다중 추상화 수준의 데이터를 위한 결정 트리 분류기

정 민 아[†] · 이 도 헌[††]

요 약

대규모 데이터 마이닝 환경에서는 이질적인 데이터베이스 혹은 파일 시스템으로부터 분석 대상 데이터를 수집하는 경우가 일반적이므로, 수집된 데이터가 서로 다른 추상화 수준(abstraction level)으로 표현되기 마련이다. 본 논문에서는 기존의 결정 트리(decision tree)를 서로 다른 추상화 수준으로 표현된 데이터에 적용할 때, 분류상 모순이 일어날 수 있음을 보이고, 그에 대한 해결방안을 제시한다. 제안하는 방법은 데이터간에 존재하는 일반화/세분화 관련성을 결정 트리의 구축 단계는 물론, 클래스 할당 단계에도 반영하여 데이터간의 의미적 연관성을 효과적으로 활용할 수 있도록 한다. 아울러 실제 데이터에 기반을 둔 실험을 통해, 제안한 방법이 기존 방법보다 분류 오류율을 현저히 줄일 수 있음을 보인다.

Decision Tree Classifier for Multiple Abstraction Levels of Data

Mina Jeong[†] · Doheon Lee[††]

ABSTRACT

Since the data is collected from disparate sources in many actual data mining environments, it is common to have data values in different abstraction levels. This paper shows that such multiple abstraction levels of data can cause undesirable effects in decision tree classification. After explaining that equalizing abstraction levels by force cannot provide satisfactory solutions of this problem, it presents a method to utilize the data as it is. The proposed method accommodates the generalization/specialization relationship between data values in both of the construction and the class assignment phase of decision tree classification. The experimental results show that the proposed method reduces classification error rates significantly when multiple abstraction levels of data are involved.

키워드 : 데이터 마이닝(Data Mining), 결정 트리(Decision Tree), 추상화 수준(Abstraction Level), 데이터 품질(Data Quality), 분류(Classification)

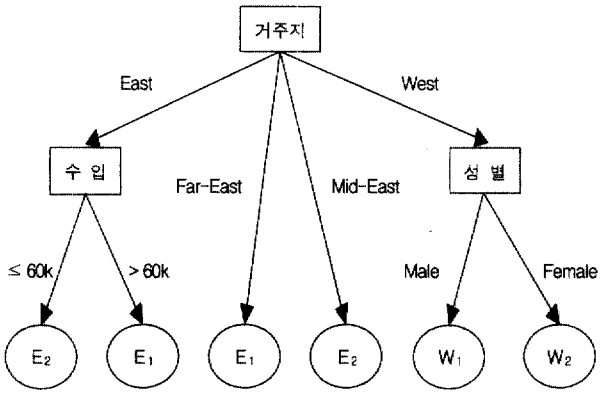
1. 서 론

데이터 분류(classification)는 대표적인 데이터 마이닝 작업 중의 하나이며 통상 학습(training)단계와 할당(assignment)단계로 구성된다. 학습 단계에서는 주어진 학습 데이터를 이용하여 분류 모델(classification model)을 구축한다. 할당 단계에서는 구축한 분류 모델을 이용하여 아직 분류되지 않은 레코드의 클래스를 결정한다. 현재까지 제안된 분류 모델은 Bayesian 분류, 유전자 알고리즘(genetic algorithms), 신경망(neural networks), 결정 트리(decision trees) 등이 있으며, 이러한 분류 모델 중 결정 트리는 인간이 이해하기 쉬운 형태를 갖고 있기 때문에 데이터 마이닝 작업에 특히 유용하다[1-6].

기존의 데이터 분류 기법은 대부분 단일 정보원(information source)으로부터 미리 잘 정리된 학습 데이터를 획득하는 것을 가정하고 있다. 하지만, 실제 데이터 마이닝 환경에서는 여러 데이터베이스 혹은 파일 시스템으로부터 학습 데이터를 추출해야 하는 경우가 일반적이다. 예를 들어 판매 항목을 기술한 한 원시 테이블에 'Coke'라고 표현되어 있는 데이터가 다른 원시 테이블에는 'Diet Coke 1.5 P.E.T'라고 표현될 수 있다. 또한 네트워크 알람 관리 시스템에서 한 지역 네트워크 관리 에이전트는 "라우터 비정상"과 같은 높은 추상화 수준으로 알람을 보고하지만, 다른 지역 네트워크 관리 에이전트는 '유형-32 라우터 메시지 오버플로우'와 같이 낮은 추상화 수준으로 알람을 보고할 수 있다[7]. 이와 같은 문제를 다중 추상화 수준 문제(multiple abstraction level problem)라 정의한다. 또한, 수작업으로 입력된 데이터에도 이러한 문제가 발생할 수 있다. 정교한 데이터 입력 소프트웨어를 이용하거나 직원 학습 프로그램(employee training programs)을 통하여 데이터 입력을 표준화하려는 노력이 있으나, 시스템의 기능상 한계 및 정보의 부재 때문에 다중 추상화 수준 문제를 완전히 피할 수는 없다[8].

† 정 회 원 : 광주과학기술원 정보통신학과 Post-Doc.
†† 정 회 원 : 한국과학기술원 바이오시스템학과 교수
논문접수 : 2002년 7월 4일, 심사완료 : 2002년 10월 29일

이러한 데이터의 다중 추상화 수준은 효과적인 결정 트리 구축에 심각한 문제를 야기할 수 있다. (그림 1)에서와 같이 서로 다른 추상화 수준으로 표현되어 있는 학습 데이터로부터 결정 트리를 구축했다고 가정하자. 'Far-East'와 'Mid-East'는 'East'의 세분화이므로 'Far-East'와 'Mid-East'의 추상화 수준과 'East'와 'West'의 추상화 수준은 다르다. 이때, 아직 분류되지 않은 레코드(Mid-East, 85k, Male)의 클래스를 결정한다고 가정하자. 한가지 방법은 루트 노드의 세 번째 가지를 따라 그 레코드는 'E_2'클래스에 속한다고 결론 짓는 것이다. 다른 방법은 'Mid-East'가 'East'에 속하기 때문에 루트 노드의 첫 번째 가지를 따르고 '수입'이 '60k'보다 크므로 '수입' 노드의 두 번째 가지를 따라 레코드가 'E_1' 클래스에 속한다고 결론 짓는 것이다. 이처럼 동일한 레코드에 대하여 서로 상반된 결론을 얻게 될 수 있다.



(그림 1) 데이터의 다중 추상화 수준을 갖는 결정 트리

이러한 데이터 품질(data quality) 문제는 오랫동안 논쟁의 주제가 되어 왔으므로[8, 9], 데이터 품질을 개선할 수 있는 다양한 데이터 정화(data cleansing) 도구들이 개발되고 있다[10-12]. 데이터 정화 도구를 이용하여 다중 추상화 수준 문제를 해결하기 위해 두 가지 방안을 고려할 수 있다. 첫 번째 방안은 상향 평준화 방법으로서, 낮은 추상화 수준을 가진 데이터를 일정한 수준으로 높이는 방법이다. 하지만, 데이터가 높은 추상화 수준을 가질수록 구체성이 떨어지기 때문에 얻게 되는 분류 모델의 구체성이 떨어지게 된다[4]. 결국 이미 확보한 유용한 정보까지 활용하지 못하는 정보 손실 문제가 발생한다. 두 번째 방안은 하향 평준화 방법으로서, 높은 추상화 수준을 가진 데이터를 일정한 수준으로 낮추는 방법이다. 하지만 데이터의 추상화 수준을 낮추기 위해서는 부가적인 정보가 필요하다. 따라서 이 방안 역시 실제 환경에서는 적용하기 어렵거나 불가능하다. 더욱이 두 가지 방안에서 데이터의 추상화 수준을 어느 수준까지 높이거나 낮춰야 하는가를 결정하기도 어렵다. 결국 기존의 데이터 정화 도구를 이용한 방법으로 다중 추상화 수준 문제를 해결하기 곤란하다.

