빅 데이터 익명화 주요 이슈

장성봉*

*금오공과대학교 산학협력단
e-mail : sungbong.jang@kumoh.ac.kr

Sung-Bong Jang*
*Dept. of Industry-Academy Cooperation, Kumoh National Institute of Technology

요 약

빅데이터를 제 3 자에게 연구용으로 배포할 때, 개인정보 보호는 해결해야 할 중요한 이슈이다. 지금까지, 다양한 *k*-익명화 소프트웨어 도구 및 알고리즘들이 등장하여 매우 유용하게 사용되긴 하였지만, 이를 빅데이터에 그대로 적용할 경우, 분류 구성, 정보 손실, 처리시간 측면에서 좋지 못한 성능을 보여왔다. 본 논문에서 이러한 문제점과 주요 이슈들을 살펴본다.

1 들어가며

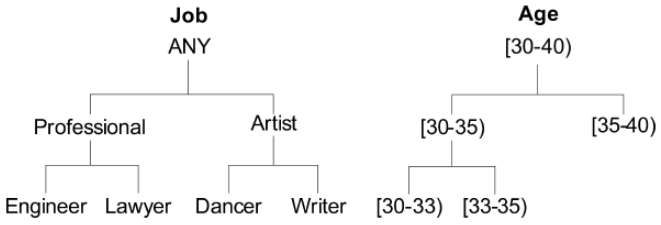
최근 들어 국내외 정부 기관이나 병원에서는 데이터를 연구 목적으로 제 3자에게 제공하는 경우가 많이 늘어나고 있으며, 제공되는 데이터 내부에는 질병정보나, 사회보장 번호, 주민번호와 같은 개인 프라이버 정보들이 많이 포함되어 있다. 이러한 프라이버시 정보들이 외부로 누출될 경우, 타인에 의해 악용될 소지가 있으므로, 아예 삭제가 되거나 또는 다른 데이터로 변경한 후 배포해야 한다. 하지만, 데이터를 삭제하더라도 준 식별자(quasi-identifiers, 데이터 베이스에서 조인연산의 기준이 되는 속성)속성에 해당하는 정보를 이용하면, 개인 프라이버시 정보를 쉽게 복구해 낼 수 있다. 이러한, 공격 형태를 속성 연결 공격(Attribute Linkage Attack) 또는 레코드 연결 공격(Record Linkage Attack)이라고 부른다[1][2]

이와 같은 공격을 방어하기 위해서 사용되는 방법이 데이터 익명화 기술이다. 익명화 기술중 가장 널리 사용되는 방법이 사마리티와 스위니가 개발한 *k*-익명성과 *l*-다양성 기술이다. *k*-익명성 방법에서는 기존의 준식별자에 해당하는 값을 좀더 일반화된 값으로 변경하여, 최소한 *k*개의 같은 레코드들이 존재하게 만드는 방법으로써, 조인 연산을 수행하더라도 동일한 레코드가 최소 *k*개 존재하게 하여, 레코드의 소유자가 누구인지 발견할 수 있는 확률을 1/k로 감소시키는 방법이다. 그러나, 민감 속성값이 모두 동일한 경우에는 *k*-익명성 기술을 적용하더라도, 레코드의 소유자가 누구인지 알 수 있다. 이러한 경우에 동일한 속성값을 다른 값으로 대체함으로써, 소유자가 누구인지 알 수 없도록 하는 기술이 마찬드라에 의해 제안한 *l*-다양성 기술이다. 현재까지 이 두 가지 기술을 응용한 파생기술과 알고리즘이 많이 연구되었으며, 해외에서는 이 기술에 기반한 소수의 익명화 소프트웨어 도구 등이 개발되어, 각 기관의 데이터베이스 관리자가 데이터 배포시에 구입, 활용하고 있다. 하지만, 기존의 익명화 도구들을 빅 데이터 익명화에 사용할 경우, 많은 문제점이 존재 한다. 본 논문에서 이와 관련된 문제점들을 소개한다.

2 빅 데이터 익명화 주요 이슈

첫째, 일반화 트리 구성의 문제점

분류 트리란 특정 속성의 데이터들을 공통된 정보로 계층화 시켜 일반화 시킨 트리를 말하며, 익명화 작업에 필수적인 데이터다. 직업과 나이에 대한 일반화 트리를 예로 들면 <그림 1>와 같다[1].



<그림 1> 일반화 트리 구성예

익명화 시스템에서는 실 데이터를 익명화 할 때, 위와 같은 일반화 트리를 참조하여 값을 변경한다. 따라서, 익명화를 위해서는 반드시 데이터베이스에 저장되어 있는 값을 기준으로 하여, 일반화 트리는 구성하여야 한다. 일반화 트리를 구성하는 방법에는 크게 두가지 방법이 있다. 첫 번째 방법은 실제 데이터베이스에 저장되어 있는 데이터 값을 모두 읽어가면서 일반화 트리를 구성하는 방법이 있고, 하둡(Hadoop)과 같은 빅데이터 처리용 서버에 요청하여 트리를 구성하는 방법이 있다. 이 두가지 모두 시간이 너무 많이 걸리고 만약, 실 데이터가 오류(error)나 누락(missing)된 필드가 존재할 경우, 트리 구성시 문제가 발생할 수 있다. 예를 들어, <표 3>과 같은 데이터가

있다 가정하자.

<표 1> 손실 데이터를 포함하고 있는 데이터

순번	이름	직업	나이
1	홍길동	변호사	34
2	김길동	치과의사	36
3	이길동	외과의사	38
.	.	.	.
89	데이비드	NULL	45
90	클라크	NULL	51
.	.	.	.

