JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 26 No. 4, pp. 1-9, April 2021
https://doi.org/10.9708/jksci.2021.26.04.001

# Secure Training Support Vector Machine with Partial Sensitive Part

Saerom Park*

*Assistant Professor, Dept. of Convergence Security Engineering, Sungshin Women's University, Seoul, Korea

[Abstract]

In this paper, we propose a training algorithm of support vector machine (SVM) with a sensitive variable. Although machine learning models enable automatic decision making in the real world applications, regulations prohibit sensitive information from being used to protect privacy. In particular, the privacy protection of the legally protected attributes such as race, gender, and disability is compulsory. We present an efficient least square SVM (LSSVM) training algorithm using a fully homomorphic encryption (FHE) to protect a partial sensitive attribute. Our framework posits that data owner has both non-sensitive attributes and a sensitive attribute while machine learning service provider (MLSP) can get non-sensitive attributes and an encrypted sensitive attribute. As a result, data owner can obtain the encrypted model parameters without exposing their sensitive information to MLSP. In the inference phase, both non-sensitive attributes and a sensitive attribute are encrypted, and all computations should be conducted on encrypted domain. Through the experiments on real data, we identify that our proposed method enables to implement privacy-preserving sensitive LSSVM with FHE that has comparable performance with the original LSSVM algorithm. In addition, we demonstrate that the efficient sensitive LSSVM with FHE significantly improves the computational cost with a small degradation of performance.

▸ **Key words**: Privacy-preserving Machine Learning, Support Vector Machine, Homomorphic Encryption, Privacy, Secure Machine Learning

[요 약]

본 연구에서는 민감 정보가 포함된 경우의 서포트 벡터 머신 (SVM) 학습 알고리즘을 제안한다. 기계 학습 모형들이 실세계의 자동화된 의사 결정을 가능하게 하였지만 규제들은 프라이버시 보호를 위해서 민감 정보들의 활용을 제한하고 있다. 특히 인종, 성별, 장애 여부와 같은 법적으로 보호되는 정보들의 프라이버시 보호는 필수이다. 본 연구에서는 완전 동형암호를 활용하여 부분적인 민감 정보가 포함된 경우에 최소 제곱 SVM (LSSVM) 모형을 효율적으로 학습할 수 있는 방법을 제안한다. 본 프레임워크에서는 데이터 소유주가 민감하지 않은 정보와 민감한 정보 모두를 가지고 있고, 이를 기계학습 서비스 제공자에게 제공할 때에 민감 정보만 암호화해서 제공하는 것을 가정한다. 결과적으로 데이터 소유자는 민감 정보를 노출시키지 않으면서도 암호화된 상태로 모형의 학습 정보를 얻을 수 있다. 모형을 실제 활용할 경우에는 모든 정보를 암호화하여 안전하게 예측 결과를 제공할 수 있도록 한다. 실제 데이터에 대한 실험을 통해 본 알고리즘이 동형암호로 구현될 경우에 원래의 LSSVM 모형과 비슷한 성능을 가질 수 있음을 확인해 볼 수 있었다. 또한, 개선된 효율적인 알고리즘에 대한 실험은 적은 성능 저하로 큰 연산 효율성을 달성할 가능성을 입증하였다.

▸ **주제어**: 프라이버시 보호 기계 학습, 공정성, 동형 암호, 프라이버시, 안전한 기계 학습

• First Author: Saerom Park, Corresponding Author: Saerom Park
*Saerom Park (psr6275@sungshin.ac.kr), Dept. of Convergence Security Engineering, Sungshin Women's University
• Received: 2021. 02. 22, Revised: 2021. 04. 02, Accepted: 2021. 04. 12.


http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

## I. Introduction

인공지능 알고리즘의 발전으로 실제 시스템의 많은 부분이 알고리즘으로 자동화 되고 데이터 기반의 의사 결정에서 인간의 개입을 최소화 하려는 노력이 계속되고 있다. 하지만, 이러한 의사 결정의 자동화를 위해서는 다량의 데이터들이 활용되어야 하므로 데이터의 프라이버시에 대한 문제점에 대해 많은 걱정을 낳고 있다. 국내에서는 2020년 데이터 3법이 통과 되고 미국 및 유럽에서도 개인 정보 및 민감 정보의 활용에 대한 법규들이 생겨나면서 민감 정보의 보호에 대한 관심이 높아지고 있기 때문에, 민감 정보들을 모형에 활용하기 위해서는 각별한 주의가 필요하다 [1-2]. 따라서 데이터의 프라이버시를 보호하면서도 기계학습 모형을 학습하고, 활용하고자 하는 여러 연구들이 활발하게 진행되고 있다 [3-6].

데이터 프라이버시 및 학습되는 모형의 정보를 보호하기 위해서 활용되는 방법은 주로 차분 프라이버시 (Differential Privacy; DP), 연합 학습 (Federated Learning; FL), 암호 기법들을 활용한 다자간 계산 (Multi-Party Computation; MPC) 이나 동형암호 (Homomorphic Encryption) 등이 있다 [4][7-9]. 이러한 방법들은 적용되는 문제의 성격이나 시나리오에 맞게 활용되고 있다. 예를 들면 연합 학습의 경우에는 주로 다수의 사용자들이 기계학습 모형의 학습에 참여할 필요가 있을 때 주로 사용되고, 차분 프라이버시 같은 경우에는 모형의 안전성이나 성능에 대한 영향을 이론적으로 계산할 필요가 있을 경우에 주로 활용이 된다. 암호 기법들을 활용한 다자간 계산이나 동형암호와 같은 방법들은 암호 체계의 안전성을 활용하여 데이터의 프라이버시를 보호하면서 계산이 가능하기 때문에 차세대 안전한 암호 기술로써 큰 관심을 얻고 있다 [4-6].

