베이지안 분류기를 이용한 소프트웨어 품질 분류*

홍 의 석**

Software Quality Classification using Bayesian Classifier*

Euyseok Hong**

■ Abstract ■

Many metric-based classification models have been proposed to predict fault-proneness of software module. This paper presents two prediction models using Bayesian classifier which is one of the most popular modern classification algorithms. Bayesian model based on Bayesian probability theory can be a promising technique for software quality prediction. This is due to the ability to represent uncertainty using probabilities and the ability to partly incorporate expert's knowledge into training data. The two models, NaïveBayes(NB) and Bayesian Belief Network(BBN), are constructed and dimensionality reduction of training data and test data are performed before model evaluation. Prediction accuracy of the model is evaluated using two prediction error measures, Type I error and Type II error, and compared with well-known prediction models, backpropagation neural network model and support vector machine model. The results show that the prediction performance of BBN model is slightly better than that of NB. For the data set with ambiguity, although the BBN model's prediction accuracy is not as good as the compared models, it achieves better performance than the compared models for the data set without ambiguity.

Keyword : Software Quality, Prediction Model, Naïve Bayes, Bayesian Belief Network

논문투고일 : 2012년 01월 27일 논문수정완료일 : 2012년 03월 08일 논문게재확정일 : 2012년 03월 13일
* 이 논문은 2011년도 성신여자대학교 학술연구조성비 지원에 의하여 연구되었음.
** 성신여자대학교 IT학부 교수

1. 서 론

소프트웨어 품질 예측은 분석이나 설계와 같은 개발 초기 단계의 산물을 사용하여 소프트웨어 개발 산물의 품질 인자들을 예측하는 작업이다. 기존의 연구들을 보면 예측하려는 품질 인자는 대부분 소프트웨어 결함에 관련된 것들이었으며 구현 시 발생하는 정확한 결함 수 예측보다는 결함 발생 유무를 예측하는 결함경향성 예측에 관한 것들이 대부분이었다. 이는 정확한 결함수를 예측하는 모델을 제작하기 어렵다는 점과 결함경향성 판단만으로도 충분히 개발 프로세스에 유용한 정보로 활용할 수 있다는 점에 기인한다. 따라서 대부분의 품질 예측 모델들은 메트릭 벡터들로 설계 개체들을 정량화 한 후 이들을 위험 그룹과 비위험 그룹으로 분류하는 분류 모델들이었다. 설계 개체의 위험도(criticality)란 개체가 구현되었을 때의 결함 경향성을 의미한다[1]. 이들 예측 모델들은 주로 과거에 얻은 위험도 결과 데이터를 학습하여 현재 프로젝트에 사용하는 훈련 모델들이었으며 훈련 알고리즘으로는 복잡한 통계 기법들이나 인공지능 기법들을 사용하였다.

위험도를 예측하는 모델은 구현 소프트웨어에서 결함을 많이 일으킬 부분들을 초기 단계에서 미리 찾아 적절한 자원 할당을 가능케 함으로써 소프트웨어 개발 전체 비용을 낮추고 구현된 소프트웨어의 품질을 높여준다. [2]는 이와 같은 장점들을 다음과 같이 정리하였다.

- 높은 신뢰성을 갖춘 시스템 구축
- 문제 모듈에 집중 통해 테스트 프로세스 개선
- 여러 설계 대안들 중 최적안 결정
- 문제 모듈들을 리팩토링 후보로 결정
- 테스트 프로세스 개선을 통한 품질 개선

품질 예측 모델들에 대한 필요성은 널리 인식되었지만 2000년대 초까지는 연구들이 많이 진행되지 않았다. 국제 저널에 발표된 연구들을 보면 1980년대 후반부터 2000년까지는 매년 5개 미만의 연구가 발표되었으며 연구 논문이 한 개도 발표되지 않은 해도 많았다. 하지만 2000년대 중반으로 오면서 매년 10개에 가까운 연구가 발표되었고 2005년 이후에는 그 수가 급격히 늘어나고 있다[2]. 이 이유는 공개 데이터 집합의 등장 때문이다. 대부분의 예측 모델들이 훈련 데이터 집합을 필요로 하는 훈련 모델이다 보니 모델 제작을 위한 데이터 집합이 필요한데 2000년대 초반까지는 매우 극소수의 집단만이 자신들의 데이터 집합으로 연구를 수행하였다. 하지만 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 시작됨으로써 이들을 사용한 연구가 빠른 속도로 증가하고 있다.

최근까지 많은 연구들이 행해져왔지만 하나의 연구 결과를 보고 어느 모델이 타 모델보다 일반적으로 낫다고 하기는 어렵다. 왜냐하면 훈련 모델들은 훈련 데이터 집합의 특성에 크게 의존하기 때문이다[3]. 본 논문의 목적은 최근까지 예측 모델 연구들에서 매우 좋은 성능을 보인 Naïve Bayes(NB) 분류기와 그다지 많은 연구가 행해지지는 않은 Bayesian Belief Network(BBN) 분류기를 [4, 5]에서 사용한 데이터 집합을 이용하여 제작하여 해당 데이터 집합에서 좋은 성능을 보였던 역전파 신경망 모델(BPM) 및 Support Vector Machine(SVM) 모델과 비교해 보는 것이다.

