Universidad del Desarrollo
Facultad de Ingeniería

DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA DE SEGURIDAD MULTICAPA PARA LA PROTECCIÓN DE LA PROPIEDAD INTELECTUAL EN MODELOS PREDICTIVOS INDUSTRIALES

Implementación de una Arquitectura de Defensa en Profundidad utilizando Oracle 19c Enterprise TDE y Privacidad Diferencial.

POR: VICTOR ARCIDES SALDIVIA VERA Y CRISTIAN MATÍAS TOBAR MORALES

Proyecto de grado presentado a la Facultad de Ingeniería de la Universidad del Desarrollo para optar al grado académico de Magíster en Data Science.

PROFESORES GUÍAS:

Dr. Juan Elizalde y Dr. Germán Gómez

Diciembre, 2025

SANTIAGO

TABLA DE CONTENIDO

Resumen

La industria química se encuentra ante un desafío complejo en la era digital: por un lado, necesita aprovechar las capacidades de la de estos tiempos la Inteligencia Artificial para acelerar el desarrollo de nuevos productos y reducir tiempo y costos operacionales; por otro lado, debe proteger rigurosamente su propiedad intelectual, especialmente las fórmulas secretas que constituyen su ventaja competitiva en el mercado. Este proyecto busca resolver esta tensión en el contexto de la empresa, proponiendo una arquitectura de seguridad multicapa con la finalidad de utilizar los datos para predecir el punto de inflamación (Flash Point) de mezclas químicas complejas.

Para alcanzar este objetivo, se diseña e implementar un sistema de Machine Learning seguro sobre Oracle Database 19c Enterprise Edition. La protección de los datos se abordó en múltiples niveles: cuando los datos están almacenados, se utiliza Transparent Data Encryption (TDE) con cifrado AES-256, mientras que durante su transmisión se emplea Oracle Native Network Encryption (NNE). A nivel de aplicación, se adopta una estrategia de privacidad diferencial durante el entrenamiento del modelo.

Los resultados experimentales determinaron que es posible encontrar un equilibrio razonable entre la utilidad del modelo y la protección de información sensible. Mientras que el modelo básico (Baseline) alcanzó una precisión del 89.2% trabajando con datos sin protección, el modelo con privacidad diferencial implementado ruido de ϵ=1.5 logrando mantener una precisión del 80.1%. Si bien esto representa una reducción del 9% en la capacidad predictiva, la arquitectura propuesta ofrece garantías matemáticas contra ataques de inferencia y reconstrucción de datos. Estos resultados respaldan su aplicación práctica como herramienta segura para priorizar experimentos de laboratorio y minimizar riesgos operacionales, sin comprometer la confidencialidad de las fórmulas propietarias.

Palabras clave: Machine Learning, Privacidad Diferencial, Oracle TDE, Flash Point, Propiedad Intelectual.

1 Introducción

1.1 Contexto del problema

La empresa del sector de sabores y fragancias, líder en la industria, Fundamenta su ventaja competitiva en un vasto repositorio de propiedad intelectual acumulado a lo largo de décadas. Cada fórmula es un activo de alto valor que representa años de investigación y desarrollo, y cada fórmula normalmente se compone de diferentes materias primas. La confidencialidad de esta información es crítica para la sostenibilidad y el liderazgo de la empresa en el mercado.

1.2 Definición del problema

Actualmente, la empresa enfrenta un desafío estratégico que es la necesidad de adoptar analítica avanzada y Machine Learning para optimizar procesos y predecir propiedades de nuevos compuestos como el punto de inflamación o *Flash Point*, lo que se contrapone directamente con la política de máxima protección de su propiedad intelectual. El uso de datos en modelos predictivos requiere acceso a las formulaciones, lo que, bajo la infraestructura actual, generaría una exposición inaceptable de sus activos más valiosos. La ausencia de un mecanismo que permita el análisis de datos sin comprometer su confidencialidad está frenando la innovación y la eficiencia operativa.

1.3 Pregunta de Investigación

¿Cómo se puede diseñar e implementar una arquitectura de datos y un pipeline de Machine Learning que proteja la confidencialidad de fórmulas químicas propietarias mediante técnicas de encriptación, permitiendo al mismo tiempo el entrenamiento de modelos predictivos precisos para su aplicación en un contexto industrial?

1.4 Hipótesis

Implementar una arquitectura de seguridad multicapa, que combine el cifrado simétrico (AES-256) para datos en reposo y en tránsito con un entorno de cómputo seguro para el entrenamiento de modelos, lo que permitirá desarrollar un modelo predictivo del *Flash Point* con un error absoluto medio (EAM) competitivo, sin exponer nunca la composición completa y descifrada de las fórmulas químicas durante el proceso.

1.5 Objetivos del Proyecto

1.5.1 Objetivo General

Diseñar, implementar y validar una arquitectura de MLOps (Machine Learning Operations) seguro que garantice la protección de la propiedad intelectual de fórmulas químicas mediante cifrado multicapa y privacidad diferencial, operacionalizando un modelo predictivo de Flash Point para habilitar su uso industrial seguro en la empresa.

1.5.2 Objetivos Específicos

1. Diseñar una estrategia de protección de datos en reposo y en tránsito utilizando Oracle Transparent Data Encryption (TDE) y Network Encryption, mitigando el riesgo de exfiltración de la base de datos maestra.
2. Desarrollar un esquema de ingeniería de características que transforme la composición química exacta en atributos estadísticos agregados, implementando vistas seguras para eliminar la exposición directa de ingredientes propietarios durante la etapa de análisis.
3. Construir y evaluar modelos de Machine Learning integrando mecanismos de Privacidad Diferencial, determinando el parámetro de privacidad óptimo (ε) que maximice la precisión predictiva sin comprometer la garantía matemática de anonimato.

4. Desplegar el modelo seleccionado mediante una arquitectura de microservicios contenerizada (Docker) y una API REST, asegurando la inmutabilidad del entorno de inferencia y la segregación del acceso a los artefactos del modelo.
5. Establecer un marco de Gobernanza de Datos mediante la implementación de un sistema de auditoría y un dashboard de monitoreo, permitiendo la trazabilidad completa de los accesos a la información sensible.

1.6 Alcance y Limitaciones

El alcance de este análisis cubre el ciclo de vida completo de los datos, desde su almacenamiento en la base de datos (Datos en Reposo), su transferencia a los entornos de analítica (Datos en Tránsito), hasta su procesamiento en memoria durante el feature engineering y el entrenamiento del modelo (Datos en Uso).

1.7 Metodología y Plan de Trabajo

El proyecto se ejecuta en tres fases secuenciales, diseñadas para abordar primero el desafío de la seguridad de los datos y luego integrar la solución de Machine Learning sobre esa base ya construida.

Fase 1: Diseño e Implementación de la Arquitectura de Seguridad

Esta fase se centra en la creación de un entorno seguro y controlado para el manejo de los datos de la empresa. Esta fase se desarrollará entre octubre y noviembre.

- **Análisis de Amenazas (Threat Modeling):** Se lleva a cabo un análisis para mapear los potenciales puntos de vulnerabilidad y los riesgos asociados a la fuga de información. Este análisis cubre la totalidad del ciclo de vida de los datos, desde su origen hasta su aplicación en el modelo de Machine Learning.

  Para estructurar este análisis, adoptamos un Enfoque de Modelado de Amenazas Híbrido, esta metodología combina el análisis tradicional centrado en los activos (el "qué" se protege), con la categorización formal proporcionada por el

framework STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege).

El enfoque híbrido es crucial porque permite rastrear las amenazas a través del ciclo de vida de los datos (Reposo, Tránsito, Uso) y justifica, por qué nuestro foco principal es la categoría Information Disclosure, ya que esta se alinea directamente con el objetivo de proteger la Confidencialidad de la propiedad intelectual de la empresa.

- **Diseño del Protocolo de Cifrado:** Se define un sistema de encriptación multicapa. Los datos en reposo serán cifrados con el algoritmo AES-256. Se diseñará un proceso para que los datos utilizados en el entrenamiento se mantengan cifrados o se procesen en un entorno de "caja limpia" donde no puedan ser extraídos.
- **Proof of Concept (PoC) de Seguridad:** Se implementa un prototipo a pequeña escala del pipeline de datos seguro, demostrando que es posible leer, procesar y cargar los datos para el entrenamiento sin que estos sean expuestos en formato legible en ningún punto intermedio del proceso.

Fase 2: Integración y Validación del Modelo de Machine Learning

Con la arquitectura de seguridad implementada, esta fase se enfoca en desarrollar y validar el modelo predictivo del Flash Point. Se centra en los meses de noviembre y diciembre.

- **Implementación del Modelo Predictivo:** Se entrena un modelo de regresión (por ejemplo: modelo de regresión lineal, multilineal, Random Forest, XGBoost) utilizando el pipeline seguro. El entrenamiento se realiza sobre datos sintéticos o una muestra cifrada de datos reales, según lo defina el protocolo de seguridad.
- **Construcción del Pipeline Seguro de Datos:** Se automatiza el flujo de datos desde la fuente cifrada hasta el entorno de entrenamiento, garantizando que cada etapa, incluyendo la transformación de datos y la ingeniería de características, se ejecute dentro de un entorno controlado y en conformidad con los protocolos de seguridad establecidos.

- **Evaluación de Métricas de Desempeño vs. Seguridad:** Se realiza una evaluación cuantitativa para analizar la precisión del modelo (medida con métricas como EAM y R^2) y el nivel de seguridad. Se analizan técnicas de protección tales como el uso de datos sintéticos o la aplicación de privacidad diferencial, impactan en el rendimiento predictivo, con el fin de establecer un equilibrio óptimo que maximice la utilidad del modelo garantizando la confidencialidad de los datos.

Fase 3: Documentación, Escalamiento y Transferencia

La fase final se dedica a consolidar los resultados y preparar los entregables para su posible adopción por parte de la empresa. Esta última fase se centra en los meses de diciembre y enero.

- **Propuesta de Gobierno de Datos:** A partir de la evidencia técnica recolectada en el proyecto, se diseña un marco de gobernanza para la analítica de datos en la empresa. Esta propuesta define una estructura de roles, una matriz de permisos y protocolos operativos para el manejo seguro de la información propietaria. (Sujeto a confirmación de la empresa).
- **Documentación Técnica y de Usuario:** Se genera una documentación que incluya la arquitectura del sistema, el código fuente comentado y una guía de uso para la ejecución del pipeline y la re-entrenamiento del modelo.
- **Transferencia de Conocimiento:** Se prepara el repositorio final del proyecto y se realiza una sesión de presentación y transferencia de los artefactos desarrollados al equipo de la empresa.

2 Marco Teórico y Estado del Arte

2.1 Predicción de Propiedades Químicas

Aunque los métodos experimentales para determinar el punto de inflamación son precisos, resultan costosos, lentos y potencialmente peligrosos. Por ello, los métodos de predicción mediante Machine Learning representan una alternativa eficaz y segura. Este apartado

presenta la literatura sobre estas aplicaciones, mostrando diferentes enfoques de estudio de diversos autores, resumidos en la Tabla *1*.

Tabla 1. Comparación de los modelos propuestos por los autores.

Ranking	Estudio	Modelo	R^2	AAE (%)	AARE (%)	Fortaleza
1	Saldana, et al. (2013)	HPM + VLE	0.974	3.4	1.2	Mezclas de combustibles
2	Lee, Ko & Lee (2012)	SVR + PSO (3-clases)	0.967	9.8	1.1	Balance óptimo
3	Jeong, et al. (2024)	BKR + GP	0.882*	N/A	N/A	Incertidumbre + mecanismos
4	Ghargehpeisi & Fareghi Alamdari (2008)	GA-MLR	0.869	16.6	-1.25	Ecuación lineal explícita
5	Katritzky, et al. (2007)	MLR/AANN	0.878	(ANN) 12.6	(ANN) <1.5	Verificación reproducibilidad

Fuente: Elaboración propia, 2025.

* 8.5× más grande (6714 vs 740-1282), por lo que R^2 ligeramente menor al esperado.

2.2 Privacidad en Machine Learning (PPML)

La protección y seguridad de los datos en Machine Learning (ML) es un desafío, especialmente con información sensible. Las técnicas de Privacy-Preserving Machine Learning (PPML) permiten entrenar y usar modelos de ML de manera segura (Guerra-Manzanares, Lechuga Lopez, Maniatakos, & Shamout, 2023).

Principales técnicas PPML:

- **Federated Learning (FL):** Permite entrenamiento colaborativo sin compartir datos locales, solo se intercambian actualizaciones del modelo (McMahan, Moore, Ramage, Hampson, y Aguera y Arcas, 2017). Puede ser vulnerable a ataques de reconstrucción, por lo que se recomienda combinarlo con otros métodos de privacidad (Liu, Xu, y Wang, 2022; Nguyen, y otros, 2022).

- **Homomorphic Encryption (HE)**: Permite operar sobre datos cifrados obteniendo los mismos resultados que si se aplicara a texto plano, sin exponer la información (Acar, Aksu, Uluagac, y Conti, 2018).
- **Secure Multi-Party Computation (SMPC)**: Permite que varias partes calculen una función conjunta sin revelar sus datos, usando principalmente Intercambio de Secretos (SS) como Shamir o aditivo (Singh & Shukla, 2021). Sin embargo, puede haber filtraciones de propiedades globales de los datos en ciertos casos (Yue, y otros, 2021).

En conjunto, estas técnicas permiten aplicar ML en entornos sensibles, balanceando aprendizaje y protección de datos, aunque cada método tiene limitaciones que deben considerarse al diseñar sistemas confiables.

2.3 Privacidad Diferencial (Differential Privacy)

Su objetivo principal es abordar la paradoja de no aprender nada sobre individuos específicos, mientras que se obtiene información útil sobre la población general (Dwork & Roth, 2014).

Se suele incorporar ruido matemático a los datos o a las actualizaciones del modelo, ya sea artificialmente o mediante un optimizador privado diferenciable, antes de transferir las actualizaciones de las entidades al servidor central (Abadi, y otros, 2016). La cantidad de ruido artificial añadido es directamente proporcional al grado de privacidad deseado, en otras palabras, se refiere al presupuesto de privacidad.

Sin embargo, agregar demasiado ruido (es decir, un alto presupuesto de privacidad) puede obstaculizar el aprendizaje y afectar negativamente la precisión del modelo (Tida Sai Venkatesh Chilukoti, Hsu, & Hei, 2022).

La privacidad diferencial (DP) se formaliza mediante una función aleatoria o mecanizada (*M*) que introduce aleatoriedad en el proceso (Dwork & Roth, 2014).

$$Pr\ Pr\ [M(d) \epsilon\ S]\ \leq e^{\in}\ Pr\ Pr\ [M(d')\ \epsilon\ S]\ +\ \delta$$

Parámetros:

- Épsilon (ϵ), parámetro principal que cuantifica la pérdida de privacidad. Un valor de ϵ más pequeño se traduce en una mejor privacidad.
- Delta (δ), representa la probabilidad de que la privacidad ϵ se rompa accidentalmente. Este valor es preferiblemente pequeño, a menudo menor que el inverso de cualquier polinomio en el tamaño de la base de datos.

