최적화에 근거한 LAD의 패턴생성 기법
(Optimization-Based Pattern Generation for LAD)

장인용(Jang In-Yong)*, 류홍서(Ryoo Hong-Seo)**
고려대학교 산업시스템정보공학과 정보학 연구실
* jjiy@kroea.ac.kr, ** hsryoo@korea.ac.kr(교신저자)

Abstract

The logical analysis of data(LAD) is an effective Boolean-logic based data mining tool. A critical step in analyzing data by LAD is the pattern generation stage where useful knowledge and hidden structural information in data is discovered in the form of patterns. A conventional method for pattern generation in LAD is based on term enumeration that renders the generation of higher degree patterns practically impossible.

In this paper, we present a new optimization-based pattern generation methodology and propose two mathematical programming medels, a mixed 0-1 integer and linear programming(MILP) formulation and a well-studied set covering problem(SCP) formulation for the generation of optimal and heuristic patterns, respectively.

With benchmark datasets, we demonstrate the effectiveness of our models by automatically generating with much ease patterns of high complexity that cannot be generated with the conventional approach.

1 서론

LAD(Logical Analysis of Data)는 데이터 집합에 숨겨진 유용한 정보를 논리적으로 발견해내는 방법론이다. 이 정보들은 데이터 마이닝의 한 분야인 데이터 분류에 효과적으로 쓰여 질 수 있다. 처음 LAD는 [3]에서 이진(binary) 데이터 집합의 분석을 위해 고안되었지만 최근에 [2]에서 그 분석영역을 일반적인 수치(numerical) 데이터 집합에까지 확장을 하였고 성공적인 결과를 보여주었다. 두 종류의 데이터들을 가지는 이원분류문제(binary classification problem)에 대해서 [2]에서 제안된 LAD는 데이터의 이진화(binarization), 서포트 집합(support set)의 선택, 패턴생성(pattern generation), 구별자(classifier) 생성의 4단계를 거쳐 데이터를 분석하게 된다. 이때 두 종류의 데이터를 구별할 수 있는 전체 데이터 속성집합의 부분 집합을 서포트 집합이라 부르며 LAD에 의해서 발견되는 데이터 집합의 숨겨진 논리적 정보를 패턴이라 한다. 패턴은 LAD 구별자의 가장 기본 단위가 되기 때문에 LAD 구별자의 분류 능력은 어떤 패턴이 발견되는가에 가장 크게 영향을 받는다고 할 수 있다. 이러한 점에서 패턴생성은 LAD에서 가장 중요한 과정이라 할 수 있으나 기존의 패턴생성 방법은 열거(enumeration)에 기반한 방법을 쓰고 있어 계산상의 한계점을 가지고 있다. 패턴이 나올 수 있는 가지 수는 패턴의 크기에 지수함수로 증가를 하기 때문에 열거법을 패턴생성의 기반으로 했을 때는 많은 패턴들을 모두 고려하는 것이 불가능하게 되고 분석해야 될 데이터 집합의 속성 크기가 클 경우 이러한 계산상의 한계점이 극대화되어 비효율적인 방법이 된다. 이러한 문제점을 해결하기 위하여 이 논문에서는 최적화에 근거한 효율적인 패턴생성 방법을 제안한다.

본 논문은 다음과 같이 구성되어 있다. 우선 두 번째 절에서는 제안될 방법의 이해를 돕기 위해 [2]에서 제안된 LAD의 일반적인 4단계의 과정에 대해서 요약을 할 것이며 세번째 절에서 최적화에 근거한 패턴생성 방법이 서술될 것이다. 네번째 절에서는 본 논문에서 제안된 방법들에 대한 실험 결과들이 보여 질것이며 마지막 절에서는 그에 대한 결론이 서술될 것이다.

2 LAD의 4단계 절차

양성(positive)과 음성(negative)의 두 가지 종류의 데이터들을 가지는 이원분류문제에 대해 LAD는 다음과 같은 4단계의 과정을 거쳐 구별자를 만들게 된다. 이때 두 종류의 데이터들의 집합을 각각 S^+, S^- 이라하고 각 집합에 포함된 포인트의 수를 m^+, m^-라 하자.

