전역적 범주화를 위한 샘플 분할 포인트를 이용한 점진적 기법

(An Incremental Method Using Sample Split Points for Global Discretization)

한 경 식 † 이 수 원 ††
(Kyong Sik Han) (Soo Won Lee)

요 약 대부분의 교사학습 알고리즘은 수치형 변수 처리의 어려움을 해결하기 위해 전처리 단계에서 연속형 변수를 범주형으로 변환시킨 후 적용된다. 이러한 전처리 단계를 전역적 범주화라 하며 빈즈(Bins)라는 클래스 분포 리스트를 이용한다. 그러나 대부분의 전역적 범주화 기법은 단일 빈즈를 필요로 하기 때문에 데이타가 대용량이고 범주화를 수행할 변수의 범위가 매우 클 경우, 단일 빈즈를 생성하기 위해 많은 정렬 및 병합을 수행해야한다. 또한, 기존의 방법은 일괄처리 방식으로 범주화를 수행하기 때문에 새로운 데이타가 추가되면 이 데이타가 반영된 범주를 생성하기 위해 처음부터 범주화를 다시 수행해야한다. 본 논문은 이러한 문제점을 해결하기 위해 샘플 분할 포인트를 추출하고 이로부터 범주화를 수행하는 기법을 제안한다. 본 논문의 접근 방법은 단일 빈즈를 생성하기 위한 병합이 필요 없기 때문에 대용량 데이타에 대한 범주화를 수행할 때 효율적이다. 본 연구에서는 실제 데이타와 가상의 데이타를 이용하여 기존의 방법과 비교 실험하였다.

키워드 : 전역적 범주화, 기계학습, 순차적 학습, 대용량 데이타셋, 데이타 마이닝

Abstract Most of supervised learning algorithms could be applied after that continuous variables are transformed to categorical ones at the preprocessing stage in order to avoid the difficulty of processing continuous variables. This preprocessing stage is called global discretization, uses the class distribution list called bins. But, when data are large and the range of the variable to be discretized is very large, many sorting and merging should be performed to produce a single bin because most of global discretization methods need a single bin. Also, if new data are added, they have to perform discretization from scratch to construct categories influenced by the data because the existing methods perform discretization in batch mode. This paper proposes a method that extracts sample points and performs discretization from these sample points in order to solve these problems. Because the approach in this paper does not require merging for producing a single bin, it is efficient when large data are needed to be discretized. In this study, an experiment using real and synthetic datasets was made to compare the proposed method with an existing one.

Key words : Global Discretization, Machine Learning, Incremental Learning, Large Dataset, Data Mining

1. 서 론

결정 트리(Decision Tree)[1,2] 및 베이시안 네트웍(Bayesian Networks)[3]등 많은 교사학습(Supervised Machine Learning) 알고리즘은 오직 범주형 변수(Categorical Variable)로만 구성된 데이타에 기초하여 연구되어 왔다. 그러나 실세계 데이타는 범주형 변수뿐만 아니라 연속형 변수(Continuous Variable)도 포함하고있기 때문에 이를 적절한 범주화 기법(Discretization Method)을 이용하여 연속형 변수를 범주형으로 변환시킨 후 기계학습에 이용한다. 범주화 기법은 크게 전역적 범주화 기법(Global Discretization Method)과 지역적 범주화 기법(Local Discretization Method)으로 분류될

· 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음
† 비 회 원 : (주)인우기술 선임연구원
hanksjang@empal.com
†† 종신회원 : 숭실대학교 컴퓨터학부 교수
swlee@computing.ssu.ac.kr
논문접수 : 2003년 7월 21일
심사완료 : 2004년 4월 30일

수 있다[4]. 전역적 범주화 기법은 기계학습에 이용하기 이전에 전체 데이타에 대해 수행하며 다중 분할 포인트(Multi-Split Points)를 생성한다. 이와 달리, 지역적 범주화 기법은 C4.5의 트리 생성 과정에서와 같이 각 노드에서 그 노드에 도달한 데이타만을 이용하여 부분적으로 범주화를 수행하며 단일 분할 포인트(Single Split Point)를 생성한다. 이중 전역적 범주화 기법은 전처리 기법으로서 기계학습 기법에 관계없이 이용할 수 있기 때문에 활용범위가 매우 넓으며 이를 결정 트리 생성에 이용할 경우, 분류의 정확도 및 트리 생성 속도에서 지역적 범주화보다 더 좋은 성능을 보이고 있다[4].

전역적 범주화기법의 대부분의 알고리즘은 데이타 자체에 범주화를 수행하는 지역적 범주화와는 달리 각 연속형 변수에 대해 정렬된 클래스 분포 리스트를 생성하고 이를 이용하여 범주화를 수행한다. 이 정렬된 클래스 분포 리스트를 빈즈(Bins)라 부른다. 범주화 기법은 빈즈에 대해 평균 클래스 엔트로피와 같은 평가함수를 적용하여 최적의 분할 포인트를 선택하여 이 분할 포인트를 이용하여 전체 구간을 두 개의 부분 구간으로 분할한다. 이와 같은 방식은 생성된 각 부분 구간에 적용되고 특정 정지 조건이 만족될 때까지 각 구간은 계속 분할되어 최적의 다중 분할 포인트를 생성한다[5,6].

그러나 이러한 방식은 단일 빈즈를 필요로 하기 때문에 데이타가 대용량이고 범주화를 수행할 변수의 범위가 매우 클 경우, 단일 빈즈를 생성하기 위해 많은 정렬 및 병합(Merge)을 수행해야한다는 문제점을 안고 있다. 즉, 이러한 기법은 범주화를 수행할 변수의 범위가 매우 클 경우, 데이타를 메모리 한계까지 읽어들인 후 정렬을 수행하고 단일 빈즈를 생성하기 위해 기존의 빈즈와 병합을 수행해야 한다. 특히, 단일 빈즈가 너무 커서 빈즈 자체를 메모리에 적재하지 못하는 경우, 디스크 상에서 병합 및 범주화를 수행해야하며 이는 매우 비효율적이다. 두 번째 문제점은 이러한 방식은 모두 일괄처리 방식으로 범주화를 수행하기 때문에 새로운 데이타가 추가되면 이 데이타가 반영된 범주를 생성하기 위해 처음부터 다시 범주화를 수행해야 한다. 이는 또 다시 많은 시간을 필요로 하며 새로운 데이타가 계속적으로 추가되는 환경에는 적합하지 못하다.