본 논문에서는 계산 능력을 가진 공공 클라우드 서비스 제공자를 통해 민감 데이터를 포함한 학습 데이터를 제공하여 학습을 할 경우에 민감 정보를 보호하면서 효율적인 학습이 진행될 수 있는 동형암호 기반 학습 방법론을 제안하고자 한다. 동형암호의 경우에는 계산 시에 다자간의 통신을 최소화 할 수 있는 계산이 가능하기 때문에, 데이터 제공자가 온라인 상태로 계산에 지속적으로 참여하고 있지 않더라도 학습이 가능하다는 특징을 갖고 있다. 하지만 암호화 하지 않은 평문을 가지고 학습하는 것보다 계산 속도가 느리고, 동형암호 연산들의 연산 효율성이 다 다르기 때문에 효율적인 계산을 위해서는 세심한 알고리즘에 대한 설계가 필요하다. 따라서 본 연구에서는 보호가 필요한 민감 정보만 암호화를 통해 보호하여 효율적인 연산이 가능하도록 하는 방법론을 제안하였다.

## II. Preliminaries

### 1. Related works

동형 암호는 덧셈과 곱셈 등을 암호화된 상태에서 수행 가능하기 때문에 다양한 기계학습 방법론들의 응용에 활용하고자 하는 연구들이 진행되어 왔다 [4-6]. 동형암호의 경우에는 덧셈과 곱셈의 연산이 가능하기 때문에 다항 연산들의 경우에는 직접적으로 암호화된 상태에서 수행할 수 있지만, 비다항 연산들의 경우에는 근사를 통해서 수행하는 것이 필요하다 [3]. 특히, 레벨 동형암호에서 곱셈의 경우에는 제한적인 횟수만큼의 연산이 가능하고, 이를 완전 동형암호로 만들 수 있는 bootstrapping 연산의 경우에는 계산 시간이 많이 소모될 수 있기 때문에 실제 활용 가능한 효율적인 적용을 위해서 기계학습 모형의 학습 보다는 추론에 적용한 연구들이 더 활발하게 이루어져왔다 [10].

하지만, bootstrapping 연산의 효율성이 증대되고, CKKS17, TFHE 와 같은 암호 체계들이 개발되면서 동형암호를 기계학습의 학습 과정에도 적용하고자 하는 연구들이 활발하게 제안되고 있다 [3][10-11]. Logistic Regression, Support Vector Machine (SVM), Linear Regression 과 같은 모형들의 학습 과정이 동형암호 상에서의 연산으로 나타내지고, 간단한 신경망 모형의 학습을 동형암호로 수행하고자 하는 연구들이 제안되었다 [4-6][12]. 대부분의 이러한 연구들은 주어진 모든 데이터의 암호화를 가정하고 진행하였기 때문에, 실제 적용하기 위해서는 계산 시간이나 암호문의 데이터 크기에 대한 제약들에 대한 고민이 중요하게 고려되어야 했다. 따라서 본 연구에서는 민감 정보가 포함된 데이터가 주어진 경우에 암호화된 민감 정보와 그렇지 않은 민감하지 않은 정보를 함께 SVM 모형 학습에 활용함으로써 학습의 효율성을 높이고자 하였다.

### 2. Fully Homomorphic Encryption

동형암호 (Homomorphic Encryption) 는 암호화된 상태에서 데이터의 연산이 가능하고, 암호화된 연산 후에 복호화를 통해서 연산 결과를 얻을 수 있기 때문에 안전한 데이터 연산을 위해 사용될 수 있다 [3]. 동형암호의 종류는 크게 다음의 세 가지로 나눌 수 있다: Partially HE, Somewhat HE, Fully HE. 이 중에서 Partially HE 의 경

우에는 암호화된 상태에서 덧셈 혹은 곱셈 연산만 가능하다. 암호화된 상태의 덧셈이 보존되는 Paillar 암호나 곱셈 연산이 보존되는 RSA 암호와 같은 암호가 이에 해당된다. Somewhat HE 의 경우에는 암호화된 상태에서 덧셈과 곱셈이 모두 가능하지만, 연산의 결과로 발생할 수 있는 노이즈의 증가가 임계점 이하일 경우에만 올바른 복호화 결과를 얻을 수 있다. Leveled HE 는 이러한 Somewhat HE 의 일종으로 가능한 (곱셈) 연산의 횟수가 제한된다. Fully HE 는 제약 없이 어떤 함수의 계산이나 가능한 경우를 뜻하고, Leveled HE에서 노이즈를 초기화해 줄 수 있는 Bootstrapping 연산이 함께 정의될 수 있으면 FHE 암호를 구축할 수 있게 된다. 완전 동형암호를 위한 Bootstrapping 연산은 시간이 오래 걸리기 때문에, 이를 효율적으로 수행하기 위한 방법들이 제안되어 왔다 [10].

완전 동형 암호는 암호화된 상태에서 임의의 연산이 가능하기 때문에 기계학습의 다양한 응용들에 활용될 수 있다. 또한 Single Instruction Multiple Domain (SIMD) 연산을 지원하는 경우에는 여러 개의 실수 혹은 정수 값들을 함께 packing 하여 하나의 암호문으로 나타낼 수 있기 때문에 더욱 효율적인 연산이 가능할 수 있다. 완전 동형암호 중에서 기계학습 응용에 각광 받고 있는 CKKS17 의 경우에는 Bootstrapping 연산을 효율적으로 수행하면서도 SIMD를 지원하고, packing 된 slots 의 이동(Rotate) 연산까지 지원하기 때문에 학습 알고리즘의 정교한 설계는 실용적인 동형암호의 기계학습에의 활용을 가능하게 한다 [3][10].

본 연구에서는 CKKS17을 기반으로 하여 방법론을 제안하였기 때문에 CKKS17에서 기본적으로 사용 가능한 암호문의 연산을 다음과 같이 제시한다. 다음의 연산들을 통해 본 제안하는 알고리즘을 나타낼 수 있게 되면 동형암호로 알고리즘의 구현이 가능해 진다.

### 3. Least Square Support Vector Machine

본 연구에서는 일부의 민감 정보가 주어졌을 경우에 학습 가능한 동형암호의 연산들로 나타내질 수 있는 SVM 학습 알고리즘을 제안한다. 제안되는 방법론은 SVM을 통해서 선형 classifier를 얻는 학습 알고리즘을 제안하기 때문에 먼저 기반으로 하는 Least Square Support Vector Machine 모형을 설명하고자 한다.