2.1 데이터의 이진화(binarization)

보통 데이터 집합은 binary, nominal, numerical의 3가지 속성(attribute)을 가지고 있다. 이중 nominal, numerical 속성들은 논리적 분석을 위해 이진화(binarization)가 되어야 한다. 우선 $v_1, v_2, \ldots, v_n$의 n개의 값을 가지는 nominal 속성은 $i = 1, \ldots, n$에 대해 다음과 같이 쉽게 이진화 될 수 있다.

$$b(x, v_i) = \begin{cases} 1 & \text{if } x = v_i, \\ 0 & otherwise. \end{cases}$$

또한 $u_1 > u_2 > \ldots > u_n$의 n개의 수치값을 가지는 numerical 속성에 대해서는 다음과 같이 서로 다른 집합에 속하는 $u_i, u_{i+1} (u_i \in S^+, u_{i+1} \in S^- \; or \; u_i \in S^-, u_{i+1} \in S^+)$ 를 이용하여 다음과 같이 cut point를 설정할 수 있다.

$$c_i = (u_i + u_{i+1})/2$$

이 cut point를 이용하여 numerical 속성 값들은 다음과 같은 두 가지 방법을 통해 이진화될 수 있다.

$$b(x, c_i) = \begin{cases} 1 & \text{if } x \geq c_i, \\ 0 & otherwise. \end{cases}$$

$$b(x, c_i, c_j) = \begin{cases} 1 & \text{if } c_i \leq x < c_j, \\ 0 & otherwise. \end{cases}$$

일반적으로 하나의 numerical 속성은 많은 수의 binary 속성으로 변환된다.

2.2 서포트 집합의 선택

앞서 설명된 이진화 단계를 통하여 이진화된 데이터 집합을 $B^+, B^- (B^+ \cap B^- = \varnothing)$라 할 때 서포트 집합은 $B^+ \cap B^- = \varnothing$ 조건을 계속 성립시켜 주는 binary 속성집합의 부분집합으로 정의가 된다. $B^+ (B^-)$에 포함된 모든 binary 속성의 수를 n, $B_{ij}^+ (\in \{0, 1\})$를 B^+의 i번째 데이터의 j 번째 속성값이라 하자. 또한 $a_k^{(i,j)}$를 다음과 같이 정의 할

$$a_k^{(i,j)} = \begin{cases} 1 & B_{ik}^+ \neq B_{jk}^-, \\ 0 & otherwise. \end{cases}$$

때 서포트 집합은 아래의 set covering problem(SCP)에 의해 선택될 수 있다.

$$\min \sum_{k=1}^{n} x_k$$

$$s.t. \sum_{k=1}^{n} a_k^{(i,j)} x_k \geq 1, \; i = 1, \ldots, m^+, j = 1, \ldots, m^-$$

$$x_k \in \{0, 1\}, \quad k = 1, \ldots, n$$

SCP는 현재 연구가 잘 진행된 분야로써 최적해 또는 그에 근접한 해를 찾아낼 수 있는 많은 수의 휴리스틱 알고리즘들이 개발되어있다. 하지만 위 문제의 해의 질이 LAD의 분류능력에 크게 영향을 주지 않으므로 주로 간단한 그리디(greedy) 휴리스틱을 주로 사용하여 문제를 푼다. LAD에서 서포트 집합의 선택 단계는 패턴생성 단계에서 고려해야 될 binary 속성의 수를 많이 줄여 줌으로써 패턴생성을 훨씬 쉽고 용이하게 만들어 준다는 측면에서 중요한 단계라 할 수 있다.