결정 트리를 구축하는 과정에 퍼지 개념을 적용한 퍼지 결정 트리는 결정 트리를 구축하는 과정에서 데이터의 부정확성(inexactness)과 불확실성(uncertainty)을 다루기 위해 연구되어 왔으며[16-18], 주로 연속적인 값을 갖는 속성을 처리하기 위해 사용되었다. 퍼지 결정 트리를 구축하기 위해 모든 속성에 대한 퍼지 집합은 사용자가 결정하며, 각 데이터 레코드의 소속 정도 값을 이용하여 결정 트리를 구축한다. 퍼지 결정 트리 구축 알고리즘은 퍼지 개념에 근거한 최적 분할 속성을 결정하는 척도를 사용하며, 결정 트리의 잎 노드에 한 개 이상의 클래스가 할당되는 특징을 갖는다. 그러나 퍼지 결정 트리를 구축하는 알고리즘을 사용할 경우 퍼지 결정 트리를 구축하기 위해 이용하는 학습 데이터들은 결정 트리를 구축하기 전에 각 레코드가 적당한 소속정도를 갖도록 퍼지화하는 부가적인 작업이 필요하다. 또한, 최적 분할 속성을 선택하기 위해 이용하는 소속정도는 미리 퍼지화하는 과정에서 계산되어 결정 트리를 구축하는 전체 과정에 이용하는데, 소속정도는 단순히 레코드가 어떤 퍼지집합에 대하여 소속되는 정도를 나타낼 뿐, 실제 위에서 제시한 데이터 값들 사이의 추상화 수준을 반영하고 있지 않다. 그러므로 퍼지 결정 트리를 이용하여 다중 추상화 문제를 해결하기 곤란하다.

본 논문에서는 결정 트리 분류에서의 다중 추상화 수준 문제를 해결하기 위한 보다 실용적인 방법을 제안한다. 데이터 값을 강제로 일반화하거나 세분화하기 보다는 존재하는 정보를 그대로 사용하며, 학습과 할당 단계 모두에서 데이터 값들 사이의 일반화/세분화 관련성을 반영한다. 본 논문의 구성은 다음과 같다. 2장에서는 결정 트리 구축을 위한 기존의 알고리즘을 설명하고, 본 논문에서 제안한 방법을 위해 수정한 최적 분할(best-split) 속성을 선택하는 부분과 학습 데이터 집합을 분할하는 부분에 대하여 기술한다. 3장에서는 데이터가 다중 추상화 수준으로 표현될 때 분류되지 않은 레코드의 클래스를 결정하는 방법에 관하여 기술한다. 4장에서는 실험을 통하여 제안한 알고리즘의 성능을 평가한다. 5장에서는 결론을 맺는다.

2. 결정 트리 구축

결정 트리를 구축하기 위해서는 학습 데이터가 필요하다. 학습 데이터는 한 개 이상의 예측 속성(predictor attriubutes)과 한 개의 클래스 레이블(class label)로 구성되는 데이터 릴레이션이다. 즉 결정 트리는 예측 속성과 클래스 레이블간의 관계를 표현하는 트리 구조라 볼 수 있다[1, 2, 4]. 본 장에서는 학습 데이터가 다중 추상화 수준으로 표현되어 있을 때 결정 트리를 어떻게 구축하는 가에 대하여 설명한다.

결정 트리는 학습 데이터의 순환적 분할(recursive partitioning)을 통해 구축된다. 결정 트리를 구축하는 첫 번째 단

계에서는 여러 예측 속성중 학습 데이터를 최적으로 분할하는 속성을 선택하는 것이다. 최적 분할 예측 속성(best-split predictor attribute)이 선택되면 학습 데이터는 그 속성의 값에 따라 부분 집합들로 분할된다. 이러한 선택-분할(selection-and-partitioning)과정은 각 부분 집합에 순환적으로 적용된다. (그림 2)는 결정 트리를 구축하기 위한 알고리즘이다.

```
ConstructTree(Node ThisNode, Relation R) {
(1) Attr = SelectBestSplit(R) ; // R을 최적으로 분할하는 속성 선택
(2) ThisNode.label = Attr ; // Attr를 현재노드의 레이블로 표시
(3) For each attribute value x of Attr
(4) {
        // Attr 속성의 값이 'x'인 레코드를 선택
(5)         NewR = SelectRecords(R, Attr, x) ;
        // 새로운 결정 노드를 생성
(6)         NewNode = NewNode() ;
        // NewNode를 현재노드의 자식노드로 함
(7)         NewNode.parent = ThisNode;
        // 자식 노드에 대하여 ConstructTree()를 호출
(8)         ConstructTree(NewNode, NewR) ; } }
```

(그림 2) 결정 트리 구축을 위한 알고리즘

(1)에서 릴레이션 R의 예측 속성 중의 하나가 최적 분할 속성으로 선택된다. 최적 분할 속성을 선택하기 위한 다양한 방법이 제안된 바 있는데, 결국 공통적으로 추구하는 것은 가급적 동일한 클래스 레이블을 갖는 부분집합으로 분할하는 속성을 선택하는 것이다[4, 12]. 예를 들어 은행 고객 정보에 대한 학습 데이터가 존재하고, 클래스 레이블은 '주거래 지점'이라고 가정하자. 또한 두 개의 예측 속성 '거주지'와 '직업'이 있다고 가정한다. '주거래 지점'과 '거주지'사이에는 관련성이 많기 때문에 '거주지'에 따라 분할하면 각 부분집합은 주로 동일한 지점들에 대한 레코드들을 포함할 것이다. 반면, '직업'에 따라 분할한다면 각 부분집합은 서로 다른 지점들에 대한 레코드들을 혼합하여 포함할 것이다. 즉 '거주지'에 따라 분할한 경우가 '직업'에 따라 분할한 경우에 비해 보다 동질적인 레코드로 구성된 부분집합을 생성시킨다. 결과적으로 '거주지'가 클래스 레이블인 '주거래 지점'에 대한 최적 분할 속성으로 선택된다.

'거주지'가 최적 분할 속성으로 선택되면 (4)에서 학습 데이터를 분할한다. 만약 학습 데이터에 '거주지'에 대한 값이 k개가 존재한다면 이 속성에 대하여 k개의 가지들이 생성된다. '거주지'값이 i번째인 학습 데이터의 레코드는 i번째 가지를 따라 할당된다. 이러한 선택-분할 과정은 (7)에서 각 부분집합에 대하여 순환적으로 수행된다.

결과적으로 학습 데이터가 다중 추상화 수준으로 표현되어 있을 때 최적 분할 속성에 따라 학습 데이터를 분할하는 부분((5)에서의 SelectRecords())과 최적 분할 속성을 선택하는 부분((1)에서의 SelectBestSplit())은 다중 추상화 수준을 고려하여 수정되어야 한다.

2.1 다중 추상화 수준을 나타내는 학습 데이터의 분할

최적 분할 속성에 따라 학습 데이터를 분할하는 부분이 수정되어야 하는 이유를 예제를 통하여 설명한다. <표 1>과 같은 은행 고객 정보에 대한 학습 데이터를 가지고 있다고 가정하자.