본 논문은 이러한 문제점을 해결하기 위해 샘플 분할 포인트(Sample Split Points)를 추출하고 이로부터 범주화를 수행하는 기법을 제안한다. 샘플 포인트는 특정 후보 분할 포인트의 해당 빈즈에 대한 인덱스 및 클래스 분포를 유지하며 이 정보는 구간에 대한 평가함수의 하위 경계값을 예측하기 위해 이용된다. 본 논문의 접근 방법은 이러한 구간에 대한 하위 경계값을 이용하여 탐색공간을 축소해가면서 최종 분할 포인트를 결정한다. 새로운 데이타가 추가되면, 이 데이타에 대한 빈즈가 생성되고 해당 샘플 포인트의 인덱스 및 클래스 분포는 이를 포함하기 위해 갱신된다. 이 샘플 포인트들은 이전 빈즈와 새로운 빈즈에 대한 분포 정보를 모두 포함하고 있으며 이전의 하나의 빈즈만 있을 때와 같은 방식으로 이용된다. 본 논문의 이러한 접근 방법은 범주화를 위해 단일 빈즈를 생성할 필요가 없기 때문에 빈즈에 대한 병합이 많이 발생하는 대용량 데이타나 데이타가 계속적으로 추가되는 상황에 특히 효율적인 방법이다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 전역적 범주화 기법 및 지역적 범주화 기법을 설명하고 이들의 문제점을 설명한다. 3장에서는 본 논문에서 제안하는 샘플 분할 포인트를 이용한 범주화 기법을 설명하고 이를 순차적으로 범주화를 수행할 수 있도록 확장한다. 4장에서는 본 접근 방법에 대해 실험하고 이를 분석한다. 마지막으로 5장은 결론 및 향후 연구를 제시한다.

2. 관련 연구

전역적 범주화의 대표적인 기법은 불순도 측정(Impurity Measure)에 기반 한 범주화 기법이다. 불순도 기반 범주화 기법은 Gini Index[1], Quinlan[2]의 Information Gain, Gain Ratio 등을 이용한 기법을 들 수 있다. 그 중 Information Gain을 이용한 기법은 분할 포인트를 선택하기 위해 후보 분할 포인트(Candidate Split Point)들의 평균 클래스 엔트로피(Average Class Entropy)를 이용하며 정의는 그림 1과 같다. IG(A,T;S)와 E(A,T;S)는 각각 Information Gain 함수 및 평균 클래스 엔트로피 함수이며 S는 예제 집합, A는 연속형 변수, T는 후보 분할 포인트, S_1 및 S_2는 S를 T로 분할할 경우의 부분 구간을 나타낸다. H(S)는 엔트로피 함수이며, k는 클래스 개수, $P(C_i, S)$는 S의 예제 중 클래스 C_i에 속할 확률을 나타낸다.

$$IG(A,T;S) = H(S) - E(A,T;S)$$

$$E(A,T;S) = \frac{|S_1|}{|S|}H(S_1) + \frac{|S_2|}{|S|}H(S_2)$$

$$H(S) = -\sum_{i=1}^{k} P(C_i,S)\log P(C_i,S)$$

그림 1 Information Gain 함수

Fayyad & Irani는 다중 분할 포인트를 생성하기 위해 특정 연속형 변수 A에 대해서 모든 가능한 분할 포인트 중 평균 클래스 엔트로피를 가장 최소화하는 T_{min}을 분할 포인트로 선택하여 구간을 분할하고 이와 같은 방법을 두 분할 구간에 재귀적으로 적용하여 특정 조건이 만족될 때까지 구간을 이진 분할하는 이진화 기법

(Binarization Method)을 이용하였다[5,6]. 특히, 가능한 후보 분할 포인트 수를 줄이기 위해, 즉 평가함수 적용 회수를 줄이기 위해 정의 1 및 정리 1의 경계 포인트(Boundary Point)를 이용하였으며 정리 1을 증명하기 위해 연속적인 두 경계 포인트 사이에의 평가 함수의 값이 볼록한 특성을 이용하였으며 이러한 특성을 갖는 함수를 볼록 함수(Convex Function)라 한다.

정의 1. 특정 연속형 변수 A에 대해 정렬된 예제 집합 S에서 경계 포인트 T의 정의는 다음과 같으며 $val_A(u)$는 예제 u의 연속형 변수 A의 값을 나타낸다.

1. 클래스가 서로 다른 예제 u, v ∈ S가 존재하며
2. $val_A(u) = T < val_A(v)$이며
3. $val_A(u) < val_A(w) < val_A(v)$를 만족하는 w가 존재하지 않는다.

정리 1. 특정 분할 포인트 T가 평균 클래스 엔트로피를 최소화하는 분할 포인트이면 T는 경계 포인트이다.

그림 2는 클래스 x, y를 갖는 예제를 정렬한 후 생성한 빈즈와 블록(Blocks)을 나타내며 블록의 경계가 경계 포인트가 된다. 그림 2에서는 평가함수의 볼록한 특성을 이용함으로써 27개의 후보 분할 포인트에서 5개의 후보 분할 포인트로 줄일 수 있음을 나타낸다.

Elomaa는 이러한 경계 포인트 정의를 더 확장하여 세그먼트(Segments)라는 개념을 정의하였다. 그림 2(d)는 세그먼트를 나타내며 세그먼트는 상대적 클래스 분포가 같은 서로 인접해 있는 블록의 집합으로 정의되며 블록은 세그먼트의 특별한 형태이다. 또한, Elomaa는 볼록 함수 이외에 Gain Ratio나 Normalized Distance Measure 와 같은 비볼록 함수(Non-Convex Function)에 대해서도 세그먼트를 이용할 수 있음을 증명하였다[7,8].

그러나 이러한 기법은 범위가 매우 클 경우, 중복되어 나오는 경우가 거의 없기 때문에 빈즈의 개수는 실제 값의 개수와 거의 같고 역시, 블록 및 세그먼트의 효과도 거의 얻을 수가 없다는 문제점을 안고 있다.

불순도 측정에 기초한 범주화 기법은 전역적 범주화뿐만 아니라 지역적 범주화에서도 가장 많이 쓰이는 방법이다. ID3[2], C4.5[2] 및 Cart[1] 등 대부분의 결정 트리 알고리즘이 이에 속하며 현재 노드에 도달한 예제 집합을 범주화를 수행할 변수의 값으로 정렬한 후 후보 분할 포인트에 평가함수를 적용하여 최적의 분할 포인트를 선택한다. 지역적 범주화 기법은 이진 분할로서 해당 노드에 오직 하나의 분할 포인트만 결정한다. 최근 들어, 대용량 데이타 처리를 위한 접근 방법에 관한 연구에서 이런 보편적인 방법과는 달리 샘플 분할 포인트를 추출한 후 이를 통해 분할 포인트를 결정하는 알고리즘이 제시되고 있다. 이러한 접근 방법은 두 개의 샘플 분할 포인트로 정의되는 구간에 대한 평가함수 값의 하위 경계값(Lower-Bounded Value) 예측에 기반을 두고 있으며 대표적인 알고리즘으로 CLOUD[9]와 BOAT[10] 등을 들 수 있다.