2.3 패턴 생성

$a_1, a_2, \ldots, a_n$의 n개의 binary 속성을 가지는 이진화된 데이터 집합(B^+, B^-)에 대해 임의의 binary 포인트가 속성 $a_i (i = 1, \ldots, n)$에 대해 1의 값을 가지는 경우를 x_i, 그렇지 않은 경우를 $\overline{x_i}$로 나타내자. 이때 위의 $x_i, \overline{x_i}$를 리터럴(literal)이라 부르고, 하나 이상의 리터럴의 conjuction으로 만들어진 것을 텀(term)이라 한다. 즉, $I = \{1, \ldots, n\}$라 할 때 $P \cap N = \varnothing$를 만족하는 I의 부분집합 P, N에 대해 텀 t를 다음과 같이 나타낼 수 있다.

$$t = \wedge_{i \in P} x_i \wedge_{i \in N} \overline{x_i}$$

텀에 포함된 리터럴의 수를 디그리(degree)라 부르고 임의의 binary 포인트 $p = \wedge_{i \in P'} x_i \wedge_{i \in N'} \overline{x_i} \; (P^\dagger \cap N^\dagger = \varnothing, P^\dagger \cup N^\dagger = I)$에 대해 $P \subseteq P^\dagger$, $N \subseteq N^\dagger$를 만족할 때 $t(p) = 1$로 나타내고 t는 p를 커버(cover)한다고 한다. 임의의 텀이 B^+에 포함된 binary 포인트를 적어도 하나 커버하며 B^-에 포함된 binary 포인트를 하나도 커버하지 않을 때 그 텀을 양성패턴이라 정의한다. 음성패턴은 양성패턴에 대칭적인 정의를 가지며 양성패턴은 아래의 열거법에 기반한 프로시저(procedure)에 의해 생성될 수 있다.

procedure *Patt_Enu*
begin
 $P := \varnothing$
 $C_0 := \varnothing$
 for $d := 1$ **to** D **do**
  if $d < D$ **then**
   $C_d := \varnothing$
  end if
  for $T \in C_{d-1}$ **do**
   $p :=$ maximal index of literal in T
   for $s := p + 1$ **to** n **do**
    for $l_{new} \in \{l_s, \overline{l_s}\}$ **do**
     $T_{new} = T \wedge l_{new}$
     for $i := 1$ **to** d **do**
      $T_{new}^\dagger := T_{new}$ with i^{th} literal dropped
      if $T_{new}^\dagger \notin C_{d-1}$ **then**
       goto ※
      end if
     end for
     if $1 \in T_{new}(B^+)$ **then**
      if $1 \notin T_{new}(B^-)$ **then**
       $P := P \cup \{T_{new}\}$
      else if $d < D$ **then**
       $C_d := C_d \cup \{T_{new}\}$
      end if
     end if

```
    ※
  end for
 end for
end for
end for
end
```

위에서 제시된 패턴생성 방법으로는 n개의 binary 속성을 가지는 데이터 집합에서 디그리 d의 패턴을 생성할 경우 $2^d\binom{n}{d}$개의 텀을 고려해야 된다는 문제점이 있다.

2.4 구별자의 생성

이전 단계에서 생성된 양성패턴을 $P_1, ..., P_r$, 음성패턴을 $N_1, ..., N_s$라 할 때 LAD의 구별자는 다음과 같이 구현될 수 있다.

$$\Delta = \sum_{k=1}^{r} w_k^+ P_k + \sum_{l=1}^{s} w_l^- N_l$$

이때 각 패턴의 가중치(w_k^+, w_l^-)는 여러 가지 방법에 의해서 결정되어 질 수 있으며 선형계획법(linear programming)이나 패턴의 포인트에 대한 coverage를 이용하는 방법 등이 있다. 새로운 포인트를 위 LAD 구별자에 대입하여 그 값이 0 보다 클 때 그 포인트는 양성 포인트 그렇지 않으면 음성 포인트라고 판단할 수 있다.

3 최적화에 근거한 패턴생성 방법

LAD의 처음 두 단계에 의해서 뽑혀진 이진화된 데이터 집합을 $\bullet \in \{+, -\}$에 대해 $B^\bullet$라 하고 n개의 binary 속성을 가진다고 가정하자. 또한 집합 $J^\bullet$를 $B^\bullet$에 포함된 binary 포인트의 인덱스 집합으로 정의하고 $\overline{\bullet}$를 $\bullet$의 반대(negation)집합으로 가정할 때 $B^\bullet$에 속하는 i번째 포인트$(B_i^\bullet)$를 적어도 하나 커버하는 $\bullet$ 패턴을 생성하는 문제는 다음에 소개되는 것과 같이 모델화 될 수 있다.