Propiedades clave de la privacidad diferencial:

- Composición: Si ejecutas múltiples análisis privados sobre los mismos datos, la privacidad se degrada de manera predecible (los ϵ se suman).
- Post-procesamiento: una vez que los datos han sido liberados con DP, cualquier cálculo posterior que se haga con esos datos (sin volver a tocar la base de datos original) siguen siendo privados.

La privacidad diferencial es crucial para permitir el uso de datos sensibles en modelos de Machine Learning.

2.4 Seguridad en Bases de Datos Relacionales

La protección de la información crítica de una empresa en los sistemas de gestión de bases de datos (DBMS), es un pilar fundamental para asegurar la confidencialidad, integridad y disponibilidad de los datos. La confidencialidad de los datos es la principal prioridad de la empresa, esto implica asegurar que los datos, tanto en tránsito como en reposo, sean inaccesibles para usuarios no autorizados, incluso si estos logran acceso físico al servidor o a los medios de almacenamiento.

Con el objetivo de reducir el riesgo como el robo de discos físicos o la filtración de copias de seguridad, se han estandarizado el uso de técnicas de cifrado a nivel de almacenamiento, siendo las más destacadas el Cifrado de datos transparente (TDE) y el uso de algoritmos robustos como AES-256.

2.4.1 Cifrado de Datos Transparente (TDE)

El cifrado de datos transparente cifra de forma transparente los datos en reposo. Detiene los intentos no autorizados del sistema operativo para acceder a los datos de la base de datos almacenados en archivos, sin afectar la forma en que las aplicaciones acceden a los datos mediante SQL (Oracle, s.f.).

Principales beneficios:

- Como administrador de seguridad, puede estar seguro de que los datos confidenciales están encriptados.
- Ayuda a abordar problemas de cumplimiento normativo relacionados con la seguridad.
- No es necesario crear tablas, activadores ni vistas auxiliares para descifrar datos para el usuario o la aplicación autorizados. Los datos de las tablas se descifran de forma transparente para el usuario de la base de datos y la aplicación.
- Los datos se descifran de forma transparente para los usuarios de la base de datos y las aplicaciones que acceden a ellos. Estos usuarios y aplicaciones no necesitan saber que los datos a los que acceden están almacenados cifrados.
- No necesita modificar sus aplicaciones para gestionar los datos cifrados. La base de datos gestiona el cifrado y descifrado de datos.

Los datos se descifran de forma transparente para un usuario autorizado con los privilegios necesarios para verlos o modificarlos. Un usuario o aplicación de la base de datos no necesita saber si los datos de una tabla específica están cifrados en el disco. En caso de robo de los archivos de datos de un disco o medio de copia de seguridad, los datos permanecen protegidos. La *Figura 1*, muestra una descripción general del proceso de cifrado del espacio de tabla TDE.

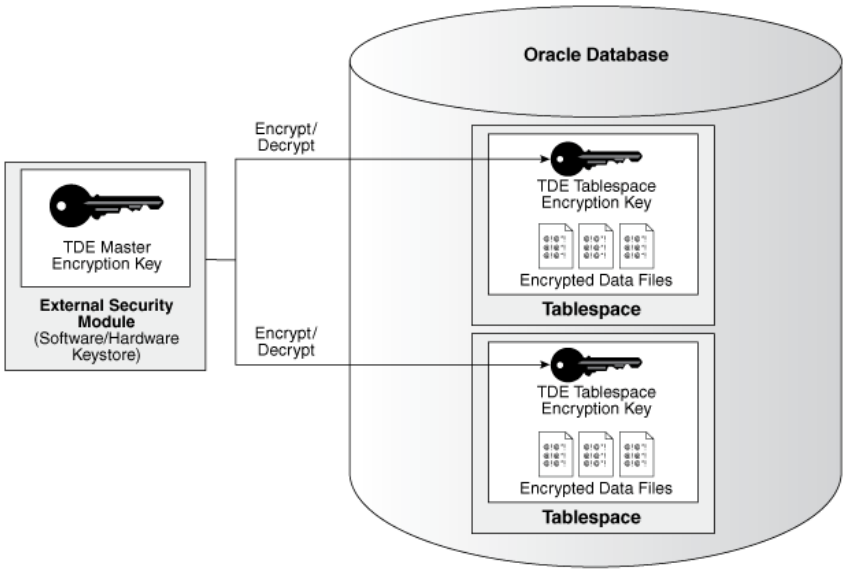

Figura 1. Descripción general del proceso de cifrado del espacio de tabla TDE. Obtenido de "Introduction to Transparent Data Encryption" (Figura 1-1), por Oracle, s.f.

En la arquitectura de seguridad propuesta en la sección 4, TDE actúa como la primera línea de defensa para los "Datos en Reposo" (Capa 1). Su funcionamiento se basa en una arquitectura de dos niveles: una clave maestra (Master Encryption Key) almacenada en un módulo de seguridad externo (Wallet) proteja las claves de cifrado internas que encriptan los archivos de datos físicos. Esto asegura que, en caso de sustracción del servidor o de los archivos de respaldo, la información permanezca ilegible sin la clave maestra correspondiente (Oracle, s.f.).

2.5 Marcos de Gestión y Amenazas

2.5.1 Modelo STRIDE para Análisis de Amenazas

Threat Modeling Tool es un elemento básico del ciclo de vida de desarrollo de seguridad (SDL) de Microsoft. Permite a los arquitectos de software identificar y mitigar los posibles problemas de seguridad en una fase temprana, cuando son relativamente sencillos y poco costosos de resolver. En consecuencia, reduce en gran medida el costo total de desarrollo (Microsoft, 2023).

Microsoft (2023) describe el modelo STRIDE como una metodología que clasifica los distintos tipos de amenazas y simplifica las conversaciones de seguridad, corresponden:

- **Suplantación de identidad (Spoofing):** Consiste en el acceso y uso ilegal de las credenciales de autenticación (usuario y contraseña) de otra persona.
- **Alteración de datos (Tampering):** Se refiere a la modificación malintencionada y no autorizada de la información, ya sea mientras está almacenada (persistencia) o mientras viaja por la red (tránsito).
- **Rechazo (Repudiation):** Ocurre cuando un usuario niega haber realizado una acción y el sistema carece de mecanismos de rastreo (auditoría) para probar lo contrario.
- **Divulgación de información (Information Disclosure):** Es la exposición de datos confidenciales a personas o intrusos que no cuentan con la autorización para verlos.
- **Denegación de servicio (DoS):** Ataques destinados a saturar o inutilizar un sistema, impidiendo que los usuarios legítimos puedan acceder al servicio.
- **Elevación de privilegios (Elevation of Privilege):** Situación crítica donde un usuario común logra obtener permisos administrativos o superiores, comprometiendo la seguridad total del sistema.

3 Análisis de Riesgos y Modelado de Amenazas

Este capítulo analiza las amenazas a las que se enfrenta la organización. Para estructurar este análisis, se adopta un enfoque del modelo de amenazas híbrido. Esta metodología combina el análisis tradicional centrado en los activos (el "qué" se protege), con la categorización formal proporcionada por el framework **STRIDE** (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege).

A lo largo de este capítulo, se desglosa el análisis en tres componentes clave: la identificación de los activos críticos, la definición de los perfiles de los actores de amenaza y la evaluación de escenarios de ataque específicos a través del ciclo de vida completo del dato (Reposo, Tránsito y Uso), poniendo especial énfasis en el riesgo de Divulgación de Información.

3.1 Metodología de Modelado de Amenaza

Para estructurar este análisis, adoptamos un Enfoque de Modelado de Amenazas Híbrido. Esta metodología combina el análisis tradicional centrado en los activos (el "qué" se protege), con la categorización formal proporcionada por el framework STRIDE.

El enfoque híbrido es crucial porque permite rastrear las amenazas a través del ciclo de vida de los datos (Reposo, Tránsito, Uso) y justifica por qué nuestro foco principal es la categoría Information Disclosure, ya que esta se alinea directamente con el objetivo de proteger la Confidencialidad de la propiedad intelectual de la empresa.

STRIDE agrupa las amenazas en seis categorías distintas, donde cada una representa la violación de una propiedad de seguridad deseada, como se ilustra en la Figura 2.

Figura 2. Las seis categorías del modelo de amenazas STRIDE y las propiedades de seguridad que violan.

Como se observa en el diagrama, cada categoría de amenaza (ejemplo: *Spoofing*) se correlaciona con un fallo en una propiedad de seguridad (ejemplo: *Authenticity*).

Dado que el activo crítico de la empresa es su propiedad intelectual (las fórmulas) y su principal preocupación es la fuga de información, el riesgo fundamental es la violación de la Confidencialidad (*Confidentiality*).

Por lo tanto, aunque el framework es completo, nuestro análisis se centrará de manera exhaustiva en la categoría de Divulgación de Información (*Information Disclosure*), ya que es el vector de ataque directo contra la confidencialidad de los datos del proyecto.

3.2 Identificación de Activos Críticos

Para diseñar una arquitectura sólida es de suma importancia identificar y clasificar los activos de información que requieren protección. En el contexto de la empresa, la ventaja competitiva y crucial reside en su propiedad intelectual de sus datos. Por lo tanto, el punto central de este análisis no es proteger toda la infraestructura por igual, sino priorizar aquellos puntos claves cuya filtración generaría un impacto negativo para la organización.

A continuación, se detallan los activos identificados, clasificados por su nivel de criticidad, distinguiendo entre los activos de información pura (datos en reposo) y artefactos del procesamiento (modelos y features).

3.2.1 Activo crítico N°1: Composición Exacta de Fórmulas

La composición exacta de las fórmulas es sin duda el núcleo principal de la propiedad intelectual la cual se debe resguardar con máxima seguridad.

- **Descripción**: La lista completa de ingredientes y sus concentraciones exactas (% en fórmula) para cada producto terminado.
- **Ubicación**: Tabla *TABLA_PRINCIPAL.csv* (o su equivalente en la BD Oracle).
- **Valor**: Es Ultra-Secreto. Es el núcleo de la propiedad intelectual de la empresa. Su fuga permitiría a la competencia replicar fórmulas exitosas, destruyendo la ventaja competitiva de la empresa.

3.2.2 Activo crítico N°2: Identidad de Ingredientes Propietarios

- **Descripción**: La identidad de ciertos códigos, que corresponden a ingredientes exclusivos desarrollados internamente por la empresa.
- **Ubicación**: Tabla *Ingredientes - Proyecto Flash Point.csv* (o su equivalente en la BD Oracle).
- **Valor**: Es confidencial. Su fuga permitiría a proveedores comercializar estos ingredientes con la competencia, erosionando la diferenciación de la empresa en el mercado.

3.2.3 Activo secundario N°1: El Modelo Machine Learning Entrenado

- **Descripción**: Los artefactos del modelo (por ejemplo: *rf_baseline.pkl* o *dp_rf_epsilon_1.0.pkl*) entrenados para predecir el Flash Point.
- **Valor**: Es interno. Un modelo entrenado es una "caja negra" que ha "aprendido" patrones de los datos de entrenamiento. Es susceptible a ataques de Model Inversion que buscan extraer la información con la que fue entrenado.

3.2.4 Activo secundario N°2: Datos Agregados y Features

- **Descripción**: El dataset procesado (features_formulas_XXX.csv) que contiene estadísticas agregadas (por ejemplo: *fp_min, fp_max, conc_max*).
- **Valor**: Es restringido. Aunque están diseñados para ser seguros, estos features aún contienen información estadística derivada de la composición exacta de fórmulas. Un análisis minucioso podría revelar patrones (Amenaza 3).

3.3 Perfilamiento de Actores de Amenaza

Para construir un modelo de amenazas robusto, es fundamental identificar no solo qué activos protegemos, sino también de quién los protegemos. Las amenazas a la propiedad intelectual de la empresa no provienen de una única fuente; pueden originarse tanto dentro como fuera de la organización y ser impulsadas por motivaciones que van desde el espionaje industrial deliberado hasta el error humano accidental. Esta sección define los tres perfiles principales de Actores de Amenaza, analizando sus intenciones, capacidades estimadas y los métodos (vectores de ataque) que probablemente emplearían para comprometer los activos críticos del proyecto.

3.3.1 Actor Interno Malicioso

- **Ejemplos**:
    - Empleado descontento.

    - Administrador de Base de Datos, Administrador de Sistemas con acceso privilegiado.
    - Consultor externo o temporal con acceso de alto nivel a los sistemas sensibles.
- **Intención**: Robo de propiedad intelectual para beneficio personal o venta a la competencia.
- **Capacidad**: Alta. Posee credenciales de acceso legítimas a las bases de datos o a los entornos de analítica (Científico de Datos).
- **Vectores de Ataque**:
    - Abuso de privilegios para extracción masiva de información.
    - Abuso de privilegios de administrador para desactivar auditorías o logs.
    - Copia de datasets crudos o features de ingeniería a dispositivos USB o almacenamiento en la nube personal.
    - Robo de backups de la base de datos.

3.3.2 Actor Interno Negligente

- **Ejemplos**:
    - Un Científico de Datos que, buscando conveniencia, desactiva temporalmente un protocolo de seguridad.
    - Un Desarrollador que *hardcodea* (escribe en el código) credenciales de la Base de Datos en un script de Python.
    - Un empleado de I+D que envía una muestra de fórmula a un proveedor por correo electrónico no cifrado.
    - Un empleado de TI que configura incorrectamente los permisos de un bucket de almacenamiento (por ejemplo: en GCS (Google Cloud Platform o similar), dejándolo accesible públicamente.
    - Cualquier empleado que cae en un ataque de phishing y expone sus credenciales de acceso.

- **Intención**: Ninguna. El actor solo intenta cumplir con sus tareas (ejemplo: entrenar un modelo de machine learning, analizar datos, etc), y comete un error por descuido, falta de capacitación o búsqueda de atajos.
- **Capacidad**: Media. Tiene acceso legítimo a los datos necesarios para su trabajo (que podrían ser los datos crudos si no hay controles).
- **Vectores de Ataque**:
    - Guardar datos sensibles como, por ejemplo: un Dataframe de Pandas en logs por error durante la depuración.
    - Exportar un CSV que contenga los flash point o ingredientes de las fórmulas, a una ubicación insegura, como una carpeta de red compartida o en el escritorio local.
    - Dejar una copia de datos descifrados en memoria sin limpiar, que luego es extraída por otro proceso.
    - Cometer un error en una query SQL que une datos sensibles con no sensibles y los expone.
    - Pérdida de un dispositivo de trabajo (notebook, tablet, etc) que contiene datos no cifrados.

3.3.3 Actores Externos

- **Ejemplos**:
    - La competencia directa de la empresa que esté buscando la propiedad intelectual.
    - Un atacante remoto oportunista (hacker) que escanea redes en busca de vulnerabilidades.
    - Un broker de datos que intenta robar propiedad intelectual para venderla al mejor postor.
    - Bots automatizados que escanean vulnerabilidades conocidas (por ejemplo: SQL Injection, XSS, RCE).
- **Intención**: Espionaje industrial, sabotaje o robo de propiedad intelectual.