<표 1> 고객 정보에 대한 학습데이터

레코드 ID	거주지	성 별	수 입	주거래지점
t_1	East	Male	90k	E_2
t_2	East	Male	70k	E_1
t_3	Far-East	Female	80k	E_1
t_4	Mid-East	Male	50k	E_2
t_5	Mid-East	Female	30k	E_2
t_6	West	Male	90k	W_1
t_7	West	Male	50k	W_1
t_8	West	Female	100k	W_2
t_9	West	Female	40k	W_2
t_{10}	West	Female	50k	W_2

이 학습 데이터는 세 개의 예측 속성 '거주지', '성별'과 '수입'과 하나의 클래스 레이블 '주거래 지점'으로 구성되어 있다. 실제 데이터 마이닝 환경에서 학습 데이터는 훨씬 많은 수의 레코드들을 포함하지만 간단한 설명을 위해 예제에서는 10개의 레코드만을 제시하였다. 일단, 예측 속성 '거주지'가 클래스 레이블인 '주거래 지점'에 대하여 최적 분할 속성으로 선택되었다고 가정하자. 최적 분할 속성을 선택하는 방법은 2.2절에서 자세하게 언급한다. 먼저 '거주지'에 따라 학습 데이터를 분할한다. 기존의 알고리즘에 의해 분할한 결과는 $\{t_1, t_2\}$, $\{t_3\}$, $\{t_4, t_5\}$, $\{t_6, t_7, t_8, t_9, t_{10}\}$과 같다. 그러나 이것은 'Far-East'와 'Mid-East'가 'East'에 속한다는 사실을 반영하지 않고 있다. 그러므로 '거주지'값이 'Far-East'이거나 'Mid-East'인 레코드들을 'East'에 대한 첫 번째 부분 집합에 포함하여 $\{t_1, t_2, t_3, t_4, t_5\}$, $\{t_3\}$, $\{t_4, t_5\}$, $\{t_6, t_7, t_8, t_9, t_{10}\}$으로 분할되도록 한다. 그러나 첫 번째와 두 번째 고객의 거주 지역이 'East'지만 실제 고객의 거주 지역이 'Far-East'이거나 'Mid-East'가 될 수 있다는 사실은 여전히 반영되지 않았다. 적절한 학습 데이터는 전체 영역에 대한 값의 분포를 반영한다고 간주할 수 있으므로 학습 데이터의 분포로부터 실제 거주 지역에 대한 확률을 얻을 수 있다. 세 개의 레코드 t_3, t_4, t_5는 'East'의 세분화된 값을 갖는다. 이들 중 한 레코드인 t_3은 'Far-East'값을 갖고, 다른 두 레코드인 t_4, t_5는 'Mid-East'값을 갖는다. 이러한 분포로부터 t_1(또는 t_2)의 '거주지'값이 실제 'Far-East'일 확률은 1/3 = 33%이다. 이와 유사하게 t_1(또는 t_2)의 '거주지'값이 'Mid-East'일 확률은 2/3 = 67%이다. 이러한 사실을 반영하여 학습 데이터는 $\{t_1, t_2, t_3, t_4, t_5\}$, $\{t_1/0.33, t_2/0.33, t_3\}$, $\{t_1/0.67, t_2/0.67, t_4, t_5\}$, $\{t_6, t_7, t_8, t_9, t_{10}\}$으로 분할되며, 이 때 t_i/μ는 레코드 t_i가 소속 정도 μ값으로 부분

집합에 포함된다는 사실을 나타내고 있다. 정의 1과 정의 2는 이러한 사실을 정형화한 것이다.

제시한 예제에서와 같이 부분 집합의 각 레코드에 대하여 부분적 소속 정도를 표현하는데 퍼지 릴레이션(fuzzy relation)개념을 도입한다.

정의 1 (퍼지 릴레이션)
퍼지 릴레이션 T를 다음과 같이 정의한다.

$$T = \{ (t, \mu_T(t)) \mid t\text{는 보통의 레코드}, \ \mu_T(t)\text{는 T에 대한 t의 소속 정도} \}$$

소속 정도 $\mu_T(t)$는 레코드가 집합에 속하는 정도를 표현한다. $\mu_T(t)=1$는 완전한 소속 정도를 의미하고, $\mu_T(t)<1$는 부분적 소속 정도를 의미한다. 보통의 릴레이션은 모든 레코드가 소속정도 1.0을 갖는 퍼지 릴레이션의 특별한 경우라 할 수 있다. 따라서 지금부터 학습 데이터는 퍼지 릴레이션으로 간주한다.

정의 2 (학습 데이터 분할)
퍼지 릴레이션 T(학습 데이터)와 속성 X의 영역에 대한 ISA 계층구조 H가 주어졌을 때 'X'값이 'x'인 레코드를 T로부터의 선택하는 SR(T, H, X, x)는 다음과 같이 정의한다.

$$SR(T,H,X,x) = SR_{direct}(T,X,x) \cup SR_{descendent}(T,H,X,x) \cup SR_{antecedent}(T,H,X,x),$$

단, $SR_{direct}(T,X,x) = \{t, \mu(t)) \mid t \in T,\ t.X = x, \mu(t) = \mu_T(t)\}$,

$SR_{descendent}(T,H,X,x) = \{t, \mu(t)) \mid t \in T, t.X \in DESC(x,H), \mu(t) = \mu_T(t)\}$,

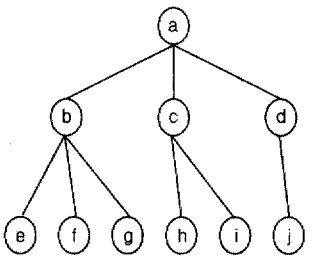
$SR_{antecedent}(T,H,X,x) = \{t, \mu(t)) \mid t \in T, t.X \in ANTE(x,H), \mu(t) = \mu_T(t) \times (Card(\{s \mid s \in T,\ s.X = x \text{ or } s.X \in DESC(x,H)\}) / Card(\{s \mid s \in T,\ s.X \in DESC(t.X,H)\}))\}$.

$ANTE(x)$와 $DESC(x)$는 ISA 계층구조 H에서 각각 x의 조상노드와 후손노드의 집합을 의미하며, $Card(T) = \Sigma\mu_t(t)$이다.

'X'값이 'x'인 레코드들을 집합 T로부터 선택한 후의 결과 집합 즉 $SR(T,H,X,x)$는 세 부분 집합으로 구성된다. 첫 번째 부분집합 $SR_{direct}(T,X,x)$는 'X'값이 'x'와 정확히 일치하는 레코드들을 포함하는 집합이다. 이때 $\mu(t)$값은 집합 T에 대한 레코드 t의 소속 정도 $\mu_T(t)$가 된다. 두 번째 부분집합 $SR_{descendent}(T,H,X,x)$는 'X'값이 'x'의 세분화인 레코드들의 집합이다. 앞 예제에서 'Far-East'는 'East'의 세분화 값이다. 이러한 경우, "*X가 세분화 값 x'이면, X는 일반화 값 x이다*" 라는 명제가 항상 참이므로, $\mu(t)$값은 $\mu_T(t)$와 같다. 세 번째 부분집합 $SR_{antecedent}(T,H,X,x)$는 'X'값이 'x'의 일반화인 레코드들의 집합이다. 이러한 경우 $\mu(t)$값은 1.0이하의 부분적 소속 정도를 갖는데, 이것은 "*X가 일반화 값 x'이면 X는 세분화 값 x이다*" 라는 명제가 부분적으로 참이기 때문이다.

정의 2에서 제시한 소속 정도 할당의 예를 보이기 위해 (그림 3)을 제시한다. 결정 트리 구축을 위한 실제 학습 데이터는 복수개의 예측 속성을 갖는 것이 일반적이나 (그림 3)에서는 이해의 편의를 돕기 위해 한 개의 예측 속성만을 제시한다. 속성 'X'에 대한 ISA 계층구조는 'X'에 대한 값들 사이의 일반화/세분화 관련성을 표현한다.