3.1 MILP에 기반한 방법

$$\min \sum_{j \in J^\bullet \setminus \{i\}} y_j$$

$$s.t. \quad \sum_{k=1}^{n} x_k = d$$

$$\sum_{k=1}^{n} \bar{a}_k^{(i,j)} x_k + y_j \geq d, \qquad j \in J^\bullet \setminus \{i\}$$

$$\sum_{k=1}^{n} \bar{a}_k^{(i,j)} x_k \leq d-1, \qquad j \in J^{\overline{\bullet}}$$

$$x_k \in \{0,1\}, \qquad k=1,...,n$$

$$0 \leq y_j \leq n, \qquad j \in J^\bullet \setminus \{i\}$$

우선 결정변수 x_k는 i번째 포인트의 k번째 리터럴이 패턴에 포함이 될지 안 될 지를 나타내며 y_j는 j번째 포인트가 패턴에 의해 커버가 되는지 안 되는 지를 나타낸다. 패턴에 의해 j번째 포인트가 커버가 되면 y_j는 0이 되고 목적함수가 y_j의 합을 최소화 하는 것이므로 결국 커버하는 포인트의 수를 최대화 하는 패턴을 찾는 모델이 된다. 또한 결정변수 x_k의 정의상 i번째 포인트는 무조건 커버가 된다. 첫 번째 제약식은 패턴의 디그리가 d가 되어야 한다는 것이며 두 번째 제약식은 패턴이 어떤 $\bullet$ 포인트를 커버하는 경우 패턴과 일치하는 리터럴의 수는 적어도 패턴의 디그리가 되어야 하며 이때 그 포인트에 대한 y_i는 y_i의 합을 최소화 시키는 문제이므로 0이 된다. 마지막 제약식은 $\overline{\bullet}$ 포인트에 대해서는 하나라도 커버하지 말아야 한다는 제약식이며 이때 그 포인트와 일치하는 패턴의 리터럴 수는 패턴의 디그리 미만이 되어야 한다는 조건을 나타내고 있다. 또한 보통 실제의 데이터 집합들은 어느 정도의 잡음(noise) 데이터들을 가지고 있는데 이 잡음 데이터들은 분류오류(classification error), 측정오류(measurement error), 속성 값의 분실(missing attribute values) 등에 의해 발생이 된다. 이중 분류오류에 의한 잡음 데이터들을 고려하기 위하여 위 MILP 모델은 아래와 같이 수정될 수 있다.

$MILP_i^\bullet$:

$$\min \sum_{j \in J^\bullet \setminus \{i\}} y_j$$

$$s.t. \quad \sum_{k=1}^{n} x_k = d$$

$$\sum_{k=1}^{n} \bar{a}_k^{(i,j)} x_k + y_j \geq d, \qquad j \in J^\bullet \setminus \{i\}$$

$$\sum_{k=1}^{n} \bar{a}_k^{(i,j)} x_k - z_j \leq d-1, \quad j \in J^{\overline{\bullet}}$$

$$\sum_{j \in J^{\overline{\bullet}}} z_j \leq \alpha m^{\overline{\bullet}}$$

$$x_k \in \{0,1\}, \qquad k=1,...,n$$

$$0 \leq y_j \leq n, \qquad j \in J^\bullet \setminus \{i\}$$

$$0 \leq z_j \leq 1, \qquad j \in J^{\overline{\bullet}}$$

위 모델에서는 분류오류를 고려하기 위하여 반대되는 데이터들을 어느 정도까지 커버하는 패턴들도 고려될 수 있도록 하였다. α값은 미리 주어지는 값으로써 실험에 의하여 결정되었다.