<표 1>을 보면, 89번 90번 환자의 경우, 데이터 값으로 NULL 값이 들어가 있다. 만약, 분류 트리를 구성하기 전에 이러한 데이터가 다량으로 포함되어 있을 경우, 구성된 분류 트리 값에 대한 신뢰도가 떨어지고 익명화에 대한 신뢰도도 마찬가지로 하락하게 된다. 따라서, 신뢰성 있고 시간이 적게 걸리는 분류 트리 구성방법이 필요하다. 특히, 대부분의 데이터베이스 관리자는 익명화에 대한 배경 지식이 전혀 없기 때문에, 직접 트리 구성 작업을 수행하기에는 많은 노력과 시간이 필요하다. 따라서, 일반화 트리 구성을 쉽게 할 수 있는 방법이 필요하다.

둘째, 정보 손실률 처리 문제

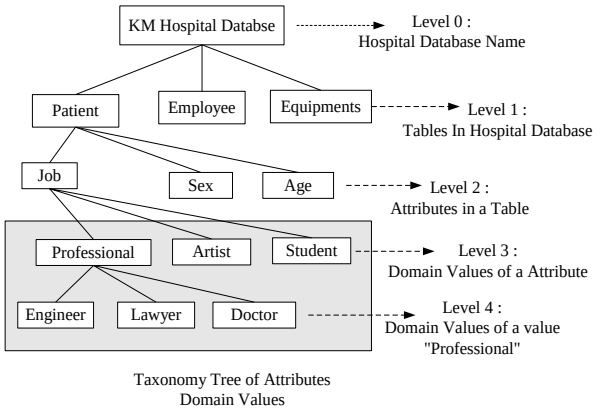
정보 손실률은 데이터 익명화를 통해 정보가 얼마나 많이 왜곡(distorted) 되었는지를 나타내는 지표로서, 아래와 같은 Xiao와 Tao가 제시한 공식을 가장 많이 사용한다[3].

$$InfoLoss(New\_Val) = \frac{Descendent(New\_Val) - 1}{Every(T_A)}$$

위 공식은 익명화를 통해 원래의 값에서 *New_Val*로 바뀐 값에 대한 정보 손실률을 계산하는 공식이다. 여기서, *New_Val*은 익명화를 위한 일반화 분류 트리상의 속성 데이터 값을 의미하며, *Descendent(New_Val)*은 분류 트리상의 후손 노드에 존재하는 속성 데이터 값을 의미 한다. 즉, 일반화를 통해 바뀌어진 속성 데이터 값이 몇 개인지를 나타내는 값이다. $Every(T_A)$는 *New_Val* 값을 가지는 테이블 속성 A가 가질 수 있는 값의 개수를 의미한다. 예를 들어, <그림 2>에서 "professional"에 대한 정보 손실률을 계산 해보자. <그림 2>를 보면, *Descendent(New_Val)*는 3이고 $Every(T_A)$는 값이 5 (job이 가지는 도메인 개수가 5)임을 알 수 있다. 따라서, 정보 손실률은, *InfoLoss* =(3-1)/5= 0.4 가 된다. 또한, 데이터베이스 내의 레코드와 테이블에 대한 정보 손실률은 아래의 공식으로 계산할 수 있다.

$$TabLoss(l) = \sum(w_i \times TabLoss(vg)), TabLoss(T) = TabLoss(l)$$

빅데이터 익명화 과정에서 *InfoLoss*값과 *TabLoss*값을 낮추기 위한 방법이 필요하다.



<그림 2>정보손실률 계산을 위한 트리 예제

셋째, 빅데이터 익명화 처리 시간 및 저장공간의 문제점을 들 수 있다. 빅 데이터 익명화 처리시, *k*-익명성 기술을 5,000개씩 행을 증가시키면서, 최대 30만개의 데이터에 대해 적용 해보았을 때, 처리시간이 지수적으로 증가한다. 만약, 백만 건 이상의 빅 데이터에 대해, 익명화를 적용할 경우, 며칠이 걸릴 수도 있음을 알 수 있다. 따라서, 익명화 처리 시간을 줄일 수 있는 방법이 필요하다. 데이터를 익명화하기 위해서 필요한 임시 저장 공간은 기본적으로 원시 테이블의 크기만큼 필요하다. 왜냐하면, 원시 데이터 베이스 테이블의 한 행(row)을 읽은 후, 준 식별자 속성값을 일반화 트리의 값으로 변경한 후, 이를 다시 익명화된 테이블에 저장해야 하기 때문이다. 하지만, 보통 빅데이터는 수 테라바이트 이상이기 때문에, 이를 모두 저장하기 위한 저장 공간이 충분하지 않은 경우가 대부분이다. 빅 데이터 익명화 도중 발생하는 많은 저장 공간을 줄이 위한 방법이 필요하다.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning(NRF-2016R1C1B1014346)

References

1. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information:*k*-anonymity and its enforcement through generalization and suppression. IEEE SRS P (1998)
2. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: *l*-diversity: privacy beyond k-anonimity. ACM TKDD. Vol.1, no.1, pp.1-52 (2007)
3. Fung, B.C.M., Wang, K., Chen, R., Yu, PS.: Handicapping attacker's confidence: An alternative to k-anonymization. Knowl. Inf. Sys. Vol.11, no.3, pp. 345-368 (2007)
4. Fung BCM, Wang K., Chen R., Yu PS. Anonymizing classification data for privacy preservation. IEEE TKDE. Vol.19., no.5, pp. 711-725(2007)