pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2013.2.4.219

Improvement of Classification Accuracy on Success and Failure Factors in Software Reuse using Feature Selection

Young-Ok Kim† · Ki-Tae Kwon††

ABSTRACT

Feature selection is the one of important issues in the field of machine learning and pattern recognition. It is the technique to find a subset from the source data and can give the best classification performance. Ie, it is the technique to extract the subset closely related to the purpose of the classification. In this paper, we experimented to select the best feature subset for improving classification accuracy when classify success and failure factors in software reuse. And we compared with existing studies. As a result, we found that a feature subset was selected in this study showed the better classification accuracy.

Keywords : Feature Selection, Key Attribute, Classification, Clustering, Data Mining, Software Reuse

특징 선택을 이용한 소프트웨어 재사용의 성공 및 실패 요인 분류 정확도 향상

김 영 옥† · 권 기 태††

요 약

특징 선택은 기계 학습 및 패턴 인식 분야에서 중요한 이슈 중 하나로, 분류 정확도를 향상시키기 위해 원본 데이터가 주어졌을 때 가장 좋은 성능을 보여줄 수 있는 데이터의 부분집합을 찾아내는 방법이다. 즉, 분류기의 분류 목적에 가장 밀접하게 연관되어 있는 특징들만을 추출하여 새로운 데이터를 생성하는 것이다. 본 논문에서는 소프트웨어 재사용의 성공 요인과 실패 요인에 대한 분류 정확도를 향상시키기 위해 특징 부분 집합을 찾는 실험을 하였다. 그리고 기존 연구들과 비교 분석한 결과 본 논문에서 찾은 특징 부분 집합으로 분류했을 때 가장 좋은 분류 정확도를 보임을 확인하였다.

키워드 : 특징 선택, 주요 속성, 분류, 클러스터링, 데이터 마이닝, 소프트웨어 재사용

1. 서 론

Morisio et al.[1]은 1994년부터 1997년까지 19개 회사로부터 소프트웨어를 재사용하여 수행된 총 24개 프로젝트에 대해 관리자들을 대상으로 코드와 분석, 설계 문서 모두를 포함하여 묻는 인터뷰를 실시하였다. 주요 재사용 규칙이 소개되었는지, 비 재사용 프로세스를 수정하였는지, 소프트웨어 개발에 직접 참여한 직원 수는 몇 명인지, 이들의 소프트웨어 프로세스 성숙도 레벨은 어느 정도인지, 어떤 응용에 해당되는 프로젝트인지, 프로젝트 크기는 얼마인지 등으로 구성된 26개 질문과 각 프로젝트에서 소프트웨어 재사용이 성공했는지 실패했는지를 묻는 1개의 질문, 그리고 각 프로젝트를 구분하는 'Project ID', 이렇게 총 28개 항목(질문)으로 구성되었다. 인터뷰의 목적은 소프트웨어 재사용에 있어 성공과 실패에 영향을 준 핵심 요소에 대한 경험적 증거를 유도해 내는 것이다. 조사 대상 프로젝트들은 다양한 비즈니스 도메인을 사용하고 있고, 객체지향 및 절차적 개발방법 모두를 사용하고 있었으므로 각 질문에 대한 답변도 다양하였다[1, pp.342-344]. 이 중 어떤 속성 조합이 성공과 실패에 영향을 주었는지 분석하기 위해 CART 알고리즘을 이용하여 분류 실험을 한 결과 5개의 핵심 속성을 찾았다. 또, 다른 연구자들이 그들의 결론을 반복 검증할 수 있도록 PROMISE Software Engineering Repository[1)]에 데이터 셋을 공개하였고, 이것은 Menzies et al.[2]의 연구에 다시 이용되었는데, J4.8 알고리즘을 이용하여 분류한 결과 8개의

† 준 회 원 : 강릉원주대학교 컴퓨터공학과 박사
†† 종신회원 : 강릉원주대학교 컴퓨터공학과 교수
논문접수 : 2012년 10월 4일
수 정 일 : 1차 2012년 11월 28일, 2차 2012년 11월 30일
심사완료 : 2012년 12월 6일
* Corresponding Author : Ki-Tae Kwon(ktkwon@gwnu.ac.kr)

