Modelos de aprendizaje automático ensamblados para la clasificación de datos desbalanceados

María Isabella Meneses Ospina

Trabajo de grado para optar al título de Ingeniera Industrial

Directora:

Yuly Andrea Ramírez Sierra

Magíster en Ingeniería Industrial

Codirector:

Henry Lamos Díaz

Ph.D. en Física - Matemática

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Estudios Industriales y Empresariales

Ingeniería Industrial

Bucaramanga

2025

Dedicatoria

A Dios y a la vida, por el honor de llamar a la UIS mi Alma Máter.

A Chavelita por tener la determinada determinación

de seguir el camino de sus propios sueños.

A mi madre, quien me impulsó a seguir mi corazón.

A todos los que confiaron y me abrazaron tiernamente:

este logro también es de ustedes.

Agradecimientos

A Mitita y Papalfredo, sé que desde el cielo están muy orgullosos de lo que soy ahora.

A mi familia, que siempre estuvo para mí, apoyándome en todo momento.

A Danna y Sofía, por hacer que esta etapa universitaria se sintiera tan especial, llena de apoyo mutuo y de una conexión que solo nosotras entendemos.

A O.V.E.J.A.S, que con amor fraterno me ayudaron a descubrir quién soy realmente.

A ustedes, gracias por llenarme de Dios y hacer de Bucaramanga mi hogar.

A la selección de Tenis de Mesa UIS, por hacerme entender que detrás de una gran victoria hay un gran equipo, que con esfuerzo, dedicación y unión lograron grandes logros.

A ti, Juli, estaré infinitamente agradecida por tenderme la mano cuando más lo necesitaba y hacerme comprender que no estoy sola.

A mi directora Yuly Ramírez, al profesor Henry Lamos y al grupo OPALO, por guiarme en mi proceso académico, investigativo y profesional, no existe mejores referentes que ustedes.

A la Universidad Industrial de Santander, todo mi esfuerzo será en pro de un mundo mejor, porque la educación, la cultura y el deporte cambian vidas.

Gracias.

Tabla de contenido

Lista de tablas

Lista de figuras

Lista de Apéndices

Resumen

Título: Modelos de aprendizaje automático ensamblados para la clasificación de datos desbalanceados[1]

Autor: María Isabella Meneses Ospina**

Palabras clave: Aprendizaje automático; Problema de desbalance de clases; Datos desequilibrados; Técnicas de re-muestreo; Clasificadores ensamblados.

Descripción:

La tarea de clasificación en el aprendizaje automático implica predecir una etiqueta de clase para cada instancia, basándose en los patrones descubiertos durante la fase de entrenamiento del modelo, para automatizar su asignación en nuevas observaciones. No obstante, surge el *problema de desbalance de clases*, originado por tendencias de distribución sesgada. Este fenómeno se presenta cuando una clase está representada por un amplio número de elementos, en comparación con los elementos de las demás clases, lo que llevaría a que probablemente los modelos de aprendizaje automático tengan un rendimiento deficiente durante su fase de validación, evidenciado en baja precisión e incapacidad de generalización. Este trabajo aborda esta problemática mediante el uso de modelos ensamblados, específicamente Random Forest y XGBoost, combinados con técnicas de re-muestreo como RUS, ROS y SMOTE. La técnica RUS demostró ser efectiva para mejorar la detección de la clase minoritaria. Sin embargo, las técnicas de sobremuestreo ROS y SMOTE en todas sus configuraciones evaluadas presentaron sobreajuste. De esta forma, a partir de distintas estrategias de re-muestreo y configuración de hiperparámetros, se logró identificar los modelos recomendables para cada conjunto de datos priorizando la métrica Recall. En el caso de la base de datos con desequilibrio extremo, XGBoost con RUS fue el modelo más recomendable, mientras que, para el conjunto de datos con desbalance moderado, Random Forest con RUS logró un mejor equilibrio entre métricas clave y generalización de los modelos. De esta forma, se destaca la importancia de integrar técnicas a nivel de datos, optimización de hiperparámetros y análisis de métricas clave para abordar este problema de desequilibrio de clases.

---

[1] Tesis de pregrado

** Facultad de Ingenierías Físico – Mecánicas. Escuela de Estudios Industriales y Empresariales, Ingeniería Industrial. Directora: Yuly Andrea Ramírez Sierra, M.Sc. Ingeniería Industrial.

Abstract

Title: Ensemble machine learning models for imbalanced data classification[2]

Author: María Isabella Meneses Ospina**

Keywords: Machine learning; Class imbalanced problem; Imbalanced data; Resampling techniques; Ensemble models.

Description:

The classification task in machine learning involves predicting a class label for each instance based on patterns discovered during the model's training phase, with the goal of automating label assignment of new observations. However, the class imbalance problem arises as a result of skewed distribution trends. This phenomenon occurs when one class is represented by a significantly larger number of instances compared to other classes, which would likely lead to machine learning models performing poorly during their validation phase, as evidenced by low accuracy and a lack of generalization ability. This study addresses this issue by utilizing ensemble models, specifically Random Forest and XGBoost, in combination with resampling techniques such as RUS, ROS, and SMOTE. The RUS technique proved to be effective in improving minority class detection. However, the ROS and SMOTE oversampling techniques in all evaluated configurations showed overfitting. Then, based on different re-sampling strategies and hyperparameter settings, the recommended models for each dataset were identified by prioritizing the Recall metric. For the extreme imbalance dataset, XGBoost with RUS was the most suitable model, while for the moderate imbalance dataset, Random Forest with RUS achieved a better balance between key metrics and model generalization. Thus, the importance of integrating data-level techniques, hyperparameter optimization and key metric analysis to address this class imbalance problem is highlighted.

---

[2] Undergraduate Thesis

** Faculty of Physical-Mechanics Engineering. School of Industrial and Business Studies, Industrial Engineering. Advisor: Yuly Andrea Ramírez Sierra, M.Sc. Industrial Engineering.

Introducción

La clasificación es una tarea central en el aprendizaje automático, con aplicaciones en campos como los diagnósticos médicos, predicciones de crisis financieras y detección de emergencias (Jo & Japkowicz, 2004; W. C. Lin et al., 2017; Tarekegn et al., 2021). Sin embargo, esta tarea se ve afectada por un fenómeno que se manifiesta en dichos entornos, en el cual, por las características inherentes del conjunto de datos desequilibrado, los datos tienden a agruparse en clases mayoritarias o minoritarias. Este fenómeno se conoce como *problema de desbalance de clases*, en donde "una clase puede estar representada por un amplio número de elementos, mientras que la otra parte se encuentra representada por unos pocos elementos" (W. C. Lin et al., 2017). Además, este problema a menudo implica un solapamiento significativo entre clases (Denil & Trappenberg, 2010), lo que puede dar lugar a la fragmentación de la clase minoritaria (Jo & Japkowicz, 2004).

Las estrategias para abordar este problema incluyen soluciones a nivel de datos, algoritmos, sensibilidad al costo y clasificadores ensamblados (W. C. Lin et al., 2017, p. 18). De estas, los modelos de aprendizaje automático ensamblados exhiben un rendimiento superior en comparación con algoritmos individuales (Kuncheva, 2014a) e incluso varios estudios respaldan la eficacia de este enfoque, destacando su habilidad para integrar múltiples clasificadores (Abellán & Castellano, 2017; He et al., 2018). No obstante, la aplicación directa de los clasificadores ensamblados sobre un conjunto de datos desequilibrado no logra resolver el problema por sí solo y tras varias investigaciones se ha demostrado con resultados positivos que combinar estos clasificadores con otras técnicas para abordar el problema de desequilibrio de clases conlleva a mejoras significativas en el rendimiento del modelo (Galar et al., 2012).

La evolución de los modelos de aprendizaje ensamblado ha permitido su combinación con diversos enfoques, entre los que se destacan las soluciones a nivel de datos en el cual se realiza un preprocesamiento de la información antes de entrenar cada clasificador (Bb et al., s/f). Esta integración permite aprovechar las ventajas intrínsecas de ambos métodos, ya que, tanto los modelos ensamblados como las soluciones a nivel de datos presentan la ventaja de ser utilizados de manera independiente del clasificador base, lo que los hace altamente versátiles. Esta flexibilidad contrasta con los enfoques a nivel de algoritmo y de sensibilidad al costo, que a menudo dependen más del contexto y del problema específico (Galar et al., 2012).

Este proyecto analiza la clasificación de datos desbalanceados mediante modelos de aprendizaje automático, explorando estrategias a nivel de datos como el submuestreo aleatorio (Random UnderSampling, RUS) y el sobremuestreo aleatorio (Random OverSampling, ROS). De esta forma, los modelos Random Forest y XGBoost, junto con las estrategias a nivel de datos fueron seleccionados por su amplio uso en la literatura y su capacidad para mejorar el rendimiento en tareas de clasificación. De esta forma, utilizando conjuntos de datos de fuentes secundarias como Kaggle y UCI Machine Learning Repository, se realizaron comparaciones entre los modelos ajustados y las estrategias seleccionadas mediante métricas como el Recall o sensibilidad.

Este documento incluye un marco teórico (sección 5) que establece las bases conceptuales, seguido de la metodología organizada en cinco fases: revisión de literatura (sección 6), preparación de datos (sección 7), modelamiento (sección 8), evaluación e interpretación (sección 9), y documentación de los resultados (sección 10). Estas etapas incluyen técnicas como re-muestreo, optimización de hiperparámetros y análisis de métricas, culminando en un repositorio que documenta el proceso en GitHub, un artículo académico y un informe en Power Bi que representa visualmente los resultados obtenidos en la investigación.

Definición del problema

El aprendizaje automático ha emergido como un avance significativo dentro del ámbito de la inteligencia artificial, habilitando la capacidad de las máquinas para aprender de la experiencia y desempeñar tareas predictivas esenciales, como la clasificación y la regresión (Murphy, 2012). Sin embargo, uno de los retos más críticos en este campo es el problema del desbalance de clases, el cual se presenta cuando un conjunto de datos tiene una distribución sesgada entre clases mayoritarias y minoritarias. Esta situación impacta negativamente el rendimiento de los modelos de clasificación, particularmente en ámbitos sensibles como los diagnósticos médicos, la detección de fraudes financieros y la identificación de defectos en procesos de manufactura (W. C. Lin et al., 2017).

El desbalance de clases plantea un desafío debido a que la mayoría de los algoritmos de aprendizaje automático están diseñados bajo el supuesto de distribuciones equilibradas entre las clases. Esto provoca que los modelos prioricen el aprendizaje de la clase mayoritaria, ignorando la importancia de los casos minoritarios, los cuales suelen ser críticos en muchas aplicaciones reales. Además, este problema se agrava con fenómenos como el solapamiento significativo entre clases y la fragmentación de la clase minoritaria, complicando aún más el aprendizaje efectivo. (Denil & Trappenberg, 2010; Jo & Japkowicz, 2004).

En respuesta a estos desafíos, la literatura ha identificado diversas estrategias para mitigar el impacto del desbalance de clases. Estas estrategias incluyen enfoques a nivel de datos, algoritmos, sensibilidad al costo y clasificadores ensamblados (W. C. Lin et al., 2017). Particularmente, los clasificadores ensamblados han demostrado ser una solución efectiva al combinar las fortalezas de múltiples modelos, lo que mejora su capacidad para abordar problemas

complejos (Kuncheva, 2014b). Sin embargo, la aplicación directa de estos clasificadores sobre conjuntos de datos desequilibrados no garantiza resultados óptimos. Estudios han demostrado que la combinación de clasificadores ensamblados con técnicas a nivel de datos incrementa el rendimiento de los modelos (Galar et al., 2012)

A pesar de los avances, persisten interrogantes sobre cómo optimizar la integración de estas estrategias para maximizar la clasificación en conjuntos de datos desbalanceados. La selección adecuada de hiperparámetros, así como la elección de las métricas de rendimiento adecuadas para evaluar la efectividad de las soluciones propuestas. Además, existe una necesidad de evaluar estas estrategias en contextos diversos para garantizar su generalizabilidad y aplicabilidad en escenarios reales.

Por tanto, este proyecto busca abordar el problema del desbalance de clases mediante un análisis detallado de la clasificación en conjuntos de datos desbalanceados, utilizando modelos ensamblados como Random Forest y XGBoost, combinados con técnicas a nivel de datos como RUS y ROS. Para ello, se emplearán conjuntos de datos secundarios provenientes de fuentes reconocidas como Kaggle y UCI Machine Learning Repository. La evaluación incluirá comparaciones entre combinaciones de estrategias y modelos, utilizando métricas como el Recall para identificar las mejores configuraciones. Este estudio contribuirá a la literatura existente al proponer un enfoque integrado y sistemático para abordar el desbalance de clases, proporcionando herramientas prácticas y resultados replicables en escenarios reales.

Generalidades del proyecto

2.1. Objetivo General

Analizar la clasificación de datos desbalanceados a partir de modelos de aprendizaje automático ensamblados.

2.2. Objetivos Específicos

Realizar una revisión de literatura para identificar los enfoques que dan solución a los conjuntos de datos con clases desbalanceadas a partir de modelos de aprendizaje automático ensamblados.

Contrastar los modelos de aprendizaje automático ajustados utilizando diferentes conjuntos de datos para validar las estrategias de solución del problema de clases desbalanceadas.

Construir una herramienta de visualización para mostrar los principales resultados de la investigación.

Elaborar un artículo de carácter publicable a partir de los resultados de la investigación.

2.3. Resultados Esperados

Revisión de literatura con los enfoques que dan solución a los conjuntos de datos con clases desbalanceadas a partir de modelos de aprendizaje automático ensamblados.

Síntesis de los resultados de la investigación en la herramienta de visualización Power BI.

Artículo científico con los hallazgos obtenidos en la investigación realizada.

Libro de trabajo de grado con las evidencias de la investigación.

Cumplimiento de objetivos

A continuación, en la tabla 1 se detalla el cumplimiento de los objetivos establecidos en este trabajo de grado. Cada objetivo se enumera junto a las páginas correspondientes que evidencian su cumplimiento.

Tabla 1

Cumplimiento de objetivos

Objetivo	***Página / Apéndice***
Realizar una revisión de literatura para identificar los enfoques que dan solución a los conjuntos de datos con clases desbalanceadas a partir de modelos de aprendizaje automático ensamblados.	**Página 36, Apéndice A**
Contrastar los modelos de aprendizaje automático ajustados utilizando diferentes conjuntos de datos para validar las estrategias de solución del problema de clases desbalanceadas.	**Página 99**
Construir una herramienta de visualización para mostrar los principales resultados de la investigación.	**Página 106, Apéndice E**
Elaborar un artículo de carácter publicable a partir de los resultados de la investigación.	**Apéndice F**

Marco teórico

En esta sección, se presentan los conceptos fundamentales que constituyen el marco teórico del presente trabajo de investigación. El análisis predictivo y su aplicación en la clasificación de conjuntos de datos desbalanceados constituyen el eje central de este estudio. Se examinan los métodos de clasificación existentes, con énfasis en la evaluación y selección de modelos adecuados para este tipo de problemas. Asimismo, se profundiza en los principales métodos ensamblados identificados en la literatura, destacando su relevancia en el contexto de análisis de datos desbalanceados.

4.1. Análisis Predictivo

Es un tipo de analítica que implica la estimación de la probabilidad de ocurrencia de eventos futuros con el propósito de comprender y categorizar la información utilizando datos históricos; y de esta manera, facilitar la toma de decisiones. Así, dependiendo de la mentalidad con la que se pretenda abordar el problema, el proceso de análisis entorno a la predicción puede variar, puesto que se podría observar desde una perspectiva de minimización del error de predicción global o para el entendimiento global (Winters, 2017).

4.1.1. *Aprendizaje Automático*

Este campo se enfoca en investigar y desarrollar técnicas y/o modelos que permitan a las máquinas aprender de manera autónoma y mejorar su rendimiento, lo que, a su vez, contribuye a la identificación de patrones complejos y facilita la toma de decisiones inteligentes basadas en datos (Bobadilla, 2020). Los problemas fundamentales en esta disciplina se engloban en el aprendizaje supervisado, no supervisado, semi-supervisado y por refuerzo. La diferencia entre el aprendizaje automático supervisado y el no supervisado radica en que el primero implica la

clasificación de muestras etiquetadas previamente para entrenar al modelo, mientras que el segundo utiliza muestras no etiquetadas que se agrupan en grupos o clases mediante técnicas de agrupamiento. El aprendizaje semi-supervisado se ocupa de conjuntos de datos en los que solo una parte está etiquetada y el resto no, lo que combina el enfoque supervisado y no supervisado. El aprendizaje por refuerzo, inspirado en mecanismos naturales, implica aprender estrategias o "políticas" mediante la interacción con un entorno real o simulado, siguiendo los principios de la evolución natural (Bobadilla, 2020). Adicionalmente, (Han et al., 2012) describen otro enfoque denominado aprendizaje activo, en el cual se optimiza la calidad del modelo adquiriendo activamente conocimientos de los usuarios humanos al etiquetar muestras ocasionalmente.

4.1.1.1. **Aprendizaje supervisado.**

Dado un conjunto etiquetado de pares de entrada-salida, conocido como *conjunto de entrenamiento*, el objetivo es aprender a asignar las salidas correspondientes (*y*) a las entradas (*x*) proporcionadas. De esta manera, cada entrada de entrenamiento, denominada "características", "atributos" o "covariables, pueden ser representadas en su configuración más sencilla como una tupla que se representa como un vector de números de D-dimensiones, un objeto estructurado complejo. De manera similar, se asume que la variable de salida o respuesta y_i puede ser una variable categórica o nominal en la que no existe un orden preestablecido entre las categorías, o bien un valor escalar real. Esto convierte al primer escenario en un problema de clasificación o de reconocimiento de patrones, y al segundo en un problema de regresión (Murphy, 2012). En esta investigación, el enfoque se centra exclusivamente en el problema de clasificación, dado el enfoque inherente de interés de este proyecto.

4.2. Métodos de clasificación

Es un proceso en el cual se busca predecir la categoría o clase de una variable cualitativa. Frecuentemente, al utilizar métodos de clasificación, se comienza por predecir la probabilidad de que cada observación pertenezca a una de las categorías de la variable cualitativa. En este sentido, estos métodos también se comportan de manera similar a los modelos de regresión, sin embargo, en general no existe una forma natural de convertir una variable de respuesta cualitativa con más de dos niveles en una respuesta cuantitativa lista para una regresión lineal (James et al., 2013). En definitiva, el objetivo de la clasificación es construir un modelo que capture las asociaciones intrínsecas entre el tipo de clase y los atributos de nuevas instancias para poder predecir con precisión el tipo de clase a partir del valor de los atributos (Maimon & Rokach, 2005).

Los problemas de clasificación son frecuentes en diversas situaciones. Por ejemplo, cuando una persona llega a urgencias con ciertos síntomas, es esencial asignar al paciente a un nivel de triage en función de su estado y los recursos disponibles. En otros contextos, como el de los bancos en línea, es indispensable que puedan determinar si una transacción realizada en el sitio es fraudulenta o no, considerando factores como la dirección IP del usuario, historial de transacciones, entre otros (James et al., 2013).

La clasificación consta de dos etapas. En la primera, llamada *fase de aprendizaje*, se construye el clasificador a partir del conjunto de entrenamiento. Este proceso implica analizar y comprender los registros de entrenamiento, asignándolos a clases predefinidas a través de reglas de clasificación, árboles de decisión o fórmulas matemáticas. De esta manera, algunos modelos desarrollados para clasificar datos de alta dimensión son el clasificador de Naive Bayes, redes neuronales, árboles de decisión o incluso máquinas de soporte vectorial (SVMs).

La segunda etapa, llamada *fase de clasificación*, busca probar y verificar el clasificador al estimar su precisión predictiva mediante un conjunto de prueba compuesto por registros individuales perteneciente a una clase predefinida y representada por un vector de atributos de n dimensiones, utilizadas en la construcción del clasificador (Han et al., 2012).

4.2.1. *Conjunto de datos desbalanceados*

En la práctica, surge una dificultad al momento de aplicar técnicas de aprendizaje automático cuando las categorías utilizadas para clasificar un conjunto de datos se encuentran con una distribución sesgada. Este desequilibrio entre las clases puede generar como resultado discrepancias entre la distribución de los datos de prueba y los datos de entrenamiento, así que, al momento de la validación es posible que el modelo llegue a predicciones deficientes, a raíz de un entrenamiento que no refleja la realidad del problema abordado (Maimon & Rokach, 2005).

De esta manera, el *problema de desbalance de clases* se convierte en un problema evidente cuando la clase de principal interés está representada por muy pocas instancias dentro del conjunto de datos. Este desequilibrio se relaciona con el concepto de aprendizaje sensible a costos, dado que los costos asociados a los errores de clasificación no son uniformes para todas las clases. En este sentido, en el campo del diagnóstico médico, evidentemente resulta mucho más costoso diagnosticar erróneamente a un paciente con cáncer como sano que diagnosticar a un paciente sano como si tuviera cáncer (Han et al., 2012).

En diversas aplicaciones adicionales, como la detección del fraude, el monitoreo de fallas o casos específicos como la detección de derrames de petróleo a partir de radar satelital, esta distribución en el conjunto de datos puede ocasionar que los modelos de aprendizaje automático o algoritmos tradicionales fallen. Esto se debe a que estos modelos asumen que los datos provienen de una distribución balanceada, con costos de error iguales para todas las clases, lo que afecta el

objetivo fundamental de la clasificación, que es minimizar el número de errores durante la validación del modelo (Han et al., 2012).

Para direccionar la solución del problema de desbalance de clases, al largo de los años se ha explorado con una variedad de estrategias, algunas tales como el sobremuestreo, submuestreo, ajuste del umbral y técnicas de conjunto o ensamblado (Han et al., 2012). Por lo tanto, enfrentar este desafío de manera especializada se convierte en una necesidad crucial en el aprendizaje automático para mejorar el rendimiento del modelo en aplicaciones donde esta distribución desigual de las clases está presente.

4.3. Evaluación y selección de modelos

Cuando se cuenta con varios modelos de diferente complejidad para predecir el comportamiento de un sistema, surge la pregunta de cuál se debería escoger y cuál de ellos ofrece un mejor rendimiento. Un enfoque natural para resolver este dilema es a través de la tasa de clasificación incorrecta, que se calcula para cada método utilizando el conjunto de entrenamiento correspondiente definido como se expone en la ecuación 1 (Murphy, 2012).

$$err(f, D) = \frac{1}{N} \sum_{i=1}^{N} I(f(x_i) \neq y_i) \qquad \textbf{(1)}$$

Donde N es el número total de instancias en el conjunto de datos; $f(x_i)$ es la predicción del clasificador para x_i; y_i es la etiqueta real de la instancia x_i; I es la función binaria que devuelve 1 si la condición dentro del paréntesis es verdadera, y en caso contrario devuelve un valor de 0.

Sin embargo, el objetivo principal es estimar el error de generalización, que representa el promedio de la tasa de clasificación incorrecta $err(f, D)$ en datos futuros. Esta estimación no puede realizarse en dicho momento, ya que depende de nutrir al modelo en el tiempo. Por tanto, la estrategia consiste en seleccionar un modelo con la complejidad apropiada a través de la división

del conjunto de datos en datos de entrenamiento y datos de validación. Esta partición se suele realizar asignado el 80% de los datos para entrenamiento y 20% para validación, aunque esta distribución se ve afectada en ocasiones cuando no se dispone de datos suficientes para realizar una estimación confiable del rendimiento futuro (Murphy, 2012).

4.3.1. *Métricas de evaluación*

Estas métricas se emplean para evaluar la precisión con la que el clasificador puede predecir las etiquetas de clase de cada instancia. Por lo tanto, es esencial desglosar conceptos antes de comenzar a aplicar estas técnicas. En primer lugar, es preciso reconocer que, al analizar un conjunto de datos, el primer paso es definir estratégicamente el objetivo de la tarea de clasificación. Esto permite identificar las muestras pertenecientes a la clase principal de interés, que a menudo es la clase minoritaria o rara, y distinguirlas de las muestras de otras clases (Han et al., 2012).

La evaluación de un clasificador se lleva a cabo a través de la matriz de confusión, en la que las columnas representan las clases pronosticadas y las filas corresponden a las clases reales. Esta matriz está compuesta por cuatro componentes principales (Maimon & Rokach, 2005):

- ***Verdaderos Negativos (*True Negatives, TN):** Es el número de muestras negativas correctamente clasificadas.
- ***Falsos Positivos (*False Positives, FP):** Es el número de muestras negativas clasificadas incorrectamente como positivas.
- ***Falsos Negativos (*False Negatives, FN):** Es el número de muestras positivas incorrectamente clasificadas como negativas
- ***Verdaderos positivos (*True Positives, TP):** Es el número de muestras positivas clasificadas correctamente como positivas.

En la Tabla 2 se presenta la matriz de confusión, donde TP (Verdaderos Positivos) y TN (Verdaderos Negativos) indican cuando el clasificador está haciendo predicciones correctas, mientras que FP (Falsos Positivos) y FN (Falsos Negativos) señalan cuando el clasificador está cometiendo errores en sus predicciones, así que lo ideal sería que tanto los FP como los FN sean nulos (Han et al., 2012).

Tabla 2

Matriz de confusión

	Predicción Positiva	**Predicción Negativa**	*Total*
Actual Positiva	TP	FN	P
Actual Negativa	FP	TN	N
Total	P'	N'	P+N

Fuente: Adaptado de (Han et al., 2011)

La matriz de confusión es aplicable tanto para problemas de clasificación binaria como a aquellos con múltiples clases. A través de ella, se logra una evaluación del rendimiento del clasificador al calcular las métricas pertinentes de interés con los datos de prueba (Han et al., 2012). A continuación, en la tabla 3 se detallan algunas métricas de evaluación.

Tabla 3

Métricas de evaluación

Métrica	**Fórmula**
Precisión (Accuracy)	$\frac{TP + TN}{P + N}$
Tasa de error (Error rate)	$\frac{FP + FN}{P + N}$
Sensibilidad (Recall)	$\frac{TP}{TP + FN}$ o $\frac{TP}{P}$

Fuente: Adaptado de (Han et al., 2011)

La *precisión* se utiliza para determinar el porcentaje de tuplas clasificadas correctamente, y esta medida también se conoce como tasa de reconocimiento. La *tasa de error*, por otro lado,

representa el porcentaje de tuplas clasificadas incorrectamente por el clasificador. Sin embargo, cuando se enfrenta un problema de desbalance de clases, estas medidas pueden llevar a creer que el clasificador es preciso, cuando en realidad no lo es. Esto se debe a que el clasificador puede tener un alto rendimiento en la clase mayoritaria pero un rendimiento deficiente en la clase minoritaria (Han et al., 2012).

Existen muchas métricas complementarias, como la exactitud, especificidad, exhaustividad o el F-score, entre otras, que ayudan a validar si el modelo realiza predicciones precisas en el ámbito de interés. No obstante, para evaluar la eficacia de un clasificador en situaciones de desbalance de clases, se proponen otras métricas, como el *Recall* o sensibilidad, que mide la proporción de etiquetas correctas predichas con respecto al número total de etiquetas verdaderas, promediado sobre todas las instancias (Tarekegn et al., 2021).

