 선택하였다. 선정 이유는 결측값이 적고, 모듈수가 많으며 결함에 관련된 속성들의 모호성이 적기 때문이다[16]. JM1 프로젝트는 총 10,885개의 모듈로 구성된 C 언어로 구현된 프로젝트이다. 그 중 결함경향 모듈은 2,106개. 비결함경향 모듈은 8,779개이다.

전처리 과정에서 정규화한 데이터에 WEKA에서 지원하는 모든 속성 선정 기법들을 적용하였다. 이는 클러스터링 알고리즘을 효율적으로 수행하기 위해 입력 데이터의 차원을 축소하기 위한 과정이다. 그 결과 8개 속성으로 선정되어 가장 의미 있는 결과를 보인 CfsSubsetEval과 21개 전체 속성을 사용하는 두 가지 경우를 실험에 사용하였다. FilteredSubsetEval은 CfsSubsetEval과 같은 결과를 내었고 PrincipalComponenets는 기존의 변수들을 가중 평균 형태로 새로운 변수들을 생성하기 때문에 WEKA에서 사용할 수 없었다. 21개 속성들의 처음 4개는 cyclomeatic complexity, 즉 v(g)로 대표되는 McCabe 기반 메트릭들, 그 뒤의 11개는 halstead volume(v), halstead length(l) 등과 같은 Halstead 기반 메트릭들, 마지막 6개는 코드 사이즈 관련 메트릭들이다. 전체 속성들로 구성된 메트릭 벡터는 다음과 같고, 차원 축소 결과는 Table 1에 나타내었다.

(loc, v(g), ev(g), iv(g), n, v, l, d, i, e, b, t, lOCode, lOComment, lOBlank, lOCodeAndComment, uniq_Op, uniq_Opnd, total_Op, total_Opnd, branchCount)

Table 1. Selected results of attribute selection

Atrribute Evaluator	Selected attributes
CfsSubsetEval	1,2,3,4,9,14,15,16: 8
PrincipalComponenets	1,2,3,4,5,6,7,8 : 8
FilteredSubsetEval	1,2,3,4,9,14,15,16: 8

4.2 평가 실험

1) 클러스터링 단계

K-means와 EM 알고리즘의 실행을 위해서는 클러스터 수 K(numClusters)를, DBSCAN은 사용자 정의 거리 eps(epsilon)와 임계값 minPt(minPoints)를 설정해야 한다. 주어진 값들에 따라 클러스터링 결과는 완전히 달라진다. K-means, EM의 K를 2부터 최대 20개로 설정하였으며, 이들과의 비교를 위해 DBSCAN의 클러스터 수도 2개 이상 20개 이하로 클러스터링 되도록 eps와 minPt 값을 설정하였다. 그 결과 eps가 0.01, 0.03, 0.06인 경우와 각각에 대해 minPt가 10, 50, 100인 경우를 실험하였다.

2) 분석 단계의 자동화

평가 실험에서 실험의 용이성을 위하여 분석 단계 중 클러스터링 결과를 결함경향 클러스터와 비결함경향 클러스터로 결정하는 부분을 자동화하였다. 자동화가 가능한 이유는 비감독형 모델이 사용하는 일반적인 입력 데이터 집합과는 달리 JM1 프로젝트는 출력 값이 존재하는 데이터 집합이기 때문이다.

Fig. 6은 결함경향 클러스터를 자동으로 결정하는 알고리즘을 기술한 것이다. 알고리즘에서 사용한 변수와 함수들의 정의는 다음과 같다: *RC*(Remaining Clusters)는 WHILE 루프를 거치면서 처리되지 않은 현재 클러스터 집합을, *FPC*(Fault-Prone Clusters)는 결정된 결함경향 클러스터 집합을 의미한다. *maxFR*은 알고리즘 사용자가 정하는 값으로 전체 모듈에 대한 결함경향 모듈의 비율의 상한값을 의미하며, 본 논문의 실험에서는 20%로 하였다. 따라서 알고리즘에 의해 선정되는 결함경향 모듈은 전체 모듈의 20% 내에서 결정된다. R_{CS}는 전체 모듈에 대한 클러스터 집합 *CS*에 속한 모듈의 비율이며, $r_{fpm}(c)$와 $r_{nfpm}(c)$는 클러스터 *c*에 속하는 모듈들 중 결함경향 모듈과 비결함경향 모듈의 비율을 반환한다. 마지막으로 *maxNfp*(*CS*)는 클러스터 집합 *CS*에서 r_{fpm}이 가장 큰 클러스터를 반환하며, *CS*가 ∅인 경우는 null을 반환한다.

```
RC ← all clusters
FPC ← ∅
maxc ← maxNfp(RC)

WHILE RC !=∅ AND r_fpm(maxc) ≥ r_nfpm(maxc) DO
   IF R_{FPC∪{maxc}} ≤ maxFR THEN
      FPC ← FPC U {maxc}
   ENDIF
   RC ← RC - {maxc}
   maxc ← maxNfp(RC)
ENDWHILE
```