1) http://promise.site.uottawa.ca/SERepository/datasets/reuse.arff

핵심 속성을 찾았다. [1]과 [2]의 연구에서 사용된 CART, J4.8은 동일한 알고리즘을 조금 다르게 변화시킨 결정트리(Decision Tree) 알고리즘이다. 본 논문은 [1]과 [2]의 연구 결과를 검증해 보고 분류의 성능을 더 높이고자 Cfs SubsetEval(CFS attribute Subset Evaluator) 특징 선택 알고리즘과 분류기를 이용하여 실험하였다. 그 결과 찾아진 속성들은 기존 연구 결과와 대부분 일치했지만 [2]와 같이 새로운 속성을 찾기도 하였다. 찾아진 특징 부분집합 원소의 개수는 4개로, 비교 논문 중 가장 작은 수로 구성된 조합을 찾았으며 이것으로 분류했을 때 더 나은 분류 정확도를 보임을 확인하였다. 이 연구들의 공통적인 목적은 찾아진 속성 조합이 소프트웨어 재사용 결정이나 재사용 소프트웨어 채택 시 우선적인 판단 기준이 될 수 있음을 보이는 것이다.

본 논문은 2장에서 배경지식을, 3장에서 데이터마이닝 기법을 이용한 실험과 결과를, 4장에서 결과 비교를, 그리고 5장에서 결론 및 향후연구로 끝맺는다.

2. 배경지식

2.1 분류 알고리즘(Classification algorithm)

■ SVM : SVM은 1995년 러시아 통계학자인 Vladimir Vapnik에 의해 제안된 커널 기반의 지도 학습 알고리즘으로[4], 분류문제에 있어 일반화 성능이 높기 때문에 많은 분야에서 응용되고 있으며, 다른 학습 알고리즘에 비해 조정해야 할 파라미터의 수가 많지 않아 비교적 간단하게 학습에 영향을 미치는 요소들을 규명할 수 있다. 선형 SVM은 두 집합 사이의 분리간격을 최대로 하는 초평면을 찾는 분류기로서 최대 마진 분류기라 불린다. 그러나 실제 입력 데이터를 적용할 경우 분리 불가능한 데이터가 존재하게 되는데, 이러한 오분류 데이터를 제거하기 위한 방법으로 슬랙 변수 ξ와 패널티 값 C를 사용한다. 또 비선형 경계를 갖는 데이터를 SVM으로 분류하기 위해서는 본래의 좌표 공간에 있는 데이터 x를 선형 분류를 가능하게 하는 새로운 차원의 좌표 공간 $\Phi(x)$로 맵핑하여 초평면을 구하여 분류하게 된다[4].

■ RBFNetwork : 통계학의 다변량 분석 및 보간 문제 해결에 이용되었던 RBF(Radial Basis Function)를 Brommhead와 Low가 신경망 모델을 구성하는데 이용함으로써 RBFNetwork가 제안되었다[5]. RBF 신경망은 입력층, 중간층, 출력층의 3계층으로 구성된 전방향 신경망(Feedforward Neural Network)이다. 입력층은 입력 벡터 공간에 해당하고 출력층은 패턴의 부류(Class)에 해당한다. 따라서 중간층을 결정하면 전체 신경망의 구조도 결정된다. 입력층과 중간층 사이의 가중치는 중간층이 결정될 때 고정되고, 중간층과 출력층 사이의 가중치는 학습을 통해 구한다[5]. RBFNetwork는 빠른 학습 시간, 일반화, 단순화의 특징으로 학습 데이터를 분류하는 작업과 비선형 시스템 모델링 등에 적용되고 있다[6].

■ NaiveBayes : NaiveBayes 분류기는 확률에 기반한 베이지언 알고리즘 중의 하나이다. 속성 집합과 클래스 변수 사이의 확률적 관계를 모델링하는 접근 방법으로 클래스의 사전 지식과 데이터로부터 획득한 새로운 증거를 결합시키는 통계 원리인 베이스 정리를 기반으로 한다[7].

X와 Y를 한 쌍의 확률 변수라 하면, 이들의 결합 확률 P(X=x, Y=y)는 X가 x의 값을 갖고 Y가 y의 값을 갖는 확률을 말한다. 그리고 조건부 확률은 다른 확률 변수의 값이 이미 알려진 경우, 한 확률 변수가 특정 값을 가질 확률을 말한다. 예를 들어 조건부 확률 P(Y=y|X=x)는 X의 값이 x로 주어졌을 때, Y가 y의 값을 취할 확률을 뜻한다. X와 Y에 대한 결합 확률과 조건부 확률은 식(1)로 표현된다.

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

식 (1)을 재배열하면 식 (2)의 베이스 정리가 된다.

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

조건부 확률은 Y의 사전확률인 P(Y)와 상반되게 Y의 사후확률로 불린다. 베이스 정리는 사후확률 P(Y), 클래스 조건부 확률 P(X|Y), 증거 P(X)의 식으로 표현된다.