4.3.2. *Validación cruzada*

Al evaluar los clasificadores, también es importante tener en cuenta otros aspectos, como su velocidad, robustez, escalabilidad y capacidad de interpretación. Por tanto, en el esfuerzo por mejorar la precisión predictiva de los clasificadores, se recurre a varias técnicas comunes, algunas tales como el enfoque de retención que implica la partición aleatoria de los datos en dos conjuntos independientes: uno de entrenamiento en donde el modelo seleccionado se desarrolla y otro de prueba en donde este se evalúa; el submuestreo aleatorio es una variante del método holdout en la que se repite el proceso k veces, y la precisión general se calcula como el promedio de las precisiones obtenidas en cada iteración; la validación cruzada es un proceso en el cual los datos se dividen en k subconjuntos exclusivos. Se realiza el entrenamiento y prueba k veces. En cada iteración, un subconjunto se reserva para prueba, y los demás se usan para entrenar el modelo. Cada muestra se utiliza igualmente para entrenamiento y prueba. La precisión para la clasificación

se obtiene como el número total de clasificaciones correctas en las k iteraciones dividido por el número total de tuplas en los datos iniciales; los métodos Bootstrap que, a diferencia de los métodos de estimación de precisión mencionados anteriormente, este selecciona las tuplas de entrenamiento dadas de manera uniforme y con reemplazo (Han et al., 2012).

4.3.3. *Búsqueda en cuadrícula*

En el proceso de optimización de hiperparámetros, la búsqueda en cuadrícula (*Grid Search*) se presenta como un enfoque exploratorio ampliamente utilizado debido a su capacidad para realizar una búsqueda exhaustiva en el espacio de hiperparámetros. Este método evalúa el rendimiento de todas las combinaciones posibles de hiperparámetros, asegurando que se identifique la configuración óptima para maximizar el rendimiento del modelo. Su naturaleza independiente implica que cada combinación única de hiperparámetros es probada de manera sistemática, lo que lo convierte en una técnica robusta pero computacionalmente intensiva (Ogunsanya et al., 2023).

4.3.4. *Curva de aprendizaje*

Las curvas de aprendizaje son herramientas utilizadas para analizar el rendimiento predictivo de un modelo a medida que varía la cantidad de esfuerzo de aprendizaje. En el contexto del aprendizaje automático, estas curvas generalmente representan la precisión predictiva en un conjunto de prueba como una función del número de ejemplos de entrenamiento utilizados. Esto permite evaluar la capacidad de generalización del modelo y diagnosticar problemas como el sobreajuste o el subajuste (Perlich, 2011).

Originalmente, las curvas de aprendizaje se introdujeron en la psicología educativa y conductual para medir el impacto del esfuerzo de aprendizaje en la memoria y la productividad. En el aprendizaje automático, se utilizan principalmente en dos contextos:

En redes neuronales artificiales, para mostrar cómo diverge el rendimiento dentro y fuera de la muestra con respecto al número de iteraciones de entrenamiento.

En el aprendizaje automático general, para ilustrar el rendimiento de generalización predictiva en función de los datos de entrenamiento disponibles (Perlich, 2011).

4.4. Técnicas de re-muestreo

Las técnicas de re-muestreo han sido propuestas como una solución potencial al problema de desequilibrio de clases, considerando el preprocesamiento de datos como un paso esencial para construir modelos efectivos utilizando algoritmos modernos de minería de datos. Entre los métodos más comunes se encuentran el *submuestreo aleatorio* y el *sobremuestreo aleatorio*, que buscan equilibrar las clases mediante la eliminación de ejemplos de la clase mayoritaria o la duplicación de ejemplos de la clase minoritaria, respectivamente. Sin embargo, estas técnicas tienen limitaciones: el submuestreo puede llevar a la pérdida de información, mientras que el sobremuestreo puede causar sobreajuste. Para abordar estas desventajas, se han propuesto técnicas alternativas más avanzadas, como SMOTE (Khairy et al., 2024).

4.4.1. *Submuestreo aleatorio (Random UnderSampling, RUS)*

Esta técnica reduce el tamaño de la muestra de la clase mayoritaria hasta alcanzar una distribución relativamente equilibrada, al descartar aleatoriamente estos ejemplos se intenta eliminar los efectos negativos de una distribución sesgada (Guo et al., 2019).

4.4.2. *Sobremuestreo aleatorio (Random OverSampling, ROS)*

Esta técnica crea nuevos ejemplos de la clase minoritaria duplicando aleatoriamente los ejemplos de la clase minoritaria de manera de que la distribución de clases sea más equilibrada (Guo et al., 2019).

4.4.3. *Sobremuestreo sintético de la minoría (Synthetic Minority Oversampling Technique, SMOTE)*

Es un algoritmo de sobremuestreo diseñado mediante la generación sintética de muestras de conceptos subrepresentados (Gulowaty & Ksieniewicz, 2019). Por lo que, en lugar de replicar los datos originales, el método genera instancias sintéticas que se crean basándose en la similitud de sus atributos con las instancias existentes y sus vecinos más cercanos (Bobbili & Cretu, 2018).

4.5. Métodos para ensamblar algoritmos

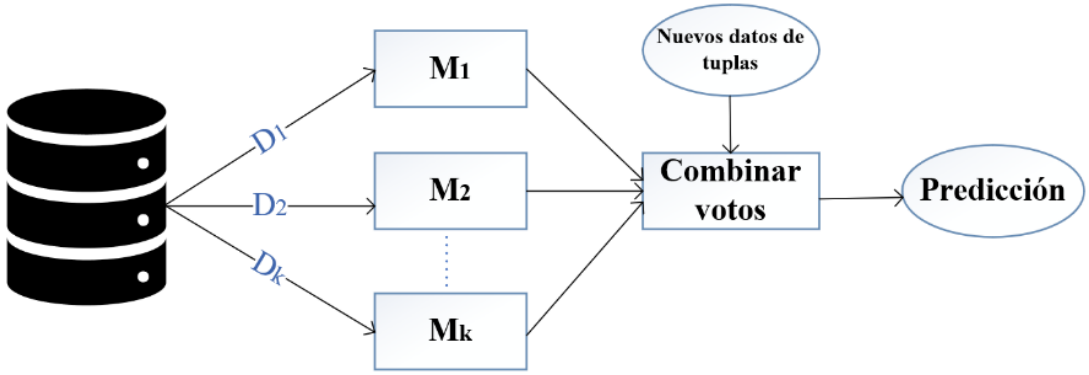
Los métodos de ensamblado son técnicas diseñadas para mejorar la precisión de los modelos de predicción combinando los resultados de múltiples clasificadores bases más simples. La idea fundamental consiste en construir un modelo final que aproveche las fortalezas individuales de estos clasificadores base, mitigando sus debilidades mediante un enfoque colectivo (Hastie et al., 2009). En estos métodos, los clasificadores base emiten votos y la predicción final del ensamblado se determina mediante la agregación de esos votos. Este proceso puede realizarse mediante esquemas como la votación mayoritaria o el promedio ponderado, dependiendo del tipo de problema y los clasificadores involucrados (Han et al., 2012). En la figura 1 se presenta el proceso general de un clasificador ensamblado, el cual es esencial para comprender la naturaleza de estos métodos, como el Random Forest y otros modelos avanzados.

Una de las principales ventajas de los métodos de ensamblado es su capacidad para reducir la variabilidad en las predicciones y, por lo tanto, disminuir el error global del modelo (Sumathi & Sivanandam, 2006). Aunque los clasificadores base pueden cometer errores individuales, el conjunto final predice incorrectamente una instancia solo si más de la mitad de los clasificadores base fallan. Por esta razón, los ensamblados tienden a ofrecer resultados superiores cuando existe

una diversidad significativa entre los clasificadores base; es decir, cuando la correlación entre ellos es baja (Han et al., 2012).

Figura 1

Naturaleza métodos ensamblados



Nota: M1: Modelo 1; M2: Modelo 2; M3: Modelo 3. Adaptado de (Han et al., 2012)

La diversidad entre los clasificadores base es un factor clave para el éxito de los métodos de ensamblado. Para lograrla, se pueden emplear diferentes estrategias. Una opción es usar distintos conjuntos de entrenamiento dividiendo el conjunto de datos en subconjuntos y entrenando clasificadores individuales con diferentes subconjuntos. Otra estrategia consiste en emplear subconjuntos distintos de características para entrenar los clasificadores, explorando así diferentes espacios de predicción. También se puede introducir diversidad utilizando diferentes modelos de clasificadores, como árboles de decisión, máquinas de soporte vectorial y redes neuronales, o aplicar métodos variados para combinar las predicciones, como votación mayoritaria, promedio ponderado o enfoques bayesianos (Kuncheva, 2001). Estas estrategias aseguran que los errores de los clasificadores base no estén correlacionados, maximizando así la robustez y precisión del modelo ensamblado.

4.5.1. *Método Bagging*

El método bagging se basa en el concepto de que, al promediar un conjunto de observaciones, se reduce la varianza. Su principal objetivo principal es disminuir la varianza y mejorar la precisión del conjunto de prueba. Esto se logra al tomar múltiples conjuntos de entrenamiento a partir de la población, construyendo un modelo de predicción separado con cada uno de ellos y luego promediando las predicciones resultantes. En otras palabras, se calcula la predicción de cada clasificador utilizando B conjuntos de entrenamiento distintos, generados mediante la técnica Bootstrap, que consiste en tomar muestras repetidas del único conjunto de datos de entrenamiento disponible. Finalmente, las predicciones se promedian para obtener un único modelo de aprendizaje estadístico con baja varianza (James et al., 2013).

4.5.1.1. **Random Forest.**

El bosque aleatorio, también denominados Random Forest, son clasificadores que consta de una colección de árboles de decisión. Cada árbol se construye utilizando un algoritmo A, un conjunto de entrenamiento S y un vector aleatorio adicional θ, donde θ se muestrea independientemente de una cierta distribución. La predicción del bosque aleatorio se obtiene mediante una votación mayoritaria entre las predicciones de los árboles individuales (Shalev-Shwartz & Ben-David, 2014a).

Cada árbol en el bosque depende de los valores de un vector aleatorio θ, que es muestreado de manera independiente con la misma distribución para todos los árboles. A medida que el número de árboles en el bosque aumenta, el error de generalización del modelo converge casi seguramente a un límite. Este error de generalización depende de dos factores clave: la fortaleza de los árboles individuales en el bosque y la correlación entre ellos. Una menor correlación entre los árboles y

una mayor fortaleza de los clasificadores individuales conducen a un mejor rendimiento del modelo (Breiman, 2001).

4.5.2. *Método Boosting*

Este enfoque utiliza una generalización de predictores lineales con el objetivo de buscar un buen equilibrio entre el sesgo y la complejidad. El proceso de aprendizaje comienza con una clase básica, que podría presentar un error de aproximación considerable, y a medida que avanza, se incrementa la capacidad representativa de la clase a la que puede pertenecer el predictor. De esta manera, el boosting proporciona una herramienta para combinar hipótesis débiles, fáciles de aprender y que superan el rendimiento del azar, aproximándose gradualmente a predictores más robustos para clases más complejas y difíciles de aprender (Shalev-Shwartz & Ben-David, 2014a).

4.5.2.1. **XGBoost.**

XGBoost, o *Refuerzo de Gradiente Extremo* (Extreme Gradient Boosting), es un algoritmo avanzado de *Árbol de Decisión con Refuerzo de Gradiente* (Gradient Boosting Decision Tree, GBDT). Diseñado para optimizar la función de pérdida mediante el cálculo de gradientes negativos y un término de regularización, este enfoque permite reducir la complejidad del modelo, suavizar la función objetivo y evitar el sobreajuste. A diferencia del GBDT clásico, XGBoost incorpora técnicas avanzadas como la expansión de Taylor de segundo orden y algoritmos de búsqueda codiciosa para optimizar divisiones, mejorando así la eficiencia y precisión en conjuntos de datos de gran escala (Chang et al., 2022; Xia et al., 2017).

Cada árbol de decisión en XGBoost se construye de manera aditiva para minimizar una función de pérdida regularizada, combinando estadísticas de gradiente de primer y segundo orden. Además, XGBoost emplea hiperparámetros avanzados como el coeficiente de

regularización (γ) y el submuestreo de columnas para controlar la complejidad del modelo y reducir el riesgo de sobreajuste (Xia et al., 2017).

4.6. Metodología KDD

El proceso de Descubrimiento de Conocimiento en Bases de Datos (por sus siglas en inglés, KDD-Knowledge Discovery in Databases) comprende la extracción de patrones en forma de reglas o funciones a partir de los datos, con el objetivo de que el usuario pueda analizarlos. Esta tarea generalmente abarca el preprocesamiento de los datos, la minería de datos y la presentación de los resultados (Timarán Pereira et al., 2016).

Los patrones o tendencia extraídas facilitan la predicción de comportamientos en un contexto específico. Este proceso abarca desde la limpieza de datos, eliminando ruido y datos incoherentes, hasta la presentación de conocimiento mediante técnicas de visualización. Así, después de la limpieza, se integra los datos, se realiza una selección de información relevante y luego se transforman los datos para su adecuada consolidación. Consecuentemente, se emplea la minería de datos para extraer patrones y finalmente se realiza la evaluación de dichos patrones para identificar conocimiento significativo y representar el conocimiento adquirido (Han et al., 2012).

Metodología

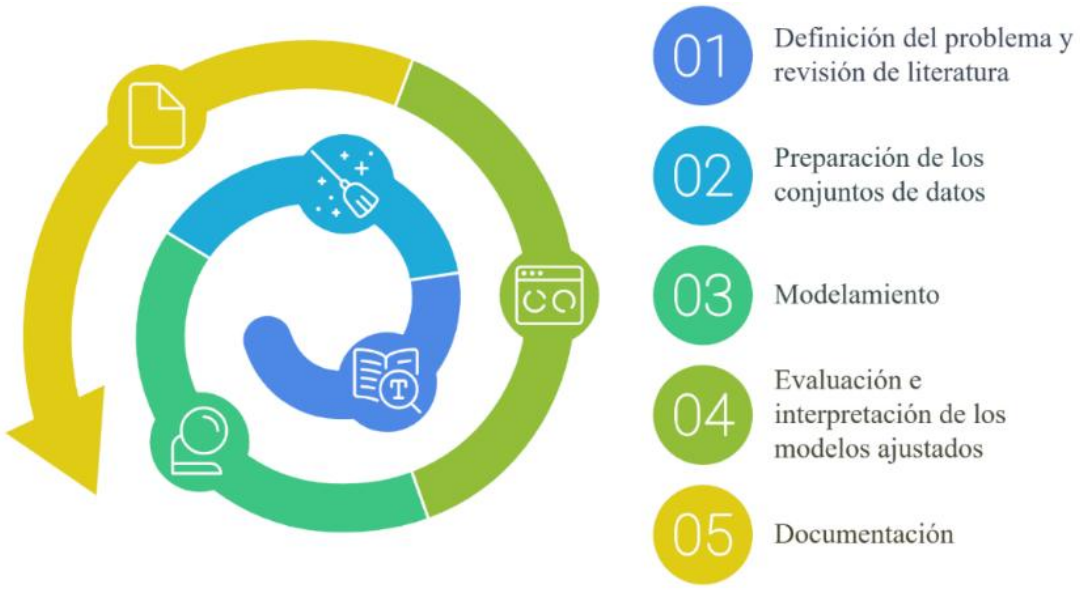
Para el desarrollo de esta investigación se aplicó la metodología de descubrimiento de conocimiento en bases de datos (Knowledge Discovery in Databases, KDD). Esta metodología fue seleccionada debido a su capacidad para proporcionar un proceso organizado de identificación de patrones válidos, novedosos, útiles y comprensibles a partir de grandes y complejos conjuntos de datos (Maimon & Rokach, 2010), alineándose con el objetivo de esta investigación: *Analizar la*

clasificación de datos desbalanceados a partir de modelos de aprendizaje automático ensamblados.

Las fases metodológicas consideradas, se presentan en la Figura 2 y son detalladas a continuación.

Figura 2

Estructura metodológica de la investigación



Fase 1: Revisión de literatura

En esta primera fase, se buscó identificar los enfoques que dan solución a los conjuntos de datos con clases desbalanceadas a partir de modelos de aprendizaje automático ensamblados y definir el problema de interés, para lo cual se llevó a cabo una revisión de literatura. Inicialmente, se definieron las palabras clave y los términos de búsqueda necesarios para garantizar la pertinencia de los resultados obtenidos en la base de datos Scopus. Posteriormente, se diseñó una ecuación de búsqueda que optimizara la recuperación de información en dicha base. Una vez seleccionados los documentos, se realizó un análisis bibliométrico para identificar tendencias, fuentes relevantes y patrones en la literatura. Finalmente, la información recopilada fue clasificada

y analizada según las temáticas específicas del estudio, permitiendo identificar las estrategias principales propuestas para abordar conjuntos de datos desbalanceados mediante modelos de aprendizaje automático ensamblados.

Fase 2: Preparación del conjunto de datos en Python con librerías adecuadas de ML

En esta fase, se seleccionaron y extrajeron dos conjuntos de datos relevantes de la plataforma *Kaggle* y del repositorio *UC Irvine Machine Learning Repository*, garantizando su pertinencia para la tarea de clasificación desbalanceada. Posteriormente, tras validar la estructura de los conjuntos de datos, se realizó un análisis exploratorio utilizando Python para comprender la distribución de las clases y caracterizar el grado de desbalance presente. Como parte del preprocesamiento, se aplicaron técnicas de re-muestreo ampliamente respaldadas por la literatura, como submuestreo aleatorio (Random Undersampling, RUS), sobremuestreo aleatorio (Random Oversampling, ROS) y la técnica de sobremuestreo de minorías sintéticas (Synthetic Minority Oversampling Technique, SMOTE). Estas estrategias se implementaron mediante librerías de Python, entre ellas *sklearn*, *matplotlib.pyplot*, *pandas* y *numpy*, asegurando un tratamiento adecuado de los datos para el desarrollo de los modelos de clasificación.

Fase 3: Modelamiento

En esta fase, se ajustaron modelos de aprendizaje automático ensamblados siguiendo las técnicas más utilizadas identificadas durante la revisión de literatura, específicamente bagging con Random Forest y boosting con XGBoost. Este proceso se llevó a cabo utilizando Python y apoyándose en librerías como sklearn, matplotlib.pyplot, pandas, numpy, entre otras, asegurando una implementación eficiente y reproducible.

Fase 4: Evaluación e interpretación

En esta fase, se validó y comparó el rendimiento de los modelos ajustados utilizando Python como herramienta principal para el procesamiento y análisis. Posteriormente, los resultados de la investigación fueron interpretados, analizados y sintetizados.

Fase 5: Despliegue y documentación

En esta etapa final, se elaboró un artículo de carácter publicable en el que se detallaron los principales hallazgos del proyecto de investigación, siguiendo los estándares académicos requeridos para su divulgación. Además, se desarrolló un documento final que recopiló y representó de manera integral los resultados obtenidos a lo largo del trabajo de investigación. Como parte de esta etapa, se empleó la herramienta de visualización Power BI para generar una representación clara y estructurada de los hallazgos, asegurando la presentación adecuada de los datos.

Fase 1: revisión de literatura

En esta sección, se exponen los principales resultados de la revisión de literatura con el objetivo de adquirir conocimientos relevantes que ofrezcan una aproximación real al problema de investigación. En una primera fase, se realiza un análisis bibliométrico para evaluar la actividad científica mediante la medición del impacto de los documentos científicos en el área temática de interés, utilizando métricas que abarcan aspectos como el contenido, la frecuencia de publicación de la revista, y la disponibilidad, entre otros. Posteriormente, al recopilar la bibliografía necesaria, se lleva a cabo un análisis y una comprensión de los distintos argumentos de los artículos.

6.1. Análisis Bibliométrico

En el marco de este estudio, se llevó a cabo un análisis bibliométrico con el propósito de identificar enfoques que dan solución a los conjuntos de datos con clases desbalanceadas a partir

de modelos de aprendizaje automático ensamblados o la combinación de estos modelos ensamblados con soluciones enfocadas en el preprocesamiento del conjunto de datos desequilibrado. Inicialmente, el diseño de la estrategia de búsqueda se orientó hacia un desglose adecuado del problema para una exploración precisa. De este modo, se dividió el problema en cuatro categorías principales: “aprendizaje automático”, “desbalance de clases”, “clasificadores ensamblados” y “técnicas de re-muestreo”, con el fin de encontrar inicialmente las palabras claves, sinónimos y/o tesauros distintivos del problema de estudio, los cuales se presentan en la tabla 4.

Tabla 4.

Palabras claves, sinónimos y/o tesauros seleccionados para la ecuación de búsqueda

Categoría	***Tesauros***
Machine learning (aprendizaje automático)	supervised learning
Class-imbalanced (desbalance de clases)	imbalanced data, imbalanced problem, imbalanced classification, imbalanced class distribution, unbalanced data
Classifier ensembles (clasificadores ensamblados)	classification algorithms, classifier, classification technique, assembled classifiers, classifier ensemble, ensembles of classifiers, ensemble model, ensemble method
Resamplig techniques (técnicas de remuestreo)	oversampling, undersampling, resampling

Posteriormente, al combinar estas palabras claves en la base de datos referencial y multidisciplinar SCOPUS mediante operadores lógicos y después de diversas iteraciones, se formuló la siguiente ecuación de búsqueda:

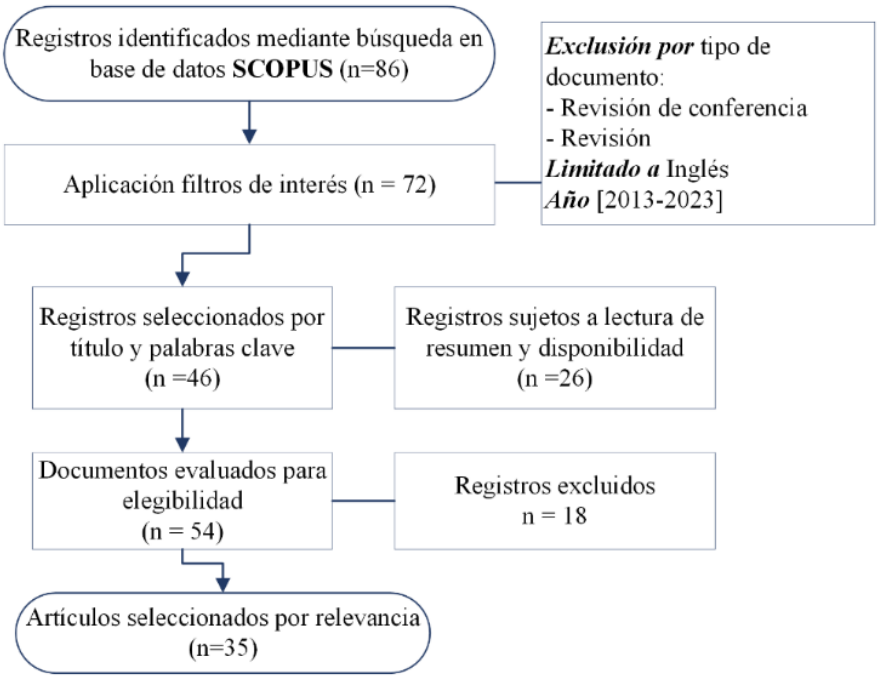
(TITLE-ABS-KEY ("Class-imbalanced" **OR** "Class-imbalance" **OR** "Imbalanc* data" **OR** "Imbalanc* problem" **OR** "Imbalanc* classification" **OR** "Imbalanc* class distribution" **OR** "unbalanced data") **AND** TITLE-ABS-KEY (“resampling techniques” **OR** resampling **OR**

oversampling **OR** undersampling) **AND** TITLE-ABS-KEY ("Machine learning" **OR** "supervised learning") **AND** TITLE-ABS-KEY ("classification algorithm*" **OR** classifier **OR** "classification technique") **AND** TITLE-ABS-KEY ("Assembled classifier*" **OR** "Classifier ensemble*" **OR** "Ensembles of classifier*" **OR** "ensemble model*" **OR** "ensemble method")).

Con base en esta ecuación de búsqueda, como se observa en la figura 3, se identificaron inicialmente 86 documentos. Se aplicaron filtros de inclusión, como periodo de publicación comprendido entre 2013 y 2023, y se limitó el idioma al inglés. Además, se excluyeron documentos tales como las revisiones. Tras estos filtros, se obtuvieron 72 artículos.

Figura 3

Flujograma de ecuación de búsqueda - julio 2023



En una primera revisión, se seleccionaron 46 mediante la evaluación de títulos y palabras. Posteriormente, se llevó a cabo la lectura del resumen de los 26 artículos restantes para evaluar su relación con el problema planteado, considerando el uso de modelos de aprendizaje automático ensamblados, de los cuales se seleccionan 8 de estos artículos. Finalmente, basándose en el criterio

de relevancia que otorga la base de datos SCOPUS, que considera indicadores de impacto como la cantidad de citas que ha recibido el artículo, la fuente de la revista, fecha de publicación, entre otros, se eligieron 35 artículos de los 54 evaluados para elegibilidad, con el fin de llevar a cabo el análisis y revisión de literatura correspondiente.

Finalmente, en el apéndice A, se encuentra disponible el análisis bibliométrico realizado, que incluye el estudio por año de publicación, países, áreas de investigación, autores y coocurrencia de palabras claves.

6.2. Análisis Preliminar de Literatura

Para abordar de manera efectiva la tarea de clasificación cuando un modelo predictivo se enfrenta al problema de *desequilibrio de clases*, es esencial no ceñirse únicamente a la aplicación de algoritmos de aprendizaje automático. Si bien la mayoría de estos algoritmos, algunos tales como los modelos de árboles de decisión para la clasificación y regresión (CART), las redes neuronales o el algoritmo de K-vecinos más cercanos (KNN), han demostrado un rendimiento superior al lograr tasas de error promedio más bajas en comparación con técnicas estadísticas tradicionales, como el modelo Probit (Galindo & Tamayo, 2000). Estos algoritmos o clasificadores al enfrentarse a un conjunto de datos con desequilibrio de clases grave o con datos poco frecuentes, no son capaces de discriminar adecuadamente entre las clases mayoritarias y las clases minoritarias (Hasanin et al., 2019).

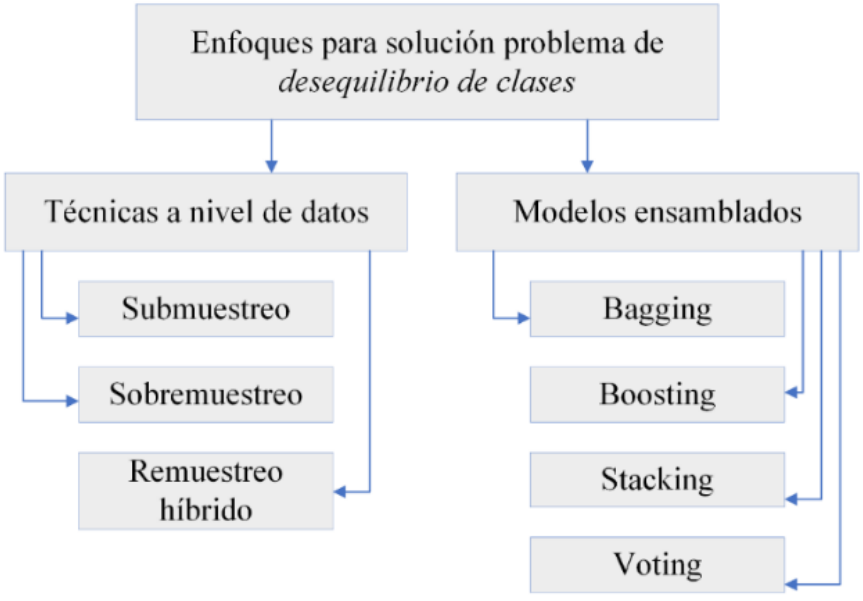
Esto se debe a que, cuando se aplica un clasificador sin acompañarlo de alguna estrategia para procesar los datos desequilibrados, existe una tendencia a ignorar la clase minoritaria (Malek et al., 2023) y considerarla como ruido (Nanni et al., 2015), debido a los supuestos establecidos para el funcionamiento del modelo, que asume una distribución equitativa de las clases y tasas de error uniformes para todas las instancias (Mwangi et al., 2022). Como resultado, el clasificador

tiende a etiquetar a casi todos los casos como pertenecientes a la clase mayoritaria, lo que puede llevar a una métrica de precisión engañosamente alta (Hasanin et al., 2019), a pesar de un rendimiento real deficiente en términos de clasificación. Por lo tanto, se hace necesario explorar y analizar los diversos enfoques existentes destinados a tratar la complejidad del problema y mejorar el rendimiento de los modelos de aprendizaje automático orientados a la clasificación.