베이지안 분류기는 모델의 입력 메트릭 속성들과 출력 속성 사이의 확률적 관계를 모델링하기 위해 베이스 정리를 사용하며, 각 입력 속성 값에 대한 출력 속성의 사후 확률을 계산하여 높은 확률을 갖는 쪽으로 입력 개체를 분류한다. 입력 속성들의 의존성 허용 여부에 따라 NB와 BBN 두 가지 모델로 나뉘며, 두 개의 베이지안 모델들을 모

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

두 구현한 연구들은 매우 적었지만 그 중 본 논문의 중요한 연구 동기를 준 것은 [6]의 결과이다. [6]은 SVM 모델의 유용성을 보이기 위해 NASA 데이터 집합들 중 4개의 프로젝트 데이터를 이용해 SVM을 포함한 9개의 모델을 구축하고 그들의 성능을 Accuracy, Precision, Recall, F-measure의 네가지 결과 값으로 평가하였다. 그 결과 베이지안 모델은 다른 측정치에서는 좋은 성능을 나타내지 못했지만 Precision에서는 두 베이지안 모델들이 네 개 프로젝트 모두에서 SVM보다 유의미하게 나은 성능을 보였다.

제 2장에서는 최근까지 연구된 대표적인 품질 분류 모델들과 베이지안 분류기를 이용한 모델들을 살펴보고 제 3장에서는 두가지 형태의 베이지안 모델의 특성과 구축 방법에 대해 설명한다. 제 4장에서는 성능 실험에 사용할 데이터 집합과 훈련 및 검증 결과로 구성된 예측성능 실험 결과에 대해 언급하고 제 5장에는 결론에 대해 기술한다.

2. 품질 예측 모델

수많은 품질 예측 모델들의 특성을 이해하기 위해서는 모델들을 몇가지 형태로 분류하는 것이 필요하다. [7]은 결함경향성 예측 모델을 분류하기 위한 분류 프레임워크를 정의하고 모델 입력 메트릭 형태와 훈련 데이터 집합의 필요 유무라는 두 개의 기준에 의해 다음과 같은 4가지 형태의 의미 있는 모델 타입을 정의하였다.

- *VI-SL*(Vector Input-Supervised Learning) 메트릭 벡터 입력 모델이면서 감독형 모델인 경우
- *VI-UL*(Vector Input-Unsupervised Learning) 메트릭 벡터 입력 모델이면서 비감독형 모델인 경우
- *SI-AC*(Scalar Input-Automatic Classification) 스칼라 메트릭 입력 모델이면서 비감독형 모델인 경우
- *SI-HC*(Scalar Input-Human Classification) 스칼라 메트릭 입력 모델이면서 사람이 분류하는 모델인 경우

감독형 모델은 모델을 학습시킬 훈련 데이터가 존재하여 감독형 알고리즘을 사용하여 제작한 모델이며, 비감독형 모델은 훈련 데이터가 존재하지 않아 클러스터링 등의 비감독형 알고리즘을 사용하는 모델이다.

기존에 제안된 수많은 품질 예측 모델들은 대부분은 VI-SL 모델들이다. 사용된 훈련 알고리즘들은 판별분석, 선형 회귀분석, 로지스틱 회귀 분석 등과 같은 통계 기법이나 분류트리, 랜덤포리스트, 역전파 신경망, CBR(Case Based Reasoning), SVM 등과 같은 인공지능 기법들이다[2, 8]. 이들의 문제점은 대부분의 개발 집단이 훈련 데이터 집합을 보유하고 있지 않거나 제한된 형태로 보유하여 실제 개발 집단에서 사용하기 어렵다는 것이다. 설사 훈련 데이터를 보유하고 있더라도 과거에 얻은 훈련 데이터가 현재 프로젝트와 비슷한 분포를 가져야 한다는 제약이 있다.

VI-UL 모델은 VI-SL 모델의 문제점을 해결하기 위한 모델이다. 입력 속성들에 대한 결과값을 모르는 상태에서 입력 값을 분석하여 결과를 얻으려면 전문가의 지식이 필요하므로 VI-UL 모델은 완전 자동화하기 어려우며 VI-SL 모델에 비해 예측 성능도 떨어지기 때문에 매우 극소수의 연구들이 존재한다. [4]는 SOM 신경망을 사용하여 입력 벡터들을 클러스터링 한 다음, 나누어진 클러스터들을 전문가가 보고 각 클러스터들의 성질(위험도 여부)을 결정하는 모델을 제안하였다. [9]는 K-means와 Neural-Gas 클러스터링을 사용하여 이와 유사한 모델을 제안하였으며 [10]에서는 더 정확한 K-means 클러스터링을 위해 초기 클러스터 위치를 지정해주는 기법을 사용하였다. [2]는 품질 예측 모델 연구의 중요한 최근 방향 중 하나로 VI-UL 모델에 대한 연구를 언급하였다. 또 다른 새로운 연구 방향은 제한된 훈련 데이터를 사용하

는 모델 구축에 관한 것이다. 즉 정확한 모델을 만들기에는 충분치 않은 훈련 데이터가 존재하는 경우이며, 출력 결과 값을 모르는 데이터가 결과 값을 아는 데이터와 함께 존재하는 경우이다. 이런 경우에는 이 두 가지 형태의 데이터들을 학습하기 위해 세미 감독형 학습 알고리즘이 사용된다. [11]은 EM(Expectation Maximization) 기법을 사용하였으며, [12]는 Naïve Bayes 알고리즘을 세미 감독형 예측 모델 구축에 사용하였다.