NaiveBayes 분류기는 조건부 독립성을 가정할 때, X의 모든 조합에 대하여 클래스 조건부 확률을 계산하는 대신 Y가 주어졌을 때 각 Xi의 조건부 확률을 계산하면 된다. 시험 항목을 분류하기 위해 NaiveBayes 분류기는 각 클래스 Y에 대한 사후확률을 계산한다[7].

■ BayesNet : 클래스 조건부 확률 P(X|Y)를 계산하기 위하여 베이지안 분류 방법의 또 다른 구현 방법인 BayesNet가 있다. NaiveBayes 분류기가 사용하는 조건부 독립 가정은 속성들 사이에 약간의 관련성이 존재할 때는 매우 엄격하다. 그에 비해 BayesNet는 클래스 조건부 확률 P(X|Y)를 모델링하는 좀 더 융통성 있는 접근 방법이다. 클래스가 주어졌을 때 모든 속성들의 조건부 독립성을 요구하는 대신, 특정 쌍의 속성들이 조건부적으로 독립적임을 명시할 수 있게 해준다[7].

BayesNet는 그래픽 모델을 사용하여 특정 영역의 사전 지식을 이용하는 방법을 제공한다. 여기에 사용된 네트워크는 변수 사이의 우연 종속성을 표시하는 데 사용될 수 있다. 그리고 네트워크의 생성은 상당한 노력과 시간이 요구되지만 일단 네트워크 구조가 결정되면 쉽게 새 변수를 추가할 수 있다. 또, 속성 정보가 빠져 있는 경우 그 속성의 모든 가능한 값들에 대한 확률을 합산하여 계산할 수 있으므로 불완전한 데이터를 다루는데 적합하다.

2.2 특징 선택(Feature Selection)

특징 선택이란 전체 속성 집합에서 분류에 결정적인 영향을 미치는 부분 집합의 조합을 찾는 기법이다. 부분 집합을 선택함으로서 관련이 없는 속성을 제거해주고 잡음도 줄여

주기 때문에 데이터마이닝 알고리즘들을 더 잘 작동하게 한다. 이러한 속성 선택의 표준 전략은 삽입, 필터, 래퍼 기법이 있다. 특히 필터 기법 중, CfsSubsetEval가 여러 연구들[8-11]에서 그 성능이 증명되었다. [10]의 연구는 기계 학습 문제를 위한 벤치마킹 데이터 셋인 WEBKB Data Set에 다양한 속성 선택 기법들을 이용하여 분류하는 실험을 하였는데, 여러 기법 중 CfsSubsetEval가 가장 우수한 결과를 보임을 증명하였다. 또, [11]의 연구는 CfsSubsetEval 기법을 이용하여 독립적인 변수이면서 쌍 간의 상관관계가 가능한 한 낮은 속성의 집합을 선택하여 분류 실험을 하였고 그 성능을 검증하였다. 즉, CfsSubsetEval 기법은 각 속성들의 예측 능력과 그들 사이의 중복 정도를 평가하여 가장 출력값과 연관성이 높은 집합을 선정하는 것이다[12].

본 논문도 CfsSubsetEval를 이용하여 속성 평가를 했고 검색 방법으로는 모든 기법을 실험한 후 같은 결과를 낸 기법들 중 한 개만 대표로 선정하였는데 GreedyStepwise, BestFirst, LinearForwardSelection이 선택되었다.

3. 실 험

3.1 실험방법

실험을 위해 자바 기반의 오픈소스 툴킷인 WEKA[2]를 사용하였다. WEKA의 분류탭에서 제공하는 기법 중 reuse.arff 데이터 셋에 대해 분류 결과를 낸 모든 기법을 사용하였다. 각 분류기들의 테스트 옵션으로는 과적합을 피하고 가장 좋은 예측 모델을 선택하는 방법인 k-분할 교차검증을 사용하였는데, 이 방법은 데이터 셋을 k개의 세트로 나눈 후 k-1개의 데이터 셋으로 학습시키고 나머지 하나로 모델을 검증하는 방법이다[3]. 이 과정을 k개의 세트 각각을 대상으로 순환한다. 일반적으로 예측 모델의 에러는 k번 실행한 테스트 데이터들에서 발생한 에러의 평균값이 된다. 알고리즘이나 세팅을 달리함에 따라 다양한 예측 모델이 나오게 되는데, k번의 테스트 후 평균 에러가 가장 작은 모델을 선택한다.

3.2 실험결과

총 28개 속성으로 구성된 데이터 셋에서 'Project ID' 속성은 24개 프로젝트를 구별만하는 속성이므로 실험에서 제외시켰다. 그리고 가장 마지막 'Success or Failure' 속성은 분류의 결과를 판단하는 평가 기준으로 사용된다. 이것을 제외하고 나머지 26개 속성을 이용하여 분류한 것과 CfsSubsetEval이 찾아준 특징 부분 집합으로 분류한 것의 비교가 Table 1에 있다.

