Universidad ORT Uruguay

Facultad de Ingeniería

Diseño e implementación de una plataforma para garantizar privacidad de datos en un contexto de machine learning as a service

Informe de Avance

Entregado como requisito del Máster en Big Data

Gonzalo Wagner - 187413

Walter Imbert - 82584

Sebastián Uriarte - 194973

2022

1. Introducción

Investigación de técnicas de encriptado para transmisión de datos seguros en consultas a modelos de Machine Learning utilizando PATE (Private Aggregation of Teachers Ensemble)

La necesidad de privacidad en los datos a consultar durante el entrenamiento de modelos de Inteligencia Artificial o Machine Learning, ha generado investigaciones que derivan en la implementación de ciertas prácticas o técnicas, que garantizan obtener un buen resultado de entrenamiento, en cuanto a precisión de los modelos obtenidos al finalizar este, sin conocer los datos reales.

Entre estas técnicas, se encuentra PATE, que en resumidas cuentas implica que las consultas realizadas por los interesados en los datos (por ejemplo data scientists), soliciten la información requerida a través de un intermediario, llamado “curador”.
Este distribuye y centraliza múltiples consultas a los llamados “*teachers*”, que son quienes contienen los posibles modelos y acceso y permiso sobre los datos necesarios para obtener una respuesta a la query del cliente. Estas respuestas luego son procesadas y consolidadas mediante un mecanismo de votación para obtener la respuesta final retornada a la solicitud del cliente.

Dado que esta técnica involucra la comunicación entre distintos agentes, desplegados en forma independiente, es necesario buscar mecanismos para transmitir de forma segura los datos entre las distintas entidades involucradas (*Student - Curator - Teacher*), por lo que se deberán implementar ciertos mecanismos de criptografía o encriptado.

Por otro lado, la utilización de PATE implica también la inclusión de cierta perturbación o “ruido” en la información devuelta, de manera que ninguna técnica con el fin de obtener los datos originales sea posible.

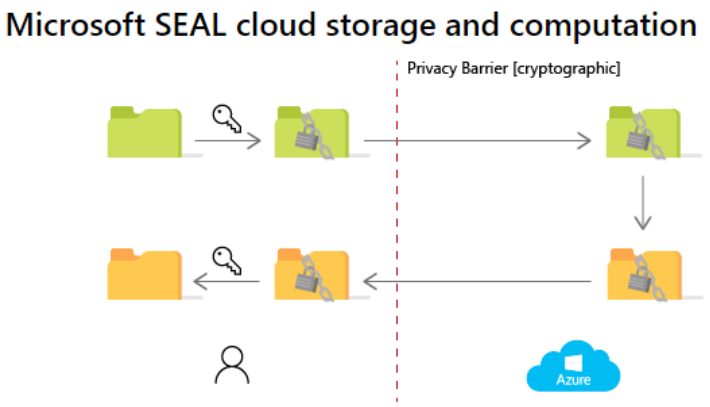
2. Trabajo realizado

Nuestro objetivo a lo largo de este trabajo es buscar generar el diseño y la implementación de una arquitectura que resuelva el problema anteriormente mencionado. Para ello, en una primera fase, el equipo se dedicó a la investigación de una librería denominada *Pyfhel*, que ofrece ciertas operaciones criptográficas que pueden ser útiles para lograr satisfactoriamente la funcionalidad requerida para nuestros objetivos.

Pyfhel es una librería disponible para el lenguaje de programación Python, que permite realizar las operaciones de suma, resta, multiplicación y producto escalar, sobre vectores encriptados, binarios, o enteros. Actúa como una API optimizada de librerías avanzadas de C++ en Python, siendo implementada sobre Afhel, una abstracción de encriptación homomórfica (implementada en C++).