Fig. 6. Algorithm to determine FP clusters

Fig. 6 알고리즘은 결함경향 클러스터 집합인 *FPC*를 구하는 것이 목적이다. 현재 *RC*에서 r_{fpm}이 가장 큰 클러스터를 *maxc*로 하고, *maxc*가 *FPC*에 들어갈 수 있는지를 검사하여 조건을 만족하면 *FPC*에 포함시키는 것이 WHILE 루프의 한 단계이다. *FPC*에 들어갈 수 있는 조건은 *maxc* 내의 결함경향 모듈 비율이 비결함경향 모듈 비율보다 크고, *maxc*가 포함된 *FPC*의 전체 모듈에 대한 비율이 *maxFR*보다 작아야 한다는 것이다. 루프는 더 이상 고려 대상인 클러스터가 없거나, 선정된 *maxc*의 비결함경향 모듈 비율이 결함경향 모듈 비율보다 커지면 종료된다. Table 2는 알고리즘이 Fig. 4의 클러스터들에 적용된 결과를 나타낸다. #은 클러스터의 모듈수를, #nfpm과 #fpm은 클러스터의 비결함경향 모듈수와 결함경향 모듈수를 의미한다. 처음 *maxc*는 r_{fpm}이 가장 큰 클러스터 3이 되며, R_3은 6%로 *maxFR* 값인 20%보다 작으므로 클러스터 3은 *FPC*에 들어간다. 루프의 다음 단계의 *maxc*는 클러스터 1이 되며 R_1은 11%로, 기존 *FPC*에 들어가도 R_{FPC}는 17%로 20%보다 작으므로

클러스터 1도 *FPC*에 들어간다. 그 다음 *maxc*는 클러스터 0이 되는데 $r_{fpm}(0)<r_{nfpm}(0)$이므로 클러스터 0은 *FPC*에 들어가지 못하고 WHILE 루프는 끝난다. 결과적으로 결함경향 클러스터 결과는 클러스터 3과 1로 Fig. 4의 그림 결과와 일치한다.

Table 2. Example of determining FP clusters

Cluster(K=5)			nfpm		fpm	
c	#	R_C	#nfpm	$r_{nfpm}(c)$	#fpm	$r_{fpm}(c)$
0	9	26%	8	89%	1	11%
1	4	11%	1	25%	3	75%
2	7	20%	7	100%	0	0%
3	2	6%	0	0%	2	100%
4	12	34%	12	100%	0	0%

3) 평가 척도

예측 모델의 성능 평가 수행을 위해 많은 평가 척도가 사용되고 있으나 본 논문에서는 [6]에서 사용한 TER(오류율, Total Error Rate), FPR(False Positive Rate), FNR(False Negative Rate)을 사용한다. TER은 전체 모듈 중에서 잘못 예측한 모듈의 비율이다. FPR은 비결함경향 모듈을 결함경향 모듈로 예측하는 오류율을, FNR은 결함경향 모듈을 비결함경향 모듈로 예측하는 오류율을 뜻한다. 결함경향이 있는 모듈을 비결함 모듈로 예측하는 것은 소프트웨어 품질에 심각한 영향을 주기 때문에 FNR이 FPR보다 신중히 살펴야 할 오류 척도이다.

Table 3. Confusion matrix

		Predicted Class	
Actual Class		Fault-Prone	Non Fault-Prone
	Fault-Prone	True Positive(A)	False Negative(C)
	Non Fault-Prone	False Positive(B)	True Negative(D)

$$TER=\frac{B+C}{A+B+C+D} \quad FPR=\frac{B}{B+D} \quad FNR=\frac{C}{A+C}$$

4.3 실험 결과

속성 선정은 전체 속성을 택한 경우가 CfsSubsetEval로 선정된 경우보다 좋은 성능을 보였기에 실험 결과는 전체 속성을 사용한 결과를 나타내었다. TER은 모델의 전체적인 예측 성능을 나타내는 값이지만 FNR이 상대적으로 FPR보다 중요하므로 TER이 낮고 FNR이 높은 모델이 좋은 성능을 보인다고 할 수는 없다. 즉, 모델 평가는 TER의 관점과 FNR의 관점에서 이루어져야 한다.