Dos de los enfoques ampliamente utilizados en el campo del aprendizaje automático son las técnicas de re-muestreo y los modelos de aprendizaje automático ensamblado, destacándose en su significativa contribución a la mejora del rendimeinto de los modelos (Guo et al., 2019). Sin embargo, los modelos ensamblados se destacan en comparación con el re-muestreo de los datos e incluso con algoritmos de clasificación no ensamblados o el proceso de selección de características; ya que, son capaces de combinar múltiples técnicas a nivel de datos y/o algoritmos para la optimización de los clasificadores, y así, aliviar indirectamente el efecto del problema de desequilibrio de datos (Quan et al., 2022).

Figura 4

Enfoques principales para abordar el problema de desequilibrio de clases



Adaptado de: (Balasubramanian et al., 2020; H. I. Lin & Nguyen, 2020; Mwangi et al., 2022; Pouriyeh et al., 2017a).

En la figura 4, se llevó a cabo un análisis que recalca la diferenciación general de los enfoques empleados en los artículos seleccionados. De esta forma, las técnicas a nivel de datos se engloban en prácticas como el submuestreo, sobremuestreo y el re-muestreo híbrido. Por otro lado, los modelos ensamblados pueden clasificarse principalmente en técnicas como bagging, boosting, staking (H. I. Lin & Nguyen, 2020; Mwangi et al., 2022; Pouriyeh et al., 2017a) y voting (Balasubramanian et al., 2020).

Estas técnicas mencionadas al ser utilizadas de manera independiente, en la mayoría de los casos producen resultados satisfactorios. No obstante, al adoptar un enfoque que combina técnicas a nivel de datos con modelos ensamblados, es factible potenciar aún más el rendimiento del modelo. Este enfoque implica entrenar un clasificador único mediante técnicas como el bagging o el boosting, luego de haber realizado el preprocesamiento de los datos. A continuación, se describen detalladamente estas técnicas y el enfoque mencionado.

Las técnicas de re-muestreo, también conocidas como soluciones a nivel de datos (Lin et al., 2017), se aplican en la etapa de preprocesamiento de los datos (Inan et al., 2021). Estas técnicas se consideran métodos externos, ya que involucran la creación de un conjunto de entrenamiento equilibrado mediante la reducción de la clase mayoritaria o el incremento de la clase minoritaria (Sun et al., 2018). Se dividen en dos categorías principales: submuestreo y sobremuestreo, en donde las formas más simples de aplicación son el submuestreo aleatorio (RUS) y el sobremuestreo aleatorio (ROS), respectivamente (Mwangi et al., 2022).

En un experimento para la detección de reseñas falsas en línea, se compararon las técnicas RUS y ROS. Ambos métodos fueron eficaces para abordar el problema de desequilibrio de clases al aumentar significativamente la precisión de la clase minoritaria en comparación con no utilizar

alguna técnica para balanceo. La técnica de RUS, en promedio, ofreció resultados superiores a la técnica de ROS en términos de precisión debido a que el ROS no se desempeñó bien cuando se aplicó en conjuntos de datos pequeños (Budhi et al., 2021). Sin embargo, es importante destacar que estas técnicas presentan limitaciones (Honnurappa & Raghavendra, 2021), como la pérdida de información en la clase mayoritaria o al sobreajuste de la clase minoritaria (Mwangi et al., 2022), lo que puede dar como resultado predicciones incorrectas debido a la alta varianza generada por la naturaleza aleatoria de estos métodos (Ng et al., 2017).

Ahora bien, existe un enfoque adicional que se deriva de las técnicas de re-muestreo y combinan tanto el submuestreo como el sobremuestreo, conocido como muestreo *híbrido.* En este método, se aplican pautas específicas, según el problema a abordar, para aumentar la clase minoritaria y reducir la clase mayoritaria, generando así un conjunto de datos mixto adecuado. Este conjunto de datos se remuestrea generalmente de manera aleatoria para garantizar que los datos de entrenamiento sean altamente variables, mitigando así los efectos de la pérdida de datos debido al submuestreo y del sobreajuste debido al sobremuestreo (H. I. Lin & Nguyen, 2020). En particular, para este enfoque, destaca el modelo híbrido SMOTETomek destaca como una técnica híbrida (Zheng et al., 2021) que permite aprovechar tanto los beneficios del submuestreo como del sobremuestreo.

En un estudio similar al anterior, pero en el contexto de la calidad del agua, se llevó a cabo una comparación del rendimiento de siete modelos de aprendizaje automático utilizando tanto la técnica de ROS como el RUS, además de explorar el enfoque híbrido ROS-RUS. Los resultados revelaron que la combinación del método híbrido, en combinación con el clasificador ensamblado Random Forest, condujo a mejoras significativas en términos de precisión, especificidad y la

puntuación F-1; y adicionalmente la técnica ROS generalmente tuvo un mejor rendimiento, pero con una ventaja mínima, seguido por el RUS (Malek et al., 2023).

En consecuencia, se han desarrollado técnicas avanzadas debido a las falencias descritas por RUS y ROS; y como en términos generales, se ha observado que el submuestreo tiende a obtener un mejor rendimiento comparado con el sobremuestreo (W. C. Lin et al., 2017). Esta tendencia ha motivado una mayor exploración y aplicación de técnicas de submuestreo en investigaciones. Sin embargo, cabe recalcar cómo se han dedicado esfuerzos a la búsqueda de nuevas formas para mejorar la eficacia del sobremuestreo, como se ilustra la figura 5.

Figura 5

Técnicas avanzadas de sobremuestreo

En este sentido, algunas técnicas de submuestreo se basan en los centroides de agrupamiento (Zheng et al., 2021) y en métodos como la edición de vecinos más cercanos (ENN) o la consideración de todos los vecinos más cercanos (ALLKNN) (Balasubramanian et al., 2020) para abordar el tratamiento de las clases. Esto se refleja, por ejemplo, en el submuestreo basado en clusterización empleando el algoritmo k-means (W. C. Lin et al., 2017). El submuestreo

incremental aleatorio (IRUS) representa otra técnica empleada, donde en lugar de eliminar aleatoriamente las muestras de la clase mayoritaria, se dividen aleatoriamente las muestras de la clase mayoritaria en partes más pequeñas, cada una con menos muestras que la minoría y luego combina cada parte de la mayoría con todas las muestras de la minoría para crear conjuntos de datos equilibrados (Ng et al., 2017). Adicionalmente, se han empleado técnicas avanzadas, como el algoritmo genético, para llevar a cabo el submuestreo, respaldando como esta técnica es significativamente superior que la mayoría de las técnicas de muestreo existentes (Krawczyk et al., 2016; Sun et al., 2018).

Por otra parte, los modelos de aprendizaje automático ensamblados se aplican después de las técnicas de preprocesamiento, en la fase de procesamiento del análisis de datos. En esta etapa, se realizan predicciones de clases en función de la técnica de ensamblado empleada (Balasubramanian et al., 2020). Entre los métodos de aprendizaje ensamblado, los modelos de *Bagging* nacen de la combinación de predicciones exactamente del mismo tipo a través de votación; *boosting* es lo mismo que el bagging, salvo que el rendimiento de los modelos anteriores influye en los nuevos modelos, mientras que *Stacking* es la combinación de modelos de diferentes tipos (Pouriyeh et al., 2017b). Por otro lado, en el enfoque voting, las predicciones generadas por los múltiples modelos de clasificación empleados se agregan a través de un mecanismo de votación, donde cada modelo contribuye con su predicción, y la predicción final se determina mediante el voto mayoritario entre estos modelos (Balasubramanian et al., 2020).

En un estudio para la detección de fraudes en tarjetas de crédito en un conjunto altamente desequilibrado con un número total de casos de fraude de 492 de un total de 284,807 transacciones, se evaluaron dos clasificadores individuales: KNN y regresión logística (LR), junto con dos modelos ensamblados: el clasificador Bagging con árboles de decisión como clasificador base y

Random Forest (RF). Además, se llevaron a cabo experimentos utilizando dos métodos tradicionales de sobremuestreo, como SMOTE y ADASYN. Los resultados revelaron que los modelos ensamblados presentaron un rendimiento promedio superior en comparación con los modelos individuales. Además, se observó que es posible mejorar significativamente el rendimiento del modelo ensamblado al combinarlo estratégicamente con técnicas tradicionales de sobremuestreo. En particular, se encontró que la aplicación de RF después de equilibrar los datos con SMOTE condujo a un rendimiento razonablemente bueno y superó a varias combinaciones posibles de enfoques (Mondal et al., 2021). De igual forma, como en la detección de fraudes, existen otras aplicaciones en las cuales se usan los modelos ensamblados como se muestra en la tabla 5 en la que se puede conocer los modelos en torno a la calificación crediticia, predicción de defectos de software y enfermedades cardiovasculares.

Tabla 5

Modelos ensamblados según aplicaciones

Aplicación	**Modelo ensamblado**
Calificación crediticia	Gradient boosting, decision tree (GBDT), Random Forest y Rotation Forest superan en rendimiento a los clasificadores ensamblados en comparación con las técnicas de Bagging y Boosting con algunos clasificadores base tales como regresión logística (LR), árboles de clasificación y regresión (CART) y máquinas de soporte vectorial (SVM) (Cao et al., 2021).
Predicción de defectos de Software	Emplean una combinación de las técnicas de Bagging y Boosting con árboles de decisión y redes Bayesianas como clasificadores base. Los resultados experimentales mostraron que los métodos propuestos combinados con la técnica SMOTE mejoran el rendimiento de los modelos en comparación con los modelos individuales (Balogun et al., 2020).
Enfermedades cardiovasculares	El modelo de predicción ensamblado por apilamiento propuesto con los clasificadores base LR, SVM, KNN, DT, RF, XGBoost y AdaBoost; y el meta aprendiz RF, combinado con la técnica de muestreo híbrido SMOTETomek obtuvo las mejores medidas de rendimiento general. Sin embargo, RF, XGBoost y AdaBoost obtuvieron mejores rendimientos que los clasificadores base LR, SVM, KNN, and DT, y especialmente RF

	mostró una notable mejora al combinarlo ya sea con técnicas de sobremuestreo, submuestreo y muestreo híbrido (Zheng et al., 2021).

Por otra parte, en la figura 6 y 7, se proporciona el resumen de los enfoques más utilizados tanto para los modelos de aprendizaje automático ensamblados como las soluciones a nivel de datos. Es importante destacar que, con el fin de facilitar su visualización, cada estrategia, como mínimo, debía contar con 2 ocurrencias para su inclusión, excluyendo el enfoque de stacking debido a su carácter reciente. El orden de presentación se basa en la relevancia de los artículos, según lo registrado en su selección.

Figura 6

Modelos de aprendizaje automático ensamblados más empleados

Autores	***BAGGING(B)***		***BOOSTING***		***VOTING***	***STACKING***
	RF	**MLP**	**AdaBoost**	**XGBoost**		
(Mwangi et al., 2022)	X					
(Mondal et al., 2021)	X					
(Cao et al., 2021)	X					
(Zheng et al., 2021)	X		X	X		X
(Budhi et al., 2021)			X			
(Balasubramanian et al., 2020)	X				X	
(Bobbili & Cretu, 2018)					X	
(Chang et al., 2022)				X		
(Malek et al., 2023)	X					
(Inan et al., 2021)				X		
(Diallo et al., 2021)		X				
(Artetxe et al., 2020)					X	
(Balogun et al., 2020)			X			
(W. C. Lin et al., 2017)	X	X				

Figura 7

Soluciones más utilizadas a nivel de datos

Autores	***REMUESTREO***				
	ROS	**RUS**	**SMOTE**	**Clustering *k-means***	**ADASYN**

(Mwangi et al., 2022)	X				
(Mondal et al., 2021)			X		X
(Zheng et al., 2021)			X		
(Budhi et al., 2021)	X	X			
(Balasubramanian et al., 2020)			X		X
(Bobbili & Cretu, 2018)			X		
(Chang et al., 2022)			X		
(Malek et al., 2023)	X	X			
(Inan et al., 2021)			X		
(Artetxe et al., 2020)		X			
(Balogun et al., 2020)			X		
(Duan et al., 2020)				X	
(W. C. Lin et al., 2017)				X	
(Nanni et al., 2015)			X		

En general los enfoques mayormente se aplicaron a conjunto de datos con clases binarias; sin embargo, cabe destacar como para escenarios que involucran más de dos clases, como por ejemplo en la clasificación de datos hiperespectrales multiclase, se ha observado que la versión estándar del modelo ensamblado Random Forest no resulta adecuada para abordar el desequilibrio de clases en comparación con el re-muestreo aleatorio (Quan et al., 2022).

En el ámbito de las métricas de rendimiento, especialmente en aplicaciones médicas y en entornos donde las clases desbalanceadas plantean un desafío, el Recall emerge como una métrica crucial. También conocido como recuperación o sensibilidad, el Recall se convierte en un indicador esencial al monitorear y reducir el número de falsos positivos, contribuyendo así a mejorar el rendimiento del modelo ensamblado (Balasubramanian et al., 2020). Adicionalmente, dado que la precisión no logra capturar completamente el rendimiento del clasificador, se sugiere recurrir a la métrica de la media geométrica armónica y otras medidas tales como la sensibilidad y la precisión.

Como recomendaciones, algunas investigaciones destacan la importancia de explorar nuevas formas de mejorar la especificidad y la puntuación AUC (Mwangi et al., 2022), y

precisamente en este contexto de desequilibrio de clases, la utilización de múltiples métricas de rendimiento se presenta como una práctica recomendada. La diversidad en estas métricas, cada una empleando los valores de la matriz de confusión de manera específica, ofrece una visión más completa y fundamentada de los resultados. Así, se fortalece la evaluación del modelo, permitiendo una comprensión más profunda de su rendimiento en diversas dimensiones (Malek et al., 2023).

En el cierre de la presente revisión de literatura, se identificó que, para abordar eficazmente el desequilibrio de clases en problemas de clasificación, es esencial reconocer que la aplicación de algoritmos de aprendizaje automático de clasificación no ensamblados no es suficiente. En cambio, se resalta la destacada versatilidad y eficacia de las técnicas de muestreo y los modelos de aprendizaje automático ensamblado para potenciar significativamente el rendimiento de los modelos, evidenciando la relevancia de enfoques como bagging con Random Forest, boosting con Adaboost y XGBoost, así como otros métodos como voting o stacking. En el ámbito de las soluciones a nivel de datos, cobran especial protagonismo técnicas como ROS, RUS y SMOTE. Asimismo, la evaluación del rendimiento se enriquece con métricas específicas como el Recall o la media geométrica armónica, facilitando comparativas rigurosas entre modelos de aprendizaje automático, ya sean ensamblados o individuales.

Fase 2: preparación de los conjuntos de datos

En esta fase, se trabajó con dos conjuntos de datos desbalanceados, con un desbalance extremo y con un desbalance moderado. Ambos conjuntos fueron preprocesados mediante normalización para garantizar la homogeneidad de sus variables. Además, se implementaron técnicas de re-muestreo como RUS, ROS y SMOTE. Estas estrategias prepararon los datos para

el desarrollo de modelos predictivos, permitiendo abordar los retos asociados al desbalance de clases de manera robusta y adaptada a las características de cada conjunto.

7.1. Extracción de conjuntos de datos

Se llevaron a cabo experimentos utilizando dos conjuntos de datos desbalanceados con clases binarias, obtenidos de la plataforma Kaggle y el repositorio UC Irvine Machine Learning Repository. El primer conjunto de datos, caracterizado por un desbalance extremo entre clases, fue seleccionado debido a su relevancia en la detección de fraudes, un problema crítico en el ámbito financiero. Por otra parte, el segundo conjunto de datos, con desbalance leve, abordó la problemática de impagos financieros. Ambos conjuntos se diferenciaron principalmente en el tamaño y la proporción de desbalance: el primero contuvo 285,299 registros y la clase minoritaria fue equivalente al 0.17%, mientras que el segundo constó de 30,000 registros y su clase minoritaria representó el 22.2%. El objetivo de este análisis fue estudiar el impacto del desbalance de clases desde diferentes perspectivas.

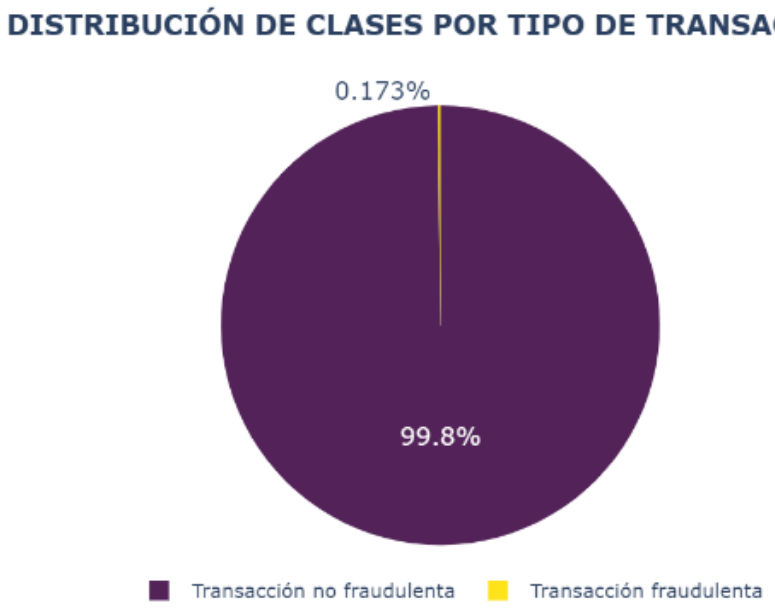
7.1.1. *Análisis exploratorio del conjunto de datos: "Credit Card Fraud Detection (BD1)"*

Este conjunto de datos incluyó transacciones de tarjetas de crédito anonimizadas realizadas por titulares europeos durante un periodo de dos días en septiembre del 2013, el cual estuvo disponible públicamente a través de la plataforma Kaggle (Kaggle, 2018). Las transacciones fueron etiquetadas como "fraudulentas" si fueron identificadas como tal, o como "no fraudulentas" si fueron legítimas. En total, se recopilaron 285,299 transacciones, de las cuales 492, es decir el 0.17%, fueron identificadas como fraudulentas, lo que reflejó un desequilibrio extremo dado que la proporción de la clase minoritaria fue menor al 1% (Malek et al., 2023). La distribución de clases se ilustró en la figura 8.

El conjunto de datos incluyó 28 características transformadas mediante Análisis de Componentes Principales (PCA), lo que impidió identificar las variables originales o conocer información sobre los datos debido a motivos de confidencialidad. Sin embargo, se dispuso de dos variables predictoras cuya naturaleza fue conocida: Time, que representó los segundos transcurridos entre la transacción actual y la primera registrada en el conjunto de datos, y Amount, que indicó el monto de la transacción.

Figura 8

Distribución de clases en conjunto de datos de detección de fraudes por tarjeta de crédito



En total, el conjunto de datos dispuso de 30 variables predictoras, mientras que la variable de respuesta, Class, tomó el valor de 1 para transacciones fraudulentas y 0 para las legítimas.

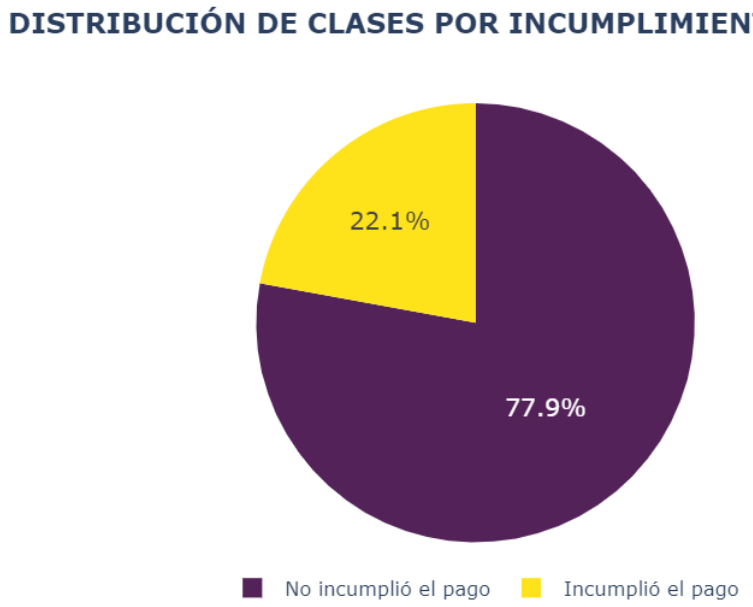
7.1.2. *Análisis exploratorio del conjunto de datos: "Default of Credit Card Clients (BD2)"*

Este conjunto de datos abordó la problemática de impagos de clientes de tarjeta de crédito en Taiwán, disponible en el repositorio UC Irvine Machine Learning Repository (Yeh, 2009). La variable respuesta, default o *impago*, es binaria: tomó el valor de 1 si el cliente no pagó su tarjeta de crédito y 0 si realizó el pago. El conjunto de datos contuvo un total de 30,000 registros, de los

cuales el 77.88% corresponden a pagos realizados y el 22.12% a casos de impago. Esta proporción indicó un desbalance leve, ya que la clase minoritaria representó entre el 20% y el 40% de los datos (Malek et al., 2023). Esta distribución de clases se ilustró en la figura 9.

Figura 9

Distribución de clases en conjunto de datos de impago de clientes de tarjetas de crédito



Además, el conjunto incluye 23 variables explicativas que describen características demográficas, financieras y comportamentales de los clientes, las cuales se describen en la tabla 6.

Tabla 6

Descripción de las variables del conjunto de datos de impago de clientes de tarjetas de crédito

Variable	Codificación original	Descripción
X1	LIMIT_BAL	Monto del crédito otorgado
X2	SEX	Género
X3	EDUCATION	Educación
X4	MARRIAGE	Estado civil
X5	AGE	Edad

X6-X11	PAY_0 à PAY_6	Historial de pagos mensuales de abril a septiembre del 2005. La escala mide desde -1 (pagado puntualmente) hasta 9 (retraso de nueve meses o más).
X12-X17	BILL_AMT1 à BILL_AMT6	Monto del estado de cuenta mensual en dólares taiwaneses de abril a septiembre del 2005, desde X12 (septiembre) hasta X17 (abril).
X18-X23	PAY_AMT1 à PAY_AMT6	Monto pagado mensualmente en dólares taiwaneses de abril a septiembre de 2005, desde X18 (septiembre) hasta X23 (abril).

7.2. Limpieza y transformación de los datos

En esta fase, se limpiaron y transformaron los datos para garantizar su posterior procesamiento.

7.2.1. *Conjunto de datos: "Credit Card Fraud Detection (BD1)"*

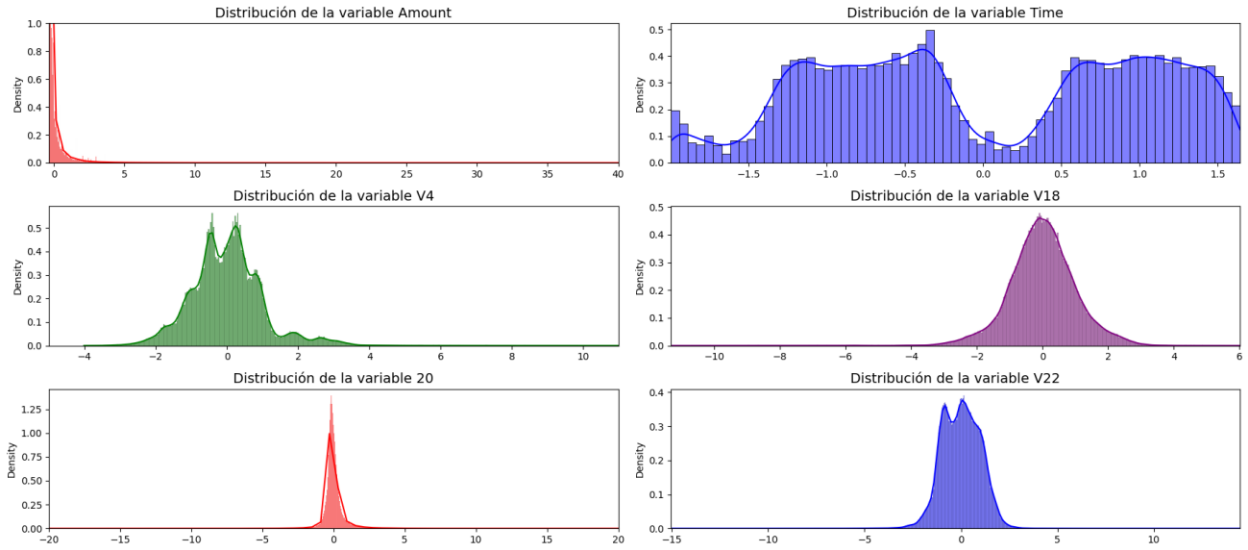
El conjunto de datos empleado no contiene valores nulos. Las características numéricas se presentan en formato decimal, mientras que la variable objetivo es cualitativa, codificada con números enteros 1 o 0. Este conjunto ya había sido sometido previamente a procesos de limpieza, lo que aseguró la ausencia de valores nulos y la coherencia en las características, eliminando la necesidad de ajustes adicionales en esta etapa.

Las variables numéricas habían sido transformadas previamente mediante Análisis de Componentes Principales (PCA). Esta técnica genera combinaciones lineales de las variables originales para maximizar la varianza y reducir la dimensionalidad, reteniendo la mayor cantidad de información posible (Hastie et al., 2009). No obstante, las variables *Time* y *Amount* permanecieron en su formato original. La figura 10 ilustra la diferencia entre las escalas: mientras que las variables transformadas presentan valores en un rango cercano a 0, *Amount* supera cantidades de 25,000 (€) y *time* sobrepasa los 160,000 (s).

La diferencia en la varianza y la distribución también indica que estas variables podrían contener información valiosa sobre transacciones inusuales, especialmente en valores extremos.

Figura 10

Distribución de las variables en conjunto de datos de detección de fraudes por tarjeta de crédito



De esta manera, para abordar esta disparidad en la escala de las variables, se estandarizó todo el conjunto de datos utilizando la función *StandardScaler ()*, que ajusta cada característica para que tenga una media de 0 y una desviación estándar de 1 (Shalev-Shwartz & Ben-David, 2014b). Esta técnica de normalización o estandarización busca igualar las escalas de las características y evitar que las unidades de medida influyan en el modelo (Han et al., 2012).

7.2.2. ***Conjunto de datos: "Default of Credit Card Clients (BD2)"***

En el conjunto de datos "Default of Credit Card Clients" fue analizado para asegurar su calidad y preparación para el modelado. Se verificó que no hay valores faltantes en las 24 columnas analizadas y la variable respuesta está correctamente definidas como entero (int64). Las variables categóricas, como X2, X3 y X4, están codificadas numéricamente, lo que facilita su uso en algoritmos de aprendizaje automático.

Algunas variables, como los saldos de facturación mensual (X12 a X17) y los montos pagados (X18 a X23), presentan rangos de valores amplios, desde cero hasta decena de miles. Esta disparidad podría influir en el rendimiento de los modelos predictivos y para evitar este efecto, se

aplicó a todo el conjunto de datos la técnica de normalización *StandardScaler ()*, que ajusta los valores para que tengan una media de 0 y una desviación estándar de 1, tal como se utilizó en el primer conjunto de datos.

