결합적 방법에 의한 귀납법칙 집합의 생성

이 창 환†

요 약

귀납법칙 생성 시스템은 데이터에서부터 법칙을 자동으로 발견하는 시스템으로서 현재 많은 연구가 진행되고 있다. 본 논문은 정보이론을 이용하여 데이터로부터 귀납법칙을 자동으로 생성하는 시스템을 제시하고 또한 귀납법칙 생성 시스템에 의하여 생성되는 규칙들 중에서 가장 좋은 성능을 보이는 규칙 집합을 구하기 위하여 이를 유전자 알고리즘과 결합시켜 최적화된 귀납법칙 집합을 탐색하는 방법을 제시하였다. 제안된 시스템의 성능을 평가하기 위하여 다수의 기계학습 데이터를 사용하여 기존의 다른 방법들과 비교하였으며, 제안된 시스템은 대부분의 경우에 좋은 정확도를 제공하였다.

An Integrated Method for Generating Inductive Rule Sets

Chang-Hwan Lee†

ABSTRACT

The rule induction system generates a set of inductive rules, and the task of selecting an optimal rule subset is one of the important problem in the area of rule induction. This paper proposes a new learning method which combines rule induction system with the paradigm of genetic algorithm. This paper shows that genetic algorithm can be effectively applied to optimal rule selection problem. The proposed system was evaluated using a set of different machine learning data sets and, showed better performance in all cases than other traditional methods.

키워드 : 인공지능, 데이터 마이닝

1. 서 론

인간의 직관으로부터 직접적으로 얻어진 규칙은 정보 전달의 어려움 등으로 인하여 부정확하며 많은 비용이 든다. 그렇기 때문에 주어진 데이터베이스로부터 전문가 수준의 자동적으로 추론되어진 규칙을 사용할 수 있다면 대단히 유용할 것이다. 이런 이유로 인하여 최근에 데이터 마이닝의 일부로서 여러 종류의 귀납법칙 생성 시스템들이 개발되어 사용되고 있다. 현재에 통용되는 귀납법칙 시스템은 PRISM (Cendrowska, 1987)[1], CN2(Clark and Niblett, 1989)[2]와 규칙 귀납 시스템의 AQ 시리즈[3] 등이 있다. 그들의 귀납법칙 생성의 방법에는 차이가 있지만 기본적인 방식은 단일 최고 규칙(single-best-rule)의 방법을 사용하는 것인데 이 방법은 한번에 하나의 규칙을 만들어 확장하며, 다음에 나올 하나의 조건에다 규칙이 부정 데이터와 일치하기 이전까지의 조건을 더해 나간다.

본 논문은 정보이론을 이용하여 귀납법칙을 자동으로 생성할 수 있는 귀납법칙 생성 시스템을 제안하고 이에 의해서 생성되는 규칙들에 대하여 유전자 알고리즘을 사용하여 최적의 규칙 집합을 결정하는 방법을 제시함으로써 궁극적으로 귀납법칙 시스템의 성능을 향상시킬 수 있는 방법에 대하여 소개하고자 한다. 본 논문에서 제시하는 귀납법칙 생성 시스템은 주어진 데이터를 읽어서 n개의 귀납법칙을 생성하며 각 법칙들은 그 중요도를 함께 표시한다.

다음 단계로서는 생성된 귀납법칙 중에서 미래의 데이터를 가장 잘 예측할 수 있는 최적의 규칙 집합을 구하는 것으로서 이는 NP 문제이며 귀납 법칙분야에서 고려해야 할 큰 문제중의 하나이다. 본 논문에서는 이 문제의 해결을 위하여 유전자 알고리즘을 사용하고자 한다. 유전자 알고리즘은 비교적 간단하면서도 최적치에 근사한 값을 제공할 수 있으며, 기계학습의 여러 분야에서 활용되고 있다.

2. 관련 연구

유전자 알고리즘과 귀납법칙생성 방법이 결합된 예로써 대표적인것은 GABIL[4]을 들 수 있다. 이는 명제 규칙들의 집합을 2 진수로 표현하고 이렇게 표현된 집합을 유전자 알고리즘으로 이용하여 규칙을 생성하는 시스템으로, 그 결과

† 정 회 원 : 동국대학교 정보통신학과 교수
논문접수 : 2001년 9월 25일, 심사완료 : 2002년 11월 21일

는 C4.5[5]나 AQ 시스템[3]과 비슷한 성능을 보이고 있다. 이와 같이 유전자 알고리즘 자체를 학습 시스템의 기본 방법으로 응용하기도 하지만 다른 학습방법과 결합하여 다중화 학습 시스템의 일부로 사용하기도 한다.