베이지안 모델을 사용하여 결함경향성을 예측한 연구들은 많지 않았다. [13]의 저자들은 NASA 데이터 집합의 중요 프로젝트 데이터들을 정제해 PROMISE 레포지토리에 등록한 연구 그룹으로, 정제한 데이터들을 사용해 Naïve Bayes 모델이 고성능 의사결정트리 알고리즘인 J48 모델보다 좋은 성능을 냄을 보였다. 이 연구를 더 많은 프로젝트로 확장한 [3]에서도 입력 데이터에 logNum 필터링을 사용한 Naïve Bayes 모델이 OneR과 J48 모델보다 더 나은 성능을 보였다. 이들은 실험에 일반적으로 쓰이는 10 교차검증 방법을 발전시킨 10×10 교차검증(10×10-way cross evaluation)을 사용하였다. 이외에도 Naïve Bayes 모델을 실험에 사용한 연구는 여러개 있었지만 [6]과 같이 다른 모델들과 성능을 비교하기 위한 비교 모델로 사용된 것들이었다. Bayesian Belief Network을 결함경향성 예측에 사용한 연구는 Naïve Bayes를 사용한 연구들보다 적다. 그 이유는 Naïve Bayes 모델은 구축이 매우 용이한 반면 Bayesian Belief Network 구축은 전문가의 지식이 필요하고 자동 구축 알고리즘을 사용해도 매우 복잡하기 때문이다. [14]는 소프트웨어 산물 메트릭, 프로세스 메트릭 및 개발 단계에서 발생하는 여러 정보값들과 결함 품질들의 인과 관계를 나타내는 프레임워크를 제안하였다. 프레임워크의 한 부분으로 객체지향 메트릭들을 입력으로 받아 클래스의 결함수와 결함경향성을 예측할 수 있는 모델을 Bayesian Belief Network으로 구축하였으며 각 노드의 조건부 확률 밀도 함수를 표현하기 위해 선형, 포아송, 로지스틱 회귀 분석을 사용하였다.

3. 모델 구축

3.1 Naïve Bayes 분류기

X와 Y를 확률변수라 할 때 이들의 결합 확률 P(X = x, Y = y)는 X가 x값을 갖고, Y가 y값을 가질 확률을 의미한다. X가 x값을 갖는다는 조건하에 Y가 y값을 가질 확률을 조건부 확률이라 하며 P(Y = y|X = x)로 표현한다. X와 Y에 대한 결합 확률과 조건부 확률의 관계는 다음과 같다.

$$P(X, Y) = P(Y|X)P(X) = P(Y)P(X|Y) \quad (1)$$

식 (1)을 정리하면 다음과 같은 수식을 얻을 수 있고 이를 베이스 정리라 한다.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \quad (2)$$

품질 분류 모델에서 n개의 입력 메트릭 변수들을 n개의 확률 변수 M_i, $i = 1, \cdots, n$라 하고 이들 입력 변수들의 집합을 $\boldsymbol{M}$이라 표현한다. 또한 출력을 나타내는 변수를 확률 변수 FP라 하고 위험 개체인 경우는 1의 값을 비위험 개체인 경우는 0의 값을 갖는다고 한다. 이런 가정 하에서 P(FP|$\boldsymbol{M}$)는 $\boldsymbol{M}$값이 정해졌다는 조건 하에서의 FP의 확률이므로 FP의 사후 확률이 된다. 분류 모델의 훈련 단계에서는 훈련 데이터 집합에 존재하는 $\boldsymbol{M}$과 FP의 모든 조합에 대하여 P(FP|$\boldsymbol{M}$)를 학습하며, 검증 데이터 T는 P(1|T)와 P(0|T)를 계산하여 더 큰 확률 값을 갖는 그룹으로 분류될 수 있다.

식 (2)에서 확률 변수 X를 입력 메트릭 확률 변수 집합인 $\boldsymbol{M}$으로 바꾸면 다음과 같다.

$$P(FP|\boldsymbol{M}) = \frac{P(\boldsymbol{M}|FP)P(FP)}{P(\boldsymbol{M})} \quad (3)$$

검증 데이터 T를 분류하기 위해 사후 확률을 비교할 때 P($\boldsymbol{M}$)은 상수값이므로 무시할 수 있고, P(FP)는 훈련 데이터 집합의 각 개체값들의 비율을 계산하여 구할 수 있다. P($\boldsymbol{M}$|FP)의 계산은 베이지안 모델의 두 가지 형태인 NB와 BBN이 다른 방법으로 계산한다.