H : 속성 X에 대한 ISA 계층구조



T : 학습 데이터

RID	X	μ(ti)	RID	X	μ(ti)
t_1	a	1.0	t_9	b	1.0
t_2	c	1.0	t_{10}	c	1.0
t_3	d	1.0	t_{11}	f	1.0
t_4	h	0.7	t_{12}	j	0.4
t_5	a	0.9	t_{13}	b	0.6
t_6	c	1.0	t_{14}	d	1.0
t_7	e	0.9	t_{15}	f	1.0
t_8	i	1.0	t_{16}	g	0.9

(그림 3) 학습 데이터의 한 속성 값에 대한 ISA 계층구조와 퍼지 릴레이션

정의 2에 의해,

$$SR(T,H,X,b) = SR_{direct}(T,X,b) \cup SR_{descendent}(T,H,X,b) \cup SR_{antecedent}(T,H,X,b) \quad (1)$$

t_9, t_{13}의 X값이 'b'이므로,

$$SR_{direct}(T,X,b) = \{t_9/1.0,\ t_{13}/0.6\} \quad (2)$$

'e', 'f'와 'g'가 'b'의 하위노드이므로 t_7, t_{11}, t_{15}와 t_{16}은 그들 중 하나의 값을 갖는다.

$$SR_{descendent}(T,H,X,b) = \{t_7/0.9,\ t_{11}/1.0,\ t_{15}/1.0,\ t_{16}/0.9\} \quad (3)$$

'a'가 'b'의 상위노드이므로 t_1, t_5는 X에 대하여 'a'를 갖는다.

$$SR_{antecedent}(T,H,X,b) = \{t_1/\mu(t_1),\ t_2/\mu(t_2)\} \quad (4)$$

DESC(a) = {b, c, d, e, f, g, h, i, j}이고 DESC(b) = {e, f, g}이므로,

$$\mu(t_1) = \mu_T(t_1) \times Card(\{t_7/0.9,\ t_9/1.0,\ t_{11}/1.0,\ t_{13}/0.6,\ t_{15}/1.0,\ t_{16}/0.9\}) / Card(\{t_2/1.0,\ t_3/1.0,\ t_4/0.7,\ t_6/1.0,\ t_7/0.9,\ t_8/1.0,\ t_9/1.0,\ t_{10}/1.0,\ t_{11}/1.0,\ t_{12}/0.4,\ t_{13}/0.6,\ t_{14}/1.0,\ t_{15}/1.0,\ t_{16}/0.9\}) = 1.0 \times 5.4/12.5 = 0.43 \quad (5)$$

$$\mu(t_5) = \mu_T(t_5) \times \text{Card}(\{t_7/0.9, t_9/1.0, t_{11}/1.0, t_{13}/0.6, t_{15}/1.0, t_{16}/0.9\}) / \text{Card}(\{t_2/1.0, t_3/1.0, t_4/0.7, t_6/1.0, t_7/0.9, t_8/1.0, t_9/1.0, t_{10}/1.0, t_{11}/1.0, t_{12}/0.4, t_{13}/0.6, t_{14}/1.0, t_{15}/1.0, t_{16}/0.9\})$$
$$= 0.9 \times 5.4/12.5 = 0.39 \quad (6)$$

위의 식 (1)~식 (5)와 식 (6)으로부터

$$SR(T, H, X, b) = \{t_9/1.0, t_{13}/0.6\} \cup \{t_7/0.9, t_{11}/1.0, t_{15}/1.0, t_{16}/0.9\} \cup \{t_1/0.43, t_5/0.39\}$$
$$= \{t_1/0.43, t_5/0.39, t_7/0.9, t_9/1.0, t_{11}/1.0, t_{13}/0.6, t_{15}/1.0, t_{16}/0.9\}$$

이와 같이 기존의 결정 트리 알고리즘에서는 t_9와 t_{13}만이 선택되나 본 논문에서 제안한 방법은 부분적인 소속 정도를 갖는 t_1, t_5, t_7, t_9, t_{11}, t_{13}, t_{15},와 t_{16}을 포함한다.

2.2 최적 분할 속성의 선택

본 절에서는 최적 분할 속성을 선택하는 방법에 관하여 기술한다. 최적 분할 속성은 학습 데이터를 가급적 동일한 클래스 레이블을 갖는 부분집합들로 분할하는 속성이다. 이러한 집합의 이질성(heterogeneity)을 평가하기 위해 많은 척도들이 존재한다[4, 12]. 본 논문에서는 정보 이론의 척도인 엔트로피(entropy)를 이용하는데, 이것은 실제 데이터 마이닝 환경에서 가장 많이 사용하는 것 중의 하나이다. 다음 정의 3에서는 엔트로피를 확장함으로써 퍼지 릴레이션의 이질성을 측정하는 방법을 정의한다[4].

정의 3 (퍼지 릴레이션의 엔트로피)

클래스 레이블 C의 영역이 $\{c_1, c_2, \cdots, c_m\}$이고, 주어진 퍼지 릴레이션 T를 속성 C의 값에 따라 $T^{c1}, \cdots, T^{cm}$으로 분할한다고 가정한다. 이때, $T^{ci} = \{(t, \mu(t)) \mid t \in T, t.C = cj, \mu(t) = \mu_T(t)\}$이다. 또한, 속성 X에 대한 영역이 $\{x_1, x_2, \cdots, x_m\}$이고, T를 X값에 따라 $T^{x1}, \cdots, T^{xm}$으로 분할한다고 가정한다. 이때, $T^{xj} = \{(t, \mu(t)) \mid t \in T, t.C = cj, \mu(t) = \mu_T(t)\}$이다. X는 분할을 위한 속성이며, X나 C에 대한 T의 엔트로피는 $info^C(T)$, $info^X(T)$로 표현하고 다음과 같이 정의한다.

$$info^C(T) = -\sum_{c_i \in C} [Card(T^{c_i})/Card(T) \times \log_2(Card(T^{c_i})/Card(T))],$$
$$info^X(T) = -\sum_{x_i \in X} [Card(T^{x_i})/Card(T) \times info^S(T)],$$

단, $S = \{c_i \mid$ 속성 값이 x_i인 레코드의 클래스 레이블 $\}$,
$Card(T^k) = \sum_{t \in T^k} \mu_{T^k}(t)$, $k \in C$ 또는 $k \in X$ 이고
$Card(T) = \sum_{t \in T} \mu_T(t)$

C4.5와 같은 기존의 방법들에서는 분할 정보(split info)와 같은 척도를 부가적으로 도입하여 레코드의 속성 값이 거의 다른 값들을 갖는 예측 속성의 편차를 줄이고자 한다[4]. 분할 정보는 한 집합에 대한 엔트로피라 할 수 있는데, 앞에서 제시한 예제에서 레코드를 구분하는 속성은 클래스 레이블이 아닌 예측 속성 '거주지'라고 할 수 있다. 정보 획득 값을 분할 정보로 나눔으로써 거의 다른 값을 갖는 속성에 대한 편차를 줄이고자 하며, 다음 정의 4에서는 퍼지 릴레이션에 대한 분할 정보 척도를 정의한다.