7.3. Enfoque a nivel de datos: técnicas de re-muestreo

El re-muestreo de datos desbalanceados constituye una estrategia fundamental para abordar problemas de desbalance en conjuntos de datos, ya que permite ajustar la proporción entre clases sin depender exclusivamente del clasificador utilizado (Salunkhe & Mali, 2016). Este enfoque genera nuevos conjuntos de datos con una distribución de clases diferente al original (Hasanin et al., 2019b), proporcionando una base más equilibrada para el desarrollo de modelos predictivos y abordando los retos asociados al desequilibrio de clases.

El re-muestreo, junto con el preprocesamiento adecuado de los datos, resulta esencial para la construcción de modelos robustos mediante algoritmos modernos de minería de datos, al reducir errores de clasificación originados por el desbalance. En la literatura, se han identificado múltiples técnicas de re-muestreo que van desde métodos básicos, como el submuestreo y el sobremuestreo aleatorio, hasta enfoques avanzados diseñados para superar las limitaciones de estas técnicas iniciales (Khairy et al., 2024). Tal como de describe en la sección 4.2, ejemplos de estos enfoques avanzados incluyen el Sobremuestreo Sintético de la Minoría (Synthetic Minority Oversampling, SMOTE), el Muestreo Sintético Adaptativo (Adaptative Synthetic Sampling, ADASYN) y enfoques híbridos como SMOTETomek que combina el sobremuestreo mediante SMOTE con técnicas de limpieza como los enlaces Tomek.

En este proyecto, se seleccionaron tres técnicas representativas para abordar el desbalance de clases en los conjuntos de datos: el Submuestreo Aleatorio (Random UnderSampling, RUS), el Sobremuestreo Aleatorio (Random OverSampling, ROS) y el Sobremuestreo Sintético de la

Minoría (Synthetic Minority Oversampling Technique, SMOTE). Estas técnicas fueron elegidas debido a que fueron las más empleadas en la literatura analizada y su relevancia en conjuntos de datos desbalanceados.

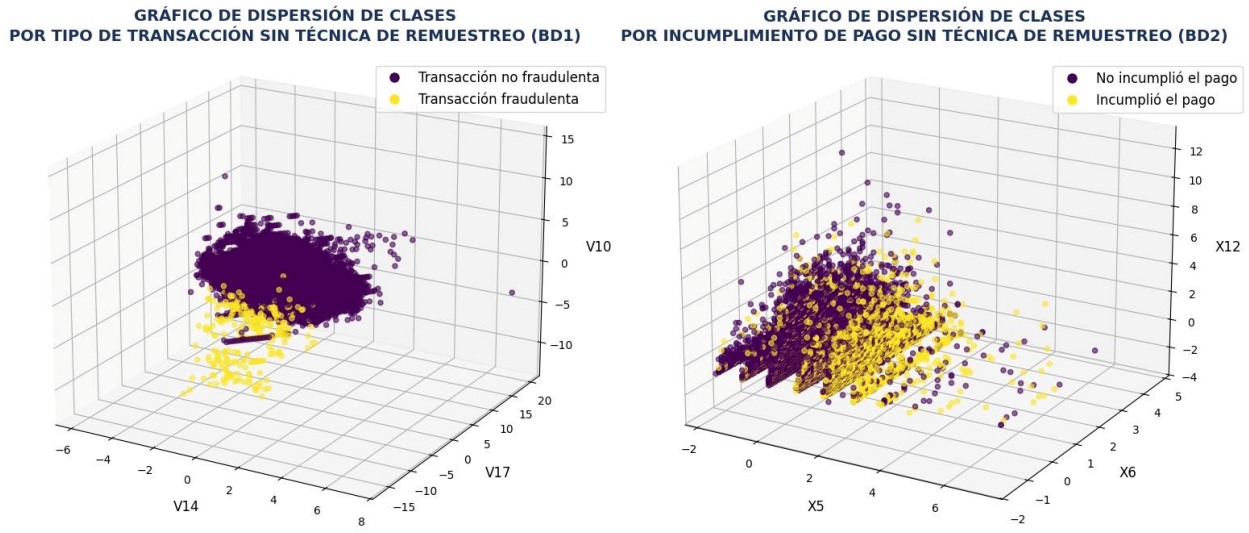
A continuación, se detalla su implementación y se analizan sus efectos sobre los conjuntos de datos “Credit Card Fraud Detection” y “Default of Credit Card Clients”, que de ahora en adelante se denotarán como BD1 y BD2 respectivamente.

7.3.1. *Distribución inicial de las clases*

Antes de aplicar técnicas de re-muestreo, se identificaron patrones significativos de desequilibrio en los conjuntos de datos analizados. En la BD1, el desbalance extremo de la clase minoritaria es evidente, con solo 492 observaciones frente a 284315 de la clase mayoritaria. Además, se observó una distribución dispersa y menos definida de la clase minoritaria, mientras que la clase mayoritaria muestra una mayor densidad y homogeneidad, como se presenta en la figura 11.

Figura 11

Distribución de clases sin aplicar técnicas de re-muestreo para la BD1 y BD2



En contraste, el desbalance en la BD2 es menos pronunciado, con 6636 observaciones en la clase minoritaria frente a 23364 en la clase mayoritaria. Sin embargo, el solapamiento entre las clases es evidente, lo que puede dificultar la capacidad de los modelos para distinguir entre ellas de manera efectiva. Este fenómeno, aunque menos crítico que el desequilibrio extremo en la BD1, plantea un reto adicional para las técnicas de re-muestreo y la capacidad predictiva de los modelos. La tabla 7 detalla la cantidad inicial de observaciones por clase para los dos conjuntos de datos que se abordarán en este proyecto.

Tabla 7

Cantidad inicial de observaciones por clase

	CLASE MINORITARIA	**CLASE MAYORITARIA**
BD1	492	284315
BD2	6636	23364

7.3.2. ***Random UnderSampling (RUS)***

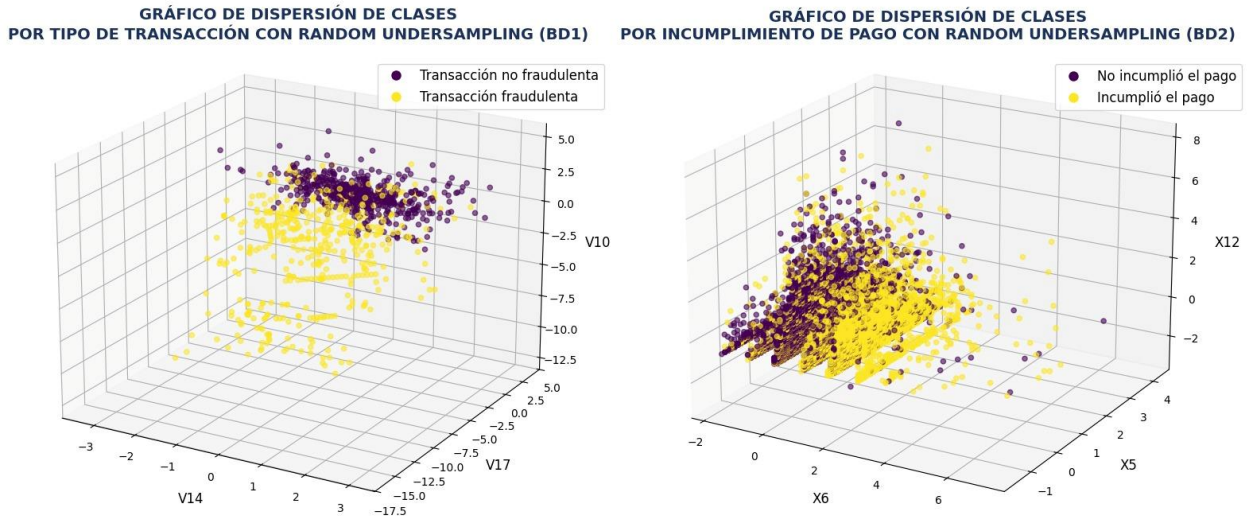
La técnica de submuestreo aleatorio elimina aleatoriamente observaciones de la clase mayoritaria para igualar su cantidad con la clase minoritaria. Implementada mediante la función *RandomUnderSampler* de la librería *imblearn.under_sampling* de *scikit-learn*, esta técnica equilibra las proporciones entre clases. Para garantizar la reproducibilidad, se utilizó una semilla fija de valor de 42.

En la BD1, las transacciones no fraudulentas, originalmente 284315, se redujeron a 492, igualando el número de observaciones de la clase minoritaria. En la BD2, la clase de clientes que no incumplieron el pago pasó de 23364 a 6636 observaciones. Como se observa en la figura 12, RUS logra un equilibrio efectivo en ambas bases de datos. Sin embargo, la eliminación de

observaciones introduce el riesgo de pérdida de información valiosa, especialmente en la BD1, donde el número original de instancias mayoritarias era significativamente mayor.

Figura 12

Distribución de clases aplicando técnica de re-muestreo RUS para la BD1 y BD2



En términos de estructura de los datos, el reequilibrio de RUS en la BD1 resalta una mayor proximidad entre las clases, aunque se observa una posible pérdida de variabilidad en la clase mayoritaria. En la BD2, si bien se reduce el solapamiento, la complejidad de las interacciones entre las clases aún persiste, sugiriendo que esta técnica puede no ser suficiente para resolver por completo el problema de desbalance en esta base de datos.

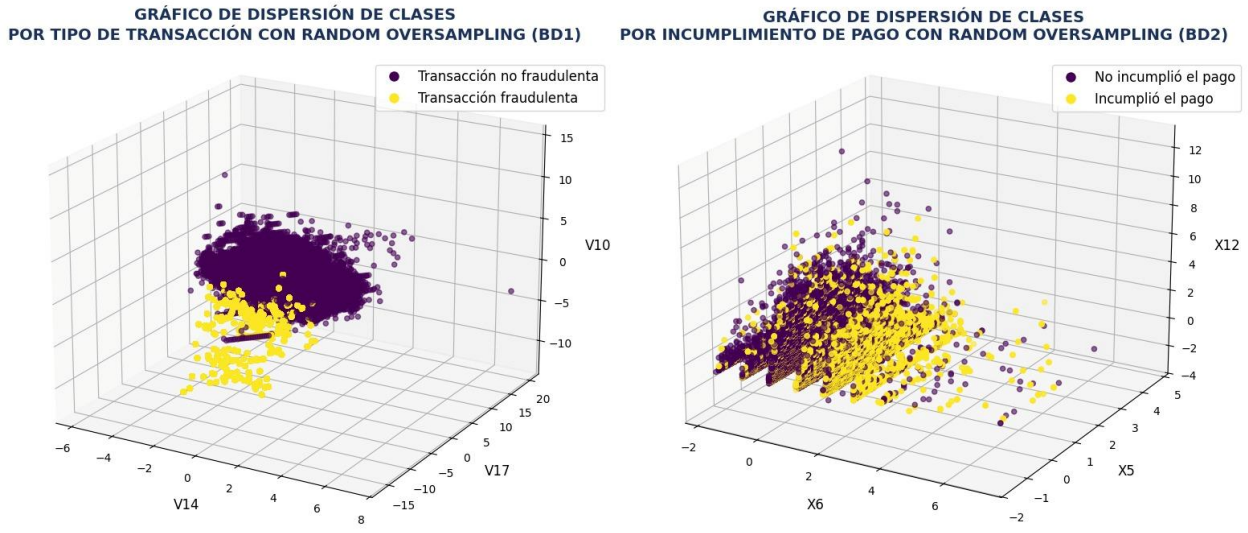
7.3.3. ***Random OverSampling (ROS)***

El sobremuestreo aleatorio equilibra las clases añadiendo copias aleatorias de observaciones de la clase minoritaria. Implementado con la función *RandomOverSampler* de la librería *imblearn.over_sampling*, esta técnica incrementó significativamente el tamaño de la clase minoritaria hasta igualar su cantidad con la clase mayoritaria. En la BD1, las instancias de la clase minoritaria aumentaron de 492 a 284315 observaciones, mientras que en la BD2 crecieron de 6636

a 23364 observaciones. La figura 13 ilustra como ROS equilibra las clases en ambos conjuntos de datos.

Figura 13

Distribución de clases aplicando técnica de re-muestreo ROS para la BD1 y BD2



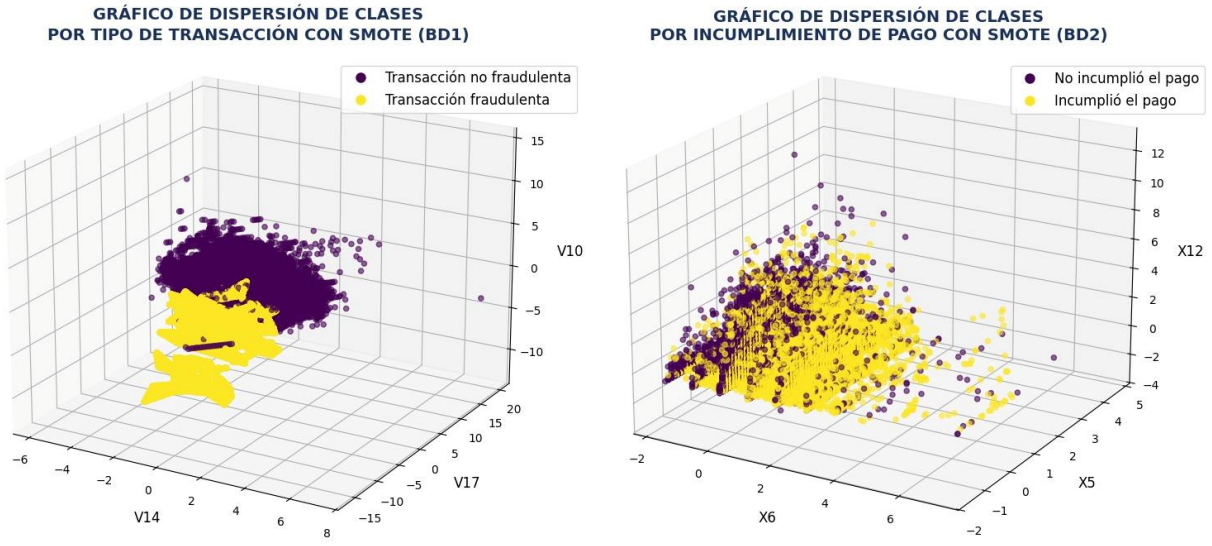
Aunque ROS mejora la proporción de clases, en la BD1, la replicación conduce a una sobre representación de los patrones existentes en la clase minoritaria, lo que podría aumentar el riesgo de sobreajuste. Por otro lado, en la BD2, ROS incrementa la densidad de la clase minoritaria lo cual ayudó un poco a resolver el solapamiento con la clase mayoritaria.

7.3.4. *Synthetic Minority Oversampling Technique (SMOTE)*

SMOTE utilizó interpolaciones entre observaciones existentes de la clase minoritaria para generar nuevas instancias sintéticas. Implementado mediante la función SMOTE de la librería *imblearn.over_sampling*, este enfoque creó variabilidad en la clase minoritaria mientras mantuvo un equilibrio entre clases. En la BD1, el tamaño de la clase minoritaria aumentó de 292 a 284315 observaciones, mientras que en la BD2 creció de 6636 a 23364 observaciones. La figura 14 muestra los resultados de esta técnica.

Figura 14

Distribución de clases aplicando técnica de re-muestreo SMOTE para la BD1 y BD2



A diferencia de ROS, SMOTE mantuvo patrones de dispersión más naturales y se evita la replicación directa de observaciones. En la BD1, SMOTE introdujo una distribución más variada en la clase minoritaria, mientras que en la BD2 generó una separación más clara entre clases, aunque el solapamiento persiste parcialmente.

En resumen, cada técnica de re-muestreo aporta un enfoque único para abordar el desbalance de clases. Sin embargo, la selección de la técnica adecuada dependerá del conjunto de datos, como se logró apreciar en el análisis realizado previamente, y del objetivo del modelo predictivo, siendo crucial evaluar estas estrategias en función del rendimiento obtenido en los modelos ensamblados descritos en la siguiente sección.

Fase 3: modelamiento

El modelamiento se centró en la construcción y optimización de los modelos Random Forest y XGBoost para abordar problemas de clasificación en escenarios con clases desbalanceadas. La selección y ajuste de hiperparámetros se realizó mediante estrategias sistemáticas para garantizar un rendimiento óptimo en términos de métricas clave y capacidad de generalización.

8.1. Modelos ensamblados

La implementación de métodos de aprendizaje ensamblados, específicamente Random Forest y XGBoost, se justifica en literatura, como se describe en la sección 4.2. Estos modelos han demostrado ser útiles en escenarios con desequilibrio de clases debido a su capacidad de combinar múltiples clasificadores débiles, como los árboles de decisión, y construir un modelo conjunto más robusto y con mejor capacidad predictiva.

El modelo Random Forest, representativo del enfoque Bagging, se distingue como una herramienta robusta para la predicción al combinar árboles de decisión descorrelacionados y suficientemente fuertes. Este balance entre baja correlación e independencia de los árboles individuales, junto con su capacidad para captar patrones relevantes en los datos, permite reducir la varianza y mejorar la generalización del modelo, haciéndolo eficaz y resistente al sobreajuste (Breiman, 2001). Esta característica lo hace adecuado para escenarios donde los datos presentan una alta complejidad o un desequilibrio significativo entre clases. En su implementación, se utilizaron funciones de la librería Scikit-learn en Python, incluyendo *train_test_split* para dividir los conjuntos de datos, *cross_val_score* para evaluar el rendimiento mediante validación cruzada, *GridSearchCV* para realizar una búsqueda sistemática de hiperparámetros y

RandomForestClassifier del módulo *sklearn.ensemble* para construir el modelo, garantizando una configuración óptima. Además, para las técnicas a nivel de datos, se emplearon módulos específicos de imblearn, tales como *RandomOverSampler* para ROS y SMOTE, y *RandomUnderSampler* para RUS, asegurando un tratamiento adecuado del desbalance en las clases.

Por otra parte, XGBoost, que representa al enfoque Boosting, sobresale como un método avanzado, eficiente y escalable, diseñado para abordar problemas complejos y manejar grandes volúmenes de datos. Este modelo incorpora técnicas de regularización para evitar el sobreajuste y permite una amplia flexibilidad en la configuración de hiperparámetros (Xia et al., 2017). Además, su capacidad para ajustar la importancia de las clases desbalanceadas mediante el parámetro "scale_pos_weight" lo hace ideal para abordar problemas de clasificación en los que la clase minoritaria es difícil de detectar. Este modelo utilizó las mismas herramientas de la librería de Scikit-learn en Python empleadas para Random Forest. Sin embargo, para construir el modelo se utilizó la librería *xgboost* en Python, importada como xgb, que permite la construcción y optimización de modelos de Gradient Boosting de manera eficiente y escalable. Esta implementación facilita el ajuste de hiperparámetros avanzados como "*learning_rate*", "*max_depth*" y "*scale_pos_weight*", fundamentales para abordar problemas con datos desbalanceados y garantizar un alto rendimiento en tareas de clasificación complejas.

Ambos modelos representan enfoques complementarios para abordar el problema del desbalance de clases en este estudio, por lo que la selección adecuada de sus hiperparámetros resulta fundamental para garantizar un rendimiento óptimo, alcanzado a través de un proceso de optimización sistemática.

8.1.1. ***Optimización de hiperparámetros***

La optimización de los hiperparámetros, también conocida como el ajuste fino del modelo, es un paso fundamental en la construcción de modelos de aprendizaje automático. A pesar de su importancia, los modelos ensamblados han recibido relativamente poca atención en lo que respecta al ajuste de estos parámetros, a pesar de que son esenciales para adaptar cada modelo base a las características específicas del problema y garantizar una mayor capacidad de predicción y generalización (Yotsawat et al., 2021).

Existen diferentes formas para realizar esta optimización, entre los que se destacan la búsqueda en cuadrícula (*grid Search)* y la búsqueda aleatoria (*randomized search*), entre otros. En este proyecto se optó por la búsqueda en cuadrícula como estrategia de ajuste fino para los modelos Random Forest y XGBoost, debido a sus ventajas. Este método considera exhaustivamente todas las combinaciones posibles de hiperparámetros dentro de un espacio definido, lo que garantiza una solución óptima. Además, emplea un enfoque simple y directo, lo que facilita su implementación (Ogunsanya et al., 2023).

Este proceso de optimización se realizó utilizando la librería *scikit-learn* de Python, con el módulo *GridSearchCV* (scikit-learn, 2025). Asimismo, se aplicó validación cruzada con 3 particiones para evaluar todas las combinaciones posibles, aumentando la confiabilidad de los resultados. Sin embargo, este enfoque también implicó un mayor costo computacional debido al número elevado de iteraciones necesarias para completar la evaluación.

La selección de los rangos de hiperparámetros evaluados se basó en recomendaciones de literatura y en buenas prácticas para la configuración de cada modelo. A continuación, en la tabla 8, se detalla la configuración evaluada para cada modelo, junto con los rangos de valores considerados:

Tabla 8

Configuración y rango de hiperparámetros ajustados para los modelos RF y XGBoost

MODELO	CONFIGURACIÓN DE PARÁMETROS	RANGO	CV	AJUSTES
RF	n_estimators	{10, 12, 14, 16, 18, 20, 50, 100, 200}	3	2916
	criterion	{"gini", "entropy"}		
	max_features	{"sqrt", "log2"}		
	max_depth	{10, 20, 30}		
	min_samples_split	{2, 5, 10}		
	min_samples_leaf	{1, 2, 4}		
XGBoost	n_estimators	{10,20,50,100,200,500}	3	19440
	max_depth	{3,4,7,9}		
	learning_rate	{0.02,0.15,0.2,0.4,0.5}		
	gamma	{0,0.25,0.5}		
	lambda	{0,0.25,0.5}		
	alpha	{0,0.25,0.5}		
	scale_pos_weight	{1, sum-wneg/sum-wpos}		
	colsample_bytree	{0.2,1}		

Para los modelos Random Forest y XGBoost, fue fundamental configurar el hiperparámetro "*n_estimators*", ya que con este se determinó el número de árboles con el cual el modelo se entrenó. Según las recomendaciones de literatura (Cao et al., 2021), el ajuste para el número de árboles se estableció en un rango de 10 a 100 para mantener un equilibrio entre configuraciones rápidas y efectivas. Además, se incluyeron valores más altos, como 200 para RF y 500 para XGBoost, con el fin de capturar patrones más complejos en los datos desbalanceados.

Para el modelo Random Forest, la elección del número de características seleccionadas por nodo "*max_features*" dependerán del problema que se trate (Hastie et al., 2009), así que se evaluaron las opciones disponibles "sqrt" y "log2", siguiendo las recomendaciones de (Breiman, 2001) para promover la diversidad entre los árboles y evitar el sobreajuste. Para el parámetro "*criterion*", se seleccionaron las opciones "gini" y "entropy". El índice de Gini se destaca por su

eficiencia computacional, mientras que la entropía, basada en la teoría de la información, puede ofrecer divisiones más informativas en problemas con clases desbalanceadas (Breiman, 2001; Hastie et al., 2009). La profundidad máxima de los árboles "*max_depth*" fue limitada a valores de 10, 20 y 30 para controlar la complejidad del modelo desde diferentes profundidades. Adicionalmente, los parámetros "*min_samples_split*" y "*min_samples_leaf*" se configuraron en valores bajos (2-10 y 1-4, respectivamente) para capturar relaciones complejas en los datos, pero evitando nodos con muy pocas muestras.

Para el modelo XGBoost, algunos parámetros clave como "*max_depth*", "*learning rate*" y "*colsample_bytree*" se optimizaron siguiendo recomendaciones y estrategias previas documentadas (Yotsawat et al., 2021). Por ejemplo, el parámetro "*max_depth*", que controla la profundidad máxima de los árboles, fue ajustado en un rango de 4 a 9, permitiendo capturar relaciones complejas sin ajustarse demasiado a los datos de la mano con garantizar eficiencia computacional. El "*learning rate*" o la tasa de aprendizaje**,** utilizado para determinar el impacto de cada iteración en la actualización del modelo, se configuró entre 0.01 y 1 en estudios previos. En nuestro proyecto, se agregó un ajuste adicional con valores de hasta 0.5, permitiendo explorar configuraciones que equilibran la convergencia estable con una velocidad de entrenamiento más rápida. Por otra parte, el parámetro "*colsample_bytree*"**,** que regula la proporción de características utilizadas en cada árbol, se configuró inicialmente entre 0.8 y 1, como se menciona en la literatura. Sin embargo, en nuestro proyecto se optó por incluir un valor más bajo, como 0.2, lo que fomenta la diversidad entre los árboles y permite analizar el impacto de usar subconjuntos más pequeños de características. Este ajuste también se asemeja a las configuraciones aplicadas en el modelo Random Forest, donde la aleatoriedad contribuye significativamente al rendimiento y la capacidad de generalización.

Adicionalmente, se consideraron otros parámetros importantes para optimizar el modelo, como el parámetro “gamma”, que establece el umbral mínimo de ganancia requerido para realizar una división en un nodo. Valores más altos de gamma reducen la complejidad del modelo al evitar divisiones insignificantes. En la configuración, se probó un rango de {0, 0.25, 0.5}, permitiendo evaluar desde divisiones sin restricciones hasta divisiones más conservadoras, optimizando la simplicidad y el rendimiento del modelo. También, los parámetros “*lambda*” y “*alpha*”, que controlan la regularización L2 y L1, respectivamente, se utilizaron para reducir el sobreajuste. La regularización L1 “*alpha*” fomenta la selección de características relevantes al penalizar ciertos pesos hasta llevarlos a cero, mientras que la regularización L2 “*lambda*” estabiliza los pesos de las hojas, previniendo variaciones extremas en árboles complejos (Chen & Guestrin, 2016). Ambos parámetros se exploraron en un rango de {0, 0.25, 0.5}. Finalmente, el parámetro “*scale_pos_weight*” se configuró como 1 y “sum_wneg/sum_wpos” para compensar el desbalance de clases y mejorar la sensibilidad del modelo hacia las clases minoritarias. Inicialmente, se evaluó el modelo sin este ajuste utilizando un valor de 1. Posteriormente, se estableció el valor basado en la proporción entre las clases (sum_wneg/sum_wpos), como lo sugiere la documentación oficial de XGBoost (xgboost, 2025).

Estas configuraciones serán evaluadas en términos de métricas clave y su impacto en la capacidad de generalización del modelo, considerando las características específicas de cada conjunto de datos.

Fase 4: evaluación e interpretación

En esta fase, se evaluaron los modelos considerando diferentes escenarios de división de los datos y métricas clave como *Recall*, *Precision* y el área bajo la curva Precision-Recall (PR),

esenciales en problemas con datos desbalanceados. Se implementó validación cruzada y se analizaron curvas de aprendizaje para garantizar una evaluación robusta y una adecuada capacidad de generalización de los modelos. Además, se exploraron técnicas de re-muestreo como RUS, ROS y SMOTE, priorizando un balance óptimo entre sensibilidad, precisión y estabilidad del modelo.

9.1. Métricas de rendimiento y evaluación de los modelos

La evaluación de las combinaciones de hiperparámetros se realizó considerando tres escenarios distintos de división de los conjuntos de datos:

1. 75% de los datos para entrenamiento y el 25% para prueba.
2. 80% de los datos para entrenamiento y el 20% para prueba.
3. 90% de los datos para entrenamiento y el 10% para prueba.