NB의 가장 중요한 특징은 FP 값이 주어졌을 때 입력 메트릭 변수들이 조건부로 독립적이라 가정하는 것이며 이는 다음과 같은 식으로 표현된다.

$$P(\boldsymbol{M}|FP=y)=\prod_{i=1}^{n}P(M_i|FP=y) \qquad (4)$$

식 (4)를 이용하면 식 (3)은 다음과 같다.

$$P(FP|\boldsymbol{M})=\frac{P(FP)\prod_{i=1}^{n}P(M_i|FP)}{P(\boldsymbol{M})} \qquad (5)$$

즉 NB 분류기는 각 검증 데이터를 식 (5)의 분자값을 더 크게 하는 그룹으로 분류한다. 만약 데이터 T에 대해 P(1|T) > P(0|T)라면 T는 FP = 1 즉 위험 그룹에 속하고, 반대이면 비위험 그룹에 속하게 된다.

메트릭 M_i가 몇 개의 한정된 값들 중 하나를 값으로 갖는 분류 속성이라면 $P(M_i|FP=y)$의 계산은 훈련 데이터로부터 고전적 확률 공식으로 쉽게 구할 수 있지만 대부분의 메트릭 값들은 연속적인 실수값들을 갖는다. 이러한 경우에 조건부 확률을 구하기 위해서는 확률 분포를 사용하여야 하며 M_i의 평균을 μ, 표준편차를 σ라 할 때 M_i의 분포를 나타내기 위해 다음과 같은 가우시안 분포를 사용한다.

$$P(M_i=m_i|FP=y_j)=\frac{1}{\sqrt{2\pi}\,\sigma_{ij}}e^{-\frac{(m_i-\mu_{ij})^2}{2\sigma_{ij}^2}} \qquad (6)$$

3.2 Bayesian Belief Network 분류기

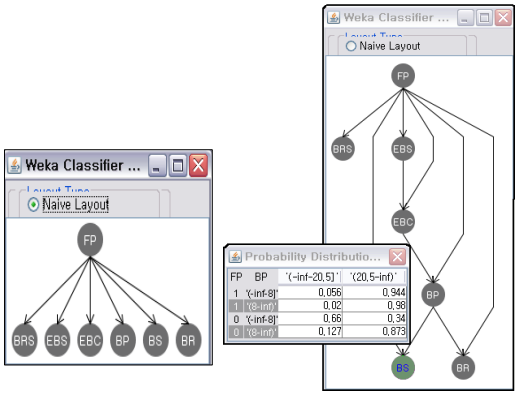
NB를 이용한 품질 분류기는 모든 입력 메트릭들이 조건부로 독립되어야한다는 엄격한 가정이 있다. 하지만 소프트웨어 개체를 정량화하는 입력 데이터의 메트릭들은 어느 정도의 의존성이 존재하는 경우가 많다. 예를 들면 품질 예측 모델에 많이 사용되는 Halstead 메트릭들을 보면 대부분 프로그램의 사이즈에 대한 메트릭들이다. LOC, 프로그램 심볼수, 연산자 수 등은 연관성이 존재한다. 몇몇 연구들에서는 이러한 엄격한 독립성 문제가 분류기의 성능에 큰 영향을 미치진 않는다고 했지만[15], 이를 논리적으로 해결한 베이지안 분류기가 BBN이다. BBN은 입력 속성들을 부분적으로만 독립적이라고 명시할 수 있게 해준다.

BBN의 두 가지 핵심 요소는 입력 변수들 사이의 종속성 관계를 표현하는 DAG(Directed Acyclic Graph)와 그래프에서 각 노드를 부모 노드들과 연관시키는 확률표이다. DAG의 각 노드는 속성을 나타내며 노드 사이의 링크는 종속관계를 표현한다. DAG 상에서 어떤 노드 A의 부모가 존재한다면 A는 A의 자손이 아닌 노드들과 조건부로 독립적이다. 본 논문에서 실험에 사용한 베이지안 모델 분류기는 WEKA[3]의 구현을 사용하였다. [그림 1]은 실험에서 사용한 데이터 집합들 중 하나인 훈련 1을 사용하여 BBN을 구축한 결과이다. 입력 메트릭은(BRS, EBS, EBC, BP, BS, BR)이며 출력은 FP이다. (a)는 최대 부모 노드를 1개로 한 경우이며, (b)는 2개 (c)는 3개로 한 경우이다. (a)는 입력인 6개의 메트릭들이 모두 조건부 독립이며 이는 NB 모델과 유사한 경우이다.

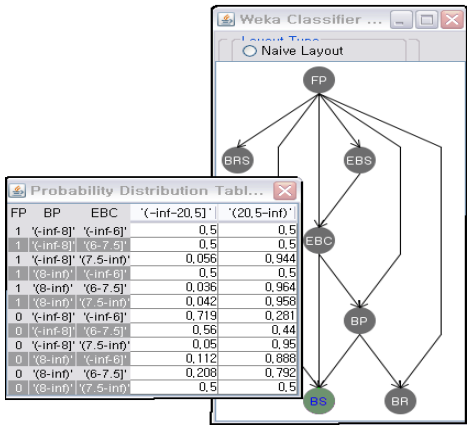
각 노드는 확률표를 가지고 있다. 노드 X가 부모를 가지고 있지 않으면 X의 확률표는 X의 사전 확률 P(X)만을 포함하며, k개의 부모 {Y_1, Y_2, …, Y_k}를 가지고 있으면 P(X|Y_1, Y_2, …, Y_k)를 포함한다. [그림 1]에서 세 가지 경우 모두 FP 노드는 부