정의 4 (퍼지 릴레이션의 분할 정보(Split info))

속성 A의 영역이 $\{a_1, a_2, \cdots, a_m\}$이고, 주어진 퍼지 릴레이션 T를 속성 A의 값에 따라 $T^{a1}, \cdots, T^{am}$으로 분할한다고 가정한다. 이때, $T^{aj} = \{(t, \mu(t)) \mid t \in T, t.A = aj, \mu(t) = \mu_T(t)\}$이다. 또한 속성 A에 대한 T의 분할 정보는 split info(A)로 표현하고 다음과 같이 정의한다.

$$split\ info(A) = -\sum_{j=1,\cdots,m} [Card(T^{a_j})/Card(T) \times \log_2(Card(T^{a_j})/Card(T))],$$
단, $Card(T^{a_j}) = \sum_{t \in T^{a_j}} \mu_{T^{a_j}}(t)$ 이고
$Card(T) = \sum_{t \in T} \mu_T(t)$

이러한 정의에 의해 <표 1>의 학습 데이터에 대한 엔트로피를 '주거래지점(FVB)'에 대하여 계산한다. 학습 데이터는 보통의 릴레이션이므로 각 레코드의 소속 정도는 1.0이다. '주거래 지점' 속성은 {'E_1', 'E_2', 'W_1', 'W_2'}의 값을 갖기 때문에 학습 데이터는 다음과 같이 4개의 부분집합들로 분할된다.

$T^{c1} = \{t_2/1.0, t_3/1.0\}$ ('FVB' = 'E_1'인 경우),
$T^{c2} = \{t_1/1.0, t_4/1.0, t_5/1.0\}$ ('FVB' = 'E_2'인 경우),
$T^{c3} = \{t_6/1.0, t_7/1.0\}$ ('FVB' = 'W_1'인 경우),
$T^{c4} = \{t_8/1.0, t_9/1.0, t_{10}/1.0\}$ ('FVB' = 'W_2'인 경우).

그러므로 $Card(T^{c1}) = 2.0$, $Card(T^{c2}) = 3.0$, $Card(T^{c3}) = 2.0$, $Card(T^{c4}) = 3.0$이고 10개의 레코드가 각각 소속 정도 1.0을 갖기 때문에 $Card(T) = 10.0$이다. 결과적으로 $info^C(T) = -[2.0/10.0 \times \log_2(2.0/10.0) + 3.0/10.0 \times \log_2(3.0/10.0) + 2.0/10.0 \times \log_2(2.0/10.0) + 3.0/10.0 \times \log_2(3.0/10.0)] = 1.97$이다.

세 개의 예측 속성 '거주지', '성별', '수입'들 중 최적 분할 속성을 선택하기 위해 먼저 '거주지(RA)'를 고려한다. 정의 2에서 제시한 분할 방법을 이용하여 '거주지'값에 따라 학습 데이터를 분할하면 다음과 같다.

$T_1 = \{t_1/1.0, t_2/1.0, t_3/1.0, t_4/1.0, t_5/1.0\}$
('RA' = 'East'인 경우),

T_2 = {t_1/0.33, t_2/0.33, t_3/1.0}
('RA' = 'Far-East'인 경우),
T_3 = {t_1/0.67, t_2/0.67, t_4/1.0, t_5/1.0}
('RA' = 'Mid-East'인 경우),
T_4 = {t_6/1.0, t_7/1.0, t_8/1.0, t_9/1.0, t_{10}/1.0}
('RA' = 'West'인 경우).

다음, 정의 3에서 제시한 식을 이용하여 클래스 레이블 '주거래 지점'에 대하여 각 부분 집합의 엔트로피를 다음과 같이 계산한다.

$$\text{info}^{FVB}(T_1) = (5.0/15.9)\times(-[\,2.0/5.0 \times \log_2(2.0/5.0) + 3.0/5.0 \times \log_2(3.0/5.0)\,]) = 0.42,$$

$$\text{info}^{FVB}(T_2) = (1.66/15.9) \times (-[\,1.33/1.66 \times \log_2(1.33/1.66) + 0.33/1.66 \times \log_2(0.33/1.66)\,]) = 0.07,$$

$$\text{info}^{FVB}(T_3) = (3.34/15.9) \times (-[\,0.67/3.34 \times \log_2(0.67/3.34) + 2.67/3.34 \times \log_2(2.67/3.34)\,]) = 0.15,$$

$$\text{info}^{FVB}(T_4) = (5.0/15.9) \times (-[\,2.0/5.0 \times \log_2(2.0/5.0) + 3.0/5.0 \times \log_2(3.0/5.0)\,]) = 0.30.$$

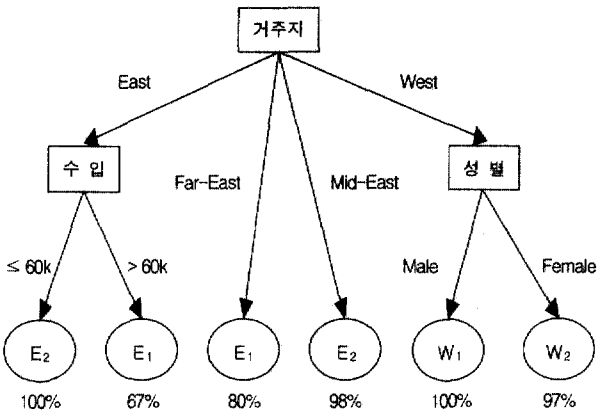
정의 4에서 제시한 식을 이용하여 클래스 레이블 '주거래 지점'에 대하여 '거주지'에 대한 부분 집합의 분할 정보 값을 다음과 같이 계산한다.

$$\text{spit info}^{RA}(T) = -[\,5.0/15.9 \times \log_2(5.0/15.9) + 1.66/15.9 \times \log_2(1.66/15.9) + 3.34/15.9 \times \log_2(3.34/15.9) + 5.0/15.9 \times \log_2(5.0/15.9)\,] = 1.86$$

이러한 엔트로피 값들에 대한 합은 0.42 + 0.07 + 0.15 + 0.30 = 0.94이다. 이때, 엔트로피 값은 1.97에서 0.94로 1.03만큼 감소되었다고 말할 수 있다. 즉, 속성 '거주지'에 의해 학습 데이터가 분할됨으로써 얻어지는 정보 획득(gain)은 1.03bits이다. 만약, 분할 정보를 적용할 경우 앞에서 계산한 바와 같이 정보 획득은 1.03이므로 결과적으로 얻는 정보 획득율(gain ratio)은 1.03/1.86 = 0.55이다. 이러한 정보 획득 값을 분할 정보로 나눔으로써 거의 다른 값들을 갖는 속성에 대한 불이익을 제거할 수 있으며, 퍼지 릴레이션에 대한 분할 정보 척도는 정의 2에서 제시한 식으로 계산될 수 있다. 이와 유사하게, 속성 '성별'과 '수입'에 대하여 각각 정보 획득 또는 정보 획득율을 계산하여 이 중 가장 높은 값을 갖는 속성을 최적 분할 속성으로 선택한다.

3. 결정 트리에 의한 클래스 레이블 할당

제안한 방법을 이용하여 구축된 결정 트리가 (그림 4)의 트리라고 가정하자.