유전자 알고리즘을 이용한 다중 전략의 방법의 다른 예로서는 Vafaie와 De Jong[6]의 연구들이 있다. 이들은 시스템의 수행 시간을 단축시키고 주위 환경의 변화 또는 잡음 등으로 인한 외부적인 장애를 극복하고자 다중 전략방법을 사용하였다. 그들은 유전자 알고리즘을 사용하여 주어진 데이터 집합을 조사하여 탐색 공간을 줄이고 줄여진 데이터의 집합들을 AQ 알고리즘으로 규칙들을 생성한 후 인식률을 계산하게끔 하는 방식을 취하였다.

본 연구에서 이러한 연구와 달리 귀납법칙 시스템에서 생성한 귀납법칙들 중에서 가장 최적의 부분 집합을 구하는 문제에 대하여 유전자 알고리즘을 적용하고자하여 전체적인 학습 시스템의 성능을 향상시키고자 한다.

3. 귀납 법칙의 생성

본 절에서는 제안된 시스템이 사용하고 있는 귀납법칙 생성 방법에 대한 내용을 간략히 설명하고자한다. 제안한 귀납법칙 시스템에서 규칙 생성의 가장 기본적인 가정은 각 규칙의 왼쪽 면에 있는 조건이 오른쪽 면의 확률 분포에 영향을 준다는 가정에서 출발한다. 직관적으로 말하자면, 특정 속성의 값이 정해질 때 목표 속성(target attribute)의 확률 분포를 현저히 변화시킨다면 이는 특정 속성의 값을 결정하는 중요한 역할을 의미한다. 예를 들어서 귀납법칙 시스템이 특정한 속성의 변수값(예를 들어서 B 속성이 b의 값을 가짐)을 우선 선택한 후, $B=b$이라는 사건이 목표 속성 A의 분포값에 어떤 영향을 끼치는 가를 점검한다. 만약 그것이 목표 속성의 확률 분포에 상당한 영향을 끼친다면 시스템은 다음과 같은 규칙이 있음을 가정한다.

$$B = b \rightarrow A = a \ with \ H$$

각 규칙은 각 규칙의 중요도를 표현하는 H값을 포함하고 있다. 규칙의 왼쪽은 논리곱(conjunction)의 형태로 되어 있으며 논리합과 부정 논리의 형태는 고려하지 않는다.

이러한 목표 속성의 확률분포의 변화 정도를 측정하는 엔트로피 함수를 정의하기 위하여 본 논문의 귀납법칙 생성 시스템은 헬링거(Hellinger) 함수[7]라고 불리는 엔트로피 함수를 사용하였다. 헬링거 함수는 다음과 같이 정의된다.

$$[\sum_{a \in A} (\sqrt{P(a)} - \sqrt{(P(a \mid b))}^2]^{\frac{1}{2}} \quad (2)$$

이 식은 속성 A에 대해서 $B=b$의 값 이전의 확률 분포와 $B=b$값 이후의 확률분포에 대하여 얼마나 차이가 있는가에 대한 계산식이다.

또 하나 규칙 생성 환경에서 고려해야할 사항은 목표 속성의 특별한 하나의 값은 규칙의 오른 편에서 나타나고 그래서 다른 값들의 확률들은 $1-P(a)$에 포함되어진다. 다시 말해 규칙의 정확도를 측정하는 데에 있어서 중요한 점은 개체가 규칙의 오른 편의 목표값 $(A=a)$에 속하는가를 검사하는 것이다. 예를 들어서 목표 속성 A가 k개 $(a_1, a_2, \cdots, a_k)$의 값을 가지고 있고 규칙은 오른 편에서 $A=a_1$이라고 가정하자. 속성 A의 확률 분포가 $(P(A=a_1), P(A \neq a_1))$와 같이 이진 확률 분포로 변환된다. 그래서 식 (2)는 다음과 같이 변환될 수 있다.

$$(\sqrt{P(a \mid b)} - \sqrt{P(a)})^2 + (\sqrt{1-P(a \mid b)} - \sqrt{1-P(a)})^2 \quad (3)$$

이 식에서 $P(a \mid b)$는 $B=b$라는 조건 하에서 $A=a$의 조건 확률을 의미한다.