이러한 알고리즘의 기본 아이디어는 다음과 같다. 우선 가능한 분할 포인트로부터 샘플 후보 포인트 x_1, x_2,..,x_m를 추출한다. 이들에 대해 평가 함수를 적용하여 그 중 최소값을 imp_{min}라 한다. 이제 남은 것은 imp_{min}이 샘플 포인트에서뿐만 아니라 모든 구간[x_i, x_j]에서도 최소값 즉, 최선임을 증명하는 것으로 BOAT의 경우, 평가함수의 볼록 특성을 이용한 다음의 정리를 이용한다.

정리 2. 특정 연속형 변수 X의 값 x 및 k개의 값을 갖는 클래스 변수 C에 대해, t.X는 예제 t의 변수 X의 값일 때, x의 클래스 i에 대한 통계치 n^i_x를 $n^i_x = |\{t : t.X \le x \wedge t.C=i\}|$이라 하며, $x_1 < x_2$를 만족하는 X의 값 x_1, x_2에 대해 k개의 클래스에 대한 통계치 벡터를 각각 $(n^1_{x1},...,n^k_{x1})$, $(n^1_{x2},...,n^2_{x2})$라 한다. $P_{x1,x2}$를 x_1과 x_2 사이에 발생할 수 있는 모든 통계치 벡터들의 집합이라 하고 S를 $(n^1_{x1},...,n^k_{x1})$, $(n^1_{x2},...,n^2_{x2})$로 유추할 수 있는

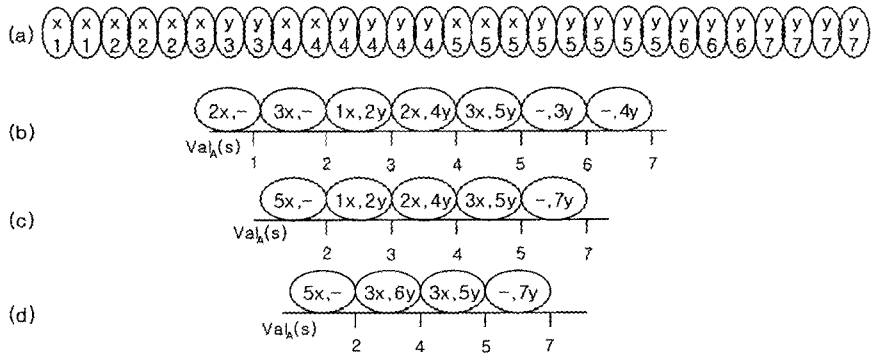

그림 2 (b) 빈즈(Bins) (c) 블록(Blocks) (d) 세그먼트(Segments)

2^k개의 가상의 통계치 벡터라 정의한다($S=\{(n^1_{x1},...,n^k_{x1}), (n^1_{x2},n^2_{x1}...,n^k_{x1}),...,(n^1_{x2},...,n^{k-1}_{x2},n^k_{x1}),(n^1_{x2},...,n^k_{x2})\}$). imp를 평가 함수라 하면 S와 $P_{x1,\ x2}$는 다음을 만족한다.

$$\min_{(n_1,...,n_k)\in S} \mathrm{imp}(n_1,...,n_k) \le \min_{(n_1,...,n_k)\in P_{x1,x2}} \mathrm{imp}(n_1,...n_k)$$

k=2일 때 x_1의 클래스 분포가 (n^1_{x1}, n^2_{x1})이고 x_2의 클래스 분포가 (n^1_{x2}, n^2_{x2})일 때 S는 이로부터 추출할 수 있는 가상의 통계치 벡터 집합으로서 $\{(n^1_{x1}, n^2_{x1}), (n^1_{x2}, n^2_{x2}), (n^1_{x1}, n^2_{x2}), (n^1_{x2}, n^2_{x1})\}$을 나타낸다.

이러한 기법은 imp_{min}이 최소값임을 증명하기 위해 모든 샘플 분할 포인트 구간 $[x_i, x_j]$에 대해 정리 2를 이용하여 2^k개의 통계치 벡터를 추출하고 이에 대해 평가함수를 적용한 후 최소값, imp^{est}를 그 구간에 대한 하위 경계값이라 한다. 구간의 imp^{est}가 imp_{min}보다 크면 이 구간은 후보 분할 구간에서 제외되고 그렇지 않으면 그 구간에 imp_{min}보다 더 낮은 평가 함수 값을 갖는 분할 포인트가 존재할 수 있기 때문에 최적의 분할 포인트 선택을 위한 검증이 이루어진다. CLOUD의 경우, 후보 분할 구간에 대한 검증은 그 구간에 속하는 모든 후보 분할 포인트를 추출하여 이를 직접 평가함으로써 이루어진다.

3. 다중 분포 리스트를 이용한 순차적 범주화

2장에서 설명한 CLOUD 및 BOAT 등의 범주화 기법은 지역적 범주화 기법으로 최적의 분할 포인트 선택을 위해 해당 노드에 도달한 예제 집합 자체를 이용하기 때문에 일반적으로 그림 2와 같은 빈즈를 이용하는 전역적 범주화 기법과 다르다. 또한, CLOUD의 경우, $imp_{min} > imp^{est}$ 면, 최적의 분할 포인트를 찾기 위해 그 구간의 모든 후보 분할 포인트을 추출하고 이를 직접 평가하기 때문에 세부적으로 검증을 필요로 하는 후보 분할 구간이 많을 경우 비효율적이며 BOAT의 경우, 특정 노드에 대한 분할 포인트를 구하기 위해 부트스트래핑(Bootstrapping) 기법을 이용하기 때문에 다중 분할 포인트를 생성하는 전역적 범주화 기법에는 적합하지 못하다.