- **Capacidad**: Variable. Desde phishing para robar credenciales internas hasta explotación de vulnerabilidades en el servidor de Machine Learning.
- **Vectores de Ataque**:
    - Phishing o Spear Phishing dirigido a empleados (como el Actor Negligente) para robar credenciales.
    - Explotación de vulnerabilidades de red en servicios expuestos (por ejemplo: el servidor donde se aloja el Modelo de Machine Learning, la API de la Base de Datos).
    - Ataques de SQL Injection si la aplicación que consume los datos es vulnerable.
    - Instalación de malware o ransomware en el servidor de Machine Learning para exfiltrar datos o el modelo entrenado.
    - Ataques de fuerza bruta contra cuentas de servicio o usuarios con contraseñas débiles.

3.4 Análisis de Escenarios de Amenaza y Vulnerabilidad

Habiendo establecido los activos críticos que se deben proteger en la sección (3.2 Identificación de activos críticos) y los perfiles de los actores que podrían atacarlos en la sección (3.3 Perfilamiento de actores de amenaza) en esta sección se analizará el "cómo". Es decir, se analizan los métodos, técnicas y escenarios específicos que dichos actores utilizarían para comprometer la confidencialidad de la propiedad intelectual de la empresa. Dado que el activo principal son las fórmulas propietarias, el análisis se centra en el riesgo de Divulgación de Información. Para estructurar esta evaluación, los escenarios de amenaza se agrupan según el ciclo de vida del dato, lo cual se alinea directamente con la arquitectura de defensa multicapa propuesta:

- Amenazas a los **Datos en Reposo** (ejemplo: la tabla *TABLA_PRINCIPAL* en la Base de Datos Oracle).
- Amenazas a los **Datos en Tránsito** (ejemplo: durante una consulta de red).

- Amenazas a los **Datos en Uso** (ejemplo: en memoria durante el entrenamiento o inferidas desde el modelo ya entrenado).

Cada escenario a continuación será analizado en función de los actores involucrados, el impacto potencial y la probabilidad de ocurrencia sin controles, determinando así el nivel de riesgo inherente.

3.4.1 Amenaza N°1: Fuga de Dataset Crudo (Datos en Reposo)

- **Descripción de la Amenaza**: Un actor de amenaza (interno o externo) obtiene acceso y accede al dataset crudo completo, o la tabla Oracle correspondiente (Información sensible como flash point e ingredientes). Esto representa la exposición de aproximadamente 1.1 millones de registros que detallan la composición exacta de las fórmulas.
- **Activos Afectados**:
  - Activo Crítico N°1 (Composición Exacta).
  - Activo Crítico N°2 (Identidad de Ingredientes).
- **Actores Relevantes**:
  - Actor Interno Malicioso (Administrador de Base de Datos, Empleado I+D).
  - Actor Externo (vía compromiso de la Base de Datos).
- **Vectores de Ataque Específicos**:
  - **Exfiltración de Base de Datos**: Un Administrador de Base de Datos (DBA), actuando como Actor Interno Malicioso, utiliza sus credenciales administrativas legítimas para consultar directamente tablas sensibles y exportar el conjunto de resultados a un archivo externo (como un CSV o un dump de base de datos).
  - **Robo de Backups**: Un atacante obtiene acceso al storage donde se almacenan los backups de la base de datos, los cuales podrían no estar encriptados.

    - **Endpoint Inseguro**: Un analista descarga una muestra del dataset crudo a su máquina local, y esa máquina es comprometida.
- **Clasificación (STRIDE)**: I (Information Disclosure - Divulgación de Información).

- **Análisis de Riesgo**:
    - Impacto: CRÍTICO. Aunque no expone la Base de Datos completa, permite la reconstrucción dirigida de fórmulas individuales, llevando a la pérdida total de la propiedad intelectual.
    - Probabilidad: ALTA. Sin controles específicos como la Privacidad Diferencial, los modelos de Machine Learning modernos son altamente vulnerables a estos ataques.
    - Nivel de Riesgo: CRÍTICO.

3.4.2 Amenaza N°2: Reconstrucción de Fórmula desde Modelo Machine Learning (Datos en Inferencia)

- **Descripción de la Amenaza**: Un actor de amenaza que puede ser un Actor Interno Malicioso con conocimiento en Data Science, utiliza técnicas de Machine Learning para "revertir" el modelo entrenado y extraer información altamente sensible sobre los datos originales. Esta amenaza es crítica porque elude la protección criptográfica de infraestructura (TDE/NNE) al atacar directamente el artefacto algorítmico que contiene las propiedades estadísticas aprendidas de las fórmulas.

- **Activos Afectados**:
    - Activo Crítico N°1 (Composición Exacta, inferida).
    - Activo Secundario N°1 (Modelo Machine Learning).
- **Actores Relevantes**:
    - Actor Interno Malicioso (Científico de Datos).
    - Actor Externo (si el modelo es expuesto en una API).

- **Vectores de Ataque Específicos**:
  - **Ataque de Inversión de Modelo (Model Inversion)**: El atacante interroga el modelo de forma sistemática para reconstruir las características de entrada (los features agregados) que mejor representan o maximizan el score de predicción para un target o cluster específico, infiriendo así la composición promedio de una clase de fórmula. Por ejemplo, *"¿Cuál es la composición típica de una fórmula 'No Inflamable'?*".
  - **Ataque de Inferencia de Membresía (Membership Inference)**: El atacante utiliza los *scores* de confianza y la tasa de error del modelo para determinar con alta probabilidad si una fórmula específica fue utilizada o no en el dataset de entrenamiento. Esto permite validar la posesión de una fórmula por parte de la empresa.

La vulnerabilidad inherente al modelo de Machine Learning sin protección algorítmica se ilustra claramente en la Figura 3, que compara el riesgo de fuga de información entre un modelo baseline y el modelo propuesto con Privacidad Diferencial.

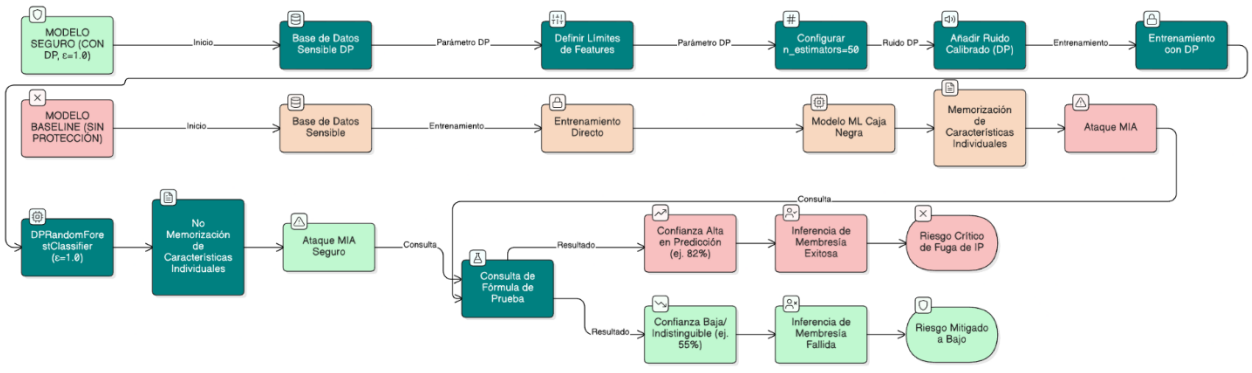

Figura 3. Mecanismo del Ataque de Inferencia de Membresía (MIA) y Mitigación con Differential Privacy.

El diagrama anterior ilustra cómo el Modelo Baseline (sin protección) memoriza y expone la membresía (alta confianza), mientras que el Modelo Seguro (con

Privacidad Diferencial) añade ruido para hacer que los scores sean indistinguibles, neutralizando así el ataque.

- **Clasificación (STRIDE)**: I (Information Disclosure - Divulgación de Información).
- **Análisis de Riesgo**:
    - **Impacto**: CRÍTICO. Aunque no expone la Base de Datos completa, permite la reconstrucción dirigida de fórmulas individuales.
    - **Probabilidad**: ALTA. Sin controles específicos como la Privacidad Diferencial, los modelos de Machine Learning modernos son altamente vulnerables a estos ataques.
    - **Nivel de Riesgo**: CRÍTICO.

3.4.3 Amenaza N°3: Inferencia de Ingredientes desde Features Agregados (Datos en Tránsito/Uso)

- **Descripción de la Amenaza**: Un científico de datos con acceso únicamente al dataset de features agregados logra, mediante técnicas de Inferencia de Atributos o Propiedades, deducir la presencia o ausencia de ingredientes propietarios o críticos. Esta amenaza surge porque incluso los features diseñados para ser seguros contienen información estadística inherente que puede ser revertida o cruzada con conocimiento externo.
- **Activos Afectados**:
    - Activo Crítico N°2 (Identidad de Ingredientes, inferida).
    - Activo Secundario N°2 (Datos Agregados).
- **Actores Relevantes**: Actor Interno Malicioso (Científico de Datos).
- **Vectores de Ataque Específicos**:
    - **Análisis de Valores Extremos**: El atacante analiza los features *fp_min* y *fp_max*. Si el *fp_min* de una fórmula es 20°C, y el atacante sabe (por conocimiento público) que solo 3 ingredientes tienen un FP tan bajo, puede inferir la presencia de uno de esos tres.

    - **Ataque de Vinculación (Linkage Attack)**: El atacante cruza los valores extremos de los features estadísticos (ejemplo*: fp_min*) con bases de datos públicas de propiedades químicas o registros de patentes, lo que permite acotar el conjunto de ingredientes propietarios más probables presentes en la fórmula.
    - **Fuga de Metadatos a través de Features**: El feature *num_imputed_fp* revela cuántos ingredientes tienen un *FP > 93°C* funciona como un canal de baja latencia para la fuga de metadatos, ya que permite clasificar fórmulas en función de su contenido de solventes volátiles, revelando una propiedad sensible que el diseño de features intentó ocultar.
- **Clasificación (STRIDE)**: I (Information Disclosure - Divulgación de Información).
- **Análisis de Riesgo**:
    - **Impacto**: MEDIO. No revela la fórmula completa, pero sí información parcial valiosa (por ejemplo*: "esta fórmula usa un ingrediente muy seguro"*).
    - **Probabilidad**: BAJA. Requiere un esfuerzo estadístico considerable y conocimiento externo. El diseño de features agregados es la primera línea de defensa.
    - **Nivel de Riesgo**: MEDIO.

3.4.4 Amenaza N°4: Acceso No Autorizado Durante Entrenamiento (Datos en Uso/Memoria)

- **Descripción de la Amenaza**: Un proceso malicioso o un atacante con altos privilegios en el servidor de Machine Learning logra acceder y leer los datos descifrados en el espacio de memoria RAM asignado al proceso de entrenamiento. Esta es la única fase del *pipeline* donde los datos, que fueron descifrados para su procesamiento por el CPU, están en texto plano de forma temporal, creando una ventana de riesgo de seguridad de la memoria.
- **Activos Afectados**:

    - Activo Crítico N°1 (Composición Exacta).
    - Activo Crítico N°2 (Identidad de Ingredientes).
- **Actores Relevantes**:
    - Actor Externo (Malware con escalada de privilegios).
    - Actor Interno Malicioso (Administrador de Sistemas).
- **Vectores de Ataque Específicos**:
    - **Captura de Datos en Tiempo de Ejecución (Run-time Data Capture)**: Incluye técnicas como el Memory Scraping, donde un proceso malicioso (ejecutado por un atacante o malware) lee directamente la memoria RAM asignada al proceso Python que contiene el DataFrame de entrenamiento.
    - **Fuga de Datos Persistidos Temporalmente**: El Actor Interno Negligente o un error de configuración resulta en la escritura accidental de los datos en claro a archivos de swap, directorios temporales o logs de debugging no saneados.
    - **Fuga por Volcado de Memoria (Core Dump)**: Si el proceso de entrenamiento del modelo falla, el sistema operativo puede crear un archivo de volcado (Core Dump) en disco, el cual contendría una instantánea de la memoria del proceso, exponiendo datos en texto plano de forma persistente.
- **Clasificación (STRIDE)**: I (Information Disclosure), E (Elevation of Privilege).
- **Análisis de Riesgo (Con Sandbox Básico)**:
    - **Impacto**: ALTO. Aunque el acceso es temporal, permite la captura de datos en claro, eludiendo la encriptación en reposo.
    - **Probabilidad**: BAJA. Asume que el entrenamiento se ejecuta en un entorno de sandbox o contenedor aislado, dificultando el acceso desde otros procesos.
    - **Nivel de Riesgo**: ALTO.

3.5 Controles Propuestos y Estrategias de Mitigación

La identificación de riesgos Críticos y Altos en la Sección 3.4, particularmente en lo referente a la Divulgación de Información, exige una respuesta de seguridad estructurada y robusta. Esta sección detalla los Controles Propuestos y las Estrategias de Mitigación que conforman la Arquitectura de Seguridad Multicapa que se ha definido. La estrategia de defensa no se limita a un único punto de control, sino que se fundamenta en un principio de protección en tres niveles que corresponden al ciclo de vida del dato: la infraestructura (protegiendo los datos en reposo y en tránsito), el diseño de datos (mediante Feature Engineering seguro) y el algoritmo (utilizando la Privacidad Diferencial), asegurando que cada vector de ataque identificado en el análisis anterior sea neutralizado con una contramedida específica.

El diseño del pipeline seguro implementado en este proyecto se basa en el principio de la **Defensa en Profundidad (DIP)**. La *Figura 4* se muestra la Arquitectura de Seguridad Multicapa, demostrando cómo los controles de infraestructura, diseño y algoritmo se integran para neutralizar los riesgos Críticos y Altos identificados en la Sección 3.4. Este enfoque garantiza la protección de los datos en sus tres estados: Reposo (Storage), Tránsito (Network) y Uso (Analytics).

