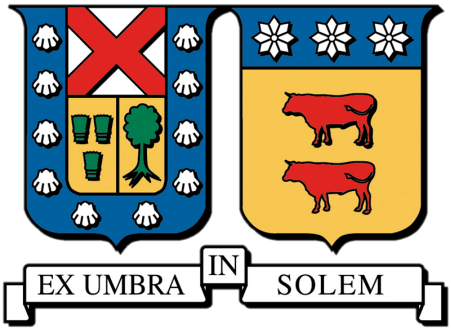

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INDUSTRIAS

Modelos predictivos de churn en telecomunicaciones: evaluación comparativa y análisis de aplicabilidad con técnicas de machine learning

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

AUTOR
JOAQUÍN AUGUSTO BERISTAIN CORREA

PROFESOR GUÍA

Dr. ESTEBAN ANDRÉS SALGADO VALENZUELA

PROFESOR CO-REFERENTE

MSc. ISMAEL ANDRÉS KAUAK VERDUGO

SANTIAGO, OCTUBRE 2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: **Modelos predictivos de churn en telecomunicaciones: evaluación comparativa y análisis de aplicabilidad con técnicas de machine learning**

Nombre del candidato(a): **Joaquín Augusto Beristain Correa**

Carrera / Grado: **Ingeniería Civil Industrial**

Campus: Vitacura **Departamento:** Departamento de Industrias

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Esteban Salgado, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por **(marcar una opción):**

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

_____

_____

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 21/10/2025 **Firma:** _____

Estudiante o Candidato(a):

Fecha: 21/10/2025 **Firma:** _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Tabla de Contenidos

1 Agradecimientos

Quiero empezar agradeciendo de todo corazón a mi madre, que siempre estuvo ahí para apoyarme. Gracias por esos momentos en los que me levantaste cuando más lo necesitaba, cuando pensaba que no me la podía, y me recordaste que si. No tengo palabras para agradecer todo el amor y apoyo que me has dado.

Agradezco mucho a Renato, quien me permitió quedarme con él durante los años que estuve estudiando en Santiago. Gracias por darme un lugar al que llegar, por tu apoyo y por estar siempre ahí para lo que necesitara. No habría sido lo mismo sin tu generosidad.

A Camila, mi polola, no sé ni cómo agradecerte. Si hoy estoy aquí, logrando todo lo que he logrado, es en gran parte gracias a ti. Fuiste una de las principales razones por las que empecé a creer en mí mismo y a pensar que sí podía. Si los últimos años fui el estudiante y ayudante que fui, fue porque tú siempre estuviste para apoyarme, darme ánimo y mostrarme que podía más de lo que creía. Gracias por todo tu amor y confianza.

Y finalmente, a Rena, que siempre estuvo ahí para hacerme reír, para hablar de todo y para darme una mano cuando lo necesitaba. Has sido uno de los pilares en este camino, y no puedo agradecerte lo suficiente por estar siempre ahí.

Gracias por acompañarme en este viaje y ser parte fundamental de cada uno de mis logros.

2 Resumen

Este estudio tiene como objetivo desarrollar un modelo predictivo que permita identificar a clientes con alta probabilidad de abandono (*churn*) en el sector de telecomunicaciones, utilizando técnicas de aprendizaje supervisado. Para ello, se utilizó el conjunto de datos IBM Telco Customer Churn, ampliamente empleado como benchmark en problemas de clasificación binaria.

El proceso se desarrolló bajo la metodología CRISP-DM, abarcando la comprensión del negocio, preparación de los datos, modelado y evaluación. Se implementaron y compararon múltiples algoritmos, entre ellos regresión logística, árboles de decisión, random forest, gradient boosting, XGBoost, CatBoost, KNN, Naive Bayes y redes neuronales (MLP). Todos los modelos fueron evaluados bajo un esquema de validación cruzada y métricas como Recall, F1-Score, ROC AUC y PR AUC.

Para comparar los modelos de forma integral, se desarrolló un análisis multicriterio que incluyó dimensiones adicionales como la eficiencia computacional (tiempos de entrenamiento y testeo) y la interpretabilidad de cada modelo. Posteriormente, se generaron más de 230.000 combinaciones de ponderaciones posibles entre estas métricas, con el fin de simular distintos escenarios de priorización por parte de los usuarios finales.

Los resultados muestran que el modelo MLP fue el más robusto, obteniendo el mejor puntaje en más del 34 % de las configuraciones. XGBoost y KNN también destacaron por su adaptabilidad a distintos criterios. Por otro lado, modelos como la regresión logística y el árbol de decisión, aunque no lideraron en desempeño predictivo, ofrecieron ventajas en interpretabilidad y velocidad de ejecución.

Este enfoque permite no solo seleccionar el modelo con mejor desempeño promedio, sino también

comprender su estabilidad frente a distintos perfiles de decisión. Lo anterior resulta clave para facilitar la implementación de soluciones basadas en inteligencia artificial en contextos reales, donde las prioridades pueden variar entre precisión, velocidad y explicabilidad.

3 Abstract

This study aims to develop a predictive model to identify customers with a high likelihood of churn in the telecommunications sector, using supervised learning techniques. The IBM Telco Customer Churn dataset was used as a benchmark for binary classification problems.

Following the CRISP-DM methodology, the process covered business understanding, data preparation, modeling, and evaluation. Multiple algorithms were implemented and compared, including logistic regression, decision trees, random forest, gradient boosting, XGBoost, CatBoost, KNN, Naive Bayes, and neural networks (MLP). All models were evaluated using cross-validation and standard classification metrics such as Recall, F1-Score, ROC AUC, and PR AUC.

To ensure a comprehensive comparison, the study incorporated additional evaluation dimensions, including computational efficiency (training and testing time) and model interpretability. An exhaustive robustness analysis was conducted by generating over 230.000 weight combinations across these metrics, simulating various prioritization scenarios.

The results show that the MLP model was the most robust, achieving the highest score in more than 34 % of all weight configurations. XGBoost and KNN also demonstrated strong adaptability across different evaluation profiles. In contrast, while models like logistic regression and decision trees did not excel in predictive performance, they provided advantages in terms of interpretability and low execution time.

4 Introducción

El mercado de las telecomunicaciones ha experimentado una transformación significativa en las últimas décadas, impulsada por el crecimiento de las tecnologías móviles, la digitalización de servicios y el acceso masivo a internet. Este sector se caracteriza por una alta penetración de mercado, márgenes competitivos ajustados y un entorno cambiante donde los clientes tienen múltiples alternativas para cambiar de proveedor con facilidad (Óskarsdóttir et al., 2017). En este contexto, las empresas enfrentan una presión constante por mantener a sus usuarios actuales, ya que el costo de adquisición de nuevos clientes suele superar con creces el costo de su retención (Athanassopoulos, 2000).

Asimismo, las compañías de telecomunicaciones recopilan grandes volúmenes de datos operacionales, demográficos y de comportamiento, como los registros de llamadas, patrones de uso y contactos con el servicio al cliente. Esta gran cantidad de información, habilita el uso de técnicas de analítica avanzada para extraer valor estratégico, particularmente en lo que respecta a la retención de clientes. (Óskarsdóttir et al., 2017). En consecuencia, el sector de las telecomunicaciones se ha convertido en un campo ideal para la aplicación de modelos predictivos, incluyendo enfoques tradicionales y algoritmos de aprendizaje automático.

En línea con esta tendencia, la fidelización de clientes ha adquirido un rol prioritario en las estrategias comerciales de las empresas del rubro. Estudios recientes indican que los clientes que permanecen en el tiempo no solo son más rentables, sino que también tienden a consumir más productos complementarios y actuar como promotores de la marca, fortaleciendo su reputación (Ganesh et al., 2000).

En el sector de las telecomunicaciones, la pérdida de clientes, conocida como *churn*, representa

uno de los principales desafíos estratégicos, ya que se trata de un mercado donde los usuarios pueden cambiar fácilmente de proveedor ante la mínima insatisfacción o incentivo competitivo. Diversos estudios han demostrado que los costos asociados a la adquisición de nuevos clientes superan significativamente los costos de retención (Athanassopoulos, 2000), lo que convierte la predicción del churn en una herramienta clave para la sostenibilidad financiera de las empresas del rubro.

Identificar anticipadamente a los clientes con alta probabilidad de abandono permite a las organizaciones diseñar campañas de fidelización más eficientes, optimizando la asignación de recursos y reduciendo las pérdidas por cancelación no gestionadas. Además, la permanencia prolongada de los clientes en la empresa se asocia con un mayor valor de vida (CLV), mayor consumo de servicios complementarios y un efecto positivo sobre la reputación corporativa a través del boca a boca (Ganesh et al., 2000).

El uso de datos históricos y actuales para anticipar comportamientos de cancelación ha evolucionado en los últimos años gracias a la incorporación de métodos de analítica avanzada y machine learning. En particular, se ha comprobado que los modelos predictivos pueden mejorar considerablemente la precisión en la identificación de potenciales desertores, sobre todo cuando se integran variables de comportamiento, socio-demográficas y de interacción con la red (Óskarsdóttir et al., 2017). Esto no solo permite anticiparse al abandono, sino también diseñar estrategias diferenciadas para cada segmento de riesgo.

La creciente disponibilidad de herramientas de analítica ha potenciado significativamente la capacidad de las empresas para anticiparse al abandono de clientes. No obstante, la efectividad de estas herramientas depende altamente de la calidad de los datos y de la capacidad de los modelos para generalizar sus predicciones en distintos contextos. En este sentido, el desarrollo de soluciones

de predicción del churn constituye un problema altamente abordado en la literatura, tanto por su impacto económico como por sus desafíos técnicos.

Dado que muchas organizaciones no pueden liberar sus datos por restricciones de confidencialidad, la comunidad científica ha recurrido a datasets abiertos para desarrollar y comparar modelos de predicción. En particular, el **IBM Telco Customer Churn dataset** ha adquirido relevancia como un benchmark ampliamente utilizado en investigaciones recientes debido a que presenta una estructura representativa del mundo de las telecomunicaciones, incorpora variables tanto demográficas como contractuales y de comportamiento. Esto lo convierte en una base sólida para evaluar el desempeño de distintos algoritmos de clasificación, comparar métricas y explorar técnicas de tratamiento de desbalance, interpretabilidad y optimización de modelos.

Asimismo, este estudio busca contribuir a la literatura existente mediante la evaluación comparativa de una variedad de algoritmos predictivos, desde enfoques tradicionales como la regresión logística hasta modelos avanzados de aprendizaje automático, incluyendo técnicas de ensamble y redes neuronales. Esta aproximación permitirá explorar la relación entre la complejidad del modelo y su rendimiento, y aportar evidencia empírica sobre buenas prácticas en la predicción del churn en contextos similares.

5 Problema de Investigación

El abandono de clientes (*customer churn*) constituye uno de los principales desafíos estratégicos en la industria de telecomunicaciones. Debido a la alta competitividad del mercado, la retención de usuarios se ha vuelto una prioridad, ya que atraer nuevos clientes suele ser significativamente más costoso que mantener a los existentes. En este contexto, la capacidad de anticipar qué clientes están en riesgo de desertar puede marcar una diferencia crítica en los resultados financieros y operativos de las empresas.

Si bien existen múltiples enfoques tradicionales para abordar este problema, como el análisis estadístico o los sistemas de reglas, en la última década se ha observado un crecimiento exponencial en la aplicación de modelos de aprendizaje automático (*machine learning*) que permiten identificar patrones complejos y no lineales en los datos históricos de los clientes (Verbeke et al., 2012). Sin embargo, estos modelos presentan desafíos importantes en su implementación práctica, especialmente en contextos donde la interpretabilidad de los resultados es tan relevante como su precisión. Por otro lado, el fenómeno del *churn* presenta una estructura inherentemente desbalanceada, donde la mayoría de los registros corresponden a clientes que no abandonan. Este desequilibrio puede sesgar los algoritmos de clasificación, generando modelos que aparentan tener un alto desempeño (por ejemplo, en términos de *accuracy*), pero que fallan al momento de identificar correctamente a los clientes que efectivamente están en riesgo.

A lo anterior se suma el hecho de que mucha investigaciones y aplicaciones reales tienden a centrarse en único modelo o en métricas poco sensibles al desbalance, lo cual dificulta evaluar el verdadero rendimiento de los enfoques predictivos y limita su aplicabilidad en escenarios de decisión real.

Frente a este panorama, se plantea la siguiente problemática central:

- ¿Qué modelo de clasificación, entre una variedad de técnicas de machine learning, ofrece el mejor equilibrio entre desempeño predictivo e interpretabilidad en un contexto de *churn* con los datos desbalanceados, utilizando el dataset de **IBM Telco Customer Churn**.

Este problema engloba múltiples dimensiones que deben ser abordadas de manera integrada:

- La evaluación del rendimiento en presencia de clases desbalanceadas.
- La comparación de modelos desde una perspectiva técnica y estratégica.
- La necesidad de explicar las predicciones para generar confianza en su uso.
- La alineación entre los objetivos de negocio y las capacidades del modelo.

En consecuencia, esta investigación no solo busca identificar el mejor modelo en términos métricos, sino también analizar su utilidad en contextos reales de toma de decisiones, considerando la importancia de comprender el porqué detrás de cada predicción.

6 Objetivos

Objetivo General

Evaluar y comparar el desempeño de distintos modelos de aprendizaje automático para la predicción de churn en clientes de telecomunicaciones, utilizando el dataset **IBM Telco Customer Churn**, con el fin de identificar enfoques que equilibren precisión, interpretabilidad y aplicabilidad práctica en la retención de clientes.

Objetivos Específicos

1. Explorar, limpiar y preparar el dataset **IBM Telco Customer Churn**, generando variables relevantes a nivel de cliente y servicio, y aplicando técnicas adecuadas de tratamiento de datos faltantes, codificación de variables categóricas y balanceo de clases.

2. Implementar y comparar una variedad de algoritmos de clasificación binaria, desde modelos lineales (como regresión logística) hasta técnicas avanzadas como árboles de decisión, métodos de ensamble (Random Forest, XGBoost, LightGBM, CatBoost) y redes neuronales.

3. Evaluar el rendimiento de los modelos mediante validación cruzada, utilizando métricas como *AUC*, *precision*, *recall*, *F1-score* y matriz de confusión, priorizando la capacidad de detectar clientes con alta probabilidad de abandono.

4. Interpretar los resultados desde una perspectiva de negocio, analizando la importancia de las variables predictoras y proponiendo recomendaciones prácticas orientadas a estrategias de retención de clientes.

7 Alcance

Este trabajo aborda el desarrollo de un sistema de predicción de churn utilizando técnicas de aprendizaje automático, con base en el dataset **IBM Telco Customer Churn**. El alcance del estudio considera todas las etapas del *pipeline* de ciencia de datos: desde la preparación y transformación de datos, hasta la selección, ajuste y validación de modelos predictivos. Asimismo, se evalúa la utilidad de los modelos desde un enfoque aplicado, identificando variables críticas para la toma de decisiones en retención. Sin embargo, el estudio no contempla la integración en sistemas empresariales ni la validación en bases de datos privadas.

8 Marco Teórico

8.1 *Churn* en telecomunicaciones

En el contexto empresarial, el término *churn* hace referencia a la pérdida de clientes o al abandono voluntario de un servicio por parte del consumidos. En el sector de las telecomunicaciones, donde las barreras de salida son bajas y la oferta es altamente competitiva, el *churn* representa una amenaza crítica para la sostenibilidad de los ingresos a largo plazo (Ganesh et al., 2000). La facilidad con la que los usuarios pueden cambiar de proveedor ante mejores condiciones, promociones o experiencias insatisfactorias convierte a la fidelización en un eje estratégico clave para las compañías del rubro.

El impacto económico del *churn* es considerable. Diversos estudios han demostrado que adquirir un nuevo cliente puede ser entre cinco y siete veces más costoso que retener a uno existente (Berson et al., 2000). Además, los clientes de larga data tienden a generar mayores ingresos, ya que presentan una mayor propensión a contratar servicios complementarios y actuar como promotores indirectos de la marca (Ganesh et al., 2000). Desde esta perspectiva, el *churn* no solo implica una pérdida directa de ingresos, sino también una reducción del valor de vida del cliente (Customer Lifetime Value o CLV), un indicador clave en la planificación financiera de las empresas de servicios (Verbeke et al., 2012).

Adicionalmente, las tasas de *churn* pueden afectar la imagen corporativa, provocar desequilibrios operacionales y exigir mayores inversiones en campañas de captación. Esto ha motivado a muchas organizaciones a desarrollar estrategias proactivas de retención, basadas en la segmentación de clientes, el monitoreo de la satisfacción, y mas recientemente, en el uso de herramientas de analítica avanzada que permiten anticipar conductas de abandono (Athanassopoulos, 2000).

En este contexto, las predicción del *churn* mediante técnicas de machine learning se ha posicionado como una solución eficaz para identificar clientes con alta probabilidad de abandono y actuar preventivamente. Este enfoque permite a las empresas enfocar sus recursos en los segmentos más vulnerables, diseñando acciones específicas que reduzcan la tasa de deserción y mejoren la eficiencia de sus campañas de fidelización (Óskarsdóttir et al., 2017).

8.2 Predicción del *churn*: desde la ciencia de datos

La predicción del *churn* ha sido ampliamente abordada desde el enfoque de la ciencia de datos, en tanto constituye un problema clásico de clasificación binaria, donde el objetivo es determinar si un cliente abandonará (*churn*) o permanecerá con la empresa. Este tipo de problema es especialmente atractivo para las telecomunicaciones debido a la abundancia de datos disponibles, tanto estructurados (por ejemplo, características del contrato, uso del servicio, historial de facturación) como no estructurados (interacciones por redes sociales, llamadas al *call center*, etc.) (Berson et al., 2000).

El proceso de predicción del *churn* se enmarca típicamente en metodologías como CRISP-DM (*Cross-Industry Standard Process for Data Mining*), que establece un ciclo iterativo compuesto por seis fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelado, evaluación y despliegue. Esta estructura ha sido ampliamente adoptada por adaptabilidad a distintos dominios y su enfoque centrado en la toma de decisiones (Wirth y Hipp, 2000).

Desde una perspectiva técnica, el éxito en la predicción del *churn* depende de diversos factores. Entre ellos destacan la calidad del dataset, la selección adecuada de variables predictoras, la elección del modelo y la correcta evaluación de su desempeño. Además, la mayoría de los datasets presentan un fuerte desbalance de clases, dado el numero de clientes que efectivamente abandonan

suele ser mucho menor que el de aquellos que permanecen. Esta característica exige un tratamiento especial, ya sea mediante técnicas de *resampling*, asignación de pesos o métricas especificas que penalicen los errores de clasificación de la clase minoritaria (Verbeke et al., 2012).

En los últimos años, el desarrollo de herramientas de aprendizaje automático ha ampliado el repertorio de métodos utilizados para abordar este problema. Los enfoques tradicionales como la regresión logística o los árboles de decisión han sido complementados o reemplazados por algoritmos mas sofisticados, como métodos de ensamble (e.g., **Random Forest**, **Gradient Boosting**) y **redes neuronales profundas** (Panimalar et al., 2024). Estos métodos permiten modelar relaciones no lineales y capturar interacciones complejas entre variables, mejorando la precisión en contextos donde las reglas de abandono no son evidentes ni lineales.

Además, el objetivo no es únicamente predecir con precisión qué clientes abandonarán, sino también proporcionar información útil para la toma de decisiones comerciales, como la identificación de segmentos de riesgo, las características asociadas al abandono, y la priorización de intervenciones estratégicas con base en el retorno potencial de cada cliente (Óskarsdóttir et al., 2017).

8.3 Modelos de aprendizaje automático aplicado al *churn*

La predicción de *churn* ha sido abordada mediante diversos algoritmos de clasificación supervisada, los cuales permiten modelar la probabilidad de abandono de un cliente a partir de variables observables. En este estudio se consideran modelos representativos de distintas familias, abarcando desde métodos estadísticos tradicionales hasta enfoques modernos basados en aprendizaje profundo. Esta variedad permite evaluar el compromiso entre precisión, interpretabilidad y complejidad computacional en la detección temprana de deserción.

Regresión Logística

La regresión logística es ampliamente reconocida como uno de los modelos estadísticos más utilizados para problemas de clasificación binaria, ya que permite modelar directamente la probabilidad de que ocurra un evento, en contraste con la regresión lineal, que predice una variable continua (Freedman, 2005). Este enfoque constituye un punto de partida clásico en tareas de predicción como el *churn*.

La lógica del modelo parte de una combinación lineal de variables predictoras, es decir:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

donde $x_1, x_2, \ldots, x_p$ representan las variables explicativas (como el tipo de contrato, el tiempo de permanencia o el uso de servicios), y $\beta_0, \beta_1, \ldots, \beta_p$ son los coeficientes del modelo que deben ser estimados. Esta combinación lineal se transforma mediante la función logística o sigmoide, la cual acota su salida al intervalo (0,1) para que pueda interpretarse como una probabilidad:

$$P(y = 1 \mid \mathbf{x}) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-\boldsymbol{\beta}^\top \mathbf{x}}}.$$

Así, el modelo entrega la probabilidad de que un cliente realice *churn* (es decir, que $y = 1$) dado su perfil representado por $\mathbf{x}$.

Sin embargo, la salida de la regresión logística corresponde a una probabilidad continua entre 0 y 1, por lo que para realizar una predicción categórica es necesario definir un umbral de decisión (*threshold*). Habitualmente, este umbral se fija en 0.5, de modo que las observaciones con probabilidad mayor a 0.5 se clasifican como abandono (*churn* = 1) y las menores como no abandono

(*churn* = 0). Este valor puede modificarse según las necesidades del negocio o los costos asociados a falsos positivos y falsos negativos, afectando directamente el balance entre *precision* y *recall* del modelo (Japkowicz y Stephen, 2002).

Durante el entrenamiento, los parámetros del modelo se ajustan maximizando la verosimilitud de observar los datos dados los parámetros. En la práctica, esto se logra minimizando la función de pérdida basada en la log-verosimilitud negativa:

$$\mathcal{J}(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \left[y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i) \right],$$

donde $\hat{p}_i$ es la probabilidad estimada de *churn* para el cliente i, y $y_i \in \{0, 1\}$ es la etiqueta real.

La verosimilitud representa la probabilidad de observar los datos efectivamente registrados dado un conjunto específico de parámetros del modelo, es decir:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} P(y_i \mid \mathbf{x}_i; \boldsymbol{\beta}).$$

Maximizar esta función implica encontrar los valores de $\boldsymbol{\beta}$ que hacen que los datos observados sean los más probables bajo el modelo. En la práctica, se trabaja con el logaritmo de la verosimilitud (*log-likelihood*) por conveniencia numérica, ya que transforma el producto en una suma y simplifica el cálculo de derivadas.

La razón por la cual maximizar la verosimilitud es equivalente a minimizar la función de pérdida se debe a que esta última se define como la *log-verosimilitud negativa*. Dado que el logaritmo es una función monótonamente creciente, maximizar $\log L(\boldsymbol{\beta})$ es matemáticamente equivalente a minimizar su negativo. Por tanto, el entrenamiento de la regresión logística puede formularse como

un problema de optimización convexa donde se busca reducir la pérdida de información entre las probabilidades predichas y las observadas.

Un aspecto crítico durante este proceso es evitar el sobreajuste (*overfitting*), fenómeno que ocurre cuando el modelo se ajusta excesivamente a los datos de entrenamiento, capturando ruido o patrones específicos que no se generalizan a nuevos datos. Esto produce un alto desempeño en el conjunto de entrenamiento, pero un deterioro notable en el conjunto de prueba.

Para mitigar este problema, se incorporan mecanismos de regularización, los cuales penalizan la complejidad del modelo al restringir la magnitud de los coeficientes. En la regresión logística, la forma más común es la regularización L2, que penaliza los coeficientes grandes mediante la adición de un término cuadrático en la función de costo:

$$\mathcal{J}_{\text{reg}}(\boldsymbol{\beta}) = \mathcal{J}(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_2^2,$$

donde λ es un hiperparámetro que controla el grado de penalización. Al imponer esta restricción, se reduce la varianza del modelo y se favorece una solución más estable y generalizable, evitando que los coeficientes crezcan en exceso para ajustarse a casos particulares del conjunto de entrenamiento. En otras palabras, la regularización actúa como un mecanismo de control de complejidad que previene el sobreajuste sin afectar significativamente la capacidad predictiva del modelo.

Uno de los principales beneficios de la regresión logística es su interpretabilidad. Cada coeficiente β_j puede interpretarse como el efecto logarítmico de la variable x_j sobre la razón de *odds* de abandono. Específicamente, un incremento unitario en x_j implica un cambio de β_j unidades en el log-*odds* de *churn*:

$$\log\left(\frac{P(y=1)}{P(y=0)}\right) = \boldsymbol{\beta}^\top \mathbf{x}.$$

Esto permite interpretar no solo si una variable influye en la probabilidad de *churn*, sino también en qué dirección y con qué intensidad.

En el contexto de la predicción del *churn*, la regresión logística actúa como un modelo base confiable, ideal para establecer un punto de comparación con modelos más complejos. Aunque su capacidad para capturar relaciones no lineales es limitada, su transparencia y facilidad de interpretación la hacen especialmente útil cuando se requiere justificar decisiones de negocio a partir de los resultados.

Naive Bayes

El clasificador *Naive Bayes* es un modelo probabilístico ampliamente utilizado en tareas de clasificación, especialmente cuando se requiere una solución rápida y con bajo costo computacional. Su fundamento se basa en el teorema de Bayes, incorporando el supuesto de independencia condicional fuerte entre las variables predictoras, dado el valor de la variable objetivo (Murphy, 2012). Este enfoque, a pesar de la simplicidad de sus supuestos, suele ofrecer un rendimiento competitivo en diversos contextos prácticos.

En el contexto del *churn*, el objetivo es estimar la probabilidad de que un cliente abandone el servicio ($Y = 1$) dado su vector de características $\mathbf{x} = (x_1, x_2, \ldots, x_p)$. Según el teorema de Bayes:

$$P(Y = k \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid Y = k)P(Y = k)}{P(\mathbf{x})}, \quad k \in \{0, 1\},$$

donde:

- $P(Y = k)$ es la probabilidad a priori de la clase k,
- $P(\mathbf{x} \mid Y = k)$ es la verosimilitud (la probabilidad de observar $\mathbf{x}$ dado que la clase es k),
- $P(\mathbf{x})$ es la probabilidad marginal de las características (igual para ambas clases, se puede omitir al clasificar).

El clasificador asigna la clase $\hat{y}$ que maximiza la probabilidad posterior:

$$\hat{y} = \arg \max_{k \in \{0,1\}} P(Y = k) \prod_{j=1}^{p} P(x_j \mid Y = k),$$

lo cual requiere asumir que las características x_j son condicionalmente independientes entre sí dado Y. Este supuesto simplifica el cálculo de la verosimilitud, pero rara vez se cumple exactamente en la práctica.