본 논문에서 제안하는 방법은 전역적 범주화에서 두 샘플 분할 포인트 x_i, x_j로 정의되는 구간 $[x_i, x_j]$에 대해 정리 2를 이용하여 하위 경계값을 예측함으로써 다중 분할 포인트를 생성하는 접근 방법이다. 그러나 CLOUD와 달리 $imp_{min} > imp^{est}$ 경우 그 구간에 속하는 모든 후보 분할 포인트를 직접 평가하지 않고 그 구간에 대한 샘플 포인트를 추출하여 더 세부 구간으로 분할함으로써 검증을 수행한다. 또한 단일 빈즈에서 범주화를 수행하는 일반적인 전역적 범주화 기법과 달리 다중 빈즈에서 범주화를 수행함으로써 순차적인 범주화도 가능하도록 한다. 본 논문의 접근 방법은 불순도 측정에 기초한 모든 평가함수에 이용될 수 있으며 본 논문에서는 특정 분할 포인트로 예제를 분할할 경우의 예상 엔트로피 감소량을 나타내는 Information Gain(정보증가)을 평가함수로 사용하였다.

3.1 샘플 포인트를 이용한 전역적 범주화

일반적으로 전역적 범주화는 그림 2와 같은 빈즈에서 수행된다. 최적의 분할 포인트를 선택하기 위해 왼쪽에서 오른쪽으로 빈즈를 읽어 가면서 각 분할 포인트에 대해 평균 클래스 엔트로피를 계산한다. 그러나 임의의 위치에 있는 샘플 분할 포인트를 추출하고 이를 평가하기 위해서는 그림 2와 같은 빈즈를 이용하는 것은 비효율적이다. 왜냐하면 특정 샘플 분할 포인트 x의 평균 클래스 엔트로피를 계산하기 위해서는 x이전의 빈즈들을 읽어야 하기 때문이다. 본 논문에서는 이러한 문제를 해결하기 위해 그림 3과 같은 누적된 빈즈를 이용한다.

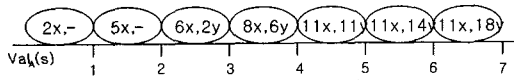

그림 3 그림 2(b)에 대한 누적된 빈즈

클래스 개수가 k일 때, 임의의 분할 포인트 x의 누적된 빈(Bin)의 통계치 벡터는 $(n^1_{x,...,}n^k_x)$이며 $n^i_x = |\{t : t.X \le x \wedge t.C=i\}|$ 이다. 누적된 빈즈는 일반 빈즈를 생성하는 것과 같이 그림 2(a)와 같은 정렬된 값-클래스 쌍을 한번 스캔함으로써 생성할 수 있다. 누적된 빈즈에서 임의의 분할 포인트 x의 평균 클래스 엔트로피 계산은 x의 빈에 이미 필요한 클래스 분포가 누적되어 있기 때문에 이전 빈들을 읽을 필요 없이 x의 빈만을 이용하여 계산할 수 있다. 임의의 구간 $[x_j, x_m]$에서 특정 분할 포인트 x_T의 평균 클래스 엔트로피 계산을 위해 누적된 빈즈의 통계치 벡터를 이용하면 그림 1의 |S|, $|S_1|$, $|S_2|$ 및 $P(C_i, S_1)$, $P(C_i, S_2)$는 그림 4와 같다. |S|, $|S_1|$ 및 $P(C_i, S_1)$등을 구하기 위해 n^i_{xj-1}을 빼서 계산했는데 그 이유는 x_T의 해당 누적된 빈의 통계치 벡터 안에는 이미 n^i_{xj-1}이 포함되어 있기 때문이다. 즉, 그림 3에서 첫 번째 분할 포인트가 2로 결정된 후 구간 [3, 7]에서 분할 포인트를 결정해야한다고 가정하자. 후보 분할 포인트 3에 대해 평균 클래스 엔트로피를 구해야한다면 그

$$|S| = \sum_{i=1}^{k}(n^i_{xm}-n^i_{xj-1}),\ |S_1| = \sum_{i=1}^{k}(n^i_{xT}-n^i_{xj-1}),\ |S_2| = \sum_{i=1}^{k}(n^i_{xm}-n^i_{xT})$$

$$P(C_i,S_1) = \frac{(n^i_{xT}-n^i_{xj-1})}{|S_1|} \quad P(C_i,S_2) = \frac{(n^i_{xm}-n^i_{xT})}{|S_2|}$$

그림 4 누적된 빈즈의 통계치 벡터를 이용한 경우, |S|, $|S_1|$, $|S_2|$, $P(C_i, S_1)$, $P(C_i, S_2)$ 수식

림 4의 식을 이용하면 $|S| = (11-5) + (18-0) = 26$, $|S_1| = (6-5) + (2-0) = 3$, $|S_2| = (11-6)+(18-2) = 21$ 이며 $P(x, S_1) = 1/3$, $P(y, S_1) = 2/3$, $P(x, S_2) = 5/21$, $P(y, S_2) = 16/21$이 된다.

누적된 빈즈를 생성한 후 최적의 분할 포인트를 선택하는 방법은 다음과 같다. 구간 $[x_1, x_m]$ 중에서 임의의 샘플 분할 포인트 q개를 추출한다. 추출된 샘플 포인트에 구간의 시작 포인트와 마지막 포인트 x_1과 x_m은 항상 포함된다. 추출된 샘플 포인트에 평균 클래스 엔트로피를 적용한 후 최하의 엔트로피 값을 Ent_{min}라 한다. Ent_{min}이 샘플 포인트에서뿐만 아니라 모든 구간$[x_i, x_{i+1}]$에서도 최선임을 증명하기 위해 정리 2를 이용한다. 각 샘플 포인트 구간 $[x_i, x_j]$에서 가상의 통계치 벡터를 추출하여 이로부터 하위 경계값, Ent^{est}를 계산하여 $Ent^{est} \geq Ent_{min}$이면 다음 구간 $[x_{i+1}, x_{i+2}]$로 넘어간다. 그렇지 않으며, $[x_i, x_{i+1}]$의 중앙값 x_{mid}을 계산하여 구간 $[x_i, x_{i+1}]$을 두 구간 $[x_i, x_{mid}]$, $[x_{mid}, m_{i+1}]$로 분할한다. 검증은 $[x_i, x_{mid}]$에 대해 계속적으로 이루어지며 구간을 더 이상 분할할 수 없는 오직 하나의 값만 가질 때까지 이루어진다. 검증을 수행하는 동안 Ent_{min}보다 더 적은 평가 값이 나오면 그 값으로 Ent_{min}을 치환하여 검증을 수행함으로써 탐색공간을 줄여나간다.