규칙 귀납에서 우리가 고려해야할 또 다른 문제는 규칙이 얼마나 일반적인(general) 규칙인가 하는 것을 결정하는 것이다. 규칙을 만족하는 데이터의 수가 많을수록 그 규칙은 더욱 일반성이 있다고 할 수 있다. 그래서 규칙의 중요도의 부분으로써 규칙의 보편성의 정도를 계산하여야 한다. 규칙의 일반성을 나타내기 위하여 귀납법칙 시스템에서는 다음 수식을 사용하였다.

$$\sqrt{P(a)} \quad (4)$$

결과적으로 H 계산은 규칙의 중요도를 주어진 규칙의 정확도와 일반성의 복합적인 계산으로 표현한다. 주어진 규칙에서 중요도의 마지막 형태는 다음과 같이 규칙의 일반성과 정확도 사이에서 생성된 식으로 정의된다.

$$H = \sqrt{P(a)} \ [(\sqrt{P(a \mid b)} - \sqrt{P(a)})^2 + (\sqrt{1-P(a \mid b)} - \sqrt{1-P(a)})^2]$$

4. 귀납법칙 생성의 방법

본 논문의 귀납법칙 시스템은 이산 속성(discrete attribute)의 형태로 된 데이터들을 읽어서 H값의 순서대로 정렬된 k개의 규칙을 생성한다. 본 시스템에서 규칙을 생성하는 방법을 간략히 설명하면 아래와 같다. 먼저, 규칙의 왼편이 한 개의 속성조건만을 갖는 단일조건(single-condition) 규칙들을 생성한다. 이들 단일 조건들 중에서 H계산값이 가장 높은 k개의 규칙들이 규칙 리스트의 형태로 저장되며 이들의 H값중 가장 작은 H값은 H_*로 정의된다. 알고리즘은 이들 단일 조건 규칙들에서 출발하여 가능한 왼쪽 면을

통한 깊이우선(depth-first) 탐색을 수행한다. 즉 이 규칙 리스트의 각 원소에 대하여 알고리즘은 더욱 특수화된(specialized) 규칙들을 생성하려고 한다. 알고리즘은 규칙 리스트에서 한 원소를 뽑고 추가적인 속성 조건을 왼편에 추가하여서 특수화시킨다. 그리고 알고리즘은 다음 절에서 설명하는 정리들 중에서 하나를 만족할 때까지 특수화를 계속하고 더 이상 특수화할 법칙이 없을때 알고리즘은 현재 상위 k개의 법칙을 출력한다.

4.1 H값을 이용한 가지치기 방법

일반적으로 귀납법칙 생성 시스템은 속성들의 숫자가 증가함에 따라 계산해야할 규칙들의 총 숫자가 폭발적으로 증가한다. 예를 들어서 데이터가 r개의 속성을 가지고 각 속성들은 $v_i(i=1,\cdots,r)$의 값들을 가진다고 가정하자. 그러면 최악의 경우 총 규칙의 개수는 다음과 같이 주어진다.

$$\prod_{i=1}^{r-1}(v_i+1)-1 \tag{5}$$

본 연구에서는 헬링거 함수의 특성들을 이용한 새로운 가지치기(pruning) 기술을 제안하고자 한다. 우선 다음과 같은 규칙을 가정하자.

$$B=b \rightarrow A=a \tag{6}$$

또한 $C=c$라는 조건을 더해서 다음과 같이 특수화된 규칙을 가정하자.

$$B=b \wedge C=c \rightarrow A=a \tag{7}$$

식 (6)과 식 (7)에서의 규칙들은 각각 Rg와 Rs로 표시하자. 그리고 규칙 Rg와 Rs의 H값을 각각 H_g와 H_s라고 가정하면 이들은 다음과 같이 정의된다.

$$\begin{aligned} Hg &= \sqrt{P(b)}\ [(\sqrt{P(a\mid b)}-\sqrt{P(a)})^2 \\ &\quad +(\sqrt{1-P(a\mid b)}-\sqrt{1-P(a)})^2 \\ &= \sqrt{P(b)}\ [2-2\sqrt{P(a\mid b)P(a)} \\ &\quad -2\sqrt{(1-P(a\mid b)(1-P(a))}] \end{aligned} \tag{8}$$

$$\begin{aligned} Hs &= \sqrt{P(bc)}\ [2-2\sqrt{P(a\mid bc)P(a)} \\ &\quad -2\sqrt{(1-P(a\mid bc))(1-P(a))}] \\ &= \sqrt{P(c\mid b)}\sqrt{P(b)}\ [2-2\sqrt{P(a\mid bc)P(a)} \\ &\quad -2\sqrt{1-P(a\mid bc))(1-P(a))}] \end{aligned} \tag{9}$$

이 경우, 속성 C의 값에 관계없이 우리는 다음과 같은 결과를 얻을 수 있다.