El modelo Naive Bayes puede adaptarse a distintos tipos de variables:

- **Variables categóricas**: Se estiman las probabilidades $P(x_j = v \mid Y = k)$ usando frecuencias relativas en los datos.
- **Variables continuas**: Se suele asumir que $x_j \mid Y = k \sim \mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$, lo que da lugar al clasificador **Naive Bayes Gaussiano**. En este caso, la **densidad condicional** se modela como:

$$f(x_j \mid Y = k) = \frac{1}{\sqrt{2\pi\sigma_{jk}^2}} \exp\left(-\frac{(x_j - \mu_{jk})^2}{2\sigma_{jk}^2}\right),$$

donde μ_{jk} y σ^2_{jk} se estiman a partir de la media y varianza muestral de x_j dentro de la clase k. La verosimilitud completa se calcula como:

$$P(\mathbf{x} \mid Y = k) = \prod_{j=1}^{p} f(x_j \mid Y = k).$$

Esta formulación permite aplicar el teorema de Bayes a vectores de atributos continuos sin necesidad de discretización previa.

Una de las principales ventajas del clasificador Naive Bayes es su eficiencia computacional, tanto en la fase de entrenamiento como en la de predicción, ya que no requiere procesos iterativos ni optimización numérica. Además, al tratarse de un modelo probabilístico, no solo proporciona una clase predicha, sino también la probabilidad posterior asociada a cada clase (Murphy, 2012).
No obstante, su principal limitación radica en el supuesto de independencia condicional. En presencia de fuertes correlaciones entre variables, el modelo puede entregar probabilidades distorsionadas y perder precisión. Aun así, su simplicidad, interpretabilidad y rendimiento competitivo en contextos bien estructurados lo convierten en una herramienta útil, especialmente como modelo base o cuando se dispone de pocos datos (Rish, 2001).

K-Nearest Neighbors (KNN)

El algoritmo de los k vecinos más cercanos (K-Nearest Neighbors, KNN) es un método de clasificación no paramétrico que se fundamenta en la medición de la similitud entre observaciones. A diferencia de los modelos que aprenden una función explícita para separar clases, KNN realiza inferencias de manera directa a partir del conjunto de entrenamiento, asignando la clase a una nueva observación según la etiqueta mayoritaria de sus k vecinos más próximos en el espacio de

características (Cover y Hart, 1967).

Dado un punto de entrada $\mathbf{x} \in \mathbb{R}^p$, el algoritmo KNN identifica el subconjunto $\mathcal{N}_k(\mathbf{x}) \subseteq \mathcal{D}$ formado por las k observaciones más cercanas a $\mathbf{x}$ bajo una métrica de distancia $d(\cdot, \cdot)$. La clase asignada se determina mediante votación mayoritaria:

$$\hat{y} = \arg\max_{c \in \mathcal{C}} \sum_{i \in \mathcal{N}_k(\mathbf{x})} \mathbb{I}(y_i = c),$$

donde $\mathcal{C}$ es el conjunto de clases posibles (en este caso, {0,1}) y $\mathbb{I}(\cdot)$ es la función indicadora.

La métrica de distancia más comúnmente utilizada es la *distancia euclidiana*:

$$d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2 = \sqrt{\sum_{j=1}^{p} (x_j - x'_j)^2},$$

aunque en problemas con variables de distinta escala o naturaleza, puede optarse por otras métricas como la distancia de Manhattan.

Desde un punto de vista estadístico, KNN puede ser interpretado como un estimador local del riesgo de pertenencia a una clase. De hecho, la probabilidad estimada de que una observación pertenezca a la clase positiva se puede definir como:

$$\hat{P}(Y = 1 \mid \mathbf{x}) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(\mathbf{x})} \mathbb{I}(y_i = 1).$$

Este enfoque convierte a KNN en un estimador de tipo kernel con ventana esférica fija, lo que lo vincula con la teoría de estimación no paramétrica (Hastie et al., 2009, sección 13.3).

KNN presenta diversas ventajas: es sencillo de implementar, no requiere suposiciones distribucionales sobre los datos y puede capturar fronteras de decisión de forma arbitrariamente compleja.

Además, su rendimiento tiende a mejorar conforme aumenta la cantidad de datos, siempre que la dimensionalidad se mantenga en niveles moderados (Hastie et al., 2009, cap. 13).

Sin embargo, el algoritmo también presenta limitaciones. Es sensible a la escala de las variables, requiere normalización previa y sufre fuertemente del fenómeno de la *maldición de la dimensionalidad*: a medida que aumenta el número de variables p, las distancias entre puntos tienden a volverse menos informativas. Además, su complejidad computacional durante la predicción es elevada, ya que requiere comparar el nuevo punto con todas las observaciones del conjunto de entrenamiento (Hastie et al., 2009, cap. 13).

La elección óptima del parámetro k es crucial. Valores bajos de k pueden generar modelos sobreajustados y sensibles al ruido (alta varianza), mientras que valores muy altos suavizan excesivamente la frontera de decisión (alto sesgo). La selección de k suele realizarse mediante validación cruzada.

En problemas como la predicción del *churn*, donde las fronteras de separación entre clases pueden ser complejas y no lineales, KNN puede ofrecer una alternativa robusta para establecer un punto de referencia, especialmente cuando se tiene un conjunto de entrenamiento suficientemente representativo.

Árboles de decisión

Los árboles de decisión son modelos de clasificación no paramétricos que segmentan el espacio de características de manera jerárquica, dividiéndolo en regiones homogéneas respecto de la variable objetivo (Hastie et al., 2009, cap. 9). A diferencia de los modelos lineales, que buscan una relación funcional explícita entre las variables independientes y la variable dependiente, los árboles construyen reglas de decisión basadas en divisiones sucesivas del espacio predictor. Gracias

a esta estructura, son altamente interpretables y han demostrado ser útiles en contextos como la predicción del *churn*, donde es fundamental explicar las razones detrás de la clasificación de un cliente como propenso al abandono (Verbeke et al., 2012).

Desde un punto de vista formal, el modelo busca construir una partición del espacio predictor $\mathbb{R}^p$ en regiones $R_1, R_2, \ldots, R_M$ tales que en cada región se asigne una clase (o probabilidad) constante. La construcción del árbol se realiza de manera recursiva, eligiendo en cada nodo una variable x_j y un umbral $t \in \mathbb{R}$ que permitan dividir el conjunto de datos en dos subconjuntos:

$$\mathcal{D}_{\text{left}} = \{(\mathbf{x}_i, y_i) : x_{ij} \leq t\}, \quad \mathcal{D}_{\text{right}} = \{(\mathbf{x}_i, y_i) : x_{ij} > t\}.$$

La selección óptima de la división se realiza minimizando una medida de impureza, que evalúa la heterogeneidad de las clases dentro de cada nodo. Las dos funciones más comunes son:

- **Índice de Gini**:

$$G(t) = 1 - \sum_{k=0}^{1} p_k^2,$$

  donde p_k representa la proporción de observaciones de clase k en el nodo.

- **Entropía de Shannon**:

$$H(t) = -\sum_{k=0}^{1} p_k \log_2 p_k,$$

  con la convención de que $0 \log 0 = 0$.

En ambos casos, el algoritmo selecciona la variable y el punto de corte que generan la mayor *ganancia de información* o reducción de impureza total:

$$\Delta I = I(\text{padre}) - \left(\frac{n_{\text{left}}}{n} I(\text{left}) + \frac{n_{\text{right}}}{n} I(\text{right}) \right),$$

donde $I(\cdot)$ puede ser Gini o entropía, y n es el número de observaciones en el nodo padre.

Una vez construido el árbol, la predicción para una nueva observación $\mathbf{x}$ se realiza recorriendo el árbol desde la raíz hasta una hoja terminal, aplicando las reglas de decisión en cada nodo. La clase predicha corresponde a la mayoría de clase en la hoja, y la probabilidad estimada del *churn* puede expresarse como:

$$\hat{P}(Y = 1 \mid \mathbf{x}) = \frac{1}{|\mathcal{D}_{\text{leaf}}|} \sum_{i \in \mathcal{D}_{\text{leaf}}} \mathbb{I}(y_i = 1),$$

donde $\mathcal{D}_{\text{leaf}}$ es el conjunto de observaciones que comparten hoja con $\mathbf{x}$.

Uno de los principales desafíos de los árboles es su tendencia a sobreajustar los datos, generando modelos excesivamente complejos que replican ruido o patrones espurios. Para mitigar este problema, se emplean técnicas como la *poda temprana*, que detiene el crecimiento si no se alcanza una mejora mínima en la impureza, o la *poda post hoc*, que elimina ramas enteras luego de entrenar el árbol completo.

Adicionalmente, se pueden restringir hiperparámetros como:

- la profundidad máxima del árbol (*max_depth*),
- el número mínimo de muestras para dividir un nodo (*min_samples_split*),
- o el número mínimo de muestras por hoja (*min_samples_leaf*).

En contextos como la predicción del *churn*, donde es relevante comprender las causas detrás de una predicción, los árboles de decisión permiten visualizar las reglas que conducen al abandono de

clientes, facilitando así la formulación de políticas de retención específicas.

Random Forest

El modelo Random Forest es una técnica de ensamblado basada en árboles de decisión, diseñada para mejorar la estabilidad y precisión de los modelos individuales mediante la agregación de múltiples predictores débiles. A diferencia de un solo árbol, que suele presentar alta varianza y propensión al sobreajuste, Random Forest mitiga este problema al introducir aleatoriedad tanto en el muestreo de datos como en la selección de variables predictoras (Breiman, 2001). Gracias a estas características, se ha utilizado con éxito en diversos problemas de clasificación, incluyendo la predicción del *churn* en contextos empresariales (Verbeke et al., 2012).

El procedimiento parte de la generación de múltiples árboles de decisión $\{T_1, T_2, \ldots, T_B\}$, cada uno entrenado sobre un subconjunto distinto del conjunto de entrenamiento, obtenido mediante **bootstrap**. Este método consiste en realizar un remuestreo *con reemplazo* de tamaño n, es decir, se seleccionan aleatoriamente n observaciones del conjunto original de n muestras, permitiendo repeticiones. Esto implica que algunos registros pueden aparecer múltiples veces en el conjunto de entrenamiento de un árbol, mientras que otros pueden no ser seleccionados. Esta variabilidad entre árboles es clave para reducir la varianza del modelo global (Breiman, 2001).

La predicción para una nueva observación $\mathbf{x}$ se realiza mediante votación mayoritaria:

$$\hat{y} = \arg \max_{k \in \{0,1\}} \sum_{b=1}^{B} \mathbb{I}(T_b(\mathbf{x}) = k),$$

y la estimación probabilística se calcula como:

$$\hat{P}(Y = 1 \mid \mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}(T_b(\mathbf{x}) = 1).$$

El número de variables consideradas por nodo, el número total de árboles B, la profundidad máxima de los árboles y el tamaño mínimo de los nodos hoja son hiperparámetros que permiten regular el modelo y adaptar su complejidad al problema.

Una de las ventajas centrales de Random Forest es su capacidad para reducir la varianza del modelo base sin incrementar significativamente el sesgo. Además, permite obtener medidas de *importancia de variables*, evaluando el impacto de cada predictor en la reducción de impureza promedio entre todos los árboles del bosque.

No obstante, al ser un ensamblado de múltiples modelos, pierde interpretabilidad respecto de un único árbol, y su entrenamiento puede ser computacionalmente costoso en escenarios con grandes volúmenes de datos o árboles muy profundos.

En el contexto de la predicción del *churn*, Random Forest es ampliamente utilizado por su robustez frente al ruido, su capacidad para manejar conjuntos desbalanceados y su habilidad para modelar relaciones no lineales y de interacción entre múltiples características del cliente.

Gradient Boosting

El modelo de Gradient Boosting pertenece a la familia de métodos de ensamblado secuencial, en los que cada modelo base se entrena para corregir los errores cometidos por el modelo anterior. A diferencia de **Random Forest**, que construye árboles de manera paralela e independiente, Gradient Boosting sigue un esquema aditivo y dependiente, empleando el descenso por gradiente como mecanismo de ajuste iterativo sobre una función de pérdida (Friedman, 2001).

Gracias a esta propiedad, ha mostrado un alto rendimiento en problemas complejos de clasificación, incluyendo la predicción del *churn*, donde las relaciones entre variables pueden ser altamente no lineales (Verbeke et al., 2012).

Desde un punto de vista formal, el objetivo es construir una función $F(\mathbf{x})$ que aproxime $f^*(\mathbf{x})$, el clasificador óptimo en términos de una función de pérdida diferenciable $\mathcal{L}(y, F(\mathbf{x}))$. El modelo se construye como una suma de funciones base:

$$F_M(\mathbf{x}) = \sum_{m=1}^{M} \nu \cdot h_m(\mathbf{x}),$$

donde:

- $h_m(\mathbf{x})$ representa el modelo base (usualmente un árbol de decisión poco profundo) en la iteración m,
- $\nu \in (0, 1]$ es la tasa de aprendizaje,
- M es el número total de iteraciones (o árboles).

El proceso comienza con una predicción inicial constante, por ejemplo:

$$F_0(\mathbf{x}) = \arg\min_{\gamma} \sum_{i=1}^{n} \mathcal{L}(y_i, \gamma),$$

y se actualiza de manera iterativa. En cada paso m, se calculan los *pseudo-residuos*:

$$r_i^{(m)} = - \left. \frac{\partial \mathcal{L}(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)} \right|_{F=F_{m-1}},$$

y se ajusta un nuevo árbol $h_m(\mathbf{x})$ sobre los datos $\{(\mathbf{x}_i, r_i^{(m)})\}$. Luego, el modelo se actualiza como:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu \cdot h_m(\mathbf{x}).$$

En el caso de clasificación binaria, la función de pérdida más utilizada es la *log-loss*:

$$\mathcal{L}(y, F(\mathbf{x})) = \log\left(1 + \exp(-2yF(\mathbf{x}))\right), \quad y \in \{-1, +1\},$$

y las probabilidades predichas se obtienen aplicando la función logística:

$$\hat{P}(Y = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp(-F_M(\mathbf{x}))}.$$

Este enfoque convierte el proceso de *boosting* en una aproximación funcional al clasificador óptimo mediante una serie de correcciones dirigidas por el gradiente.

Gradient Boosting se caracteriza por varios hiperparámetros críticos, entre ellos:

- M: número total de iteraciones (o árboles),
- ν: tasa de aprendizaje, que regula la contribución de cada nuevo árbol,
- la profundidad de los árboles base (*max_depth*),
- el tamaño mínimo de muestras por hoja (*min_samples_leaf*),
- y el submuestreo (*subsample*) como mecanismo adicional de regularización.

Una ventaja clave de este modelo es su capacidad para capturar interacciones complejas y estructuras no lineales, con un control fino del sobreajuste a través del aprendizaje lento y validación cruzada. Sin embargo, su entrenamiento puede ser computacionalmente costoso, y su interpretación es menos directa que en los árboles individuales.

En contextos de *churn*, donde los patrones de abandono son a menudo sutiles y dispersos entre múltiples variables, Gradient Boosting ha demostrado ser especialmente eficaz. Variantes modernas como **XGBoost**, **LightGBM** o **CatBoost** se basan en esta misma lógica, optimizando su implementación y escalabilidad en grandes volúmenes de datos.

XGBoost

El algoritmo XGBoost (eXtreme Gradient Boosting) es una implementación optimizada del método de Gradient Boosting, desarrollada por Chen y Guestrin (2016), que introduce mejoras en velocidad, eficiencia computacional y capacidad de regularización (Chen y Guestrin, 2016). Este enfoque ha mostrado alto rendimiento en tareas de clasificación con datos estructurados y ha sido aplicado con éxito en problemas como la predicción del *churn* en el sector de telecomunicaciones, alcanzando métricas de precisión y F-score superiores a otros enfoques de ensamblado (Sikri et al., 2024; Swetha y Dayananda, 2021).

Al igual que su antecesor, XGBoost construye el modelo como una suma aditiva de funciones base:

$$F_M(\mathbf{x}) = \sum_{m=1}^{M} f_m(\mathbf{x}), \quad f_m \in \mathcal{F},$$

donde cada f_m representa un árbol de decisión de regresión (CART), y $\mathcal{F}$ es el espacio de árboles posibles. La innovación clave de XGBoost radica en que optimiza una función objetivo regularizada, compuesta por dos términos:

$$\mathcal{L}^{(m)} = \sum_{i=1}^{n} \ell(y_i, F_{m-1}(\mathbf{x}_i) + f_m(\mathbf{x}_i)) + \Omega(f_m),$$

donde:

- $\ell(\cdot, \cdot)$ es la función de pérdida (por ejemplo, *log-loss* para clasificación),
- $\Omega(f_m) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$ es el término de regularización que penaliza la complejidad del árbol,
- T es el número de hojas, y w_j el peso asignado a la hoja j.

Esta estructura permite controlar tanto el ajuste del modelo como su capacidad predictiva, reduciendo el riesgo de sobreajuste mediante los hiperparámetros γ (penalización por profundidad) y λ (penalización L2 sobre los pesos de las hojas).

Una diferencia importante respecto al **Gradient Boosting** tradicional es el uso de una expansión de segundo orden de la función de pérdida para aproximar los incrementos del modelo. En cada iteración, se computan los pseudo-residuos como:

$$g_i = \frac{\partial \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i}, \quad h_i = \frac{\partial^2 \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i^2},$$

y el modelo actual se actualiza ajustando un nuevo árbol sobre los datos $(\mathbf{x}_i, g_i, h_i)$, optimizando el incremento del modelo según:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + f_m(\mathbf{x}).$$

El árbol f_m se construye dividiendo nodos que maximizan una ganancia regularizada, considerando tanto el ajuste del modelo como su complejidad. Para una división que genera subconjuntos R_L y R_R, la ganancia se calcula como:

$$\text{Gain} = \frac{1}{2}\left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right) - \gamma,$$

donde G y H son las sumas de gradientes y hessianos por cada nodo.

Además de su formulación estadística, XGBoost introduce una serie de optimizaciones prácticas:

- paralelización de la construcción de árboles,
- manejo eficiente de valores faltantes,
- submuestreo por columnas y filas,
- y poda anticipada de ramas no rentables.

Estas características permiten que XGBoost sea altamente eficiente incluso en grandes volúmenes de datos, manteniendo una excelente capacidad predictiva.

En problemas de *churn*, XGBoost ha demostrado ser especialmente efectivo por su capacidad de manejar conjuntos de datos desbalanceados, de estimar variables relevantes y de mantener un buen equilibrio entre interpretabilidad parcial y precisión.

LightGBM

LightGBM (*Light Gradient Boosting Machine*) es una implementación optimizada de **Gradient Boosting** desarrollada por Microsoft, que se distingue por su alta eficiencia en velocidad de entrenamiento, bajo consumo de memoria y excelente rendimiento predictivo (Ke et al., 2017). Al igual que **XGBoost**, se basa en la construcción aditiva de árboles de decisión como función base, pero introduce innovaciones clave en la estrategia de crecimiento y en la representación de las variables, lo que le permite escalar fácilmente a millones de instancias.

Formalmente, el modelo mantiene la estructura general del *boosting*:

$$F_M(\mathbf{x}) = \sum_{m=1}^{M} f_m(\mathbf{x}),$$

donde cada función $f_m(\mathbf{x})$ representa un árbol de regresión entrenado para aproximar el gradiente negativo de una función de pérdida.

Una de las principales innovaciones de LightGBM es su estrategia de construcción de árboles. A diferencia de algoritmos como **XGBoost**, que utilizan un crecimiento *nivel por nivel* (*level-wise*), LightGBM emplea un crecimiento *hoja por hoja* (*leaf-wise*). En cada iteración, el algoritmo identifica la hoja con la mayor ganancia potencial y la divide, generando árboles más profundos en menor tiempo. Esto permite una reducción más eficiente de la función de pérdida, aunque también puede incrementar el riesgo de sobreajuste si no se controla adecuadamente la profundidad máxima o el número de hojas.

Además, en este método se incorpora un mecanismo de *feature binning*, donde las variables continuas se discretizan previamente en un número limitado de intervalos (por defecto, 255). Esta transformación permite acelerar drásticamente el proceso de evaluación de divisiones y reducir el uso de memoria, ya que las estadísticas necesarias para construir los árboles se almacenan y procesan en forma de histogramas.

Desde el punto de vista de la optimización, LightGBM sigue el mismo enfoque de expansión de segundo orden utilizado en **XGBoost**. En cada iteración se calculan los gradientes g_i y hessianos h_i de la función de pérdida, y se entrena un árbol que minimice la siguiente expresión:

$$\mathcal{L}^{(m)} \approx \sum_{i=1}^{n} \left[g_i f_m(\mathbf{x}_i) + \frac{1}{2} h_i f_m^2(\mathbf{x}_i) \right] + \Omega(f_m),$$

donde $\Omega(f_m)$ penaliza la complejidad del árbol de acuerdo a la cantidad de hojas y los pesos

asignados.

Entre sus ventajas prácticas, LightGBM destaca por su:

- capacidad de manejar directamente variables categóricas sin codificación *one-hot*,
- compatibilidad con entrenamiento distribuido y multi-hilo,
- alto rendimiento en problemas con datos de alta dimensión y gran volumen.

Estas propiedades hacen de LightGBM una herramienta particularmente adecuada para tareas de clasificación como la predicción de *churn*, donde es crucial mantener un balance entre eficiencia computacional, escalabilidad y precisión predictiva.

CatBoost

CatBoost es un algoritmo de *boosting* por gradiente desarrollado por Yandex, diseñado específicamente para manejar eficientemente variables categóricas sin requerir transformaciones externas como codificaciones *one-hot* o *label encoding*. Esta característica lo convierte en una alternativa altamente efectiva para problemas con datos estructurados, como la predicción del *churn*, donde es común encontrar una gran cantidad de atributos no numéricos que contienen información relevante para la clasificación (Dorogush et al., 2018).

Al igual que otros algoritmos de *boosting*, CatBoost construye un modelo aditivo compuesto por árboles de decisión entrenados secuencialmente. La predicción se define como una suma de funciones base:

$$F_M(\mathbf{x}) = \sum_{m=1}^{M} f_m(\mathbf{x}),$$

donde cada función $f_m \in \mathcal{F}$ representa un árbol de regresión entrenado para minimizar una función de pérdida diferenciable a partir de los gradientes negativos. Sin embargo, CatBoost introduce dos innovaciones fundamentales que lo distinguen de otros métodos: el manejo ordenado de variables categóricas y una estrategia de entrenamiento conocida como *ordered boosting*.

En cuanto al tratamiento de variables categóricas, CatBoost evita la creación de columnas adicionales o pérdidas de información mediante una codificación basada en estadísticas condicionales del objetivo. Esta codificación se realiza en función del orden aleatorizado de las observaciones, de manera que la representación numérica de una categoría se calcule solo a partir de instancias anteriores. Formalmente, para una variable categórica c, la transformación asignada a la observación i es:

$$\tilde{c}_i = \frac{\sum_{j<i} \mathbb{I}(c_j = c_i) \cdot y_j + a \cdot \mu}{\sum_{j<i} \mathbb{I}(c_j = c_i) + a},$$

donde μ es la media global del objetivo, a es un parámetro de suavizado, y el orden $j < i$ corresponde a una permutación aleatoria de las observaciones. Este mecanismo evita la filtración de información futura (*target leakage*) y permite preservar la relación entre la categoría y el objetivo sin aumentar la dimensionalidad del *dataset*.

En paralelo, CatBoost emplea una técnica denominada *ordered boosting* para reducir el sesgo introducido por el uso simultáneo de los datos en la estimación de los gradientes y en el entrenamiento del modelo. A diferencia del *boosting* tradicional, que actualiza los modelos con base en el conjunto completo de datos, CatBoost entrena cada iteración utilizando solo una fracción ordenada de los datos, asegurando que las predicciones para una observación nunca dependan de su propio valor. Este enfoque reduce el riesgo de sobreajuste y mejora la estabilidad del proceso de entrenamiento.

A nivel computacional, CatBoost ofrece un rendimiento competitivo, con soporte nativo para ejecución en CPU y GPU, y requiere menor ajuste de hiperparámetros que otros algoritmos de *boosting*. Además, su arquitectura permite aprovechar estructuras internas eficientes para el manejo de grandes volúmenes de datos.

En el contexto de *churn*, CatBoost destaca por su capacidad para modelar relaciones complejas entre atributos categóricos como tipo de plan, método de facturación o características del servicio contratado, manteniendo al mismo tiempo una buena capacidad de generalización y un bajo riesgo de *overfitting*.

Red Neuronal

Las redes neuronales artificiales (ANN, por sus siglas en inglés) son modelos computacionales inspirados en la arquitectura del sistema nervioso biológico y ampliamente utilizadas en tareas de clasificación por su capacidad para aproximar funciones no lineales complejas (Goodfellow et al., 2016). A diferencia de los métodos basados en árboles, una red neuronal opera como una composición de funciones paramétricas organizadas en capas, donde la información se propaga mediante una secuencia de transformaciones lineales y no lineales. Esta estructura la convierte en una herramienta particularmente flexible, capaz de modelar interacciones sutiles entre variables en contextos como la predicción del *churn* (Keramati et al., 2014).

Una red neuronal típica de tipo *feedforward* está compuesta por una capa de entrada, una o más capas ocultas, y una capa de salida. Cada capa oculta transforma el vector de entrada mediante una operación lineal seguida de una función de activación no lineal (Goodfellow et al., 2016, cap.6). Formalmente, si se considera una red con una sola capa oculta, la predicción $\hat{y}$ para una observación $\mathbf{x} \in \mathbb{R}^p$ se define como:

$$\hat{y} = \sigma\left(\mathbf{w}^{(2)\top} \cdot \phi\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right) + b^{(2)}\right),$$

donde:

- $\mathbf{W}^{(1)} \in \mathbb{R}^{h \times p}$ es la matriz de pesos que conecta la entrada con la capa oculta,
- $\phi(\cdot)$ es la función de activación de la capa oculta (por ejemplo, ReLU, tanh o sigmoide),
- $\mathbf{w}^{(2)} \in \mathbb{R}^h$ y $b^{(2)} \in \mathbb{R}$ son los pesos y sesgo de la capa de salida,
- $\sigma(\cdot)$ es la función sigmoide, usada para producir una salida probabilística entre 0 y 1 en problemas de clasificación binaria.

El entrenamiento del modelo consiste en encontrar los parámetros $\theta = \{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{w}^{(2)}, b^{(2)}\}$ que minimicen una función de pérdida sobre los datos de entrenamiento, típicamente la entropía cruzada:

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)\right].$$

Para resolver este problema de optimización, se utiliza el algoritmo de retropropagación junto con un método de descenso por gradiente (SGD, Adam, RMSprop, entre otros), actualizando los parámetros de la red en función de los gradientes parciales de la pérdida con respecto a cada parámetro.

Una característica relevante de las redes neuronales es su capacidad para generalizar patrones complejos siempre que dispongan de suficientes datos y una adecuada regularización. Dado que son modelos altamente paramétricos, pueden incurrir en sobreajuste si no se controlan aspectos como

la profundidad de la red, el número de neuronas por capa, la tasa de aprendizaje o la presencia de ruido en los datos. Para mitigar este riesgo, se emplean técnicas como regularización L_2, *dropout*, y *early stopping*.

En la práctica, las redes neuronales han demostrado ser competitivas en la predicción del *churn* cuando se dispone de bases de datos suficientemente grandes y representativas. Si bien su interpretabilidad es limitada en comparación con modelos basados en reglas como los árboles de decisión, su capacidad de modelar relaciones complejas y no lineales las convierte en una herramienta potente para capturar señales sutiles de comportamiento en los clientes, tales como patrones de consumo, combinaciones de servicios contratados, o la interacción entre variables demográficas y contractuales.