그러나 임의의 구간 $[x_i, x_j]$의 하위 경계를 계산하기 위해서는 $(n^1_{xi},...,n^k_{xi})$, $(n^1_{xj},...,n^k_{xj})$로 추출되는 2^k개의 가상의 통계치 벡터가 필요하기 때문에 k가 클 경우, 2^k가 $[x_i, x_j]$ 사이의 모든 후보 분할 포인트 개수보다 많은 경우가 발생할 수 있다. 따라서 본 논문에서는 구간 $[x_i, x_j]$의 후보 분할 포인트 개수가 2^k보다 적으면 하위 경계를 계산하지 않고 바로 $[x_i, x_j]$ 안의 후보 분할 포인트를 검증하는 방법을 택하였다.

3.2 순차적 전역적 범주화

일반적으로 범주화를 위한 빈즈는 단일 리스트 형태로 저장된다. 그러나 빈즈를 단일 리스트 형태로 구성할 경우, 데이타가 대용량이고 범주화를 수행할 변수의 범위가 매우 크면 단일 리스트를 생성하기 위해 많은 정렬 및 병합을 수행해야한다는 문제점을 안고 있다. 즉, 일반적으로 전역적 범주화 알고리즘은 빈즈를 생성하기 위해 데이타를 메모리 한계까지 읽어들여 정렬한 후 빈즈를 생성하고 데이타가 더 존재하면 데이타 읽기, 정렬, 빈즈 생성을 반복한 후 단일 리스트 형태의 빈즈를 구성하기 위해 기존의 빈즈와 병합한다. 병합은 O(N)의 시간에 수행되지만 리스트의 크기가 너무 커 디스크에서 병합을 수행할 경우 비효율적이다. 또한 기존의 범주화 기법은 거의 일괄처리 방식으로 범주화를 수행하기 때문에 새로운 데이타가 추가되면 이 데이타가 반영된 범주를 생성하기 위해 처음부터 다시 범주화를 수행해야 한다. 이는 또 다시 많은 시간을 필요로 하며 새로운 데이타가 계속적으로 추가되는 환경에는 적합하지 못하다.

본 논문은 이러한 문제점을 해결하기 위해 단일 리스트형태의 빈즈를 구성하지 않고 트리 형태의 범주 모델을 생성하고 새로운 빈즈가 추가되면 이를 범주 모델에 반영하여 순차적으로 범주를 생성하는 기법을 제안한다. 범주 모델은 그림 5와 같이 선택된 분할 포인트가 노드가 되고 왼쪽 서브 트리는 노드의 값보다 작으며 오른쪽 트리는 노드의 값보다 크다는 특성을 갖는다. 범주 트리의 노드에는 분할 포인트 선택에 이용한 샘플 포인트가 유지되며, 그림 5는 분할 포인트가 2인 노드 정보를 나타낸다.

그림 5(a)는 구간 [1, 7]에서 샘플 분할 포인트를 추출한 상태를 나타내며 각 샘플 포인트는 해당 빈에 대한 누적된 통계치 벡터를 가지고 있다. 새로운 빈즈가 추가되면 샘플 포인트는 그림 5(b)와 같이 새로 추가된 빈즈를 포함하도록 갱신된다. 이는 이진 탐색을 이용하여 수행하며 샘플 포인트가 갱신되면서 이진 탐색을 위한 탐색 구간을 줄여 갱신 속도를 향상시킨다. 즉, 샘플 포인트 3을 갱신하기 위해서는 [2, 8]에서 해당 빈을 탐색했지만 5를 갱신하기 위해서는 [2.5, 8]사이에서 탐사를 수행한다.

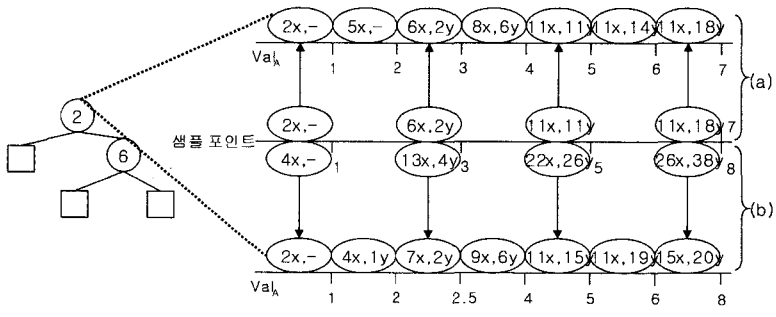

그림 5 범주 트리와 노드 정보

그림 5에서 새로 추가된 빈즈에 대한 인덱스는 샘플 포인트가 유지하고 있는 값보다 같거나 적으면서 가장 큰 빈에 생성된다. 즉, 샘플 포인트 sp는 다중 빈즈에 대해 sp가 유지하고 있는 값과 같거나 적으면서 가장 큰 빈에 대한 인덱스를 가지고 있다. 그림 5(b)에서 샘플 포인트 3은 3보다 작으면서 가장 큰 2.5에 링크 된다. 새로운 빈즈에 대한 인덱스가 생성되면 해당 빈의 클래스 분포는 샘플 포인트의 통계치 벡터에 누적된다. 샘플 포인트와 같거나 적으면서 가장 큰 빈의 클래스 정보를 샘플 포인트에 누적시키는 이유는 샘플 포인트 sp의 통계치 벡터 $(n^1_x,...,n^k_x)$를 $n^i_x = |\{t : t.X \le x \wedge t.C=i\}|$ 같이 유지함으로써 샘플 포인트에 평가함수를 바로 적용할 수 있도록 하기 위해서이다. 또한, [그림 5]에서 마지막 샘플 포인트 7은 새로 추가된 빈즈의 마지막 빈에 인덱스 되도록 설정되고 샘플 포인트 값을 7에서 8로 변경하였다. 그렇지 않으면 후보 분할 포인트 8은 분할 포인트 고려 대상에서 제외되기 때문이다.

그림 6은 새로운 빈즈에 대한 정보를 현재 노드의 샘플 포인트에 반영한 후 순차적으로 범주화를 수행하는 알고리즘을 나타낸다. $[n^1_{prev},...,n^k_{prev}]$은 클래스 수가 k일 때, 이전 구간의 분할 포인트의 누적 통계치 벡터를 나타내며 그림 4와 같이 이용되며 왼쪽 서브 트리에는 이 통계치 벡터가 그대로 전달되며 오른쪽 서브 트리에는 현재 노드의 분할 포인트의 누적된 통계치 벡터가 전달된다. 그림 6의 step3(a)는 구간에 대한 하위 경계값 예측이 비효율적인 경우로서 이때는 그 구간에 속하는 모든 후보 분할 포인트를 추출하여 이를 직접 평가한다. 그림 6의 step4와 같이 자식 노드가 유지하고 있는 샘플 포인트를 수정하는 이유는 다음과 같다. 그림 5에서 현재 루트 노드의 분할 포인트는 2이며 왼쪽 노드의 구간은 [1, 2)이며 오른쪽 자식 노드의 구간은 [3, 7]이다. 새로운 빈즈 추가 후 루트 노드의 분할 포인트가 2에서 3으로 변경된다면 왼쪽 자식 노드의 구간은 [1, 3)이고 오른쪽 자식 노드의 구간은 [4, 8]로 변경되기 때문에 샘플 포인트 역시 이에 맞게 변경해야한다.