En cada escenario, se implementó una validación cruzada con 3 particiones (cv=3), lo que permitió seleccionar el modelo que alcanzara el mejor rendimiento en términos de *Recall*, una métrica prioritaria para el análisis de datos desbalanceados. El *Recall*, también conocido como sensibilidad o tasa de verdaderos positivos, mide la capacidad del modelo para identificar correctamente los casos positivos dentro de la clase minoritaria, lo que resulta esencial en problemas donde las consecuencias de los falsos negativos son críticas. Su fórmula se presenta en la ecuación (2):

$$recall = \frac{TP}{TP + FN} = sensibilidad = TP_{rate} \quad (2)$$

El rango de valores de *Recall* es de 0 a 1(Sun et al., 2018), donde el valor más alto indica una mayor capacidad del modelo para identificar correctamente los casos positivos.

Una vez identificado el modelo con el mejor Recall, se analizaron sus curvas de aprendizaje utilizando una validación cruzada más robusta con 10 particiones (cv=10). Este análisis permitió

evaluar la capacidad de generalización del modelo en datos no vistos y comprender su comportamiento en términos de sesgo y varianza.

Adicionalmente, otras métricas seleccionadas en este estudio fueron:

- ***Precision***: Evalúa el porcentaje de ejemplos correctamente clasificados como pertenecientes a la clase minoritaria entre todos los ejemplos que el modelo predijo como clase minoritaria. Su fórmula se presenta en la ecuación (3) (Sun et al., 2018):

$$precision = \frac{TP}{TP + FP} \quad (3)$$

  Al igual que el *recall*, la *precision* varía entre 0 y 1, siendo una métrica clave para evaluar qué tan precisas son las predicciones positivas.

- ***Precision-Recall (PR)***: Representa el área bajo la curva de precisión-recall, integrando ambas métricas para proporcionar una evaluación más completa del balance entre sensibilidad y precisión (Davis & Goadrich, 2006).

El uso de estas métricas respondió a la necesidad de obtener una evaluación integral del rendimiento de los modelos, especialmente en problemas donde la clase minoritaria tiene un alto impacto práctico. Según estudios previos, emplear múltiples métricas permite abordar las limitaciones individuales de cada una (Malek et al., 2023). Aunque el *Recall* y la *Precision* son efectivas para medir el rendimiento, en muchos casos se observa que un aumento en el *Recall* puede venir acompañado de una disminución en la *Precision*, incrementando así los falsos positivos (Mondal et al., 2021). Para mitigar estas limitaciones, se utilizó la matriz de confusión como herramienta complementaria para analizar las tasas de falsos positivos y falsos negativos.

Para equilibrar estas métricas, ampliamente en la literatura se emplean estrategias como AUC (Artetxe et al., 2020; Choudhary & Shukla, 2021; Honnurappa & Raghavendra, 2021; Nanni et al., 2015; Salunkhe & Mali, 2016; Shah et al., 2022). Sin embargo, cuando se trata de conjuntos

de datos desbalanceados, las curvas Precision-Recall (PR) ofrecen una representación más informativa del rendimiento del modelo. En el análisis elaborado por (Davis & Goadrich, 2006) estableció que una curva que domina en el espacio ROC también lo hará en el espacio PR, justificando su uso en problemas con clases desbalanceadas. Por lo tanto, el análisis de la curva PR se priorizó para evaluar modelos en escenarios desbalanceados, proporcionando una representación más precisa y robusta del rendimiento del algoritmo.

Este proceso se aplicó bajo cuatro enfoques distintos a nivel de datos: técnicas sin re-muestreo, RUS, ROS y SMOTE, utilizando los tres escenarios de entrenamiento mencionados. La selección de modelos no solo se basó en su capacidad para maximizar el Recall, sino también en su capacidad de generalización, evaluada mediante las curvas de aprendizaje, el ajuste óptimo de hiperparámetros mediante búsqueda en cuadrícula y las métricas de rendimiento seleccionadas.

9.2. Resultados de los modelos y técnicas de re-muestreo

En esta sección se presenta y analiza los resultados obtenidos al aplicar diversas estrategias de modelamiento a dos conjuntos de datos con clases desbalanceadas. Los resultados se organizaron en función de las estrategias empleadas: sin re-muestreo, submuestreo (RUS) y sobremuestreo (ROS y SMOTE). Se evaluó métricas clave como *Recall*, *Precision* y *AUPRC* para los modelos Random Forest y XGBoost, considerando diferentes tamaños de conjuntos de entrenamiento (75%, 80% y 90%) y las configuraciones óptimas de hiperparámetros que maximizaban el rendimiento en la detección de la clase minoritaria. Todas las configuraciones obtenidas se encuentran en el Apéndice B para el modelo Random Forest y en el Apéndice C se encuentran los resultados para el modelo XGBoost. Además, se implementaron y analizaron curvas de aprendizaje para evaluar la capacidad de generalización de los modelos, lo que permitió identificar su comportamiento en términos de sesgo y varianza. Finalmente, se incluye

una discusión sobre las fortalezas, limitaciones y comportamiento de los modelos, proporcionando un análisis integral del rendimiento obtenido en distintos escenarios.

9.2.1. *Estrategia sin re-muestreo*

En este apartado, se analizó el rendimiento de los modelos Random Forest y XGBoost en dos conjuntos de datos con diferentes grados de desbalance, sin aplicar técnicas de re-muestreo. Esta estrategia permitió establecer una línea base para evaluar las limitaciones de los modelos en escenarios desbalanceados.

9.2.1.1. **Conjunto de datos "Credit Card Fraud Detection".**

Este conjunto de datos presentó un desbalance extremo, donde solo el 0.17% de las observaciones correspondieron a transacciones fraudulentas. La presencia de un desbalance tan marcado en las clases supuso un reto importante al tratar de identificar la clase minoritaria, especialmente porque los fraudes tienen un impacto crítico en el ámbito financiero; ya que, detectar de manera precisa estas anomalías resulta esencial para reducir pérdidas económicas y preservar la confianza de los clientes en las instituciones (Jason Brownlee, 2020).

9.2.1.1.1. ***Resultados con Random Forest.***

Al evaluar el modelo Random Forest utilizando los tamaños de entrenamiento. En todos los escenarios, el *Recall* mostró valores estables entre 0.7792 y 0.7825, tal como se muestra en la tabla 9, lo que evidencia una limitada capacidad del modelo para detectar instancias de fraude en un contexto de desbalance extremo. Aunque el modelo logró identificar una proporción considerable de fraudes, aproximadamente el 22% de los casos quedaron sin detectar, lo cual puede tener consecuencias significativas en aplicaciones prácticas.

Por otro lado, la *Precisión* alcanzó valores altos, entre 0.9209 y 0.9512, lo que indica que la mayoría de las predicciones de fraude realizadas por el modelo son correctas. Esto es valioso

para evitar falsos positivos que podrían interrumpir transacciones legítimas. Sin embargo, al analizar conjuntamente *Recall* y *Precisión*, se observó que el modelo priorizó evitar falsos positivos sobre la detección de fraudes, dejando una cantidad significativa de casos de fraude sin identificar. Este comportamiento refleja un sesgo hacia la clase mayoritaria, lo que limita su efectividad en problemas de desequilibrio extremo.

Tabla 9

Métricas sin técnicas de re-muestreo para la BD1 con el modelo Random Forest

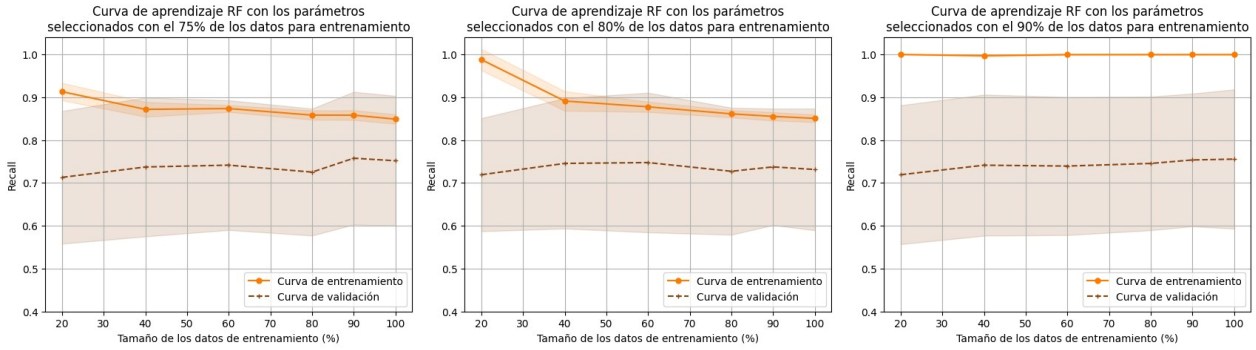
% Train size	Parámetros RF	Recall	Precision	AUPRC
75%	*n_estimators*=14, *criterion*=gini, *max_features*=sqrt, *max_depth*=20, *min_samples_split*=2, *min_samples_leaf*=2	0,7811	0,9209	0,8210
80%	*n_estimators=50*, *criterion=entropy*, *max_features*= sqrt, *max_depth=1*0, *min_samples_split*=2, *min_samples_leaf*=1	0,7792	**0,9512**	**0,8440**
90%	*n_estimators*=100, *criterion*=gini, *max_features=sqrt*, *max_depth=3*0, *min_samples_split*=2, *min_samples_leaf*=1	**0,7825**	0,9420	0,8403

La métrica AUPRC (Área bajo la curva de precisión-recall), que equilibra *Precisión* y *Recall* considerando diferentes umbrales, presentó valores entre 0.8210 y 0.8440, dependiendo del tamaño del conjunto de entrenamiento. Este rango reflejó la capacidad del modelo para mantener un balance razonable entre ambas métricas. Con el 75% de datos de entrenamiento, el AUPRC fue de 0.8210, lo que sugiere que menor cantidad de datos puede limitar el equilibrio entre *Precision* y *Recall*. En el escenario con el 80% de entrenamiento, el AUPRC aumentó a 0.8440, reflejando un balance entre las predicciones correctas y la capacidad de detección de fraudes. En el caso del 90%, aunque el AUPRC se mantuvo alto, en 0.8403, se observó un ligero descenso en comparación

con el 80%, lo que podría atribuirse al aumento de complejidad en los hiperparámetros, como el mayor número de estimadores (n_estimators=100) y la profundidad del árbol (max_depth=30), lo que puede introducir sobreajuste, tal como se aprecia en su respectiva curva de aprendizaje en la figura 15, en donde exhibe un *Recall* perfecto en entrenamiento y una brecha amplia y persistente en validación, lo que limitó su capacidad de generalización hacia la clase minoritaria.

Figura 15

Curvas de aprendizaje para el modelo Random Forest BD1 sin aplicar técnicas de re-muestreo



Adicionalmente, al observar las otras curvas de aprendizaje, se visualizó diferencias clave en la capacidad del modelo Random Forest para generalizar en este contexto de desequilibrio. La configuración del 75% presentó un rendimiento sólido, pero con fluctuaciones en la curva de validación que reflejaron cierta inestabilidad. En contraste, la configuración del 80% logró el mejor equilibrio entre sesgo y varianza, destacándose por una curva de validación estable y consistente que sugiere una generalización más robusta. Aunque persisten limitaciones en la detección de la clase minoritaria, esta configuración se posicionó como la opción adecuada para escenarios desbalanceados, al mantener un balance adecuado entre rendimiento y estabilidad.

Estos resultados resaltan la importancia de no solo optimizar el tamaño de los datos de entrenamiento, sino también de complementar el modelo con el ajuste de hiperparámetros adecuado para el modelo ensamblado Random Forest. La configuración del 80% se presentó como

la opción recomendable, pero requiere estrategias adicionales para abordar plenamente el impacto del desbalance extremo.

9.2.1.1.2. ***Resultados con XGBoost.***

El modelo XGBoost fue evaluado bajo la estrategia sin re-muestreo utilizando tres configuraciones principales, considerando el ajuste del parámetro "*scale_pos_weight*" en dos valores: {sum_wneg / sum_wpos, 1}, respectivamente. Este parámetro asigna un peso mayor a la clase minoritaria durante el entrenamiento, y su valor se calcula como la proporción entre la cantidad de observaciones de la clase mayoritaria y la clase minoritaria en el conjunto de datos. Dado que se utilizaron conjuntos de entrenamiento del 75%, 80% y 90%, esta proporción varió ligeramente, obteniendo valor promedio aproximado de 577. Este ajuste permitiría al modelo compensar el desbalance extremo en los datos, enfocando más su atención en la detección de la clase minoritaria.

Tal como se observa en la tabla 10, las configuraciones sin "*scale_pos_weight"* mostraron un rendimiento equilibrado en términos de *Precisión* y *Recall*, con valores de *Recall* que oscilaron entre 0.8021 y 0.8117. Aunque esto representa una mejora leve respecto al modelo Random Forest, aún dejó entre un 18% y un 20% de los fraudes sin identificar, lo que puede ser problemático en este contexto en donde la detección de fraudes es crítica. La *Precision*, por otro lado, fue alta (0.9298 a 0.9571), indicando que la mayoría de las predicciones realizadas como fraude fueron correctas, lo que minimizó los falsos positivos. El *AUPRC* también fue competitivo, con valores entre 0.8497 y 0.8544, siendo la configuración del 90% la más equilibrada y recomendable en este grupo, logrando el mejor balance entre métricas clave y generalización, tal como se observa en las curvas de aprendizaje de la figura 16, que para esta configuración mostraron una brecha moderada

entre entrenamiento y validación, indicando una generalización adecuada con un equilibrio entre sesgo y varianza.

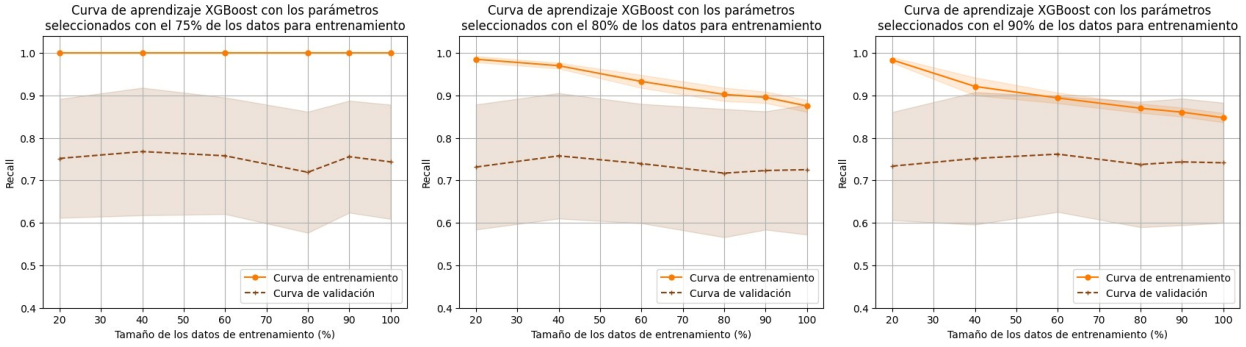
Tabla 10

Métricas sin técnicas de re-muestreo para la BD1 con el modelo XGBoost

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=200, max_depth=7, alpha=0, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0.25, learning_rate=0.4*	0,8023	**0,9571**	0,8497
80%	*n_estimators=50, max_depth=3, alpha=0, booster=gbtree, colsample_bytree=1, gamma=0.5, lambda=0.25, learning_rate=0.4*	0,8021	0,9298	0,8519
90%	*n_estimators=100, max_depth=3, alpha=0.25, booster=gbtree, colsample_bytree=1, gamma=0.25, lambda=0.25, learning_rate=0.15*	**0,8117**	0,9384	**0,8544**

Figura 16

Curvas de aprendizaje para el modelo XGBoost con BD1 sin aplicar técnicas de re-muestreo



Al incorporar el parámetro "*scale_pos_weight*", con un valor promedio aproximado de 577 (calculado como la relación entre la cantidad de observaciones de la clase mayoritaria y la minoritaria: 284807/ 492), el modelo cambió drásticamente su enfoque, priorizando mucho más la

detección de fraudes al maximizar el *Recall*. Esta configuración, como se evidencia en la Tabla 11, logró valores de *Recall* mucho más altos, oscilando entre 0.8787 y 0.8934, lo que significa que solo entre un 10% y un 12% de los fraudes quedaron sin detectar. Sin embargo, este enfoque tuvo un costo significativo en *Precisión*, que disminuyó drásticamente a un rango de 0.0387 a 0.1623, indicando un aumento considerable en los falsos positivos. El *AUPRC* también se redujo en comparación con las configuraciones originales, con valores entre 0.6131 y 0.7482.

Tabla 11

Métricas sin técnicas de re-muestreo para la BD1 con el modelo XGBoost con parámetro "scale_pos_weight"

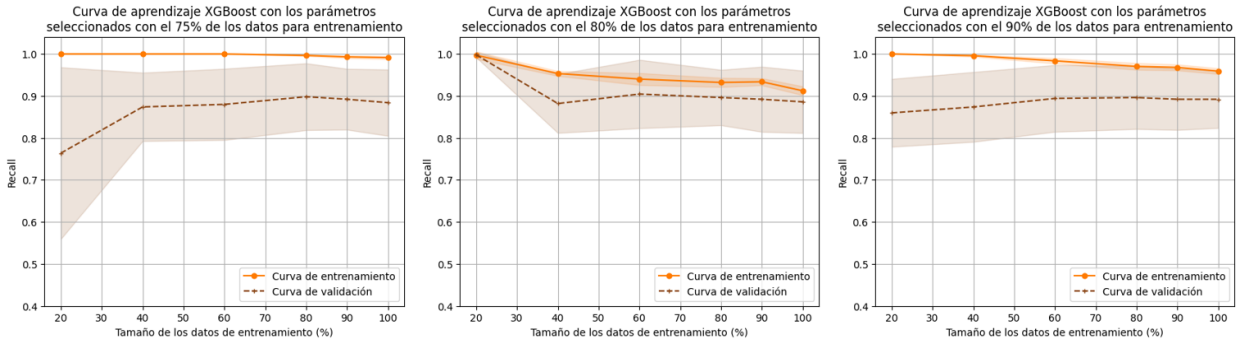
% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=20, max_depth=3, alpha=0.25, booster=gbtree, colsample_bytree=1, gamma=0.25, lambda=0.25, learning_rate=0.4, scale_pos_weight=577*	0,8787	**0,1623**	**0,7482**
80%	*n_estimators=10, max_depth=3, alpha=0, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0.25, learning_rate=0.02, scale_pos_weight=577*	**0,8934**	0,0387	0,6131
90%	*n_estimators=10, max_depth=3, alpha=0, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0, learning_rate=0.5, scale_pos_weight=577*	0,8902	0,0954	0,7231

El análisis de las curvas de aprendizaje de la figura 17 mostró que, en la configuración del 75%, las curvas iniciales presentaron una brecha considerable entre entrenamiento y validación, que se redujo con más datos. El modelo alcanzó un *Recall* de validación de hasta 0.90, aunque su *Precision* y *AUPRC* (0.1623 y 0.7482, respectivamente) reflejan un balance limitado. En la configuración del 80%, las curvas evidenciaron una mejora en la generalización con más datos,

aunque un sobreajuste inicial produjo un *Recall* perfecto (1.00) con conjuntos pequeños, estabilizándose posteriormente entre 0.89 y 0.90. Sin embargo, su *Precision* extremadamente baja (0.0387) y el *AUPRC* más bajo (0.6131) limitan su aplicabilidad. Por el contrario, la configuración del 90% mostró la brecha más reducida entre entrenamiento y validación, con un *Recall* constante entre 0.89 y 0.90; y un *AUPRC* de 0.7231, posicionándola como la opción más equilibrada para maximizar la detección de fraudes.

Figura 17

Curvas de aprendizaje para el modelo XGBoost con BD1 sin aplicar técnicas de re-muestreo con "scale_pos_weight"



En conclusión, la configuración de 90% en ambos escenarios se destacó como la más recomendable, lo que sugiere que, en este caso, un mayor tamaño del conjunto de entrenamiento permitió al modelo aprender patrones más representativos y mejorar su capacidad de generalización. Además, se evidenció diferencias notables en el enfoque y el rendimiento entre estas configuraciones. La configuración del 90% con "*scale_pos_weight*" es particularmente efectiva en escenarios donde la sensibilidad es prioritaria, alcanzando un *Recall* constante de hasta 0.90 y mostrando una generalización razonable. Sin embargo, aunque este ajuste permite al modelo compensar el desbalance extremo en los datos al enfocar más su atención en la clase minoritaria, también introduce falencias significativas, como un incremento notable en los falsos positivos, que

comprometen su precisión global. Por otro lado, en aplicaciones donde es igualmente importante minimizar los falsos positivos, la configuración del 90% sin "*scale_pos_weight*" podría ser más apropiada debido a su mejor balance general entre *Recall* y *Precisión*. Al comparar la métrica AUPRC, las configuraciones sin "*scale_pos_weight*" presentan valores más altos (0.8497 a 0.8544) en contraste con las configuraciones ajustadas con este parámetro (0.6131 a 0.7482), lo que indica que, sin este ajuste, el modelo puede lograr una mayor capacidad de generalización. Estas observaciones subrayaron la importancia de adaptar la elección del modelo XGBoost según los objetivos específicos y las limitaciones del contexto operativo.

9.2.1.2. **Conjunto de datos: "Default of Credit Card Clients".**

Este conjunto de datos presentó una distribución de clases moderada, donde el 22.12% de las observaciones correspondieron a casos de incumplimiento de pagos. Este patrón representó un desafío importante al tratar de reconocer la clase menos representada, dado que los incumplimientos son eventos de alta relevancia para las instituciones financieras. Contar con métodos efectivos para identificar a los clientes con mayor probabilidad de impago permite diseñar estrategias preventivas enfocadas en mitigar riesgos, limitar pérdidas económicas y garantizar la estabilidad financiera de la organización (SAS, 2025).

9.2.1.2.1. ***Resultados con Random Forest.***

En estas configuraciones del modelo Random Forest a diferencia de la base de datos 1 (BD1), los resultados obtenidos reflejaron un rendimiento considerablemente inferior en términos de *Recall*, *Precision* y *AUPRC* en la tabla 12. El *Recall* para BD2, mostró valores entre 0.3736 y 0.3799, lo que indica que el modelo solo identifica alrededor del 37% de los casos de fraude. Esta cifra es significativamente más baja en comparación con los resultados de BD1, lo que sugiere que las características de BD2 dificultan la detección de fraudes de manera efectiva. Este rendimiento

limitado también podría estar relacionado con las particularidades de la distribución de datos o con una mayor complejidad en las relaciones entre las variables predictoras y la clase objetivo.

En cuanto a la *Precisión*, los resultados oscilaron entre 0.6531 y 0.6596. Aunque estos valores indican que una proporción moderada de las predicciones positivas del modelo son correctas, refleja que el modelo no está capturando adecuadamente los casos de impago, pero cuando los hace, sus predicciones son razonablemente confiables. Sin embargo, este comportamiento no fue ideal para el contexto en específico, debido a que también la métrica AUPRC, que equilibra *Recall y Precisión* considerando distintos umbrales, presentó valores que van de 0.5416 a 0.5505, lo que representa un rendimiento limitado en términos de balance entre la detección y la precisión de las predicciones.

Tabla 12

Métricas sin técnicas de re-muestreo para la BD2 con el modelo Random Forest

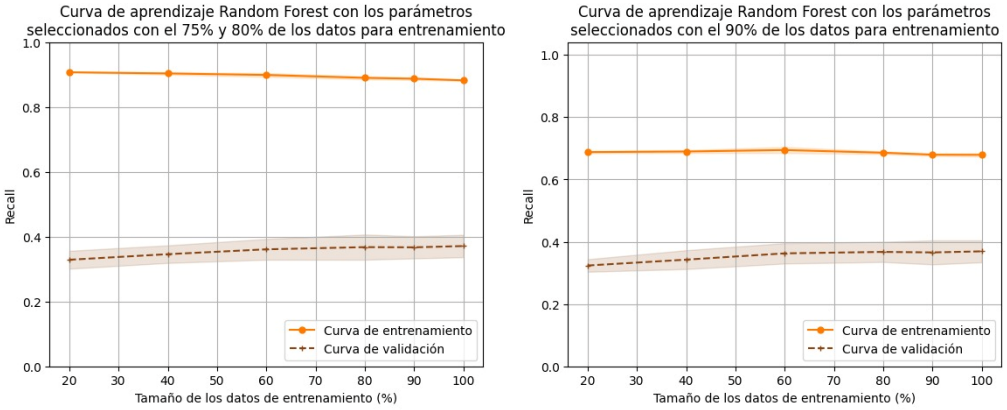
% Train size	**Parámetros RF**	**Recall**	**Precision**	**AUPRC**
75%	*n_estimators=50*, *criterion=gini*, *max_features=sqrt*, *max_depth=30*, *min_samples_split=**5***, *min_samples_leaf*=1	**0,3799**	0,6531	**0,5505**
80%	*n_estimators=50*, *criterion=gini*, *max_features=sqrt*, *max_depth=30*, *min_samples_split=**5***, *min_samples_leaf*=1	0,3770	0,6571	0,5416
90%	*n_estimators=50*, *criterion=gini*, *max_features=sqrt*, *max_depth=30*, *min_samples_split=**10***, *min_samples_leaf*=1	0,3736	**0,6596**	0,5464

El análisis de las curvas de aprendizaje de la figura 18 mostró diferencias claras en las configuraciones evaluadas. Para las configuraciones del 75% y 80%, que utilizaron los mismos hiperparámetros, se observó una tendencia consistente. El *Recall* de entrenamiento comenzó alto

y disminuyó gradualmente, mientras que el *Recall* de validación aumentó progresivamente hasta estabilizarse en 0.37. Este comportamiento indica una mejora progresiva en la generalización del modelo, aunque persisten limitaciones significativas en la detección de fraudes.

Figura 18

Curvas de aprendizaje para el modelo Random Forest BD2 sin aplicar técnicas de re-muestreo



En la configuración del 90%, el *Recall* de entrenamiento fue menor desde el inicio y se mantuvo relativamente estable, mientras que el *Recall* de validación mostró una tendencia similar a las configuraciones anteriores, estabilizándose también en 0.37. Este comportamiento refleja un menor grado de sobreajuste en comparación con las configuraciones del 75% y 80%, pero el modelo aún enfrenta dificultades significativas para mejorar su capacidad predictiva.

En conclusión, los resultados obtenidos con Random Forest en BD2 resaltaron las limitaciones del modelo para manejar escenarios de desequilibrio moderado como en este caso de impago de tarjetas de crédito. Aunque la configuración del 75% mostró un rendimiento ligeramente superior en términos de *AUPRC*, todas las configuraciones evaluadas presentaron un rendimiento subóptimo en *Recall* y *Precision*, lo que limita la aplicabilidad del modelo en este contexto. Estos hallazgos subrayan la necesidad de explorar ajustes adicionales de los hiperparámetros, la implementación de estrategias complementarias, como técnicas de re-muestreo

que se aplicarán posteriormente o modelos más complejos, para mejorar su rendimiento en este tipo de problemas.

9.2.1.2.2. ***Resultados con XGBoost.***

El modelo XGBoost fue avaluado sin aplicar el parámetro "*scale_pos_weight*", mostrando un rendimiento limitado en la detección de la clase minoritaria. Los valores de *Recall* oscilaron entre 0.3828 y 0.3865 (ver Tabla 13), indicando que el modelo identificó aproximadamente el 38% de los casos de impago, un rendimiento similar al observado con Random Forest. Sin embargo, la *Precisión* alcanzó valores moderados, entre 0.5616 y 0.5682, lo que refleja que solo entre el 56% y el 57% de las predicciones positivas realizadas por el modelo correspondieron a casos reales de impago.