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

모가 없으므로 P(FP) 정보만 확률표에서 가지고 있으면 되지만 나머지 노드들은 조건부 확률 정보들을 가지고 있어야한다. (a)의 BP 노드의 확률표는 P(BP|FP)의 정보만을 가지고 있지만 (b)의 BS 노드는 FP와 BP를 부모로 가지고 있으므로 P(BS|FP, BP)의 값들을 확률표로 가지고 있어야한다. (b)의 그림에 나타낸 확률표가 그것이며 표의 첫째칸은 P(BS∈(-inf-20.5] | FP = 1, BP∈(-inf-8]) = 0.056이라는 의미이다. (c)의 BS 노드는 FP, BP, EBC를 부모로 가지고 있으므로 P(BS|FP, BP, EBC) 정보를 나타내야하며 확률표의 첫째 칸은 P(BS∈(-inf-20.5] | FP = 1, BP∈(-inf-8], EBC∈(-inf-6]) = 0.5라는 의미이다.



(a) 최대부모수 1경우 (b) 최대부모수 2경우



(c) 최대부모수 3경우

[그림 1] 훈련 1의 BBN 구축 결과

BBN 모델 구축 작업은 DAG 즉 네트워크 위상과 모든 노드의 확률표를 구축하는 것이다. 위상은 도메인 전문가의 노드들에 대한 원인-결과 지식을 통해 구축될 수도 있고 훈련 데이터 집합을 이용해 자동화된 알고리즘으로 구축될 수도 있다. 본 논문은 후자의 방법을 사용하였으며 K2 알고리즘을[16] 사용하였다. 노드들의 확률표들은 NB 모델에서 기술한 것과 유사한 방법으로 훈련 데이터 집합으로부터 쉽게 구축할 수 있다. 검증 데이터의 분류는 DAG와 확률표로 계산한 확률들을 이용하여 FP의 사후확률을 구해 NB 모델과 유사한 방법으로 이루어진다.

4. 실험

4.1 데이터집합

실험에 사용한 데이터 집합은 [4, 5]에서 사용한 데이터 집합을 사용하였다. 모델의 입력 대상은 객체지향 실시간 시스템 명세 언어인 SDL로 작성한 설계 명세이며 설계 블록을 정량화하는 메트릭 벡터는(BRS, EBS, EBC, BP, BS, BR)이다. 이는 모델의 입력 벡터가 되며 각 메트릭 벡터 값에 대해 출력 클래스를 판단하는 값인 FP가 존재한다. FP는 위험 개체는 1, 비위험 개체는 0값을 갖는다.

실험에는 두 개의 훈련 데이터 집합과 두 개의 검증 데이터 집합을 사용하였다. 독립적인 검증 데이터가 있기 때문에 실험 시 교차검증을 사용하지 않았다. 훈련 1, 훈련 2는 각각 300개의 개체 데이터를 가지고 있는 훈련 데이터 집합이며 훈련 1은 오류 개체가 32개이고 훈련 2는 34개이다. 검증 1, 검증 2는 각각 200개의 개체 데이터를 가지고 있는 검증 데이터 집합이며 둘 다 23개의 오류 개체를 가지고 있다. 훈련 2, 검증 2는 훈련 1, 검증 1에 비해 위험 그룹과 비위험 그룹간의 차이를 크게 해 위험도 판단이 애매한 개체들을 없앤 데이터 집합을 나타낸다. 모든 집합은 0과 1사이로 정규화된 데이터와 비정규화 데이터 즉 원본 데이터

의 두 경우로 실험하였다.

4.2 속성 선정

본 실험의 입력 벡터의 크기는 6으로 작지만 다양한 결과를 보기 위해 입력 데이터 집합에 대해 속성 선정을 통한 차원 축소 작업을 진행하였다. 차원 축소 기법은 WEKA에서 제공하는 기법들 전부를 고려하였으며 모든 속성을 선택한 기법들과 두 훈련 집합 모두 2개 이하의 속성을 선택한 기법들을 제외하고 같은 결과를 낸 기법들은 한 개만 대표 기법으로 선정하였다. 그 결과 차원 축소 기법으로 많이 사용되는 PCA(Principal Component Analysis), 각 속성들의 예측 능력과 그들 사이의 중복 정도를 평가하여 가장 출력값과 연관성이 높은 집합을 선정하는 방법인 CfsSubsetEval, 속성집합 값들의 출력값에 대한 일관성 정도로 속성 집합을 평가하는 방법인 ConsistencySubsetEval이 선정 되었다.

<표 1>은 3가지 기법으로 수행한 속성 선정 결과이다. PCA는 훈련 1과 훈련 2의 속성선정 결과가 같지만 나머지 두 기법은 결과가 틀리다. 하지만 정규화 여부는 속성 선정 결과에 영향을 미치지 않았다. 예를 들면 CfsSubsetEval을 훈련 1에 적용한 결과는 정규화 데이터와 비정규화 데이터 모두 2, 3, 5, 6번째 속성을 갖는(EBS, EBC, BS, BR) 이다.

