정보이론을 이용한 연속패턴생성방법

이창환, 이소민°
동국대학교 정보통신공학과
(chlee@dgu.edu, somin7@dgu.edu)

An Information-Theoretic Method for Sequential Pattern Analysis

Chang-Hwan Lee, So-Min Lee°
Dept. of Information & Communications Engineering, Dongguk University

요 약

시차를 두고 발생한 사건속에서 잠재해있는 패턴을 발견하는 연속패턴(sequential pattern) 생성기술은 데이터 마이닝 분야에서 최근 관심을 모으고있는 분야이다. 본 연구는 정보이론을 이용하여 데이터베이스로부터 연속패턴을 자동으로 발견하는 방법에 관한 내용이다. 본 연구에서 제시하는 방법은 기존의 방법과는 달리 테이블내의 모든 속성간의 연속패턴 관계를 탐지할 수 있으며 헬링거(Hellinger) 변량을 이용하여 발견된 연속패턴들의 중요도를 측정할 수 있다. 또한 헬링거 변량의 함수적인 특성을 분석하여 연속패턴 추출의 복잡도를 줄이기 위한 두 가지의 법칙이 제안되었다.

1. 서 론

주어진 데이터베이스로부터 전문가 수준의 지식을 자동으로 추출하여 사용할 수 있다면 대단히 유용할 것이다. 본 연구는 데이터 마이닝의 기술 중에서 연속패턴의 생성에 대한 연구로서 시계열 데이터로부터 잠재해있는 연속패턴을 정보이론을 이용하여 IF-THEN 의 형식으로 자동 생성하는 기술에 관한 연구이다. 연속패턴의 생성에 대한 연구는 현재 몇 가지의 방법이 제안되어 있으며 상용화되어있는 알고리즘도 있다. 이들 중에서 현재 가장 널리 사용되고있는 연속패턴의 방법은 IBM 에서 개발한 GSP 알고리즘[1] 이다. GSP 알고리즘은 역시 IBM 에서 개발한 연관(association) 알고리즘 [2] 의 기능을 확장하여 연관패턴을 생성할 수 있게 하였다.

GSP 알고리즘의 연속패턴 생성기술은 고객이 제품을 구입한 데이터를 분석하여 각 제품의 구매형태가 시차에 따라서 어떠한 패턴을 보이는지를 발견하는 기술로서 이 기술들은 구체적으로 고객의 구매 기록 데이터를 분석하여서 다음과 유사한 법칙들을 생성한다.

IF 제품=TV, THEN 제품=VCR

이 법칙의 의미는 TV를 구매한 고객은 추후에 VCR을 구입한다는 의미이다. 기존의 연속패턴생성의 방법들은 IF 부분에 오직 한 개의 속성값을 허용하며 THEN 부분에는 IF 부분에서 사용된 속성만이 사용될 수 있다. 또한 이 계열의 많은 알고리즘들은 시간창(time window)을 두어서 주어진 시간간격 이내에 발생한 데이터만을 고려하여 법칙을 생성한다.

하지만 이 방법은 전체 데이터 중에서 특정한 한 개의 속성(대부분 제품)에 대해서만 시간에 따라 발생하는 연속패턴을 탐색할 수 있어서 많은 수의 제품을 취급하는 백화점이나 유통업체 등의 제한된 부분에서만 여러 제품들의 구매형태가 시차에 따라 어떤 패턴을 보이는지를 조사할 수 있다. 또한 한 개의 속성에 대해서만 연속패턴을 탐지할 수 있기 때문에 그 속성에 영향을 주는 다른 속성의 영향을 전혀 측정할 수 없다는 것이 문제점이다. 이와 같이 기존의 연속패턴 방법은 판매 제품의 종류가 작을 경우에 이들 제품간의 판매 연관성보다는 이들 제품을 각각 구입하게 하는 다른 속성의 원인 분석이 더욱 중요한 정보가 될 것이며 따라서 한 개의 속성 내에서의 연속패턴을 제공하는 기능은 소수 제품의 환경에서는 거의 의미를 찾을 수 없다.