분류기 중에는 SVM, RBFNetwork, BayesNet, Naive Bayes가 분류 성능이 좋았고, 속성 선택기인 CfsSubsetEval의 검색 방법 중에는 GreedyStepwise가 가장 성능이 좋았다. 이것이 실험 I에 해당된다.

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

Table 1. Experiment I
Accuracy(%) : Correctly Classified Instances

Classifier	Test option (CV)	All attributes	CfsSubsetEval		
			Greedy Stepwise 8,9,11,13,14,16,17,19,24	BestFirst 3,8,9,11,13,14,16,17,18,19,24	LinearForward Selection 3,8,9,11,13,14,16,17,19,24
SVM	**3**	**83.33%**	**100%**	**100%**	**100%**
	5	**79.17%**	**100%**	**100%**	**100%**
	10	**87.5%**	**100%**	**100%**	**100%**
RBFNetwork	**3**	**91.67%**	**100%**	**95.83%**	**95.83%**
	5	**95.83%**	**100%**	**100%**	**95.83%**
	10	**91.67%**	**100%**	**100%**	**100%**
BayesNet	**3**	**95.83%**	**100%**	**100%**	**100%**
	5	**100%**	**100%**	**100%**	**100%**
	10	**100%**	**100%**	**100%**	**100%**
NaiveBayes	**3**	**95.83%**	**100%**	**100%**	**100%**
	5	**95.83%**	**100%**	**100%**	**100%**
	10	**95.83%**	**100%**	**100%**	**100%**
Bagging	3	83.33%	91.67%	91.67%	91.67%
	5	87.5%	91.67%	91.67%	91.67%
	10	95.83%	95.83%	95.83%	95.83%
Classification ViaClustering	3	75%	91.67%	83.33%	91.67%
	5	83.33%	91.67%	91.67%	91.67%
	10	87.5%	100%	100%	100%
MultiClass Classifier	3	91.67%	100%	91.67%	95.83%
	5	91.67%	100%	100%	100%
	10	91.67%	95.83%	95.83%	95.83%
RotationForest	3	87.5%	91.67%	95.83%	87.5%
	5	95.83%	91.67%	91.67%	95.83%
	10	83.33%	95.83%	91.67%	91.67%
AD Tree	3	91.67%	91.67%	91.67%	91.67%
	5	91.67%	91.67%	91.67%	91.67%
	10	91.67%	91.67%	91.67%	91.67%
RandomForest	3	87.5%	95.83%	87.5%	91.67%
	5	91.67%	95.83%	87.5%	95.83%
	10	95.83%	91.67%	91.67%	100%
RandomTree	3	83.33%	95.83%	83.33%	91.67%
	5	75%	87.5%	91.67%	95.83%
	10	87.5%	91.67%	95.83%	91.67%
평균		89.77	**96.09**	94.82	95.83

이렇게 찾아진 부분 집합 9개를 전체 집합으로 하여 실험 II와 II'을 통해 부분 집합의 원소 개수를 더 줄여 보았다. Table 2는 실험 I의 결과를 검증하기 위해 9개 전체 집합에서 해당 속성을 제거하면서 분류 성능에 영향을 주는가를 확인하는 것이다. 속성을 제거해도 분류에 영향을 주지 않는 8, 11, 13, 17번 속성을 찾았고, 이들을 제거한 후 5개의 속성으로 다시 분류 실험을 하였다. 그러나 4개 분류기에서 공통적으로 1개의 인스턴스를 오분류하여 분류 성능이 떨어졌고, 분류에 영향을 준 1개의 인스턴스를 찾기 위해 EM 클러스터링을 이용하였다.

클러스터링 알고리즘은 사용자나 콘텐츠의 군집을 발견하는데 가장 많이 사용되는 알고리즘으로, 자동으로 관련 아이템 군집을 찾아주며 그 결과는 분류기나 예측기 또는 협업 필터링을 만드는데 사용할 수 있다[13].

Fig. 1은 속성의 각 도메인들이 클러스터에 기여하는 영향도를 나타내고 있다. 프로젝트가 소프트웨어 재사용에 성공했는지 실패했는지 두 가지 경우이므로 생성된 클러스터도 두 개다. 이 영향도 점수를 아래 Table 3과 같이 분석해 본 결과, 분류기나 클러스터링에서는 'failure'로 분류했는데 reuse.arff 데이터 셋에서는 'success'로 표시되어 있는 22번 인스턴스를 발견하였다. 'success' 클러스터에 속하려면 영향도 점수 합이 60.373이상이어야 하는데 22번 인스턴스의 경우 34.592이고 이는 'failure'에 해당한다.