3.2 SCP에 기반한 방법

집합 $O_{ik}^\bullet$를 $\{j=1,...,m^\bullet | B_{ik}^\bullet = B_{jk}^\bullet\}$로 정의하고 목적함수의 계수 c_k를 $m^\bullet / |O_{ik}^\bullet|$로 정의할 때 패턴생성 문제는 다음과 같은 SCP로 나타낼 수 있다.

$SCP_i^{\bullet}$:

$$\min \sum_{k=1}^{n} c_k x_k$$

$$s.t. \sum_{k=1}^{n} a_k^{(i,j)} x_k \geq 1, \quad j \in J^{\overline{\bullet}}$$

$$x_k \in \{0,1\}, \quad k=1,...,n$$

결정변수 x_k는 먼저 소개된 MILP 모델에서와 같은 의미를 가진다. 따라서 위 모델도 결정변수의 정의상 i번째 포인트는 어떤 해가 나오더라도 항상 커버가 된다. 제약식은 $\overline{\bullet}$ 포인트를 커버하지 말아야 한다는 조건을 나타낸다. 위 모델에서는 MILP 모델에서 변수 y_j를 통해 고려되었던 패턴이 커버하는 데이터 포인트의 수를 목적함수의 계수 c_k를 통하여 고려하고 있으며 주어진 목적함수에서는 최소의 디그리를 가지는 패턴을 생성하도록 하였다. 이는 [2]에서 언급되어 있듯이 생성된 패턴이 단순할 수 록 LAD의 분류 성능이 향상되는 현상을 반영한 것이며 필요 이상으로 복잡한 구별자를 만들 필요가 없다는 Occam's razor에도 부합되는 것이라 할 수 있다. 또한 SCP 모델의 경우 분류오류에 의한 잡음 데이터들을 고려하기 위하여 해를 구하기 위하여 사용되는 그리디 휴리스틱을 모든 제약식이 만족되지 않더라도 일정 수준이상 만족시킬 경우 종료하도록 수정을 하여 MILP에서와 같이 반대되는 포인트를 어느 정도 커버하는 패턴들도 고려가 되도록 하였다.

3.3 패턴생성 프로시저

a_k를 k번째 리터럴을 위한 문자 리터럴이라 할 때 위에서 소개된 모델의 해 $\mathbb{X}$를 통하여 생성되는 패턴 p는 다음과 같이 나타낼 수 있다.

$$p = \wedge_{\substack{x_k=1, B_{ik}=1 \\ k=1,...,n}} a_k \wedge_{\substack{x_k=1, B_{ik}=0 \\ k=1,...,n}} \overline{a_k}$$

위에서 소개된 두 가지 모델을 이용하여 데이터 집합의 모든 포인트를 커버하는 숨겨진 패턴들을 찾기 위해 다음과 같은 프로시저를 생각해 볼 수 있다. 아래에서 집합 R_p는 패턴 p의 리터럴의 인덱스 집합으로 정의되고 M은 임의의 매우 큰 수를 나타낸다.

procedure *Patt_Opt*
begin
 for $\bullet \in \{+, -\}$ **do**
  $P^{\bullet} := \varnothing$
  $C^{\bullet} := \{1, ..., m^{\bullet}\}$
  $q_i := 0, \forall i \in \{1, ..., m^{\bullet}\}$
  while $C^{\bullet} \neq \varnothing$ **do**
   Choose $i \in C^{\bullet}$
   for $p \in P^{\bullet}$ **do**
    if $p(B_i^{\bullet}) = 1$ **then**
     for $k \in R_p$ **do**
      $c_k \leftarrow M$
     end for
    end if
   end for
   Formulate and solve $MILP_i^{\bullet}$ ($SCP_i^{\bullet}$)
   Generate a pattern p from a solution
   $P^{\bullet} \leftarrow P^{\bullet} \cup p$
   for $i \in C^{\bullet}$ **do**
    if $p(B_i^{\bullet}) = 1$ **then**
     $q_i \leftarrow q_i + 1$
    end if
   end for
   $C^{\bullet} \leftarrow C^{\bullet} \setminus \{i \in C^{\bullet} | q_i = r\}$
  end while
 end for
end