정리 1 : 목표 속성의 클래스 개수를 m이라고 할 때 H_s값은 다음의 경계값을 초과할 수 없다.

$$\begin{aligned} Hs \le \max\{ &\sqrt{P(a\mid b)}\sqrt{P(b)}\ [2\sqrt{m}-2\sqrt{P(a)}], \\ &2\sqrt{P(a)}-\sqrt{(1-P(a\mid b)}\sqrt{P(b)}[2\sqrt{P(a)} \\ &+2\sqrt{(1-P(a)}]\} \end{aligned} \tag{10}$$

정리 2 : 만약 조건 확률 R_s가 1이라면 R_s의 H값은 R_g의 H값을 초과할 수 없다.

$$IF\ P(a\mid b)=1,\ Hs \le Hg$$

이 법칙들은 속성 C에 관한 추가 정보없이 H_s값의 경계를 예상하게 할 수 있게 하였다. 정리 1의 의미는 만약 H_s값의 경계값이 현재의 최소 H값($H*$)보다 작다면 시스템은 더 이상 진행할 특수한 규칙들을 생성할 필요도 없다. 또한 추가로 사용되는 가지치기 방법인 정리 2의 의미는 현재 진행중인 규칙의 조건 확률이 1이라면, 시스템은 더 이상 특수한 규칙을 생성할 필요가 없음을 알 수 있다.

4.2 목표 속성값의 결정

생성된 규칙들은 귀납적인 성질에 기인하여 서로 모순된 몇 가지 규칙들이 존재할 확률이 있다. 새로운 개체의 목표 속성값을 결정하기 위하여 각 귀납 규칙들의 H값은 규칙들의 가중치(weight)로써 간주될 수 있다. 새로운 개체가 주어지면 시스템은 생성된 전체 규칙들중에서 새로운 개체가 만족하는 개체를 선택하고 선택된 규칙들의 H값과 목표속성의 값을 저장한다. 최종적으로 가장 큰 H값을 모은 목표 속성의 값의 최종 분류(classification)의 답으로써 선택된다.

5. 귀납법칙 집합의 선택

지금까지 규칙 귀납 시스템이 주어진 데이터에 대하여 이들을 잘 설명할 수 있는 귀납 규칙들을 순서대로 생성하는 방법에 대하여 설명하였다. 하지만 규칙 귀납 시스템에 있어서 몇 개의 규칙들을 생성하는 것이 최선인가에 대한 답을 얻는 것은 대단히 어려운 문제이다. 많은 규칙들의 생성이 본래의 데이터를 바르게 설명해 준다고 할 수도 없으며 오히려 데이터에 대한 과적합(overfitting)을 초래할 수도 있기 때문이다. 다시 말해 규칙의 생성 순위가 낮은 규칙들로 인해 원래의 데이터가 잘못 설명되어질 수 있다는 것이다. 더우기 생성된 각 규칙들간에 발생할 수 있는 모순(contradiction) 현상은 최적의 규칙 집합을 얻기 위해 고려해야 할 또 다른 문제이다. 일반적인 데이터의 경우 많은 잡음과 누락치(missing value) 등을 포함하고 있으며 이런 불분명한 데이터의 집합은 시스템으로 하여금 서로 상이한 규칙들을 만들어 내게 하는 요인이 될 수 있다. 위와 같은 이유로 많은 규칙들 가운데 최적의 규칙 집합을 구하는 것은 중요한 문제이며, 이를 해결하기 위한 간단한 방법들로 맹목적

방법(Brute-force method)과 탐욕적 방법(Greedy method) 등을 고려해 볼 수 있다.