4. 실험 및 성능 평가

범주화 알고리즘에 관한 성능 평가로서 분류의 정확도와 범주를 생성하는 속도를 가장 많이 사용한다. 그러나 본 논문에서 제안하는 접근 방법을 이용하여 생성된 범주는 같은 평가함수를 사용할 경우, Elomma의 세그먼트를 이용한 접근 방법이 생성하는 범주와 일치하기 때문에 두 방법 모두 같은 구조의 분류기를 생성하며 따라서 두 방법의 분류의 정확도도 같다. 따라서 본 논문에서는 범주를 생성하는 속도에 대해서만 실험하였으며, Pentium-III 1G, 메모리 128MB, WINDOW 2000 환경 하에서 모든 실험을 수행하였다.

첫 번째 실험은 모든 데이타가 메모리에 적재될 수 있는 소용량 데이타에 대해 평가함수 적용 횟수 및 속도에 대해 실험하였으며 실험 데이타는 UCI 도매인[11]

incremental_discretization(node, bins, $[n^1_{prev},...,n^k_{prev}]$)

step1. node에 있는 샘플 포인트가 bins의 해당 bin을 링크 하도록 갱신

step2. 각 샘플 포인트에 평균 클래스 엔트로피를 적용하여 최소 엔트로피를 Ent_{min}로, 그 샘플 포인트를 best로 설정

step3. 모든 샘플 포인트 sp_i에 대해

step3(a). 2^k 가 $[sp_i, sp_{i+1}]$의 후보 분할 포인트 개수보다 크면

step3(a1). $[sp_i, sp_{i+1}]$사이의 모든 분할 포인트에 대해 클래스 평균 엔트로피 계산한 후 그 중 최소값을 Ent^{est}로 치환

step3(a2). $Ent_{min} > Ent^{est}$ 이면 Ent_{min}를 Ent^{est}, 그 sp를 best로 치환 후 stpe3으로

step3(b). 그렇지 않으면 구간 $[sp_i, sp_{i+1}]$의 하위 경계값, Ent^{est} 계산

step3(b1). $Ent_{min} > Ent^{est}$ 이면 $[sp_i, sp_{i+1}]$의 중간 샘플 포인트 sp_{mid}를 $[sp_i, sp_{i-1}]$ 사이에 추가하고 stcp3으로

step3(b2) 그렇지 않으면 step3으로

step4. 최적의 분할 포인트 best와 이전의 분할 포인트가 서로 다르면 자식 노드의 샘플 포인트 변경

step5. 모든 자식 노드에 대해

incremental_discretization(node->left, bins, $[n^1_{prev},...,n^k_{prev}]$)

incremental_discretization(node->right, bins, $[n^1_{best},...,n^k_{best}]$)

그림 6 순차적 전역적 범주화 알고리즘

표 1 사용한 UCI 실험 데이타]

데이타 셋	예제수	수치형 변수 갯수	전체 변수 갯수	클래스 갯수
german	1000	7	20	2
iris	150	4	4	3
wine	178	13	13	3
waveform	5000	21	21	3
satimage	4435	36	36	6
segment	2310	19	19	7
shuttle	43500	9	9	7

표 2 실험 결과

	평가함수 적용횟수		시간(초)	
	세그먼트 사용	제안하는 방법	세그먼트 사용	제안하는 방법
german	60	146	0.451	0.461
iris	138	170	0.361	0.361
wine	996	1842	0.492	0.531
waveform	43061	43158	1.482	1.893
satimage	9205	11919	1.663	1.843
segment	28716	39242	1.082	1.482
shuttle	2703	3990	2.432	2.523

중에서 추출하였다. 표 1은 사용된 실험 데이타의 특성을 나타내며 표 2는 실험 결과를 나타낸다.

표 2의 두 번째, 세 번째 칼럼은 평가함수 적용 횟수를 비교하였으며 네 번째, 다섯 번째 칼럼은 범주화를 위해 필요한 시간을 비교하였다. 실험 결과를 보면, 제안하는 방법이 세그먼트를 사용하는 방법보다 좋지 못하다. 그 이유 중 하나는 클래스 수가 많아 구간 $[x_i, x_j]$의 하위 경계값을 계산하기 위해 필요한 통계치 벡터의 개수, 2^k가 구간 $[x_i, x_j]$ 사이의 모든 후보 분할 포인트 개수보다 큰 경우가 많이 발생했기 때문이다. 이러한 경우에 본 논문에서는 구간에 대한 하위 경계값을 예측하지 않고 실제 모든 후보 분할 포인트를 평가하기 때문에 세그먼트를 이용하는 경우보다 더 많은 분할 포인트를 평가하는 경우가 발생하였다. 또 다른 이유 중에 하나는 실험한 데이타의 수가 많지 않고 수치형 변수의 범위가 크지 않기 때문에 구간에 대한 하위 경계값을 예측함으로써 얻을 수 있는 탐색 공간 축소 효과를 얻지 못했기 때문이다. 그러나 실제 수행 시간의 비교를 보면 표 2와 같이 수행 시간의 차가 1초미만으로 매우 근소한 차이밖에 나지 않았다. 그 이유는 범주화를 위한 필요한 빈즈가 모두 메모리에 적재되었으며 메모리 상에는 분할 포인트 평가가 매우 빠른 시간에 이루어졌기 때문이다.

두 번째 실험은 범주화에 필요한 빈즈가 메모리에 적재되지 못하는 대용량 데이타에 대해 본 논문의 접근 방법의 성능을 평가하였다. 이를 위한 실험 데이타로서 [12]에서 제안한 가상의 데이타를 이용하였다. 가상의 데이타는 9개의 변수와 한 개의 클래스 변수로 구성되어 있으며 10개의 분류 함수를 포함하고 있어서 다양한 복잡도에 따른 데이타를 생성할 수 있다. 본 실험에서는 표 3과 같은 분류 함수 5, 9를 사용하였다. 분류 함수-5는 세 개의 변수가 클래스 분류에 결정적인 역할을 수행하며 분류 함수-9에서는 클래스가 다른 4개의 변수의 선형적 조합에 의해 결정된다.