Tabla 13

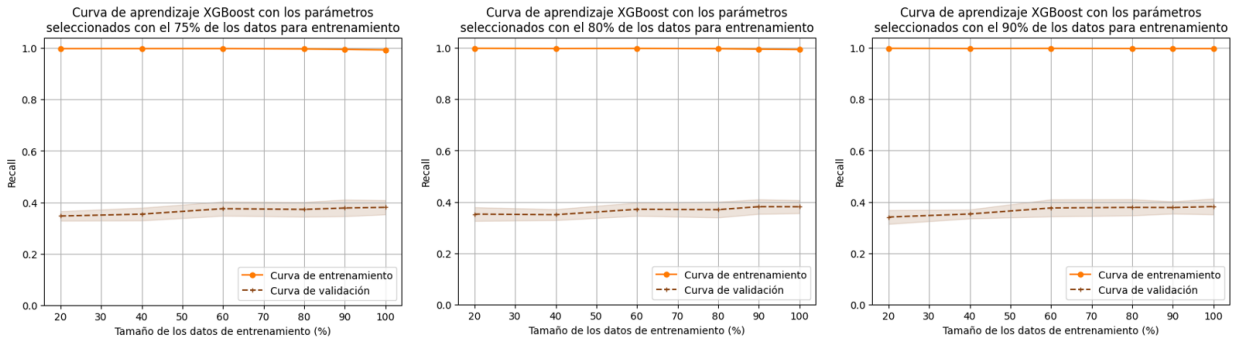
Métricas sin técnicas de re-muestreo para la BD2 con el modelo XGBoost sin el parámetro "scale_pos_weight"

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=500, max_depth=7, alpha=0.25, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0.25, learning_rate=0.4*	**0,3865**	0,5616	0,4963
80%	*n_estimators=500, max_depth=7, alpha=0.25, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0, learning_rate=0.4*	0,3855	0,5669	**0,5507**
90%	*n_estimators=500, max_depth=9, alpha=0.5, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0, learning_rate=*0.5	0,3828	**0,5682**	0,4932

EL *AUPRC* presentó valores entre 0.4963 y 0.5507, con el mejor resultado en la configuración del 80%. Esto sugiere que el modelo logra un balance ligeramente superior entre *Recall* y *Precision* con esta configuración, aunque persistieron limitaciones en la capacidad de predicción. Adicionalmente, las curvas de aprendizaje, como se aprecia en la figura 19, revelaron un sobreajuste significativo en todas las configuraciones, con un *Recall* de entrenamiento de 1 y un *Recall* de validación estabilizado entre 0.37 y 0.38; evidenciando una capacidad limitada capacidad para generalizar en datos no vistos. Sin embargo, la configuración del 80% mostró un mejor balance general, logrando el mayor valor de *AUPRC* (0.5507) entre las configuraciones evaluadas. En comparación, las configuraciones del 75% y 90% presentaron valores de *AUPRC* más bajos y comportamientos similares, sugiriendo que el cambio en los parámetros de profundidad y regularización no tuvo un impacto significativo en el rendimiento del modelo.

Figura 19

Curvas de aprendizaje para el modelo XGBoost con BD2 sin aplicar técnicas de re-muestreo



Cuando se incorporó el parámetro "*scale_pos_weight*", este fue ajustado en aproximadamente 3.52 (calculado como la relación entre la cantidad de observaciones de la clase mayoritaria y la minoritaria: 23364 / 6636). Este ajuste permitió al modelo priorizar la detección de la clase minoritaria, logrando un aumento notablemente en el *Recall*, que alcanzó valores entre 0.6421 y 0.6427 (ver Tabla 14). Sin embargo, este ajuste afectó negativamente a la *Precisión*, que

disminuyó a un rango de 0.4584 a 0.4640 incrementando los falsos positivos. El *AUPRC*, con valores entre 0.5507 y 0.5520, reflejó un rendimiento consistente en las configuraciones evaluadas y mayor en comparación con las configuraciones sin "*scale_pos_weight*", destacándose la configuración del 80% como la más equilibrada entre *Recall* y *Precision.* Este resultado se complementó con las curvas de aprendizaje (ver Figura 20), las cuales evidenciaron una menor brecha entre entrenamiento y validación en comparación con las configuraciones sin este ajuste; en especial con la configuración del 80%, lo que sugiere un mejor balance entre el sesgo y varianza en comparación con las otras configuraciones con la modificación del parámetro *"scale_pos_weight"*. Aunque persistieron limitaciones en la capacidad de generalización, este escenario mostró un rendimiento más estable, posicionándolo como la opción más adecuada para abordar el desafío de la detección de impagos en un contexto de desbalance moderado. Estas observaciones resaltan la importancia de utilizar métricas complementarias como AUPRC y las curvas de aprendizaje para evaluar el comportamiento integral del modelo y seleccionar la configuración óptima.

Tabla 14

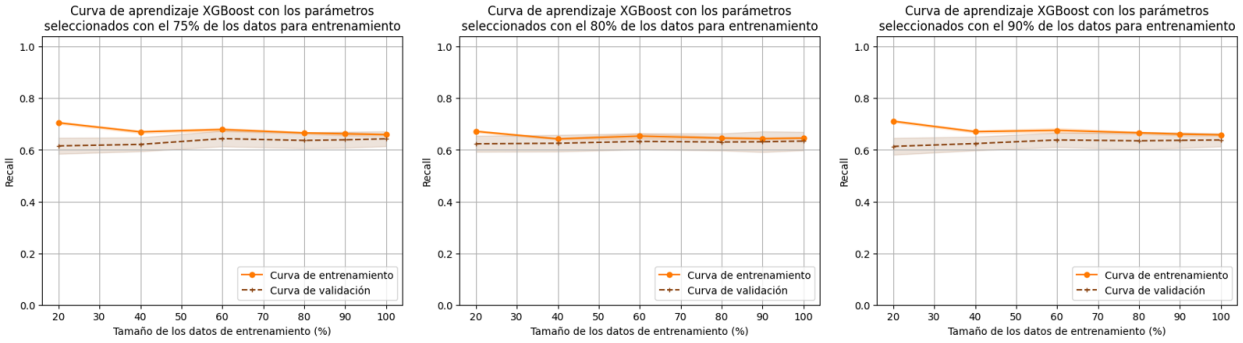
Métricas sin técnicas de re-muestreo para la BD2 con el modelo XGBoost con parámetro "scale_pos_weight"

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=20, max_depth=3, alpha=0.25, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0.5, learning_rate=0.5, scale_pos_weight': 3.5*	0,6422	0,4624	**0,5520**
80%	*n_estimators=20, max_depth=3, alpha=0.5, booster=gbtree, colsample_bytree=0.2, gamma=0.5, lambda=0.25, learning_rate=0.4, scale_pos_weight': 3.5*	**0,6427**	**0,4640**	0,5507

90%	*n_estimators=20, max_depth=3, alpha=0.5, booster=gbtree, colsample_bytree=1, gamma=0, lambda=0.5, learning_rate=0.5, scale_pos_weight': 3.5*	0,6421	0,4584	0,5510

Figura 20

Curvas de aprendizaje para el modelo XGBoost con BD2 sin aplicar técnicas de re-muestreo con "scale_pos_weight"



En conclusión, el modelo XGBoost presentó un rendimiento moderado en la detección de impagos en BD2, mostrando un comportamiento superior comparable con el modelo Random Forest en términos generales. El ajuste del parámetro *"scale_pos_weight"* resultó clave para mejorar la sensibilidad del modelo, logrando un incremento significativo en el *Recall* hasta valores de 0.6427. Sin embargo, este ajuste introdujo una disminución considerable en la *Precisión*, lo que refleja un incremento en los falsos positivos. Este comportamiento pone de manifiesto las limitaciones inherentes al ajuste del parámetro, ya que, aunque permite priorizar la detección de la clase minoritaria, compromete la precisión global del modelo.

La configuración con el 80% de datos de entrenamiento se destacó en ambos enfoques, con y sin *"scale_pos_weight"*, como la más equilibrada entre las métricas clave. En particular, con el ajuste, se alcanzaron valores más altos de *AUPRC* y un mejor balance en las curvas de aprendizaje, lo que sugiere una mayor capacidad de generalización. Estas observaciones resaltan la importancia

de evaluar métricas complementarias, como el *AUPRC*, junto con las curvas de aprendizaje, para analizar el comportamiento integral del modelo y tomar decisiones fundamentadas. Además, subrayan la necesidad de ajustar los parámetros del modelo en función de los objetivos específicos y las limitaciones operativas, especialmente en problemas donde los compromisos entre métricas clave son inevitables.

9.2.2. *Estrategia con submuestreo (RUS)*

En esta sección se analizó el rendimiento de los modelos Random Forest y XGBoost aplicando la técnica de submuestreo aleatorio en los conjuntos de datos "Credit Card Fraud Detection" y "Default of Credit Card Clients.

9.2.2.1. **Conjunto de datos "Credit Card Fraud Detection".**

Este conjunto de datos, compuesto por 284,315 registros, aborda la problemática de detección de transacciones fraudulentas, con solo un 0.17% de observaciones pertenecientes a la clase minoritaria (fraudes). Este desbalance extremo representa un desafío considerable para los modelos predictivos, ya que la cantidad limitada de ejemplos positivos dificulta la capacidad de los algoritmos para identificar patrones representativos de la clase minoritaria.

9.2.2.1.1. ***Resultados con Random Forest.***

Los resultados del modelo Random Forest aplicando previamente la técnica de submuestreo aleatorio, tal como se observa en la Tabla 15, mostraron una mejora significativa en el *Recall*, con valores que oscilaron entre 0.9162 y 0.9212, indicando una mayor capacidad del modelo para identificar instancias de fraude. La *Precisión* también mostró valores elevados, entre 0.9644 y 0.9671, lo que refleja que la mayoría de las predicciones realizadas como fraudes fueron correctas. Además, la métrica *AUPRC* presentó valores destacados, variando entre 0.9739 y 0.9814, lo que sugiere un balance óptimo entre Recall y Precision.

Tabla 15

Métricas con técnica de re-muestreo RUS para la BD1 con el modelo Random Forest

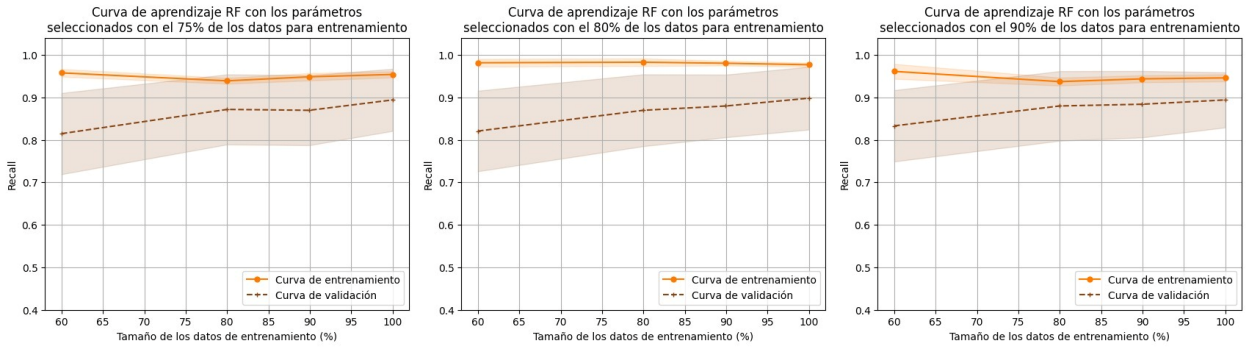
% Train size	Parámetros RF	Recall	Precision	AUPRC
75%	*n_estimators=14*, *criterion=gini*, *max_features=sqrt*, *max_depth=1*0, *min_samples_split*=10, *min_samples_leaf*=1	0,9167	0,9644	**0,9814**
80%	*n_estimators=14*, *criterion=gini*, *max_features=* sqrt, *max_depth=1*0, *min_samples_split*=2, *min_samples_leaf*=1	0,9162	0,9665	0,9802
90%	*n_estimators=1*0, *criterion*=gini, *max_features=log2*, *max_depth=10*, *min_samples_split=10*, *min_samples_leaf*=1	**0,9212**	**0,9671**	0,9739

El análisis de las curvas de aprendizaje (Figura 21) respalda estos resultados. En el conjunto de entrenamiento del 75%, las curvas iniciales mostraron valores inconsistentes "nan" debido a la falta de datos suficientes para entrenar el modelo. Sin embargo, conforme aumentó el tamaño de entrenamiento, el *Recall* en validación se estabilizó en 0.89, con un *Recall* de entrenamiento consistente de 0.95, lo que indica un equilibrio entre sesgo y varianza.

En el conjunto del 80%, las curvas evidenciaron una mejora similar en la generalización, con un *Recall* de validación que alcanzó 0.90. El *Recall* de entrenamiento fue ligeramente superior, situándose en 0.98, lo que refleja un leve sobreajuste. Finalmente, en el conjunto del 90%, el modelo mostró la brecha más reducida entre las curvas de entrenamiento y validación, logrando un *Recall* de validación consistente de 0.89 y un *Recall* de entrenamiento de 0.95.

Figura 21

Curvas de aprendizaje para el modelo Random Forest BD2 aplicando RUS



En conclusión, la aplicación de la técnica de submuestreo RUS mejoró notablemente el rendimiento del modelo Random Forest en este conjunto de datos. La configuración del 90% destacó por su equilibrio entre sesgo y varianza, logrando resultados consistentes en *Recall*, *Precision* y *AUPRC*. Estas mejoras resaltan la eficacia del submuestreo para abordar el desbalance extremo en problemas de detección de fraudes.

9.2.2.1.2. ***Resultados con XGBoost***

El modelo XGBoost también fue evaluado aplicando la técnica de submuestreo RUS para equilibrar las clases. Los resultados, presentados en la Tabla 16, mostraron un notable aumento en el *Recall*, con valores que oscilan entre 0.9289 y 0.9595, lo que refleja una pequeña mejora en la detección de fraudes en comparación con Random Forest. La *Precision* también alcanzó niveles elevados pero inferiores al Random Forest, variando entre 0.9247 y 0.9639, lo que indica que la mayoría de las predicciones positivas realizadas por el modelo fueron correctas. Además, la métrica *AUPRC* alcanzó valores sobresalientes, situándose entre 0.9822 y 0.9865, destacándose la configuración del 90% por su equilibrio entre las métricas clave.

Tabla 16

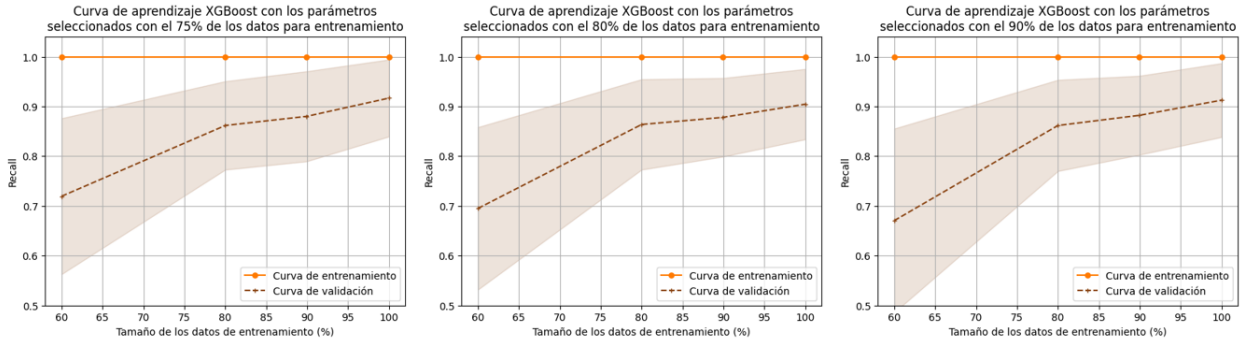
Métricas con técnica de re-muestreo RUS para la BD1 con el modelo XGBoost

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=50, max_depth=7, alpha=0.5*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.25*, *learning_rate*=0.5	**0,9595**	0,9247	0,9822
80%	*n_estimators=200, max_depth=7, alpha=0*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.25*, *learning_rate*=0.4	0,9289	0,9592	0,9848
90%	*n_estimators=100, max_depth=4, alpha=0*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.5*, *learning_rate*=0.4	0,9347	**0,9639**	**0,9865**

El análisis de las curvas de aprendizaje (Figura 22) respalda estos resultados. Aunque visualmente las curvas de los escenarios del 75%, 80% y 90% muestraron patrones similares, las diferencias en las métricas clave permiten distinguir su rendimiento. En el conjunto del 75%, el modelo presentó un sobreajuste considerable, con un *Recall* de entrenamiento que se mantuvo en 1.00 y un *Recall* de validación que aumentó progresivamente hasta alcanzar 0.92. Este patrón sugiere que, aunque el modelo logró detectar fraudes con alta precisión, podría estar menos adaptado a datos no vistos. En el conjunto del 80%, el *Recall* de validación se estableció en 0.90, mostrando un mejor equilibrio entre sesgo y varianza. Finalmente, en el conjunto del 90%, las curvas mostraron la brecha más reducida entre entrenamiento y validación, logrando un *Recall* de validación de 0.91 y un *AUPRC* más alto (0.9865). Estas diferencias, aunque sutiles, reflejaron cómo los parámetros como "*max_depth*" y "*learning_rate*" influyeron en el comportamiento del modelo y permitieron seleccionar la configuración más adecuada según los objetivos.

Figura 22

Curvas de aprendizaje para el modelo XGBoost para la BD1 aplicando RUS



Al comparar los resultados de los modelos Random Forest y XGBoost bajo la estrategia de submuestreo aleatorio (RUS), se identificaron diferencias clave en su rendimiento y comportamiento. El modelo Random Forest destacó por mantener valores de *Precision* más consistentes, oscilando entre 0.9644 y 0.9671, lo que indicó una mayor proporción de predicciones correctas como fraudes y minimizó el riesgo de falsos positivos. En contraste, XGBoost mostró una ventaja significativa en Recall, alcanzando hasta 0.9595 en la configuración del 75%, lo que lo hace más adecuado en contextos donde la sensibilidad, es decir, la capacidad para identificar la clase minoritaria es crítica.

En términos de AUPRC, XGBoost superó ligeramente a Random Forest, alcanzando un valor máximo de 0.9865 en la configuración del 90%, frente a 0.9739 de Random Forest. Este resultado sugiere que XGBoost logra un mejor balance entre *Recall* y *Precision* en configuraciones con mayor proporción de datos de entrenamiento, haciéndolo más adecuado en escenarios donde la detección precisa y equilibrada es prioritaria. Sin embargo, las curvas de aprendizaje mostraron que Random Forest ofreció una generalización más consistente en las configuraciones del 75% y 80%, con una menor brecha entre las métricas de entrenamiento y validación. En contraste, XGBoost presentó un sobreajuste más pronunciado en estos escenarios. Esto destaca la

importancia de ajustar parámetros clave como "*max_depth*" y "*learning_rate*" para reducir el sobreajuste y optimizar el rendimiento del modelo.

9.2.2.2. **Conjunto de datos: "Default of Credit Card Clients".**

Este conjunto de datos incluye 30,000 observaciones relacionadas con clientes de tarjetas de crédito, de las cuales el 22.12% corresponde a la clase minoritaria (impagos). Aunque presenta un desbalance moderado en comparación con BD1, su complejidad radica en el solapamiento entre clases, lo que dificulta la separación precisa de instancias por parte de los modelos de aprendizaje automático.

9.2.2.2.1. ***Resultados con Random Forest.***

El modelo Random Forest fue evaluado aplicando la técnica de submuestreo RUS con configuraciones optimizadas para maximizar el *Recall*. Los resultados se presentan en la Tabla 17, mostrando un rendimiento moderado en términos de *Recall* y *Precision*, con valores que oscilan entre 0.6594 y 0.6680, y entre 0.7006 y 0.7137, respectivamente. Esto indica que el modelo logró identificar entre el 65.9% y el 66.8% de los casos de impago, mientras que la mayoría de las predicciones positivas realizadas por el modelo fueron correctas. Además, el *AUPRC* mostró valores consistentes entre 0.7577 y 0.7652, reflejando un balance razonable entre *Recall* y *Precision*.

Tabla 17

Métricas con técnica de re-muestreo RUS para la BD2 con el modelo Random Forest

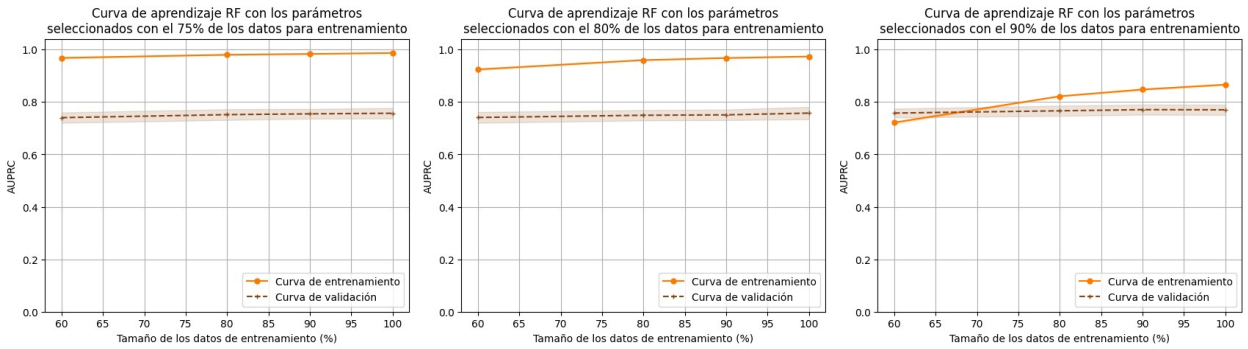
% Train size	Parámetros RF	Recall	Precision	AUPRC
75%	*n_estimators=14*, *criterion=gini*, *max_features=sqrt*, *max_depth=20*, *min_samples_split=2*, *min_samples_leaf*=2	0,6614	0,7067	0,7577

80%	*n_estimators=10*, *criterion=entropy*, *max_features=* sqrt, *max_depth=30*, *min_samples_split=10*, *min_samples_leaf*=2	**0,6680**	0,7006	0,7602
90%	*n_estimators=10*, *criterion*=gini, *max_features=log2*, *max_depth=10*, *min_samples_split=10*, *min_samples_leaf*=1	0,6594	**0,7137**	**0,7652**

El análisis de las curvas de aprendizaje (Figura 23) complementó estos hallazgos. En la configuración del 75%, las curvas iniciales presentaron valores inconsistentes debido a la insuficiencia de datos para entrenar el modelo. Sin embargo, conforme se incrementó el tamaño de entrenamiento, el *Recall* en validación se estabilizó en 0.76, mientras que el *Recall* de entrenamiento aumentó progresivamente hasta alcanzar 0.99. Este comportamiento reflejó un leve sobreajuste.

Figura 23

Curvas de aprendizaje para el modelo Random Forest BD2 aplicando RUS



En la configuración del 80%, se observó un patrón similar. El *Recall* de validación alcanzó 0.76 de manera consistente, mientras que el *Recall* de entrenamiento se mantuvo en 0.97, sugiriendo un equilibrio entre sesgo y varianza. Por otro lado, la configuración del 90% mostró un comportamiento distinto, con una brecha más reducida entre las curvas de entrenamiento y validación. El *Recall* en validación se estabilizó en 0.77, mientras que el Recall de entrenamiento

alcanzó 0.87. Esto sugiere que esta configuración logra una generalización más efectiva en comparación con las configuraciones anteriores en este conjunto de datos bajo la estrategia de submuestreo RUS.

9.2.2.2.2. ***Resultados con XGBoost***

El modelo XGBoost, al aplicar la técnica de submuestreo RUS en este conjunto de datos, mostró un rendimiento similar al de Random Forest en *Recall*, con valores entre 0.6672 y 0.6686, como se refleja en la tabla 18, evidenciando una mejora sustancial frente al escenario sin RUS. La *Precisión,* aunque ligeramente inferior (0.6879 a 0.6952) al modelo Random Forest, se mantuvo en un rango cercano. El cuanto al AUPRC, se obtuvieron valores entre 0.7426 y 0.7583, con el valor más alto en la configuración del 75%. Este comportamiento sugieró un balance aceptable entre las métricas clave, aunque con margen de mejora en la capacidad predictiva global.

Tabla 18

Métricas con técnica de re-muestreo RUS en la BD2 con el modelo XGBoost

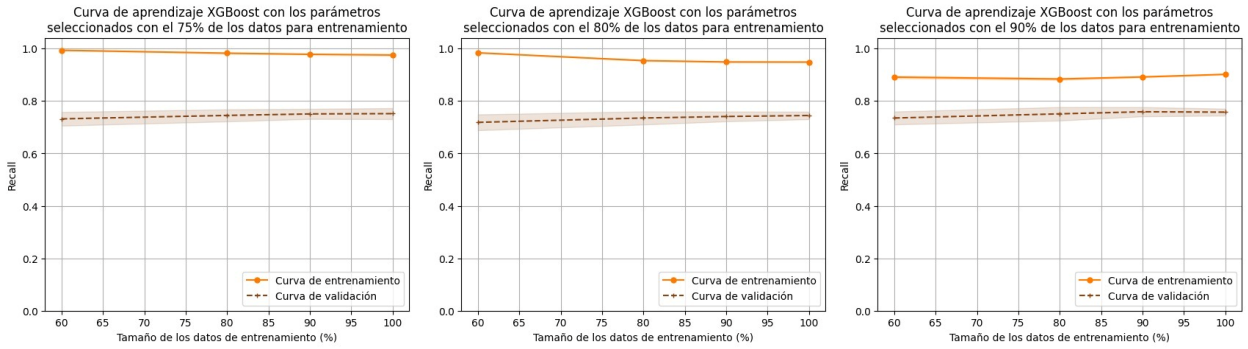
% Train size	**Parámetros XGBoost**	**Recall**	**Precision**	**AUPRC**
75%	*n_estimators=50, max_depth=7, alpha=0.25*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.25*, *learning_rate*=0.4	0,6672	**0,6952**	**0,7583**
80%	*n_estimators=500, max_depth=3, alpha=0*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.5*, *learning_rate*=0.4	0,6684	0,6879	0,7426
90%	*n_estimators=100, max_depth=4, alpha=0*, *booster=gbtree*, *colsample_bytree=1*, *gamma=0*, *lambda=0.5*, *learning_rate*=0.4	**0,6686**	0,6912	0,7548

El análisis de las curvas de aprendizaje (Figura 24) reveló que la configuración del 75% presentó un sobreajuste más pronunciado, con un *Recall* de entrenamiento constante en 0.98 y un

Recall de validación estabilizado en 0.75. Por otro lado, la configuración del 90%, aunque el *Recall* de validación mejoró ligeramente a 0.76, su menor *Recall* de entrenamiento (0.90) indica que no aprovechó completamente los datos disponibles, lo que podría limitar su rendimiento en aplicaciones prácticas. En contraste, la configuración del 80% alcanzó un mejor equilibrio entre sesgo y varianza, con un *Recall* de validación constante en 0.75 y de entrenamiento en 0.95, , posicionándose como la opción más robusta en términos de estabilidad y generalización.

Figura 24

Curvas de aprendizaje para el modelo XGBoost en la BD2 aplicando RUS



Comparando los resultados de los modelos con RUS frente a los obtenidos sin esta técnica, se observa una mejora sustancial en el Recall, que pasó de un rango de 0.3828-0.3865 a valores entre 0.6672-0.6686, evidenciando la eficacia del submuestreo para abordar el desbalance de clases. El AUPRC también mejoró significativamente, con valores que oscilaron entre 0.7426 y 0.7583 con RUS, en comparación con el rango de 0.5507-0.5520 sin esta técnica, lo que reflejó un mejor balance entre Recall y Precision. No obstante, los resultados generales siguen siendo moderados, indicando que, aunque RUS es efectivo para mitigar parcialmente el desbalance, no aborda completamente las limitaciones del problema.

9.2.3. *Estrategia con sobremuestreo (ROS y SMOTE)*

En esta sección se aplicaron técnicas de sobremuestreo como ROS (Random Over Sampling) y SMOTE (Synthetic Minority Oversampling Technique) .