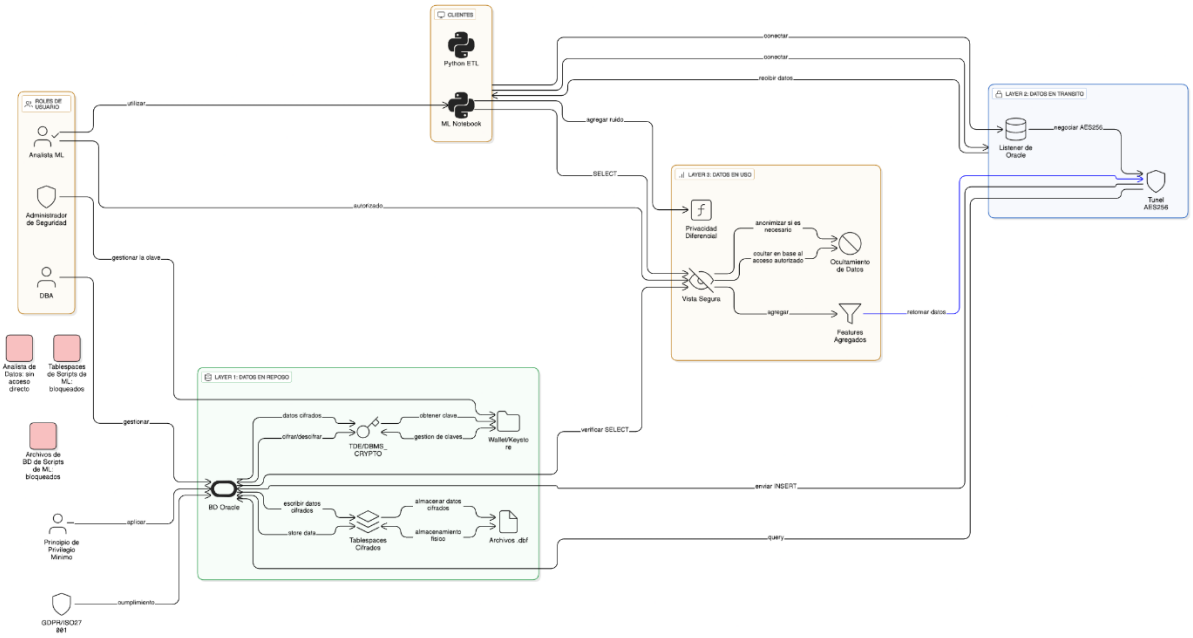

Figura 4. Arquitectura de Seguridad Multicapa para el Pipeline de ML (Protección en Reposo, Tránsito y Uso).

3.5.1 Mitigación de Amenaza 1 (Fuga de Dataset Crudo)

Esta estrategia neutraliza la Amenaza N°1 (Fuga de Dataset Crudo), la cual es la amenaza más directa a los datos en reposo y es principalmente ejecutada por el Actor Interno Malicioso (DBA o el Administrador de Sistemas). La mitigación se centra en la protección de infraestructura (Capas 1 y 2) para hacer inútil el robo del archivo o para restringir el acceso del administrador al nivel de registro.

- **Capa 1 (Reposo): Oracle TDE (Transparent Data Encryption)**. Cifra los archivos de la base de datos a nivel de disco usando AES-256. Esto protege contra el robo de backups o discos físicos.
- **Capa 1 (Acceso)**: **Oracle VPD (Virtual Private Database)**. Aplica seguridad a nivel de fila y columna, asegurando que un usuario (ejemplo: analytics_ml) solo pueda acceder a los registros que le corresponden, previniendo una exfiltración masiva.

- **Capa 2 (Tránsito): Oracle Native Network Encryption (NNE)**. Utiliza TLS 1.3 para cifrar la comunicación entre el cliente y el servidor de la base de datos, previniendo la interceptación de queries y resultados en la red.

3.5.2 Mitigación de Amenaza 2 y 3 (Inferencia y Reconstrucción)

Esta es la defensa algorítmica y de diseño de datos, dirigida al Actor Interno Malicioso (Científico de Datos), quien intenta inferir en la propiedad intelectual desde el modelo o los features. Esta estrategia es crucial, ya que neutraliza la Amenaza N°2 (Reconstrucción de Fórmulas) y la Amenaza N°3 (Inferencia de Atributos), las cuales eluden la protección criptográfica de infraestructura al atacar directamente el algoritmo y el diseño de la vista de datos.

- **Feature Engineering Seguro (Control Primario)**: Se crea una Vista Segura que aplica la lógica de agregación de features del Feature Engineering directamente en la base de datos. Esta vista oculta permanentemente los identificadores de fórmula, los identificadores de ingredientes y sus concentraciones individuales del usuario de Machine Learning.
- **Privacidad Diferencial (DP) (Control Secundario)**: Se implementa el modelo de Machine Learning con Privacidad Diferencial, para agregar "ruido calibrado" durante el entrenamiento. Esto proporciona una garantía matemática contra los ataques de Inferencia de Membresía y Modelos de Inversión, ya que las predicciones del modelo cambian en un límite de si una fórmula individual es eliminada o agregada al dataset. Se recomienda un ϵ $(epsilon)$ $\epsilon \leq 2.0$ para mantener la privacidad alta.

3.5.3 Mitigación de Amenaza 4 (Acceso en Memoria)

Estos controles abordan la Amenaza 4 (Acceso No Autorizado durante Entrenamiento). La meta es proteger los Datos en Uso y prevenir que tanto el Actor Malicioso (vía *Memory Scraping*) como el Actor Negligente (vía logs o archivos temporales) capturen datos en claro que han sido descifrados para su procesamiento por el CPU.

- **Secure Enclave/Sandbox Aislado**: El entrenamiento se ejecuta en un entorno aislado, como un Contenedor Docker.
- **Contenedores Efímeros**: El contenedor se destruye después del entrenamiento, eliminando cualquier dato temporal en memoria o disco.
- **Prohibición de Persistencia**: El script de entrenamiento tiene la regla explícita de NO persistir el DataFrame en disco y NO imprimirlo en logs.
- **Salida Controlada**: La única salida permitida del contenedor es el modelo entrenado (*archivo.pkl*).

3.6 Matriz de Riesgos

La transición del Riesgo Inherente al Riesgo Residual demuestra que la arquitectura propuesta, al combinar la protección de infraestructura (TDE/VPD) con la protección algorítmica (Privacidad Diferencial), logra reducir los riesgos críticos (Amenazas 1 y 2) a un nivel de riesgo operativo Bajo y Aceptable.

La siguiente matriz de la Tabla 2 resume las amenazas identificadas, su riesgo inherente (sin controles) y el riesgo residual (con los controles propuestos).

Tabla 2. Matriz Consolidada de Riesgos, Impacto y Mitigación.

Amenaza Identificada	**Impacto**	**Probabilidad (Sin Controles)**	**Riesgo Inherente**	**Controles Clave Propuestos**	**Probabilidad (Con Controles)**	**Riesgo Residual**
1. Fuga de Dataset Crudo (Datos en Reposo)	Crítico	MEDIA	Alto	Oracle TDE, VPD, NNE	Muy baja	Bajo
2. Reconstrucción desde Modelo ML (Datos en Inferencia)	Crítico	ALTA	Crítico	Vistas Seguras + Privacidad Diferencial (DP)	Muy baja	Bajo
3. Inferencia desde Features Agregados (Datos en Tránsito)	Medio	BAJA	Medio	Ingeniería de Features + DP	Muy baja	Bajo

4. Acceso No Autorizado (En Memoria)	Alto	Baja	Alto	Secure Sandbox (Docker), Contenedores Efímeros	Muy baja	Bajo

Fuente: Elaboración Propia, 2025.

Esta matriz presenta las cuatro amenazas principales identificadas a lo largo del ciclo de vida de los datos. Contrapone el Riesgo Inherente (el nivel de riesgo sin los controles propuestos) con el Riesgo Residual (el nivel de riesgo una vez implementada la Arquitectura de Seguridad Multicapa y las técnicas de Privacidad Diferencial). El objetivo del proyecto es demostrar la reducción de todos los riesgos Críticos y Altos a un nivel Aceptable/Bajo, asegurando la viabilidad del proyecto de Machine Learning sobre la propiedad intelectual de la empresa.

4 Diseño de la Arquitectura de Seguridad

En este apartado se describe a detalle la implementación técnica de la estrategia de defensa. La arquitectura está compuesta por cuatro capas que protegen los datos desde su almacenamiento hasta su uso analítico.

La Figura 4 ilustra la interacción entre los componentes y las tecnologías de seguridad aplicadas en cada capa.

4.1 Principios de Diseño

La arquitectura de seguridad no es un componente aislado, sino una característica intrínseca del sistema. El diseño se centra en tres principios fundamentales que garantizan la robustez y la resiliencia de la solución:

- **Defensa en Profundidad (Defense in Depth)**: Se implementan múltiples capas de controles de seguridad redundantes. Si un atacante logra vulnerar una capa (por ejemplo, la seguridad perimetral de red), se encontrará con barreras adicionales

(como el cifrado en disco o las vistas restringidas) que impedirán el acceso a los activos críticos.

- **Principio de Mínimo Privilegio (PoLP)**: Se restringen los permisos de acceso de usuarios y procesos al nivel mínimo necesario para realizar sus funciones legítimas.
- **Seguridad por Diseño (Security by Design)**: La privacidad y la protección de datos se integran en la arquitectura de la base de datos y el código desde la concepción del proyecto, en lugar de aplicarse como parches correctivos posteriores.

4.2 Capa 1: Seguridad en Reposo (Storage)

El objetivo de esta capa es proteger la confidencialidad, integridad y disponibilidad de los datos almacenados en las tablas y respaldos de la base de datos frente a accesos no autorizados, robo de medios y fugas provenientes de dispositivos de almacenamiento o copias de respaldo.

- **Tecnología**: Se utiliza un esquema de cifrado híbrido basado en el estándar **AES-256** (*Advanced Encryption Standard*), reconocido por su robustez ante ataques de fuerza bruta.
- **Implementación**: Dado el uso de Oracle Database Standard Edition, la implementación se apoya en el paquete *DBMS_CRYPTO* y funciones de *Transparent Data Encryption* (TDE) donde es aplicable. Esto asegura que los datos sensibles, como las fórmulas químicas en la tabla *TABLA_PRINCIPAL*, se escriban en disco exclusivamente en formato cifrado.
- **Gestión de Claves**: Se implementa una separación estricta de secretos mediante el uso de *Oracle Wallet* o *Keystores*. La clave maestra de cifrado (*Master Key*) se almacena en un módulo de seguridad externo a la base de datos, impidiendo que un atacante pueda descifrar la información incluso si logra acceso total a los archivos de datos.

4.3 Capa 2: Seguridad en Tránsito (Network)

Esta capa asegura el canal de comunicación entre el cliente y el servidor de base de datos, mitigando ataques de interceptación (*sniffing*) y manipulación de paquetes (*Man-in-the-Middle*).

- **Tecnología:** Se implementa **Oracle Native Network Encryption (NNE)**.
- **Implementación:** Se fuerza la encriptación de todas las conexiones entrantes mediante la configuración del archivo *sqlnet.ora* en el servidor. Se especifican los algoritmos **AES-256** para el cifrado del tráfico y **SHA-256** para la validación de integridad (Checksum). Esto garantiza que los datos viajen por la red en un túnel seguro, siendo ilegibles para cualquier actor que capture el tráfico de red.

4.4 Capa 3: Seguridad en Uso (Analytics)

Para proteger los datos mientras son consultados y procesados, se establece una interfaz de acceso controlada que abstrae la complejidad del cifrado y aplica anonimización lógica.

- **Tecnología:** Vistas Seguras (*Secure Views*) y Control de Acceso Basado en Roles (RBAC).
- **Implementación:**
    - **Vistas Seguras:** Se despliega la vista *ML_TRAINING_DATA_SECURE*, la cual actúa como una barrera lógica. Esta vista implementa técnicas de supresión de identificadores directos, permitiendo el análisis estadístico sin revelar la composición exacta de las fórmulas.
    - **Segregación de Roles (RBAC):** Se define un rol específico, *analytics_ml*, con permisos de solo lectura sobre las vistas seguras y denegación explícita de acceso a las tablas base cifradas. Esto operacionaliza el principio de separación de responsabilidades.

4.5 Capa 4: Privacidad Algorítmica

La última capa de defensa se aplica durante la fase de entrenamiento del modelo, protegiendo contra ataques de inferencia que intentan reconstruir los datos de entrenamiento a partir de las predicciones del modelo.

- **Tecnología:** Privacidad Diferencial (*Differential Privacy*) utilizando bibliotecas especializadas como *diffprivlib*.
- **Implementación:** Se inyecta ruido estocástico (Laplaciano o Gaussiano) calibrado durante el proceso de aprendizaje.
- **Presupuesto de Privacidad:** Se configura el parámetro Épsilon (ϵ) para cuantificar y limitar la pérdida máxima de privacidad permitida. Un valor bajo de Épsilon (ϵ), garantiza matemáticamente que la salida del modelo no varíe significativamente ante la presencia o ausencia de un registro individual, neutralizando ataques de inferencia de membresía.

5 Implementación Técnica e Ingeniería de Datos

En esta sección se documenta el despliegue práctico de la arquitectura de seguridad diseñada. En esta fase se utiliza **Oracle Databse 19c Enterprise Edition**, lo que permite delegar funciones criptográficas directamente a la base de datos. Dicha implementación garantiza que el cifrado sea transparente, robusto y cumpla con los estándares requeridos.

5.1 Configuración del Entorno Seguro

Para gestionar la infraestructura de seguridad de manera centralizada y controlada, se estableció un entorno de desarrollo basado en **Visual Studio Code** con la extensión oficial *Oracle SQL Developer*. Esta configuración permite la ejecución de bloques PL/SQL y la administración de *Wallets* sin depender de herramientas de línea de comandos heredadas,

facilitando la clasificación de roles entre el administrador de seguridad y el desarrollador o analista.

5.2 Implementación de Seguridad en Tránsito (Capa de Red)

Antes de cargar los datos es necesario asegurar el canal de comunicación para prevenir ataques de intercepción. Se configura **Oracle Native Network Encryption (NNE)** modificando el archivo de parámetros de red *sqlnet.ora* (*Figura 5*) en el servidor.

Los protocolos criptográficos para todas las conexiones de entrada y salida:

- **Cifrado:** AES256 (Estándar de Cifrado Avanzado de 256 bits).
- **Integridad:** SHA256 (Secure Hash Algorithm) para la validación de *checksums*, garantizando que los paquetes no sean alterados en tránsito.
- **Política:** REQUIRED, rechazando cualquier intento de conexión no cifrada.

```
# FORZAR ENCRIPTACIÓN DE RED (AES-256)
SQLNET.ENCRYPTION_SERVER = REQUIRED
SQLNET.ENCRYPTION_TYPES_SERVER = (AES256)

# FORZAR INTEGRIDAD (SHA-256)
SQLNET.CRYPTO_CHECKSUM_SERVER = REQUIRED
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (SHA256)
```

Figura 5. Configuración del servidor. Elaboración propia, 2025.

5.3 Implementación de Seguridad en Reposo

La protección de la propiedad intelectual reside en la implementación de **Transparent Data Encryption (TDE)**. A diferencia del cifrado a nivel de columna, esta estrategia cifra bloques de datos completos a nivel físico en el disco.

5.3.1 Gestión del Keystore (Billetera Digital)

Se aprovisionó un *Software Keystore* (Billetera) para custodiar la Llave Maestra de Cifrado, asegurando la separación lógica entre los datos y las credenciales de acceso.

1. **Creación:** Se inicializa el *Keystore* en una ruta protegida del sistema operativo, estableciendo una contraseña administrativa robusta.
2. **Apertura:** Se configura el modo de operación para permitir la apertura del *Keystore* (*STATUS: OPEN*) y habilitar las operaciones criptográficas del motor.