Support Vector Machine 은 maximum margin 선형 분류기를 학습하는 방법으로 학습 데이터 $\{(x_1, y_1), \dots, (x_n, y_n)\}, x_i \in \mathbb{R}^d, y_i \in \{-1,1\}$가 주어졌을 때 다음의 최적화 문제를 통해서 나타낼 수 있다.

$$\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + c\sum_{i=1}^{n}\xi_i$$
$$s.t.\ \ y_i(w^T x_i + b) \geq 1-\xi_i \quad \text{for } i=1,\dots,n$$
$$\xi_i \geq 0$$

위의 문제는 soft-margin 에 대한 regularization parameter $C>0$ 를 포함하고 있다. 위의 soft-margin SVM 모형의 제약 조건을 포함하여 hinge loss 의 형태로 나타내면 다음과 같다.

$$L(w,b) = \frac{1}{2} \| w \|^2 + c\sum_{i=1}^{n}\max(0, 1-y_i(w^T x_i + b))$$

Least Square SVM 모형의 경우에는 기존의 SVM 모형이 hinge loss를 통해서 제약조건을 나타낼 수 있는 것과는 달리 squared error를 통해서 제약 조건을 나타낼 수 있는 약간 변형된 문제 형태를 가지고 있다 [13]. Least Square SVM 모형의 원래 문제는 다음과 같다.

$$\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + c\sum_{i=1}^{n} e_i^2$$
$$s.t.\ \ y_i(w^T x_i + b) = 1-e_i \ \text{ for } \ i=1,\dots,n$$

위의 문제에서부터 Lagrangian 함수를 통해 쌍대 변수 $\alpha \in \mathbb{R}^n$에 대해서 KKT 조건으로부터 얻어지는 선형 시스템은 다음과 같다.

$$\begin{bmatrix} XX^T + cI_n & 1_n \\ 1_n^T & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ b \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix}$$

위의 식에서 $X \in \mathbb{R}^{n\times d}$ 은 학습데이터의 예측 변수들로 이루어진 행렬이고, $Y \in \{-1,1\}^n$ 는 학습 데이터의 타겟 변수들로 이루어진 벡터이다. 쌍대 문제가 선형 시스템으로 나타내질 수 있기 때문에 동형 암호를 사용한 SVM 모형의 학습을 위해서 least square problem 으로 변형 후에 gradient descent 방법을 적용해서 해를 구하는 방법을 제안하여 LSSVM 학습을 수행하는 방법이 제안되었다. 하지만, 이전 방법의 경우에는 전체의 커널 행렬이 모두 보호해야 하는 민감한 변수들을 포함하고 있다고 가정하였기 때문에 전체 행렬에 대해서 암호화를 수행하고, 암호화된 $(n+1)\times(n+1)$ 행렬 간 곱셈이 요구되었기 때문에 연산 속도가 오래 걸렸다 [4]. 본 연구에서는 위의 선형 시스템의 특징을 사용해서 일부의 민감 정보만 주어졌을 경우에 빠르게 학습을 수행할 수 있는 방법을 제안하고자 한다.

## III. The Proposed Method

### 1. LSSVM with sensitive part

본 연구에서는 주어진 학습 데이터가 일부의 민감 정보를 포함하고 있는 경우를 가정하고, 민감 정보의 종류에

따라 동형암호를 통해 효율적인 SVM 학습을 할 수 있는 방법론을 제안하고자 한다. 고려되는 민감 정보의 종류는 다음 두 가지 경우이다. 첫 번째는 타겟 데이터 레이블이 민감 정보인 경우로 이 경우에는 예측 변수들은 모두 민감하지 않은 정보로 이루어져 있고, 예측하고자 하는 정보가 민감한 경우를 나타낸다. 두 번째는 예측 변수 중의 일부가 민감 정보인 경우로, 해당 경우에는 나머지 예측 변수들은 민감하지 않은 정보를 나타내고, 타겟 레이블 또한 민감 정보를 포함하지 않은 경우를 나타낸다.

제안되는 방법론에서는 민감 정보의 프라이버시를 보호하기 위해서 동형암호를 활용하기 때문에 학습에 사용되는 연산들을 최대한으로 동형암호 연산들로 쉽게 적용될 수 있는 연산들로 표현하고자 한다. LSSVM을 통해서 학습을 수행할 경우에는 위의 선형 시스템에서의 행렬에 역행렬을 구하는 과정이 필요하다. 여기에서 중요하게 고려되어야 할 사항은 암호화된 데이터와 연산을 하게 되는 데이터는 encoding 과정을 통해서 암호문과 연산 가능한 형태로 표현되어야 하며, 연산의 결과는 암호문으로 주어지게 된다. 따라서 연산 과정 중에 한 번이라도 암호문과의 연산이 필요한 경우에는 이후 연산들이 모두 암호문 간의 연산으로 변환되어야 한다. 따라서 암호화되어야 하는 민감 정보와 민감하지 않은 정보들을 구분해서 나타내는 것이 효율적인 학습을 가능하게 한다.

하지만, 첫 번째 경우인 타겟 데이터가 민감 정보인 경우에는 선형 시스템의 역행렬을 암호화하지 않은 상태로 구할 수 있으므로 암호화 하지 않은 채로 계산 가능한 역행렬과 민감 정보인 y 와의 암호화된 연산만 수행하게 되면 암호화된 채로 $\alpha, b$를 구할 수 있다. 따라서 행렬과 벡터 간의 암호화된 상태에서의 효율적인 연산을 통해서 SVM 모형의 학습이 가능해 진다.

반면에, 민감 정보가 예측 변수에 포함된 경우에는 암호화된 데이터가 포함된 행렬의 역행렬을 구하는 것이 쉽지 않기 때문에 효율적인 학습을 위해서는 역행렬을 효율적이면서도 수치적으로 불안정하지 않도록 계산하는 것이 필요하다. 따라서, 다음 장에서 이러한 경우에 대해서 SVM모형을 효율적으로 학습할 수 있는 방법론을 제안하고자 한다.

## 2. Sensitive SVM for homomorphic encryption (SenSVMHE)

II-3 장에서 설명한 LSSVM 모형은 선형 커널 함수를 가정한 가장 기본적인 모형을 가정하였기 때문에 $XX^T \in \mathbb{R}^{n\times n}$ 행렬을 포함하게 된다. 이러한 SVM 모형을 비선형 분류기로 확장하기 위해서 커널 함수를 도입하게 되면, 데이터들 간의 내적 연산을 임의의 비선형 커널 함수로 나타낼 수 있다. 결과적으로 비선형 LSSVM 모형의 학습을 위한 선형 시스템은 $XX^T \in \mathbb{R}^{n\times n}$ 대신에 $K \in \mathbb{R}^{n\times n}, K_{i,j} = k(x_i, x_j)$ 를 사용해서 나타낼 수 있다.