9.2.3.1. **Conjunto de datos "Credit Card Fraud Detection".**

Este conjunto de datos, compuesto por 284,315 registros, aborda la problemática de detección de transacciones fraudulentas, con solo un 0.17% de observaciones pertenecientes a la clase minoritaria (fraudes). Este desbalance extremo representa un desafío considerable para los modelos predictivos, ya que la cantidad limitada de ejemplos positivos dificulta la capacidad de los algoritmos para identificar patrones representativos de la clase minoritaria

9.2.3.1.1. ***Resultados con Random Forest.***

El modelo Random Forest, al aplicar las técnicas de sobremuestreo ROS y SMOTE, como se presenta en las Tablas 19 y 20, mostró un rendimiento sobresaliente en todas las métricas clave (*Recall, Precision y AUPRC*), alcanzando valores perfectos o cercanos a 1 en los conjuntos de validación. Este rendimiento indicó una detección perfecta de fraudes, eliminando falsos negativos y minimizando falsos positivos. La alta *Precision* (0.9995–0.9999) reflejó que casi todas las predicciones positivas realizadas por el modelo corresponden a casos reales de fraude. El *AUPRC*, con valores de 0.9999 o cercanos, valida aún más el balance óptimo logrado entre *Recall* y *Precision*, destacando el excelente rendimiento del modelo al discriminar entre las clases en los datos sobremuestreados.

Tabla 19

Métricas con técnica de re-muestreo ROS para la BD1 con el modelo Random Forest

% Train size	**Parámetros RF**	**Recall**	**Precision**	**AUPRC**
75%	*n_estimators*=10, *criterion=entropy*, *max_features=log2*, *max_depth*=20,	1	0,9999	0,9999

	min_samples_split=2, *min_samples_leaf*=1			
80%	*n_estimators=200*, *criterion=gini*, *max_features*= sqrt, *max_depth=2*0, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9999	1
90%	*n_estimators*=10, *criterion*=gini, *max_features=sqrt*, *max_depth=20*, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9999	0,9999

Tabla 20

Métricas con técnica de re-muestreo SMOTE para la BD1 con el modelo Random Forest

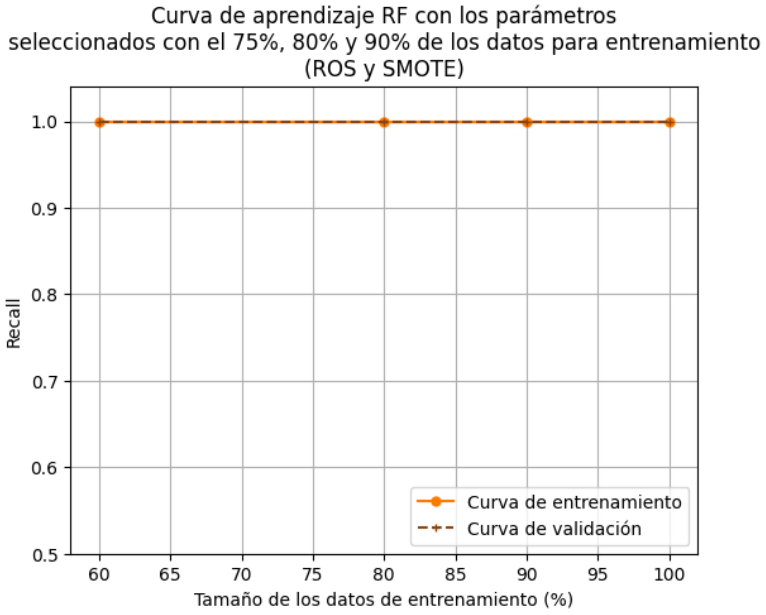
% Train size	**Parámetros RF**	**Recall**	**Precision**	**AUPRC**
75%	*n_estimators=5*0, *criterion=gini*, *max_features=sqrt*, *max_depth*=20, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9995	0,9999
80%	*n_estimators=200*, *criterion=gini*, *max_features*= sqrt, *max_depth=2*0, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9996	0,9999
90%	*n_estimators=2*0, *criterion*=gini, *max_features=sqrt*, *max_depth=20*, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9996	0,9999

El análisis de las curvas de aprendizaje (Figura 25) reveló un comportamiento uniforme para ambas técnicas, con valores perfectos de *Recall* tanto en entrenamiento como en validación desde etapas tempranas, indicando que el modelo logró aprender perfectamente las relaciones presentes en los datos balanceados. Aunque este rendimiento parece ideal, también sugiere un

posible sobreajuste, ya que los datos sobremuestreados replican o sintetizan casos de la clase minoritaria, generando una representación que no refleja con precisión la distribución real de los datos. Este comportamiento perfecto de las métricas indicó que el modelo no sería capaz de generalizar adecuadamente a nuevos datos no balanceados, limitando su aplicabilidad en escenarios reales.

Figura 25

Curvas de aprendizaje para el modelo Random Forest en la BD1 aplicando RUS



9.2.3.1.2. ***Resultados con XGBoost.***

El modelo XGBoost también fue evaluado utilizando ROS y SMOTE, obteniendo resultados superiores en todas las métricas clave, como se presenta en las Tablas 21 y 22. Los valores de Recall, Precision y AUPRC alcanzaron 1 o valores cercanos en todas las configuraciones, lo que indicó una detección perfecta de fraudes en los conjuntos de validación. Las curvas de aprendizaje no proporcionaron información adicional debido al ajuste perfecto mostrado en todas las configuraciones, con valores constantes de Recall y Precision en

entrenamiento y validación. De esta forma, aunque estas métricas y las curvas de aprendizaje que arrojan resultados perfectos son prometedores, también indican un sobreajuste severo, ya que el modelo podría no ser capaz de generalizar de manera efectiva a nuevos datos no balanceados.

Tabla 21

Métricas con técnica de re-muestreo ROS para la BD1 con el modelo XGBoost

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=500, max_depth=9, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9972	0,998
80%	*n_estimators=200, max_depth=7, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9994	0,9999
90%	*n_estimators=500, max_depth=4, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9971	0,9998

Tabla 22

Métricas con técnica de re-muestreo SMOTE para la BD1 con el modelo XGBoost

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=500, max_depth=9, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9992	1
80%	*n_estimators=500, max_depth=9, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9992	1
90%	*n_estimators=500, max_depth=9, alpha=0, booster=gbtree, colsample_bytree=0.2, gamma=0, lambda=0, learning_rate*=0.02	1	0,9993	1

Al igual que con Random Forest, las técnicas ROS y SMOTE mostraron resultados similares en XGBoost, logrando métricas cercanas a 1 en todas las configuraciones evaluadas.

Aunque SMOTE presentó una ligera ventaja en *Precision* y *AUPRC* en algunas configuraciones, ambos métodos lograron balancear eficazmente las clases y maximizar las métricas claves en los datos balanceados. Sin embargo, los resultados obtenidos con XGBoost reflejaron una limitada capacidad de generalización hacia nuevos datos no balanceados, debido al sobreajuste inducido por el sobremuestreo.

9.2.3.2. **Conjunto de datos: "Default of Credit Card Clients".**

Este conjunto de datos incluye 30,000 observaciones relacionadas con clientes de tarjetas de crédito, de las cuales el 22.12% corresponde a la clase minoritaria (impagos). Aunque presenta un desbalance moderado en comparación con BD1, su complejidad radica en el solapamiento entre clases, lo que dificulta la separación precisa de instancias por parte de los modelos de aprendizaje automático.

9.2.3.2.1. ***Resultados con Random Forest.***

El modelo Random Forest fue evaluado bajo las técnicas de sobremuestreo ROS y SMOTE en el conjunto de datos BD2, buscando determinar su impacto en la capacidad del modelo para manejar el desequilibrio de clases. En ambas técnicas, los resultados consignados en la tabla 23 y 23, muestran un ajuste perfecto para todas las configuraciones evaluadas. Tanto el *Recall* como la *Precision* alcanzaron valores cercanos a 1, lo que indicó que el modelo logró detectar correctamente todas las instancias de impago mientras mantuvo un nivel mínimo de falsos positivos. El área bajo la curva de precisión-recall (*AUPRC*) también se mantuvo en valores óptimos, reforzando la efectividad del modelo en estos escenarios desbalanceados.

En ROS, los valores de *Precision* fueron ligeramente superiores en la configuración del 75%, mientras que en SMOTE los valores se mantuvieron consistentes en todas las configuraciones. Sin embargo, esta diferencia es mínima y no afecta significativamente la

capacidad general del modelo para manejar el desequilibrio de clases. Por lo tanto, ambas técnicas ofrecen resultados equivalentes con Random Forest para BD2.

Tabla 23

Métricas con técnica de re-muestreo ROS para la BD2 con el modelo Random Forest

% Train size	Parámetros RF	Recall	Precision	AUPRC
75%	*n_estimators*=10, *criterion=entropy*, *max_features=log2*, *max_depth*=20, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9999	0,9999
80%	*n_estimators=200*, *criterion=gini*, *max_features*= sqrt, *max_depth=2*0, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9999	1
90%	*n_estimators*=10, *criterion*=gini, *max_features=sqrt*, *max_depth=20*, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9999	0,9999

Tabla 24

Métricas con técnica de re-muestreo SMOTE para la BD2 con el modelo Random Forest

% Train size	Parámetros RF	Recall	Precision	AUPRC
75%	*n_estimators=5*0, *criterion=gini*, *max_features=sqrt*, *max_depth*=20, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9995	0,9999
80%	*n_estimators=200*, *criterion=gini*, *max_features*= sqrt, *max_depth=2*0, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9996	0,9999
90%	*n_estimators=2*0, *criterion*=gini, *max_features=sqrt*, *max_depth=20*, *min_samples_split*=2, *min_samples_leaf*=1	1	0,9996	0,9999

9.2.3.2.2. ***Resultados con XGBoost.***

El modelo XGBoost fue evaluado aplicando también las técnicas de sobremuestreo ROS y SMOTE. Los resultados presentados en la tabla 25 y 26, reflejan un comportamiento similar al observado en Random Forest, con un ajuste perfecto en las métricas evaluadas. En ambas técnicas, el *Recall* alcanzó 1 en todas las configuraciones, y la *Precision* se mantuvo en valores óptimos, con diferencias marginales que favorecen a SMOTE en la configuración del 90%. Este comportamiento refuerza que ambos enfoques de sobremuestreo son igualmente efectivos para BD2.

Tabla 25

Métricas con técnica de re-muestreo ROS para la BD2 con el modelo XGBoost

% Train size	**Parámetros XGBoost**	**Recall**	**Precision**	**AUPRC**
75%	*n_estimators=500, max_depth=9, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9972	0,998
80%	*n_estimators=200, max_depth=7, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9994	0,9999
90%	*n_estimators=500, max_depth=4, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9971	0,9998

Tabla 26

Métricas con técnica de re-muestreo SMOTE para la BD2 con el modelo XGBoost

% Train size	Parámetros XGBoost	Recall	Precision	AUPRC
75%	*n_estimators=500, max_depth=9, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9992	1
80%	*n_estimators=500, max_depth=9, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9992	1
90%	*n_estimators=500, max_depth=9, alpha=0*, *booster=gbtree*, *colsample_bytree=0.2*, *gamma=0*, *lambda=0*, *learning_rate*=0.02	1	0,9993	1

La comparación con estrategias previas, como las configuraciones sin re-muestreo y con RUS, destaca que ROS y SMOTE garantizan un balance perfecto entre *Recall* y *Precision*. Sin embargo, esto también puede evidenciar un posible sobreajuste, dado que el rendimiento es consistentemente óptimo en todos los escenarios evaluados. En aplicaciones reales, donde la distribución de datos puede variar, este comportamiento podría no replicarse con la misma efectividad. Además, aunque RUS no alcanzó el nivel de precisión de las técnicas de sobremuestreo, ofrece un enfoque más conservador que minimiza el riesgo de sobreajuste.

En conclusión, tanto Random Forest como XGBoost mostraron métricas sobresalientes bajo las técnicas de sobremuestreo. Sin embargo, este rendimiento perfecto en métricas clave refleja un claro sobreajuste, lo que limita su capacidad de generalización a nuevos datos no balanceados. Por lo tanto, la selección entre estas técnicas debe considerar el contexto específico del problema, evaluando cuidadosamente el riesgo de sobreajuste y la aplicabilidad de los modelos en datos representativos de la distribución original.

9.2.4. *Contraste de modelos y técnicas*

En esta sección se muestran los resultados comparativos de los modelos Random Forest y XGBoost, seleccionados previamente para cada conjunto de datos (BD1 y BD2) con base en las mejores configuraciones de parámetros identificadas. Estas configuraciones, derivadas del análisis de métricas clave como Recall, Precision, AUPRC, en conjunto con la evaluación de las curvas de aprendizaje, permitiendo evaluar el rendimiento bajo diferentes técnicas de re-muestreo (sin re-muestreo, RUS, ROS y SMOTE). El objetivo de este análisis no se limita a comparar el rendimiento de los modelos en la detección de la clase minoritaria, sino también a identificar patrones relevantes que permitan evaluar la robustez y aplicabilidad de cada técnica en distintos escenarios de desbalance. Las configuraciones específicas de parámetros empleadas se detallan en el Apéndice D, proporcionando un contexto adicional para la interpretación de los resultados.

Los conjuntos de datos evaluados, BD1 y BD2, presentan características contrastantes que impactan tanto el rendimiento de los modelos como los recursos computacionales requeridos. BD1 constó de 284315 observaciones, con solo un 0.17% de registros pertenecientes en la clase minoritaria, lo que representó un escenario de desbalance extremo, donde el limitado número de ejemplos positivos plantea un desafío considerable para los modelos. Por otro lado, BD2, con 30,000 observaciones y aproximadamente un 22% de registros en la clase minoritaria, refleja un desbalance moderado, que, aunque menos extremo, sigue representando un reto a nivel de datos puesto que presenta un solapamiento entre clases.

El tamaño de los conjuntos de datos tuvo un impacto directo en los tiempos de procesamiento, particularmente en el uso de técnicas de re-muestreo, como se evidencia en la tabla 27. La BD1, al ser casi 10 veces más grande que BD2, demandó mayores recursos computacionales, especialmente con técnicas de sobremuestreo como ROS y SMOTE. En la BD1,

los tiempos de procesamiento alcanzaron hasta 39 horas con Random Forest y 31.6 horas con XGBoost, mientras que en BD2 los mismos modelos requirieron solo 1.4 horas y 5.1 horas, respectivamente. En contraste, RUS destacó como la técnica más eficiente, con tiempos mínimos de procesamiento de 0.3 horas y 1.7 horas en la BD1, y 0.5 horas y 2.9 horas en la BD2 para Random Forest y XGBoost, respectivamente. Estos resultados posicionan a RUS como una opción viable en escenarios donde los recursos computacionales son limitados.

Tabla 27

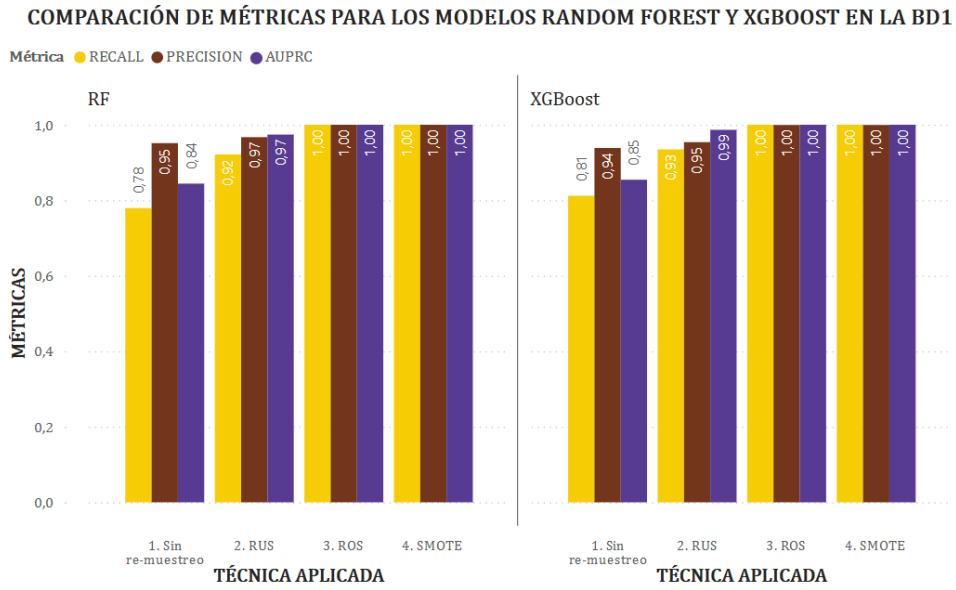
Tiempos de procesamiento (en horas) para los modelos Random Forest y XGBoost bajo diferentes técnicas de re-muestreo en los conjuntos de datos BD1 y BD2

Conjunto de datos	Modelo	Duración (h) Sin técnica de re-muestreo	Duración (h) RUS	Duración (h) ROS	Duración (h) SMOTE
BD1	RF	18.6	0.3	24.5	39
BD1	XGBoost	27.1	1.7	32	31.6
BD2	RF	0.9	0.5	1.2	1.4
BD2	XGBoost	7.7	2.9	4.8	5.1

Los resultados sin re-muestreo sirvieron como línea base para evaluar las limitaciones de los modelos en la detección de la clase minoritaria. En general, la BD1 (figura 28), a pesar de poseer una proporción extremadamente baja de esta clase, los modelos ensamblados mostraron mejores resultados que en BD2 (figura 29), a pesar de que este último presenta un desbalance menos extremo. En BD1, XGBoost mostró un rendimiento ligeramente superior, particularmente cuando el conjunto de entrenamiento abarcó el 90% de los datos, en comparación con el 80% recomendado para Random Forest. Por su parte, en BD2, la capacidad predictiva de XGBoost fue favorecida mediante el ajuste del parámetro "*scale_pos_weight*", que permitió un equilibrio entre *Precision* y *Recall* y mejoró la detección de la clase minoritaria, sacrificando precisión en menor medida que en BD1.

Figura 26

Comparación de métricas para los modelos Random Forest y XGBoost en la BD1



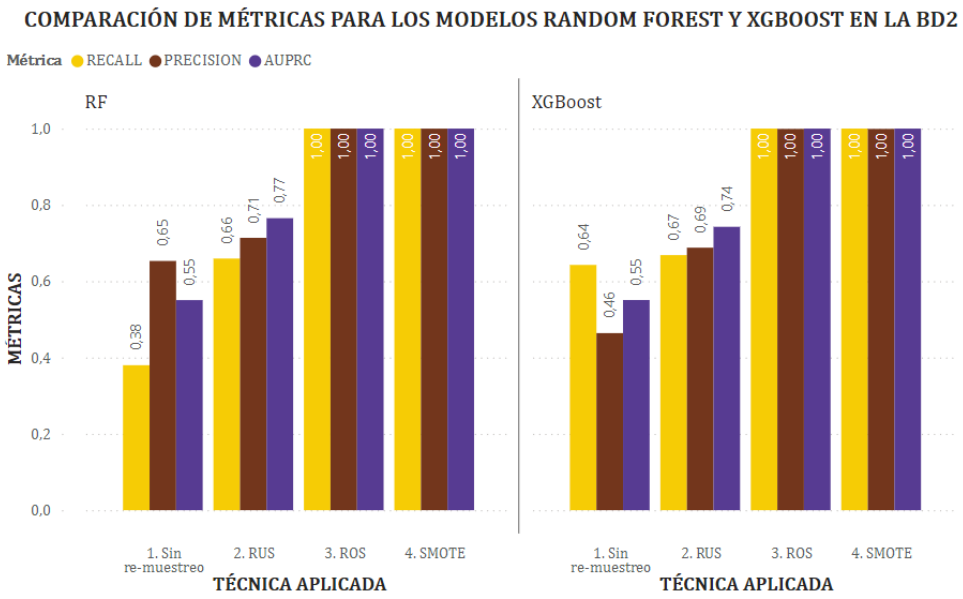
Las técnicas de re-muestreo ofrecieron resultados interesantes. En BD1, RUS mejoró significativamente el Recall, especialmente con XGBoost, logrando un balance favorable entre el rendimiento del modelo y los tiempos computacionales. En BD2, RUS, logró ciertas mejoras en Recall y AUPRC en ambos modelos, aunque con limitaciones puesto que solo se pudo capturar el 67% de los casos de impago de las tarjetas de crédito, resultado similar obtenido al modelo XGBoost sin técnica de re-muestreo.

Por su parte, ROS y SMOTE lograron métricas cercanas a 1 en Recall, Precision y AUPRC en ambos conjuntos de datos, indicando un rendimiento ideal en datos balanceados. Sin embargo, este comportamiento también evidenció un sobreajuste significativo, reflejando que los modelos aprenden patrones específicos de los datos balanceados en lugar de generalizar correctamente. Este sobreajuste estuvo acompañado de un aumento en la complejidad de los modelos, los cuales requirieron entre 200 y 500 estimadores para alcanzar dichos resultados. Por esta razón, las

matrices de confusión asociadas a estas técnicas no se consideran en el análisis posterior, ya que las métricas perfectas obtenidas no son representativas de su rendimiento en aplicaciones prácticas.

Figura 27

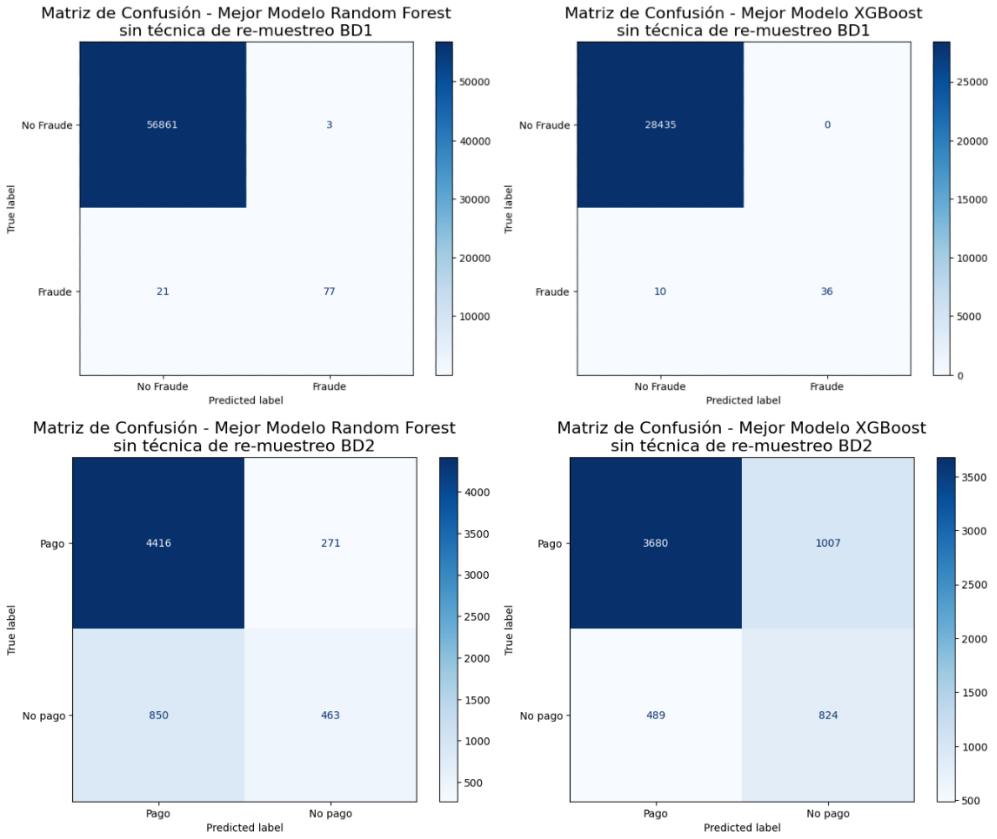
Comparación de métricas para los modelos Random Forest y XGBoost en la BD2



El análisis de las matrices de confusión proporciona una perspectiva crítica sobre el comportamiento de los modelos sin técnicas de re-muestreo (Figura 30). En BD1, se observa que XGBoost logra reducir los falsos negativos (10) en comparación con RF (21), manteniendo una alta precisión con un número mínimo de falsos positivos (0 y 3, en RF y XGBoost respectivamente). Este resultado destaca a XGBoost como el modelo más sensible para la detección de la clase minoritaria en un escenario de desbalance extremo. Por otro lado, en BD2, aunque XGBoost mantiene una ventaja en Recall debido a un menor número de falsos negativos (489 frente a 850 de RF), incurre en un mayor número de falsos positivos (1007 frente a 271 de RF), lo que podría traducirse en costos operativos más altos o interrupciones en transacciones legítimas. Estos resultados refuerzan la importancia de evaluar las implicaciones prácticas de los errores en cada contexto.

Figura 28

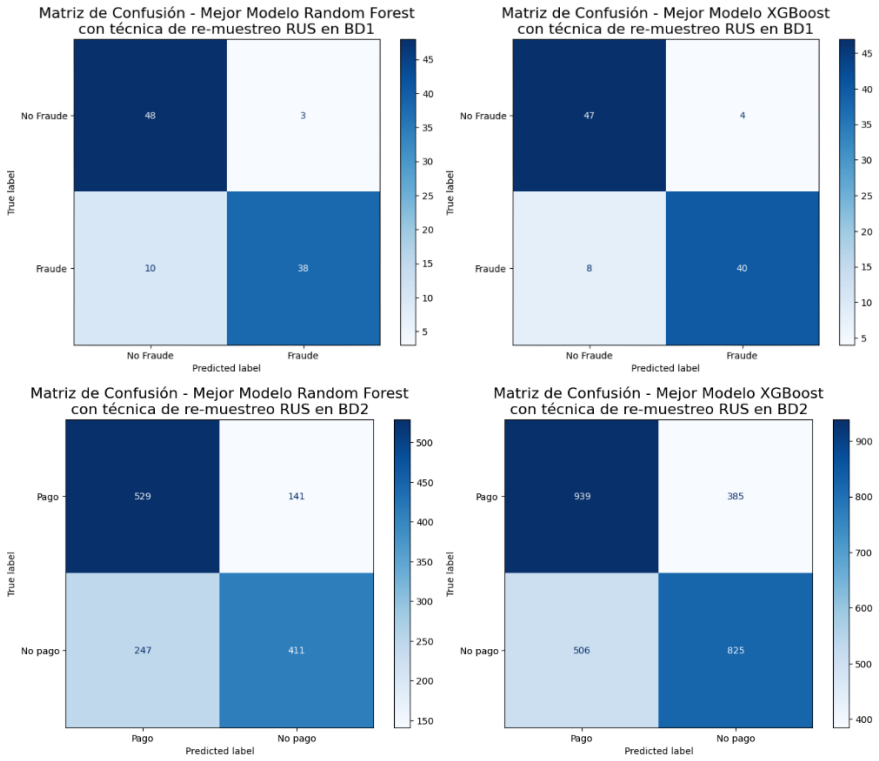
Matriz de confusión para los modelos Random Forest y XGBoost sin técnica de re-muestreo



La técnica de submuestreo aleatorio (RUS) presentó una mejora considerable en el Recall, como se ilustra en la Figura 31. En BD1, RF redujo los falsos negativos de 21 a 10, mientras que XGBoost logró el mejor rendimiento con solo 8 falsos negativos, aunque esto conllevó un ligero aumento en falsos positivos (de 3 a 4 en RF y de 0 a 4 en XGBoost). Este comportamiento posiciona a XGBoost con RUS como la mejor opción en BD1 para escenarios donde la detección de la clase minoritaria es crítica, como la prevención de fraudes. En BD2, RF con RUS logró un mejor equilibrio al reducir los falsos negativos de 825 a 411 y mantener un menor número de falsos positivos (141) en comparación con XGBoost, que alcanzó 385 falsos positivos. Este balance hace que RF con RUS sea más adecuado para aplicaciones donde la precisión operativa es prioritaria, como en la detección de impagos.