〈표 1〉 차원축소 결과

훈련 데이터 / 속성선정기법	훈련 1	훈련 2
CfsSubsetEval	2, 3, 5, 6 : 4	1, 3, 5 : 3
ConsistencySubsetEval	2, 4, 5 : 3	1, 2 : 2
PCA	1, 2, 3, 4 : 4	1, 2, 3, 4 : 4

4.3 훈련 결과

모델의 예측 성능을 평가하는 척도는 Precision, F-measure, Recall, ROC Area 등 여러 가지가 있으나 예측 오류를 세분화하는 평가 척도가 다양한 측면의 성능 평가를 가능케 해준다. 예측 모델의 예측 오류는 두 가지 형태가 존재한다. Type I 오류란 비위험 개체를 위험 개체로 판단하는 오류이고, Type II 오류는 위험 개체를 비위험 개체로 판단하는 오류이다. 전자보다는 후자가 소프트웨어 품질에 심각한 영향을 끼치게 되므로 후자가 더욱 중요한 오류이다. Type I 오류 결과는 '비위험 개체를 위험 개체로 선정한 수/비위험 개체수' 형태로 나타내고 Type II 오류 결과는 '위험 개체를 비위험 개체로 선정한 수/위험 개체수'로 나타낸다.

<표 2>는 훈련 1과 훈련 2에 대한 훈련 결과를 나타낸다. 즉, 교차검증을 사용하지 않고 전체 훈련 집합에 대해 훈련이 완료된 상태에서 각 훈련 집합에 대한 학습 오류 정보를 나타낸다. BBN(k)는 네트워크 노드의 최대 부모수를 k로 했을 때 구축된 베이지안 네트워크를 의미한다.

〈표 2〉 훈련 결과

모델 / 훈련 데이터		훈련 1		훈련 2	
		Type I	Type II	Type I	Type II
NB (비정규화)	차원축소안함	20/268	2/32	7/266	0/34
	CfsSubsetEval	19/268	1/32	3/266	2/34
	Consistency SubsetEval	15/268	4/32	4/266	1/34
	PCA	16/268	6/26	5/266	0/34
NB (정규화)	차원축소안함	16/268	2/32	7/266	0/34
	CfsSubsetEval	18/268	2/32	3/266	2/34
	Consistency SubsetEval	14/268	4/32	4/266	1/34
	PCA	14/268	5/32	5/266	0/34
BBN(1)	차원축소안함	18/268	0/32	5/266	0/34
	CfsSubsetEval	18/268	0/32	1/266	3/34
	Consistency SubsetEval	10/268	4/32	2/266	0/34
	PCA	20/268	0/32	5/266	0/34
BBN(2)	차원축소안함	18/268	0/32	3/266	2/34
BBN(3)	차원축소안함	10/268	4/32	3/266	2/34

세 개의 BBN 모델은 정규화와 비정규화 데이터

에 대한 결과가 일치하였고, NB는 훈련 2의 경우만 일치하였다. BBN(2)와 BBN(3)는 차원축소를 하지 않은 경우만을 나타내었으며 BBN(3)의 Type I 오류의 경우를 제외하고는 두 모델은 BBN(1)보다 더 나은 결과를 보이진 않았다. 전체적으로 보면 차원 축소는 학습 오류를 줄이는데 큰 영향을 미치지 않았으며, 당연히 훈련 2의 학습 오류가 적었다. NB와 BBN의 학습 결과도 유의미한 차이를 보이진 않았지만, BBN이 PCA를 사용하였을 때 훈련 1에 대한 Type I 오류와 CfsSubsetEval을 사용하였을 때 Type I, II 오류를 제외하고는 모든 경우에서 BBN이 NB보다 근소하게 나은 결과를 보였다.

4.4 검증 결과

두 모델의 예측 정확도를 평가하기 위해 두 가지 훈련 데이터 집합으로 훈련시킨 모델들에 두 가지 검증 데이터 집합들을 적용하였으며 <표 3>은 그 결과이다.

일단 BBN 모델들의 검증 결과를 보면 BBN(3)의 Type I 오류를 제외하고는 모든 경우에서 BBN(2)와 BBN(3)의 오류가 BBN(1)보다 같거나 많은 결과를 보였다. 따라서 향후 기술에서 BBN은 BBN(1)을 의미한다. BBN은 훈련 결과와 마찬가지로 검증 결과도 데이터의 정규화 여부가 결과에 영향을 미치지 않았다. NB의 경우는 훈련 2의 경우는 정규화와 비정규화 데이터를 적용한 결과가 같았지만 검증 데이터의 실험 결과는 조금 달랐다. NB의 경우 큰 차이는 없지만 정규화 데이터를 사용한 결과가 비정규화 데이터를 사용한 결과보다 조금 더 나았다.

두 가지 오류를 (Type I, Type II) 형태로 나타낸다. 훈련 1로 훈련시킨 모델에 검증 1을 적용한 경우, 즉 훈련 1-검증 1의 결과를 보면 NB는 (13, 3)에 가까운 결과가 많고 BBN은 (10, 5)에 가까운 결과와 두 개의 다른 결과가 나타났다. BBN이 Type I 오류는 적으나 중요한 Type II 오류가 많으므로 이 경우는 두 모델 중 어떤 것이 낫다고 할 수 없다. 특이한 점은 두 베이지안 모델 모두 이 경우에 매우 많은 오류를 냈다는 점이다.