본 연구에서 제시하는 연속패턴 생성 방법은 테이블내의 모든 속성들의 값에 대하여 서로의 연속패턴 연관관계를 계산할 수 있다. 따라서 훨씬 중요한 의사결정의 정보를 제공하며 훨씬 다양한 분야에 대하여 적용할 수 있다. 따라서 본 논문의 방법에서 생성하는 법칙은 훨씬 광범위한 정보를 제공하며 훨씬 많은 응용 범위를 가진다.

2. 본 연구의 내용

본 연구에서 제시하는 방법은 데이터를 읽어서 다음과 같은 형식의 연속패턴 법칙을 생성하는 내용이다. 예를 들어 B_{i_1}, B_{i_2}, B_{i_3} 을 테이블의 속성이라고 하고 b_{j_1}, b_{j_2}, b_{j_3} 를 각 속성에 대한 임의의 값으로 가정하자. 이때 본 알고리즘에서 생성되는 법칙은 다음과 같은 형식으로 표현된다.

$$\text{IF } B_{i_1}=b_{j_1} \wedge B_{i_2}=b_{j_2} \wedge B_{i_3}=b_{j_3} \text{ ;··· THEN } A=a_k$$

이 법칙의 의미는 IF 부분에 있는 $B_{i_{1-k}} \wedge B_{i_{2-k}} \wedge B_{i_{3-k}}$ ·· 의 조건을 만족하는 행위가 발생하였을 때 그 후에 $A=a_k$를 만족하는 행위가 발생하는 패턴이 있다는 의미이다.

이와 같은 연속패턴 법칙 생성의 기본적인 가정은 IF 부분의 조건이 THEN 부분의 확률 분포에 영향을 준다는 가정에서 출발한다. 직관적으로 말하면, 특정 속성(attribute)의 값이 정해질 때 목표 속성(target attribute)의 확률 분포를 현저히 변화시킨다면 이는 특정 속성의 값을 결정하는 중요한 역할을 의미한다. 따라서 시스템은 우선 속성의 변수 값(예를 들어서 속성 B 가 b 의 값을 가짐)을 우선 선택하고, B=b 이라는 사건이 목표 속성 A 의 분포 값에 어떤 영향을 끼치는 가를 점검한다. 만약 그것이 목표 속성의 확률 분포에 상당한 영향을 끼친다면 시스템은 다음과 같은 규칙이 있음을 가정한다.

IF B = b, THEN A = a with H

각 규칙은 각 규칙의 중요도를 표현하는 H 값을 포함하고 있다. 여기서 THEN 부분에는 오직 한 개의 속성 값이 나타나도록 하였고 규칙의 왼쪽 IF 부분은 논리곱(conjunction)의 형태로 되어 있다. 또한 논리합(logical OR)과 부정 논리(logical not)의 형태는 고려하지 않는다.

본 연구에서는 이와 같은 목표 속성의 확률분포의 변화 정도를 측정하기 위하여 헬링거(Hellinger) 엔트로피 함수 [3] 를 사용하였다. 본 연구의 연속패턴 환경에서 위와 같은 법칙의 경우 헬링거 함수는 다음과 같이 정의된다.

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

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

연속패턴 생성의 법칙에서 고려해야할 사항은 목표 속성의 값 중에서 가장 빈번히 나타나는 하나의 값은 규칙의 오른 편에서 나타나고 다른 값들은 확률 $1-P(a)$ 에 포함되어지는 것이다. 다시 말해 규칙의 정확도를 측정하는 데에 있어서 중요한 점은 특정 데이터가 규칙의 오른 편의 목표 값(A=a)에 속하는가를 검사하는 것이다. 예를 들어서 목표 속성 A 가 k 개 (a_1, a_2, ... , a_k) 의 값을 가지고 있고 연속패턴규칙은 오른 편에서 A=a_1 이라고 가정하자. 그러면 속성 A 의 확률 분포가 ($P(A=a_1)$, $P(A \neq a_1)$) 와 같이 2 진의 확률 분포로 변환된다. 그래서 수식 (1) 은 다음과 같이 변환될 수 있다.