우선 쉽게 생각해 볼 수 있는 방법으로는 맹목적 방법을 들 수 있다. 그러나 이 방법은 생성된 규칙의 수를 n개라고 할 때 $O(2^n)$의 복잡도를 갖기 때문에 현실적으로 시스템 상에 구현한다는 것은 사실상 의미가 없다.

이런 이유로 구현하기가 쉽고 복잡도 역시 $O(n)$으로 낮은 탐욕적 방법을 시도할 수 있다. 탐욕적 방법은 먼저 첫 번째의 순위를 가진 규칙과 원래의 데이터와의 비교하여 정확도를 구하고, 다음에는 첫 번째 규칙과 두 번째 규칙으로 이루어진 규칙 집합과 비교함으로써 정확도를 구하게 된다. 이렇게 순차적으로 다음 순위의 규칙들을 하나씩 더해나가며 각각의 정확도를 계산하게 되고 계산된 적합도들 중에서 가장 높은 값을 가지는 규칙 집합을 선택하게 된다. 이러한 원리 때문에 규칙 순위상 상위에 있는 규칙들이 대부분 선택되어지고 하위의 규칙들은 제외되므로 만약 최적의 규칙 집합이 {Rule 1, Rule 5, Rule 7, …}의 형태로 이루어져 있다고 가정하면 탐욕적 방법의 알고리즘으로는 최적의 규칙 집합을 구할 수 없게 된다. 이와 같이 맹목적 방법과 탐욕적 방법이 가진 문제점은 최적의 규칙 집합을 구하기 어렵게 하므로 본 논문에서는 귀납법칙 선택의 방법으로 유전자 알고리즘을 사용하려 하는 것이다.

본 연구는 생성된 귀납법칙의 최적의 부분 집합을 구하기 위해 유전자 알고리즘을 적용하고자한다. 유전자 알고리즘은 재생산(reproduction), 교배(crossover) 그리고 돌연변이(mutation)의 3가지 기능을 가지고 탐색을 하는 알고리즘이다. 제안한 시스템에서는 유전자 알고리즘의 적용을 위해 하나의 염색체(chromosome)는 규칙 귀납 시스템에서 생성된 하나의 규칙집합을 의미한다. 그리고 염색체의 각 비트는 귀납법칙 생성 시스템에서 생성된 규칙이 선택되어진다면 1로 그렇지 못하면 0으로 만든다. 이렇게 만들어진 모집단을 가지고 유전자 알고리즘을 시작하면, 처음에는 규칙 귀납 시스템에서 생성된 규칙들로부터 확률적으로 일부의 규칙들이 선택되고 선택된 규칙들의 집합들간에 교배(crossover)가 이루어진다. 유전자 알고리즘의 적용을 위하여 또한 고려해야 할 사항은 각 염색체마다 해당하는 적합도(fitness)를 계산해야 한다.

본 알고리즘에서 각 염색체의 적합도는 기본적으로 염색체로 표현된 법칙집합이 데이터를 얼마나 정확히 분류할 수 있는지의 정도에 따라 결정된다. 먼저 각 데이터는 염색체 내의 각 법칙에 대하여 IF 조건의 해당여부를 조사하고 해당되는 경우는 법칙의 목표속성과 H값을 누적시킨다. 그리고 이 작업을 데이터 전체에 대하여 반복적으로 수행하여서 목표속성의 값에 대한 H값들이 누적되어 졌으면 그 중 가장 높은 H값을 가진 클래스의 목표 속성과 원래 데이터 클래스의 목표 속성을 비교한다. 만약 비교한 두 개의 값이 일치하는 경우의 개수를 c라 하고 전체 데이터의 개수를 t라 한다면 시스템의 적합도는 다음과 같다.

$$\text{적합도} = \frac{c}{t}$$

위와 같은 방법으로 원래의 데이터와 선택된 규칙들을 비교할 경우 선택된 규칙들 중 어느 한 가지 규칙에도 만족하지 않는 데이터들이 존재할 수 있다. 본 연구에서는 이렇게 만족하지 않은 데이터들은 원래 데이터가 갖는 클래스 속성 중에서 분포도가 가장 높은 클래스 속성으로 포함시키는 방법을 사용하였다.