일부의 민감 정보가 주어진 경우에 효율적인 학습을 위해서 본 연구는 커널 함수에 대한 한가지 가정을 바탕으로 한다. 먼저, 주어진 학습 데이터의 예측 변수가 다음과 같이 민감한 정보 $x^s$ 와 민감하지 않은 정보 $x^{ns}$ 를 포함하고 있다고 하면, 커널 함수가 다음과 같이 분해 될 수 있다고 가정한다.

$$\begin{aligned} k(x_i,x_j) &= k_1(x_i^s, x_j^s) + k_2(x_i^{ns}, x_j^{ns}) \\ &= x_i^s x_j^s + k_2(x_i^{ns}, x_j^{ns}) \end{aligned}$$

위의 커널 함수는 민감한 정보에 대한 부분과 그렇지 않은 부분으로 분해될 수 있고 이를 통해 커널 행렬 $K$ 도 다음과 같이 나타내진다.

$$K = K_s + K_{ns} = X_s X_s^T + K_{ns},\ (K_{ns})_{i,j} = k_2(x_i,x_j)$$

위의 식에서 $X_s \in \mathbb{R}^n$ 은 학습 데이터의 민감 정보들로 이루어진 벡터를 나타낸다. 가정을 바탕으로 효율적으로 선형 시스템의 역행렬을 계산하기 위해서 Schur Complement를 적용할 수 있다. 여기에서 Schur complement를 통해 역행렬을 구하기 위해서는 다음의 조건이 만족되어야 한다.

- $\widetilde{K} = K + cI_n$ 이 역행렬이 존재
- $S = -1_n^T (K + cI_n)^{-1} 1_n \neq 0$

위의 조건이 만족되기 때문에, 선형 시스템의 역행렬은 다음과 같이 나타내 진다.

$$\begin{bmatrix} \widetilde{K}^{-1} + \frac{1}{S}\widetilde{K}^{-1}1_n 1_n^T \widetilde{K}^{-1} & -\frac{1}{S}\widetilde{K}^{-1}1_n \\ -\frac{1}{S}1_n^T\widetilde{K}^{-1} & \frac{1}{S} \end{bmatrix}$$

위의 역행렬의 계산에서 민감 정보가 포함된 부분은 $\widetilde{K}, S$ 이고, $S$의 계산에도 $\widetilde{K}^{-1}$ 부분이 민감 정보를 포함한 부분이기 때문에 동형암호를 통한 효율적 연산을 위해서는 $\widetilde{K}^{-1}$의 계산이 가장 중요한 부분이 된다. 앞에서 커널 행렬에 대한 가정을 바탕으로, Sherman – Woodbery - Morrison 공식을 적용하면 다음과 같다.

$$\begin{aligned} \widetilde{K}^{-1} &= (K_{ns} + cI_n + X_s X_s^T)^{-1} \\ &= (K_{ns} + cI_n)^{-1} - \frac{(K_{ns} + cI_n)^{-1} X_s X_s^T (K_{ns} + cI_n)^{-1}}{1 + X_s^T (K_{ns} + cI_n)^{-1} X_s} \end{aligned}$$

위의 식에서 주목해야 할 부분은 암호화가 필요한 민감한 정보로 이루어진 $X_s$ 부분과 나머지 부분 $(K_{ns}+cI_n)^{-1}$ 이 구분될 수 있다는 것이다. 해당 식을 바탕으로 LSSVM 모형의 선형 시스템의 해를 나타내면 다음과 같다.

$$S = -1_n \widetilde{K}^{-1} 1_n$$
$$= -1_n^T (K_{ns}+cI_n)^{-1} 1_n + \frac{1_n^T (K_{ns}+cI_n)^{-1} X_s X_s^T (K_{ns}+cI_n)^{-1} 1_n}{1 + X_s^T (K_{ns}+cI_n)^{-1} X_s}$$

$$b = -\frac{1}{S} 1_n^T \widetilde{K}^{-1} Y$$
$$= -\frac{1}{S}\left( 1_n^T (K_{ns}+cI_n)^{-1} Y - \frac{1_n^T (K_{ns}+cI_n)^{-1} X_s X_s^T (K_{ns}+cI_n)^{-1} Y}{1 + X_s^T (K_{ns}+cI_n)^{-1} X_s} \right)$$

$$\alpha = \widetilde{K}^{-1} Y + \frac{1}{S} 1_n^T \widetilde{K}^{-1} Y \widetilde{K}^{-1} 1_n$$
$$= (K_{ns}+cI_n)^{-1} Y - \frac{X_s^T (K_{ns}+cI_n)^{-1} Y}{1 + X_s^T (K_{ns}+cI_n)^{-1} X_s} (K_{ns}+cI_n)^{-1} X_s - b\widetilde{K}^{-1} 1_n$$

동형암호를 활용하면 실수들이 packing된 암호문 간의 요소별 덧셈 및 곱셈 연산이 가능하고, packing된 실수들의 rotation 연산이 가능하다. 하지만, 덧셈 연산에 비해 곱셈 및 rotation 연산 속도가 느리기 때문에 효율적인 연산을 위해서는 행렬-벡터 간의 곱셈 연산을 효율적으로 수행할 수 있는 방법이 필요하다. 따라서, 위의 식으로부터 암호화된 상태로 계산이 필요한 곱셈 연산과 관련된 부분을 보면 다음과 같다.