〈표 3〉 검증 결과

모델 \ 검증데이터			검증 1		검증 2	
			Type I	Type II	Type I	Type II
NB (비정규화)	차원축소 안함	훈련 1	23/177	3/23	18/177	0/23
		훈련 2	2/177	10/23	0/177	0/23
	CfsSubset Eval	훈련 1	17/177	4/23	12/177	1/23
		훈련 2	5/177	13/23	0/177	1/23
	Consistency SubsetEval	훈련 1	12/177	5/23	10/177	1/23
		훈련 2	6/177	11/23	0/177	1/23
	PCA	훈련 1	16/177	3/23	14/177	1/23
		훈련 2	6/177	9/23	0/177	0/23
NB (정규화)	차원축소 안함	훈련 1	13/177	3/23	12/177	0/23
		훈련 2	4/177	10/23	1/177	0/23
	CfsSubset Eval	훈련 1	13/177	3/23	10/177	1/23
		훈련 2	5/177	13/23	0/177	1/23
	Consistency SubsetEval	훈련 1	12/177	4/23	10/177	2/23
		훈련 2	6/177	11/23	0/177	1/23
	PCA	훈련 1	13/177	2/23	11/177	1/23
		훈련 2	3/177	9/23	2/177	0/23
BBN (1)	차원축소 안함	훈련 1	10/177	5/23	3/177	1/23
		훈련 2	3/177	5/23	0/177	0/23
	CfsSubset Eval	훈련 1	10/177	5/23	2/177	1/23
		훈련 2	1/177	16/23	0/177	3/23
	Consistency SubsetEval	훈련 1	6/177	6/23	3/177	6/23
		훈련 2	11/177	11/23	0/177	0/23
	PCA	훈련 1	13/177	3/23	6/177	0/23
		훈련 2	12/177	5/23	0/177	0/23
BBN (2)	차원축소 안함	훈련 1	10/177	5/23	3/177	1/23
		훈련 2	3/177	9/23	0/177	0/23
BBN (3)	차원축소 안함	훈련 1	6/177	6/23	3/177	6/23
		훈련 2	3/177	9/23	0/177	0/23

훈련 1-검증 2의 결과로 관찰할 것은 매우 일반적인 데이터로 훈련시킨 모델이 실제 데이터와 가까운 분리된 분포를 갖는 검증 데이터에 좋은 성능을 보이는 가에 대한 것으로 예측 모델의 중요한 성질인 일반화와 관련이 있다. 이 경우는 BBN이 NB에

대해 좋은 결과를 보였으며 NB는 Type I 오류가 매우 높게 나타났다. 가장 좋은 결과는 BBN이 차원 축소를 안한 경우와 CfsSubsetEval의 경우 (3, 1)과 (2, 1)이 되었다.

훈련 2-검증 1의 경우는 큰 의미가 없다. 왜냐하면 위험 그룹과 비위험 그룹의 차이가 명확한 데이터로 훈련된 모델이 차이가 매우 애매한 입력 집합에 대해 정확한 판단을 내리기 어렵기 때문이다. 주목할만 것은 두 베이지안 모델들은 이 경우 Type II 오류는 훈련 1-검증 1의 경우보다 많았지만 Type I 오류는 BBN이 ConsistencySubsetEval을 사용한 경우만 제외하고는 적었다는 것이다. BBN이 차원축소를 안한 경우가 (3, 5)로 가장 좋은 결과를 보였다.

훈련 2-검증 2의 경우는 대부분 모델들에서 거의 오류가 없다. 왜냐하면 훈련 데이터나 검증 데이터 모두 애매성을 없앤 집합이기 때문이다. BBN의 CfsSubsetEval의 경우는 3개의 Type II 오류를 내어서 특이하게 좋지 않은 결과를 보였다.

결과적으로 베이지안 모델은 타 모델에 비해 훈련 데이터와 검증 데이터가 위험도 구분이 불분명한 경우에는 좋은 성능을 보이지 않았다. 하지만 구분이 명확한 훈련 데이터를 사용한 경우에는 좋은 성능을 보였다. 두 베이지안 모델 중에서는 BBN이 NB보다 나은 성능을 보였으며, 차원 축소를 행하지 않은 BBN이 가장 좋은 성능을 보였다.

4.5 타 모델과 성능 비교

예측 모델 초기 연구들부터 오류 역전파 신경망은 좋은 성능을 보였으며, SVM은 최근 수년간 좋은 성능을 보인 연구들이 발표되었다. 따라서 훈련 데이터를 사용하는 품질 예측 모델들 중 좋은 예측 성능을 보인 이들 모델들과 베이지안 모델의 결과를 비교하였다.