Figura 29

Matriz de confusión para los modelos Random Forest y XGBoost con técnica de re-muestreo RUS



A partir del análisis de las matrices de confusión y las métricas clave, se destacan recomendaciones claras para cada conjunto de datos. En la BD1, el conjunto de datos mayor y un desbalance extremo, XGBoost con RUS es la opción recomendada por su sensibilidad superior y su capacidad para minimizar los falsos negativos, un factor esencial en la detección de fraudes. En el caso de la BD2, un conjunto más pequeño y con desbalance moderado, Random Forest con RUS sobresale como el modelo más equilibrado, ofreciendo un rendimiento competitivo con un menor impacto en términos de falsos positivos y un costo computacional más bajo.

9.2.5. ***Discusión***

Los resultados obtenidos destacan la importancia de adaptar las estrategias de modelado al contexto específico del problema y al grado de desbalance presente en los datos. Tanto Random

Forest como XGBoost demostraron comportamientos diferenciados dependiendo de las técnicas de re-muestreo y configuraciones de hiperparámetros utilizadas, reflejando la necesidad de un enfoque integral para evaluar su rendimiento.

Las estrategias sin re-muestreo evidenciaron las limitaciones de ambos modelos para detectar efectivamente la clase minoritaria en escenarios de desequilibrio extremo, como en el conjunto de datos de detección de fraudes. Aunque métricas como la *Precision* fueron altas, el *Recall* permaneció en niveles subóptimos, dejando un porcentaje significativo de fraudes sin detectar. Por el contrario, las estrategias con submuestreo (RUS) lograron mejorar considerablemente el *Recall* en ambos conjuntos de datos, equilibrando las métricas clave y destacándose como una solución viable para manejar el desbalance. Sin embargo, estas técnicas mostraron una reducción en la *Precision*, lo que subraya la necesidad de un balance cuidadoso según las prioridades del problema.

Por su parte, las técnicas de sobremuestreo (ROS y SMOTE) ofrecieron resultados perfectos en todas las métricas evaluadas, lo que podría reflejar un posible sobreajuste al replicar o sintetizar instancias de la clase minoritaria. Aunque estas técnicas son efectivas para mejorar el rendimiento en escenarios controlados, su aplicabilidad a situaciones reales puede ser limitada debido a la mayor probabilidad de que los modelos aprendan patrones poco representativos del comportamiento general.

La incorporación del parámetro "*scale_pos_weigh*t" en XGBoost mostró un impacto positivo al priorizar el Recall, aunque a costa de una disminución significativa en la Precision. Esto resalta la importancia de ajustar los modelos en función de las consecuencias prácticas de los errores, especialmente en problemas donde los falsos positivos y negativos tienen costos distintos.

El análisis de normalización y reducción de dimensionalidad evidenció beneficios adicionales en términos de estabilidad y eficiencia computacional. Aunque no se observaron diferencias significativas en las métricas clave, la normalización contribuyó a una ejecución más eficiente y a una mejor visualización de los resultados. Estas mejoras pueden ser críticas en escenarios donde el tiempo de procesamiento y la interpretabilidad del modelo son factores relevantes. En conclusión, los resultados enfatizan la necesidad de combinar estrategias de re-muestreo, ajuste de hiperparámetros y técnicas de preprocesamiento para abordar efectivamente problemas con datos desbalanceados. La elección del enfoque óptimo debe considerar el contexto específico del problema y las implicaciones prácticas de las decisiones tomadas.

Fase 5: documentación

En esta etapa final, se consolidaron y organizaron todos los resultados obtenidos y recursos generados a lo largo del proyecto, asegurando su accesibilidad, reproducibilidad y transparencia. Esta documentación no solo respalda los hallazgos del estudio, sino que también construye una base para futuras investigaciones o aplicaciones prácticas relacionadas con problemas de clasificación en datos desbalanceados. A continuación, se describen los principales componentes de esta fase:

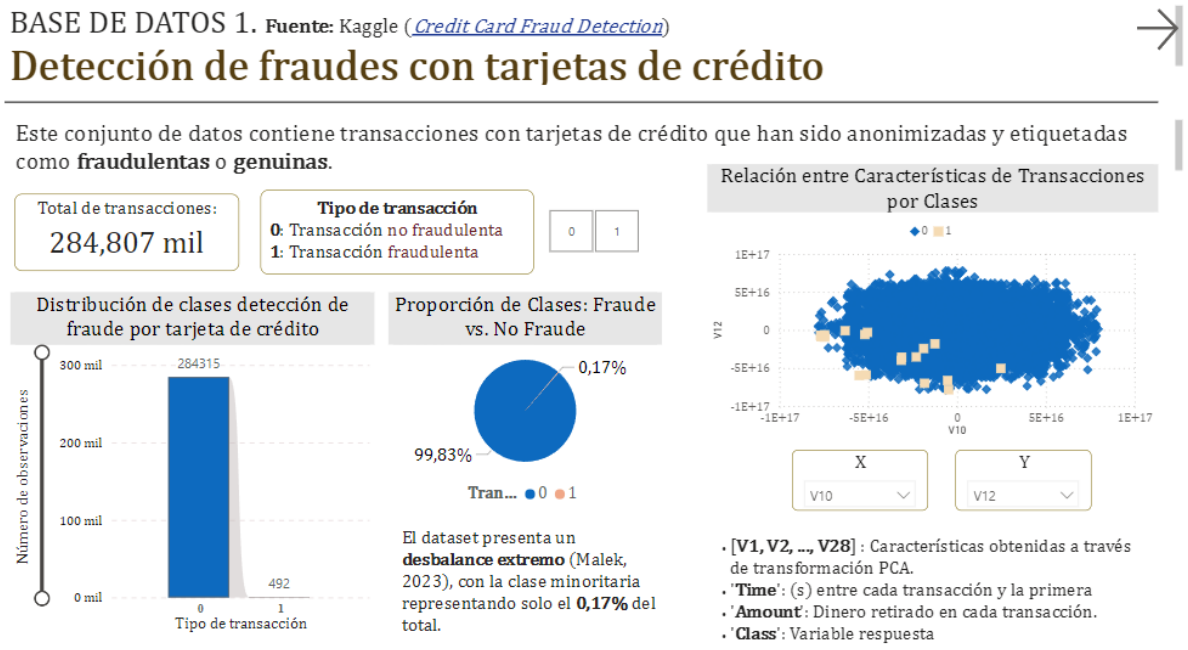
10.1. Visualización de resultados en Power Bi

Se desarrollaron visualizaciones interactivas en Power BI para analizar e interpretar los resultados obtenidos con los modelos Random Forest y XGBoost bajo diferentes técnicas de re-muestreo. Estas visualizaciones incorporan métricas claves, permitiendo identificar patrones y contrastar el impacto de las estrategias aplicadas en ambos conjuntos de datos. El informe completo está

disponible a través del siguiente enlace público en la nube de Power Bi (ver aquí). Además, la figura 30 presenta una captura de pantalla de una de las principales visualizaciones, mientras que un PDF con el informe completo puede consultarse en el Apéndice E.

Figura 30

Captura de pantalla de visualización generada en Power Bi



10.2. Artículo Académico

Un artículo académico titulado “Soluciones para conjuntos de datos con clases desbalanceadas combinadas con modelos de aprendizaje automático ensamblados” fue entregado y presentado en el **Coloquio Nacional de Estadística 2023**. En este trabajo se destacaron los objetivos, la metodología y los principales hallazgos relacionados con la clasificación en datos desbalanceados. Este artículo completo se encuentra disponible en el Apéndice F, y la documentación sobre la participación en el evento se puede consultar en el Apéndice G.

10.3. Repositorio en GitHub

Para garantizar la transparencia, reproducibilidad y accesibilidad de este proyecto, se creó un repositorio público el cuál puede ser consultado (aquí). Este repositorio organiza y centraliza todos los recursos relacionados, facilitando su consulta y reutilización. Entre los materiales incluidos se encuentran visualizaciones en formato Power BI, artículo académico presentado en el Coloquio Nacional de Estadística 2023; y Scripts de Python utilizados para la implementación de modelos, el preprocesamiento de datos y el análisis de resultados.

Conclusiones

Este estudio permitió identificar enfoques clave para abordar el problema del desbalance de clases en problemas de clasificación, destacando la efectividad de modelos ensamblados como Random Forest y XGBoost en combinación con estrategias de re-muestreo. Entre las técnicas evaluadas, el submuestreo aleatorio (RUS) emergió como la estrategia más adecuada para escenarios reales, al mejorar la detección de la clase minoritaria (*Recall*) sin comprometer en exceso la *Precisión*. Este hallazgo es particularmente relevante para problemas donde la clase minoritaria tiene un impacto crítico, como la detección de fraudes financieros o impagos. En cambio, aunque técnicas como ROS y SMOTE lograron métricas perfectas, se evidenció que estas introducen sobreajuste significativo, limitando su capacidad de generalización y haciendo que su uso no sea viable en aplicaciones prácticas para estos casos.

El análisis comparativo de los modelos permitió descubrir que la elección óptima depende del grado de desbalance y el tamaño del conjunto de datos. En escenarios de desbalance extremo, como BD1, XGBoost combinado con RUS se destacó por su capacidad para reducir los falsos

negativos, mejorando la detección de la clase minoritaria, mientras mantenía un nivel razonable de precisión al evitar un aumento excesivo en los falsos positivos. Este rendimiento refuerza su aplicabilidad en contextos donde es primordial detectar instancias de la clase minoritaria, incluso a costa de un leve incremento en los falsos positivos. Por otro lado, en conjuntos de datos con desbalance moderado y con el conjunto de datos con menor observaciones, como BD2, Random Forest con RUS sobresalió por lograr un mejor balance entre *Recall* y *Precisión,* reduciendo los falsos positivos y adaptándose mejor a escenarios donde las interrupciones en transacciones legítimas son críticas. La combinación de estas observaciones subraya que no existe una solución única y universal, sino que la estrategia debe ajustarse al contexto del problema.

Además, el tamaño del conjunto de datos influyó directamente en el comportamiento de los modelos y técnicas evaluadas. En conjuntos pequeños con una mayor proporción de la clase minoritaria, como BD2, los modelos presentaron un rendimiento más equilibrado y menor complejidad computacional. En contraste, en conjuntos grandes y con desbalance extremo, como BD1, se requirió una mayor sensibilidad hacia la clase minoritaria, priorizando estrategias como RUS para abordar de manera efectiva este reto. Estos resultados refuerzan la importancia de considerar tanto el tamaño como la distribución de las clases al diseñar las estrategias de modelado.

Adicionalmente, la construcción de una herramienta de visualización interactiva contribuyó de manera significativa a la interpretación de los resultados. Esta herramienta integró métricas clave como *Recall, Precisio*n y *AUPRC,* además de curvas de aprendizaje, permitiendo explorar el impacto de las diferentes técnicas evaluadas de manera intuitiva. Esto no solo facilita la comunicación de los hallazgos, sino que también constituye un recurso práctico para la toma de decisiones en problemas reales con datos desbalanceados.

Finalmente, los resultados obtenidos se consolidaron en un artículo presentado en el coloquio nacional de estadística sede Medellín en el año 2023 con los resultados de la revisión de literatura, que presenta un análisis detallado de las estrategias evaluadas y su aplicabilidad en contextos reales. Este producto académico sintetiza las contribuciones del presente trabajo y aporta evidencia valiosa para guiar futuras investigaciones en el campo del aprendizaje automático aplicado a problemas con datos desbalanceados. En conjunto, este estudio destaca la importancia de combinar estrategias de re-muestreo, ajuste de hiperparámetros y herramientas de visualización para abordar problemas complejos, priorizando enfoques adaptados al contexto específico del problema y las implicaciones prácticas de las decisiones de modelado.

Recomendaciones

Se recomienda ajustar cuidadosamente las estrategias de re-muestreo y los hiperparámetros de los modelos según las características del conjunto de datos y las prioridades del problema, priorizando métricas como el Recall en escenarios donde los falsos negativos tienen un alto impacto. Es importante considerar la incorporación de transformaciones como la normalización para mejorar la eficiencia computacional y facilitar la interpretación de resultados, especialmente en problemas de alta dimensionalidad. También se sugiere ampliar el alcance de los análisis replicando este enfoque en otros dominios, lo que permitiría validar la generalización de las estrategias utilizadas. Finalmente, es esencial monitorear el rendimiento del modelo en aplicaciones prácticas, garantizando su efectividad ante posibles cambios en la distribución de los datos o el entorno operativo.

Referencias Bibliográficas

Abellán, J., & Castellano, J. G. (2017). A comparative study on base classifiers in ensemble methods for credit scoring. *Expert Systems with Applications*, *73*, 1–10. https://doi.org/10.1016/j.eswa.2016.12.020

Artetxe, A., Graña, M., Beristain, A., & Ríos, S. (2020). Balanced training of a hybrid ensemble method for imbalanced datasets: a case of emergency department readmission prediction. *Neural Computing and Applications*, *32*(10), 5735–5744. https://doi.org/10.1007/s00521-017-3242-y

Balasubramanian, S., Kashyap, R., Cvn, S. T., & Anuradha, M. (2020, diciembre 1). Hybrid Prediction Model for Type-2 Diabetes with Class Imbalance. *Proceedings of the 2020 IEEE International Conference on Machine Learning and Applied Network Technologies, ICMLANT 2020*. https://doi.org/10.1109/ICMLANT50963.2020.9355975

Balogun, A. O., Lafenwa-Balogun, F. B., Mojeed, H. A., Adeyemo, V. E., Akande, O. N., Akintola, A. G., Bajeh, A. O., & Usman-Hamza, F. E. (2020). SMOTE-Based Homogeneous Ensemble Methods for Software Defect Prediction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12254 LNCS*, 615–631. https://doi.org/10.1007/978-3-030-58817-5_45

Bb, J., Deckert, M., Stefanowski, J., & Wilk, S. (s/f). *Integrating Selective Pre-processing of Imbalanced Data with Ivotes Ensemble*.

Bobadilla, J. (2020). *Machine learning y deep learning: usando Python, Scikit y Keras*. Ra-Ma.

Bobbili, N. P., & Cretu, A.-M. (2018). Adaptive weighting with SMOTE for learning from imbalanced datasets: A case study for traffic offence prediction. *CIVEMSA 2018 - 2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications, Proceedings*. https://doi.org/10.1109/CIVEMSA.2018.8439957

Breiman, L. (2001). *Random Forests* (Vol. 45).

Budhi, G. S., Chiong, R., & Wang, Z. (2021). Resampling imbalanced data to detect fake reviews using machine learning classifiers and textual-based features. *Multimedia Tools and Applications*, *80*(9), 13079–13097. https://doi.org/10.1007/s11042-020-10299-5

Cao, W., He, Y., Wang, W., Zhu, W., & Demazeau, Y. (2021). Ensemble methods for credit scoring of Chinese peer-to-peer loans. *Journal of Credit Risk*, *17*(3), 1–37. https://doi.org/10.21314/JCR.2021.005ï

Chang, C. C., Li, Y. Z., Wu, H. C., & Tseng, M. H. (2022). Melanoma Detection Using XGB Classifier Combined with Feature Extraction and K-Means SMOTE Techniques. *Diagnostics*, *12*(7). https://doi.org/10.3390/diagnostics12071747

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. https://doi.org/10.1145/2939672.2939785

Choudhary, R., & Shukla, S. (2021). A clustering based ensemble of weighted kernelized extreme learning machine for class imbalance learning. *Expert Systems with Applications*, *164*. https://doi.org/10.1016/j.eswa.2020.114041

Davis, J., & Goadrich, M. (2006). *The Relationship Between Precision-Recall and ROC Curves*.

Denil, M., & Trappenberg, T. (2010). Overlap versus imbalance. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6085 LNAI*, 220–231. https://doi.org/10.1007/978-3-642-13059-5_22

Diallo, M., Xiong, S., Emiru, E. D., Fesseha, A., Abdulsalami, A. O., & Elaziz, M. A. (2021). A hybrid multilayer perceptron under-sampling with bagging dealing with a real-life imbalanced rice dataset. *Information (Switzerland)*, *12*(8). https://doi.org/10.3390/info12080291

Duan, H., Wei, Y., Liu, P., & Yin, H. (2020). A novel ensemble framework based on K-means and resampling for imbalanced data. *Applied Sciences (Switzerland)*, *10*(5). https://doi.org/10.3390/app10051684

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. En *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* (Vol. 42, Número 4, pp. 463–484). https://doi.org/10.1109/TSMCC.2011.2161285

Galindo, J., & Tamayo, P. (2000). Credit Risk Assessment Using Statistical and Machine Learning: Basic Methodology and Risk Modeling Applications. En *Computational Economics* (Vol. 15).

Gulowaty, B., & Ksieniewicz, P. (2019). SMOTE Algorithm Variations in Balancing Data Streams. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11872 LNCS*, 305–312. https://doi.org/10.1007/978-3-030-33617-2_31

Guo, H., Diao, X., & Liu, H. (2019). Improving undersampling-based ensemble with rotation forest for imbalanced problem. *Turkish Journal of Electrical Engineering and Computer Sciences*, *27*(2), 1371–1386. https://doi.org/10.3906/elk-1805-159

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining: concepts and techniques* (Third Edition). Elsevier. https://doi.org/10.1016/C2009-0-61819-5

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L., & Seliya, N. (2019a). Examining characteristics of predictive models with imbalanced big data. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0231-2

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L., & Seliya, N. (2019b). Examining characteristics of predictive models with imbalanced big data. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0231-2

Hastie, Trevor., Tibshirani, Robert., & Friedman, Jerome. (2009). *The Elements of Statistical Learning Data Mining, Inference, and Prediction, Second Edition* (2nd ed. 2009.). Springer New York. https://doi.org/10.1007/978-0-387-84858-7

He, H., Zhang, W., & Zhang, S. (2018). A novel ensemble method for credit scoring: Adaption of different imbalance ratios. *Expert Systems with Applications*, *98*, 105–117. https://doi.org/10.1016/j.eswa.2018.01.012

Honnurappa, S., & Raghavendra, B. K. (2021). A Highly Robust Heterogenous Deep Ensemble Assisted Multi-Feature Learning Model for Diabetic Mellitus Prediction. *International Journal of Performability Engineering*, *17*(11), 926–937. https://doi.org/10.23940/ijpe.21.11.p3.926937

Inan, M. S. K., Hasan, R., & Alam, F. I. (2021). A Hybrid Probabilistic Ensemble based Extreme Gradient Boosting Approach for Breast Cancer Diagnosis. *2021 IEEE 11th Annual*

Computing and Communication Workshop and Conference, CCWC 2021, 1029–1035. https://doi.org/10.1109/CCWC51732.2021.9376007

James, Gareth., Witten, Daniela., Hastie, Trevor., & Tibshirani, Robert. (2013). An Introduction to Statistical Learning with Applications in R . En *An Introduction to Statistical Learning with Applications in R* (1st ed. 2013.). Springer New York.

Jason Brownlee. (2020, agosto 21). *Imbalanced Classification with the Fraudulent Credit Card Transactions Dataset*. https://machinelearningmastery.com/imbalanced-classification-with-the-fraudulent-credit-card-transactions-dataset/?utm_source=chatgpt.com.

Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, *6*(1), 40–49. https://doi.org/10.1145/1007730.1007737

Kaggle. (2018). *Credit Card Fraud Detection*. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

Khairy, M., Mahmoud, T. M., & Abd-El-Hafeez, T. (2024). The effect of rebalancing techniques on the classification performance in cyberbullying datasets. *Neural Computing and Applications*, *36*(3), 1049–1065. https://doi.org/10.1007/s00521-023-09084-w

Krawczyk, B., Galar, M., Jeleń, Ł., & Herrera, F. (2016). Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. *Applied Soft Computing Journal*, *38*, 714–726. https://doi.org/10.1016/j.asoc.2015.08.060

Kuncheva, L. I. (2001). COMBINING CLASSIFIERS: SOFT COMPUTING SOLUTIONS. En *Pattern Recognition* (pp. 427–451). WORLD SCIENTIFIC. https://doi.org/10.1142/9789812386533_0015

Kuncheva, L. I. (2014a). *Combining Pattern Classifiers: Methods and Algorithms, 2nd Edition*. Wiley.

Kuncheva, L. I. (2014b). *Combining Pattern Classifiers: Methods and Algorithms, 2nd Edition*. Wiley.

Lin, H. I., & Nguyen, M. C. (2020). Boosting minority class prediction on imbalanced point cloud data. *Applied Sciences (Switzerland)*, *10*(3). https://doi.org/10.3390/app10030973

Lin, W. C., Tsai, C. F., Hu, Y. H., & Jhang, J. S. (2017). Clustering-based undersampling in class-imbalanced data. *Information Sciences*, *409–410*, 17–26. https://doi.org/10.1016/j.ins.2017.05.008

Maimon, Oded., & Rokach, L. (2005). *Data Mining and Knowledge Discovery Handbook* (1st ed. 2005.). Springer US.

Maimon, Oded., & Rokach, Lior. (2010). Data Mining and Knowledge Discovery Handbook. En *Data Mining and Knowledge Discovery Handbook* (2a ed.). Springer US. https://doi.org/10.1007/978-0-387-09823-4

Malek, N. H. A., Yaacob, W. F. W., Wah, Y. B., Md Nasir, S. A., Shaadan, N., & Indratno, S. W. (2023). Comparison of ensemble hybrid sampling with bagging and boosting machine learning approach for imbalanced data. *Indonesian Journal of Electrical Engineering and Computer Science*, *29*(1), 598–608. https://doi.org/10.11591/ijeecs.v29.i1.pp598-608

Mondal, I. A., Haque, M. E., Hassan, A. M., & Shatabda, S. (2021). Handling Imbalanced Data for Credit Card Fraud Detection. *24th International Conference on Computer and Information Technology, ICCIT 2021*. https://doi.org/10.1109/ICCIT54785.2021.9689866

Murphy, K. P. (2012). Machine learning a probabilistic perspective . En *Machine learning a probabilistic perspective*. MIT Press.

Mwangi, P. I., Nderu, L., Mutanu, L., & Mwigereri, D. G. (2022). Hybrid Ensemble Model for Handling Class Imbalance Problem in Big Data Analytics. *International Conference on*

Electrical, Computer, and Energy Technologies, ICECET 2022. https://doi.org/10.1109/ICECET55527.2022.9872764

Nanni, L., Fantozzi, C., & Lazzarini, N. (2015). Coupling different methods for overcoming the class imbalance problem. *Neurocomputing*, *158*, 48–61. https://doi.org/10.1016/j.neucom.2015.01.068

Ng, W. W. Y., Zhang, Y., & Zhang, J. (2017). *BSMBoost for Imbalanced Pattern Classification Problems*. https://doi.org/10.0/Linux-x86_64

Ogunsanya, M., Isichei, J., & Desai, S. (2023). *Grid Search Hyperparameter Tuning in Additive Manufacturing Processes*. www.sciencedirect.com

Perlich, C. (2011). Learning Curves in Machine Learning. En *Encyclopedia of Machine Learning* (pp. 577–580). Springer US. https://doi.org/10.1007/978-0-387-30164-8_452

Pouriyeh, S., Vahid, S., Sannino, G., De Pietro, G., Arabnia, H., & Gutierrez, J. (2017a). A comprehensive investigation and comparison of Machine Learning Techniques in the domain of heart disease. *Proceedings - IEEE Symposium on Computers and Communications*, 204 – 207. https://doi.org/10.1109/ISCC.2017.8024530

Pouriyeh, S., Vahid, S., Sannino, G., De Pietro, G., Arabnia, H., & Gutierrez, J. (2017b). A comprehensive investigation and comparison of Machine Learning Techniques in the domain of heart disease. *Proceedings - IEEE Symposium on Computers and Communications*, 204 – 207. https://doi.org/10.1109/ISCC.2017.8024530

Quan, D., Feng, W., Dauphin, G., Wang, X., Huang, W., & Xing, M. (2022). A Novel Double Ensemble Algorithm for the Classification of Multi-Class Imbalanced Hyperspectral Data. *Remote Sensing*, *14*(15). https://doi.org/10.3390/rs14153765

Salunkhe, U. R., & Mali, S. N. (2016). Classifier Ensemble Design for Imbalanced Data Classification: A Hybrid Approach. *Procedia Computer Science*, *85*, 725–732. https://doi.org/10.1016/j.procs.2016.05.259

SAS. (2025). *Gestión del riesgo del crédito*. https://www.sas.com/es_mx/insights/risk-management/credit-risk-management.html.

scikit-learn. (2025). *GridSearchCV*. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

Shah, S. M. A., Usman, S. M., Khalid, S., Rehman, I. U., Anwar, A., Hussain, S., Ullah, S. S., Elmannai, H., Algarni, A. D., & Manzoor, W. (2022). An Ensemble Model for Consumer Emotion Prediction Using EEG Signals for Neuromarketing Applications. *Sensors*, *22*(24). https://doi.org/10.3390/s22249744

Shalev-Shwartz, S., & Ben-David, S. (2014a). *Understanding Machine Learning From Theory to Algorithms*. Cambridge University Press. https://doi.org/10.1017/CBO9781107298019

Shalev-Shwartz, S., & Ben-David, S. (2014b). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press. https://doi.org/DOI: 10.1017/CBO9781107298019

Sumathi, S., & Sivanandam, S. N. (2006). Introduction to data mining and its applications . En *Introduction to data mining and its applications* (1st ed. 2006.). Springer-Verlag.

Sun, B., Chen, H., Wang, J., & Xie, H. (2018). Evolutionary under-sampling based bagging ensemble method for imbalanced data classification. *Frontiers of Computer Science*, *12*(2), 331–350. https://doi.org/10.1007/s11704-016-5306-z

Tarekegn, A. N., Giacobini, M., & Michalak, K. (2021). A review of methods for imbalanced multi-label classification. En *Pattern Recognition* (Vol. 118). Elsevier Ltd. https://doi.org/10.1016/j.patcog.2021.107965

Timarán Pereira, S. R., Hernández Arteaga, I., Caicedo Zambrano, S. J., Hidalgo Troya, A., & Alvarado Pérez, J. C. (2016). Descubrimiento de patrones de desempeño académico con árboles de decisión en las competencias genéricas de la formación profesional. En *Descubrimiento de patrones de desempeño académico con árboles de decisión en las competencias genéricas de la formación profesional*. Universidad Cooperativa de Colombia. https://doi.org/10.16925/9789587600490

Winters, R. (2017). *Practical Predictive Analytics*.

xgboost. (2025). *XGBoost Parameters*. https://xgboost.readthedocs.io/en/stable/parameter.html.

Xia, Y., Liu, C., Li, Y. Y., & Liu, N. (2017). A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, *78*, 225–241. https://doi.org/10.1016/j.eswa.2017.02.017

Yeh, I. (2009). *Default of Credit Card Clients [Dataset]*. https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients . https://doi.org/https://doi.org/10.24432/C55S3H

Yotsawat, W., Wattuya, P., & Srivihok, A. (2021). Improved credit scoring model using XGBoost with Bayesian hyper-parameter optimization. *International Journal of Electrical and Computer Engineering*, *11*(6), 5477–5487. https://doi.org/10.11591/ijece.v11i6.pp5477-5487

Zheng, H., Sherazi, S. W. A., & Lee, J. Y. (2021). A Stacking Ensemble Prediction Model for the Occurrences of Major Adverse Cardiovascular Events in Patients with Acute Coronary

Syndrome on Imbalanced Data. *IEEE Access*, *9*, 113692–113704.

https://doi.org/10.1109/ACCESS.2021.3099795