La encriptación homomórfica, por otro lado, es un sistema de cifrado que permite realizar operaciones algebraicas (suma, resta, multiplicación, etc.) sobre datos encriptados. Mecanismos con esta característica permiten, concretamente, que se obtenga el mismo resultado encriptando dos valores independientemente para luego sumar los resultados obtenidos, que sumando primero los valores originales y luego encriptar la suma.
Concretamente, a efectos del presente trabajo, esto permitirá que ciertas operaciones a realizarse por parte del curator, se realicen sobre valores encriptados en lugar de sobre resultados reales, lo que permite una desencriptación más tardía y por tanto mayores garantías de seguridad y privacidad.

2.1. Instalación y dependencias



Afhel, librería en la que está basada Pyfhel, trabaja mediante la utilización de librerías implementadas en C++, por lo que es necesario instalar Microsoft SEAL (en Windows) o PALISADE (Linux).

Además del propio lenguaje de programación Python, para lograr ejecutar el prototipo desarrollado hasta el momento, es necesario instalar asimismo dependencias adicionales, tales como las librerías *numpy* y *flask.* Todas estas están disponibles para su instalación mediante el gestor de paquetes *pip.*

Notas:

- En nuestro caso instalamos SEAL lo cual también implicó instalar y configurar herramientas de C++.
- Una ventaja adicional de utilizar SEAL, es que puede ser desplegado en otros sistemas operativos como Android, por lo que a modo de ejemplo, mediante Pyfhel podríamos utilizar como Teacher un dispositivo con ese sistema operativo.

Para ejecutar el proyecto, basta con clonar el repositorio provisto (https://hdl.handle.net/20.500.12381/2375) y ejecutar el archivo *main.py.*

2.2. Transmisión segura de datos mediante encriptación:

Dos entidades que necesitan intercambiar información de manera segura, deben hacer uso de un mecanismo llamado encriptación, el cual consiste básicamente en transformar (encriptar o cifrar) los datos a transmitir, de manera tal que un tercer agente malicioso no pueda recuperar la información al no poder interpretar el mensaje ni deshacer la transformación a la que se ha sometido.

Existen dos mecanismos, llamados de clave simétrica y asimétrica respectivamente. Cuando se utiliza clave simétrica, emisor y receptor, utilizan la misma clave para “encriptar” y “desencriptar”. Si se utiliza clave asimétrica, en cambio, se utilizan dos claves, una llamada clave pública (la que se intercambia y se utiliza para cifrar mensajes) y otra llamada clave privada (que no se intercambia, y se utiliza para descifrar).

Dichas claves tienen una relación interna matemática, basada en métodos numéricos y propiedades de números primos, que permite que solo quien tenga las claves indicadas sea capaz de deshacer la encriptación de un mensaje cifrado.

De manera resumida, el mecanismo de comunicación es el siguiente: al inicio de la misma, el receptor de datos envía al emisor su clave pública, quien producirá el mensaje que se desea intercambiar de manera segura, y lo encriptará con la clave pública recibida. Una vez finalizado esto, se producirá la transmisión del mensaje cifrado, que al ser recibido por el receptor de manera correcta, podrá ser (únicamente) desencriptado con la clave privada asociada a la clave pública inicialmente enviada, pudiendo entonces acceder a la información de manera correcta.

Pyfhel utiliza claves asimétricas, por lo que deberá implementarse el mecanismo para intercambio de dichas claves. En particular, en un futuro se propone agregar un segundo mecanismo de encriptación (basado en el uso de HTTPS), para asegurar