8.4 Evaluación de modelos

En problemas de clasificación binaria como la predicción del *churn*, donde la clase positiva representa una fracción reducida del total de observaciones, la evaluación del desempeño de los modelos debe considerar cuidadosamente el desbalance de clases. En estos escenarios, métricas globales como la accuracy pueden ser engañosas, ya que un modelo que simplemente predice siempre la clase mayoritaria podría alcanzar una alta tasa de aciertos sin detectar prácticamente ningún caso de *churn* (Japkowicz y Stephen, 2002).

Para cuantificar el rendimiento, se parte del análisis de la matriz de confusión, que resume los posibles resultados de las predicciones:

Real \Predicho	No churn	Churn
No churn	TN	FP
Churn	FN	TP

donde:

- TP (*true positives*): *churn* correctamente predicho,
- FP (*false positives*): *no churn* predicho erróneamente como *churn*,
- FN (*false negatives*): *churn* no detectado por el modelo,
- TN (*true negatives*): *no churn* correctamente identificado.

A partir de esta matriz se derivan las métricas fundamentales de evaluación:

- **Accuracy**: proporción total de aciertos, incluyendo ambas clases:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}.$$

  Esta métrica puede ser útil cuando las clases están balanceadas, pero pierde valor en situaciones con alta asimetría entre clases.

- **Precision**: fracción de predicciones positivas que son efectivamente verdaderas:

$$\text{Precision} = \frac{TP}{TP + FP}.$$

- **Recall** (también llamado *sensitivity* o *true positive rate*): fracción de casos positivos reales

correctamente identificados:

$$\text{Recall} = \frac{TP}{TP + FN}.$$

- **F1-score**: promedio armónico entre *precision* y *recall*, útil para balancear ambas métricas en un solo valor:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

En problemas de *churn*, donde el objetivo principal es identificar a los clientes en riesgo, el *recall* adquiere una importancia estratégica, ya que maximizarlo permite reducir los *false negatives*, es decir, los casos en que se pierde a un cliente sin haber sido detectado a tiempo. Aunque un alto *recall* puede venir acompañado de una mayor tasa de *false positives* (bajo *precision*), este *trade-off* puede ser deseable desde una perspectiva empresarial, dado que las acciones de retención suelen tener costos marginales más bajos que la pérdida de un cliente.

Una métrica que permite evaluar el comportamiento del modelo frente a distintos criterios de clasificación es el **AUC-ROC** (*area under the receiver operating characteristic curve*). Esta curva representa cómo varían las métricas de desempeño del modelo al modificar el **umbral de decisión**, es decir, el valor a partir del cual la probabilidad predicha por el modelo se transforma en una clase positiva o negativa. Por defecto, este umbral suele ser 0,5: si la probabilidad de *churn* es mayor a 0,5, se clasifica como abandono. Sin embargo, al mover este umbral se obtienen diferentes tasas de verdaderos positivos (*TPR*) y falsos positivos (*FPR*), lo cual permite analizar el compromiso entre sensibilidad y especificidad del modelo.

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN}.$$

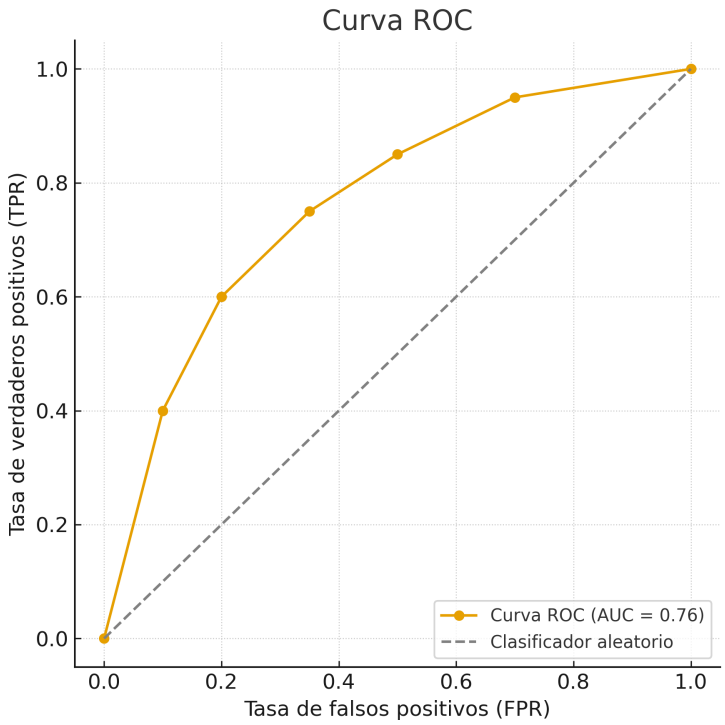

Figura 1: Curva ROC: representa el desempeño del modelo bajo distintas configuraciones del umbral de clasificación. El área bajo la curva (AUC) cuantifica la capacidad discriminativa del modelo. *Fuente: Elaboración propia.*

El área bajo la curva (AUC) resume la capacidad del modelo para distinguir entre clases. Un valor de 0,5 corresponde a un clasificador aleatorio, mientras que un valor cercano a 1 indica una excelente capacidad discriminativa. Una de sus ventajas clave es su invariancia ante el desbalance de clases, lo que la convierte en una métrica robusta para comparar modelos en contextos asimétricos. En escenarios con clases desbalanceadas, como es habitual en problemas de *churn*, resulta especialmente informativo el uso del **AUC-PR** (*Area Under the Precision–Recall Curve*), ya que se enfoca en la calidad de las predicciones positivas sin verse influido por la abundancia de verdaderos negativos. A diferencia del AUC-ROC, el AUC-PR permite evaluar directamente la capacidad del modelo para identificar correctamente a la clase minoritaria, lo que lo convierte en una métrica clave cuando el interés está centrado en detectar casos de alto riesgo o bajo volumen.

Finalmente, para evaluar el desempeño real de un modelo en datos no vistos y evitar el riesgo de sobreajuste, es fundamental emplear esquemas de validación adecuados. La **validación cruzada** k-fold es una técnica estándar que consiste en dividir el conjunto de datos en k particiones, entrenando el modelo k veces, cada vez dejando una partición distinta como conjunto de validación. La media de las métricas obtenidas en cada iteración ofrece una estimación robusta del rendimiento esperable.

Esta metodología no solo permite evaluar el modelo con mayor estabilidad estadística, sino que también ayuda a identificar modelos con alta varianza, es decir, aquellos que muestran un desempeño sobresaliente sobre el conjunto de entrenamiento, pero que fallan al generalizar. Esta diferencia entre desempeño observado y real es la manifestación típica del *overfitting*, y su mitigación es uno de los objetivos principales del proceso de validación.

En síntesis, el uso combinado de métricas como *recall*, *F1-score*, *AUC-ROC* y *AUC-PR*, junto con la validación cruzada, permite evaluar de forma robusta el desempeño de modelos en problemas de *churn*, priorizando no solo la precisión global, sino también la utilidad práctica de las predicciones en función del objetivo del negocio.

8.5 Tratamiento del desbalance de clases en la predicción de *churn*

En los problenas de clasificación binaria, como la prefición del churn, es habitual engrentar un desbalance de clases, donde los casos positivos (clientes que abandonan) representan una fracción reducida del total. Este desequilibrio puede generar modelos sesgados hacia la clase mayoritaria, disminuyendo su capacidad para identificar correctamente los abandonos (He y Garcia, 2009).

Para mitigar este problema, la literatura propone distintas estrategias. Entre las más comunes se

encuentran las técnicas de remuestreo, como el oversampling de la clase minoritaria o el undersampling de la mayoritaria, que buscan equilibrar la distribución de observaciones. También existen métodos más avanzados, como SMOTE(Synthetic Minority Over-sampling technique) (Chawla et al., 2002), que genera instancias sintéticas de la clase minoritaria a partir de sus vecinos más cercanos, reduciendo el riesgo de sobreajuste.

Otra línea de abordaje consiste en ajustar el proceso de aprendizaje mediante ponderación de clases (*class weights*), otorgando mayor penalización a los errores cometidos sobre la clase minoritaria. Finalmente, la selección de métricas adecuadas, como el Recall, el F1-Score o el AUC-PR, es esencial para evaluar correctamente el rendimiento del modelo en escenarios desbalanceados (Japkowicz y Stephen, 2002).

En conjunto, estas estrategias permiten mejorar la sensibilidad del modelo y asegurar que las predicciones sean útiles para la toma de decisiones en contextos reales de retención de clientes.

8.6 Desafíos en la predicción de *churn*

El desarrollo de modelos predictivos para identificar clientes con alta probabilidad de abandono implica una serie de desafíos tanto técnicos como prácticos, que deben ser considerados cuidadosamente en el diseño, entrenamiento y evaluación de los algoritmos. Estos desafíos no solo afectan la calidad de las predicciones, sino también su aplicabilidad efectiva en contextos operacionales reales, donde las decisiones derivadas de un modelo pueden tener implicancias económicas, éticas y comerciales.

Uno de los principales obstáculos en este tipo de problemas es el **desbalance de clases**. En la mayoría de los datasets reales de *churn*, la proporción de clientes que efectivamente abandonan es significativamente menor que la de aquellos que permanecen. Esta asimetría induce sesgos en el

aprendizaje automático, ya que los modelos tienden a favorecer la clase mayoritaria si no se aplican mecanismos específicos para contrarrestarlo. Como se discutió anteriormente, el uso de métricas como *recall*, *F1-score* y *AUC* resulta esencial para evaluar el verdadero rendimiento sobre la clase minoritaria.

Otro desafío central es el trade-off entre interpretabilidad y precisión. Modelos altamente flexibles como **redes neuronales**, **Gradient Boosting** o métodos de ensamblado tienden a ofrecer mayor capacidad predictiva, pero a costa de una menor transparencia en su lógica de decisión. Este tipo de modelos, comúnmente denominados *black-box*, pueden ser difíciles de justificar ante usuarios no técnicos o áreas comerciales que requieren comprender las razones detrás de cada predicción. Por el contrario, modelos lineales o árboles de decisión simples permiten una interpretación directa de las variables y sus pesos, pero usualmente con menor capacidad de ajuste.

Para abordar este dilema, se han desarrollado herramientas que permiten interpretar modelos complejos sin sacrificar su precisión. Destacan entre ellas los valores de **SHAP** (*SHapley Additive exPlanations*), que asignan a cada variable una contribución cuantitativa a la predicción individual de cada cliente, basándose en teoría de juegos. Otras técnicas comunes incluyen el análisis de importancia de variables (*feature importance*) y el análisis de sensibilidad, que evalúa cómo cambia la predicción ante perturbaciones en los valores de entrada. Estas herramientas son clave para alinear el modelo con la toma de decisiones informada y para detectar posibles sesgos.

La capacidad de generalización también representa un reto importante. Un modelo puede mostrar un rendimiento sobresaliente en los datos disponibles, pero deteriorarse al aplicarse en contextos operacionales con nuevas condiciones de mercado, cambios en el comportamiento del cliente o modificaciones en la oferta de servicios. Por ello, es fundamental que el desarrollo del modelo contemple validaciones rigurosas, monitorización post-despliegue, y capacidad de actualización

ante datos recientes.

Finalmente, es necesario considerar las implicancias éticas y comerciales de utilizar modelos automáticos para predecir *churn*. Por un lado, los errores del modelo pueden implicar decisiones de negocio inadecuadas, como ofrecer beneficios a clientes que no estaban en riesgo o ignorar señales tempranas de abandono. Por otro lado, el uso de variables sensibles (como edad, género o ubicación) podría dar lugar a prácticas discriminatorias si no se controlan adecuadamente los sesgos algorítmicos. En este sentido, la transparencia, la explicabilidad y el control de sesgos no solo son deseables desde un punto de vista técnico, sino también necesarios para asegurar una implementación responsable.

En conjunto, estos desafíos exigen un enfoque equilibrado entre desempeño, interpretabilidad y robustez, permitiendo que los modelos predictivos de *churn* no solo alcancen buenos resultados cuantitativos, sino que también sean útiles, confiables y éticamente aceptables en su aplicación práctica.

9 Metodología

La metodología de esta investigación se fundamenta en el enfoque CRISP-DM (Cross-Industry Standard Process for Data Mining), ampliamente reconocido como uno de los marcos más robustos y versátiles para el desarrollo de proyectos de ciencia de datos. Este modelo define un proceso iterativo compuesto por seis fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelado, evaluación y despliegue. Su estructura ha sido valorada por su independencia de herramientas específicas, aplicabilidad transversal a múltiples dominios, y claridad en la ejecución paso a paso (Wirth y Hipp, 2000).

A más de dos décadas desde su formulación, CRISP-DM continúa siendo un estándar de facto tanto en la industria como en la academia, gracias a su adaptabilidad a contextos actuales de analítica avanzada, inteligencia artificial y aprendizaje automático. Entre sus principales fortalezas se destacan la trazabilidad del proceso analítico, la replicabilidad de los resultados y la alineación explícita con los objetivos estratégicos del negocio.

En el contexto de este estudio, se adoptará el enfoque CRISP-DM para estructurar de forma metódica el análisis predictivo del abandono de clientes (*churn*), utilizando un conjunto de datos preexistente. Cabe señalar que la etapa de despliegue será excluida del proceso metodológico, dado que el alcance de la investigación es comparativo y exploratorio, y no contempla la implementación de soluciones productivas.

A continuación, se presenta el esquema metodológico propuesto para guiar el desarrollo del estudio:

Una de las principales ventajas de CRISP-DM es su capacidad para separar claramente el análisis de datos del entendimiento del problema de negocio, permitiendo que ambas dimensiones avancen

en paralelo pero coordinadamente. En el caso específico del *churn*, donde las decisiones estratégicas y operativas pueden depender directamente de las predicciones generadas por el modelo, esta característica es especialmente valiosa.

Figura 2: Diagrama que muestra el proceso de CRISP-DM. *Fuente: Wikipedia.*

9.1 Fases de la metodología

1. **Comprensión del negocio:** Se comienza por contextualizar el fenómeno de abandono de clientes en la industria de telecomunicaciones, estableciendo la relevancia económica y operativa del problema. Se plantea el objetivo predictivo y se determinan las restricciones o necesidades del negocio que condicionan el desarrollo del modelo, tales como la interpretabilidad, el tiempo de inferencia o el tipo de intervención asociada a los resultados.

2. **Comprensión de los datos:** En esta fase se caracterizará el dataset **IBM Telco Customer Churn**, identificando la cantidad de observaciones, el tipo y naturaleza de las variables, y la distribución de la variable objetivo. Se realizará un análisis exploratorio de datos (EDA) para

detectar patrones, outliers, y relaciones entre variables, así como para dimensionar el grado de desbalance de clases existentes.

3. **Preparación de los datos:** Esta etapa contempla la limpieza y transformación del dataset con el fin de optimizar su calidad para el modelado. Se aplicarán técnicas como el tratamiento de valores nulos, codificación de variables categóricas, escalamiento de variables numéricas, y la generación de nuevas variables derivadas. Además, se abordará el problema de desbalance de clases, evaluando estrategias como *undersampling*, *oversampling* o técnicas sintéticas como **SMOTE**.

4. **Entrenamiento y selección de modelos:** Se entrenarán distintos modelos de clasificación binario, siguiendo una lógica de complejidad creciente: desde modelos lineales (como regresión logística), hasta métodos más avanzados como **árboles de decisión**, **random forest**, **XGBoost**, **LightGBM**, **CatBoost** y **redes neuronales artificiales**. La selección de algoritmos estará motivada tanto por su desempeño esperado como por su capacidad de interpretación y generalización.

5. **Evaluación del desempeño:** Se aplicará validación cruzada estratificada (k-fold) para obtener estimaciones robustas del rendimiento de cada modelo. Las métricas a utilizar serán especialmente sensibles al desbalance de clases: **recall**, **F1-score**, **área bajo la curva ROC (AUC-ROC)** y matriz de confusión. Se priorizará la capacidad del modelo para detectar correctamente a los clientes en riesgo de abandono, maximizando el recall sin comprometer excesivamente la precisión.

6. **Interpretabilidad de resultados:** Una vez entrenados los modelos, se analizarán las varia-

bles más influyentes en la predicción del churn mediante métodos como **feature importance**, **coeficiente estimados (en modelos lineales)** y **Valores SHAP**. Esto permitirá explicar el comportamiento del modelo tanto a nivel global como a nivel individual, facilitando la interpretación de sus decisiones desde una perspectiva estratégica.

9.2 Herramientas y entorno de desarrollo

El desarrollo del estudio se realizará en lenguaje de programación **Python**, debido a su amplia adopción en la comunidad de ciencia de datos y su ecosistema robusto de librarías orientadas a análisis estadístico, modelamiento predictivo y visualización de datos.

El análisis será implementado en el entorno Jupyter Notebook, utilizando las siguientes librerías;

- **Pandas** y **NumPy** para manipulación y análisis de datos.
- **Matplotlib** y **Seaborn** para visualización exploratoria.
- **Scikit-learn** para modelamiento predictivo, validación cruzada y métricas de evaluación.
- **Imbalanced-learn** para el tratamiento del desbalance de clases (SMOTE, undersampling).
- **XGBoost**, **LightGBM** y **CatBoost** como algoritmos de ensamble.
- **SHAP** para la interpretación de modelos complejos mediante valores de contribución por variable.

Todo el desarrollo se ejecutará en un entorno local con las siguientes especificaciones:

- **Sistema operativo:** Windows 11 64-bits
- **Procesador:** Intel(R) CORE(TM) i7-10750H CPU @ 2.60 GHz (2.59 GHz)

- **Memoria RAM:** 16 GB

- **Versión de Python:** 3.10.18

10 Resultados

10.1 Análisis Exploratorio de Datos (EDA)

El análisis exploratorio de datos tiene como objetivo caracterizar el conjunto de datos utilizado en este estudio, identificar patrones generales, evaluar la calidad de los datos y detectar posibles problemas como desbalance de clases, valores atípicos o relaciones no evidentes entre variables. Este análisis preliminar es esencial para fundamentar las decisiones de preparación y modelamiento posterior.

El dataset utilizado corresponde a **IBM Telco Customer Churn**, el cual contiene información de clientes de una compañía de telecomunicaciones, incluyendo características demográficas, servicios contratados, historial de facturación y una etiqueta binaria que indica si el cliente abandonó el servicio (*churn*).

Dimensiones y estructura del dataset

El dataset esta compuesto por 7.043 registros correspondientes a clientes individuales y 20 variables predictoras y una variable objetivo (*churn*). Las variables incluyen tanto atributos categóricos como numéricos. Un resumen general se presenta a continuación:

Variable	Valores no-nulos	Tipo de variable
customerID	7043 non-null	object
gender	7043 non-null	object
SeniorCitizen	7043 non-null	int64
Partner	7043 non-null	object
Dependents	7043 non-null	object
tenure	7043 non-null	int64
PhoneService	7043 non-null	object
MultipleLines	7043 non-null	object
InternetService	7043 non-null	object
OnlineSecurity	7043 non-null	object
OnlineBackup	7043 non-null	object
DeviceProtection	7043 non-null	object
TechSupport	7043 non-null	object
StreamingTV	7043 non-null	object
StreamingMovies	7043 non-null	object
Contract	7043 non-null	object
PaperlessBilling	7043 non-null	object
PaymentMethod	7043 non-null	object
MonthlyCharges	7043 non-null	float64
TotalCharges	7043 non-null	object
Churn	7043 non-null	object

Tabla 1: Resumen de variables incluidas en el dataset y tipo de dato. *Fuente: Elaboración Propia.*

A continuación, se presenta una descripción general del significado de cada variable contenida en el dataset.

- **customerID:** Identificador único de cada cliente, sin valor predictivo.
- **gender:** Género del cliente (masculino o femenino).
- **SeniorCitizen:** Indica si el cliente es adulto mayor (1 = sí, 0 = no).
- **Partner:** Si el cliente tiene pareja.
- **Dependents:** Cuantas personas tiene el cliente a su cargo.
- **tenure:** Tiempo (en meses) que el cliente ha permanecido en la compañia.

- **PhoneService:** Indica si el cliente tiene servicio telefónico.
- **MultipleLines:** Si posee más de una línea telefónica.
- **InternetService:** Tipo de conexión a Internet (DSL, Fibre Óptica o No internet).
- **OnlineSecurity:** Si el cliente tiene contratado un servicio de seguridad en línea.
- **OnlineBackup:** Si el cliente tiene servicio de respaldo en línea.
- **DeviceProtection:** Si el cliente cuenta con protección de dispositivos.
- **TechSupport:** Si el cliente dispone de soporte técnico.
- **StreamingTV:** Si el cliente posee servicio de televisión en streaming.
- **StreamingMovies:** Si el cliente posee servicio de películas en streaming.
- **Contract:** Tipo de contrato del cliente (mensual, anual o bianual).
- **PaperlessBilling:** Si utiliza facturación electrónica.
- **PaymentMethod:** Método de pago del cliente (cheque electrónico, tarjeta, transferencia, etc.).
- **MonthlyCharges:** Cargo mensual promedio del cliente.
- **TotalCharges:** Monto total facturado durante todo el periodo de relación.
- **Churn:** Variable objetivo; indica si el cliente abandonó el servicio (1 = churn, 0 = permanece).

Preparación inicial del dataset

Antes de proceder con la visualización y análisis exploratorio de datos, se realizó una preparación preliminar con el objetivo de asegurar la integridad y consistencia del conjunto de datos. En primer lugar, se eliminó la variable *customerID*, dado que corresponde a un identificador único sin valor predictivo, y su inclusión podría introducir ruido o redundancia en el modelado posterior (Kuhn y Johnson, 2013).

Adicionalmente, se identificaron discrepancias en el tipo de datos de la variable *TotalCharges*. Esta se encontraba en el formato *object*, lo cual impide su tratamiento como variable numérica y puede generar errores en los procesos de análisis estadístico o de aprendizaje automático. Al intentar convertirla al tipo *float*, se detectó la presencia de 11 valores vacíos representados como espacios en blanco. Al intentar convertir la variable TotalCharges al tipo numérico, se detectó la presencia de valores vacíos representados como espacios en blanco. Estos registros correspondían a clientes con tenure = 0, es decir, usuarios recién incorporados que aún no registraban facturación. Dado que en estos casos no existe un valor real de cobro que pueda imputarse de manera coherente, y considerando además que representaban un porcentaje marginal del total de observaciones, se optó por eliminarlos. Esta decisión se alinea con las buenas prácticas de tratamiento de datos faltantes cuando los valores ausentes no son atribuibles al azar ni poseen un referente lógico para su estimación (Little y Rubin, 2019).

Visualización de datos

Como primer paso del análisis, se exploró la distribución de la variable objetivo *churn*, que indica si un cliente abandonó (*Yes*) o no (*No*) el servicio. Para ello, se construyeron dos repre-

sentaciones gráficas: un gráfico de distribución porcentual y un gráfico de barras de frecuencias absolutas.

Los resultados muestran un marcado desbalance de clases, donde aproximadamente el 26,6 % de los clientes corresponde a la clase positiva y el 73,4 % corresponde a la clase negativa. Este desequilibrio es característico en problemas de abandono de clientes y representa un desafío metodológico relevante, ya que muchos algoritmos de clasificación tienden a favorecer a la clase mayoritaria, afectando negativamente la sensibilidad del modelo (He y Garcia, 2009). Por esta razón, se tomará especial precaución en las etapas posteriores de modelado y evaluación, utilizando métricas robustas al desbalance e implementando estrategias de balanceo en el preprocesamiento.

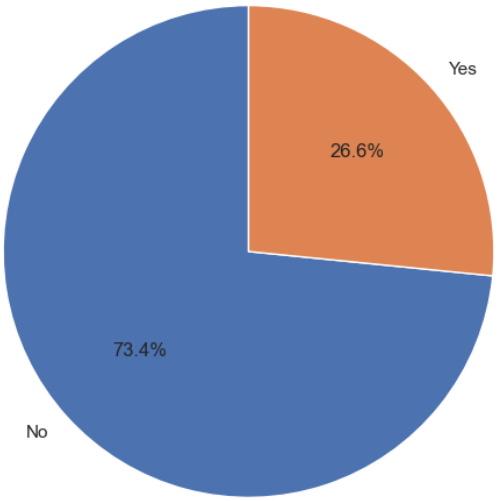

Figura 3: Distribución de *churn* en porcentaje. *Fuente: Elaboración propia.*

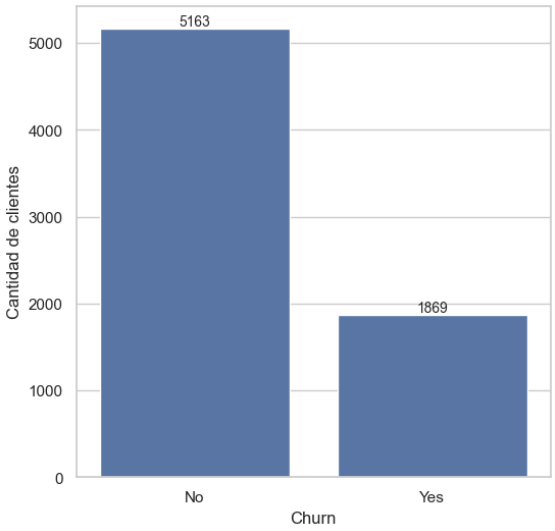

Figura 4: Distribución de *churn* en valores absolutos. *Fuente: Elaboración propia.*

Análisis de variables numéricas

Para examinar la distribución de las variables numéricas, se utilizaron gráficos de estimación de densidad mediante núcleos (*Kernel Density Estimation*, KDE), una técnica que permite visualizar la forma de la distribución de variables continuas de forma suavizada. Esta aproxima-

ción facilita la identificación de patrones relevantes como asimetrías, concentración de datos o presencia de múltiples modos (Silverman, 1986), particularmente útil en problemas de clasificación.

Los gráficos fueron generados para las variables TENURE, MONTHLYCHARGES y TOTALCHARGES, diferenciando entre clientes que abandonaron y aquellos que permanecieron, lo que permite explorar visualmente el comportamiento de estas variables según la clase objetivo.

- En la variable TENURE, se observa que los clientes que abandonan tienden a concentrarse en los primeros meses de relación con la compañía. Esto sugiere que el abandono ocurre predominantemente en etapas tempranas del ciclo de vida del cliente, mientras que los clientes que permanecen presentan una distribución más uniforme e incluso bimodal, con una alta densidad hacia los extremos de tiempo de permanencia.