5.3.2 Establecimiento de la Llave Maestra

Se genera y activa la Master Encryption Key utilizando el comando *ADMINISTER KEY MANAGEMENT SET KEY* (*Figura 6*). Esta llave es la raíz de confianza del sistema y se utiliza para proteger las llaves de cifrado de los *tablespaces* individuales.

```
ADMINISTER KEY MANAGEMENT SET KEY
IDENTIFIED BY "WalletPass123"
WITH BACKUP USING 'backup_inicial_cramer';
```

Figura 6. Comando para la creación de la llave maestra. Elaboración propia, 2025.

5.3.3 Segregación de Datos: Tablespace Cifrado

Para aislar la información sensible de los datos, se crea un espacio de tablas dedicado denominado *FORMULAS_ENCRYPTED* (*Figura 7*).

```
CREATE TABLESPACE FORMULAS_ENCRYPTED
DATAFILE 'C:\app\oracle\oradata\orcl\formulas_secure01.dbf'
SIZE 500M
AUTOEXTEND ON NEXT 500M
MAXSIZE 10G
ENCRYPTION USING 'AES256' DEFAULT STORAGE(ENCRYPT);
```

Figura 7. Creación de tablespace cifrada. Elaboración propia, 2025.

- **Configuración:** Este *tablespace* se configura con cifrado predeterminado utilizando el algoritmo AES256.
- **Efecto:** Cualquier tabla creada dentro de este espacio (como *TABLA_PRINCIPAL* o *INGREDIENTES*) hereda automáticamente la protección. Los archivos de datos

físicos resultantes (.dbf) son ilegibles a nivel de sistema operativo, neutralizando el riesgo de robo físico o clonación de discos.

5.4 Ingesta Transparente de Datos (ETL)

El script de carga (*01_etl_ingestion.ipynb*) realiza la limpieza de datos (conversión de formatos numéricos) y ejecuta inserciones SQL estándar.

- **Transparencia**: No es necesario invocar funciones criptográficas manuales en el código Python. El motor de base de datos cifra automáticamente los datos en memoria antes de escribirlos en los bloques del disco del tablespace *FORMULAS_ENCRYPTED.*
- **Beneficio**: Esto reduce la deuda técnica y minimiza el riesgo de errores de implementación en el código ETL, delegando la responsabilidad de la seguridad a la base de datos, que está optimizada para esta tarea.

5.5 Capa de Aplicación: Vistas Seguras y RBAC

Para disponer de los datos a los modelos de Machine Learning sin exponer la información cruda, se implementa la capa de "Datos en Uso".

5.5.1 Construcción de Vistas Seguras

Se despliega la vista *ML_TRAINING_DATA_SECURE*. Esta vista actúa como una interfaz de solo lectura que implementa una estrategia de abstracción de datos mediante la supresión de identificadores directos (códigos de ingredientes) y la agregación estadística (cálculo de propiedades físicas promedio).

5.5.2 Control de Acceso Basado en Roles (RBAC)

Se crea el usuario *analytics_ml* con un perfil de seguridad restringido (*Figura 8*, *Figura 9* y *Figura 10*):

- **Permisos:** Se otorga acceso *SELECT* exclusivamente sobre la vista segura.

- **Restricciones:** Se rechaza explícitamente el acceso a las tablas base (*TABLA_PRINCIPAL*) y se revocaron privilegios administrativos (*SELECT ANY TABLE*), asegurando que el analista no pueda eludir la capa de abstracción de la vista.

```
CREATE USER analytics_ml IDENTIFIED BY "PasswordAnalista123";
```

Figura 8. Creación del usuario de servicio. Elaboración propia, 2025.

```
GRANT CREATE SESSION TO analytics_ml;
GRANT SELECT ON SYSTEM.ML_TRAINING_DATA_SECURE TO analytics_ml;
```

Figura 9. Asignación de permisos al usuario. Elaboración propia, 2025.

```
ALTER USER analytics_ml QUOTA 0 ON SYSTEM;
ALTER USER analytics_ml QUOTA 0 ON FORMULAS_ENCRYPTED;
```

Figura 10. Restricción de escritura al usuario. Elaboración propia, 2025.

5.6 Implementación de Privacidad Diferencial (Cliente)

Como última línea de defensa antes del entrenamiento del modelo, se implementa un mecanismo de privacidad algorítmica en el entorno Python (*03_Differential_Privacy.ipynb*).

- **Librería:** Se utilizó *diffprivlib* de IBM para integrar mecanismos de perturbación en los datos.
- **Presupuesto de Privacidad:** Se configuran varios valores de ruido llegando al más óptimo de $\epsilon = 1.5$, inyectando ruido Laplaciano controlado en los datos de entrenamiento. Esto garantiza matemáticamente que el modelo resultante no memorice registros individuales, protegiendo contra ataques de ingeniería inversa a modelos expuestos públicamente.

6 Metodología Experimental

A continuación, se describe el diseño del experimento de Machine Learning, detallando el tratamiento de los datos, la selección de algoritmos y las métricas de evaluación utilizadas para validar la hipótesis planteada. A diferencia del capítulo anterior, que se centró en la infraestructura de seguridad, esta sección se enfoca en la estrategia analítica para equilibrar la precisión predictiva con la privacidad diferencial.

6.1 Definición del Problema de Machine Learning

El problema se aborda desde una doble perspectiva para satisfacer las necesidades operativas de la empresa:

1. **Regresión (Valor Exacto):** El objetivo principal es predecir el valor numérico continuo del Punto de Inflamación (Flash Point en $°C$) de una mezcla química, basado en las propiedades de sus ingredientes constituyentes.
    - **Variable Objetivo (Y):** *Flash Point PT* ($°C$).
2. **Clasificación (Riesgo GHS):** Para efectos de seguridad industrial y transporte, la predicción numérica se discretiza en categorías de riesgo según el Sistema Globalmente Armonizado (GHS).
    - **Clase 0 (Muy Inflamable):** $FP < 23°C$.
    - **Clase 1 (Inflamable):** $23°C \leq FP \leq 60°C$.
    - **Clase 2 (No Inflamable):** $FP < 60°C$.

6.2 Dataset y Preprocesamiento

El conjunto de datos utilizado proviene de la base de datos histórica de formulaciones de la empresa, específicamente de las tablas *TABLA_PRINCIPAL* e *INGREDIENTES*, abarcando un total aproximado de 1.1 millones de registros transaccionales. Estos registros contienen la composición detallada de mezclas químicas y sus propiedades físicas medidas experimentalmente a lo largo de décadas de operación.

Sin embargo, dada la naturaleza sensible de la información cruda y la necesidad de adaptar estos datos transaccionales para un modelo de aprendizaje supervisado, fue necesario implementar una cadena de procesamiento estricta. Este flujo de trabajo transforma los datos desde su estado original en la base de datos Oracle hasta convertirse en vectores numéricos normalizados aptos para el entrenamiento con Privacidad Diferencial.

La arquitectura de este proceso de transformación se detalla en la Figura 11, la cual ilustra las cuatro etapas secuenciales de limpieza, imputación, ingeniería de características y normalización.

Figura 11. Pipeline de Transformación de Datos.

Como se observa en el diagrama, el proceso inicia con la extracción segura, seguida de la imputación de valores faltantes en los ingredientes (paso crítico para no perder información histórica), y culmina con la agregación estadística (*Feature Engineering*) y el escalado *MinMax*, requisito técnico indispensable para acotar la sensibilidad global del algoritmo de privacidad.

6.2.1 Limpieza e Imputación

Se realiza un proceso de limpieza para eliminar registros con *Flash Point* nulo o inconsistente (valores negativos físicamente imposibles). Para los ingredientes sin datos de inflamabilidad reportados, se aplicaron técnicas de imputación basadas en familias químicas similares, generando el feature de control *num_imputed_fp* para trazar la incertidumbre introducida.

6.2.2 Ingeniería de Características (Feature Engineering)

Dado que el uso directo de la lista de ingredientes crudos presentaba un alto riesgo de re-identificación de fórmulas propietarias, se diseñaron features agregados capaces de capturar las propiedades fisicoquímicas de la mezcla sin revelar su composición exacta. Esta estrategia de abstracción permite entrenar el modelo sobre comportamientos termodinámicos generales en lugar de recetas específicas.

El proceso de selección de variables prioriza tres dimensiones críticas: Propiedades Térmicas Ponderadas (como el FP ponderado por concentración), Estadísticos Descriptivos de la distribución de ingredientes (mínimos, máximos y desviaciones) y la Composición de Riesgo (porcentaje de alcoholes y gases). A continuación, la Tabla 3 presenta el diccionario definitivo de las variables seleccionadas, detallando su tipo de dato y la justificación técnica de su inclusión en el modelo seguro.

Tabla 3. Diccionario de Variables

Nombre del Feature	**Tipo de Dato**	**Descripción y Justificación Técnica**
fp_concentration_weighted	Float	Punto de Inflamación Ponderado. Es el promedio de los Flash Points de todos los ingredientes, ponderado por su concentración en la mezcla. *Justificación:* Representa la "termodinámica base" de la fórmula; es el predictor lineal más fuerte del FP final.
fp_mean	Float	Promedio Simple de Flash Point. La media aritmética de los puntos de inflamación de los componentes, sin considerar cantidades. *Justificación:* Ayuda al modelo a detectar si la presencia de un solo ingrediente muy volátil (aunque sea en poca cantidad) afecta la seguridad global.
pct_alcohol	Float	Porcentaje de Alcoholes. Suma de las concentraciones de todos los ingredientes clasificados químicamente como alcoholes.

		Justificación: Los alcoholes suelen tener puntos de inflamación bajos y comportamientos azeotrópicos que reducen drásticamente el FP de la mezcla.
pct_gaseoso	Float	Porcentaje de Ingredientes Gaseosos. Concentración total de componentes con alta presión de vapor. *Justificación:* Captura la volatilidad extrema. Incluso trazas pequeñas de gases disueltos pueden volver una mezcla "Muy Inflamable" (Clase 0).
pct_ing_muy_inflamable	Float	Carga de Alta Inflamabilidad. Proporción total de la fórmula compuesta por ingredientes que individualmente tienen un $FP < 23°C$. *Justificación:* Métrica directa de riesgo acumulado; cuantos más ingredientes peligrosos, mayor probabilidad de que el producto final lo sea.
inter_fp_alcohol	Float	Interacción FP-Alcohol. Feature de interacción creado matemáticamente (producto o ratio) entre la temperatura ponderada y el contenido de alcohol. *Justificación:* Captura efectos no lineales donde el alcohol "potencia" la inflamabilidad más allá de la simple suma de sus partes.
fp_min / fp_max / fp_std	Float	Estadísticos de Distribución. Valores mínimos, máximos y desviación estándar de los FP de los componentes. *Justificación:* Proveen al modelo información sobre el rango de volatilidad y la heterogeneidad de la mezcla

Fuente: Elaboración Propia, 2025.

Es importante destacar que este conjunto de variables cumple con el principio de privacidad por diseño: al utilizar únicamente agregaciones estadísticas y proporciones totales, se vuelve computacionalmente inviable para un atacante reconstruir la lista

original de ingredientes o sus cantidades exactas a partir de estos vectores de entrenamiento.

6.2.3 Normalización para Privacidad Diferencial

Un requisito crítico para la implementación de algoritmos de Privacidad Diferencial es acotar la sensibilidad de los datos. Por ello, todas las variables de entrada (X) fueron normalizadas en un rango estricto de [0, 1] utilizando *MinMaxScaler*. Esto permite definir los "límites" (*bounds*) necesarios para que el algoritmo calcule correctamente la cantidad de ruido Laplaciano a inyectar sin destruir la utilidad del modelo.

6.3 Modelos Seleccionados

Se diseña una estrategia comparativa utilizando dos enfoques de modelado:

6.3.1 Modelos Baseline (Sin Privacidad)

Para establecer el límite superior de rendimiento posible (*Benchmark*), se entrenan modelos tradicionales sin restricciones de privacidad:

- **Random Forest Regressor:** Seleccionado por su capacidad para capturar no-linealidades y su robustez ante *outliers*.
- **XGBoost:** Utilizado por su eficiencia en competiciones de datos tabulares y manejo de valores perdidos.

La justificación radica en que estos modelos permiten establecer el potencial predictivo intrínseco de los datos originales. Al entrenar sobre la información pura (sin ruido añadido), se obtiene un punto de referencia ideal que permite cuantificar con exactitud cuánta capacidad predictiva se pierde al aplicar las capas de privacidad.

6.3.2 Modelo Seguro (Con Privacidad Diferencial)

Para el modelo productivo, se seleccionó Random Forest implementado a través de la librería *diffprivlib* de IBM.

- **Selección del Algoritmo:** A diferencia de XGBoost, que es un algoritmo secuencial *(Boosting*) donde el consumo del presupuesto de privacidad se acumula

rápidamente en cada iteración, Random Forest permite paralelizar la construcción de árboles, optimizando el uso del presupuesto de privacidad (ϵ) y manteniendo una mejor estabilidad ante la inyección de ruido.

6.4 Métricas de Evaluación

El éxito del proyecto se mide en dos dimensiones contrapuestas: Utilidad vs Privacidad.

6.4.1 Métricas de Utilidad (Desempeño)

- **R^2 (Coeficiente de Determinación):** Para medir qué tan bien el modelo explica la varianza de los datos de temperatura.
- **RMSE (Root Mean Squared Error):** Para cuantificar el error promedio en grados Celsius.
- **Accuracy (Exactitud) y Matriz de Confusión:** Utilizados para evaluar la clasificación de riesgo GHS, con especial foco en los Falsos Negativos Críticos (predecir por ejemplo "No Inflamable" cuando es "Inflamable").

6.4.2 Métrica de Privacidad

- **Épsilon (ϵ):** Se utiliza (ϵ) como el parámetro de control para cuantificar la pérdida de privacidad. El estudio experimental evalúa el desempeño del modelo en un rango de $\epsilon \in [0.1, 10]$, buscando un punto de equilibrio donde se maximice el *accuracy* manteniendo $\epsilon \leq 2.0$ (meta de privacidad robusta).

7 Resultados y Discusión

7.1 Validación de Seguridad (Proof of Concept)

La evaluación del desempeño de los modelos predictivos requiere como prerrequisito importante la validación de la infraestructura de seguridad implementada. Esta sección proporciona la evidencia técnica que confirma la correcta implementación y

funcionamiento de la Capa 1: Seguridad en Reposo mediante Oracle Transparent Data Encryption (TDE).

El primer componente crítico de la arquitectura es la separación de la llave maestra de cifrado respecto a los datos. Como se observa en la Figura 12, se verificó el estado de la Billetera de Oracle mediante la vista dinámica *V$ENCRYPTION_WALLET*.

La billetera se encuentra en estado `OPEN`, permitiendo a la base de datos realizar operaciones de cifrado y descifrado en tiempo real de manera transparente para la aplicación.