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

이 식에서 $P(a|b)$ 는 $B=b$ 라는 조건 하에서 $A=a$ 의 조건 확률을 의미한다. 구체적으로 얘기하면 먼저 THEN 부분에 나타나는 목표 속성에 대한 이전확률분포 분포를 구한 다음 IF 부분의 조건을 만족한 상태에서의 목표속성에 대한 확률분포인 이후확률분포를 계산한다. 이후확률분포는 IF 부분의 행위가 THEN 부분의 행위보다 먼저 시행된 데이터의 개수만 고려를 하여 계산한다. 이와 같은 방식으로 이전확률분포와 이후확률분포를 계산한 후에 이들이 서로 얼마나 상이한가의 정도를 헬링거 엔트로피 함수를 사용하여 계산한다. 이와 같이 계산된 엔트로피 함수의 값이 해당 법칙의 정확도를 의미한다.

다음과 같은 연속패턴 법칙에 대한 정확도를 계산한다고 가정하자. IF A=a ∧ B=b, THEN C=c

이 경우 속성 C 의 이전확률분포를 계산하는 것은 법칙생성 알고리즘의 경우와 동일하다. 하지만 이후확률분포를 계산할 때 본 알고리즘은 전체 데이터 중에서 조건 A=a ∧ B=b 를 만족하는 각 데이터마다 해당 데이터의 행위자(보통 고객임) 데이터가 끝날 때까지의 잔여 레코드 중에서 조건 A=a ∧ B=b 를 만족하는 모든 레코드를 원래의 데이터에 추가되는 것으로 가정하고 이렇게 수정된 데이터의 분포를 이용하여 이후확률분포를 계산한다.

구체적인 예를 들어서 <그림 1> 의 경우 다음의 법칙에 대한 정확도를 계산해보자.

IF 고객=C1 ∧ 제품=P1, THEN 제품=P2

이 법칙의 목표속성은 제품이며 제품에 대한 이전확률분포는 쉽게 계산할 수 있다. 그 다음으로 이후확률분포를 계산할 때 먼저 IF 부분의 조건 고객=C1 ∧ 제품=P1 을 만족하는 레코드는 (1), (3), (7) 임을 알 수 있다. 따라서 레코드 (1) 에 의하여 레코드 (3) 과 (7) 이 원래의 데이터에 추가되는 것으로 가정하며 레코드 (3) 에 의하여 레코드 (7) 이 추가되는 것으로 가정한다. 즉 레코드 (7) 은 이 법칙에 대하여 3 번 추가된다. 이와 같이 수정된 데이터의 분포를 이용하여 이후확률분포를 계산한다. 그리고 이러한 이전확률분포와 이후확률분포의 차이를 헬링거 함수로 측정한 값이 위의 법칙에 대한 정확도가 된다.

연속패턴 생성에서 우리가 고려해야할 또 다른 문제는 규칙이

고객	일자	제품		
C1	D1	P1		(1)
		P4		(2)
	D2	P1		(3)
		P2		(4)
		P3		(5)
	D3	P2		(6)
		P1		(7)
...	...	...	...	

<그림 1 : 연속패턴 생성의 예제>

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 본 시스템에서는 다음 수식을 사용하였다.

$$\sqrt{P(a)} \quad (3)$$

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.