이렇게 찾아진 22번 인스턴스를 노이즈 값으로 보고 실험에서 제외시킨 후 다시 분류를 하였다. 이것이 실험 II에 해당된다. 즉, 실험 II는 22번 인스턴스를 제외한 23개 인스턴스와 5개의 속성을 이용하여 분류한 것이고, Table 4를 통해 실험 II를 검증한 결과 9번 속성 제거 시 분류 정확도가 향상됨을 알고 실험 II'에서는 이를 제외한 4개의 속성만으로 분류 실험을 하여 Table 5처럼 주요 속성의 조합을 찾을 수 있었고 Table 6에 보이는 것처럼 100%의 분류 정확도를 보임을 확인하였다.

```
                              Cluster
Attribute                          0       1
                                (0.61)  (0.39)
================================================
Development Approach
  OO                            8.5954  8.4046
  proc                          8.0007  1.9993
  not_available                 1.0513  1.9487
  [total]                      17.6474 12.3526
Non-Reuse Processes Modified
  yes                          14.9117  3.0883
  no                             1.724   7.276
  NA                            1.0117  1.9883
  [total]                      17.6474 12.3526
Human Factors
  yes                          15.5157  2.4843
  no                            1.1317  8.8683
  [total]                      16.6474 11.3526
Domain Analysis
  yes                           9.6025  1.3975
  no                            7.0332  8.9668
  NA                            1.0117  1.9883
  [total]                      17.6474 12.3526
Configuration Management
  yes                          14.9117  3.0883
  no                             1.724   7.276
  NA                            1.0117  1.9883
  [total]                      17.6474 12.3526
```

Fig. 1. EM Clustering run information
(0 : success, 1 : failure)

Table 2. Experiment I verification process

Attribute	Classifier											
	SVM			RBFNetwork			BayesNet			NaiveBayes		
	Cross-validation			Cross-validation			Cross-validation			Cross-validation		
	3	5	10	3	5	10	3	5	10	3	5	10
8,9,11,13,14,16,17,19,24	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 8	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 9	100%	100%	100%	100%	100%	95.83%	100%	100%	100%	100%	100%	100%
without 11	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 13	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 14	95.83%	95.83%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 16	95.83%	100%	100%	100%	100%	100%	95.83%	95.83%	95.83%	95.83%	95.83%	95.83%
without 17	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 19	100%	100%	100%	95.83%	100%	100%	100%	100%	100%	100%	100%	100%
without 24	95.83%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table 3. Misclassified 22nd instance

Instance	9	14	16	19	24	27	Impact score					Total
1	OO	yes	yes	yes	yes	success	8.5954	14.9117	15.5157	9.6025	14.9117	63.537
2	OO	yes	yes	yes	yes	success	8.5954	14.9117	15.5157	9.6025	14.9117	63.537
3	OO	no	no	no	no	failure	8.4046	7.276	8.8683	8.9668	7.276	40.792
4	OO	no	no	no	no	failure	8.4046	7.276	8.8683	8.9668	7.276	40.792
5	OO	no	no	no	no	failure	8.4046	7.276	8.8683	8.9668	7.276	40.792
6	OO	yes	yes	no	yes	success	8.5954	14.9117	15.5157	7.0332	14.9117	60.968
7	OO	yes	yes	no	yes	success	8.5954	14.9117	15.5157	7.0332	14.9117	60.968
8	OO	yes	no	no	no	failure	8.4046	3.0883	8.8683	8.9668	7.276	36.604

9	OO	yes	no	no	no	failure	8.4046	3.0883	8.8683	8.9668	7.276	36.604
10	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
11	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
12	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
13	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
14	OO	no	no	no	yes	failure	8.4046	7.276	8.8683	8.9668	3.0883	36.604
15	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
16	proc	yes	yes	yes	yes	success	8.0007	14.9117	15.5157	9.6025	14.9117	62.942
17	proc	yes	yes	no	yes	success	8.0007	14.9117	15.5157	7.0332	14.9117	60.373
18	OO	yes	yes	no	yes	success	8.5954	14.9117	15.5157	7.0332	14.9117	60.968
19	OO	no	no	no	yes	failure	8.4046	7.276	8.8683	8.9668	3.0883	36.604
20	OO	yes	yes	no	yes	success	8.5954	14.9117	15.5157	7.0332	14.9117	60.968
21	OO	yes	yes	no	yes	success	8.5954	14.9117	15.5157	7.0332	14.9117	60.968
22	OO	no	yes	yes	no	success	8.5954	1.724	15.5157	7.0332	1.724	34.592
23	proc	NA	no	NA	NA	failure	1.9993	1.9883	8.8683	1.9883	1.9883	16.833
24	not_available	no	yes	no	no	failure	1.9487	7.276	2.4843	8.9668	7.276	27.952