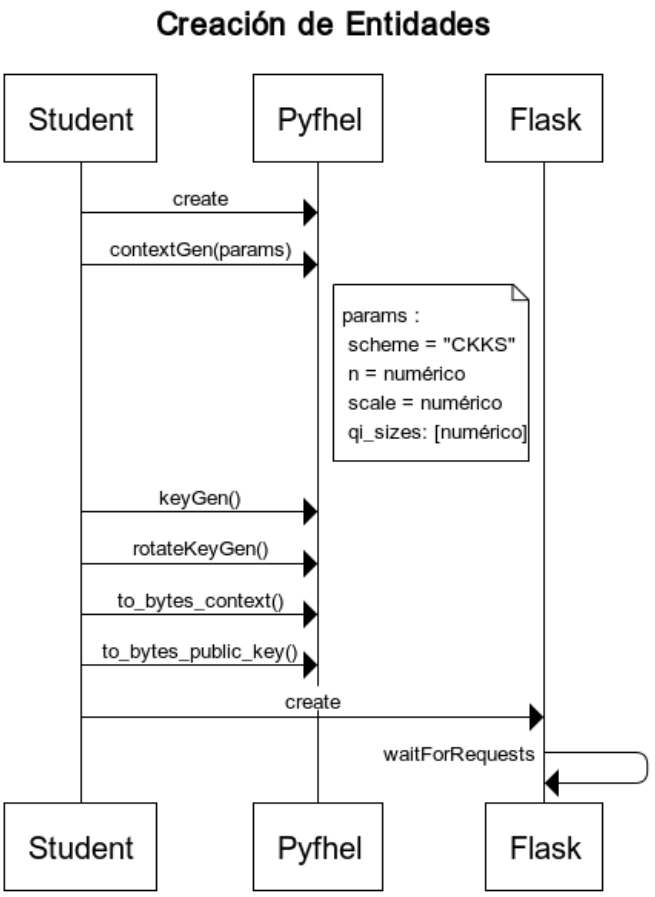
que la comunicación de claves públicas entre los distintos agentes del sistema se realice de manera segura.

2.3. Implementación de un prototipo

Dentro de los objetivos planteados para este trabajo, y planteando la utilización de Pyfhel como primer librería que podría permitir la utilización de funcionalidades de criptografía necesarias para cumplir satisfactoriamente con los mismos, se plantea la realización de un prototipo de los distintos agentes (Student - Curator - Teacher) y su implementación bajo esta, de forma de que estos logren comunicarse exitosamente de manera segura y garantizando la privacidad de los datos..

Dado que, como se mencionó anteriormente, cada una de estas entidades estará desplegada de manera independiente, es necesario introducir mecanismos de comunicación entre las partes. Concretamente, en este primer acercamiento se utiliza la librería Flask para recibir peticiones a través de la web[1].

Implementamos 3 tipos de procesos, uno para el Curator, otro para los Students, y finalmente otro para los Teachers.



[1]: Flask es un mini framework Python que permite publicar servicios web utilizando el patrón MVC (Model-View-Controller)

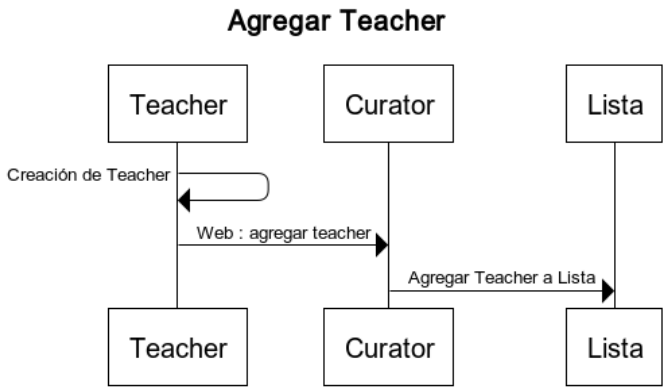
Las tres entidades utilizan Pyfhel, por lo tanto, tienen un proceso similar: lo inician, setean sus parámetros (tal como las claves a utilizar durante el intercambio de datos) y finalmente, de corresponder, se crea una aplicación Flask esperando solicitudes (web).

Una vez que cada una de las entidades están seteadas y activas, los datos pueden comenzar a intercambiarse. Dicho escenario será iniciado por el Student, quien tiene la necesidad de consumir datos referentes a cierto modelo de Machine Learning.