(그림 4) 신뢰도를 갖는 결정 트리의 예

각 단말 노드에 부착된 레이블은 클래스를 할당하기 위한 신뢰도를 나타낸다[4]. 아직 분류되지 않은 레코드인 (East, Male, 85k, Unknown)의 '주거래 지점' 즉 클래스 레이블이 알려지지 않았다고 할때 레코드의 클래스를 결정한다고 하자. 기존의 방법을 이용할 경우, 레코드는 뿌리 노드로부터 첫 번째 가지와 '수입' 노드의 두 번째 가지를 따라 67%의 신뢰도를 갖는 'E_1'클래스에 할당된다. 그러나 'East'가 사실 'Mid-East'일 수 있다는 사실을 반영해야 한다. 이때, 전체 학습 데이터 중 '거주지' 속성 값이 'Mid-East'인 레코드가 85%라고 가정하자. 이러한 분포를 반영하여 뿌리 노드의 세 번째 가지를 따라 'E_2'클래스에 신뢰도 85% × 98% = 83%로 할당될 수 있다. 따라서 'E_2'에는 83%의 신뢰도로 할당될 수 있으며, 'E_1'에는 67%의 신뢰도로 할당될 수 있다. 결국 레코드는 이 중 가장 큰 신뢰도 값을 갖는 'E_2'에 할당된다. (그림 5)는 이러한 클래스 할당 과정에 대한 알고리즘이다.

```
AssignClass(DecisionNode Attr, Record R) {
(1)  Attr이 단말노드일 경우, return (Attr.Decision) ;
(2)  Child = R.Attr과 같은 레이블을 갖는 가지에 연결된 노드
(3)  Answer = AssignClass(Child, R) ;
(4)  R.Attr의 일반화인 레이블을 갖는 가지에 대하여
          {
(5)       Child = 현재 가지에 연결된 노드 ;
(6)       Temp = AssignClass(Child, R) ;
(7)       If Temp.Confidence > Answer.Confidence,
                then Answer = Temp ; }
(8)  R.Attr의 세분화인 레이블을 갖는 가지에 대하여
          {
(9)       Child = 현재 가지에 연결된 노드 ;
(10)      Weight = 모든 세분화값을 갖는 레코드 중 현재 세분화
                값을 갖는 레코드의 비율 ;
(11)      Temp = Weight × AssignClass(Child, R) ;
(12)      If Temp.Confidence > Answer.Confidence,
                then Answer = Temp ; }
(13) return(Answer) ; }
```

(그림 5) 다중 추상화 수준을 갖는 데이터를 위한 클래스 할당 알고리즘

기존 할당 과정은 (1)부터 (3)까지이며, 데이터의 일반화/

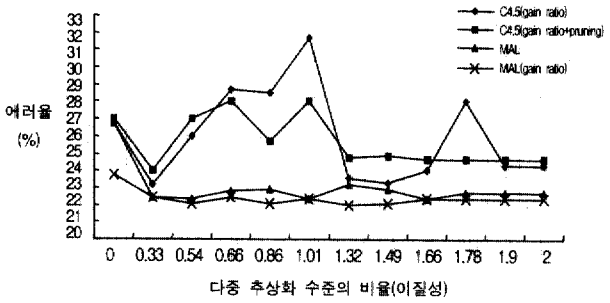
세분화 관련성을 반영하여 클래스를 할당하기 위해 (4)에서 (7)까지와 (8)에서 (12)까지를 첨가하였다. 결과적으로 아직 분류되지 않은 레코드는 최고 신뢰도 값을 갖는 클래스에 최종적으로 할당된다.

4. 성능 평가

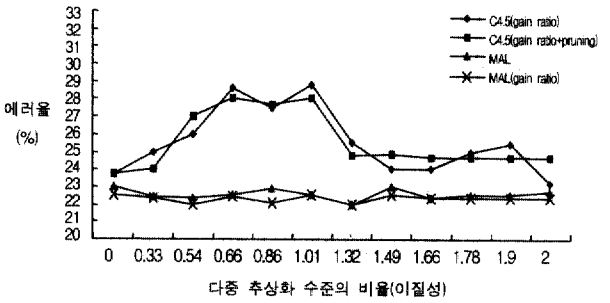
본 절에서는 제안한 방법의 분류 정확성을 분석하기 위해 벤치마크 데이터 집합으로 실험한 결과를 제시한다. 실험 결과는 데이터가 다중 추상화 수준으로 표현되어 있을 때 제안한 방법을 이용할 경우 분류 에러율이 현저하게 감소함을 보인다. 실험은 Solaris 7 운영체제와 128MB RAM, 10GB 하드 디스크를 장착한 Axil-Ultima 167 워크스테이션에서 수행하였다. 제안한 방법이 기존 방법인 C4.5보다 분류 정확성이 높다는 것을 보이기 위해 다중 추상화 수준으로 표현된 학습 데이터가 필요하다. 그러나 결정 트리 분류에서 데이터의 다중 추상화 수준을 고려한 연구가 거의 이루어지지 않았기 때문에 공개된 정보원으로부터 그러한 데이터 집합을 얻는 것은 쉽지 않다. 그래서 UCI Machine Learning Repository에 공개된 census income database에서 각각 다른 년도의 벤치마크 데이터 집합을 구하고, 데이터 집합이 다양한 추상화 수준으로 표현되도록 추상화 수준에 대한 분포를 임의로 조정하였다. 각 데이터 집합은 편의상 데이터 집합 1과 데이터집합 2로 표기한다. 원래의 벤치마크 데이터 집합은 6개의 연속형 속성과 8개의 명목형 속성을 갖으며, 데이터집합 1과 데이터집합 2 모두 4만개 이상의 레코드를 포함한다. 또한 연간수입 $50,000를 기준으로 하여 고소득 거주자 또는 저소득 거주자를 분류하기 위한 인구조사 정보를 갖는다. 벤치마크 데이터 집합은 각 속성의 모든 값들이 같은 추상화 수준으로 표현되어 있기 때문에 실제 환경에서 발생하는 상황을 모사하기 위해 랜덤 함수를 사용해 추상화 수준을 조정하였다. 예를 들어, 속성 'workclass'의 속성 값인 'Federal-gov'를 갖는 레코드 중 일부를 일반화 값인 'gov'로 치환하였다. 이와 같이 치환되는 데이터의 비율을 조절함으로써 데이터가 다양한 추상화 수준으로 표현될 수 있도록 생성하였으며 이러한 다양한 추상화 수준을 이질성(heterogeneity)으로 표현한다. 이질성은 정보 이론에서의 앤트로피를 이용하였다. 데이터 집합의 이질성이 0일 때, 모든 데이터는 같은 추상화 수준으로 표현되어 있으며, 이질성이 높아질수록 데이터의 추상화 수준은 다양한 추상화 수준으로 표현되어 있다고 할 수 있다.

실험은 분할 정보(split info)를 이용할 때의 C4.5, 분할 정보를 이용하여 가지치기를 수행했을 경우의 C4.5, 제안한 알고리즘에 획득값을 이용했을 경우, 제안한 알고리즘에 분할 정보를 이용했을 경우의 분류 에러율을 각각 비교 분석하였다.

(그림 6)(a)와 (그림 6)(b)는 데이터 집합의 이질성에 따른 분류 에러율을 실험한 결과이다. 각 실험결과에서 제안한 알고리즘은 표기상 'MAL'로 표현한다. 데이터 집합의 이질성이 0일때 제안한 방법의 분류 에러율은 C4.5 방법을 이용하여 분류했을 때와 같으며, 이질성이 증가할수록 제안한 방법의 성능은 70%까지 향상되었다.