Table 4. Experiment II verification process

Attribute	Classifier											
	SVM			RBFNetwork			BayesNet			NaiveBayes		
	Cross-validation			Cross-validation			Cross-validation			Cross-validation		
	3	5	10	3	5	10	3	5	10	3	5	10
9,14,16,19,24	95.65%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 9	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 14	95.65%	95.65%	95.65%	100%	100%	95.65%	100%	100%	100%	100%	100%	100%
without 16	95.65%	91.30%	82.61%	95.65%	100%	100%	95.65%	95.65%	95.65%	95.65%	95.65%	95.65%
without 19	95.65%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
without 24	95.65%	95.65%	95.65%	100%	100%	95.65%	100%	100%	100%	100%	100%	100%

Table 5. Feature subset in Experiment II'

Attribute		Response	Question
14	Non-Reuse Processes Modified	yes(16)	>= 1 nonreuse processes modified
		no(7)	no nonreuse processes modified
16	Human Factors	yes(16)	human factors handled; e.g. via awaness, training, and motivation program
		no(8)	
19	Domain Analysis	yes(9)	domain analysis was performed
		no(14)	
24	Configuration Management	yes(16)	configuration management used
		no(7)	

Table 6. Experiment I, II, II' for finding the key attributes

Classifier	Test option (CV)	All attributes	Feature selection				
			CfsSubsetEval 8,9,11,13,14,16,17,19,24(I)	after experiment I verification 9,14,16,19,24		without 22th instance 9,14,16,19,24 (II)	without 22th instance 14,16,19,24 (II′)
SVM	3	83.33%	100%	95.83%	14 10 9	95.65%	100%
	5	79.17%	100%	91.67%	14 11 8	100%	100%
	10	87.5%	100%	91.67%	14 11 8	100%	100%
RBF Network	3	91.67%	100%	95.83%	14 10 9	100%	100%
	5	95.83%	100%	95.83%	14 10 9	100%	100%
	10	91.67%	100%	95.83%	14 10 9	100%	100%
Bayes Net	3	95.83%	100%	95.83%	14 10 9	100%	100%
	5	100%	100%	95.83%	14 10 9	100%	100%
	10	100%	100%	95.83%	14 10 9	100%	100%
Naive Bayes	3	95.83%	100%	95.83%	14 10 9	100%	100%
	5	95.83%	100%	95.83%	14 10 9	100%	100%
	10	95.83%	100%	95.83%	14 10 9	100%	100%

Table 7은 나머지 분류기들을 이용하여 실험 II′에서 찾은 주요 속성 4개를 검증한 것이다. 전체 속성과 전체 인스턴스를 모두 이용하여 분류한 것에 비해, 22번 노이즈 인스턴스를 제거하고 주요 속성 4개만을 이용하여 분류한 실험 II′의 결과가 3, 5, 10-교차 검증 중 2개 이상에서 모두 향상된 분류 결과를 보였다. 또, MultiClassClassifier, Rotation Forest, AD Tree, RandomTree 분류기의 경우 3, 5, 10-교차 검증 모두에서 향상된 결과를 보였음을 확인하였다.

4. 결과비교

실험 I은 CfsSubsetEval+GreedyStepwise 속성 선택기에 의해 총 26개 중 9개의 속성이 선택되어 이 속성 조합으로 분류했고 4개 분류기 모두 100%의 분류 정확도를 보였다. 실험 I의 결과를 검증하기 위해 9개의 속성을 한 개씩 제거하면서 실제 분류에 영향을 주는 5개의 속성을 찾았지만 Table 6의 실험 I과 II 사이의 결과처럼 분류 성능이 모두 떨어졌고 공통적으로 1개 인스턴스를 잘못 분류했는데, 이 인스턴스를 찾기 위해 EM 클러스터링 기법을 이용하였다. EM 클러스터를 통해 얻은 정보를 이용하여 Table 3과 같이 계산한 결과 22번 인스턴스가 노이즈로 발견되어 실험 II, II′에서는 이 인스턴스를 제거하였다. SVM의 CV-3만 제외하면 모든 경우에서 100%의 분류 성능을 보였고 5개 속성에 대해 검증한 결과 Table 4에 보이는 것처럼 9번 속성 제거 시 분류 정확도가 향상되어 실험 II′에서는 9번 속성을 제거하고 4개 속성만으로 실험하여 분류기 모두에서 100%의 분류 정확도를 보였다. 그리고 나머지 분류기들을 이용하여 실험 II′을 검증한 것이 Table 7에 있고 대체로 분류 성능이 향상되었음을 확인하였다.