정의 1 : 다음과 같은 규칙

$\bigwedge_i B_i = b_{ij} \rightarrow A = a_k$의 H 값은 다음과 같이 정의된다.

```
0.
1.  Input: IF B1=b1 & B2=B2 & . . . & Bk=bk, THEN A=a
2.  Output: posterior prob. distribution of attribute A
3.
4.  read entire data file D and calculate prior distribution of attribute A;
5.  for each record R in D do
6.     if R satisfies B1=b1 & B2=B2 & . . . & Bk=bk
7.        then
8.           PERSON := the value of attribute person of this record;
9.           do
10.             if the current record satisfies B1=b1 & B2=B2 & . . . & Bk=bk
11.                then recalculate the posterior dist. of attribute A
12.                        using the current value of A;
13.             read next record in D;
14.          while (PERSON = the value of attribute person)
15.       else
16.          continue;
17.    end-if
18. end-for
19. return (the posterior prob. distribution of attribute A)
```

<그림 2 : 연속생성에서의 이후확률분포 계산코드>

$$\sqrt{P(\bigwedge_i B_i = b_{ij})}\left[\left(\sqrt{P(A=a_k|\bigwedge_i B_i=b_{ij})}-\sqrt{P(A=a_k)}\right)^2+\left(\sqrt{(1-P(A=a_k|\bigwedge_i B_i=b_{ij})}-\sqrt{(1-P(A=a_k)}\right)^2\right]$$

위의 식에서 B_i 는 i 번째 속성을 나타내며 b_{ij} 는 속성 B_i 의 j 번째 값이다.

3. 연속패턴생성의 방법

연속패턴생성 알고리즘을 수행하기 위해서 데이터는 <그림 1>과 같이 행위자(제품 판매의 경우에는 고객)와 시간 및 목적물(제품 판매의 경우에는 제품)의 속성을 포함하고 있어야 한다. 또한 알고리즘의 수행을 위해서 데이터는 행위자와 시간의 순서대로 정렬(sorting)을 시켜야하며 연속형 속성은 이산 속성(discrete attribute)의 형태로 변환되어야한다. 본 논문의 연속패턴생성 알고리즘의 전체적인 기능은 위와 같이 정리된 데이터의 값들을 읽고서 k 개의 가장 의미있는 연속패턴법칙을 생성하는 알고리즘이다.

본 시스템에서 규칙을 생성하는 방법을 간략히 설명하면, 먼저, 규칙의 왼편이 한 개의 속성조건만을 갖는 단일조건(single-condition)규칙들을 생성한다. 알고리즘은 이들 단일 조건 규칙들에서 출발하여 가능한 왼쪽 면을 통한 깊이우선(depth-first) 탐색을 수행한다. 단일 조건들 중에서 H 계산값이 가장 높은 k 개의 규칙들이 규칙 리스트의 형태로 저장된다. 이들의 H 값중 가장 작은 H 값은 H_* 로 정의된다. 이 규칙 리스트의 각 원소에 대하여 알고리즘은 더욱 세분화된(specialized) 규칙들을 생성하려고 한다. 즉 알고리즘은 규칙 리스트에서 한 원소를 뽑고 추가적인 속성 조건을 왼편에 추가하여서 세분화시킨다. 알고리즘은 다음 절에서 설명하는 정리들 중에서 하나를 만족할 때까지 세분화를 계속한다. 이 과정을 더 이상 세분화시킬 법칙이 없을 때까지 반복하고 최종적으로 k 개의 가장 H 값이 높은 법칙을 출력한다.

3.1 H 값을 이용한 가지치기 방법

연속패턴 생성 시스템이 다루는 데이터는 속성들의 숫자가 증가함에 따라 계산해야할 규칙들의 총 숫자는 폭발적으로 증가한다. 예를 들어서 데이터가 r 개의 속성을 가지고 각 속성들은 v_i (i=1, ..., r) 의 값들을 가진다고 가정하자. 그러면 최악의 경우 총 규칙의 개수는 다음과 같이 주어진다.

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

우리는 헬링거 함수의 특성들을 이용한 가지치기 기술을 제시한다. 우선 다음과 같은 규칙을 가정하자.

IF $B = b$, THEN $A = a$ (4)

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

IF $B = b \wedge C = c$, THEN $A = a$ (5)

공식 (6)과 (7) 에서의 규칙들을 각각 Rg 와 Rs 라고 하자. 규칙 Rg 와 Rs 의 H 값을 각각 H_g 와 H_s 라고 가정하면 이들은 다음과 같이 정의된다.