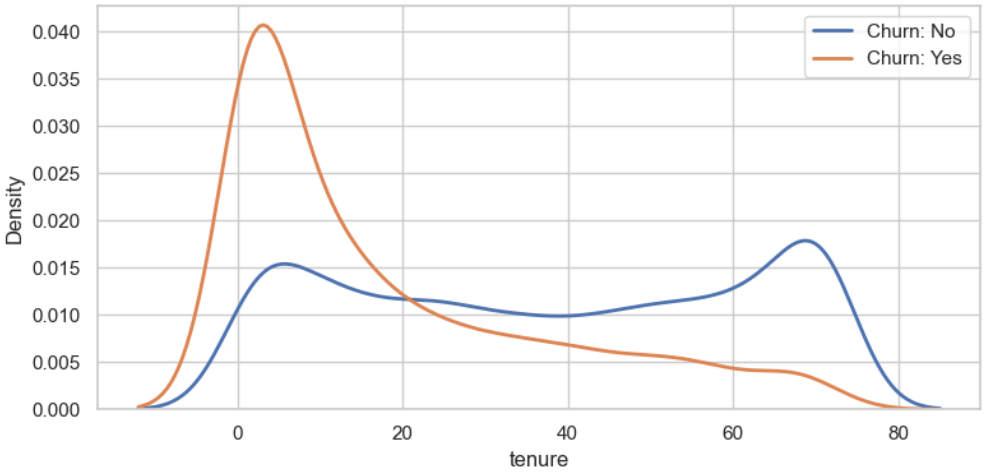

Figura 5: Gráfico KDE para la variable TENURE. *Fuente: Elaboración propia.*

- En cuanto a MONTHLYCHARGES, los clientes que abandonan se concentran en el rango de cargos mensuales medios a altos, con una densidad notable entre los 70 y 100 USD. Por otro lado, los clientes que no abandonan presentan una mayor densidad en rangos más bajos,

particularmente bajo los 30 USD.

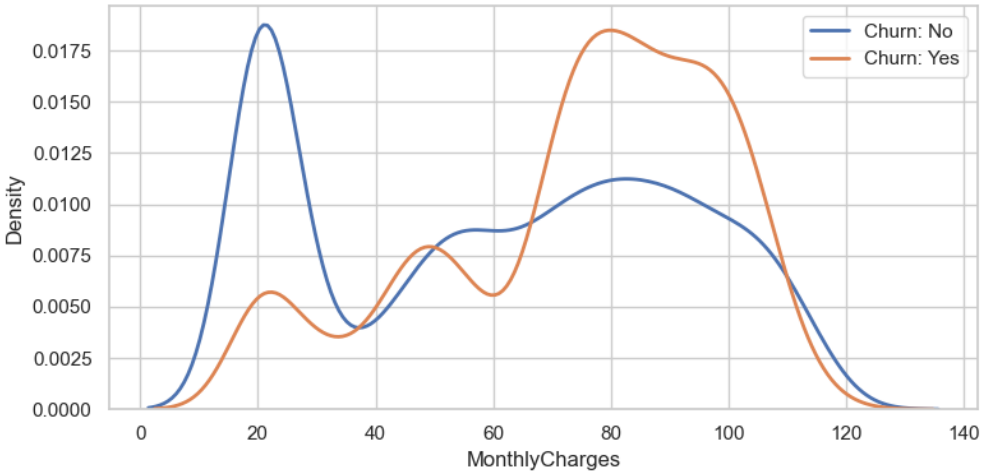

Figura 6: Gráfico KDE para la variable MONTHLYCHARGES. *Fuente: Elaboración propia.*

- Finalmente, la variable TOTALCHARGES muestra que los clientes que abandonan tienen, en su mayoría, bajos niveles acumulados de facturación. Los clientes que no abandonan presentan una distribución más dispersa y extendida.

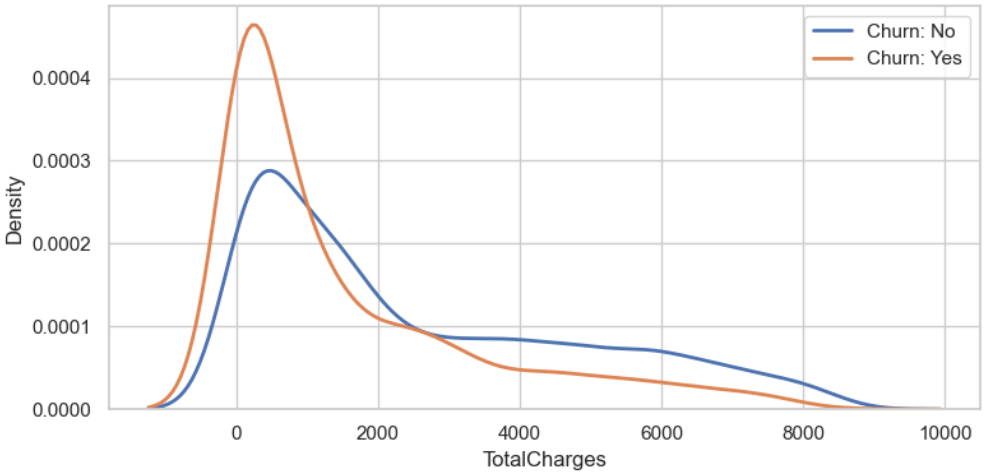

Figura 7: Gráfico KDE para la variable TOTALCHARGES. *Fuente: Elaboración propia.*

Con el objetivo de profundizar el análisis de las variables numéricas y evaluar posibles diferencias significativas entre los grupos de clientes que abandonaron y los que permanecieron, se incorpora-

ron gráficos de tipo *boxplot* para las variables TENURE, MONTHLYCHARGES y TOTALCHARGES. Los *boxplots* permiten observar la distribución central, la dispersión y la presencia de valores atípicos, siendo herramientas útiles para comparar estadísticamente la variabilidad entre grupos (McGill et al., 1978).

- En el caso de TENURE, se evidencia una diferencia clara entre grupos. Los clientes que abandonan el servicio presentan una mediana de permanencia considerablemente más baja, con un rango intercomunicador (IQR) más estrecho y concentración en los primeros meses. En cambio, quienes permanecen muestran mayor dispersión y una mediana cercana a los 40 meses, lo que reafirma que el abandono ocurre mayoritariamente durante los primeros períodos de relación contractual.

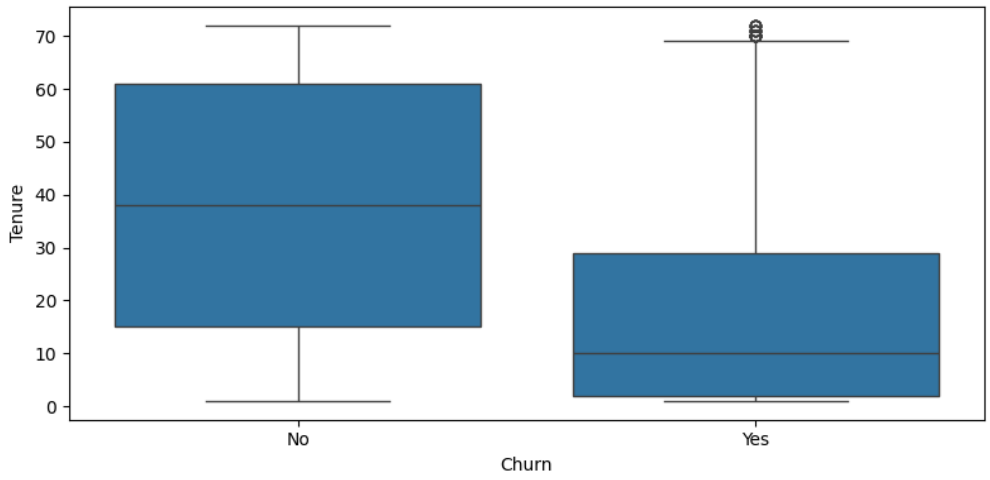

Figura 8: Diagrama de caja y bigotes (*Boxplot*) para la variable TENURE. *Fuente: Elaboración propia.*

- Respecto a MONTHLYCHARGES, los clientes que abandonan el servicio presentan una mediana más alta que los que no abandonan, con una distribución menos simétrica. Este comportamiento sugiere que tarifas mensuales más elevadas podrían estar asociadas a una mayor

probabilidad de churn, posiblemente por percepción de alto costo o falta de satisfacción en relación al servicio entregado.

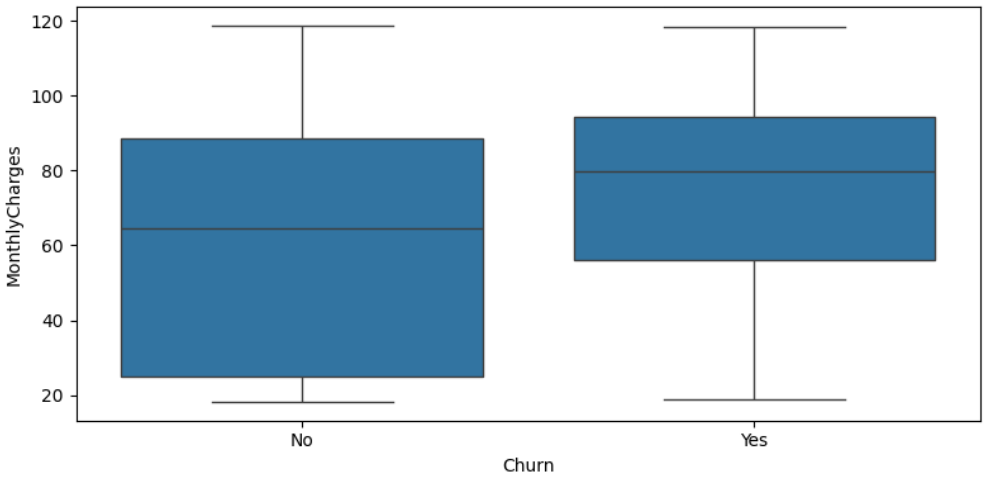

Figura 9: Diagrama de caja y bigotes (*Boxplot*) para la variable MONTHLYCHARGES. *Fuente: Elaboración propia.*

- En el boxplot de TOTALCHARGES se observa que los clientes que abandonan presentan valores más bajos y una menor dispersión, mientras que quienes permanecen muestran una distribución más amplia y medianas superiores. En este último grupo se identifican valores atípicos hacia los rangos altos, lo que podría asociarse a clientes con contratos de larga duración o con múltiples servicios contratados.

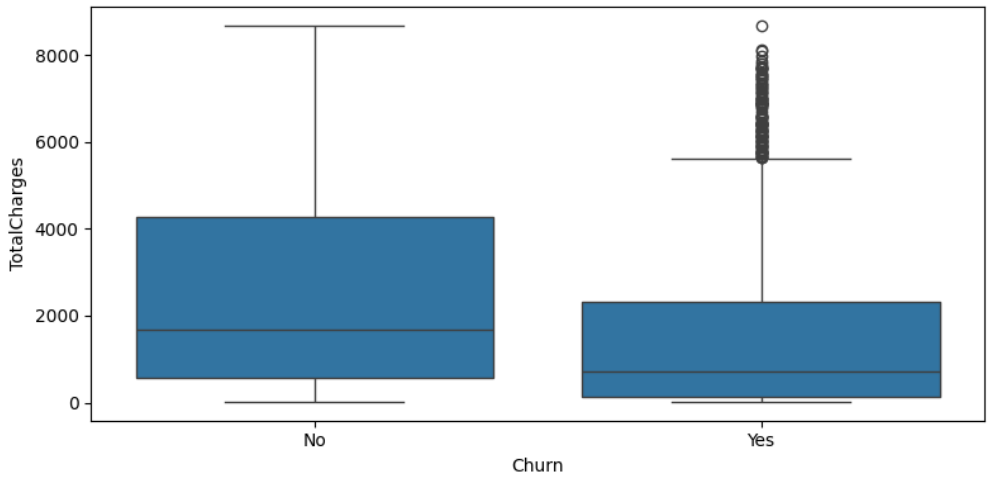

Figura 10: Diagrama de caja y bigotes (*Boxplot*) para la variable TOTALCHARGES. *Fuente: Elaboración propia.*

Con el fin de comprender el patrón de permanencia de los clientes en la compañía, se analizó la distribución de la variable TENURE, la cual representa el tiempo en meses que un cliente ha mantenido su relación contractual. Para ello, se generó un histograma que muestra la frecuencia absoluta de clientes para cada valor de *tenure*.

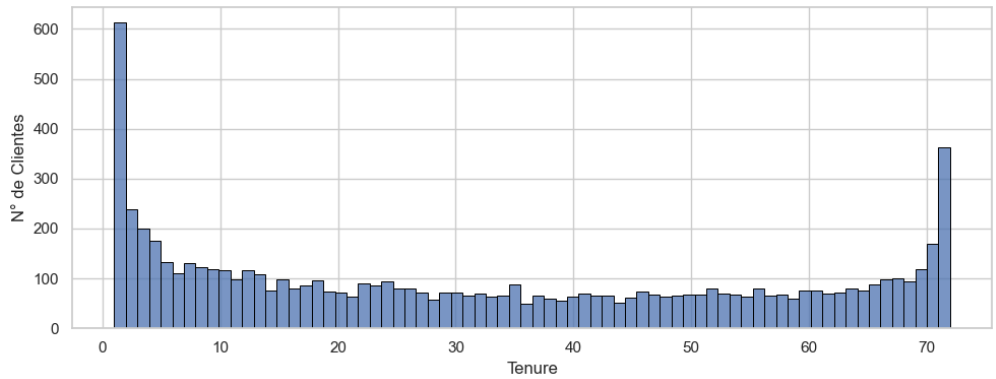

Figura 11: Número de clientes según variable TENURE. *Fuente: Elaboración Propia.*

La figura (11) revela tres concentraciones notorias:

1. Un pico en el mes 1, correspondiente a clientes muy recientes, posiblemente en periodo de prueba o en las primeras facturaciones.

2. Una distribución relativamente uniforme en los valores intermedios, con una ligera tendencia descendente en clientes de permanencia media.

3. Un pico final en torno al mes 72, que refleja clientes de largo plazo, probablemente con contratos extendidos o alta fidelización.

Este patrón sugiere la coexistencia de dos segmentos muy diferenciados: clientes que abandonan rápidamente y clientes altamente fidelizados, con una franja intermedia más dispersa. La identificación de estas dinámicas es clave, ya que el tiempo de permanencia suele ser un predictor robusto del churn en industrias de suscripción y servicios (Ahn et al., 2006).

Análisis de variables categóricas

Para evaluar la posible relación entre las variables categóricas y el *churn*, se generaron gráficos de barras segmentados por categoría. El objetivo fue identificar patrones que pudieran explicar diferencias en el comportamiento de retención.

- CONTRACT: En la figura (12) se observa que los clientes con contrato "Month-to-month" presentan una proporción de abandono significativamente mayor que aquellos con contratos de mayor duración. Esto sugiere que los contratos más largos actúan como un factor de retención, posiblemente por compromisos contractuales o beneficios asociados.

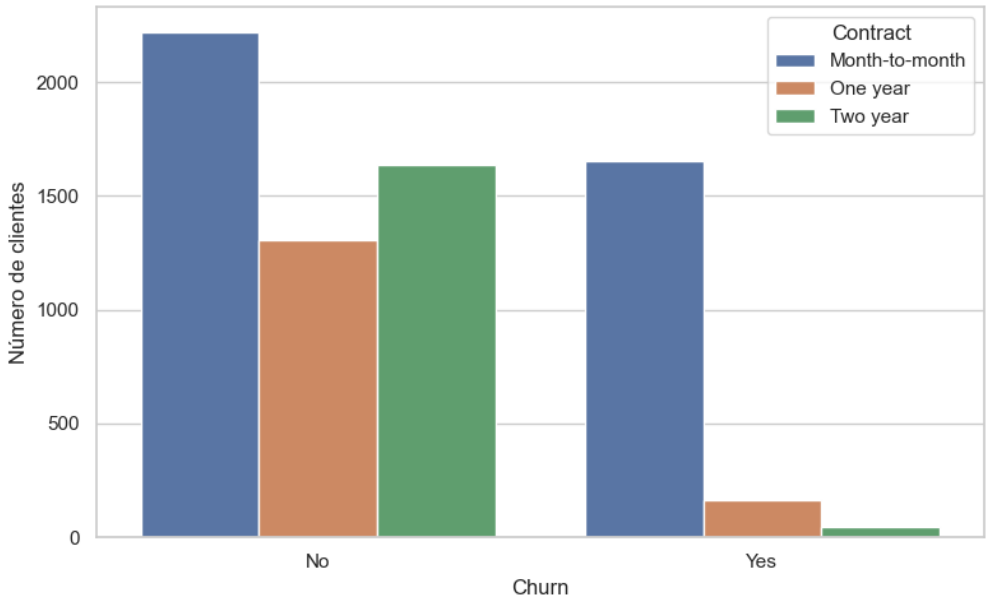

Figura 12: Distribución del churn según variable contract. *Fuente: Elaboración propia.*

- DEPENDENTS: En la figura (13) se observa la presencia de dependientes parece asociarse a una menor tasa de abandono. Los clientes sin dependientes muestran una mayor proporción de *churn*, lo que podría vincularse a diferencias en estabilidad financiera, uso del servicio o compromiso con el proveedor.

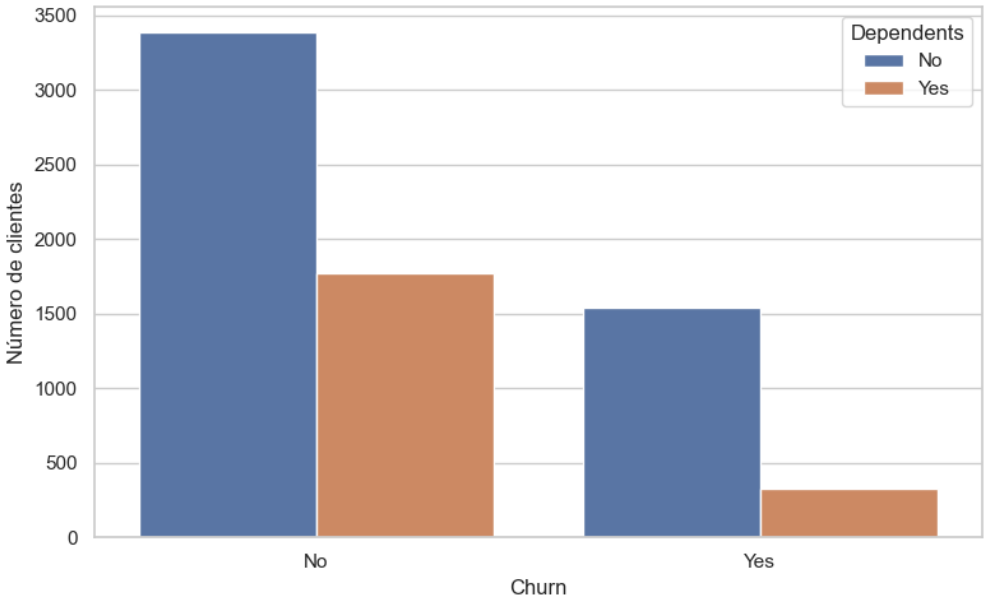

Figura 13: Distribución del churn según variable Dependents. *Fuente: Elaboración propia.*

- DEVICE PROTECTION: En la figura (14) se observa que los clientes que cuentan con protección de dispositivo presentan una menor tasa de abandono que aquellos que no la tienen. La categoría "No internet service" muestra niveles bajos de churn, probablemente porque corresponde a un grupo distinto de usuarios con necesidades más limitadas de conectividad.

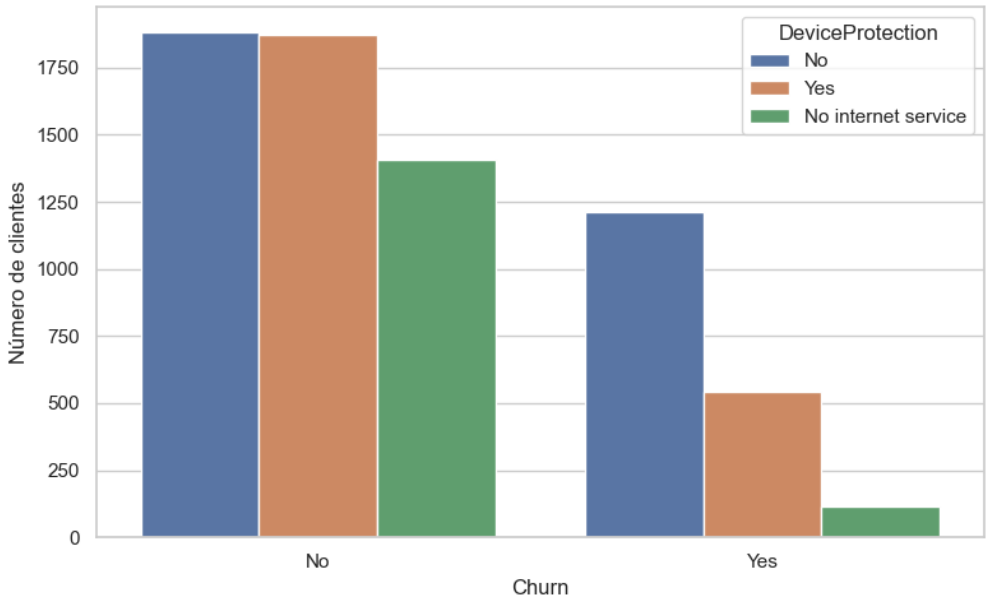

Figura 14: Distribución del churn según variable DeviceProtection. *Fuente: Elaboración propia.*

- GENDER: En la figura (15) no se evidencian diferencias significativas en la tasa de abandono según género. Las proporciones de churn son similares entre hombres y mujeres, lo que indica que el género no es un factor determinante en este dataset.

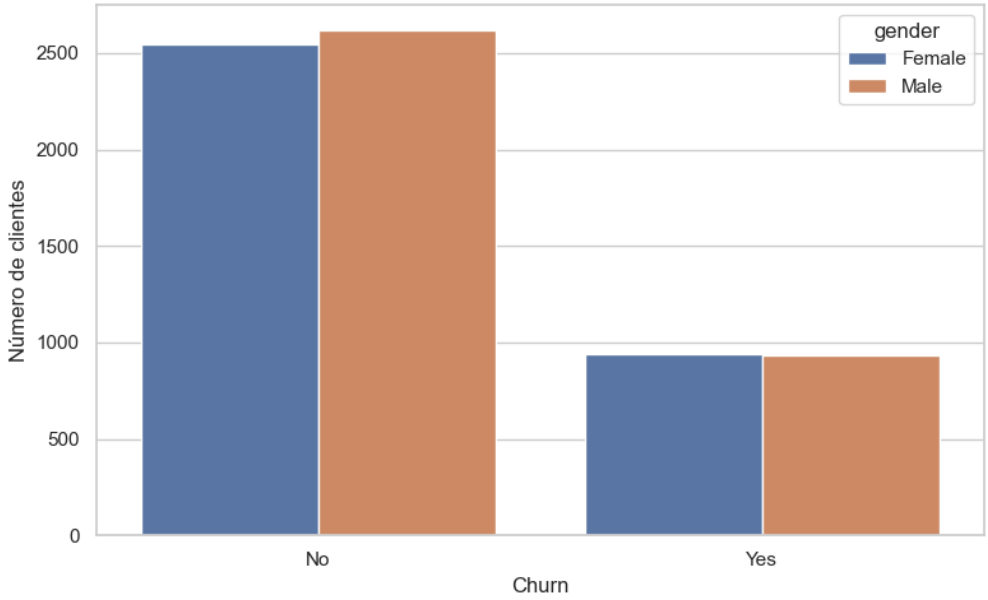

Figura 15: Distribución del churn según variable gender. *Fuente: Elaboración propia.*

- INTERNET SERVICE: En la figura (16) se observa que el tipo de servicio de internet tiene un impacto notable: los clientes con fibra óptica exhiben una tasa de abandono considerablemente mayor que los de DSL. Esto podría deberse a costos más altos, experiencias de servicio o competencia más intensa en este segmento.

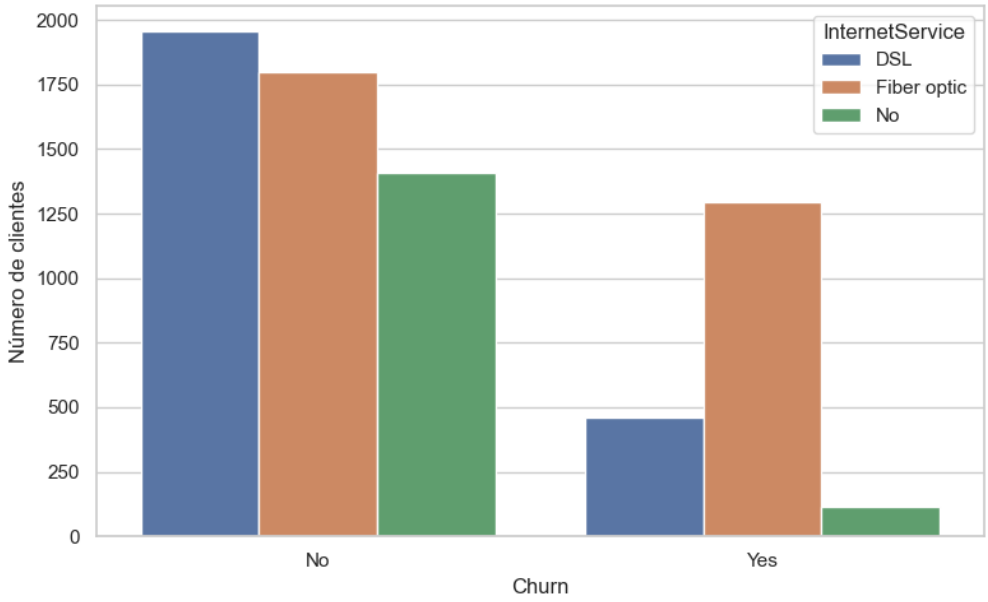

Figura 16: Distribución del churn según variable InternetService. *Fuente: Elaboración propia.*

- MULTIPLELINES: En la figura (17) se observa que los clientes sin servicio telefónico muestran una baja incidencia de churn. Entre quienes sí tienen servicio telefónico, la tasa de abandono es similar, independientemente de si poseen múltiples líneas o no, lo que sugiere que esta variable no presenta un efecto fuerte por sí sola.

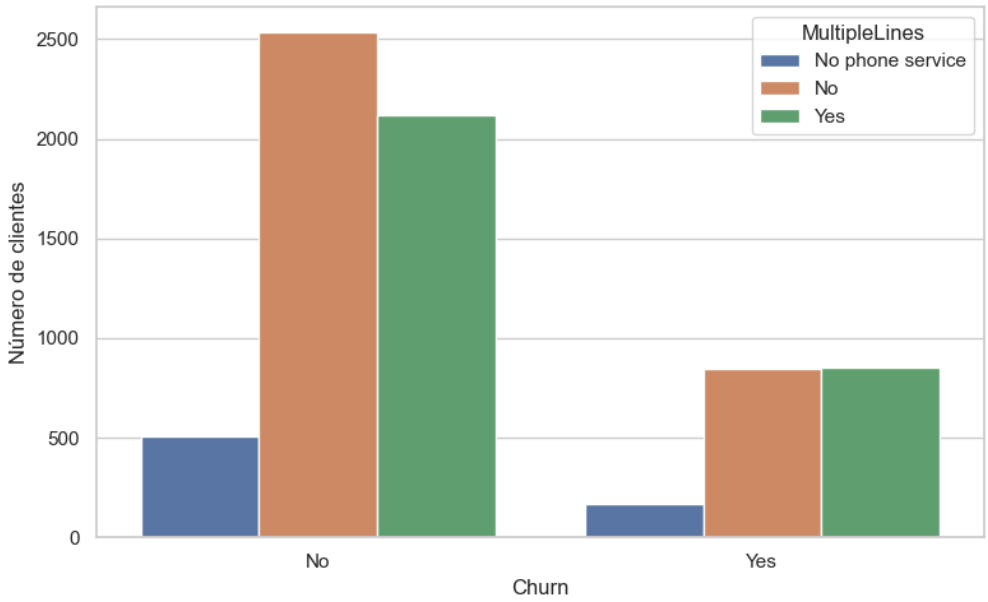

Figura 17: Distribución del churn según variable MultipleLines. *Fuente: Elaboración propia.*

- ONLINEBACKUP: En la figura (18) se observa que los clientes que no cuentan con servicio de respaldo en línea presentan una tasa de abandono considerablemente mayor que aquellos que sí lo tienen. La categoría "No internet service"muestra niveles muy bajos de *churn*, lo que es consistente con un grupo de usuarios menos expuestos a la pérdida de datos o a la necesidad de servicios complementarios.