유전자 알고리즘을 이용하여 계속적인 세대 교배를 통해서 최적의 규칙 집합을 탐색해 나가면서 또 하나 고려해야 할 사항은 언제 알고리즘의 수행을 멈출지에 대한 사항이다. 첫 번째 방법으로서는 알고리즘의 종료시점을 적합도가 증가하다가 감소하는 시점에서 잡기도 하지만 이 경우는 구해진 법칙 집합이 국부적 최적값(local optimal)이 나올 수 있으므로 본 연구에서는 임의로 정한 횟수(예를 들면 500회, 1000회 등)만큼 실행시킨 후, 그 중에서 가장 높은 적합도를 가진 규칙들을 선택하게끔 했다.

본 연구에서 제안한 시스템은 맹목적 방법과 탐욕적 방법이 가진 단점을 유전자 알고리즘을 사용하여 해결하기는 하지만, 데이터로부터 규칙을 생성하고 생성한 규칙들로부터 다시 원래의 데이터와 비교하여 적합도를 구하기 때문에 수행의 복잡도가 높은편이다. 만약 주어진 데이터의 수를 d, 염색체의 수를 c, 모집단의 크기를 p, 그리고 세대수를 n이라고 하면 유전자 알고리즘으로 규칙들을 선택하는데 소요되는 복잡도는 $d \times c \times p \times n = O(dcpn)$가 되며, 특히 데이터의 수가 아주 큰 경우에는 시스템의 수행 시간에 상당한 영향을 끼친다. 일반적으로 염색체의 수나 모집단의 크기 등은 수백에서 수천 정도의 크기를 가지므로 비교적 제한되어 있다고 볼 수 있지만 데이터의 수는 이보다는 훨씬 크기 때문에 시스템의 수행 시간을 크게 좌우하게 되며 이 문제는 앞으로 계속 연구해야 할 문제다.

6. 실험 결과

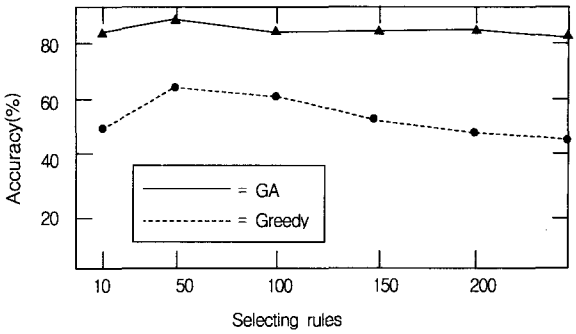
본 시스템의 실험을 위해서 Postoperative Patient Data와 Car Evaluation Database를 사용하였으며 이 데이터들은 University of California Irvine machine Learning database repository에서 채택하였다. Postoperative Patient Data는 외과 수술 후 환자들의 회복의 정도를 기록해 놓은 것들이다. 수술 후 나타나는 환자의 저체온증 현상은 대단히 중요한 문제로 이 데이터 집합의 속성들은 환자의 체온과 혈압 등에 대한 것들로 구성되어 있다. Car Evaluation Database는 자동차의 각 요구 사양에 맞추어 자동차를 평가해 놓은

것들이다. 요구 사항은 속성들로 포함되어 있고 가격(사는 가격과 유지비)과 기술(편안함과 안정도 등)의 6가지 속성으로 구성되어 있다.

〈표 1〉 실험 데이터

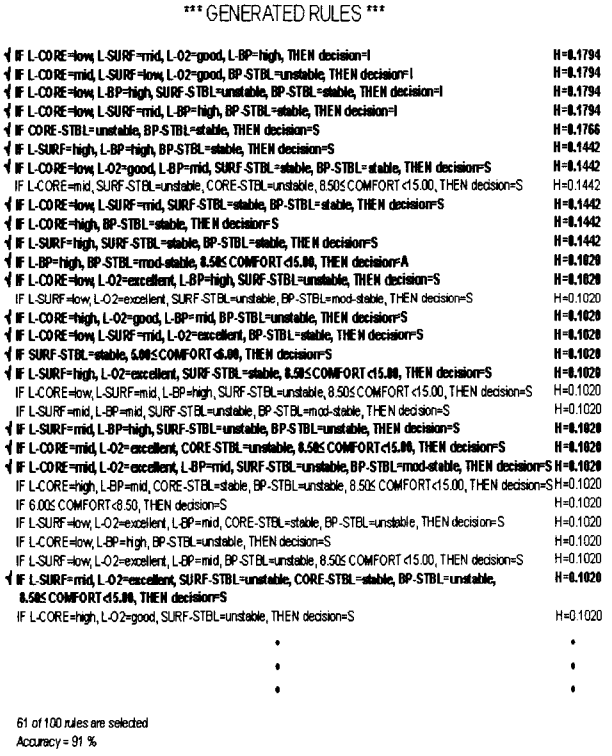
	속성수	레코드수	목표속성의 값
Postoperative Patient Data	90	9	3
Car Evaluation Data	1728	7	4