베이지안 모델의 경우 훈련 결과에서 정규화 데이터를 사용한 경우가 비정규화 데이터를 사용한 경우보다 학습 오류가 적었고, 품질 예측 프로세스의 경우 정규화 데이터를 사용하는 것이 일반적이므로 타 모델과의 비교는 정규화 데이터를 사용한 경우로 국한하였다. 비교 모델들은 베이지안 모델들 중 가장 좋은 결과를 보인 차원 축소를 안한 BBN, Radial Basis와 Polynomial을 커널 함수로 사용하는 SVM, 역전파 신경망을 사용한 BPM이다[5]. BBN은 훈련 1-검증 1의 경우는 다른 모델보다 좋지 않은 성능을 보였고, 훈련 1-검증 2의 경우는 비슷한 성능을 보였다. 하지만 훈련 2로 학습시킨 경우에는 가장 좋은 성능을 보였다. 이는 위험도 구분이 명확한 훈련 데이터에서 매우 좋은 성능을 보인 검증 실험 결과의 관찰과도 일치한다.

〈표 4〉 타 모델과 성능 비교

모델 \ 훈련 데이터			검증 1		검증 2	
			Type I	Type II	Type I	Type II
BBN	차원축소 안함	훈련 1	10/177	5/23	3/177	1/23
		훈련 2	3/177	5/23	0/177	0/23
SVM	Radial Basis	훈련 1	0/177	15/23	0/177	2/23
		훈련 2	1/177	11/23	0/177	0/23
	Polynomial	훈련 1	0/177	13/23	0/177	2/23
		훈련 2	1/177	12/23	0/177	0/23
BPM		훈련 1	2/177	4/23	4/177	0/23
		훈련 2	6/177	11/23	0/177	0/23

5. 결 론

본 논문에서는 베이지안 모델의 대표적인 두 형태인 Naïve Bayes 모델과 Bayesian Belief Network 모델에 SDL로 작성한 설계 명세 데이터를 적용해 보았으며, 그 결과 차원 축소를 하지 않은 Bayesian Belief Network 모델의 성능이 약간 더 좋았다. 기존의 좋은 성능을 보였던 오류 역전파 신경망 모델과 SVM 모델과 제안 모델을 비교한 결과는 판단이 애매한 부분을 삭제하여 위험 그룹과 비위험 그룹 간에 차이를 크게 한 데이터 집합으로 훈련한 경우는 제안 모델이 더 나은 성능을

보였지만 차이가 애매한 데이터 집합의 경우는 더 나은 성능을 보이지 않았다. 향후 연구 방향은 원인-결과 지식을 이용하여 도메인 지식이 많이 첨가된 Bayesian Belief Network 모델을 구축하여 평가하는 것이다.

참 고 문 헌

[1] Ebert, C., "Fuzzy classification for software criticality analysis : *Expert Systems with Applications*, Vol.11, No.3(1996), pp.323-342.

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

[3] Menzies, T., J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors", *IEEE Trans Software Engineering*, Vol.33, No.1(2007), pp.2-13.

[4] 홍의석, "훈련 데이터집합을 사용하지 않는 소프트웨어 품질예측 모델," 「정보처리학회논문지」, 제10-D권, 제4호(2003), pp.689-696.

[5] 홍의석, "Support Vector Machine을 이용한 초기 소프트웨어 품질 예측," 「한국IT서비스학회지」, 제10권, 제2호(2011), pp.235-245.

[6] Elish, K. O. and M. O. Elish, "Predicting defect prone software modules using support vector machines", *J. Systems Software*, Vol. 81, No.5(2008), pp.649-660.

[7] 홍의석, "소프트웨어 품질 예측 모델을 위한 분류 프레임워크," 「한국콘텐츠학회논문지」, 제10권, 제6호(2010), pp.134-143.

[8] Catal, C. and B. Diri, "A systematic review of software fault prediction studies", *Expert Systems with Applications*, Vol.36, No.4(2009), pp.7346-7354.

[9] Zhong, S., T. M. Khoshgoftaar, and N. Seliya, "Analyzing Software Measurement Data with Clustering Techniques", *IEEE Intelligent Systems*, Vol.19, No.2(2004), pp.20-27.

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

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

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

[13] Menzies, T., J. DiStefano, A. Orrego, and R. Chapman, "Assessing predictors of software defects", *Proc. workshop on Predictive software models*, 2004.

[14] Pai, G. J. and J. B. Dugan, "Empirical analysis of software fault content and fault proneness using Bayesian methods", *IEEE Trans. Software Engineering*, Vol.33, No.10 (2007), pp.675-686.

[15] Turhan, B. and A. Bener, "Analysis of Naive Bayes' assumptions on software fault data : An empirical study", *Data and Knowledge Engineering*, Vol.68, No.2(2009), pp. 278-290.

[16] Cooper, G. F. and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data", *Machine Learning*, Vol.9, No.4(1992), pp.309-347.

◈ 저 자 소 개 ◈

홍 의 석 (hes@sungshin.ac.kr)

서울대학교 계산통계학과 전산과학전공에서 1992년과 1994년에 각각 학사, 석사 학위를 취득하였으며 1999년에 서울대학교 전산과학과에서 박사학위를 취득하였다. 1999년부터 2002년까지 안양대학교 디지털미디어학부 교수로 근무하였으며, 2002년부터 현재까지 성신여자대학교 자연과학대학 IT학부 교수로 재직 중이다. 연구 관심분야는 소프트웨어공학 분야 중 소프트웨어 품질, 웹 기반 응용 기술 등이다.