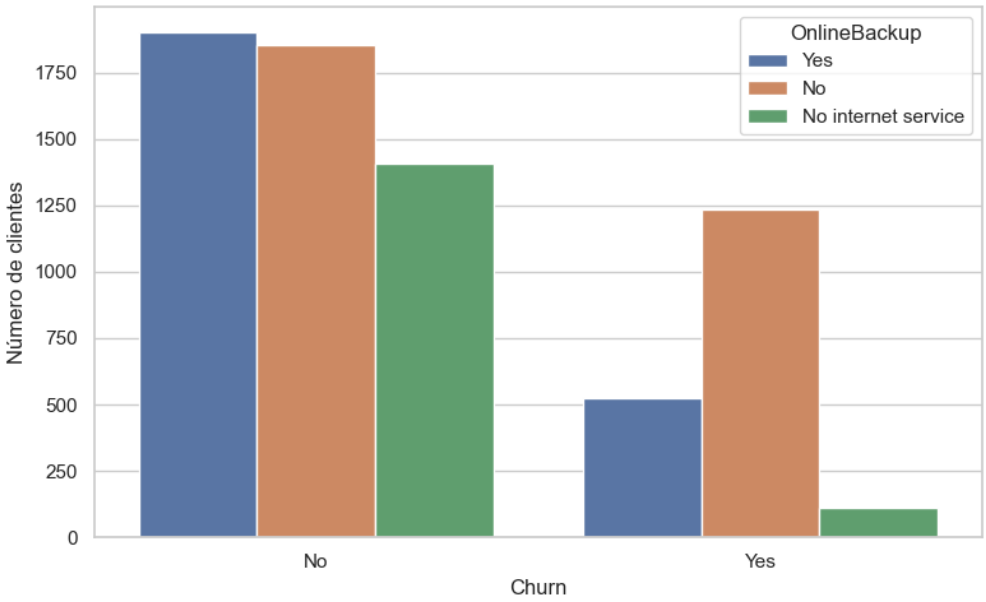

Figura 18: Distribución del churn según variable OnlineBackup. *Fuente: Elaboración propia.*

- ONLINESECURITY: En la figura (19) se observa que los clientes sin servicio de seguridad en línea presentan una tasa de *churn* marcadamente superior, mientras que quienes cuentan con este servicio muestran una menor propensión al abandono. Al igual que en el caso anterior, "No internet service" mantiene una incidencia muy reducida de churn, probablemente por pertenecer a un segmento con necesidades de conectividad limitadas.

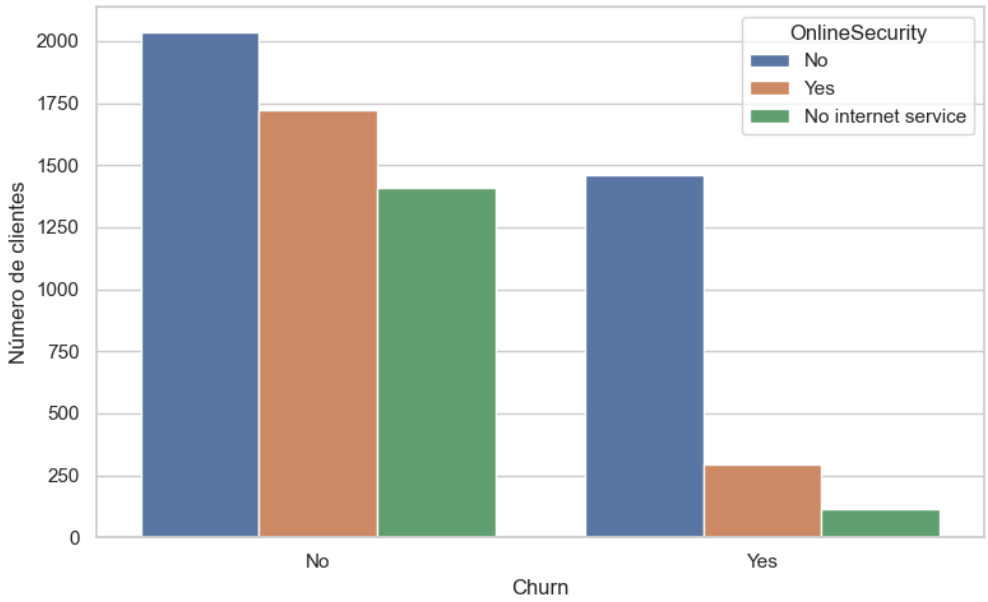

Figura 19: Distribución del churn según variable OnlineSecurity. *Fuente: Elaboración propia.*

- PAPERLESSBILLING: En la figura (20) se observa que los clientes que utilizan facturación electrónica presentan una tasa de churn más elevada que aquellos que reciben facturación física. Este resultado podría reflejar una mayor concentración de usuarios con contratos de corta duración o menor antigüedad, aunque esta relación no se confirma directamente con los datos disponibles.

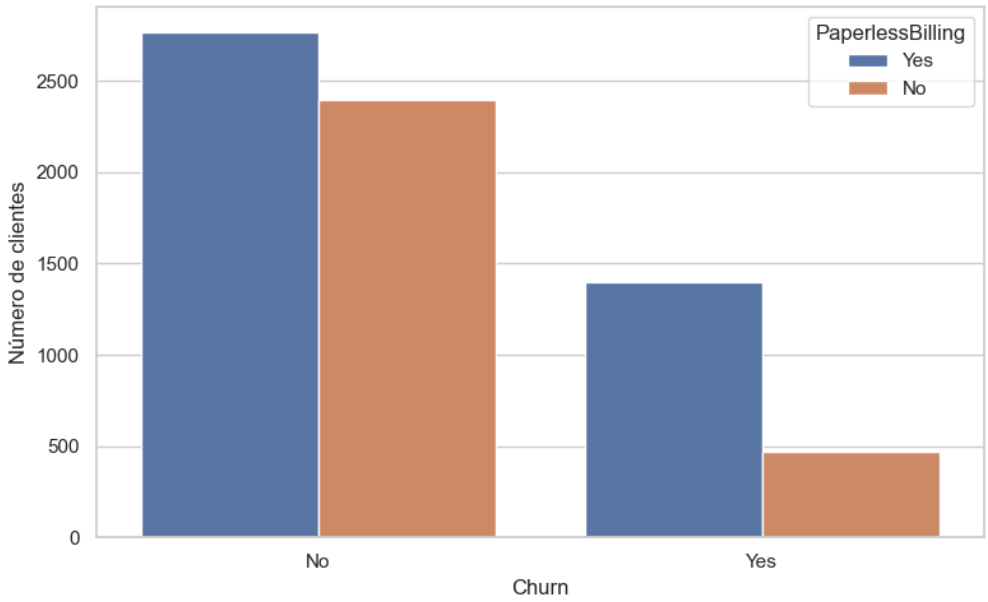

Figura 20: Distribución del churn según variable PaperlessBilling. *Fuente: Elaboración propia.*

- PARTNER: En la figura (21) se observa que los clientes sin pareja muestran una proporción de abandono superior a los que tienen pareja. Este hallazgo es consistente con la hipótesis de que la estabilidad familiar o de pareja puede estar asociada a una mayor fidelidad al servicio.

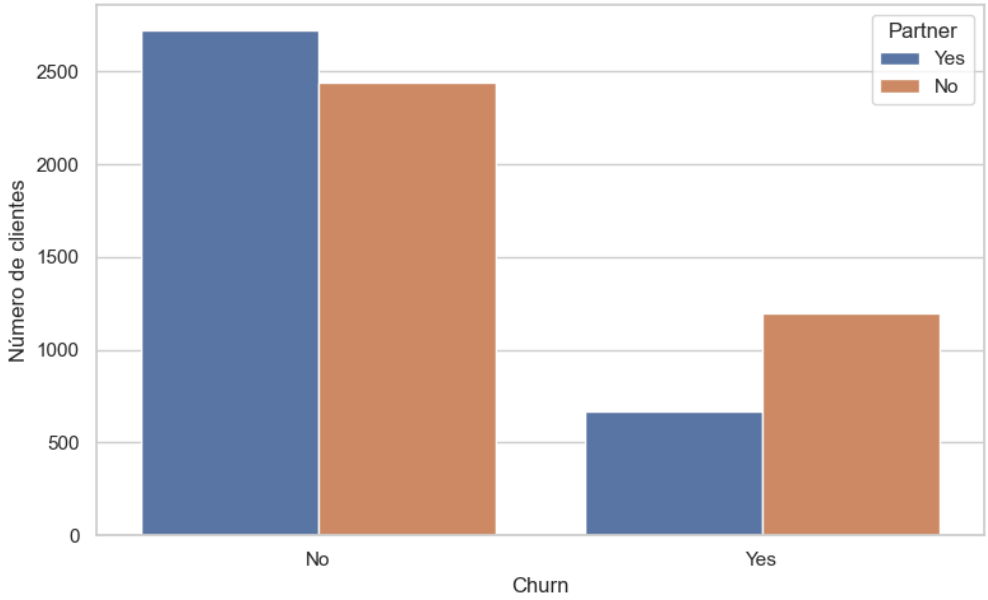

Figura 21: Distribución del churn según variable Partner. *Fuente: Elaboración propia.*

- PAYMENTMETHOD: En la figura (22) se observa que el método de pago parece influir en la tasa de churn: el pago mediante "Electronic check" registra los niveles más altos de abandono, mientras que los métodos automáticos como "Bank transfer" o "Credit card" muestran menor churn. Esto podría deberse a la facilidad y automatización de los pagos recurrentes, que reducen la probabilidad de cancelación.

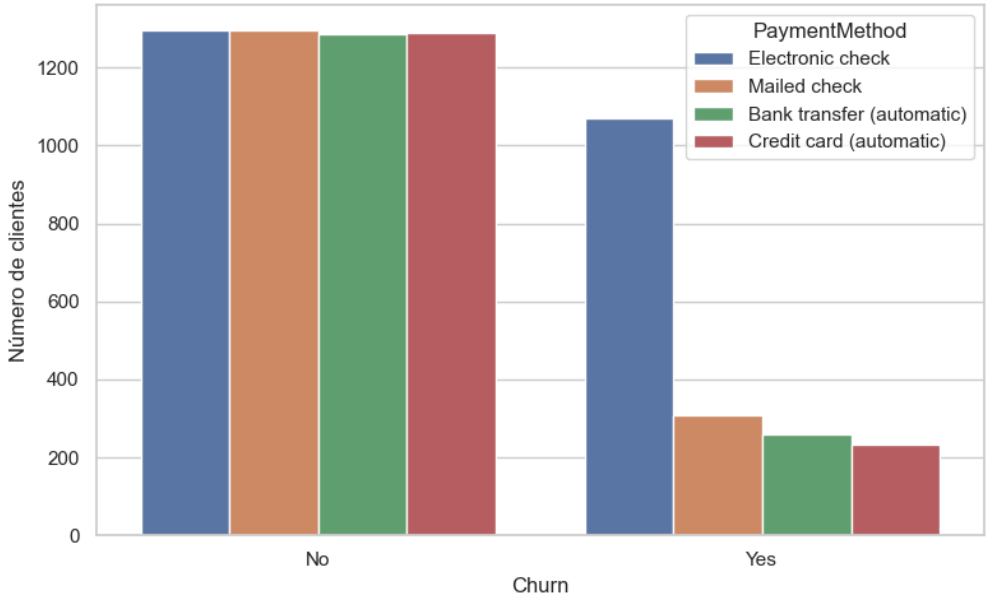

Figura 22: Distribución del churn según variable PaymentMethod. *Fuente: Elaboración propia.*

- PHONESERVICE: En la figura (23) se observa que los clientes sin servicio telefónico presentan una tasa de abandono muy baja. Entre aquellos que cuentan con este servicio, la tasa de churn es moderada, lo que sugiere que la presencia de línea telefónica no constituye un factor determinante por sí solo en la decisión de abandonar el servicio.

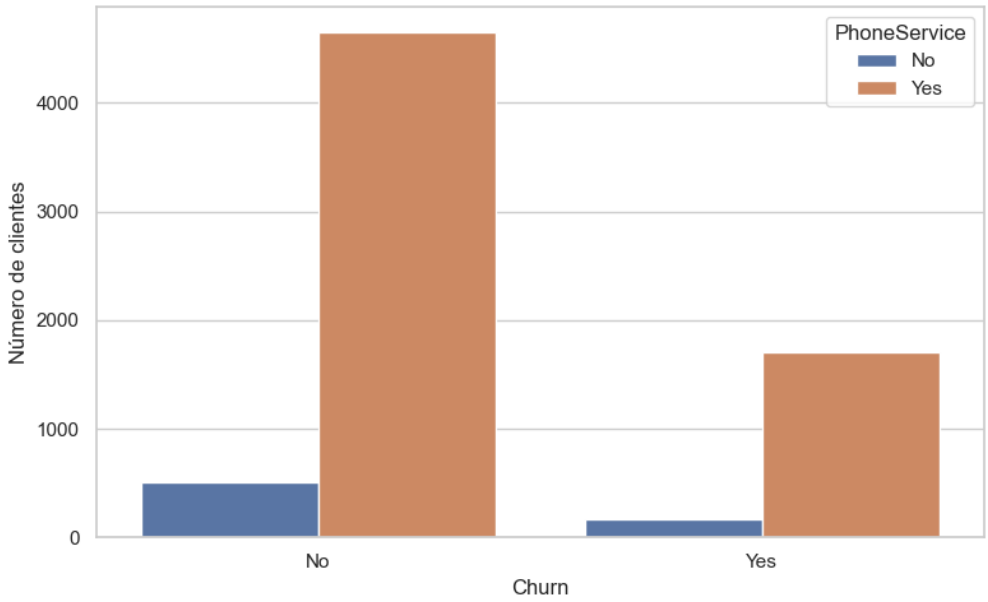

Figura 23: Distribución del churn según variable PhoneService. *Fuente: Elaboración propia.*

- SENIORCITIZEN: En la figura (24) se observa que los clientes clasificados como adultos mayores (SeniorCitizen = 1) presentan una proporción de abandono más alta que los clientes más jóvenes. Sin embargo, la mayoría de la base corresponde a no adultos mayores, por lo que este grupo concentra un menor volumen absoluto de churn. Este patrón sugiere que la edad podría ser un factor de riesgo moderado, aunque su impacto debe analizarse en conjunto con otras variables socioeconómicas.

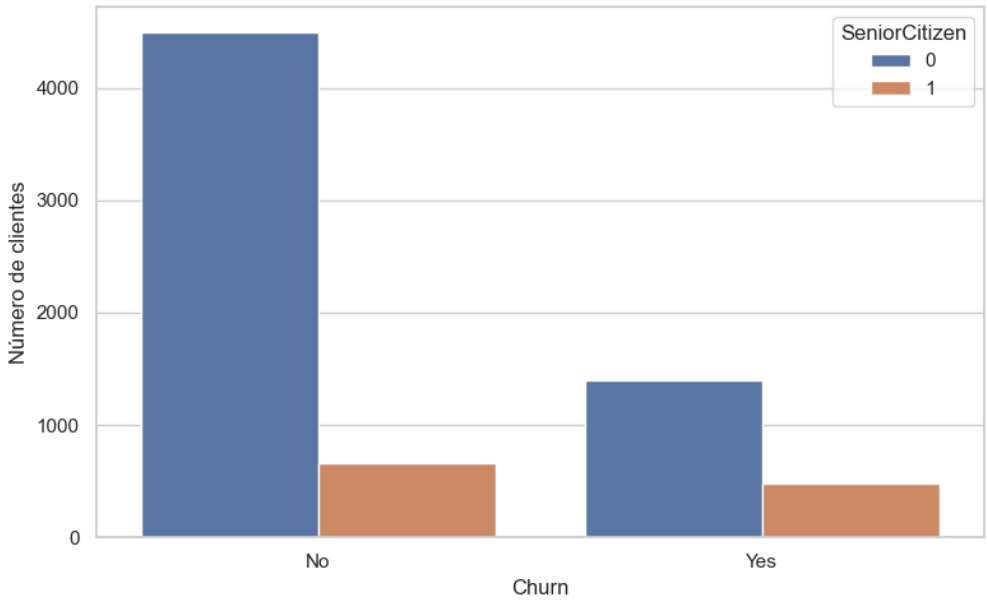

Figura 24: Distribución del churn según variable SeniorCitizen. *Fuente: Elaboración propia.*

- STREAMING MOVIES: En la figura (25) se observa que tanto quienes tienen como quienes no tienen servicio de streaming de películas muestran niveles de churn similares, aunque ligeramente menores en los que cuentan con el servicio. La categoría "No internet service" mantiene un churn bajo, lo que es consistente con un perfil de cliente con menor uso de servicios complementarios.

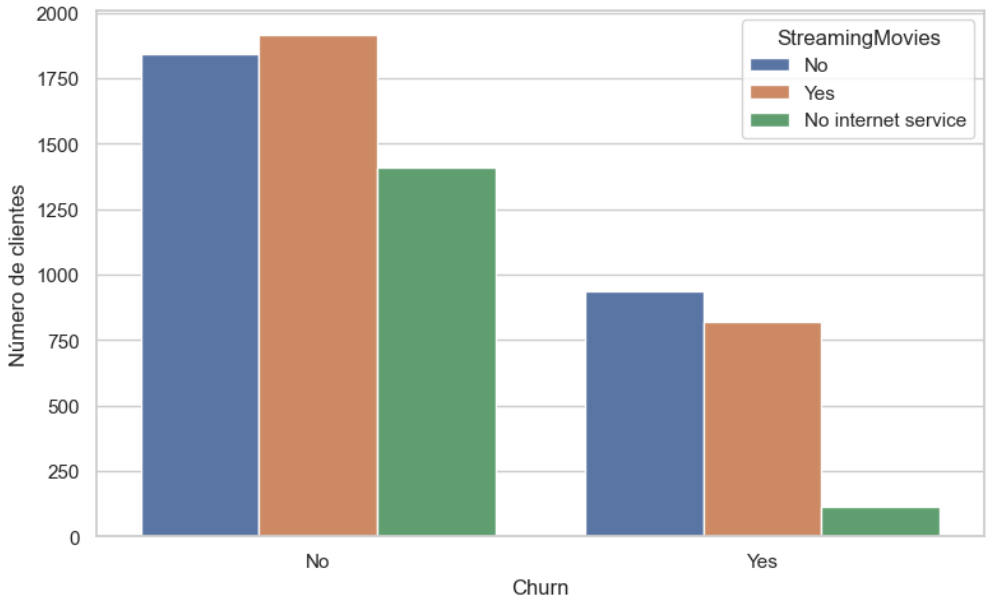

Figura 25: Distribución del churn según variable StreamingMovies. *Fuente: Elaboración propia.*

- STREAMING TV: En la figura (26) se observa que de forma análoga a StreamingMovies, la disponibilidad de servicio de streaming de televisión no muestra una diferencia marcada en las tasas de churn. Los clientes sin internet presentan tasas de abandono muy reducidas, probablemente por su baja integración en los servicios de conectividad más avanzados.

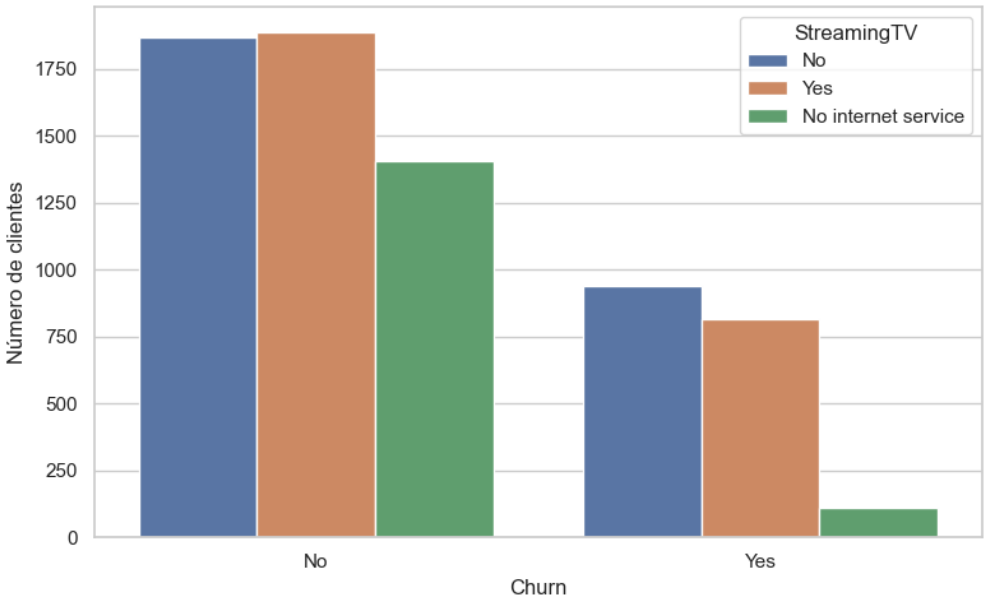

Figura 26: Distribución del churn según variable StreamingTV. *Fuente: Elaboración propia.*

- TECHSUPPORT: En la figura (27) se observa que los clientes que no cuentan con soporte técnico (TechSupport = No) exhiben una tasa de churn significativamente mayor que aquellos que sí disponen del servicio. Esto sugiere que el soporte técnico podría ser un factor relevante en la retención de clientes, ya sea por la resolución de problemas o por el valor percibido que aporta a la experiencia de usuario.

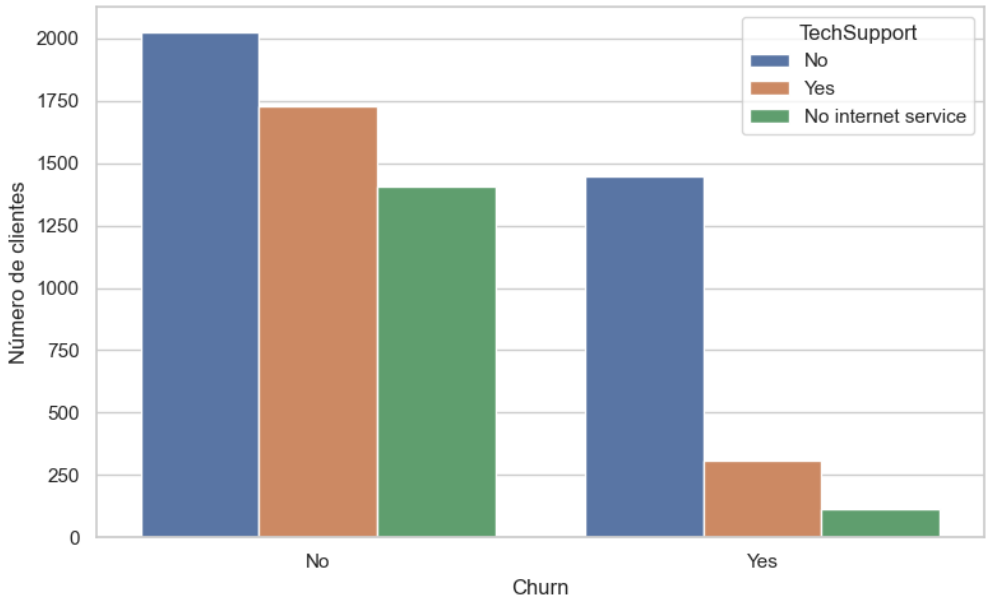

Figura 27: Distribución del churn según variable TechSupport. *Fuente: Elaboración propia.*

10.2 Procesamiento de Datos

Transformación de variables categóricas y escalado de variables numéricas

Dentro de la fase de procesamiento de datos, se realizaron diversas transformaciones y codificaciones con el objetivo de preparar el conjunto de datos para su uso en modelos de aprendizaje automático.

En primer lugar, las variables categóricas con más de dos categorías fueron transformadas en variables ficticias (*dummy variables*) mediante el método de codificación *one-hot* con n-1 categorías. Este procedimiento permite representar información categórica en formato numérico, evitando la trampa de las variables ficticias (*dummy variable trap*), que provoca colinealidad perfecta en modelos lineales (Gujarati y Porter, 2009).

Por otra parte, las variables binarias con respuestas "*Yes*" y "*No*" fueron recodificadas a formato numérico binario (1 y 0, respectivamente), garantizando compatibilidad con los distintos algorit-

mos empleados. Con estas transformaciones se obtuvo una base de datos completamente numérica y estructurada, apta para las etapas posteriores de modelamiento.

Detección y tratamiento de multicolinealidad mediante VIF

Para cuantificar la multicolinealidad entre las variables explicativas se calculó el VIF (*Variance Inflation Factor*), una métrica que estima cuánto aumenta la varianza de los coeficientes debido a la correlación entre predictores. Valores de VIF superior a 10 son generalmente indicativos de multicolinealidad severa, aunque en contextos prácticos valores por encima de 5 pueden considerarse preocupantes (Kutner et al., 2005).

Variable	**VIF**	**Variable**	**VIF**
MultipleLines_No_phone_service	∞	MultipleLines_Yes	7,29
PhoneService	∞	DeviceProtection_Yes	6,92
TechSupport_No_internet_service	$1,95 \cdot 10^9$	OnlineBackup_Yes	6,79
OnlineSecurity_No_internet_service	$1,58 \cdot 10^9$	TechSupport_Yes	6,48
InternetService_No	$1,58 \cdot 10^9$	OnlineSecurity_Yes	6,34
OnlineBackup_No_internet_service	$3,55 \cdot 10^8$	Contract_Two_year	2,65
StreamingTV_No_internet_service	$2{,}56 \cdot 10^8$	PaymentMethod_Electronic_check	1,98
StreamingMovies_No_internet_service	$1,3 \cdot 10^8$	PaymentMethod_Mailed_check	1,86
DeviceProtection_No_internet_service	$7,61 \cdot 10^7$	Contract_One_year	1,63
MonthlyCharges	854,63	PaymentMethod_Credit_card_(automatic)	1,56
InternetService_Fiber_optic	148,5	Partner	1,46
StreamingMovies_Yes	24,16	Dependents	1,38
StreamingTV_Yes	24,08	PaperlessBilling	1,21
TotalCharges	10,81	SeniorCitizen	1,15
tenure	7,58	gender	1,00

Tabla 2: Tabla de VIF por variable. *Fuente: Elaboración propia.*

En la Tabla 2 se observa que múltiples variables presentan valores extremadamente altos de VIF, e incluso algunos resultan infinitos,evidenciando redundancia perfecta. Este fenómeno ocurre principalmente en variables categóricas dummy asociadas a servicios no contratados ("*No internet service*" y "*No phone service*"), las cuales están perfectamente correlacionadas con otras categorías

de la misma variable original. Por ejemplo:

- INTERNETSERVICE: VIF = $1,57 \cdot 10^9$
- ONLINESECURITY: VIF = $1,57 \cdot 10^9$
- TECHSUPPORT_NO_INTERNET_SERVICE: VIF = $1,94 \cdot 10^8$
- MULTIPLELINES_NO_PHONE_SERVICE: VIF = ∞
- PHONESERVICE: VIF = ∞

Estos resultados confirman que dichas variables no aportan información nueva y generan problemas de estimación, como inestabilidad de coeficientes y varianzas infladas (Dormann et al., 2013). Para mitigar este problema, se decidió unificar las categorías que representan ausencia del servicio con la categoría "*No*" dentro de cada variable original. Por ejemplo, en MultipleLines, la categoría "*No phone service*" se fusionó con "*No*" y en variables como ONLINESECURITY o STREAMINGTV, la categoría "*No intenet service*" se trató como "*No*".