Table 4와 Table 5는 K-means와 EM의 실행 결과 중 의미 있는 성능을 보인 것들을 나타낸 것이다. K-means의 TER과 FPR은 클러스터 수 K가 4일 때 가장 낮았고, FNR은 K가 11일 때 가장 낮았다. 이를 NASA의 JM1을 사용한 [6]의 실험 결과[5]와 비교해 보았을 때 K가 4인 경우는 TER과 FPR이 더 좋은 대신 FNR이 좋지 않았고, K가 11인 경우는 FPR이 조금 좋지 않았지만 나머지는 비슷한 결과를 보였다. 이 같은 차이는 [6]에서 평가 실험에 사용한 JM1은 NASA 데이터 원본을 정제한 것으로 본 실험에서 사용한 PROMISE의 JM1과 다른 데이터 정제 작업을 거쳤기 때문이라 판단된다.

Table 4. Experimental results of K-means

K	TER	FPR	FNR
4	19.6	4	84.66
6	25.93	15	71.51
11	25.71	17.33	60.64
13	23.2	12.6	67.38
15	23.32	13.46	64.39

Table 5. Experimental results of EM

K	TER	FPR	FNR
4	19.29	0	99.57
6	20.34	17.09	61.68
11	20.5	6.74	77.84
13	25.83	17.8	59.31
15	20.29	6.49	77.83

K-means와 마찬가지로 EM 역시 K가 4인 경우 가장 좋은 TER과 FPR값을 보였지만 FNR은 극도로 나쁜 결과를 보였다. FNR은 K가 13인 경우에 가장 좋았지만 K가 6인 경우와 큰 차이는 없었다. 전체적으로 보면 K가 같은 경우, EM이 K-means에 비해 TER 면에서는 좋은 결과를 보이고, FNR 면에서는 좋은 결과를 보이지 못했다. 하지만 그 차이는 매우 미미하므로 본 실험 결과로는 K-means와 EM 중 어느 것이 유의미하게 좋은 성능을 보인다고 할 수는 없다.

Table 6. Experimental results of DBSCAN

eps \ minPt		0.01	0.03	0.06
10	TER	27.8	21.4	26.2
	FPR	13.4	9.31	14.9
	FNR	88.1	71.6	73.5
50	TER	33.4	23.3	26.3
	FPR	22.1	13.1	16
	FNR	85.0	65.8	69.5
100	TER	69.1	25.6	22.1
	FPR	5.22	16.6	7.7
	FNR	84.4	63.2	82.3

5) [6]의 결과는 그래프로 나타나있어 정확한 수치를 판단하기 어렵지만 대략 (TER, FPR, FNR)은 (23, 13, 62)로 판독된다.

DBSCAN은 K-means, EM과 달리 클러스터 수를 설정하지 않고 거리 eps와 임계값 minPt을 설정하여 클러스터링한다. Table 6은 eps가 0.01, 0.03, 0.06일 때 minPt가 10, 50, 100인 9가지 경우의 결과이다. 9번의 클러스터링 실험 모두에서 클러스터링이 되지 않은 모듈들이 존재하였는데, 그러한 모듈들은 하나의 클러스터로 할당하여 실험 결과를 내었다. TER이 가장 낮은 결과는 21.4이지만 전체적인 TER의 값은 K-means, EM보다 높다. FNR의 가장 낮은 결과는 63.2로 역시 K-means, EM의 가장 낮은 결과보다 높지만 전체적으로 볼 때는 안 좋다고 보기는 어렵다. DBSCAN이 약간 낮은 성능을 보이는 것은 클러스터링이 안 된 부분들이 존재하기 때문이라 판단된다.