(a) 데이터집합 1



(b) 데이터집합 2

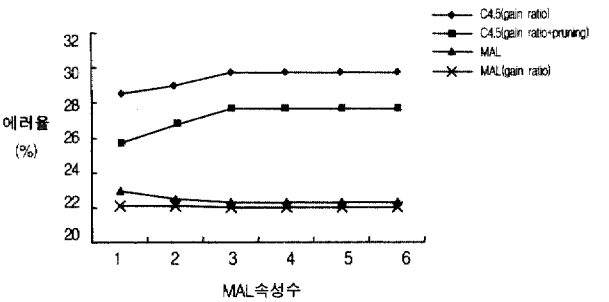
(그림 6) 추상화 수준의 이질성에 따른 에러율

(그림 6)에서 제시한 바와 같이 C4.5에서는 분류에러율과 MAL 데이터의 비율은 서로 독립적이며, 서로 관련성이 없으므로 그래프에서 나타나는 증감 현상은 특별히 어떤 경향을 보이지 않는다. MAL방법에서는 이질성이 1.0인 경우는 속성값 'gov'가 70%정도, 'Federal-gov', 'State-gov', 'Local-gov'도 각각 10%정도 존재한다. 이러한 경우 구축된 결정 트리에 결정 노드로 다중 추상화 수준을 갖는 속성이 포함된다. 그러므로 제안한 방법을 이용하여 결정트리를 구축할 경우 추상화 수준을 고려하여 'gov'와 'Federal-gov', 'State-gov', 'Local-gov'의 관련성을 반영하므로 분류 에러율이 아주 낮아 C4.5와의 차이가 아주 큼을 알 수 있다. 그러나 데이터 집합의 이질성이 1.0보다 클 경우 분류 정확성은 감소된다. 이러한 현상은 데이터의 속성이 너무 다중 추상화 수준으로 표현될 경우 그 속성이 최적 분할 속성으로 선택될 수 없기 때문에 발생한다. 즉, 구축된 결정 트리가 결정 노드로 그러한 속성을 결정노드에 아예 포함하지 않기 때문이다. 결과적으로 데이터 집합의 이질성이 과다하게 높은 경우에는 제안한 방법이 성능 향상에 별로 영향을 미치지 않는다는 것을 알 수 있다.

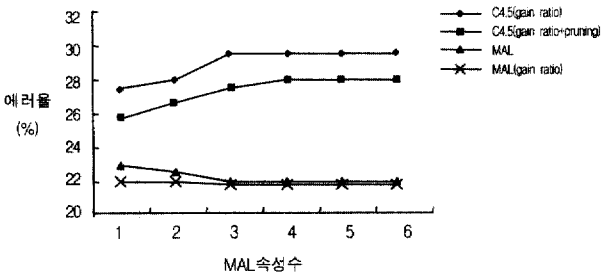
다음은 다중 추상화 수준을 갖는 속성의 수가 증가함에 따른 분류 정확성을 분석하였다. 이 실험을 위한 데이터 집

합은 UCI 데이터 집합의 속성 'workclass'외에 다른 속성에 대하여 다양한 추상화 수준을 갖도록 속성 값을 첨가하여 다중 추상화 수준을 갖는 속성의 수를 증가하였다. 예를 들어 속성 'education'의 속성 값인 '10th', '11th', '12th'가 'high school'을 나타내므로 'HS'를 이들 값에 대한 일반화 값으로 생성하여 첨가하였다.

이 실험을 위한 데이터 집합은 원래의 UCI 데이터 집합에 대하여 다중 추상화 수준으로 표현되는 속성의 수를 증가하면서 각각 다르게 생성하였다. (그림 7)(a)와 (그림 7)(b)에서 다중 추상화 수준을 갖는 속성의 수가 증가함에 따라 제안한 방법의 분류 에러율이 감소됨을 알 수 있다. 또한 다중 추상화 수준을 갖는 속성의 수가 증가함에 따라 C4.5 방법과 제안한 방법의 에러율의 차이는 커짐을 알 수 있다.



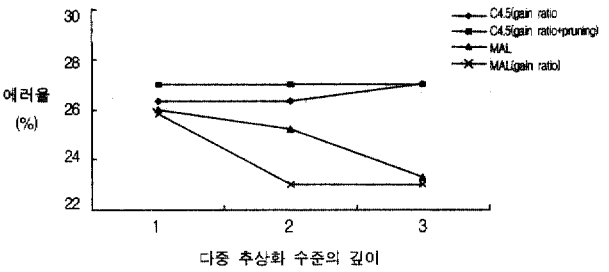
(a) 데이터집합 1



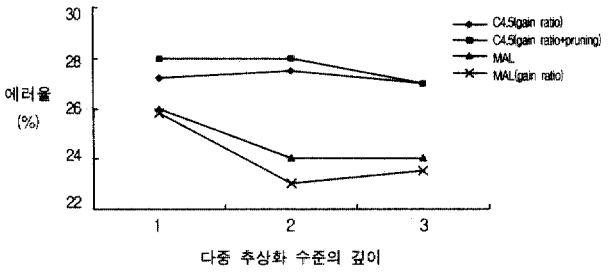
(b) 데이터집합 2

(그림 7) 데이터의 다중 추상화 수준을 갖는 속성의 수에 대한 에러율

(그림 8)(a)와 (그림 8)(b)는 어떤 속성의 다중 추상화 수준의 깊이가 증가했을 경우의 분류 정확성을 비교 한 결과이다. 이 실험을 위한 데이터 집합은 모두 같은 이질성을 갖는 분포로 되어 있으며, 앞의 실험에 사용한 다양한 추상화 수준을 갖는 속성에 대하여 보다 더 세분화된 값을 첨가하여 다양한 추상화 수준을 갖는 속성의 깊이만을 증가하였다. 예를 들어 'State-gov'의 세분화 값으로 'East-State-gov'와 'West-State-gov'를 새로 첨가하여 생성하였으며, 추상화 수준의 깊이를 증가하기 위해 'East-State-gov'의 세분화 값으로 'Far-East-State-gov', 'Mid-East-State-gov'의 값을 첨가하여 생성하였다. 결과적으로 다중 추상화 수준을 갖는 속성의 깊이가 증가함에 따라 C4.5와 제안한 알고리즘의 분류 에러율의 차이는 커짐을 알 수 있다.



(a) 데이터집합 1



(b) 데이터집합 2

(그림 8) 다중 추상화 수준을 갖는 속성 깊이 증가에 대한 에러율

본 논문에서 제안한 알고리즘은 결정 트리를 구축할 때 가장 중요한 부분인 최적 분할 선택 부분과 학습 데이터 분할 부분에서 기존의 방법을 확장한 것으로 다른 알고리즘에서도 기본적으로 처리하는 과정이다. 최근 제안된 결정 트리 알고리즘의 수행 속도를 빠르게 하는 방법들은 본 논문에서 제안한 알고리즘에도 적용이 가능하다. 다음에 제시한 결정 트리 구축 및 실험 데이터 분류 시간에 대한 분석 결과는 제안한 알고리즘과 C4.5의 수행 시간이 거의 차이가 없다는 것을 보이며, SLIQ[14], SPRINT[15], RainForest[1], BOAT[2]에서 성능을 향상시키는 부분은 본 논문에서 제안한 알고리즘에도 적용 가능하다고 말할 수 있다. 예를 들어, SLIQ는 순서형 속성을 사전 정렬하도록 함으로써 결정 트리 구축의 속도를 빠르게 하며, 대용량의 데이터 집합을 위해 메모리와 디스크에 상주하는 데이터 구조를 사용한다. SPRINT는 병렬 수행을 지원하며, RainForest는 기존의 결정 트리 알고리즘을 확장하기 위한 프레임워크(Framework)로서 데이터베이스 크기에 따라 확장되는 특징을 갖는다. 또한 BOAT는 학습 데이터가 동적으로 변화할 때 트리를 재구축하지 않고 처음 구축한 트리를 갱신한다.

결정 트리를 구축하는 시간 복잡도와 구축된 결정 트리에 의해 실험 데이터를 분류하는 시간 복잡도에 대하여 C4.5와 제안한 알고리즘를 비교하여 점근적 표기법으로 제시하였다. 결정 트리 구축에 대한 시간 복잡도는 m을 속성의 수라 하고 n을 레코드 수라 가정하였을 때, 기존의 알고리즘인 경우 $O(m^2n)$이다. 또한 제안한 알고리즘의 시간 복잡도도 $O(m^2n)$인데, 이는 다중 추상화 수준을 갖는 속성에 대한 트리의 수와 깊이로 인하여 상수인자가 다소 크지만 점근적 표기법을 사용했을 때 기존의 알고리즘과 동일한 연

산 시간을 갖는다. 그리고 실험 데이터에 대한 분류 시간은 구축된 결정 트리의 깊이를 h라 할때, 기존의 알고리즘인 경우 O(h)이며 제안한 알고리즘의 시간 복잡도는 다중 추상화 수준으로 표현된 데이터 레코드가 결정 트리의 모든 노드에서 고려되지 않을 수 있기 때문에 연산 시간이 결정 트리의 깊이에 비례하여 O(h)로 기존의 알고리즘과 동일하다.