Table 7. Experiment II' verification using others

Classifier	Test option	All attributes	Experiment II′ verification
Bagging	3	83.33%	**95.65%**
	5	87.5%	**95.65%**
	10	95.83%	91.30%
ClassificationVia Clustering	3	75%	69.57%
	5	83.33%	**95.65%**
	10	87.5%	**95.65%**
MultiClass Classifier	3	91.67%	**100%**
	5	91.67%	**100%**
	10	91.67%	**100%**
RotationForest	3	87.5%	**100%**
	5	95.83%	**100%**
	10	83.33%	**100%**
AD Tree	3	91.67%	**95.65%**
	5	91.67%	**95.65%**
	10	91.67%	**95.65%**
RandomForest	3	87.5%	**95.65%**
	5	91.67%	**95.65%**
	10	95.83%	95.65%
RandomTree	3	83.33%	**95.65%**
	5	75%	**95.65%**
	10	87.5%	**95.65%**

Table 8에 세 연구의 비교결과가 있다. 'n/a'(not analyzed) 표시는 해당 속성을 실험에서 제외했다는 표시인데, [1]의 연구에서는 해당 속성의 도메인이 너무 많거나 너무 적다는 이유로 'Application Domain', 'Size of Baseline' 속성을 제외했으나 [2]의 연구와 본 연구에서는 제외시킬 이유가 없어 전체 속성을 대상으로 실험하였다. 또 해당 속성이 분류에 영향을 미치지 않았다는 의미로 '×' 표시를, 의미있는 영향을 주었다는 의미로 '√' 표시를 사용하였다.

'Top Management Commitment', 'Repository', 'Human Factors', 'Reuse Processes Introduced', 'Non-Reuse Processes Modified' 가 세 연구에서 공통적으로 주요 속성으로 선택되었다. 특히 'Repository' 속성의 경우 전체 속성을 이용하여 분류한 결과와 이 속성 하나를 제거한 후의 분류 결과가 같았다. 즉 분류에 전혀 영향을 미치지 않는 속성 중 하나에 해당되지만 [1, pp.343, Table2]의 'Repository' 항목에 대한 인터뷰 결과를 보면 총 24개 프로젝트 중 23개가 'yes'로 답하여 경험상 주요 속성 중 하나로 인정하여 세 연구 모두에서 '√' 표시를 했다.

Table 8. Comparison of the results

attribute	Morisio et al.	Menzies et al.	this paper		
			I	II	II′
Application Domain	n/a	×	×	×	×
Size of Baseline	n/a	√	√	×	×
Production Type	√	×	×	×	×
Top Management Commitment	√	√	√	×	×
Reuse Approach	×	√	√	×	×
Domain Analysis	×	√	√	√	√
SP maturity	×	×	×	×	×
Software Staff	×	×	×	×	×
Overall Staff	×	×	×	×	×
Staff Experience	×	×	×	×	×
Type of Software	×	×	×	×	×
Development Approach	×	×	√	√	×
Software and Product	×	×	×	×	×
Origin	×	×	×	×	×
# assests	×	×	×	×	×
Qualification	×	×	×	×	×
Rewards Policy	×	×	×	×	×
Work Products	×	×	×	×	×
Independent Team	×	×	×	×	×
When Assests Built	×	×	×	×	×
Configuration Management	×	×	√	√	√
Key Reuse Roles Introduced	×	√	×	×	×
Repository	√	√	√	√	√
Human Factors	√	√	√	√	√
Reuse Processes Introduced	√	√	√	×	×
Non-Reuse Processes Modified	√	√	√	√	√

기존 연구들에서는 선택되지 않았으나 실험 I, II, II′에서 모두 선택된 것으로 'Configuration Management' 속성이 있다. 이는 '형상관리'로 소프트웨어 소스 코드, 개발 환경, 빌드 구조 등 전반적인 환경 관리 체계를 정의하고 있다. 그리고 하나의 소프트웨어 산출물을 생성하기 위해 필요로 하는 아이템들과 공정 방식의 정의, 그리고 재생성을 위한 전반적인 환경까지 베이스라인화하여 관리하는 방식 전체를 의미함으로[14], 소프트웨어 재사용 여부를 결정할 때 충분한 정보를 줄 수 있고 또 주요 속성임이 본 논문의 실험 결과 검증되었다.