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

$$=\sqrt{P(b)}[2-2\sqrt{P(a|b)P(a)}-2\sqrt{(1-P(a|b))(1-P(a))}] \quad (6)$$

$$H_s = \sqrt{P(bc)}[2-2\sqrt{P(a|bc)P(a)}-2\sqrt{(1-P(a|bc))(1-P(a))}]$$

$$=\sqrt{P(a|b)}\sqrt{P(b)}[2-2\sqrt{P(a|bc)P(a)}-2\sqrt{1-P(a|bc))(1-P(a))}] \quad (7)$$

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

정리 1 : 목표 속성의 클래스 개수를 m 이라고할 때 H_s 값은 다음의 경계값을 초과할 수 없다.

$$H_s \le \max\{\ \frac{\sqrt{P(a|b)}\sqrt{P(b)}[2\sqrt{m}-2\sqrt{P(a)}]}{2\sqrt{P(a)}-\sqrt{(1-P(a|b)}\sqrt{P(b)}[2\sqrt{P(a)}+2\sqrt{(1-P(a)}]}\}$$

정리 2 : 만약 조건 확률 R_s 가 1 이라면 R_s 의 H 값은 R_g의 H 값을 초과할 수 없다.

IF P(a|b) = 1, $H_s \le H_g$

이 법칙들은 속성 C 에 관한 추가정보없이 H_s 값의 경계를 예상하게 할 수 있게 한다. 따라서 만약 H_s 값의 경계값이 현재의 최소 H 값(H^*) 보다 작다면 현재의 규칙에 대하여서는 시스템은 더 이상 진행할 특수한 규칙들을 생성할 필요가 없다.

또한 추가로 사용되는 가지치기 방법은 현재 진행중인 규칙의 조건 확률이 1 이라면, 정리 2 에 의하여 시스템은 더 이상 특수한 규칙을 생성할 필요가 없음을 알 수 있다.

4. 실험결과

본 연구의 방법은 C++ 언어를 이용하여 구현되었으며 실험의 내용은 다음과 같다. 본 연구의 실험을 위하여 사용한 데이터는 <그림 3>와 같은 내용을 포함하고있다. 이 데이터는 어느 유통업체의 거래데이터로 일부분을 발췌하여 사용하였으며 또한 제품의 숫자를 조금 축소화하였다. <그림 3>은 전체 데이터의 일부분으로 시간 속성은 제품 구입 시간을 연속형 숫자, 장소는 제품을 구입한 장소의 코드로, 판매사원도 코드로 표시하였다. 또한 제품 속성은 구입한 제품ID를 의미하고 있다.

이와 같은 데이터를 이용하여 본 알고리즘을 수행한 결과 생성된 연속패턴 중에서 가장 상위의 10 법칙을 <그림 4>에서 보여준다. 법칙1의 경우는 E14사원에게 I9제품을 구입하는 고객은 나중에 I12를 구입한다는 의미이다. 이는 기존의 연속패턴 생성 알고리즘에서 제공하는 기능을 포함하고 있음을 알 수 있다. 또한, 법칙10 의 경우는 E19사원에게 물건을 구입한 고객은(구입 제품의 종류에 관계없이)나중에 I40제품을 구입하는 경향이 있음을 보여준다. 이러한 법칙도 본 알고리즘에서 생성하는 새로운 연속패턴의 종류이다.

5. 결론

본 연구는 시차를 가지고 발생하는 데이터에 잠재하고있는 연속패턴을 자동으로 탐지하여 법칙의 형태로 제공하는 방법을 정보이론의 헬링거 변량을 이용하여 제안하였다. 기존의 연속패턴 방법이 오직 한 개의 속성에 대해서만 연속패턴을 탐지하는데 비하여 본 연구에서 제시하는 방법은 여러 속성간의 연속패턴을 탐지하는 기능을 제공할 수 있었으며 이는 사용자에게 훨씬 많은 정보를 제공할 수 있다. 본 연구의 방법은 구현되어서 실험 데이터를 이용하여 실험하였으며 데이터 속에 잠재 하고 있는 연속패턴을 효과적으로 탐지할 수 있음을 알 수 있었다. 한가지 고려할 사항으로 본 연구는 알고리즘의 수행속도를 개선하기 위하여 두 가지 정리를 사용하여 가지치기 방법을 제공한다. 하지만 아직도 속성의 숫자가 아주 많은 데이터의 경우에는 수행속도를 더욱 감소시킬 수 있는 방법이 추가로 필요하며 이는 추후의 연구과제가 될 것이다.