Luego de unificar las categorías de las variables ONLINESECURITY, ONLINEBACKUP, DEVICEPROTECTION, TECHSUPPORT, STREAMINGTV, STREAMINGMOVIES Y MULTIPLELINES, se volvió a calcular el VIF con el objetivo de evaluar la presencia de multicolinealidad. Los resultados mostraron que, si bien la mayoría de las variables presentaban valores aceptables, la variable MONTHLYCHARGES mantenía un VIF elevado, lo que indicaba una alta correlación con otras variables del modelo. La literatura advierte que mantener variables con alta colinealidad puede generar inestabilidad en las estimaciones de los coeficientes, inflar los errores estándar y dificultar la interpretación de los modelos (O'Brien (2007), James et al. (2013)), por lo que se optó por eliminar MONTHLYCHARGES del conjunto de datos.

Tras esta eliminación, se observó que las variables TOTALCHARGES y tenure aún presentaban VIF superiores a los recomendados, especialmente TOTALCHARGES, cuya correlación con tenure seguía siendo considerable. Dado que este escenario podía continuar afectando la estabilidad y robustez del modelo, se decidió remover también TOTALCHARGES, quedando un conjunto de predictores sin indicadores problemáticos de multicolinealidad.

Finalmente, se verificó nuevamente el factor de inflación de la varianza (VIF) para las variables restantes, confirmando la ausencia de multicolinealidad significativa. Los valores obtenidos se presentan en la Tabla 3.

Variable	**VIF**
tenure	2.827609
InternetService_No	2.692944
Contract_Two_year	2.633220
InternetService_Fiber_optic	2.005259
PaymentMethod_Electronic_check	1.973648
PaymentMethod_Mailed_check	1.837890
StreamingMovies	1.634755
StreamingTV	1.625973
Contract_One_year	1.625660
PaymentMethod_Credit_card_(automatic)	1.560625
TechSupport	1.481353
DeviceProtection	1.480118
Partner	1.462369
MultipleLines	1.423682
OnlineSecurity	1.415235
OnlineBackup	1.380823
Dependents	1.380811
PhoneService	1.354768
PaperlessBilling	1.208329
SeniorCitizen	1.153168
gender	1.001769

Tabla 3: Valores finales del factor de inflación de la varianza (VIF) para las variables seleccionadas. *Fuente: Elaboración propia.*

Exceptuando el término constante, todos los valores se mantuvieron por debajo del umbral crítico

de 5, lo que confirma la estabilidad del conjunto final de predictores y la ausencia de colinealidad significativa.

División del conjunto de datos

Para evaluar el desempeño de los modelos de manera objetiva y evitar sesgos, el conjunto de datos procesado se dividió en dos subconjuntos: **entrenamiento** y **prueba** (*train/test split*). Siguiendo las recomendaciones de Kuhn y Johnson (2013), se asignó el 80 % de los datos al conjunto de entrenamiento y el 20 % restante al conjunto de prueba. Esta proporción permite disponer de suficientes observaciones para el entrenamiento de los modelos, manteniendo un subconjunto independiente para la evaluación final.

La división se realizó de forma estratificada, preservando la proporción original de clases en ambos subconjuntos. Esta estratificación es especialmente importante en problemas de clasificación con clases desbalanceadas, ya que garantiza que tanto el conjunto de entrenamiento como el de prueba representen adecuadamente la distribución real del problema (He y Garcia, 2009).

Manejo de desbalance de clases con SMOTE

Durante el análisis exploratorio se identificó un desbalance en la variable objetivo, con un 26,6 % de clientes que abandonaron el servicio y un 73,4 % que permanecieron. Esta distribución sesgada puede inducir a los modelos de clasificación a favorecer la clase mayoritaria, disminuyendo su capacidad para detectar correctamente a los clientes propensos a abandonar.

Para abordar el desbalance de clases, se utilizó la técnica SMOTE (Synthetic Minority Oversampling Technique) propuesta por Chawla et al. (2002). Esta metodología fue seleccionada por sobre alternativas como el Random Oversampling o el Random Undersampling, ya que genera

instancias sintéticas de la clase minoritaria a partir de la interpolación entre observaciones reales, evitando la simple duplicación de registros y reduciendo el riesgo de sobreajuste.

A diferencia del Random Oversampling, que puede provocar redundancia y menor capacidad de generalización, y del Random Undersampling, que elimina información potencialmente valiosa de la clase mayoritaria, SMOTE mantiene el equilibrio entre ambas clases sin pérdida de datos. Además, su integración dentro del pipeline de validación cruzada permite evaluar el rendimiento de los modelos sobre conjuntos balanceados sin comprometer la independencia de los datos de prueba.

En concordancia con las recomendaciones metodológicas de Fernández et al. (2018), SMOTE se implementó como parte de un *pipeline* integrado, aplicándose exclusivamente sobre los subconjuntos de entrenamiento dentro de cada *fold* de la validación cruzada. Esto garantiza que el modelo no tenga acceso a datos del conjunto de validación al momento de generar las muestras sintéticas, evitando así cualquier forma de fuga de información (*data leakage*).

La implementación se realizó utilizando la librería *imbalanced-learn*, ajustando los parámetros de SMOTE para balancear ambas clases dentro de cada *fold*. Esta estrategia permite evaluar los modelos de forma realista, manteniendo el desbalance original en el conjunto de prueba, y se orienta a optimizar especialmente métricas sensibles a la clase minoritaria, como el *Recall*, la *F1-score* y el *AUC-PR*, fundamentales en escenarios donde el costo de una falsa negativa es considerable.

10.3 Entrenamiento y evaluación

Antes de proceder con la presentación individual de los modelos, se implementó un flujo estándar de entrenamiento y evaluación que se aplicó de manera uniforme a todos ellos, con el objetivo de garantizar la comparabilidad de los resultados. Este proceso comenzó con la división del conjunto de datos en subconjuntos de entrenamiento y prueba, asegurando la estratificación para mantener la proporción de clases original. Posteriormente, se incorporó la técnica SMOTE dentro de un *pipeline*, de manera que el sobremuestreo de la clase minoritaria se aplicara únicamente sobre los datos de entrenamiento en cada pliegue de validación cruzada, evitando fugas de información (*data leakage*).

Asimismo, dentro del *pipeline* se incluyó un paso de escalado de variables numéricas, aplicado únicamente a los modelos que son sensibles a la magnitud o a la escala de las características. Esta integración permitió automatizar el preprocesamiento y asegurar la coherencia del flujo de entrenamiento entre los distintos algoritmos.

Requieren escalado	**No requieren escalado**
Regresión Logística	Árbol de Decisión
K-Nearest Neighbors (KNN)	Random Forest
Red Neuronal (MLP)	Gradient Boosting
Naive Bayes	XGBoost
	LightGBM
	CatBoost

Tabla 4: Modelos según requerimiento de escalado de variables. *Fuente: Elaboración propia.*

En cada *pipeline*, el modelo correspondiente se ajustó utilizando una búsqueda exhaustiva de hiperparámetros mediante *GridSearchCV*, con validación cruzada estratificada de cinco pliegues y la métrica AUC-ROC como criterio de selección. Este enfoque permitió identificar la configuración óptima de cada algoritmo en términos de su capacidad para discriminar entre clientes que

abandonan y aquellos que permanecen, incluso en escenarios de clases desbalanceadas.

Una vez seleccionados los mejores hiperparámetros, se evaluó cada modelo utilizando predicciones cruzadas para obtener métricas de rendimiento robustas: *accuracy*, *precision*, *recall*, *F1-score*, *specificity* y AUC-ROC. Además, se generaron las curvas ROC correspondientes y las matrices de confusión, con el fin de ofrecer una visión tanto global como detallada del comportamiento de cada clasificador. Los tiempos de entrenamiento y predicción también fueron registrados para complementar el análisis comparativo.

A continuación, se presentan los resultados obtenidos para cada uno de los modelos considerados, siguiendo este esquema metodológico común.

Regresión Logística

La regresión logística fue empleada como modelo base para el análisis comparativo, dadas sus ventajas en términos de simplicidad, interpretabilidad y utilidad como referencia frente a modelos más sofisticados. El entrenamiento se realizó mediante el *pipeline* descrito en la sección anterior, que incluye balanceo con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros obtenidos correspondieron a una regularización L2 con un valor de $C = 0{,}05$, utilizando el *solver* `lbfgs`. El parámetro C controla la intensidad de la regularización, siendo el inverso del parámetro de penalización λ ($C = 1/\lambda$). Valores más pequeños de C implican una regularización más fuerte, lo que favorece modelos más simples y con menor riesgo de sobreajuste, mientras que valores grandes reducen el efecto de la penalización y permiten un ajuste más cercano a los datos. .

Los resultados obtenidos se resumen en la Tabla 5, mientras que la Figura 28 muestra la curva

ROC y la Figura 29, la matriz de confusión correspondiente.

Métrica	Valor (%)
Accuracy	73,70
Precision	50,36
Recall	75,67
F1-Score	60,47
Specificity	72,99
AUC-ROC	82,33
AUC-PR	62,07

Tabla 5: Métricas de desempeño para regresión logística. *Fuente: Elaboración propia.*

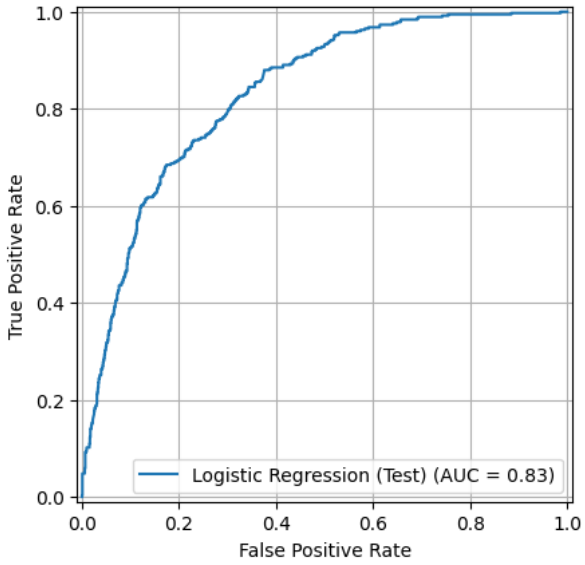

Figura 28: Curva ROC para regresión logística. *Fuente: Elaboración propia.*

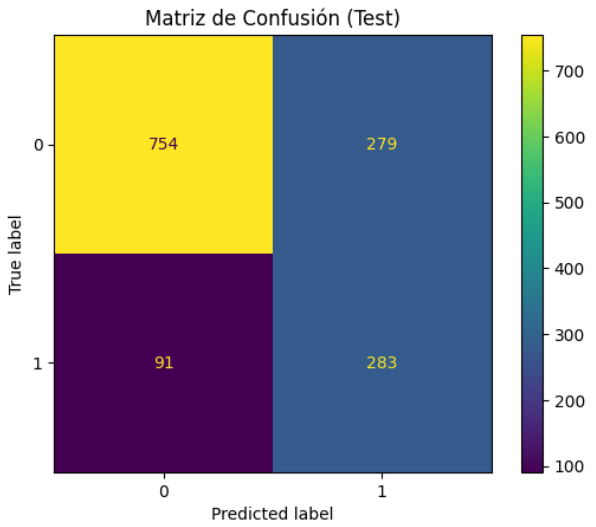

Figura 29: Matriz de confusión para regresión logística. *Fuente: Elaboración propia.*

Además del rendimiento predictivo, se consideró la eficiencia computacional del modelo. En el caso de la regresión logística, el tiempo de entrenamiento fue de aproximadamente 2,89 segundos, mientras que el tiempo de predicción durante la validación cruzada fue de 0,67 segundos.

K-Nearest-Neighbors (KNN)

El modelo K-Nearest Neighbors (KNN) fue considerado dentro del análisis comparativo por su simplicidad, carácter no paramétrico y capacidad para adaptarse a distribuciones complejas sin requerir una función explícita de decisión. Se utilizó el mismo *pipeline* descrito previamente, incorporando balanceo con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros obtenidos correspondieron a un número de vecinos igual a 200, distancia tipo Manhattan ($p = 1$) y pesos uniformes, es decir, todos los vecinos contribuyen por igual a la predicción.

Una vez entrenado sobre los datos balanceados, el modelo fue evaluado sobre el conjunto de prueba. Los resultados se resumen en la Tabla 6, mientras que la Figura 30 presenta la curva ROC y la Figura 31, la matriz de confusión respectiva.

Métrica	**Valor (%)**
Accuracy	69,51
Precision	45,94
Recall	83,16
F1-Score	59,18
Specificity	64,57
AUC-ROC	83,27
AUC-PR	61,81

Tabla 6: Métricas de desempeño para K-Nearest Neighbors. *Fuente: Elaboración propia.*

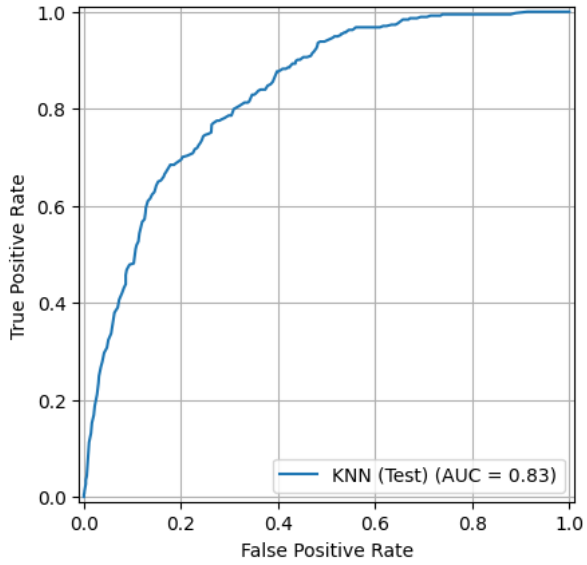

Figura 30: Curva ROC para KNN. *Fuente: Elaboración propia.*

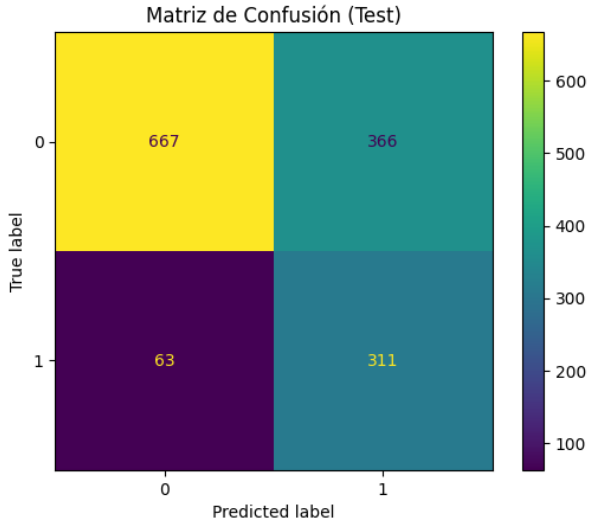

Figura 31: Matriz de confusión para KNN. *Fuente: Elaboración propia.*

En términos de eficiencia computacional, el modelo KNN presentó un tiempo de entrenamiento de aproximadamente 6,17 segundos, mientras que el tiempo de predicción en el proceso de validación cruzada alcanzó los 2,10 segundos.

Árbol de decisión

El modelo de Árbol de Decisión fue incluido en el análisis por su capacidad para representar reglas de decisión interpretables y adaptarse a relaciones no lineales entre las variables. Se utilizó el mismo *pipeline* descrito anteriormente, incorporando balanceo con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros obtenidos correspondieron a una profundidad máxima de 5 niveles, un mínimo de 10 muestras por hoja (MIN_SAMPLES_LEAF), y un mínimo de 2 muestras para realizar una división interna (MIN_SAMPLES_SPLIT). Además, no se aplicó poda por complejidad, ya que CCP_ALPHA fue igual a 0.

El modelo fue entrenado sobre los datos balanceados y evaluado sobre el conjunto de prueba. Los resultados se resumen en la Tabla (7), mientras que la Figura (32) muestra la curva ROC y la Figura (33), la matriz de confusión correspondiente.

Métrica	**Valor (%)**
Accuracy	69,94
Precision	46,18
Recall	79,14
F1-Score	58,33
Specificity	66,60
AUC-ROC	81,47
AUC-PR	55,82

Tabla 7: Métricas de desempeño para Árbol de Decisión. *Fuente: Elaboración propia.*

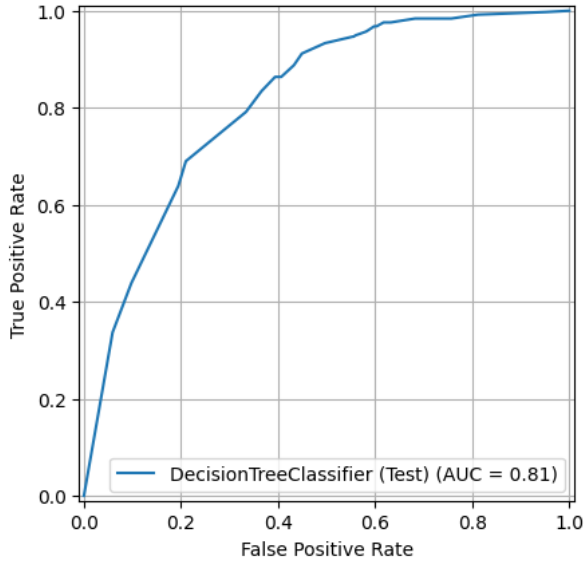

Figura 32: Curva ROC para Árbol de Decisión. *Fuente: Elaboración propia.*

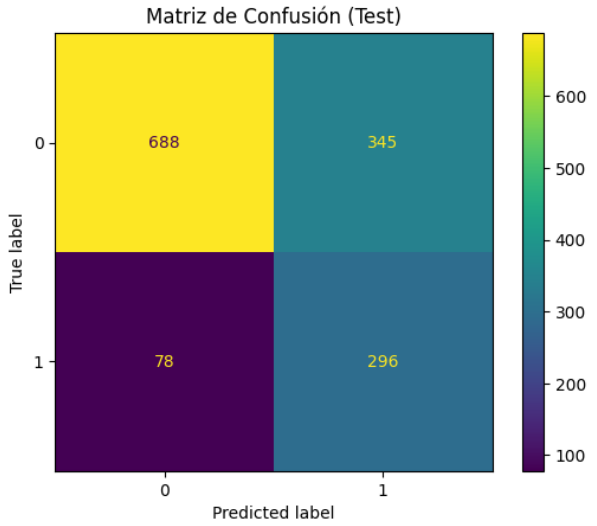

Figura 33: Matriz de confusión para Árbol de Decisión. *Fuente: Elaboración propia.*

Respecto a su desempeño computacional, el modelo de Árbol de Decisión registró un tiempo de entrenamiento de aproximadamente 2,63 segundos y un tiempo de predicción durante la validación cruzada de 0,53 segundos.

Naive Bayes

El modelo Naive Bayes fue considerado por su bajo costo computacional, buen desempeño en contextos de alta dimensionalidad y su fundamento probabilístico, el cual asume independencia condicional entre las variables predictoras. Se implementó el *pipeline* previamente descrito, que contempla balanceo de clases con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

El mejor resultado se obtuvo con un valor de suavizado (VAR_SMOOTHING) igual a 1×10^{-9}, lo que permite evitar problemas numéricos asociados a probabilidades nulas.

Tras entrenar el modelo con los datos balanceados, se procedió a su evaluación sobre el conjunto de prueba. Las métricas obtenidas se presentan en la Tabla 8, mientras que la Figura 34 muestra la curva ROC y la Figura 35, la matriz de confusión correspondiente.

Métrica	**Valor (%)**
Accuracy	72,85
Precision	49,32
Recall	77,27
F1-Score	60,21
Specificity	71,25
AUC-ROC	81,88
AUC-PR	58,10

Tabla 8: Métricas de desempeño para Naive Bayes. *Fuente: Elaboración propia.*

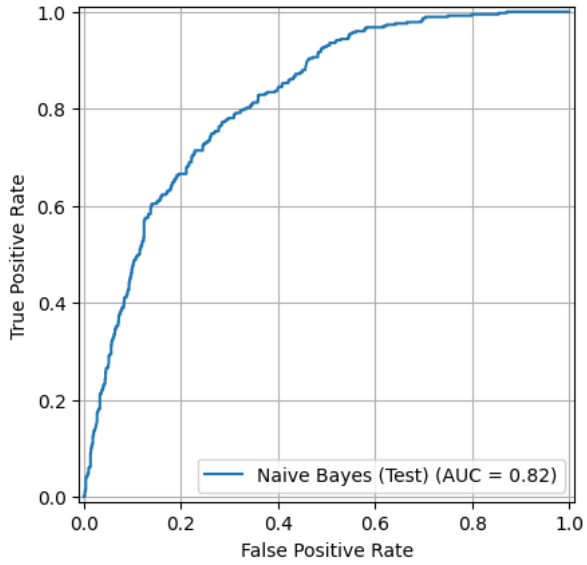

Figura 34: Curva ROC para Naive Bayes. *Fuente: Elaboración propia.*

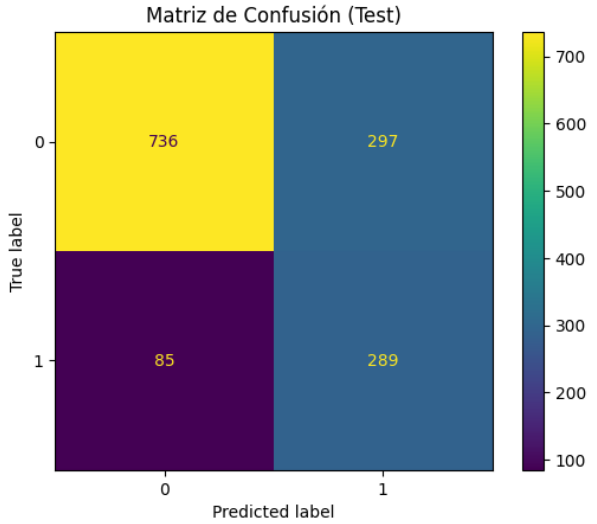

Figura 35: Matriz de confusión para Naive Bayes. *Fuente: Elaboración propia.*

En cuanto a su eficiencia computacional, el modelo Naive Bayes presentó un tiempo de entrenamiento de aproximadamente 2,12 segundos, y un tiempo de predicción durante la validación cruzada de 0,59 segundos.

Random Forest

El modelo Random Forest fue incluido en el análisis debido a su robustez, capacidad para manejar conjuntos de datos con alta dimensionalidad y buen rendimiento en tareas de clasificación. Este ensamble de árboles de decisión permite reducir el sobreajuste y mejorar la generalización del modelo. Se aplicó el *pipeline* descrito anteriormente, que contempla el uso de SMOTE para balanceo de clases, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros encontrados fueron: una profundidad máxima de 5 niveles, un total de 250 estimadores, un mínimo de una muestra por hoja, y selección aleatoria de variables mediante

la raíz cuadrada del total de *features* (MAX_FEATURES = "SQRT").

Una vez entrenado sobre los datos balanceados, el modelo fue evaluado sobre el conjunto de prueba. Las métricas de desempeño se presentan en la Tabla 9, mientras que la Figura 36 muestra la curva ROC y la Figura 37, la matriz de confusión correspondiente.

Métrica	**Valor (%)**
Accuracy	74,70
Precision	51,62
Recall	76,47
F1-Score	61,64
Specificity	74,06
AUC-ROC	82,67
AUC-PR	60,86

Tabla 9: Métricas de desempeño para Random Forest. *Fuente: Elaboración propia.*

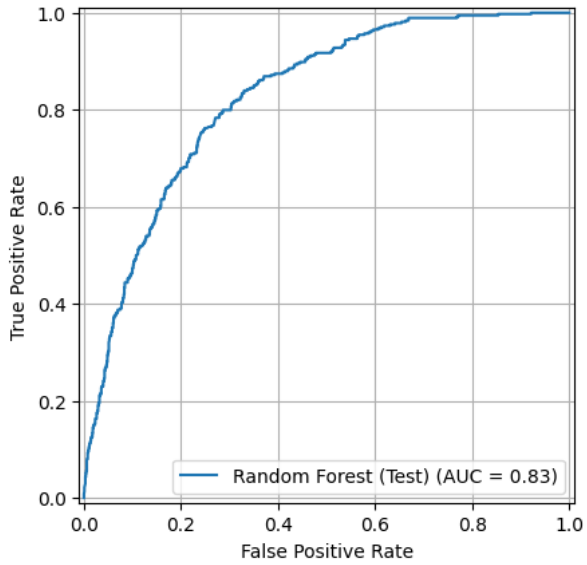

Figura 36: Curva ROC para Random Forest. *Fuente: Elaboración propia.*

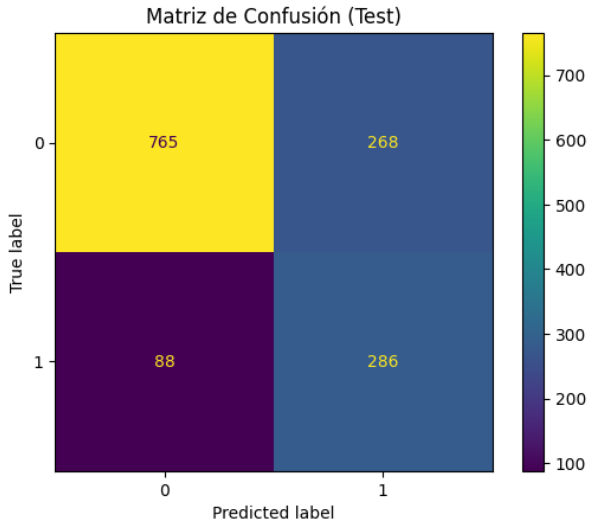

Figura 37: Matriz de confusión para Random Forest. *Fuente: Elaboración propia.*

En términos computacionales, el modelo Random Forest requirió aproximadamente 4,16 segundos para su entrenamiento. En tanto, el tiempo de predicción durante la validación cruzada fue de 7,58 segundos, el más alto entre los modelos evaluados.

Gradient Boosting

El modelo Gradient Boosting fue incorporado en el análisis por su capacidad para construir predictores fuertes a partir de clasificadores débiles mediante un enfoque secuencial. Este algoritmo es especialmente útil en contextos donde se requiere capturar interacciones complejas entre variables. Se aplicó el *pipeline* estandarizado que considera balanceo con SMOTE, validación cruzada estratificada y optimización de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros identificados fueron: una tasa de aprendizaje (LEARNING_RATE) de 0,01, una profundidad máxima de 5 niveles, 200 estimadores, y una fracción de muestreo por iteración igual a 0,8 (SUBSAMPLE).