(1) $X_s^T (K_{ns}+cI_n)^{-1} X_s$

(2) $1_n^T (K_{ns}+cI_n)^{-1} X_s$

(3) $X_s^T (K_{ns}+cI_n)^{-1} Y$

(4) $(K_{ns}+cI_n)^{-1} X_s$

암호화된 상태로 곱셈이 필요한 경우는 위의 (1)-(4)의 네 경우로 이루어져 있고, $\frac{1}{S}, \frac{1}{1+X_s^T (K_{ns}+cI_n)^{-1} X_s}$를 계산하기 위해서 비다항 함수인 역수를 계산해야 하는 부분에 대해서는 근사 알고리즘을 사용한다. 나머지 연산들은 얻어진 암호문들의 덧셈 혹은 뺄셈을 통해 구할 수 있다. (1)-(4) 의 연산을 보면, (2)-(4) 의 경우에는 $n \times n$ 행렬과 암호화된 $X_s$ 간의 연산이 필요하고, (1) 의 경우에는 (4)의 결과와 $X_s$의 암호문 간의 내적 연산을 통해 얻어질 수 있다. 따라서 전체 데이터들을 모두 암호화한 상태에서 계산하는 경우보다 민감한 정보만을 암호화하는 경우 암호문 간의 연산이 감소하게 되므로 연산 효율성의 증대가 가능하다.

하지만, 여전히 위의 계산을 위해서는 학습 데이터의 수 $n$이 큰 경우에 암호화해야 하는 행렬의 크기가 크기 때문에 다음 장에서는 low-rank approximation을 통해서 연산 효율성을 향상시킬 수 있는 방법을 제안하고자 한다.

## 3. Efficient SenSVMHE through low-rank approximation (ESenSVMHE)

이 장에서 제안하는 방법인 ESenSVMHE 의 경우에는 커널 행렬에서 민감 정보를 제외한 나머지 부분인 $\widetilde{K} = (K_{ns} + cI_n)$에 대해서 low-rank approximation을 수행함으로써 암호화된 데이터의 행렬-벡터 연산에서의 추가적인 효율성 향상이 가능하다. 특히, $\widetilde{K}$ 의 경우에는 민감하지 않은 정보로 이루어져 있고, 그 중에 민감하지 않은 정보들로부터의 커널 행렬인 $K_{ns}$의 경우에는 positive semi- definite symmetric matrix 이기 때문에 비음수 실수 고유값을 가진 고유값 분해가 가능하다. $K_{ns}$에 대해서 고유값 분해를 수행한 결과가 다음과 같다고 가정하자.

$$K_{ns} = V \Lambda V^T = \sum_{i=1}^{n} \lambda_i v_i v_i^T$$
$$V = [v_1 \mid \cdots \mid v_n] \in \mathbb{R}^{n \times n}$$
$$\Lambda = diag([\lambda_i]_{i=1,\ldots,n}) \in \mathbb{R}^{n \times n}$$

위에서 $V$는 고유벡터들로 이루어진 직교 행렬이고, $\Lambda$는 $K_{ns}$의 고유값 $\lambda_n \geq \lambda_{n-1} \geq \cdots \geq \lambda_1 \geq 0$ 로 이루어진 대각 행렬이다. 위의 결과로부터, $(K_{ns}+cI_n)^{-1}$는 다음과 같이 나타낼 수 있다.

$$(K_{ns}+cI_n)^{-1} = \sum_{i=1}^{n} \frac{1}{\lambda_i + c} v_i v_i^T$$

따라서 $\| (K_{ns}+cI_n)^{-1} - Z \|_F^2$ 를 최소화하는 rank $p$ 행렬을 구하면 다음과 같이 나타내 진다.

$$Z_p = \sum_{i=1}^{p} \frac{1}{\lambda_i + c} v_i v_i^T$$

이 때, $\| (K_{ns}+cI_n)^{-1} - Z_p \|_F = \frac{1}{\lambda_{p+1}+c}$가 되기 때문에 충분히 큰 고유 값을 가지는 경우에는 좋은 근사 결과를 얻게 될 수 있다. 위의 근사 행렬을 가지고 앞의 SVM 의 동형암호 곱셈 연산을 수행하게 될 경우 필요한 두 벡터 $w_1, w_2$ 에 대한 $w_1 (K_{ns}+cI_n)^{-1} w_2$ 의 근사를 구하는 식을 나타내 보면, 다음과 같다.

$$w_1 (K_{ns}+cI_n)^{-1} w_2 \approx \sum_{i=1}^{p} \frac{1}{\lambda_i + c} (w_1^T v_i)(w_2^T v_i)$$

위의 식을 통해서 low-rank approximation을 적용할 경우에 (1)-(4) 에 대한 계산 효율성이 향상될 수 있는 특성을 살펴보면 다음과 같다.

- $n \times n$ 행렬과 벡터 연산 대신에 p 개의 고유 벡터와 내적 연산 수행: $v_1^T X_s, \ldots, v_p^T X_s$

• 계산된 $v_1^T X_s, ..., v_p^T X_s$ 결과들을 (1)-(4) 계산을 위해 재사용 가능

따라서, 동형암호로 LSSVM 학습 알고리즘을 구현할 때 일부 민감 정보가 주어진 경우에 대해 효율적인 연산이 가능해 진다.

## 4. HE representations for ESenSVMHE

지금까지 제안된 방법론들을 바탕으로 동형암호 적용 시에 가능한 연산들로 나타내기 위해서 아래의 표기법들을 활용하고자 한다. 아래 표기에서 간단하게 나타내기 위해서 암호 체계의 키 정보들은 생략하였고, 실수 벡터를 하나의 ciphertext 로 packing 하였다고 가정하였다. 본 논문에서 활용한 HEAAN 의 자세한 암호 체계에 대한 설명은 다음의 논문을 참고하면 된다.

- Encryption/Decryption: $\vec{c} \leftarrow \mathrm{Enc}(\mathrm{X}) / \mathrm{Dec}(\vec{c}) \approx \mathrm{X}$
- Addition/Substraction:
$$\text{for } \vec{c_1} = \mathrm{Enc}(\mathrm{X}_1), \vec{c_2} = \mathrm{Enc}(\mathrm{X}_2),$$
$$\mathrm{Dec}(\vec{c_1} \pm \vec{c_2}) \approx \mathrm{X}_1 \pm \mathrm{X}_2$$
- Multiplication: $\mathrm{Dec}(\mathrm{Mult}(\vec{c_1}, \vec{c_2})) \approx X_1 \circ X_2$
- Rotation: $\mathrm{Rotate}(\vec{c}, \mathrm{r})$

위의 연산 중에서 Multiplication 의 경우에는 암호화된 상태에서의 두 개의 벡터 사이의 Hardmard 곱셈 연산에 대응하는 연산을 위해서 Rescale 연산이 필요하지만 생략하여 나타내었다. 또한 Rotation 연산의 경우에는 $r > 0$ 인 경우에는 벡터를 index가 증가하는 방향으로 r 만큼 이동하는 것에 해당 하게 되고, $r < 0$의 경우에는 반대 방향으로 이동하는 것을 나타낸다.