대용량 데이타에 대한 실험을 위해 세그먼트를 이용하는 기존의 방법과 본 논문의 접근 방법은 다음과 같이 구현하였다. 우선, 데이타를 메모리 한계까지 읽어들인 후 여러 수치형 변수 중 메모리를 가장 많이 차지하고 있는 변수를 선택하여 이를 빈즈 형태로 디스크에 저장하였다. 이 과정은 데이타를 모두 읽을 때까지 계속 수행되며 데이타의 끝에 도달하면 세그먼트를 사용하는 경우, 메모리에 존재하는 빈즈와 디스크에 저장한 빈즈와 병합하여 하나의 빈즈를 생성한 후 세그먼트 조건을 만족하는 경계 포인트만 평가하여 범주화를 수행하였다. 세그먼트를 사용할 경우, 세그먼트 형태로 디스크에 저장하지 않고 빈즈 형태로 저장한 이유는 세그먼트로 저장할 경우 계속 추가되는 데이타로 인하여 세그먼트 조건이 위배되는 상황이 발생할 수 있기 때문이다.

그러나 단일 빈즈를 생성하기 위한 병합을 수행할 때 변수와 범위가 크고 데이타가 많을 경우 이를 메모리에서 수행하지 못하는 경우가 발생하였다. 사용한 실험 데이타에서는 hvalue 변수의 경우가 이러한 경우였으며 이 문제를 해결하기 위해 디스크에서 병합을 수행하였으며 디스크 상에서 범주화를 수행하였다. 본 논문에서

표 3 분류함수-5, 9와 변수 설명

분류함수-5 Group A:
((age<40)∧(((50K≤salary≤100K) ? (100K≤loan≤300K) : (200K≤loan≤400K))) ∨
((40≤age<60)∧(((75K≤salary ≤125K) ? (200K≤loan≤400K) : (300K≤loan≤500K)))∨
((age≥60) ∧(((25K≤salary ≤75K) ? (300K≤loan≤500K) : (100K≤loan≤300K))))
분류함수-9 Group A:
disposable > 0
where disposable = (0.67×(salary+commission) -5000×elevel - (0.2×loan - 10000))

변수	값
salary	20k에서 150k까지 균등하게 분포
commission	salary≥75k⇒commission=0 그렇지 않으면 10k에서 75k까지 균등하게 분포
age	20에서 80까지 균등하게 분포
elevel	0부터 4까지 균등하게분포
car	1부터 20까지 균등하게 분포
zipcode	1부터 9까지 균등하게 분포
hvalue	0.5×k×100000에서 1.5×k×100000까지 균등하게 분포(k는 zipcode 값)
hyears	1부터 30까지 균등하게 분포
loan	0부터 500k까지 균등하게 분포

표 4 함수-5, 함수-9에 대한 평가함수 적용 횟수

	함수-5		함수-9	
	세그먼트 사용	제안하는 방법	세그먼트 사용	제안하는 방법
100만	1609762	46375	2458155	297298
200만	2383513	60505	3570125	617439
300만	3000729	97954	4365818	859788
400만	3569657	107193	5180158	983812
500만	4121110	103268	5884317	1089906

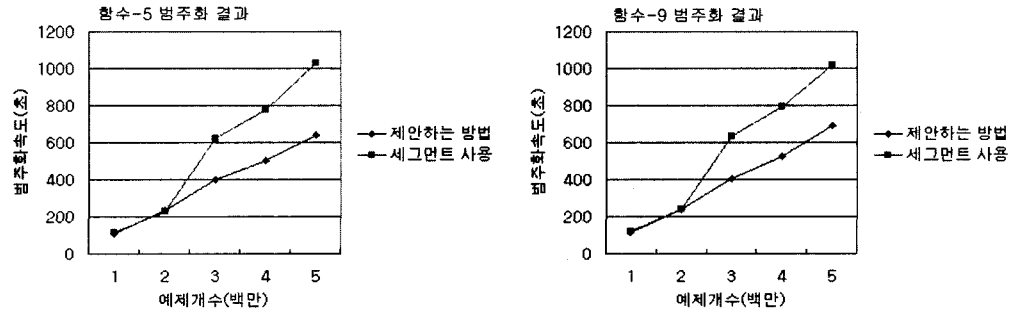

그림 7 함수-5, 함수-9에 대한 범주화 속도

는 디스크 상에서의 범주화를 위해 임의 접근 파일(Random Access File)을 이용하였으며 임의 접근 파일을 이용한 이유는 특정 클래스 분포를 읽기 위해 처음부터 읽지 않고 파일 포인터를 이용하여 해당 위치로 바로 접근할 수 있는 방법을 제공하기 때문이다.

표 4 및 그림 7은 함수-5와 함수-9에 대한 일괄처리 방식으로 범주화를 수행할 경우 결과이며 예제의 개수를 1백만 레코드에서 5백만 레코드까지 증가시키면서 성능을 비교하였다. 표 4는 각 접근 방법의 분할 포인트 결정을 위한 평가함수 계산 횟수를 비교하였다. 표 4를 보면, 제안하는 방법이 세그먼트를 사용하는 것보다 적은 수의 평가함수를 계산하였다. 그 이유는 클래스 수가 두개이기 때문에 구간에 대한 하위 경계값을 계산하기 위해 필요한 통계치 벡터의 개수가 네개밖에 필요하지 않았으며 또한, 데이타가 대용량이고 수치형 변수의 범위가 크기 때문에 하위 경계값을 통한 탐색공간 축소 효과를 얻을 수 있었기 때문이다. 그림 7은 범주화를 수행하는데 필요한 시간을 나타내며 제안하는 방법이 세그먼트를 사용하는 방법보다 더 좋은 성능을 보임을 알 수 있다. 그 가장 중요한 이유는 세그먼트를 사용할 경우 단일 빈즈를 생성하기 위해 병합을 수행해야하지만 본 논문에서 제안하는 방법은 다중 빈즈에서도 범주를