Tras entrenar el modelo con los datos balanceados, se evaluó su desempeño sobre el conjunto de prueba. Los resultados se muestran en la Tabla 10, mientras que las Figuras 38 y 39 presentan la curva ROC y la matriz de confusión, respectivamente.

Métrica	**Valor (%)**
Accuracy	73,92
Precision	50,63
Recall	75,67
F1-Score	60,66
Specificity	73,28
AUC-ROC	82,76
AUC-PR	62,94

Tabla 10: Métricas de desempeño para Gradient Boosting. *Fuente: Elaboración propia.*

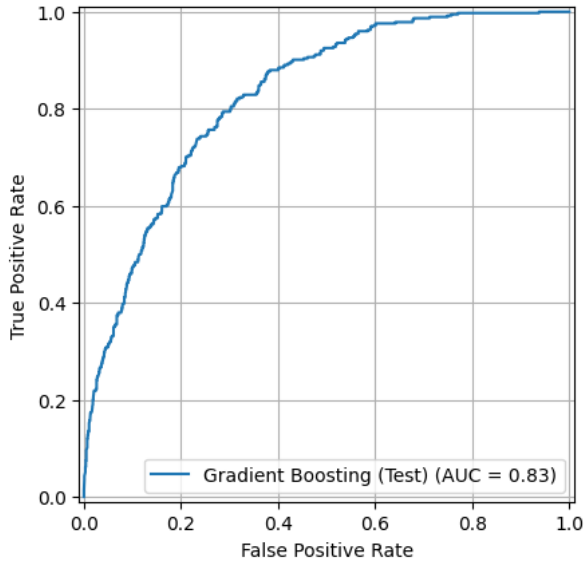

Figura 38: Curva ROC para Gradient Boosting. *Fuente: Elaboración propia.*

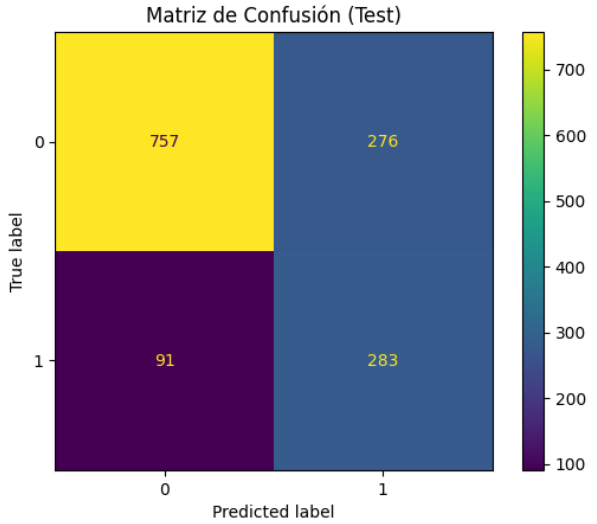

Figura 39: Matriz de confusión para Gradient Boosting. *Fuente: Elaboración propia.*

En cuanto a su eficiencia computacional, el modelo Gradient Boosting registró un tiempo de entrenamiento de aproximadamente 5,50 segundos. No obstante, el tiempo de predicción durante la validación cruzada fue considerablemente mayor, alcanzando los 19,36 segundos, lo que lo posiciona como el modelo más lento en esta etapa.

XGBoost

El modelo XGBoost fue incluido en la comparación por su alta eficiencia computacional, capacidad para manejar datos desbalanceados y por incorporar técnicas avanzadas de regularización, lo que lo convierte en uno de los algoritmos más utilizados en competencias de ciencia de datos. Se utilizó el *pipeline* previamente establecido, que contempla balanceo de clases con SMOTE, validación cruzada estratificada y búsqueda de hiperparámetros mediante *GridSearchCV*.

Los mejores parámetros encontrados fueron: una tasa de aprendizaje (LEARNING_RATE) de 0,01, una profundidad máxima de 4 niveles, 200 estimadores, un SUBSAMPLE de 0,8, y un COLSAM-

PLE_BYTREE también de 0,8, lo que permite controlar la aleatoriedad en la selección de variables. El modelo fue entrenado con los datos balanceados y evaluado sobre el conjunto de prueba. Las métricas de rendimiento se muestran en la Tabla 11, mientras que la Figura 40 presenta la curva ROC y la Figura 41, la matriz de confusión correspondiente.

Métrica	**Valor (%)**
Accuracy	73,99
Precision	50,69
Recall	78,61
F1-Score	61,64
Specificity	72,31
AUC-ROC	82,68
AUC-PR	61,63

Tabla 11: Métricas de desempeño para XGBoost. *Fuente: Elaboración propia.*

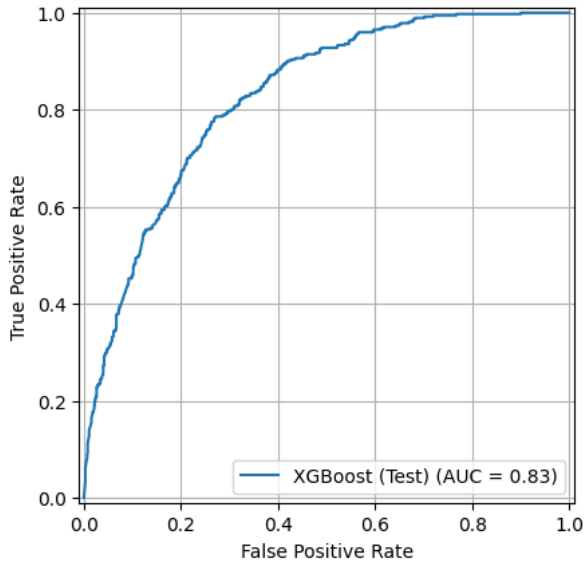

Figura 40: Curva ROC para XGBoost. *Fuente: Elaboración propia.*

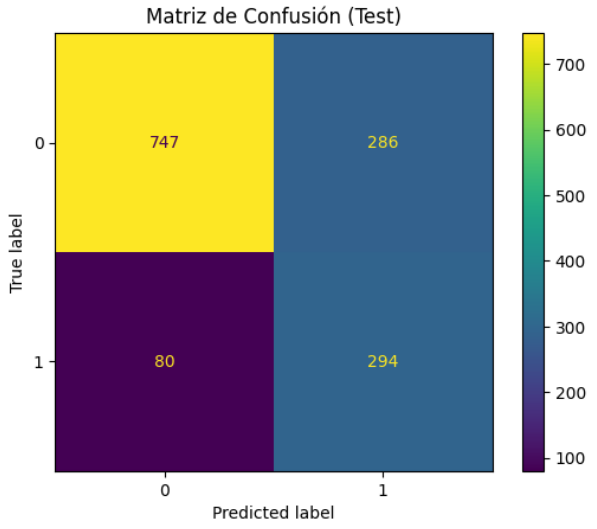

Figura 41: Matriz de confusión para XGBoost. *Fuente: Elaboración propia.*

Desde una perspectiva computacional, XGBoost mostró un tiempo de entrenamiento relativamente bajo, con aproximadamente 1,82 segundos. En cuanto a la predicción durante la validación cruzada, esta tomó alrededor de 2,41 segundos.

LightGBM

El modelo LightGBM fue incorporado en el análisis por su eficiencia computacional, excelente capacidad de escalabilidad y desempeño competitivo en tareas de clasificación con grandes volúmenes de datos. Este algoritmo basado en *gradient boosting* optimiza el tiempo de entrenamiento al utilizar histogramas y técnicas de reducción de complejidad. Se utilizó el *pipeline* previamente descrito, incluyendo balanceo con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

La mejor combinación de parámetros incluyó: una tasa de aprendizaje (LEARNING_RATE) de 0,001, 300 estimadores, un total de 31 hojas por árbol (NUM_LEAVES), y fracciones de muestreo (SUBSAMPLE) y de columnas por árbol (COLSAMPLE_BYTREE) iguales a 0,8.

El modelo fue entrenado con los datos balanceados y evaluado sobre el conjunto de prueba. Las métricas obtenidas se resumen en la Tabla 12, y se presentan visualmente en la Figura 42 (curva ROC) y en la Figura 43 (matriz de confusión).

Métrica	**Valor (%)**
Accuracy	74,41
Precision	51,29
Recall	74,60
F1-Score	60,78
Specificity	74,35
AUC-ROC	82,77
AUC-PR	62,30

Tabla 12: Métricas de desempeño para LightGBM. *Fuente: Elaboración propia.*

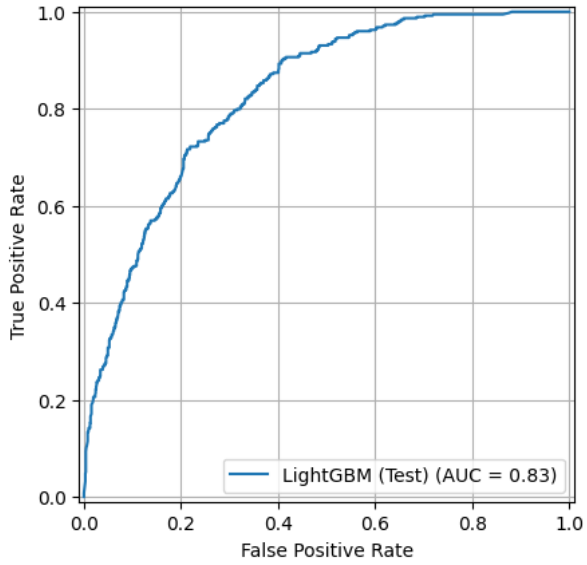

Figura 42: Curva ROC para LightGBM. *Fuente: Elaboración propia.*

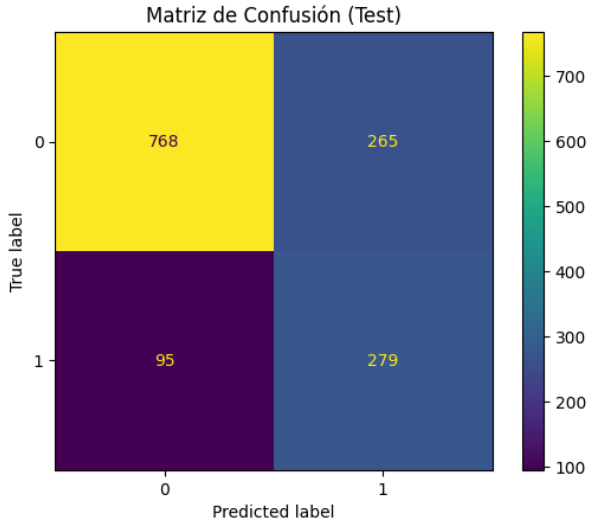

Figura 43: Matriz de confusión para LightGBM. *Fuente: Elaboración propia.*

En lo que respecta a eficiencia computacional, el modelo LightGBM requirió aproximadamente 7,22 segundos para su entrenamiento y 2,74 segundos para realizar las predicciones durante la validación cruzada.

CatBoost

El modelo CatBoost fue considerado en el análisis por su excelente rendimiento en datos categóricos y su capacidad para evitar el sobreajuste gracias a sus técnicas de ordenamiento y regularización internas. Este algoritmo de *gradient boosting* desarrollado por Yandex se destaca por no requerir preprocesamiento extenso y por manejar automáticamente variables categóricas. Para su entrenamiento, se utilizó el *pipeline* estándar, que incluye balanceo con SMOTE, validación cruzada estratificada y ajuste de hiperparámetros mediante *GridSearchCV*.

Los mejores hiperparámetros obtenidos fueron: una tasa de aprendizaje (LEARNING_RATE) de 0,01, 400 iteraciones, una profundidad de árbol de 2 niveles, regularización L2 igual a 9 (L2_LEAF_REG),

y un parámetro de temperatura para BAGGING de 0,5.

El modelo fue entrenado sobre los datos balanceados y evaluado con el conjunto de prueba. Las métricas de rendimiento se detallan en la Tabla 13, mientras que las Figuras 44 y 45 muestran la curva ROC y la matriz de confusión, respectivamente.

Métrica	**Valor (%)**
Accuracy	74,13
Precision	50,87
Recall	78,07
F1-Score	61,60
Specificity	72,70
AUC-ROC	82,65
AUC-PR	62,39

Tabla 13: Métricas de desempeño para CatBoost. *Fuente: Elaboración propia.*

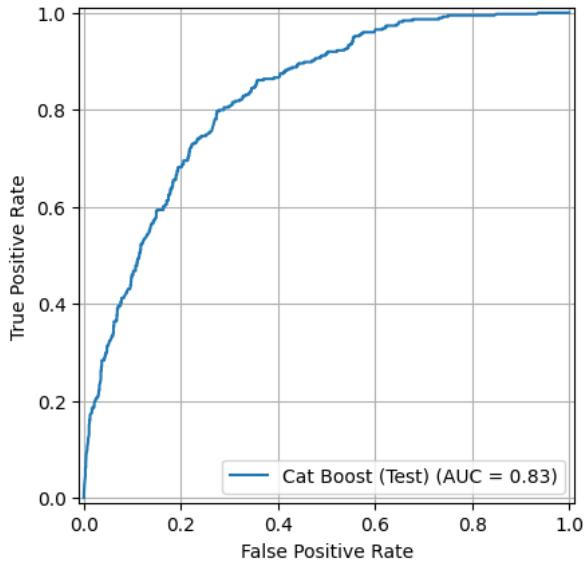

Figura 44: Curva ROC para CatBoost. *Fuente: Elaboración propia.*

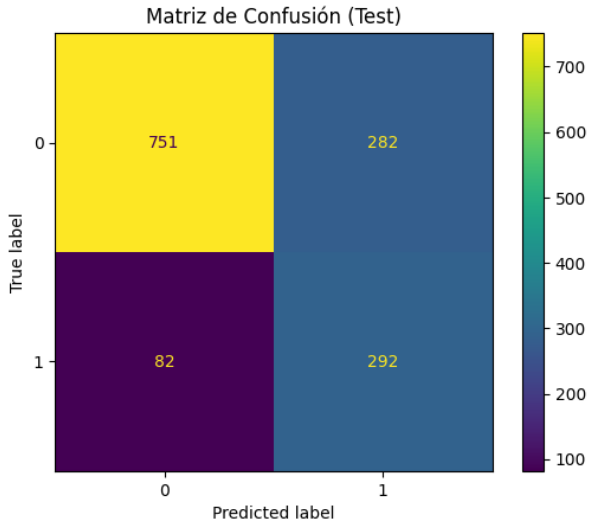

Figura 45: Matriz de confusión para CatBoost. *Fuente: Elaboración propia.*

En términos de eficiencia computacional, el modelo CatBoost requirió aproximadamente 5,29 segundos para completar su entrenamiento, mientras que el tiempo de predicción durante la validación cruzada fue de 11,57 segundos, posicionándolo entre los modelos más lentos en esta etapa.

Red Neuronal (MLP)

El modelo de Red Neuronal Multicapa (MLP, por sus siglas en inglés) fue incorporado al análisis como representante de arquitecturas de aprendizaje profundo, con capacidad para capturar relaciones no lineales complejas entre las variables. A diferencia de los modelos más simples, el MLP permite la composición de múltiples capas ocultas, lo que incrementa su capacidad expresiva. Se utilizó el *pipeline* estándar, incluyendo balanceo con SMOTE, validación cruzada estratificada y búsqueda de hiperparámetros mediante *GridSearchCV*.

Los hiperparámetros óptimos identificados fueron: una capa oculta de 64 neuronas con función de activación TANH, regularización L2 (ALPHA) de 0,001, tamaño de batch de 64, tasa de aprendizaje constante (CONSTANT) con valor inicial de 0,001.

El modelo fue entrenado sobre los datos balanceados y evaluado con el conjunto de prueba. Las métricas obtenidas se presentan en la Tabla 14, mientras que la Figura 46 muestra la curva ROC y la Figura 47, la matriz de confusión.

Métrica	**Valor (%)**
Accuracy	73,63
Precision	50,25
Recall	79,41
F1-Score	61,55
Specificity	71,54
AUC-ROC	83,52
AUC-PR	62,73

Tabla 14: Métricas de desempeño para Red Neuronal Multicapa (MLP). *Fuente: Elaboración propia.*

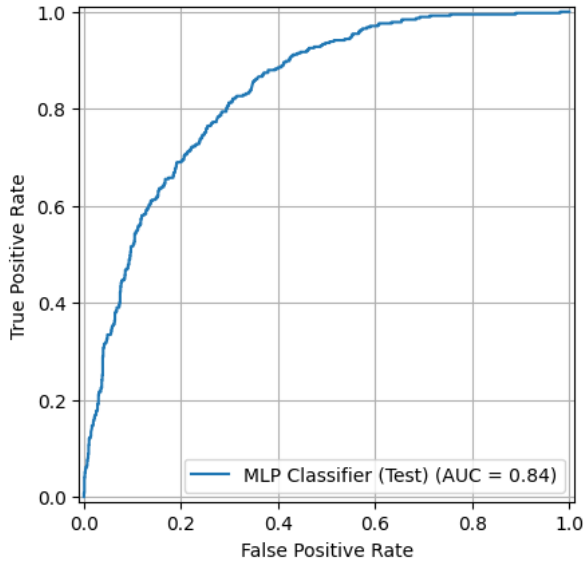

Figura 46: Curva ROC para MLP. *Fuente: Elaboración propia.*

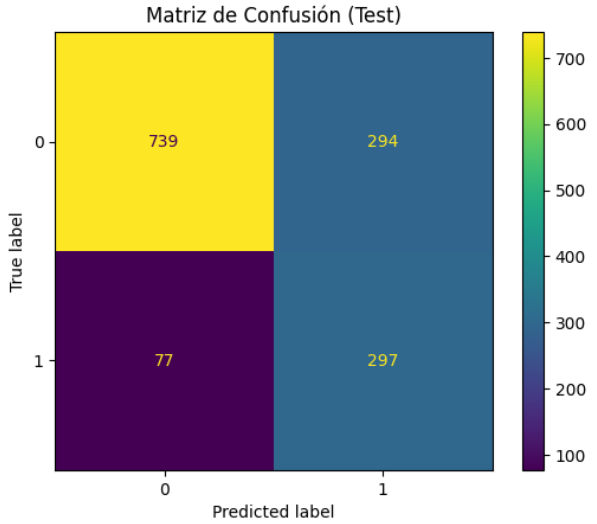

Figura 47: Matriz de confusión para MLP. *Fuente: Elaboración propia.*

Respecto a su eficiencia computacional, el modelo MLP presentó un tiempo de entrenamiento de aproximadamente 2,46 segundos. Sin embargo, su tiempo de predicción durante la validación cruzada fue considerablemente mayor, alcanzando los 10,90 segundos.

10.4 Interpretación de modelos

La interpretación de resultados es fundamental para evaluar la validez y aplicabilidad de los modelos. En el caso de los algoritmos lineales y de naturaleza probabilística, como la regresión logística o Naive Bayes, la interpretación se puede realizar de manera directa a través de los coeficientes o probabilidades estimadas. Para los modelos más complejos y de difícil interpretación, como Random Forest, Gradient Boosting, XGBoost, LightGBM, CatBoost, redes neuronales y otros métodos de ensamble, se empleó la librería SHAP (*SHapley Additive exPlanations*), la cual permite cuantificar el aporte de cada variable en la predicción individual y global, otorgando transparencia al comportamiento del modelo.

Para comprender el razonamiento detrás de las predicciones del modelo, se utilizó el método SHAP (*SHapley Additive exPlanations*). Este enfoque, basado en la teoría de juegos cooperativos, calcula el aporte promedio de cada variable a la predicción, otorgando una medida de importancia consistente y localmente explicable (Lundberg y Lee (2017)).

En los gráficos de valores SHAP, cada punto representa una observación del conjunto de datos. El eje x indica el impacto de la variable en la predicción (positivo o negativo), mientras que el color refleja el valor original de la característica (rojo = alto, azul = bajo). Las variables situadas hacia la derecha incrementan la probabilidad de *churn*, y las que se ubican hacia la izquierda la reducen. La dispersión vertical muestra la variabilidad de este efecto entre distintos clientes.

De este modo, SHAP permite interpretar tanto la relevancia global de las variables como su influencia individual, proporcionando una visión transparente del comportamiento del modelo.

Interpretación Regresión Logística

La regresión logística fue seleccionada como modelo base debido a su alta interpretabilidad, bajo costo computacional y capacidad para entregar estimaciones probabilísticas sobre la variable objetivo. En el contexto del churn, este tipo de modelo permite cuantificar cómo influye cada variable en la propensión de abandono de un cliente, lo que resulta especialmente valioso para la toma de decisiones comerciales.

En este modelo, el signo del coeficiente indica la dirección del efecto: un valor positivo implica un aumento en la probabilidad de churn, mientras que un valor negativo sugiere un efecto protector. Asimismo, los *odds ratios* permiten interpretar el efecto multiplicativo sobre la razón de odds. La Tabla 15 resume los principales coeficientes estimados y sus respectivos *odds* para cada variable.

Variable	Coeficiente	Odds Ratio
InternetService_Fiber_optic	0,8803	2,4116
PaymentMethod_Electronic_check	0,3855	1,4703
StreamingMovies	0,3361	1,3990
PaperlessBilling	0,3227	1,3809
MultipleLines	0,3007	1,3503
StreamingTV	0,2723	1,3129
SeniorCitizen	0,1243	1,1323
PaymentMethod_Mailed_check	0,0870	1,0878
DeviceProtection	0,0156	1,0152
PaymentMethod_Credit_card (automatic)	0,0022	1,0022
gender	-0,0015	0,9985
Partner	-0,0069	0,9931
OnlineBackup	-0,1661	0,8469
Dependents	-0,3608	0,6971
PhoneService	-0,3666	0,6934
TechSupport	-0,4290	0,6565
OnlineSecurity	-0,5156	0,5971
Contract_One_year	-0,7010	0,4969
tenure	-0,8145	0,4429
InternetService_No	-0,8189	0,4409
Contract_Two_year	-1,1269	0,3240

Tabla 15: Coeficientes y odds ratio del modelo de regresión logística. *Fuente: Elaboración propia.*

Entre los factores que aumentan significativamente la probabilidad de abandono destacan los clientes con InternetService_Fiber_optic (*odds ratio* = 2,41), aquellos que utilizan PaymentMethod_Electronic_check (1,47) y quienes tienen habilitados servicios de entretenimiento como StreamingMovies y StreamingTV. Este grupo parece estar compuesto por usuarios digitales más exigentes, posiblemente más sensibles a la percepción de valor recibido frente a lo pagado.

Por otro lado, se observa que variables como Contract_Two_year (0,32), Contract_One_year (0,50) y el tiempo de permanencia tenure (0,44) tienen un fuerte efecto protector, disminuyendo la propensión al churn. Este resultado es coherente con la literatura, que indica que los contratos de largo plazo actúan como mecanismos efectivos de retención (Ganesh et al., 2000).

Asimismo, el modelo revela que los servicios complementarios como ONLINESECURITY, TECH-SUPPORT y ONLINEBACKUP reducen significativamente el riesgo de abandono, al contribuir a una mayor percepción de soporte y valor agregado por parte del cliente. En contraste, variables como GENDER, PARTNER y DEVICEPROTECTION presentan coeficientes cercanos a cero, indicando un impacto marginal o nulo en el comportamiento de churn.

Interpretación del modelo K-Nearest Neighbors (KNN)

El modelo K-Nearest Neighbors clasifica observaciones en función de la clase predominante entre sus vecinos más cercanos en el espacio de características, utilizando métricas de distancia como criterio principal. Esta metodología no requiere entrenamiento en el sentido tradicional, pero sí se ve afectada por la dimensionalidad de los datos y la distribución de las observaciones.

Una de las principales fortalezas del modelo KNN es su capacidad para capturar relaciones locales complejas, lo que le permite adaptarse bien a estructuras no lineales en los datos. Sin embargo, esto viene a costa de una baja interpretabilidad global, ya que las predicciones se basan en configuraciones específicas del espacio de características que no son fácilmente generalizables.

Con el fin de aportar mayor comprensión al comportamiento del modelo, se realizó un análisis de interacción utilizando valores SHAP. Este análisis permite identificar combinaciones de variables cuyo efecto conjunto sobre la predicción es relevante, aún cuando sus efectos individuales puedan ser débiles.

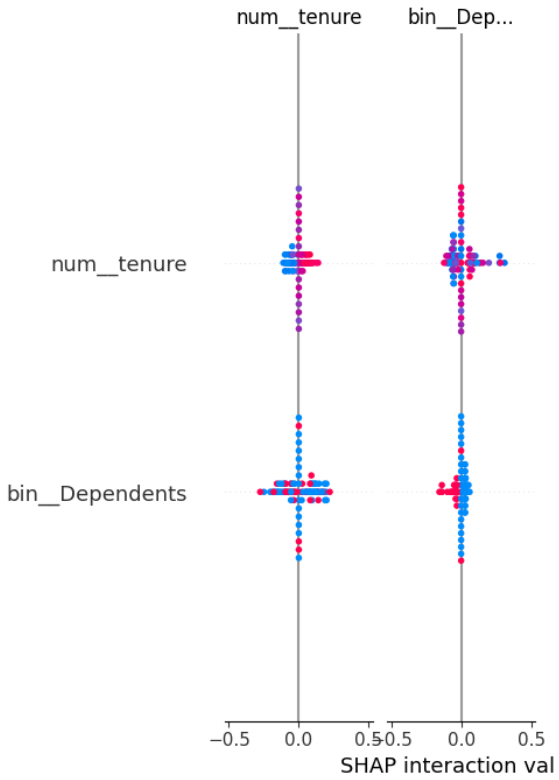

Figura 48: Valores SHAP de interacción entre TENURE y DEPENDENTS en el modelo KNN. *Fuente: Elaboración propia.*

Como se observa en la Figura 48, existe una interacción negativa entre baja permanencia (TENURE) y ausencia de dependientes. En individuos que poseen ambas características, los valores SHAP tienden a ser positivos, lo que indica una mayor contribución a la predicción de abandono. Esta relación refleja un perfil de cliente con baja vinculación al servicio, tanto en términos temporales como personales.

En cambio, cuando el valor de TENURE es alto o el cliente posee personas a cargo, la interacción resulta en valores SHAP más cercanos a cero o negativos, reflejando una menor propensión al churn. Esta información es especialmente relevante para interpretar los resultados del modelo KNN, el cual carece de una estructura explícita como la que poseen los árboles de decisión o la regresión logística.

Aunque KNN no ofrece coeficientes ni reglas claras, la incorporación de herramientas de explicación post-hoc como SHAP permite extraer información útil y visualizar los factores que influyen en las decisiones del modelo, incluso en presencia de relaciones no lineales o interacciones cruzadas.

Interpretación del modelo Árbol de Decisión

El Árbol de Decisión construido en este estudio permite descomponer el espacio de predicción en una jerarquía de reglas lógicas, lo que facilita la identificación de patrones de abandono según combinaciones específicas de características. En este caso, el modelo se entrenó con una profundidad máxima de cinco niveles, conservando una estructura relativamente simple que favorece su interpretabilidad.