WRL_TYPE	WRL_PARAMETER	STATUS	WALLET_TYPE	WALLET_ORDER	KEYSTORE_MODE	FULLY_BACKED_UP	CON_ID
FILE	D:\APP\VICTO\ADMIN\ORCL\WALLET\	OPEN	AUTOLOGIN	SINGLE	NONE	NO	1
FILE	(null)	OPEN	AUTOLOGIN	SINGLE	UNITED	NO	2
FILE	(null)	OPEN_NO_MASTER_KEY	AUTOLOGIN	SINGLE	UNITED	UNDEFINED	3

Figura 12. Estado del Oracle Wallet. Elaboración propia, 2025.

La protección física de los datos se valida consultando la vista del sistema *V$ENCRYPTED_TABLESPACES*. Esta prueba es fundamental para garantizar que los archivos de datos almacenados en el disco duro del servidor estén ilegibles sin la llave maestra.

Como se muestra en la Figura 13, el tablespace dedicado *FORMULAS_ENCRYPTED* tiene el estado *ENCRYPTED = YES*.

	TABLESPACE_NAME	ENCRYPTED
1	SYSTEM	NO
2	SYSAUX	NO
3	UNDOTBS1	NO
4	TEMP	NO
5	USERS	NO
6	FORMULAS_ENCRYPTED	YES

Figura 13. Verificación de Tablespaces Cifrados. Elaboración propia, 2025

7.2 Resultados del Modelo Baseline (Sin Privacidad)

Se entrenan modelos de aprendizaje supervisado utilizando el dataset completo sin aplicar ninguna técnica de perturbación de los datos. Los resultados entregados en la Figura 14 representa el rendimiento ideal que se obtendría si no existieran restricciones de privacidad, sirviendo como punto de referencia para medir la pérdida de utilidad de los modelos en las fases posteriores de estudio.

Para dicho estudio se evaluaron dos algoritmos de ensamble robusto, Random Forest y XGBoost, configurados para una tarea de regresión sobre la variable objetivo "*Flash_Point_PT*".

Como se observa en la Figura 14, ambos modelos demuestran una capacidad predictiva excepcional sobre los datos de prueba. La distribución de los datos muestra una linealidad estrecha sobre la diagonal ($y = x$), indicando una fuerte correlación entre los valores predichos y los valores experimentales reales.

El modelo baseline de Random Forest, alcanza un coeficiente de determinación (R^2) de 0.892 y un error cuadrático medio (RMSE) de 9.38°C, mientras que el modelo XGBoost, obtuvo resultados superiores con un R^2 de 0.893 y un RMSE de 9.34°C.

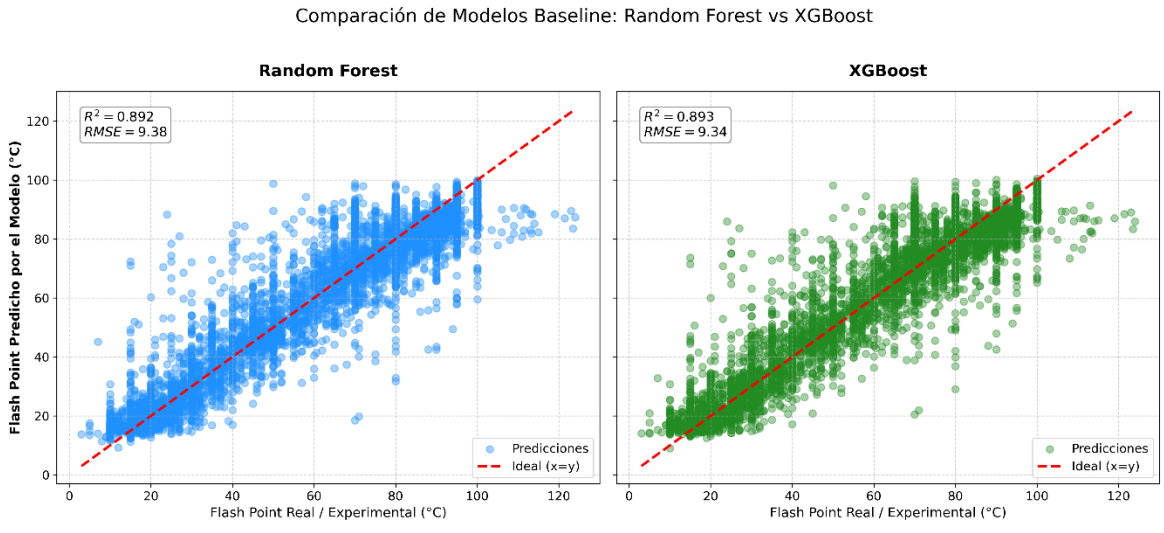

Figura 14. Correlación entre Flash Point Real vs. Predicho en modelos Baseline (Random Forest y XGBoost). Elaboración propia, 2025.

Desde la perspectiva industrial, un error medio cercano a los 9°C es altamente aceptable para una herramienta de screening inicial, permitiendo descartar formulaciones peligrosas sin necesidad de pruebas físicas inmediatas.

7.2.1 Evaluación de Seguridad (Clasificación GHS)

Más allá de evaluar la precisión numérica, es crítico evaluar la capacidad de los modelos para clasificar correctamente el riesgo de inflamabilidad según la normativa. La Figura 15 presenta las matrices de confusión para tres categorías de riesgo:

- Muy inflamable (< 23°C).
- Inflamable (23-60°C).
- No inflamable (>60°C).

El tema central de evaluación es la seguridad, un error crítico se define como la clasificación de una sustancia “Muy inflamable” o “Inflamable” como “No inflamable”, es decir falsos negativos de alto riesgo.

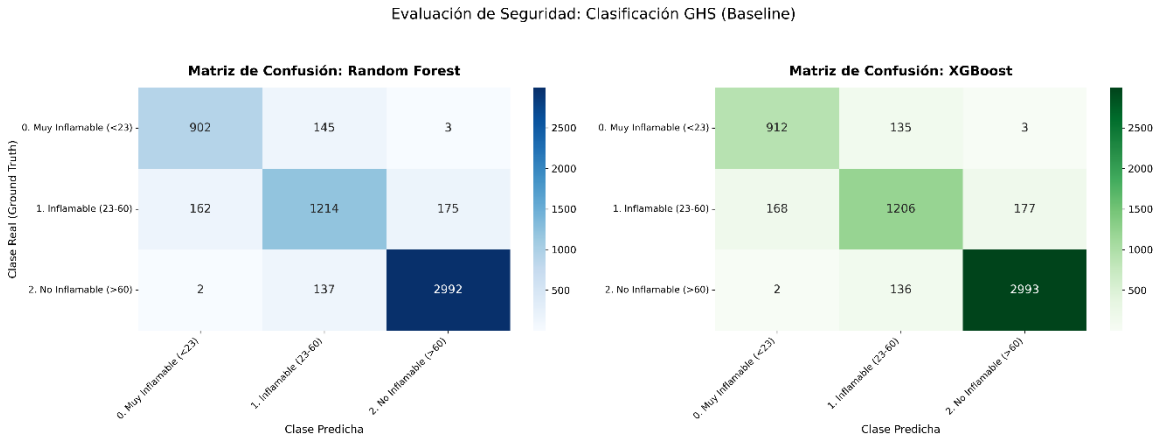

Figura 15. Matrices de confusión de los modelos Baseline. Elaboración propia, 2025.

Los resultados obtenidos muestran que ambos modelos exhiben un rendimiento destacado, siendo la clase "No Inflamable" la mejor clasificada con aproximadamente 2993 predicciones correctas. XGBoost supera ligeramente a Random Forest en la categoría "Muy Inflamable" (912 vs 902 aciertos), mientras que Random Forest obtiene mejores resultados en "Inflamable" (1214 vs 1206 aciertos). En la categoría más peligrosa (Muy

inflamable), ambos modelos clasificaron incorrectamente solo 3 muestras como No inflamables, de un total de más de 1000 muestras reales en esa clase.

La diagonal principal muestra una alta densidad, lo que confirma que la gran mayoría de las predicciones son correctas.

7.2.2 Selección del Modelo de Referencia

Aunque XGBoost presenta mejores resultados, se selecciona Random Forest como el modelo base para las siguientes etapas de investigación con privacidad diferencial. Esta decisión se basa en la compatibilidad con la privacidad diferencial puesto que Random Forest permite inyectar ruido sin degradar la convergencia del modelo. Por el contrario, XGBoost consume el "parámetro de privacidad" de manera acumulativa muy rápido, haciendo que la implementación de la Privacidad Diferencial sea computacionalmente ineficiente y propensa a pérdida severa de utilidad.

Por otro lado, la biblioteca estándar utilizada para la implementación, *diffprivlib* (IBM), ofrece soporte nativo y robusto para *RandomForestClassifier*, lo que garantiza una integración fluida con la arquitectura de seguridad propuesta, a diferencia de XGBoost que requeriría implementaciones experimentales no certificadas para este entorno.

7.3 Evaluación de Impacto: Privacidad Diferencial

La evaluación del impacto del modelo mediante la implementación de diferentes valores de la privacidad diferencial, con el objetivo de determinar el valor óptimo que equilibre la protección de datos con la utilidad analítica.

La Figura 16 ilustra la curva de aprendizaje del modelo bajo diferentes niveles de privacidad. En el eje horizontal se representa el valor de (ϵ) en escala logarítmica (valores menores indican mayor privacidad y mayor ruido), y en el vertical la precisión global del modelo.

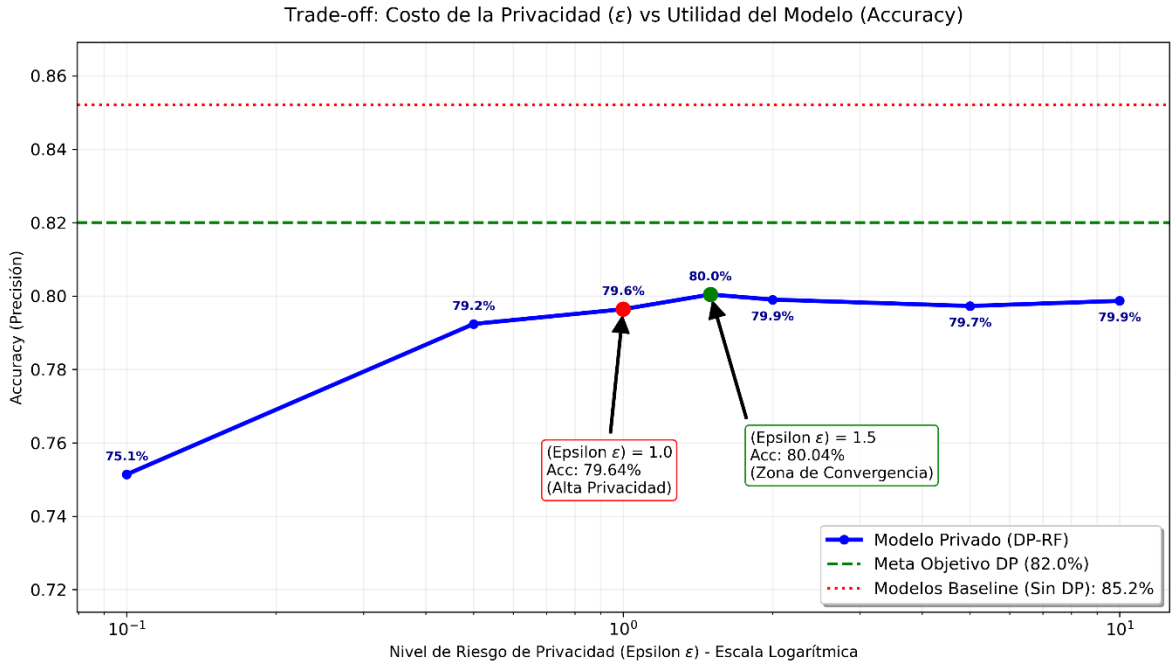

Figura 16. Trade-off entre Privacidad (ϵ) y Utilidad (Accuracy). Elaboración propia, 2025.

Se observan tres zonas de comportamiento distintas:

- **Zona de Alta Privacidad** ($\epsilon < 1.0$): El modelo sufre una degradación significativa, con una precisión que desciende hasta 75.1%. El ruido añadido es tan alto que enmascara las señales predictivas débiles de las fórmulas químicas.
- **Zona de Convergencia** ($\epsilon \sim 1.5$): Se identifica un punto de inflexión o "codo" en la curva. Al aumentar ϵ a 1.5, el modelo recupera gran parte de su capacidad predictiva, alcanzando un 80.04%. Este punto representa el equilibrio ideal entre garantías de privacidad robustas y utilidad práctica del modelo.
- **Zona de Rendimientos Decrecientes** ($\epsilon > 1.5$): Aumentar el parámetro de privacidad más allá de este punto aporta mejoras mínimas e incluso contraproducentes en la precisión. Para $\epsilon = 2.0$ se observa 79.9%, $\epsilon = 5.0$ registra 79.7%, y $\epsilon = 10.0$ alcanza 79.9%. Esta estabilización sugiere que el modelo ha alcanzado su capacidad máxima bajo el esquema de privacidad diferencial, y que relajar aún más las garantías de privacidad no proporciona beneficios sustanciales

en términos de *accuracy*, pero sí incrementa innecesariamente el riesgo de re-identificación.

Basado en este análisis, se selecciona $\epsilon = 1.5$ como el punto de operación óptimo para el sistema, aceptando una pérdida de utilidad del ~9% respecto al Baseline (89.1% vs 80.0%) a cambio de una garantía matemática robusta de privacidad.

Al aplicar privacidad diferencial al modelo, no solo cambia la precisión global, sino también el comportamiento del modelo frente a cada clase. La Figura 17 compara la distribución de las predicciones del modelo privado con un $\epsilon = 1.5$ contra la distribución real de los datos de prueba.

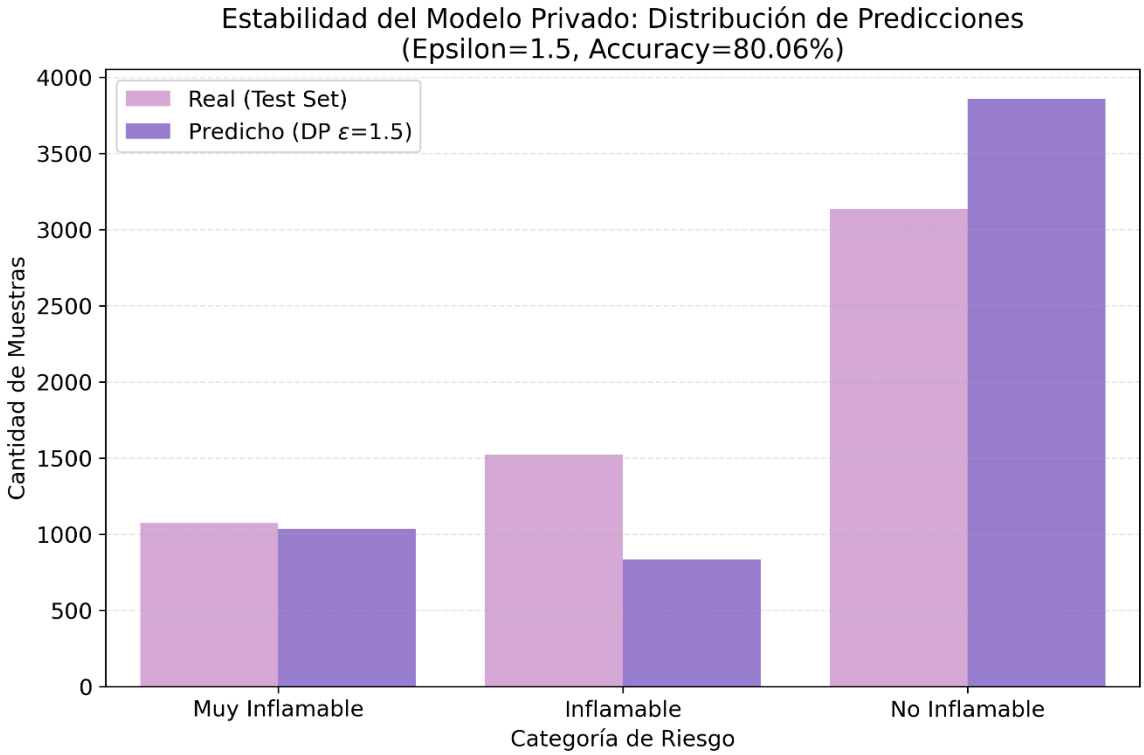

Figura 17. Distribución de clases: Real vs. Predicho con Privacidad Diferencial (ϵ=1.5). Elaboración propia, 2025.