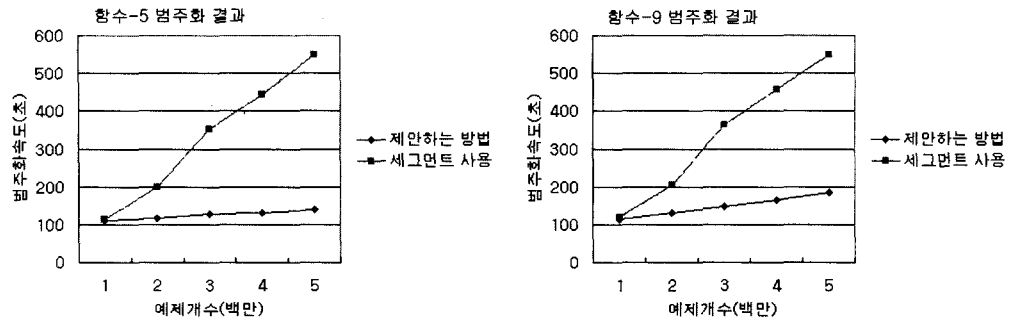

그림 8 함수-5, 함수-9에 대한 순차적 범주화 속도

생성할 수 있기 때문에 병합을 필요로 하지 않기 때문이다. 특히, 디스크에서 병합을 수행할 경우 많은 시간을 필요로 하였다. 그림 7에서 보면 예제 개수 3백만에서 세그먼트의 수행시간이 갑자기 증가하였는데 이는 예제 2백만까지는 hvalue의 병합이 발생하지 않았으며 또한 모든 변수가 메모리 상에서 범주화가 수행되었지만 3백만 레코드에서는 hvalue에 대해 메모리 상에서 병합을 할 수 없어 디스크 상에서 병합을 수행했기 때문이다.

그림 8은 함수-5와 함수-9에 대해 순차적으로 범주화를 수행한 결과이며 예제의 개수를 1백만 레코드씩 추가하여 범주화 속도를 비교하였다. 이 경우 그림 8과 같이 본 논문에서 제안하는 접근 방법이 더 좋은 성능을 보였는데 그 이유는 새로 추가된 데이타가 반영된 범주를 생성하기 위해 기존의 방법은 유지하고 있는 빈즈와 병합을 수행해야하지만 본 논문에서 제안하는 방법은 기존 빈즈와 병합 없이 유지하고 있는 샘플 포인트의 인덱스와 클래스 분포 정보만 갱신함으로서 새로 추가된 데이타가 반영된 범주를 생성할 수 있기 때문이다.

5. 결론 및 향후 연구

본 논문에서는 대용량 데이타로부터 범주를 생성하기 위해 샘플 포인트를 이용하는 접근 방법을 제안하였으며, 샘플 포인트에 특정 후보 분할 포인트의 해당 빈즈에 대한 인덱스 및 클래스 분포를 유지하였다. 두 개의 샘플 포인트에서 유지하는 클래스 분포 정보로부터 유도되는 가상의 통계치 벡터로부터 구간에 대한 평가 함수의 하위 경계값을 예측할 수 있었으며 이를 최종 분할 포인트 결정에 이용하였다. 또한 제안하는 방법은 새로운 데이타가 추가될 경우, 이를 반영한 범주를 생성하기 위해 유지하고 있는 샘플 포인트의 인덱스와 클래스 분포 정보만 갱신함으로서 새로 추가된 데이타가 반영된 범주를 생성할 수 있기 때문에 새로운 데이타가 계속적으로 추가되는 환경에 더 효율적임을 알 수 있었다. 본 논문의 이러한 접근 방법은 단일 빈즈를 생성하기 위해 병합을 수행해야하는 기존의 문제점을 제거하였으며 실험을 통해 이를 증명하였다.

향후 연구로는 첫째, 구간에 관한 하위 경계값을 결정하기 위해 필요한 통계치 벡터의 개수 축소에 관한 연구이다. 본 논문의 접근 방법에서 특정 구간의 하위 경계값을 예측하기 위해서는 k개의 클래스가 있을 때, 2^k 개의 가상의 통계치 벡터를 유추하고 이들에 평가함수를 적용하여 그 중 최소값을 그 구간에 대한 하위 경계값으로 예측하였다. 이와 같은 접근 방법은 클래스 개수, k가 많게 되면 평가해야할 통계치 벡터가 많아지기 때문에 세그먼트를 이용하는 기존의 방법보다 비효율적이다. 두 번째 향후 연구로는 본 논문에서 제안하는 범주화 기법과 기존의 기계학습 알고리즘과의 통합이다. 범주화 기법은 기계학습의 정확도 및 수행 속도를 향상시키기 위한 전처리 단계로서 많이 사용되어 왔다. 본 논문의 접근 방법을 기계학습과 통합할 경우 많은 성능 향상이 기대되며 특히, 본 논문의 접근 방법의 순차적인 특성으로 인해 순차적인 기계학습에 좋은 효과를 가져올 것으로 기대된다.

참 고 문 헌

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, 1984.

[2] J. R. Quinlan. Induction of decision trees. *Machine Learning,* 1:8 1-106, 1986.

[3] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning,* 29, pp. 131-161, 1997.

[4] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous Features. *Proceedings of Twelfth International Conference on Machine Learning,* pp. 194-202, 1995.

[5] U. M. Fayyad, K. B. Irani. On the handling of continuous-valued attributes in decision tree gene-

ration. *Machine Learning*, 8, 87-102, 1992.
[6] U. M. Fayyad, K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning, *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, pp. 1022-1027.
[7] T. Elomaa and J. Rousu. General and efficient multisplitting of numerical attributes. *Machine Learning*, 36:200-244, 1999.
[8] T. Elomaa and J. Rousu. Generalizing boundary points. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, Menlo Park, CA, 2000. AAAI Press. In press.
[9] K. Alsabti, S. Ranka, and V. Singh. CLOUDS: A Decision Tree Classifier for Large Datasets. In *Proc. KDD-98*, New York City, New York, 1998.
[10] J. Gehrke, V. Ganti, R. Ramakrishnan, and W. Loh. Boat-- optimistic decision tree construction. *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1999.
[11] C. J. Merz, P. M. Murphy. *UCI repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science, 1996.
[12] R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):914-952, 1993.

한 경 식
1997년 숭실대학교 공과대학 인공지능학과 졸업(학사). 1999년 숭실대학교 정보과학대학 컴퓨터학과 졸업(석사). 2002년 숭실대학교 정보과학대학 컴퓨터학과 박사수료. 2002년~현재 (주)인우기술 선임연구원. 관심분야는 인공지능, CRM, 데이타마이닝

이 수 원
1982년 서울대학교 계산통계학과 학사 1984년 한국과학기술원 전산학과 석사 1994년 U. of Southern California 전산학과 박사. 1995년~현재 숭실대학교 컴퓨터학부 부교수. 관심분야는 인공지능, 데이타마이닝, 에이전트, 기계학습