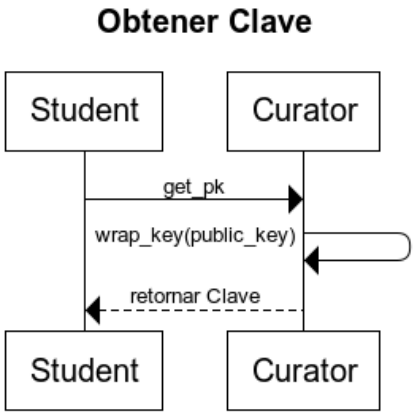
Entidades

Curator

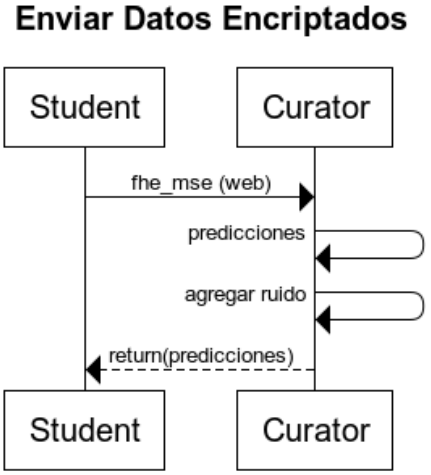
- Inicialización
- Agregar teacher (web)
  Agrega a una lista de teachers el teacher que solicitó ser registrado.



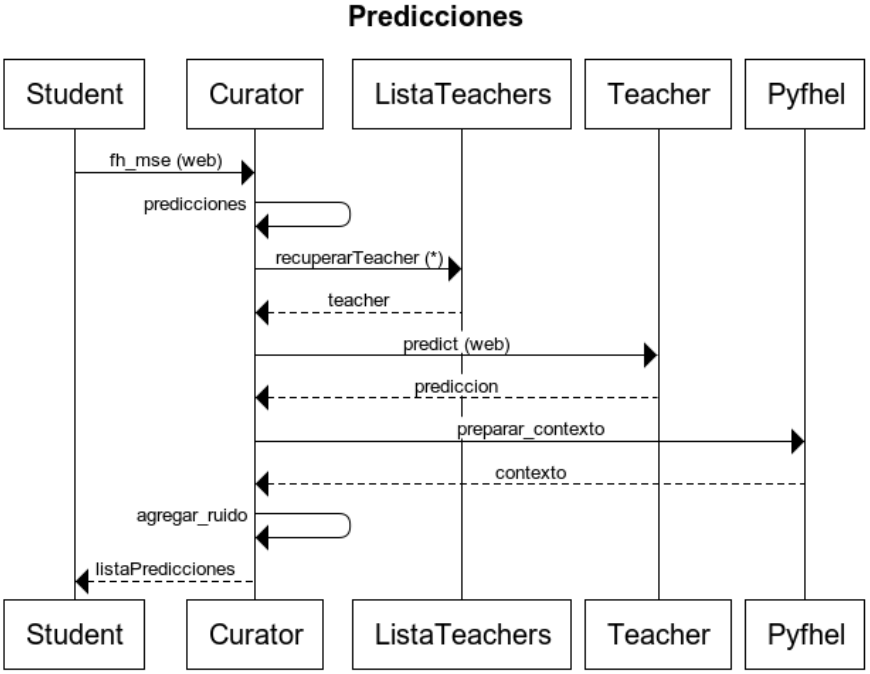
- Obtener clave pública (web)
  Envía la clave pública a quien la solicite



- Enviar datos encriptados homomórficamente (web)
  Realiza las predicciones del modelo, les agrega ruido, y las envía a quien la solicitó de manera segura.

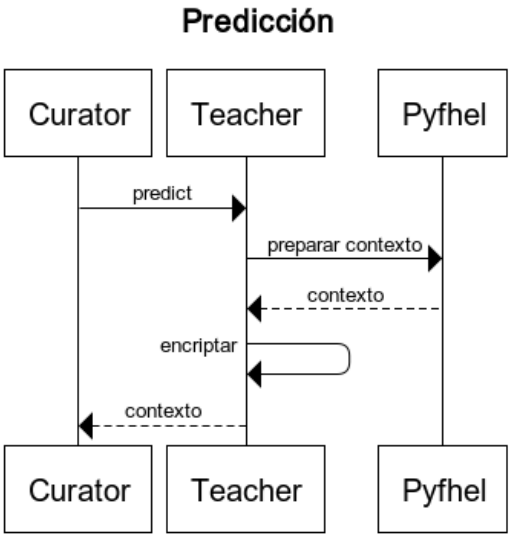


- Predicciones
  Solicita a todos los teachers registrados las predicciones del modelo.