동형암호 연산들로 암호화된 연산을 수행하기 위해서는 여러 개의 실수 값들을 어떻게 하나의 암호문으로 packing 시킬 것인가에 대한 전략이 필요하다. 본 연구 논문에서는 n 차원 벡터들을 하나의 암호문으로 packing 시키면서 병렬화가 가능한 연산들을 multithreading을 통해서 동시에 계산하여 계산 효율성을 높이고자 하였다. Algorithm1에서 ESenSVMHE 알고리즘을 동형암호 연산들로 나타내었다. 특히, 실수 벡터 $a \in \mathbb{R}^n$ 암호문과 연산 가능하도록 polynomial encoding을 수행한 것을 $[a]$ 로 표기하였고, 암호문 간의 연산과 동일한 notation을 사용하였다.

Algorithm 1. Efficient Sensitive SVM for Homommorphic Encryption (ESenSVMHE)

**Input:** $X_s, Y, (K_{ns} + cI_n)^{-1}$, the p-smallest eigenvalues $0 \le \lambda_1 \le ... \le \lambda_p$ and the corresponding eigenvectors $v_1, ..., v_p$ for $K_{ns}$
**Output:** $\vec{c_a}, \vec{c_b}$ s.t. $\mathrm{Dec}(\vec{c_a}) \approx \alpha$, $\mathrm{Dec}(\vec{c_b}) \approx b$
**<<Calculate Components (1)-(4)>>**
$\vec{c_x} \leftarrow \mathrm{Enc}(\mathrm{X_s})$
**for** $i = 1, ..., p$ **do**
  $\vec{c_0} \leftarrow \mathrm{Mult_{ip}}(\vec{c_x}, [\frac{1}{\sqrt{\lambda_i + c}} \mathrm{v_i}])$
  $\vec{c_{t1}} \leftarrow \mathrm{Mult}(\vec{c_0}, \vec{c_s})$,
  $\vec{c_{t2}} \leftarrow \mathrm{Mult}(\vec{c_0}, [\frac{1}{\sqrt{\lambda_i + c}} 1_n^T v_i 1_n])$
  $\vec{c_{t3}} \leftarrow \mathrm{Mult}(\vec{c_0}, [\frac{1}{\sqrt{\lambda_i + c}} v_i^T Y 1_n])$
  $\vec{c_{t4}} \leftarrow \mathrm{Mult}(\vec{c_{t1}}, [\frac{1}{\sqrt{\lambda_i + c}} v_i])$
  **if** $i = 1$ **then**
    $\vec{c_1} = \vec{c_{t1}}, \vec{c_2} = \vec{c_{t2}}, \vec{c_3} = \vec{c_{t3}}, \vec{c_4} = \vec{c_{t4}}$
  **else**
    $\vec{c_1} \leftarrow \vec{c_1} + \vec{c_{t1}}, \vec{c_2} \leftarrow \vec{c_2} + \vec{c_{t2}},$
    $\vec{c_3} \leftarrow \vec{c_3} + \vec{c_{t3}}, \vec{c_4} \leftarrow \vec{c_4} + \vec{c_{t4}}$
  **end if**
**end for**
**<<Calculate the solution>>**
$\vec{c_{t1}} \leftarrow \mathrm{Inv}([1_n] + \vec{c_1})$
$\vec{c_{t2}} \leftarrow \mathrm{Mult}(\mathrm{Mult}(\vec{c_2}, \vec{c_2}), \vec{c_{t1}})$
$\vec{c_{S_{\mathrm{inv}}}} \leftarrow \mathrm{Inv}(\vec{c_{t2}} - [1_n^T (K_{ns} + cI_n)^{-1} 1_n 1_n])$
$\vec{c_{t2}} \leftarrow \mathrm{Mult}(Mlt(\vec{c_2}, \vec{c_3}), \vec{c_{t1}})$
$\vec{c_b} \leftarrow \mathrm{Mult}(\vec{c_{S_{\mathrm{inv}}}}, \vec{c_{t2}} - [1_n^T (K_{ns} + cI_n)^{-1} Y 1_n])$
$\vec{c_{t2}} \leftarrow \mathrm{Mult}(\mathrm{Mult}(\vec{c_3}, \vec{c_{t1}}), \vec{c_4})$
$\vec{c_\alpha} \leftarrow [(K_{ns} + cI_n)^{-1} Y] - \vec{c_{t2}}$
$\quad - \mathrm{Mult}(\vec{c_b}, [(K_{ns} + cI_n)^{-1} 1_n])$

Algorithm1 의 연산에서 $\mathrm{Mult_{ip}}(\cdot, \cdot)$ 부분은 두 원래 벡터에 대한 내적 연산의 결과에 대한 암호문을 계산할 수 있는 연산으로 Algorithm2 와 같이 동형암호의 연산들로 나타낼 수 있다 [4,5,6]. 또한 $\mathrm{Inv}(\cdot)$ 은 벡터의 모든 요소에 대해서 역수를 근사적으로 계산해 주는 함수이며 기존 논문에서 제안된 연산을 사용하였다 [14]. 또한, 비다항 함수에 대한 연산의 비용이 크기 때문에 총 2번의 $\mathrm{Inv}(\cdot)$ 함수만 적용하고 해당 결과를 재사용할 수 있도록 하였다. 메인 학습 알고리즘인 Algorithm1을 살펴보면, 크게 (1)-(4) 부분을 계산하는 부분과 계산된 결과를 바탕으로 최종 학습 파라미터인 $\alpha, b$ 를 계산하는 부분으로 이루어져 있다. Algorithm 1에서 암호화된 계산 중에 반복되는 부분을 최대한 활용함으로써 보다 효율적인 연산이 가능하도록 하였다.