위 프로시저는 $\bullet$집합의 모든 binary 포인트를 적어도 r번 커버할 때 까지 패턴을 만들게 된다. 이때 같은 포인트에 대해서 같은 패턴이 나오는 것을 방지하기 위해 현재 대상이 되는 포인트를 커버하는 패턴들의 리터럴에 대해서는 모델의 목적함수의 계수를 M으로 바꾸었다. 모든 포인트를 적어도 r번 커버하는 이유는 [2]에서 언급되어 있듯이 LAD 구별자의 분류능력을 향상시키기 위해서이고 [2]에서는 SCP에 의해 고려되었다.

4 실험 결과

4.1 실험 데이터 및 방법

본 논문에서 고안된 패턴생성 방법의 능력을 기존의 방법과 비교하여 보여주기 위하여 [2]에서 사용된 것과 같은 실험 데이터 집합[5]과 실험방법을 사용하여 결과를 비교 하였다. 첫 번째로 이전 제시된 방법으로 생성된 패턴으로 만들어진 구별자의 분류 정확도를 [2]에서 제시된 정확도와 비교를 하였으며 두 번째로 기존 패턴생성 방법으론 불가능한 서포트 집합의 선택 없이 데이터 이진화 후 직접 패턴생성 방법을 적용시켜 실험을 하였다. 실험에 사용된 데이터 집합의 정보는 표1에 정리되어 있으며 학습 데이터의 비율은 50%로 하여 30번의 holdout 방법을 통하여 구별자의 정확도를 측정하

Dataset	Number of		
	classes	attributes	points
Wisconsin breast cancer (WBC)	2	9	683
Cleveland heart disease (CHD)	2	13	297
Pima Indian diabetes (PID)	2	8	768
credit card scoring (CCS)	2	15	653
Boston housing (BH)	2	13	506
congressional voting (CV)	2	16	435

표1 데이터 집합 정보

였다. 또한 [2]의 실험과는 달리 분실 속성 값을 가지는 데이터 포인트는 모두 실험에서 제외 되었으며 Boston housing[5] 데이터 집합의 경우 [6]에서는 2개의 클래스를 가진 분류문제로 간주하고 있으며 [4]에서는 3개의 클래스를 가진 분류문제로 간주하고 있으나 비교대상인 [2]에서는 [6]에서와 같이 간주하고 있으므로 본 논문에서도 2개의 클래스를 가진 문제로 간주 하였다.

4.2 LAD의 구현 방법

우선 위에서 제시된 패턴생성 방법이 포함된 LAD의 구현은 Intel 포트란 8.0을 이용하였으며 실험은 리눅스 PC(Intel 2.66 GHz, 512RAM)에서 이루어졌다. 서포트 집합의 선택과 패턴생성을 위한 SCP 문제의 경우 간단한 그리디 휴리스틱을 구현하여 풀었으며 패턴생성을 위한 MILP의 경우 CPLEX 9.0 이용하여 해를 구하였다. 그리디 휴리스틱에서 사용된 룰은 집합 I_j를 열 j에 의해 커버되는 행의 인덱스 집합으로 정의하고 집합 M_u를 현재의 해로 커버가 되지 않는 행의 인덱스 집합으로 정의할 때 다음과 같다.

$$j^* \leftarrow argmax\{j=1,...,n : |I_j \cap M_u|\}$$

4.3 수치 결과

우선 표2에서는 제시된 패턴생성 방법으로 만들어진 LAD 구별자의 분류 정확도를 기존의 패턴생성 방법과 비교를 하였다.