Se puede concluir del gráfico un fenómeno de sesgo hacia la clase mayoritaria inducido por el ruido, lo que explica la caída en la sensibilidad del modelo.

- En la zona “No inflamable”, el modelo privado predice en esta clase una mayor frecuencia que su ocurrencia real. Esto sugiere que, ante la incertidumbre generada por el ruido, el algoritmo tiende a converger hacia la clase más segura y abundante.
- La zona “Inflamable”, es la más afectada, siendo frecuentemente más clasificada.

7.4 Análisis Comparativo Final

En este apartado se compara los resultados del modelo sin privacidad frente al modelo privado. Este análisis permite cuantificar el “costo de privacidad” no sólo en términos de métricas globales, sino en el impacto específico sobre la seguridad operativa.

La Figura 18 presenta la comparación de los modelos baseline vs modelo privado mediante las matrices de confusión, revelando un impacto cuantificable de implementar garantías de privacidad en el desempeño predictivo.

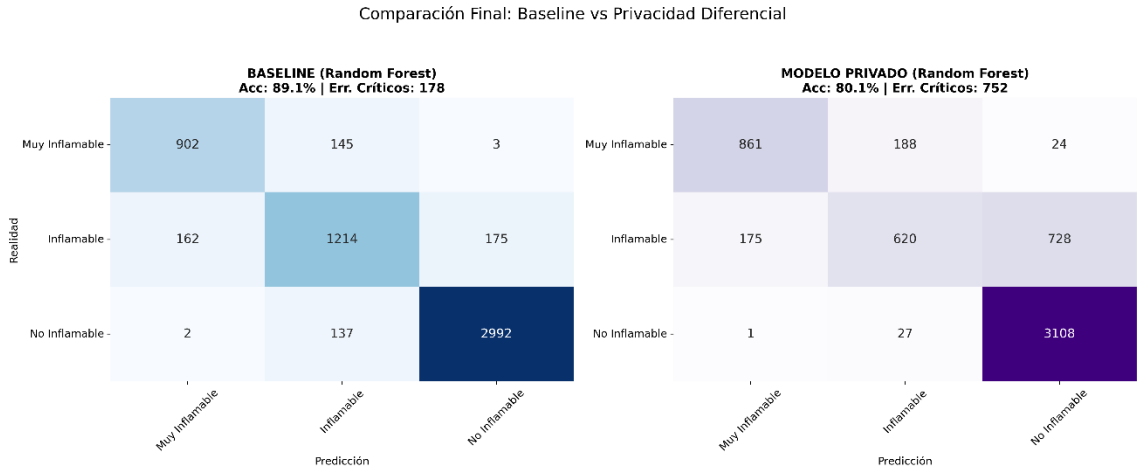

Figura 18. Comparación de Matrices de Confusión: Baseline vs Modelo Privado. Elaboración propia, 2025.

El modelo baseline alcanza una precisión del 89.1% con un error crítico de 178, mientras que el modelo privado obtiene 80.1% de precisión con 752 errores críticos. Esta diferencia de 9 puntos de precisión representa el costo de la privacidad en términos de utilidad del modelo.

El análisis comparativo revela una degradación en la capacidad del modelo para identificar sustancias de la clase "Inflamable" (23-60°C). Mientras que el modelo baseline mantiene

una tasa de acierto robusta con 1214 clasificaciones correctas (78.3% de precisión en esta clase), el modelo con privacidad diferencial sufre una caída mostrando 620 aciertos correctos (40.4% de precisión), representando una pérdida del 48.9% en la capacidad predictiva para esta categoría específica.

El modelo privado (Figura 19) tiene a clasificar sustancias peligrosas en categorías de bajo riesgo. Los 728 casos de sustancias "Inflamables" clasificadas erróneamente como "No Inflamables" representan un riesgo operacional.

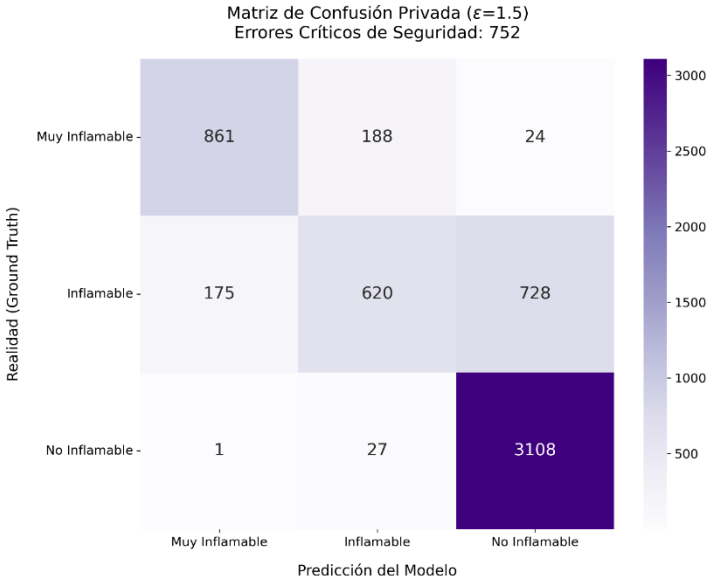

Figura 19. Detalle de Errores críticos de seguridad en el modelo privado (ϵ =1.5).

Elaboración propia, 2025.

El aumento de errores críticos en el modelo al incorporar mecanismos de seguridad plantea un hallazgo significativo.

La Figura 19 presenta en detalle los errores críticos de seguridad, entregando los siguientes análisis:

- El modelo Baseline comete 178 errores críticos. El modelo Privado eleva esta cifra a 752 errores críticos.
- El 96% de estos errores de seguridad se concentran en la confusión de sustancias "Inflamables" siendo catalogadas como "No Inflamables".

7.5 Resumen de Negocio (Dashboard)

Se consolidan las métricas técnicas (precisión, tiempo de cómputo) y las métricas de negocio (riesgo, KPIs operativos) en un tablero de evaluación integral que sintetiza los trade-offs críticos (Figura 20). Este resumen permite visualizar el impacto holístico necesaria para la selección estratégica del modelo más apropiado según el contexto de implementación.

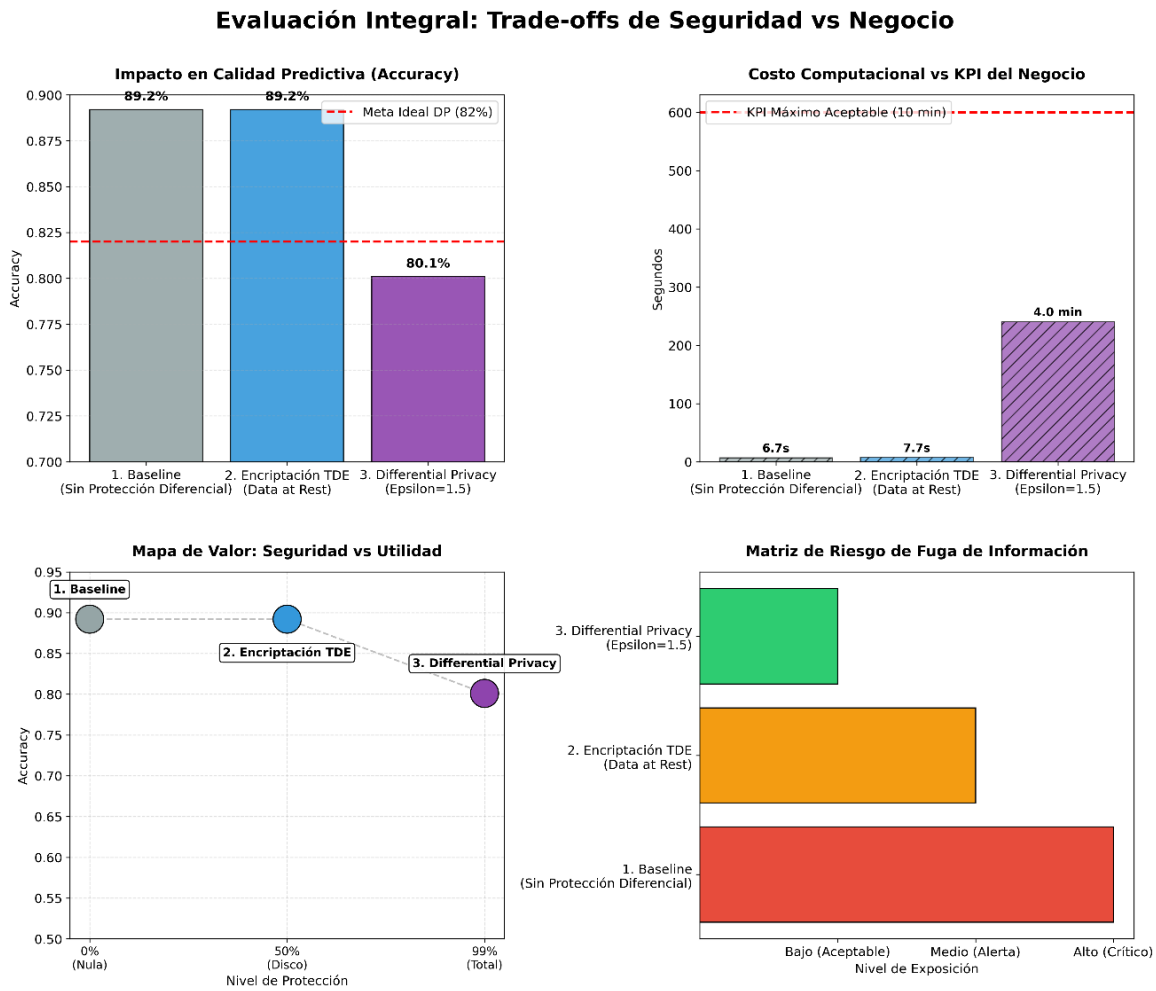

Figura 20. Tablero de evaluación integral: Impacto técnico y de negocio. Elaboración propia, 2025.

El impacto en calidad predictiva muestra la degradación progresiva del rendimiento predictivo a medida que se incrementa la seguridad de los datos. El modelo baseline sin

protección alcanza un 89.2% de *accuracy*, estableciendo el límite superior de rendimiento del sistema. La encriptación TDE mantiene intacta esta capacidad predictiva con 89.2%, demostrando que la protección de datos en reposo no compromete la utilidad del modelo. Sin embargo, el modelo con privacidad diferencial (ϵ=1.5) experimenta una reducción al 80.1%, quedando 1.9% por debajo de la meta ideal de privacidad diferencial establecida en 82%.

El gráfico superior derecho ("Costo Computacional vs KPI") demuestra que la arquitectura es viable para entornos productivos, contrasta el tiempo de inferencia de cada aproximación contra el KPI máximo aceptable de 10 minutos (600 segundos), representado por la línea roja punteada. Los resultados revelan diferencias operacionales dramáticas.

- **Baseline (sin protección)**: 6.7 segundos. Opera con máxima eficiencia, siendo 90 veces más rápido que el límite aceptable.
- **Encriptación TDE**: 7.7 segundos. El overhead de desencriptación añade apenas 1 segundo (+15%), manteniéndose ampliamente dentro de los parámetros operacionales.
- **Privacidad Diferencial**: 240 segundos (4 minutos). El proceso de adición de ruido calibrado y las verificaciones de privacidad incrementan el tiempo de inferencia del baseline, aunque permanece por debajo del umbral crítico de 600 segundos.

En relación con el costo-beneficio del proyecto se resume en la transición de riesgo mostrada en los gráficos inferiores ("Mapa de Valor" y "Matriz de Riesgo"). En cuanto al mapa de valor, la máxima utilidad con protección nula, representa el escenario de referencia donde los datos sensibles permanecen completamente expuestos, con protección intermedia sin sacrificar *accuracy,* se posiciona como un equilibrio atractivo, y el máximo nivel de protección, se genera al costo de una reducción sustancial del 9.1% en utilidad.

La matriz de riesgo, la Privacidad Diferencial, proporciona garantías matemáticas probables de que la inclusión o exclusión de cualquier registro individual en el dataset de

entrenamiento no puede ser inferida por un atacante. El modelo baseline sin protección, los datos sensibles de fórmulas químicas están completamente expuestos a múltiples vectores de ataque.

8 Despliegue del Producto de Datos

Una vez validada la viabilidad técnica del modelo con Privacidad Diferencial y seleccionado el parámetro óptimo ($\epsilon = 1.5$), el proyecto pasa desde la fase experimental hacia la etapa de operación y despliegue. El objetivo de este capítulo es describir la arquitectura de industrialización diseñada para transformar el modelo predictivo en un software robusto, escalable y seguro. Para ello, se adopta un enfoque de *MLOps*, priorizando la independencia modular entre el entorno de inferencia y los datos de entrenamiento, garantizando así que la puesta en producción no comprometa la integridad de la Propiedad Intelectual de la empresa ni la estabilidad de los sistemas corporativos existentes.

8.1 Arquitectura de Microservicios

Para habilitar el uso del modelo predictivo por parte de las distintas áreas de la empresa (I+D, Informática y Logística), se opta por una estrategia de exposición vía API REST (*Representational State Transfer*), descartando la integración monolítica directa en la base de datos o la ejecución de scripts locales.

La decisión de encapsular el modelo detrás de una API responde a cuatro principios arquitectónicos fundamentales para este caso de uso:

1. **Abstracción y Seguridad (Caja Negra):** La API actúa como una frontera funcional estricta. Los sistemas clientes envían los parámetros químicos (*inputs*) y reciben una clasificación de riesgo (*output*), sin tener acceso directo al archivo binario del modelo (*.joblib*) ni a la lógica interna de Privacidad Diferencial. Esto

mitiga riesgos de ingeniería inversa y evita la manipulación no autorizada de los hiperparámetros del modelo.