Algorithm 2. Inner product of two vectors $\mathrm{Mult}_{ip}(\cdot\ ,\cdot\ )$

**Input:** $\vec{c_1} = \mathrm{Enc}(X_1), \vec{c_2} = \mathrm{Enc}(X_2)$
**Output:** $\vec{c_{res}}$ s.t. $\mathrm{Dec}(\vec{c_{res}}) \approx (X_1^T X_2)1_n$
$\vec{c_{res}} \leftarrow \mathrm{Mult}(\vec{c_1}, \vec{c_2})$
**for** $i = 1, \ldots, 2^{\lceil \log_2 n \rceil}$ **do**
  $\vec{c_t} \leftarrow \mathrm{Rotate}(\vec{c_{res}}, -2^j)$
  $\vec{c_{res}} \leftarrow \vec{c_{res}} + \vec{c_t}$
**end for**

## IV. Experiments

### 1. Experimental design

본 논문에서는 앞에서 제안된 Algorithm1 의 타당성을 검증하기 위한 실험을 수행하였다. Algorithm1의 구현 시에 필요한 $Mult_{ip}$는 Algorithm2를 이용하여 구현하였다. 실험에 사용한 데이터는 대학원 입학 가능성을 예측하는 Graduate Admission 데이터로 다음 Table 1 와 같은 변수들로 이루어져 있다. SVM 모형을 통해 예측을 수행하기 위해서 ChanceofAdmit 변수를 확률이 0.7 이상인 경우에는 1, 0.7 미만인 경우에는 -1로 두고 예측을 수행하였다. 이 때, 목표 변수의 비율은 약 5:3 이 된다. 또한 예측 변수 중에 하나를 sensitive attribute로 가정하여 암호화 하였으며, 민감하지 않은 변수들에 대해서는 비선형 커널을 사용하여 모형을 학습하였다. 본 연구에서는 동형암호, 그 중에서도 특히 CKKS17에서 가능한 연산들만을 python으로 직접 구현하여 실험을 수행하였다. 따라서 행렬은 벡터의 형태로 packing 하였으며, SIMD 덧셈, 곱셈 및 slot-wise rotation 등의 연산으로 Algorithm2에 따라 벡터-벡터 내적을 계산하고, 내적 연산을 기반으로 행렬-벡터 연산을 수행하였다. 특히, 정확한 계산이 불가능한 비다항 함수인 $\mathrm{Inv}(\cdot\ ,\cdot\ )$ 부분도 에서 제안된 iterative method (iteration=3) 를 사용하여 구현하였다. 위의 방법론을 동형암호로 직접 구현하기 위해서는 HEAAN, SEAL, TenSEAL과 같은 Library 들을 사용할 수 있다.

Table 1. Data decryption for Garduate Admission

| | Number of Data | Predictive attributes | Target |
|---|---|---|---|
| Train data | 300 | GRE Score (GRE), TOEFL Score(TOE), University Rating(ER), SOP, LOR, CGPA, Research(RE) | Chance of Admit |
| Test data | 100 | | |

### 2. Experimental results

Graduate Admission 데이터에 대한 실험 결과는 다음 Table 2 와 같다. Table 2 는 목표 변수를 제외한 나머지 변수들을 민감 변수로 설정하였을 때의 정확도 결과를 나타내고 있다. 이 때에 민감 변수로 선택된 $X_s$는 Table 1 에서의 약자를 사용하여 나타내었다. 커널로는 rbf $(k(x,x') = \exp(-\gamma \| x - x' \|^2)$ 과 polynomial $(k(x,x') = (\gamma x^T x' + 1)^d)$ 의 다양한 파라미터들에 대한 실험을 수행 하였다. Table에서 그냥 수치는 비다항 연산들을 근사한 SenSVMHE 알고리즘을 적용한 결과들을 나타내고, 괄호 안의 수치는 일반적인 LSSVM 학습을 수행하였을 때의 정확도를 나타낸다.

Table 2. Prediction performance in accuracy of SenSVMHE (LSSVM) for different sensitive attributes $X_s$

| $X_s$ | rbf kernel ($\gamma$) | | polynomial kernel ($d, \gamma$) | | | |
|---|---|---|---|---|---|---|
| | 0.1 | 1.0 | 2, 0.1 | 2, 1.0 | 3, 0.1 | 3, 1.0 |
| GRE | 0.82 (0.82) | 0.82 (0.80) | 0.80 (0.81) | 0.82 (0.82) | 0.84 (0.83) | 0.78 (0.78) |
| TOE | 0.82 (0.82) | 0.78 (0.79) | 0.80 (0.80) | 0.82 (0.83) | 0.81 (0.81) | 0.76 (0.77) |
| UR | 0.82 (0.81) | 0.79 (0.82) | 0.80 (0.79) | 0.81 (0.81) | 0.80 (0.80) | 0.78 (0.78) |
| SOP | 0.79 (0.81) | 0.76 (0.75) | 0.82 (0.81) | 0.81 (0.79) | 0.82 (0.81) | 0.75 (0.75) |
| LOR | 0.81 (0.81) | 0.78 (0.80) | 0.79 (0.80) | 0.79 (0.79) | 0.81 (0.81) | 0.78 (0.78) |
| CGPA | 0.82 (0.82) | 0.79 (0.77) | 0.81 (0.81) | 0.82 (0.82) | 0.80 (0.80) | 0.78 (0.78) |
| RE | 0.81 (0.81) | 0.77 (0.77) | 0.80 (0.80) | 0.80 (0.79) | 0.81 (0.81) | 0.76 (0.74) |

위의 결과로부터 동형암호를 사용하기 위해서 사용한 근사들이 적은 반복 횟수 (iteration =3)을 사용했음에도 불구하고 일반적인 LSSVM에 대해 비슷한 예측 성능을 달성할 수 있음을 확인할 수 있다.

Fig. 1 은 SenSVMHE 에서 low-rank approximation 을 적용하였을 때에 근사 시에 사용한 rank 의 변화에 따른 Algorithm1 (ESenSVMHE) 에 대한 정확도 변화를 보여준다. 해당 실험에서는 polynomial kernel $(k(x,x') = (0.1x^T x' + 1)^3)$을 사용하였고, University Rating(UR) 변수를 민감 변수로 놓았다. ESenSVMHE 적용 시에 rank 의 변화에 따른 정확도의 변화는 다음 Fig. 1과 같다. Fig. 1에서 x 축은 커널 행렬의 근사를 위해서 사용한 고유벡터의 수 (근사한 커널 행렬의 rank)를 나타내고, y 축은 분류 정확도를 나타내고 있다.