5. 결론 및 향후 연구

지난 수십 년간 매우 많은 연구들이 결함 예측 분야에서 수행되었지만 비감독형 모델에 관한 연구의 수는 매우 극소수이다. 하지만 결함 예측 모델의 중요성이 점점 커지면서 비감독형 모델의 필요성이 급속도로 증가하고 있다. 왜냐하면 대부분의 개발 회사들은 훈련 데이터 집합을 보유하고 있지 않기 때문에 모델의 훈련 과정에 이들을 필요로 하는 감독형 모델을 사용할 수 없기 때문이다. 본 논문은 기존 연구들에서 사용하지 않은 대표적인 클러스터링 알고리즘들인 EM, DBSCAN을 이용한 비감독형 모델을 제작하여 기존 연구들에서 사용한 K-means 모델과 성능을 비교하였다. 그 결과 전체 오류율 측면에서는 EM이 K-means보다 아주 약간 나은 결과를 보였으며 DBSCAN은 두 모델보다 좋지 않은 결과를 보였다. 또 다른 중요한 오류 측정치인 FNR 측면에서는 어느 모델이 낫다고 할 수 없었다. 따라서 기존 연구들이 사용한 K-means 대신 EM 모델을 사용가능하며, 자동으로 클러스터 수가 결정되는 EM 모델을 사용하면 보다 쉽게 비감독형 모델을 구축할 수 있다는 결론을 얻었다.

향후 연구로는 비감독형 모델의 구축에서 가장 많은 비용과 노력이 드는 분석 단계의 효율적인 수행을 위해 클러스터 수가 자동으로 결정되는 클러스터링 알고리즘들을 사용한 모델들을 제작하여 성능을 비교하고, 이들 모델들과 세미 감독형 모델들의 성능이 감독형 모델들의 성능과 비교하여 어느 정도에 도달하는지를 실험을 통해 알아볼 것이다.

참 고 문 헌

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

[2] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, “A Systematic Literature Review on Fault Prediction Performance in Software Engineering,” *IEEE Trans. Software Engineering*, Vol.38, No.6, pp.1276-1304, 2012.

[3] T. Y. Kim, Y. K. Kim, and H. S. Chae, “An Experimental Study of Generality of Sogtware Defects Prediction Models based on Object Oriented Metrics,” *The KIPS Transactions: Part D*, Vol.16, No.3, pp.407-416, 2009.

[4] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, “Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings,” *IEEE Trans. Software Engineering*, Vol.34, No.4, pp. 485-496, 2008.

[5] Q. Song, Z. Jia, M. Shepperd, S. Ying and J. Liu, “A General Software Defect-Proneness Prediction Framework,” *IEEE Trans. Software Engineering*, Vol.37, No.3, pp.356-370, 2011.

[6] 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.

[7] S. Zhong, T. M. Khoshgoftaar, and N. Seliya, “Unsupervised learning for expert-based software quality estimation,” *Proc. of HASE*, 2004.

[8] E. S. Hong, “A software quality prediction model without training data set,” *The KIPS Transactions: Part D*, Vol.10, No.4, pp.689-696, 2003.

[9] C. Catal, U. Sevim, and B. Diri, “Software fault prediction of unlabeled program modules,” *Proc. of WCE*, 2009.

[10] 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.

[11] N. Seliya and T. M. Khoshgoftaar, “Software quality analysis of unlabeled program modules with semisupervised clustering,” *IEEE Trans. Systems, Man and Cybernetics*, Vol.37, No.2, pp.201-211, 2007.

[12] N. Seliya and T. M. Khoshgoftaar, “Software quality estimation with limited fault data: A semi supervised learning perspective,” *Software Quality Journal*, Vol.15, No.3, pp.327-344, 2007.

[13] C. Catal and B. Diri, “Unlabeled Extra Data do not Always Mean Extra Performance for Semi-Supervised Fault Prediction,” *Expert Systems*, Vol.26, No.5, pp.458-471, 2009.

[14] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*, Prentice Hall, 1988.

[15] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, “Reflections on the NASA MDP data sets,” *IET Software*, Vol.6, No.6, pp.549-558, 2012.

[16] E. S. 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, 2013.

홍 의 석

e-mail : hes@sungshin.ac.kr
1992년 서울대학교 계산통계학과(학사)
1994년 서울대학교 계산통계학과(석사)
1999년 서울대학교 전산과학과(박사)
1999년~2002년 안양대학교 디지털미디어 학부 교수
2002년~현 재 성신여자대학교 IT학부 교수
관심분야 : Software Quality Prediction, Software Metrics, Mining Software Repositories

박 미 경

e-mail : parkmikyeong@gmail.com
2011년~현 재 성신여자대학교 IT학부 학사과정
관심분야 : Software Engineering, Software Quality, Data mining