5. 결 론

본 논문은 소프트웨어 재사용의 성공과 실패를 결정하는 주요 요인을 찾고자 했던 기존 연구들과 같은 목적을 갖고 그 결과들을 검증해 보고, 분류 성능을 좀 더 향상시키고자 특징 선택과 분류, 클러스터링을 이용하여 실험하였다. 그 결과, 기존 연구들이 찾은 특징 부분 집합과 비슷한 속성들을 찾아냈고, 분류기가 오분류했던 인스턴스를 클러스터링의 영향도 점수를 이용하여 찾을 수 있었다. 그리고 이렇게 찾아진 특징 부분 집합으로 분류한 결과 기존 연구들보다 더 나은 정확도를 보임을 확인하였다. 이러한 연구들을 통해 소프트웨어 재사용 결정 시 또는 재사용 소프트웨어 채택 시 우선적인 판단 기준을 제시해 줄 수 있다.

향후 연구는 빅 데이터에 속성 선택과 클러스터링을 이용하여 노이즈나 이상치를 찾아 이 요소들이 데이터 마이닝의 실험 결과에 미치는 영향들을 분석해 보는 것이다.

참 고 문 헌

[1] Morisio, M., Ezran, M., Tully, C., "Success and failure factors in software reuse", IEEE Transactions on Software Engineering, Vol.28, Issue 4, pp.340-357, 2002.

[2] Menzies, T., Di Stefano, J.S., "More success and failure factors in software reuse", IEEE Transactions on Software Engineering, Vol.29, Issue 5, pp.474-477, 2003.

[3] IH Witten, E Frank, "Data Mining : Practical Machine Learning Tools and Techniques", Second Edi., Morgan Kaufmann, 2005.

[4] PN Tan, M Steinbach, V Kumar, "Introduction to data mining", Addison-Wesley, 2006.

[5] Masahide Watanabe, Kaihei Kuwata, Ryu Katayama., "Adaptive Tree-Structured Self Generating Radial Basis Function and its Application to Nonlinear Identification Problem", PROCEEDINGS of The 3rd International Conference on Fuzzy Logic, Neural Nets and Soft Computing, pp.167-170, 1994.

[6] Young-Sup Hwang, Sung-Yang Bang, "An Efficient Method to Construct a Radial Basis Function Neural Network Classifier", Journal of KIISE : Software and Applications, Vol.24, Issue 5, pp.451-460, 1997.

[7] Satnam Alag, "Collective Intelligence in Action", Manning Publications Co., 2009.

[8] MA Hall, "Correlation-based feature selection for machine learning", lri.fr, 1999.

[9] Lei Yu, Huan Liu, "Feature Selection for High-Dimensional Data : A Fast Correlation-Based Filter Solution", Proceedings of the Twentieth International Conference on Machine Learning(ICML-2003), Washington DC, 2003.

[10] K. Selvakuberan, M. Indradevi, Dr. R. Rajaram "Combined Feature Selection and classification - A novel approach for the categorization of web pages", Journal of Information and Computing Science, Vol.3, Issue 2, pp.083-089, 2008.

[11] Karim O. Elish, Mahmoud O. Elish, “Predicting defect-prone software modules using support vector machines”, The Journal of Systems and Software, Vol.81, Issue 5, pp.649-660, 2008.

[12] K Michalak, H Kwasnicka, “Correlation-based feature selection strategy in classification problems”, Int. J. Appl. Math. Comput. Sci., Vol.16, Issue 4, pp.503 - 511, 2006.

[13] Richard J. Roiger, Michael W. Geatz, “Data mining a tutorial-based primer”, Addison Wesley, 2003.

[14] “CMMI for Development, Version 1.2”, Carnegie Mellon University, 2006.

[15] Jürgen Börstler, “Feature-Oriented Classification for Software Reuse”, Proceedings SEKE '95, The 7th International Conference on Software Engineering and Knowledge Engineering, Rockville, MD, USA, 1995.

김 영 옥

e-mail : kim052@gwnu.ac.kr

1997년 강릉원주대학교 컴퓨터공학과(학사)

2003년 강릉원주대학교 컴퓨터공학과 (교육학석사)

2013년 강릉원주대학교 컴퓨터공학과 (공학박사)

관심분야 : 소프트웨어 신뢰도, 데이터 마이닝 등

권 기 태

e-mail : ktkwon@gwnu.ac.kr

1986년 서울대학교 계산통계학과(학사)

1988년 서울대학교 계산통계학과(이학석사)

1993년 서울대학교 계산통계학과(이학박사)

1996년 Univ. of Southern California, Post-Doc.

1990년～현 재 강릉원주대학교 컴퓨터공학과 교수

관심분야 : 소프트웨어 비용산정, 소프트웨어 메트릭스, 소프트웨어 아키텍처, 소프트웨어 신뢰도, 데이터 마이닝 등