Dataset	MILP	SCP	기존방법
WBC	96.1±0.9	96.6±0.9	96.9±0.9
CHD	80.4±3.2	82.1±2.8	82.3±1.7
PID	74.4±1.8	74.8±1.6	71.9±1.9
CCS	86.5±1.4	86.3±1.6	85.4±1.2
BH	84.5±2.3	84.6±2.1	84.0±1.6
CV	95.2±1.1	95.2±1.1	96.2±1.1

표2 분류 정확도 비교

분류 정확도에 있어서는 CHD 데이터 집합에서 MILP에 의한 패턴생성법이 약간 낮게 나왔고 PID 데이터 집합의 경우에는 기존방법이 본 논문에서 제안된 방법보다 낮게 나온 것을 제외하고는 제안된 방법과 기존방법 사이에 통계적으로 유의한 차이가 거의 없는 듯 보였다. 그러나 [2]에서 실행된 실험에서는 데이터 집합의 불확실성을 가만하기 위하여 분실된 속성 값을 가지는 데이터 포인트를 모두 실험에 포함키고 분류 오류 및 측정 오류까지 모두 고려하였고 본 논문에서 실행된 실험에서는 분류 오류만을 고려하였음을 생각할 때 고무적인 결과라 할 수 있다. 표3에서는 서포트 집합의 선택 없이 모든 binary 속성을 이용할 경우 열거 가능한 텀의 수를 보여주고 있으며 이는 열거법으로론 서포트 집합의 선택 없이 패턴생성은 불가능 한 것임을 잘 나타내 주고 있다. 표4에서는 모든 binary 속성을 이용하여 SCP에 기반한 패턴생성의 정확도와 패턴생성에 걸린 시간을 보여주고 있다. 서포트 집합의 선택 없이도 정확도의 희생 없이 1초도 안되는 시간 안에 빠르게 효과적으로 패턴이 생성되었음을 잘 보여주고 있다.

Dataset	서포트 집합 선택		가능한 디그리 5 텀의 수
	전	후	
WBC	64	14	$2^5 \times 7.6 \times 10^6$
CHD	219	14	$2^5 \times 4.0 \times 10^9$
PID	561	37	$2^5 \times 2.6 \times 10^{11}$
CCS	503	23	$2^5 \times 1.2 \times 10^{12}$
BH	682	30	$2^5 \times 4.5 \times 10^{11}$
CV	48	15	$2^5 \times 1.7 \times 10^6$

표3 열거 가능한 텀의 수

Dataset	SCP	
	정확도	경과시간
WBC	96.65±0.75	0.02±0.01
CHD	82.73±2.10	0.01±0.01
PID	74.86±2.01	0.37±0.03
CCS	86.00±1.36	0.16±0.05
BH	85.14±2.13	0.27±0.03
CV	96.41±1.13	0.01±0.01

표4 SCP에 기반한 방법의 성능

5 결론

본 논문에서는 LAD에서의 패턴생성 문제를 MILP와 SCP로 모형화 하고 그것을 이용한 구체적인 패턴생성 프로시저가 소개되었다. 실험을 통하여 제안된 방법이 기존의 열거법에 기반한 방법의 분류 능력을 저하시키지 않으면서 열거법의 계산상의 한계점을 보완해 효과적이고 빠르게 패턴을 생성시켰음을 증명하였다. 이는 LAD를 통한 데이터 분석을 보다 실용적이고 효율적으로 개선시키는 결과를 가져다 줄 것이다.

참고문헌

[1] E. Boros, P.L. Hammer, T. Ibaraki, and A. Kogan. Logical Analysis of Numerical Data. *Mathematical Programming*, 79:163-190, 1997.

[2] E. Boros, P.L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik. An Implementation of Logical Analysis of Data. *IEEE Transactions on Knowledge and Data Engineering*, 12:292-306, 2000.

[3] Y. Crama, P.L. Hammer and T. Ibaraki. Cause-Effect Relationships and Partially Defined Boolean Functions. *Annals of Operations Research*, 16:299-326, 1988.

[4] T.S. Lim and W.Y. Loh. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. *Machine Learning*, 40:203-228, 2000.

[5] P.M. Murphy and D.W. Aha. Uci repository of machine learning databases: Readable data repository. Department of Computer Science, University of California at Irvine, CA, 1994. Website at http://www.ics.uci.edu/mlearn/MLRepository.html.

[6] S.K. Murphy and D.W. Aha. A system for induction of oblique decision trees. Artificial Intelligence Research, 2:1-32, 1994.