Teacher

- Inicialización
- Predecir (web)
  Consulta el modelo, encripta la información y la devuelve.



Student

- Inicialización
- Solicitar información

Comunicación

En resumen, el mecanismo de comunicación entre las partes será el siguiente: en un inicio, se ejecutan los tres agentes. Se inicializan los objetos vinculados a Pyfhel, con el Student y Teachers creando sus pares de claves pública y privada. Cada Teacher, dado que el número y la naturaleza e implementación de ellos puede ser variable, se registra o suscribe (en forma similar a lo que plantea el patrón de diseño Observer) respecto del Curator como para que este tenga noción de su existencia y así pueda dirigirle peticiones. En esta request también se efectúa el intercambio de claves, enviando cada Teacher su clave pública hacia el Curator.

Cuando la petición por parte del Student se produce, este envía, junto con los posibles datos asociados a la request, su clave pública al Curator, que la envía a su vez al Teacher. Éste, cada uno de ellos ejecuta su modelo y encripta el resultado con la clave recibida, el Curator suma todos los resultados obtenidos (posible pues se utiliza criptografía homomórfica), y lo retorna al Student que desencripta y realiza las posibles restantes operaciones necesarias sobre el resultado real.

2.3. Sugerencias de cambio

Cambios arquitectónicos

La versión inicial fué realizada como prueba de concepto evaluando la tecnología. Muchos cambios se creen necesarios para una implementación definitiva.

A modo de ejemplo, la utilización de buenas prácticas de diseño se consideran convenientes. La implementación actual tiene un alto acoplamiento y baja cohesión. Por ejemplo, cualquiera de las tres entidades interactúan con Pyfhel, Flask y las propias entidades del sistema. Es decir, tiene demasiadas conexiones entre objetos (alto acoplamiento). Por otro lado, las responsabilidades asignadas a cada entidad, son también demasiadas, crean claves, crean la plataforma web, crean las conexiones con Pyfhel.

Esto deberá revisarse en un diseño definitivo.

Evaluación del rendimiento

Por ejemplo, es necesario testear Flask en un ambiente estresado por excesivas solicitudes. Esto quizás determine que la tecnología a utilizar para las solicitudes web sea otra.

3. Bibliografía

1. Papernot, N.; Abadi, M.; Erlingsson, U.; Goodfellow, I.; Talwar, K. Semi-supervised knowledge transfer for deep learning from private training data. arXiv 2016, arXiv:1610.05755. https://arxiv.org/pdf/1610.05755.pdf
2. “The Boston House-Price Data.” Department of Statistics, Carnegie Mellon University, http://lib.stat.cmu.edu/datasets/boston.
3. Yovine, S.; Mayr, F.; Sosa, S.; Visca, R. An Assessment of the Application of Private Aggregation of Ensemble Models to Sensible Data. Mach. Learn. Knowl. Extr. 2021, 3, 788-801. https://doi.org/10.3390/make3040039
4. J. Ramas, A. Rodríguez, S. Zanotta. Implementación de las prácticas de MLOps para PATE. Trabajo Final de Carrera, 2022. https://hdl.handle.net/20.500.12381/2362
5. ibarrond: Pyfhel. https://github.com/ibarrond/Pyfhel
6. Microsoft-Seal: https://www.microsoft.com/en-us/research/project/microsoft-seal/
7. Palisade: https://palisade-crypto.org/
8. R. Shokri, V. Shmatikov. Privacy-preserving deep learning. ACM SIGSAC Conference on Computer and Communications Security, CCS ’15, 2015
9. Yi, X., Paulet, R., Bertino, E. (2014). Homomorphic Encryption. In: Homomorphic Encryption and Applications. SpringerBriefs in Computer Science. Springer, Cham.