(그림 1)은 Postoperative Patient 데이터에서 규칙 귀납 시스템에 의해 생성된 규칙들 중에서 유전자 알고리즘에 의해 선택된 규칙들을 보여준다. 이 실험에서는 생성된 100개의 규칙들 중 규칙 1, 2, 3 등 최종 61개가 선택되었으며 규칙 8, 14 등은 상대적으로 누락되었음을 알수 있다. 이 경우 전체 생성된 100개의 법칙 중에서 28개가 최종적으로 선택되었으며 이때의 분류 정확도는 91%였다. Postoperative Patient 데이터의 경우에는 대체적으로 상위 순위의 법칙들은 대부분 포함되어 있으며 하위 순위의 법칙들은 드물게 포함되어 있음을 알 수 있다. (그림 2)는 유전자 알고리즘과 탐욕적 방법을 비교한 그래프로서 이 실험에서는 모집단의 크기를 100으로 하고 세대수를 500으로 수행한 실험이다. 그림에서 알수 있듯이 유전자 알고리즘을 사용하는 경우에 평균 20% 정도의 향상된 정확도를 보인다.

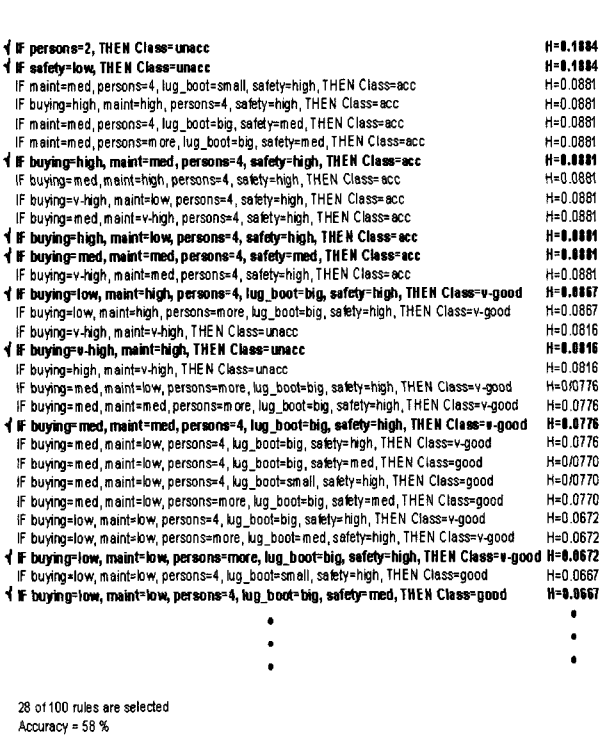


(그림 2) Postoperative Patient Data 에 대한 법칙 선택 방법의 비교(GA의 경우 모집단=100, 세대수=500)

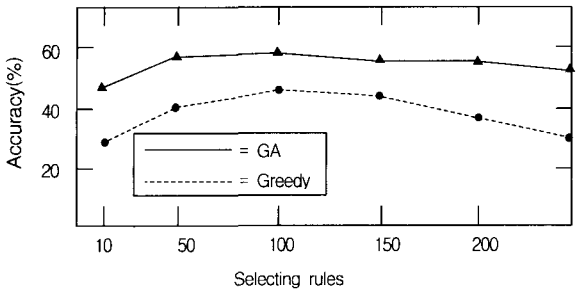
(그림 3)은 Car Evaluation 데이터에서 규칙 귀납 시스템에 의해 생성된 규칙들 중에서 유전자 알고리즘에 의해 최종 선택된 규칙들을 보여준다. 이 실험에서는 생성된 100개의 규칙들 중 규칙 1, 2, 7 등 최종 61개가 선택되었으며 규칙 3, 4, 5, 6, 8 등은 상대적으로 누락되었음을 알 수 있다. 이 경우 전체 생성된 100개의 법칙 중에서 61개가 최종적으로 선택되었으며 이때의 분류 정확도는 58%였다. Car Evaluation 데이터의 경우는 상위 순위의 법칙들에서도 많은 부분이 포함되어있지 않으며 상하위 순위의 법칙들이 골고루 포함되어 있음을 알 수 있다. Car Evaluatioh 데이터의 경우에는 데이터의 노이즈 등으로 인하여 상위에서 생성되는 법칙들의 예측도(predictibility)가 높지않으므로 이같