고객번호	시간	장소	판매사원	제품
1	1	0	0	I3
1	1	0	0	I26
1	42	0	0	I76
1	42	2	1	I81
1	100	3	2	I3
1	100	3	2	I31
1	100	3	2	I42
...	...	...		...
2	2	3	2	I6
2	2	3	2	I15
...	...	...		...

<그림 3 : 연속패턴 데이터>

No.	IF	THEN	확률
1	사원= E14, 제품= I9	제품= I12	0.0025
2	사원= E6, 제품= I1	제품= I2	0.0025
3	사원= E12, 제품= I1	제품= I2	0.0025
4	사원= E24, 제품= I2	제품= I21	0.0025
5	사원= E20, 제품= I4	제품= I5	0.0024
6	지점= 사당, 제품= I4	제품= I5	0.0024
7	지점= 창동, 사원= E17, 제품= I26	제품= I31	0.0024
8	사원= E9, 제품= I40	제품= I1	0.0015
9	사원= E24, 제품= I9	제품= I81	0.0014
10	사원= E19	제품= I40	0.0009

<그림 4 : 연속패턴 생성의 결과>

참고문헌

[1] R.Agrawal and R.Srikant, *Mining sequential pattern*, Conf. Data Engineering(ICDE'95)

[2] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami, *Mining association rules between sets of items in large databases*. In Proc. of the ACM SIGMOD Conference on Management of Data, pages 207-216, Washington, D.C., May 1993.

[3] Beran, R. J., *Minimum Hellinger Distances for Parametic Models*, Ann. Statistics, Vol. 5, pp.445-463, 1977.

[4] H. Mannila, H Toivonen, and A.I. Verkamo, *Discovery of freque -nt episodes in event sequences*, Data Mining and Knowledge Discovery, 1998

[5] H.Lu, J.Han, and L.Feng., *Stock movement and n-dimensional inter-transaction association rules*, DMKD'98

[6] C.Bettini, X.Sean Wang, and S.Jajodia, *Mining temporal relation -ships with multiple granularities in time sequences*, Data Engineering Bulletin, 1998

[7] M.J.Zaki, N.Lesh, and M.Ogihara. PLANMINE: *Sequence mining for plan failures*, Conf. Knowledge Discovery and Data Mining (KDD'98)

[8] J.Han, G.Dong, and Y.Yin, *Efficient mining of partial periodic patterns in time series database*, Conf. Data Engineering (ICDE' 99)

[8] Chang-Hwan Lee, *Learning Inductive Rules Using Hellinger Measure*, Applied Artificial Intelligence, 13(8), 743-762, 1999

[10] J.Han, J.Pei, B.Mortazavi-Asl, Q.Chen, U.Dayal, and M.-C.Hsu. preespan, *Frequent pattern-projected sequential pattern mining*, Conf. Knowledge Discovery and Data Mining(KDD'00)

[11] B.-K. Yi,N. Sidiropoulos, T.Johnson, H.V.Jagadish, C.Faloutsos, and A.Biliris, *Online data mining for co-evolving time sequences, Conf. Data Engineering*(ICDE'00)

[12] M. A. Roytberg, *A search for common patterns in many sequences*, Computer Applications in the Biosciences, 8(1) 57-64, 1992.

[13] M. Vingron and P. Argos, *A fast and sensitive multiple sequence alignment algorithm*, Computer Applications in the Biosciences, 5 115-122, 1989.