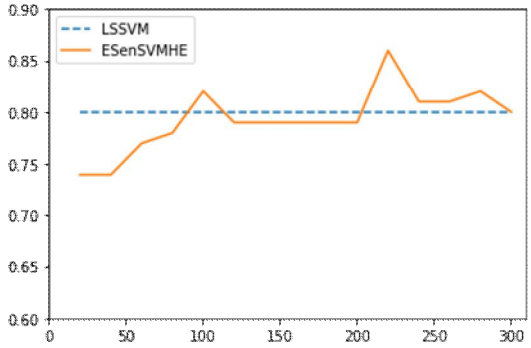


Fig. 1. Change of accuracy for different approximation rank

위의 결과로부터 비선형 커널을 사용할 경우에 전체 학습 데이터 수인 300보다 적은 70~80 정도의 rank 만 사용하여도 ESenSVMHE가 원래의 전체 행렬을 사용한 SenSVMHE 와 비슷한 성능을 나타낼 수 있음을 확인해 볼 수 있다. 따라서 동형암호로 구현 시에 성능 저하를 최소화 하면서 훨씬 더 효율적인 연산 속도를 달성할 수 있을 것으로 기대된다.

## V. Conclusions

본 연구 논문에서는 동형암호를 사용해서 일부의 민감한 변수가 주어진 데이터에 대해서 효율적으로 SVM 모형 학습 하는 방법을 제안하였다. SVM 모형의 학습을 위해서 LSSVM 모형을 통해 선형 시스템으로 나타내진 쌍대 문제를 얻을 수 있었다. 커널 행렬을 민감 변수에 대한 부분과 민감하지 않은 변수에 대한 부분으로 나눌 수 있다고 가정하면, Schur complement와 Sherman – Woodbery – Morrison 공식을 사용하여 암호화가 필요한 민감 변수 부분과 민감하지 않은 변수에 대한 부분을 구분할 수 있었다. 동형 암호 상에서의 곱셈 연산이 덧셈 연산보다 연산 비용이 높고, 비다항 연산에 대해서는 근사 연산만이 가능하기 때문에 암호화된 상태에서 해당 연산들을 재사용하여 연산 비용을 낮추고자 하였다. 또한 low-rank approximation을 통해서 추가적인 연산 비용의 감소가 가능하다.

본 연구 논문에서는 한 가지의 민감 변수가 주어진 경우에 SVM 학습 알고리즘을 제안하였다. 추후 연구를 진행하여 두 개 이상의 민감 변수를 가지고 있을 경우에도 효율적으로 연산이 가능한 연구로 확장시키면, 더 많은 상황에서도 민감 변수를 보호하면서도 효율적으로 SVM 알고리즘의 학습이 가능해질 수 있을 것으로 기대된다.



## ACKNOWLEDGEMENT

This work was supported by the Sungshin Women's University Research Grant of 2019

## REFERENCES

[1] Jagielski, Matthew, et al. "Differentially private fair learning." *International Conference on Machine Learning*. PMLR, (2019).

[2] Sunhwan Lee and Jongsu Park. "Legal Problems of the Guideline on De-identification of Personal Information and Ways to Improve the Personal Information Protection Legislation." *Public Law*, 45(2) (2016): 257-287.

[3] Cheon, J. H., Kim, A., Kim, M., and Song, Y., "Homomorphic encryption for arithmetic of approximate numbers." In *International Conference on the Theory and Application of Cryptology and Information Security*, Springer, Cham. (2017): 409-437.

[4] Park, S., Byun, J., Lee, J., Cheon, J. H., and Lee, J., HE-friendly algorithm for privacy - preserving SVM training. *IEEE Access, 8,* (2020): 57414-57425.

[5] Kim, M., Song, Y., Wang, S., Xia, Y., and Jiang, X., "Secure logistic regression based on homomorphic encryption: Design and evaluation." *JMIR medical informatics,* 6(2) e19. (2018)

[6] Chen, H., Gilad-Bachrach, R., Han, K., Huang, Z., Jalali, A., Laine, K., and Lauter, K. "Logistic regression over encrypted data from fully homomorphic encryption." *BMC medical genomics,* 11(4) (2018): 3-12.

[7] Dwork, C., "Differential privacy: A survey of results." In *International conference on theory and applications of models of computation*. Springer, Berlin, Heidelberg (2018): 1-19

[8] Bonawitz, Keith, et al. "Towards federated learning at scale: System design." *arXiv preprint arXiv:1902.01046* (2019).

[9] Ben-David, A., Nisan, N., and Pinkas, B., "FairplayMP: a system for secure multi-party computation." In *Proceedings of the 15th ACM conference on Computer and communications security* (2018): 257-266.

[10] Cheon, J. H., Han, K., Kim, A., Kim, M., and Song, Y., "Bootstrapping for approximate homomorphic encryption." In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Cham. (2018): 360-384

[11] Chillotti, I., Gama, N., Georgieva, M., and Izabachène, M., "TFHE: fast fully homomorphic encryption over the torus." *Journal of Cryptology, 33*(1), (2018): 34-91.

[12] Hall, R., Fienberg, S. E., and Nardi, Y. "Secure multiple linear

regression based on homomorphic encryption." *Journal of Official Statistics, 27*(4) (2011): 669.

[13] Polat, K., and Güneş, S. "Breast cancer diagnosis using least square support vector machine." *Digital signal processing,* 17(4), (2007): 694-701.

[14] Cheon, J. H., Kim, D., Kim, D., Lee, H. H., and Lee, K., "Numerical method for comparison on homomorphically encrypted numbers." In International Conference on the Theory and Application of Cryptology and Information Security, Springer, Cham. (2020): 415-445

## Authors

Saerom Park received the B.S. and Ph.D. degrees in industrial engineering from Seoul National University, in 2013 and 2018, respectively. Dr. Park is currently an Assistant Professor with the Department of Convergence Security Engineering, Sungshin Women's University, Seoul, South Korea. Her research interests include kernel machines, representation learning, transfer learning, and secure machine learning.