5. 결 론

본 논문에서는 서로 다른 추상화 수준으로 표현되는 데이터를 기존의 결정 트리 방법을 이용하여 분류할 때 모순이 일어날 수 있음을 보이고, 그에 대한 해결 방안을 제시하였다. 또한 데이터 정화 도구를 이용하여 강제로 추상화 수준을 맞추는 것이 문제를 해결할 수 없다는 것을 설명하였다. 본 논문에서 제안한 방법은 현재 존재하는 정보를 그대로 이용하며, 결정 트리를 구축하는 단계와 구축된 결정 트리를 이용하여 분류되지 않은 레코드의 클래스를 할당하는 단계에서 데이터 값들 사이의 일반화/세분화 관련성을 반영한다. 세분화 값을 갖는 레코드는 세분화 값이 일반화 값에 포함되므로 그 값의 일반화 값을 갖는 레코드와 같이 분류된다. 반면, 일반화 값을 갖는 레코드는 학습 데이터에 존재하는 모든 세분화 값들에 대한 특정 세분화 값들의 분포를 반영하는 소속 정도로 세분화 값을 갖는 레코드와 같이 분류된다. 이러한 부분적 소속 정도를 나타내기 위해 퍼지 릴레이션의 개념을 도입하였다. 제안한 방법은 기존의 퍼지 결정 트리 구축방법에서 결정 트리를 구축하기 전 사용자가 정의한 퍼지 집합을 기준으로 데이터 집합을 퍼지화해야 하는 부가적인 작업이 필요없다. 또한, 소속정도는 본 논문에서 제안한 데이터 추상화 수준을 나타내는 ISA 계층 구조와 전체 데이터 분포를 이용하여 자동으로 계산할 수 있으므로 데이터 집합을 따로 퍼지화할 필요가 없다. 분류되지 않은 레코드에 클래스를 할당하는 과정에서 기존의 퍼지 결정 트리 방법에서는 모든 노드의 값을 다 고려하여 클래스를 결정하지만, 제안한 방법에서는 다중 추상화 수준을 갖는 속성을 갖는 노드만 고려하여 클래스를 결정하기 때문에 분류과정이 단순하다. 성능 평가는 UCI Machine Learning Repository의 벤치마크 데이터 집합을 다중 추상화 수준을 갖도록 수정하여 이루어 졌다. 그 결과 데이터가 서로 다른 추상화 수준으로 표현될 때 기존의 방법보다 분류 에러율이 현저히 줄었음을 보였다. 본 논문에서 제안한 방법은 지역 네트워크 관리 에이전트로부터 서로 다른 추상화 수준으로 표현되어 있는 알람을 받아 처리하는 통합 네트워크 알람 관리 시스템에 적용하고자 한다.

참 고 문 헌

[1] J. Gehrke, R. Ramakrishinan and V. Ganti, "RainForest-A Framework for Fast Decision Tree Construction of Large Datasets," *Data Mining and Knowledge Discovery*, Vol.4, pp.127-162, 2000.

[2] J. Gehrke, V. Ganti, R. Ramakrishnan and W. Loh, "BOAT Optimistic Decision Tree Construction," *In Proc. of ACM SIGMOD Conf.*, Philadelphia, Pennsylvania, pp.169-180, June, 1999.

[3] M. Berry and G. Linoff, *Data Mining Techniques For Marketing, Sales, and Customer Support*, Wiley and Sons, 1997.

[4] J. Quinlan, *C4.5 : Programs for Machine Learning*, Morgan Kaufmann Pub., 1993.

[5] M. Mehta, R. Agrawal and J. Rissanen, "SLIQ : A Fast Scalable Classifier for Data Mining," *Proc. of the Fifth Int'l Conference on Extending Database Technulogy (EDBT)*, Avignon, France, March, 1996.

[6] J. Shafer, R. Agrawal, M. Mehta, "SPRINT : A Scalable Parallel Classifier for Data Mining," *Proc. of the 22th Int'l Conference on Very Large Databases*, Mumbai (Bombay), India, September, 1996.

[7] K. Hatonen, M. Klemettinen, H. Mannila, P. Ronkainen and H. Toivonen, "Knowledge Discovery from Telecommunication Network Alarm Databases," *In Proc. of the 12th International Conference on Data Engineering*, New Orleans, Louisiana, pp.115-122, February, 1996.

[8] L. English, *Improving Data Warehouse and Business Information Quality-Method for Reducing Costs and Increasing Profits*, Wiley & Sons, 1999.

[9] R. Wang, V. Storey and C. Firth, "A Framework for Analysis of Data Quality Research," *IEEE Transactions on Knowledge and Engineering*, Vol.7, No.4, pp.623-640, August, 1995.

[10] Trillium Software System, "A Practical Guide to Achieving Enterprise Data Quality," White Paper, Trillium Software, 1998.

[11] J. Williams, *Tools for Traveling Data, DBMS*, Miller Freeman Inc., June, 1997.

[12] Vality Technology Inc., "The Five Legacy Data Contaminants You Will Encounter in Your Warehouse Migration," White Paper, Vality Technology Inc., 1998.

[13] G. Klir and T. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice-Hall Int'l Inc., 1988.

[14] C. Shannon, "The Mathematical Theory of Communication," The Bell System Tech., 1948.

[15] C. Batini, S. Ceri and Navathe, *Conceptual Database Design*, Benjamin Cummings, Inc., 1992.

[16] X. Wang and H. Jiarong, "On the handling of fuzziness for continuous-valued attributes in decision tree generation," *Fuzzy Sets and Systems 99*, pp.283-290, 1998.

[17] C. Janikow, "Fuzzy decision trees : issues and methods,"

IEEE Transactions on, Systems, Man and Cybernetics, Part B, Vol.28, Issue.1, pp.1-14, February, 1998.

[18] M Dong, R. Kothari, "Look-ahead based fuzzy decision tree induction," *IEEE Transactions on Fuzzy Systems*, Vol.9, Issue.3, pp.461-468, June, 2001.

정 민 아

e-mail : majung@kjist.ac.kr

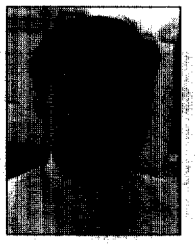
1992년 전남대학교 전산통계학과(학사)

1994년 전남대학교 대학원 전산통계학과 (이학석사)

2002년 전남대학교 대학원 전산통계학과 (이학박사)

2002년~현재 광주과학기술원 정보통신학과 Post-Doc.

관심분야 : 데이터마이닝, 데이터베이스, 정보보호



이 도 헌

e-mail : dhlee@mail.kaist.ac.kr

1990년 한국과학기술원 전산학과(학사)

1992년 한국과학기술원 전산학과 (공학석사)

1995년 한국과학기술원 전산학과 (공학박사)

1996년~2002년 전남대학교 전산학과 및 의학과 조교수

1999년~2000년 미국 Univ. of Texas at Austin, 방문교수

2001년~현재 ACM Transactions on Internet Technology, Associate Editor

2001년~현재 한국데이터마이닝학회 이사

2002년 미국 National Institute of Health, 방문연구원

2002년~현재 한국과학기술원 바이오시스템학과 부교수

관심분야 : 바이오정보학, 데이터마이닝, 데이터베이스