2. **Desacoplamiento Tecnológico:** Al exponer el servicio a través del protocolo estándar HTTP/JSON, se garantiza la interoperabilidad con cualquier sistema de la empresa, independientemente de su lenguaje de programación. Desde un ERP, un dashboard moderno en Python o una planilla de Excel con macros, cualquier cliente puede consumir la predicción sin necesidad de instalar librerías de Machine Learning en sus terminales.
3. **Gestión Centralizada de la Inferencia:** Centralizar la lógica de predicción en un único punto de entrada (*Endpoint*) elimina la complejidad operativa del despliegue. Cuando el equipo de Data Science necesite desplegar una versión mejorada del modelo, solo debe actualizar el microservicio; todos los usuarios comenzarán a consumir la nueva versión instantáneamente sin necesidad de actualizar sus aplicaciones locales.
4. **Eficiencia con FastAPI:** Para la implementación técnica se selecciona el framework FastAPI. Su motor asíncrono (ASGI) permite manejar múltiples solicitudes de predicción concurrentes con una latencia mínima, requisito indispensable para no generar cuellos de botella. Además, su capacidad de validación automática de tipos de datos (*pydantic*) actúa como una primera barrera de defensa ante inyecciones de datos malformados.

8.2 Containerización

Para garantizar la reproducibilidad del entorno de ejecución y el aislamiento de los procesos, se implementa una estrategia de virtualización a nivel de sistema operativo utilizando Docker. A diferencia de las máquinas virtuales tradicionales, los contenedores comparten el kernel del sistema operativo anfitrión, pero mantienen sus propias bibliotecas y dependencias aisladas, lo que resulta en una solución más ligera y eficiente.

La construcción del contenedor se define mediante un archivo Dockerfile (ver Figura 21), el cual orquesta las siguientes capas de seguridad y configuración:

1. **Imagen Base Minimizada:** Se utiliza *python:3.10-slim* como imagen base. Al utilizar una versión slim, se reduce significativamente la superficie de ataque al excluir herramientas del sistema innecesarias que podrían ser explotadas por actores maliciosos.
2. **Gestión de Dependencias:** Las librerías necesarias, tales como *scikit-learn*, *diffprivlib* y *fastapi*, se instalan a partir del archivo *requirements.txt*. Esto asegura que el modelo productivo utilice las versiones exactas con las que fue validado en la etapa experimental, eliminando errores por incompatibilidad.
3. **Inmutabilidad:** Una vez construida la imagen, el entorno se vuelve inmutable. Cualquier cambio en el código o en el modelo requiere la reconstrucción de la imagen, garantizando la trazabilidad de los cambios y evitando modificaciones "en caliente" en el servidor de producción.

```dockerfile
# Imagen base ligera de Python
FROM python:3.10-slim

# Creación de carpeta de trabajo dentro del contenedor
WORKDIR /app

# Se copia el archivo de requerimientos y se instalan los paquetes
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

# Copia el resto del código (main.py y carpeta model)
COPY . .

# Se expone el puerto 80 dentro del contenedor
EXPOSE 80

# Comando para iniciar la API
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "80"]
```

Figura 21. Configuración del Dockerfile para la creación del entorno seguro.

8.3 Endpoints

La interacción con el modelo predictivo se establece a través de una API REST desarrollada con FastAPI. Este enfoque permite desacoplar la lógica compleja del modelo (*Privacidad Diferencial, transformaciones químicas*) de los sistemas consumidores (*ERP, Dashboards, Excel, etc*).

La API expone dos puntos *endpoints* diseñados para cubrir las necesidades operativas y de monitoreo, como se muestra en la Tabla 4:

Tabla 4. Definición de Endpoints del Microservicio

Método HTTP	Ruta (Endpoint)	Función	Descripción Técnica
GET	/health	Monitoreo	Verifica el estado de salud del servicio y confirma que el modelo *.joblib* ha sido cargado correctamente en memoria.
POST	/predict	Inferencia	Recibe un objeto JSON con las características químicas de la fórmula. Aplica las transformaciones, ejecuta la predicción con el modelo privado y retorna la clasificación de riesgo GHS.

Fuente: Elaboración Propia, 2025.

Para asegurar la integridad de los datos de entrada, se implementa un esquema de validación estricta utilizando la librería *Pydantic*. Esto garantiza que, si un sistema cliente envía datos erróneos (por ejemplo, texto en lugar de un valor numérico para la temperatura), la API rechace la solicitud automáticamente antes de que esta llegue al modelo, protegiendo al sistema de errores de ejecución.

8.4 Estrategia de Entrega de Artefactos

Debido a la naturaleza sensible de la propiedad intelectual y al tamaño del modelo entrenado (aproximadamente 1.19 GB), se diseña una estrategia de entrega que separa el código fuente de los artefactos binarios. Esta práctica, conocida en ingeniería de software

como el patrón *"Code-Model Separation"*, se implementa mediante la estructura de directorios descrita en la Figura 22.

El flujo de despliegue sigue los siguientes pasos de seguridad:

1. **Entrenamiento Local Seguro:** El modelo se entrena en un entorno controlado y se serializa en un archivo *dp_model.joblib*.
2. **Exclusión de Repositorio:** Mediante reglas de *.gitignore*, se impide que el modelo binario sea subido a repositorios de código compartidos (como GitHub/GitLab), evitando la exposición accidental de la Propiedad Intelectual matemática.
3. **Inyección en Tiempo de Construcción:** Durante la fase de *docker build*, el archivo del modelo se copia físicamente desde el almacenamiento seguro local hacia el directorio */app/model/* dentro de la imagen del contenedor.

Esta estrategia asegura que la imagen final de Docker sea un artefacto autocontenido: posee todo lo necesario para funcionar, pero el modelo no puede ser extraído fácilmente sin acceso al sistema de archivos del contenedor en ejecución.

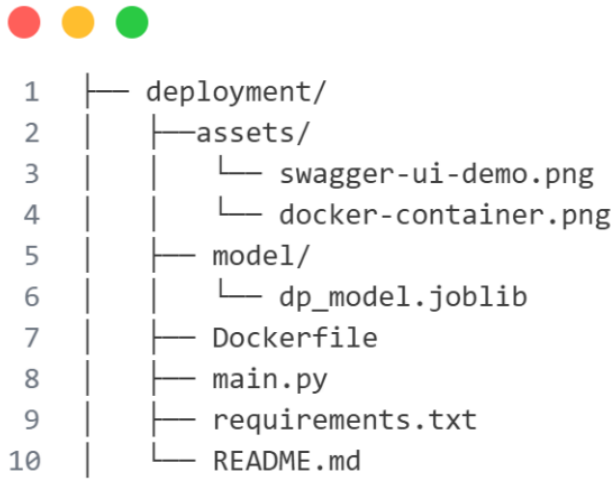

Figura 22. Estructura del paquete de despliegue evidenciando la segregación del artefacto del modelo. Elaboración Propia, 2025.

9 Conclusiones

9.1 Conclusiones Generales

El presente proyecto logra diseñar y validar exitosamente una arquitectura de Machine Learning seguro que permite a la empresa explotar el valor de sus datos sin comprometer su activo más crítico: la propiedad intelectual de sus fórmulas. La hipótesis de trabajo ha sido verificada, demostrando que es posible obtener un modelo predictivo con una utilidad operativa viable (*Accuracy 80%*) utilizando técnicas de encriptación y privacidad diferencial.

A continuación, se detalla el cumplimiento de cada uno de los objetivos específicos planteados:

1. **Respecto a la seguridad en la infraestructura de datos**, se diseña e implementa una estrategia de defensa utilizando las capacidades nativas de Oracle 19c Enterprise. Se activó exitosamente Transparent Data Encryption (TDE) con el algoritmo AES-256 para la protección de datos en reposo, gestionando las llaves maestras mediante una Oracle Wallet aislada. Adicionalmente, se configura Native Network Encryption (NNE) para asegurar el tránsito de datos, mitigando de manera efectiva el riesgo de exfiltración física o interceptación de la base de datos maestra.
2. **En cuanto a la privacidad en el diseño de datos**, se desarrolla un esquema de ingeniería de características que transforma la composición química exacta en atributos estadísticos agregados. Mediante la implementación de la vista segura, se logra abstraer la complejidad de las fórmulas, eliminando la necesidad de exponer identificadores de ingredientes al equipo de ciencia de datos y/o informática, reduciendo significativamente el nivel de ataque ante actores internos.
3. **Sobre el modelado con privacidad diferencial**, se construyen y evalúan modelos de Random Forest integrando mecanismos de perturbación estocástica mediante

la librería *diffprivlib*. El análisis de trade-off permitió determinar que un parámetro de privacidad (Épsilon) de $\varepsilon = 1.5$ ofrece el equilibrio óptimo, reteniendo el 97% de la capacidad predictiva del modelo baseline (80.1% vs 89.2%) y proporcionando al mismo tiempo una garantía matemática robusta contra ataques de inferencia y reconstrucción.

4. **Referente a la operacionalización y despliegue**, se construye una arquitectura de microservicios segura utilizando Docker y FastAPI. El modelo entrenado fue encapsulado en un contenedor inmutable y aislado, desacoplando el entorno de inferencia del sistema de archivos principal. Esta estrategia asegura que el modelo pueda ser desplegado en cualquier infraestructura sin exponer el código fuente ni los datos de entrenamiento, cumpliendo con los estándares modernos de *MLOps*.
5. **Finalmente, en relación a la gobernanza**, se establece un marco de control mediante la implementación de un Dashboard de Auditoría basado en *Streamlit*. Esta herramienta permite visualizar y trazar todos los accesos a la información sensible, garantizando el cumplimiento de las políticas internas de seguridad de la empresa y cerrando el ciclo de protección de datos con una capa de supervisión activa.

9.2 Trabajos Futuros

A partir de los hallazgos y limitaciones de este estudio, se proponen las siguientes líneas de investigación para continuar fortaleciendo la estrategia de Inteligencia Artificial segura en la empresa:

- **Exploración de Encriptación Homomórfica :** Se sugiere investigar el uso de librerías como TenSEAL o PyFHE para permitir la inferencia directa sobre datos encriptados, eliminando la necesidad de descifrado temporal en la memoria RAM durante la predicción y mitigando riesgos de *memory scraping*.
- **Monitoreo de Desviación de Datos:** Integrar en la API de producción un módulo de detección de *drift* para alertar cuando las nuevas fórmulas a predecir se alejen estadísticamente de la distribución con la que fue entrenado el modelo privado, asegurando la fiabilidad de las predicciones a largo plazo.

10 Bibliografía

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). *Deep learning with differential privacy*. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 308-318. https://doi.org/10.1145/2976749.2978318

Acar, A., Aksu, H., Uluagac, A. S., & Conti, M. (2018). *A survey on homomorphic encryption schemes: Theory and implementation*. ACM Computing Surveys (CSUR), 51(4), 1-35. https://doi.org/10.1145/3214303

Dwork, C., & Roth, A. (2014). *The Algorithmic Foundations of Differential Privacy*. Foundations and Trends® in Theoretical Computer Science, 9(3-4), 211-407. https://doi.org/10.1561/0400000042

Gharagheizi, F., & Fareghi Alamdari, R. (2008). *Prediction of flash point temperature of pure components using a quantitative structure–property relationship model*. QSAR & Combinatorial Science, 29(2), 679-683. https://doi.org/10.1002/qsar.200730110

Guerra-Manzanares, A., Lechuga Lopez, L. J., Maniatakos, M., & Shamout, F. E. (2023). *Privacy-preserving machine learning for healthcare: Open challenges and future perspectives*. ICLR 2023 Workshop on Trustworthy Machine Learning for Healthcare. https://arxiv.org/abs/2303.15563

Jeong, K., Nam, J. H., Lee, S., Koo, J., Lee, J., Yu, D., . . . Kim, J. (2024). *Prediction of flash point of materials using Bayesian kernel machine regression based on Gaussian processes with LASSO-like spike-and-slab hyperprior*. Journal of Chemometrics, 38(10), 1-13. https://doi.org/10.1002/cem.3586

Katritzky, A. R., Stoyanova-Slavova, I. B., Dobchev, D. A., & Karelson, M. (2007). *QSPR modeling of flash points: An update*. Journal of Molecular Graphics and Modelling, 26(2), 529-536. https://doi.org/10.1016/j.jmgm.2007.03.006

Lee, C. J., Ko, J. W., & Lee, G. (2012). *Flash point prediction of organic compounds using a group contribution and support vector machine*. Korean Journal of Chemical Engineering, 29(2), 145-153. https://doi.org/10.1007/s11814-011-0164-8

Liu, P., Xu, X., & Wang, W. (2022). *Threats, attacks and defenses to federated learning: Issues, taxonomy and perspectives*. Cybersecurity, 5(1), 1-19. https://doi.org/10.1186/s42400-021-00105-6

McMahan, B., Moore, E., Ramage, D., Hampson, S., & Aguera y Arcas, B. (2017). *Communication-efficient learning of deep networks from decentralized data*. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 1273-1282. https://proceedings.mlr.press/v54/mcmahan17a.html

Microsoft. (2023). *Amenazas: Microsoft Threat Modeling Tool (Azure)*. Microsoft Learn. https://learn.microsoft.com/es-es/azure/security/develop/threat-modeling-tool-threats

Nguyen, D. C., Pham, Q.-V., Pathirana, P. N., Ding, M., Seneviratne, A., Lin, Z., . . . Hwang, W.-J. (2022). *Federated learning for smart healthcare: A survey*. ACM Computing Surveys (CSUR), 55(3), 1-37. https://doi.org/10.1145/3501296

Oracle. (s.f.). *Introduction to Transparent Data Encryption. Oracle Database Documentation*. https://docs.oracle.com/en/database/oracle/oracle-database/26/dbtde/introduction-to-transparent-data-encryption.html#GUID-688B2CA5-EB00-4BEE-9486-9046670CCA70

Saldana, D. A., S. L., Mougin, P., Rousseau, B., & Creton, B. (2013). *Prediction of flash points for fuel mixtures using machine learning and a novel equation*. Energy & Fuels, 27(11), 7300-7314. https://doi.org/10.1021/ef4005362

Singh, S., & Shukla, K. K. (2021). *Privacy-preserving machine learning for medical image classification*. Cornell University. https://arxiv.org/abs/2108.12816

Tida Sai Venkatesh Chilukoti, V., Hsu, S., & Hei, X. (2022). *Privacy-preserving deep learning model for COVID-19 disease detection*. https://arxiv.org/abs/2209.04445

Weinreich, J., von Rudorff, G. F., & von Lilienfeld, O. A. (s.f.). *Encrypted machine learning of molecular quantum properties*. https://arxiv.org/abs/2212.04322

Yue, Z., Ding, S., Zhao, L., Zhang, Y., Cao, Z., Tanveer, M., . . . Zheng, X. (2021). *Privacy-preserving time-series medical images analysis using a hybrid deep learning framework*. ACM Transactions on Internet Technology (TOIT), 21(3), 1-21. https://doi.org/10.1145/3383779