(그림 1) Postoperative Patient Data 에서의 규칙 선택



(그림 3) Car Evaluation Database 에서의 규칙 선택



(그림 4) Car Evaluation Database 에 대한 법칙선택 방법의 비교(GA의 경우 모집단=100, 세대수=500)

은 현상이 발생하는것으로 보인다. (그림 4)은 유전자 알고리즘과 탐욕적 방법을 비교한 그래프로서 이 실험에서는 모집단의 크기를 100으로 하고 세대수를 500으로 수행한 실험이다. 전체적으로 정확도가 상대적으로 높지않은 현상이지만 그림에서 알수 있듯이 유전자 알고리즘을 사용하는 경우에 평균 10% 이상의 높은 정확도를 보이고있다.

7. 결 론

본 연구는 정보이론을 이용한 귀납법칙 생성시스템을 제시하였고 계산의 복잡도를 감소하기위한 몇가지 법칙을 제시하였다. 그리고 귀납법칙 생성에 있어서 또다른 문제인 최적의 규칙 집합을 찾는 문제를 해결하기 위하여 유전자 알고리즘을 사용하였고 유전자 알고리즘은 이런 문제 해결에 있어서 더 높은 효율을 얻을 수 있음을 확인할 수 있었다.

추후 연구로서는 데이터의 속성수가 많은 경우에 귀납법칙 생성시에 아직도 계산의 복잡도가 높으며 따라서 이 경우에 추가적인 방법이 필요할 것으로 보인다.

참 고 문 헌

[1] Cendrowska, J., "PRISM : An Algorithms for Inducing Modular Rules," Int. J. of Man-Machine Studies, Vol.27 pp.349-379, 1987.

[2] Clark, P. and T. Niblett, "The CN2 Induction Algorithms," Machine Learning, Vol.3, pp.261-283, 1989.

[3] Bloedorn, E. and R. S. Michalski, "Data-driven Constructive Induction in AQ17-DCI : A Method and Experiments," Reports of Machine Learning and Inference Laboratory, Center for Artificial Intelligence, George Mason University, 1991.

[4] De Jong, K. A., Spears, W. H. and Golden, D. F., "Using Genetic Algorithms for Concept Learning," Machine Learning, Vol.13, pp.161-188, 1993.

[5] Quinlan, J. R., "C4.5 : Programs for Machine Learning," San Mateo, CA : Morgan kaufmann, 1993.

[6] Jerzy, W. Bala and Kenneth, A. De Jong and Peter, W. Pachowicz, "Multistrategy Learning from Engineering Data by Integrating Inductive Generalization and Genetic Algorithms," Machine Learning, Vol.4, pp.417-487, 1994.

[7] Beran, R. J., "Minimum Hellinger Distance for Parametric Models," Ann. Statics, Vol.5, pp.445-463, 1977.

[8] Ryszard S. Michalski, "Inferential Theory of Learning," Machine Learning, Vol.4, pp.3-61, 1994.

[9] De Jong, K. "Learning with Genetic Algorithms : An Overview," in Machine Learning, Vol.3, No.3, pp.123-138, 1988.

[10] Haleh Vafaie and Kenneth De Jong "Improving a Rule Induction System using Genetic Algorithms," Machine Learning, Vol.4, pp.453-469, 1994.

[11] DeJong, K. A. Spears, W. M. and Gordon, D. F. "Using Genetic Algorithms for Concepts learning," Machine Learning, pp.161-188, 1993.

[12] Booker, L. B., Goldberg, D. E. and Holland, J. H. "Classifier Systems and Genetic Algorithms," Artificial Intelligence, pp.235-282, 1989.

이 창 환

e-mail : chlee@dgu.ac.kr

1982년 서울대학교 계산통계학과 학사

1988년 서울대학교 계산통계학과 석사

1994년 University of Connecticut 공학박사

1982년~1987년 한국기계연구소

1994년~1995년 AT&T Bell Laboratories

1996년~현재 동국대학교 정보통신학과 부교수

관심분야 : 기계학습, 데이터마이닝