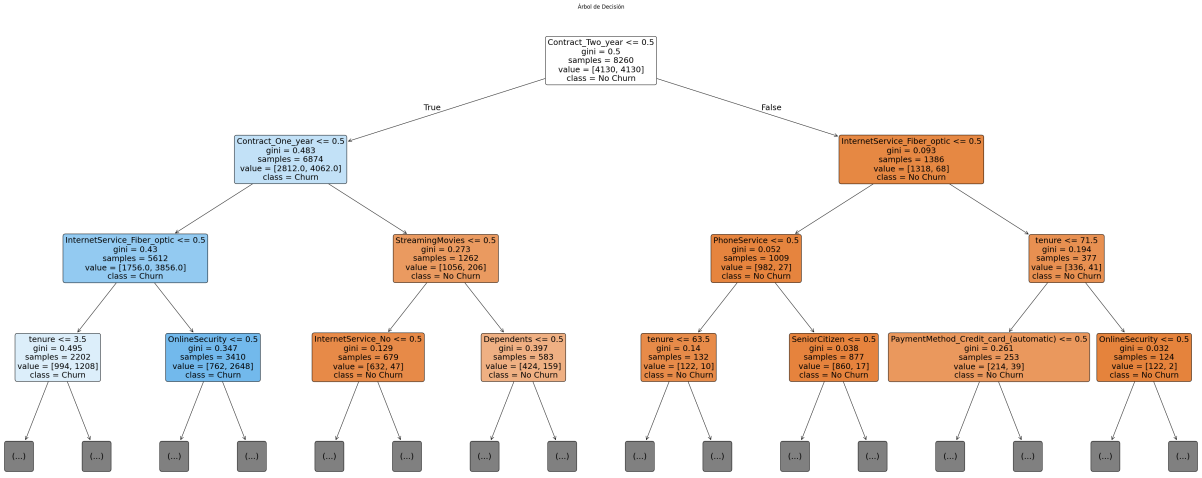

Figura 49: Árbol de Decisión entrenado para predecir churn. *Fuente: Elaboración propia.*

El nodo raíz del árbol corresponde a la variable CONTRACT_TWO_YEAR, lo que indica que esta fue considerada la variable más discriminante para predecir el churn. Los clientes que no tenían un contrato de dos años fueron derivados a una segunda división según la variable CONTRACT_ONE_YEAR. Esto revela que la duración contractual es el principal criterio utilizado por

el modelo para segmentar a los clientes en riesgo.

En ramas posteriores, destacan las variables INTERNETSERVICE_FIBER_OPTIC, ONLINESECURITY, STREAMINGMOVIES, DEPENDENTS, PHONESERVICE y TENURE como atributos utilizados en los nodos intermedios. Este patrón jerárquico refleja cómo el modelo utiliza combinaciones de servicio, permanencia y tipo de contrato para estimar la propensión al abandono.

De forma más específica, se observa que:

- Los clientes sin contrato de dos años y sin contrato de un año, que además tienen servicio de internet por fibra óptica y baja permanencia (TENURE $\leq$ 3,5 meses), son clasificados predominantemente como propensos al churn.
- En contraste, quienes sí tienen contrato de dos años, servicio de fibra y una alta permanencia (TENURE $>$ 71,5), son mayoritariamente clasificados como clientes fieles.
- La ausencia de servicios como ONLINESECURITY o TECHSUPPORT tiende a aparecer en nodos que conducen hacia la clase CHURN, mientras que su presencia se asocia con rutas que terminan en NO CHURN.

La importancia de cada variable en la construcción del árbol fue evaluada mediante el criterio de disminución de impureza (índice Gini). La Tabla 16 resume estos valores.

Variable	Importancia
CONTRACT_TWO_YEAR	0,3956
CONTRACT_ONE_YEAR	0,3301
INTERNETSERVICE_FIBER_OPTIC	0,0833
TENURE	0,0761
ONLINESECURITY	0,0474
INTERNETSERVICE_NO	0,0436
STREAMINGMOVIES	0,0152
DEPENDENTS	0,0037
DEVICEPROTECTION	0,0014
PAYMENTMETHOD_CREDIT_CARD (AUTOMATIC)	0,0013
STREAMINGTV	0,0005
PHONESERVICE	0,0004
GENDER	0,0003
TECHSUPPORT	0,0003
SENIORCITIZEN	0,0002
PARTNER	0,0002
PAPERLESSBILLING	0,0000
ONLINEBACKUP	0,0000
MULTIPLELINES	0,0000
PAYMENTMETHOD_ELECTRONIC_CHECK	0,0000
PAYMENTMETHOD_MAILED_CHECK	0,0000

Tabla 16: Importancia de las variables en el Árbol de Decisión. *Fuente: Elaboración propia.*

La interpretación del árbol muestra cómo ciertas combinaciones de atributos conducen a decisiones consistentes con los patrones de comportamiento observados en el dominio del churn. Su transparencia lo convierte en una herramienta particularmente útil para comunicar decisiones basadas en datos a audiencias no técnicas.

Interpretación del modelo Naive Bayes

El clasificador Naive Bayes utilizado en este estudio se basa en la estimación de probabilidades condicionales para cada clase, asumiendo independencia entre las variables predictoras. Su interpretación se apoya en la comparación de los promedios por clase, considerando que cada variable sigue una distribución gaussiana dentro de cada grupo (CHURN vs NO CHURN).

La Tabla 17 presenta los valores promedio de cada variable para ambas clases, mientras que la Tabla 18 muestra sus respectivas desviaciones estándar. Estos valores permiten identificar aquellas variables que presentan diferencias significativas entre clases, aportando a la discriminación del modelo.

Variable	**No Churn**	**Churn**	**Dif**
TENURE	0,2116	-0,6150	0,8266
CONTRACT_TWO_YEAR	0,3191	0,0229	0,2962
INTERNETSERVICE_NO	0,2709	0,0562	0,2147
ONLINESECURITY	0,3346	0,1284	0,2062
DEPENDENTS	0,3467	0,1535	0,1932
TECHSUPPORT	0,3400	0,1486	0,1914
PARTNER	0,5324	0,3481	0,1843
CONTRACT_ONE_YEAR	0,2557	0,0739	0,1818
PAYMENTMETHOD_CREDIT_CARD(AUTOMATIC)	0,2489	0,1130	0,1359
ONLINEBACKUP	0,3751	0,2636	0,1115
PAYMENTMETHOD_MAILED_CHECK	0,2479	0,1602	0,0877
DEVICEPROTECTION	0,3639	0,2816	0,0823
GENDER	0,5036	0,5036	0
PHONESERVICE	0,9015	0,9106	-0,0091
MULTIPLELINES	0,4109	0,4620	-0,0511
STREAMINGTV	0,3676	0,4421	-0,0745
STREAMINGMOVIES	0,3709	0,4464	-0,0755
SENIORCITIZEN	0,1283	0,2471	-0,1188
PAPERLESSBILLING	0,5397	0,7662	-0,2265
PAYMENTMETHOD_ELECTRONIC_CHECK	0,2521	0,5963	-0,3442
INTERNETSERVICE_FIBER_OPTIC	0,3499	0,7060	-0,3561

Tabla 17: Promedios por clase del modelo Naive Bayes. *Fuente: Elaboración propia.*

Variable	No Churn	Churn	Dif
TENURE	0,9796	0,7818	0,1978
CONTRACT_TWO_YEAR	0,4661	0,1462	0,3199
INTERNETSERVICE_NO	0,4444	0,2280	0,2164
ONLINESECURITY	0,4719	0,3203	0,1516
DEPENDENTS	0,4759	0,3475	0,1284
TECHSUPPORT	0,4737	0,3419	0,1318
PARTNER	0,4989	0,4650	0,0339
CONTRACT_ONE_YEAR	0,4362	0,2539	0,1823
PAYMENTMETHOD_CREDIT_CARD(AUTOMATIC)	0,4324	0,3083	0,1241
ONLINEBACKUP	0,4841	0,4296	0,0545
PAYMENTMETHOD_MAILED_CHECK	0,4318	0,3617	0,0701
DEVICEPROTECTION	0,4811	0,4389	0,0422
GENDER	0,5000	0,4908	0,0092
PHONESERVICE	0,2981	0,2798	0,0183
MULTIPLELINES	0,4920	0,4897	0,0023
STREAMINGTV	0,4821	0,4889	-0,0068
STREAMINGMOVIES	0,4831	0,4877	-0,0046
SENIORCITIZEN	0,3345	0,4206	-0,0861
PAPERLESSBILLING	0,4984	0,4104	0,0880
PAYMENTMETHOD_ELECTRONIC_CHECK	0,4342	0,4836	-0,0494
INTERNETSERVICE_FIBER_OPTIC	0,4769	0,4501	0,0268

Tabla 18: Desviación estándar por clase del modelo Naive Bayes. *Fuente: Elaboración propia.*

Del análisis se desprende que las mayores diferencias entre clases se encuentran en variables como TENURE, donde los clientes que no abandonan presentan valores significativamente más altos (0,21 vs -0,61), y en variables contractuales como CONTRACT_TWO_YEAR (0,319 vs 0,023) y CONTRACT_ONE_YEAR (0,256 vs 0,074), indicando una relación negativa entre duración de contrato y probabilidad de churn.

Asimismo, se identifican diferencias relevantes en el uso de medios de pago: los clientes que abandonan utilizan en mayor proporción ELECTRONIC_CHECK (0,596 vs 0,252) y en menor proporción CREDIT_CARD (AUTOMATIC) (0,113 vs 0,249), lo que sugiere patrones financieros diferenciados. En cuanto a los servicios contratados, variables como ONLINESECURITY, TECHSUPPORT y DEVICEPROTECTION presentan valores promedio notablemente inferiores en la clase CHURN, lo que

indica menor adopción de servicios complementarios entre los clientes que abandonan.

El modelo Naive Bayes, aunque basado en una suposición fuerte de independencia condicional entre variables, logra capturar patrones útiles a partir de estas diferencias distribucionales, con un buen equilibrio entre simplicidad computacional e interpretabilidad.

Interpretación del modelo Random Forest

El modelo Random Forest, al estar compuesto por un conjunto de árboles de decisión entrenados sobre subconjuntos aleatorios de los datos y variables, permite capturar relaciones no lineales y complejas entre las características, mejorando la generalización respecto a árboles individuales. Su desempeño es sólido tanto en precisión como en sensibilidad, pero su interpretabilidad se ve reducida en comparación con modelos simples, debido a la opacidad del ensamble.

La importancia de las variables fue evaluada mediante la métrica de disminución media de la impureza (Gini). La Tabla 19 muestra los valores de importancia relativa de cada predictor. Las variables más influyentes fueron TENURE (0,183), CONTRACT_TWO_YEAR (0,172), INTERNETSERVICE_FIBER_OPTIC (0,109), INTERNETSERVICE_NO (0,104), y CONTRACT_ONE_YEAR (0,096), lo cual destaca el rol central de la permanencia y el tipo de contrato en la predicción del churn.

Variable	Importancia
tenure	0,1830
Contract_Two_year	0,1721
InternetService_Fiber_optic	0,1092
InternetService_No	0,1044
Contract_One_year	0,0966
OnlineSecurity	0,0916
TechSupport	0,0683
PaymentMethod_Electronic_check	0,0461
Dependents	0,0335
OnlineBackup	0,0196
Partner	0,0188
PaymentMethod_Credit_card (automatic)	0,0148
PaymentMethod_Mailed_check	0,0124
DeviceProtection	0,0082
PaperlessBilling	0,0058
StreamingMovies	0,0040
StreamingTV	0,0031
PhoneService	0,0030
MultipleLines	0,0021
gender	0,0016
SeniorCitizen	0,0009

Tabla 19: Importancia de variables en Random Forest. *Fuente: Elaboración propia.*

Para complementar esta interpretación, se exploraron las interacciones entre variables mediante valores SHAP. En particular, se analizó la interacción entre TENURE y DEPENDENTS, dos atributos que mostraron una contribución significativa tanto individual como conjunta.

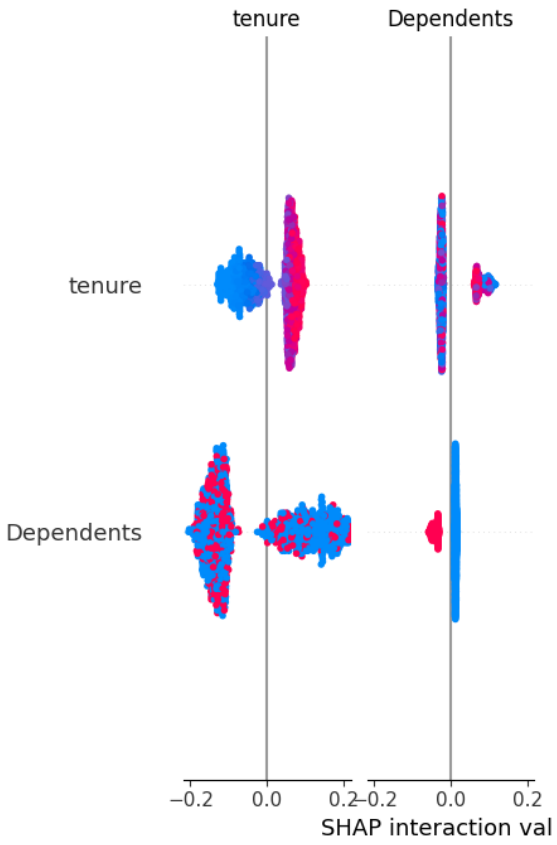

Figura 50: Interacción SHAP entre TENURE y DEPENDENTS en Random Forest. *Fuente: Elaboración propia.*

En la Figura 50 se observa que los clientes con baja permanencia y sin dependientes tienden a presentar mayores valores SHAP positivos, es decir, mayor contribución hacia la predicción de abandono. Esta combinación de factores indica un perfil de usuario de alta vulnerabilidad. En cambio, a medida que aumenta el tiempo de permanencia o cuando existen dependientes, los valores SHAP tienden a reducirse, reflejando una menor propensión al churn.

Esta capacidad del modelo Random Forest para captar interacciones no triviales es una de sus principales fortalezas, aunque se requieran herramientas como SHAP para su explicación detallada. La incorporación de estas visualizaciones permite interpretar parcialmente las decisiones del modelo, y extraer patrones útiles desde una perspectiva de negocio.

Interpretación del modelo Gradient Boosting

El modelo Gradient Boosting combina múltiples árboles de decisión débiles de forma secuencial, optimizando iterativamente la función de pérdida mediante gradientes. Esta metodología permite capturar relaciones complejas y no lineales entre las variables, mostrando usualmente un buen desempeño predictivo, especialmente en contextos con datos estructurados.

A diferencia de los modelos lineales o basados en distancia, Gradient Boosting no ofrece coeficientes explícitos. Por tanto, su interpretación requiere herramientas específicas, como la importancia de variables basada en ganancia de información o el análisis post-hoc con SHAP. La Tabla 20 presenta la importancia relativa de las variables según su contribución agregada al modelo.

Variable	Importancia
Contract_Two_year	0,2827
Contract_One_year	0,2353
tenure	0,1402
InternetService_Fiber_optic	0,0692
OnlineSecurity	0,0606
InternetService_No	0,0594
TechSupport	0,0368
Dependents	0,0238
PaymentMethod_Mailed_check	0,0189
OnlineBackup	0,0181
PaymentMethod_Credit_card (automatic)	0,0140
StreamingMovies	0,0116
gender	0,0066
PhoneService	0,0048
StreamingTV	0,0048
PaperlessBilling	0,0039
Partner	0,0024
MultipleLines	0,0020
PaymentMethod_Electronic_check	0,0019
DeviceProtection	0,0015
SeniorCitizen	0,0015

Tabla 20: Importancia de variables en Gradient Boosting. *Fuente: Elaboración propia.*

Para obtener una interpretación más granular, se utilizó el método SHAP, el cual asigna un valor de contribución a cada variable en cada predicción. La Figura 51 muestra un gráfico de resumen de valores SHAP para las principales variables.

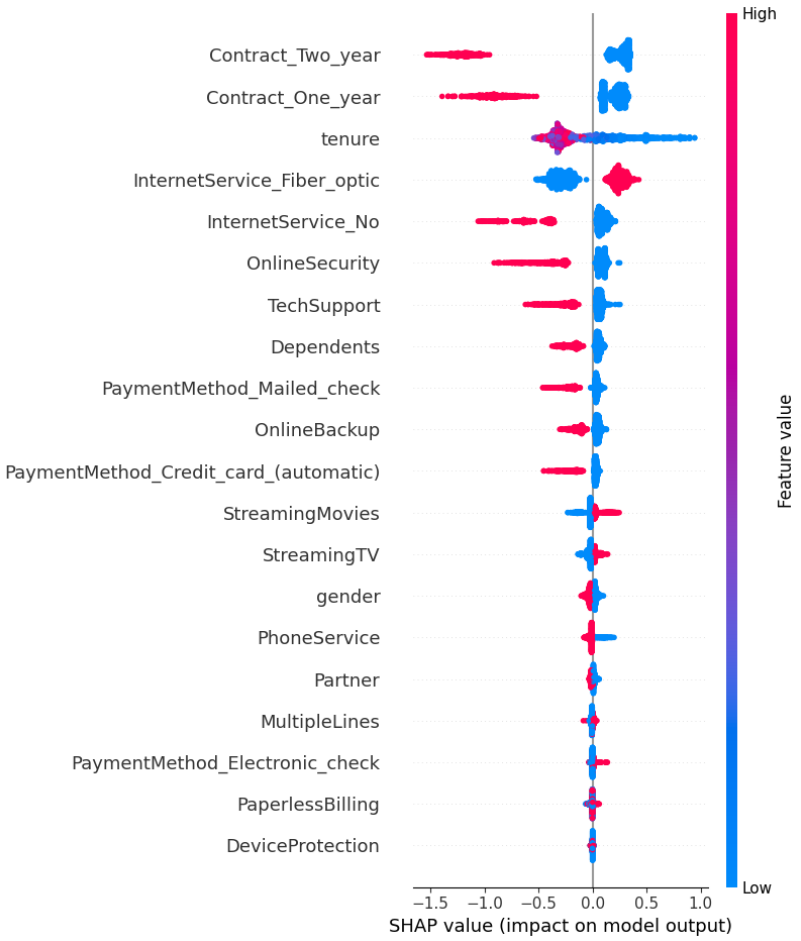

Figura 51: Valores SHAP del modelo Gradient Boosting. Cada punto representa una observación. *Fuente: Elaboración propia.*

Se observa que las variables CONTRACT_TWO_YEAR, CONTRACT_ONE_YEAR y TENURE tienen un impacto significativo sobre las predicciones, lo que coincide con el análisis de importancia. En estos casos, los valores bajos (coloreados en azul) se asocian con valores SHAP positivos —es decir, aumentan la probabilidad de abandono— mientras que los valores altos (en rojo) tienen un efecto amortiguador, reduciendo dicha probabilidad.

Este patrón es evidente, por ejemplo, en la variable CONTRACT_TWO_YEAR, donde su ausencia genera un fuerte empuje hacia la predicción de CHURN, y su presencia contribuye negativamente a esa misma predicción (reflejando retención).

Asimismo, ONLINESECURITY, TECHSUPPORT y DEPENDENTS también presentan impactos diferenciados según sus valores, aunque de menor magnitud. El gráfico SHAP permite además detectar cierta dispersión vertical en las variables más influyentes, lo que sugiere la existencia de interacciones o efectos no lineales que el modelo ha logrado capturar de forma implícita.

En resumen, Gradient Boosting se comporta como un modelo potente y flexible, cuya interpretabilidad puede ser complementada eficazmente mediante el uso de herramientas como SHAP para visualizar la contribución individual y colectiva de cada característica.

Interpretación del modelo XGBoost

XGBoost (*Extreme Gradient Boosting*) es una variante optimizada del algoritmo de boosting por gradiente, diseñada para mejorar tanto la eficiencia computacional como el rendimiento predictivo. Su robustez proviene de técnicas como regularización L1/L2, manejo eficiente de valores faltantes y reducción del sobreajuste mediante subsampling y *shrinkage*.

En este estudio, la importancia de variables se calculó en función de la ganancia de información promedio aportada en los árboles que componen el ensamble. La Tabla 21 resume los resultados obtenidos.

Variable	Importancia
CONTRACT_TWO_YEAR	0,3251
CONTRACT_ONE_YEAR	0,1992
TENURE	0,1411
INTERNETSERVICE_NO	0,0717
INTERNETSERVICE_FIBER_OPTIC	0,0710
ONLINESECURITY	0,0599
TECHSUPPORT	0,0403
PAYMENTMETHOD_MAILED_CHECK	0,0327
DEPENDENTS	0,0356
ONLINEBACKUP	0,0257
PAYMENTMETHOD_CREDIT_CARD (AUTOMATIC)	0,0185
STREAMINGMOVIES	0,0174
PAYMENTMETHOD_ELECTRONIC_CHECK	0,0144
PARTNER	0,0084
PHONESERVICE	0,0082
GENDER	0,0073
PAPERLESSBILLING	0,0054
STREAMINGTV	0,0050
MULTIPLELINES	0,0035
DEVICEPROTECTION	0,0032
SENIORCITIZEN	0,0018

Tabla 21: Importancia de variables en XGBoost. *Fuente: Elaboración propia.*

Los resultados muestran que las variables contractuales (CONTRACT_TWO_YEAR y CONTRACT_ONE_YEAR) junto con la permanencia del cliente (TENURE) explican la mayor proporción de las decisiones del modelo. A estas se suman INTERNETSERVICE_FIBER_OPTIC, INTERNETSERVICE_NO y ONLINESECURITY, que también presentan una contribución relevante. En contraste, variables demográficas como GENDER o SENIORCITIZEN apenas inciden en el resultado.

Para un análisis más detallado, se aplicó la metodología SHAP, que permite cuantificar el impacto marginal de cada variable sobre la predicción individual. La Figura 52 muestra el gráfico de resumen correspondiente.

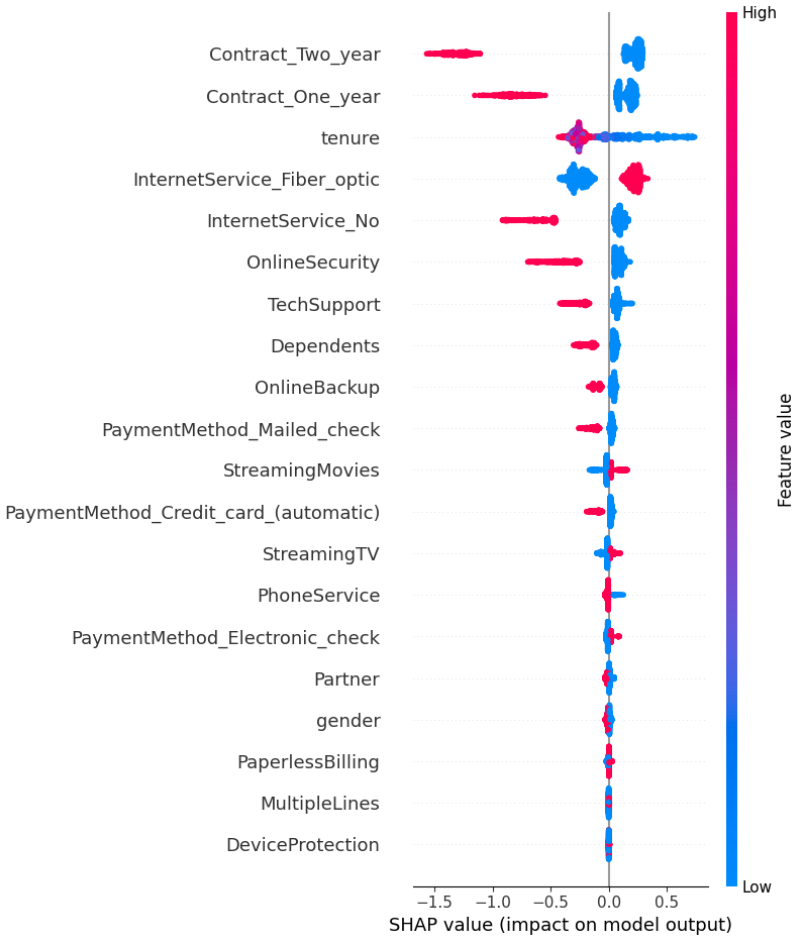

Figura 52: Valores SHAP del modelo XGBoost. Cada punto representa una observación. *Fuente: Elaboración propia.*

Se aprecia que CONTRACT_TWO_YEAR, CONTRACT_ONE_YEAR y TENURE generan los mayores cambios en la probabilidad de churn. En estos casos, valores bajos (azules) tienden a desplazar la predicción hacia el abandono, mientras que valores altos (rojos) refuerzan la permanencia. Por su parte, INTERNETSERVICE_FIBER_OPTIC muestra un patrón inverso, donde su presencia aumenta la propensión al churn, consistente con lo observado en modelos previos.

En síntesis, XGBoost confirma la relevancia de la duración contractual y la permanencia como principales determinantes del churn, complementando este efecto con características relacionadas al tipo de servicio y la seguridad en línea. La capacidad del modelo para capturar interacciones

y relaciones no lineales lo posiciona como una herramienta predictiva de alta potencia, aunque dependiente de técnicas de interpretación post-hoc como SHAP.

Interpretación del modelo LightGBM

LightGBM es un algoritmo de *gradient boosting* optimizado para rendimiento y escalabilidad, que construye árboles de decisión mediante una estrategia *leaf-wise*, lo que le permite encontrar divisiones más profundas con menor pérdida de información. Esta estructura lo hace especialmente eficiente al trabajar con grandes volúmenes de datos y características categóricas.
En este caso, la variable más influyente fue TENURE, seguida por DEPENDENTS, TECHSUPPORT, ONLINESECURITY y ONLINEBACKUP. A diferencia de los modelos anteriores, LightGBM resalta con mayor fuerza la relevancia de las variables relacionadas con soporte técnico y servicios digitales.

Variable	Importancia
TENURE	1995
DEPENDENTS	790
TECHSUPPORT	670
ONLINESECURITY	643
ONLINEBACKUP	542
PAYMENTMETHOD_MAILED_CHECK	516
PAYMENTMETHOD_CREDIT_CARD (AUTOMATIC)	497
INTERNETSERVICE_NO	460
INTERNETSERVICE_FIBER_OPTIC	383
CONTRACT_ONE_YEAR	356
GENDER	321
PARTNER	292
PHONESERVICE	252
CONTRACT_TWO_YEAR	243
STREAMINGTV	228
STREAMINGMOVIES	211
DEVICEPROTECTION	196
PAPERLESSBILLING	166
MULTIPLELINES	140
PAYMENTMETHOD_ELECTRONIC_CHECK	71
SENIORCITIZEN	28

Tabla 22: Importancia de variables en LightGBM. *Fuente: Elaboración propia.*

En el caso de LightGBM, las importancias se reportan como valores enteros, ya que corresponden al número total de divisiones en las que cada variable fue utilizada (*importance type = split*). En otros modelos, como XGBoost o CatBoost, la importancia se calcula a partir de la ganancia promedio de información o se normaliza respecto al total, lo que genera valores decimales.

Para complementar el análisis, se utilizó la explicación basada en SHAP (SHapley Additive exPlanations), que permite evaluar el impacto individual de cada característica en la predicción. La Figura 53 muestra los valores SHAP para las observaciones del conjunto de validación.

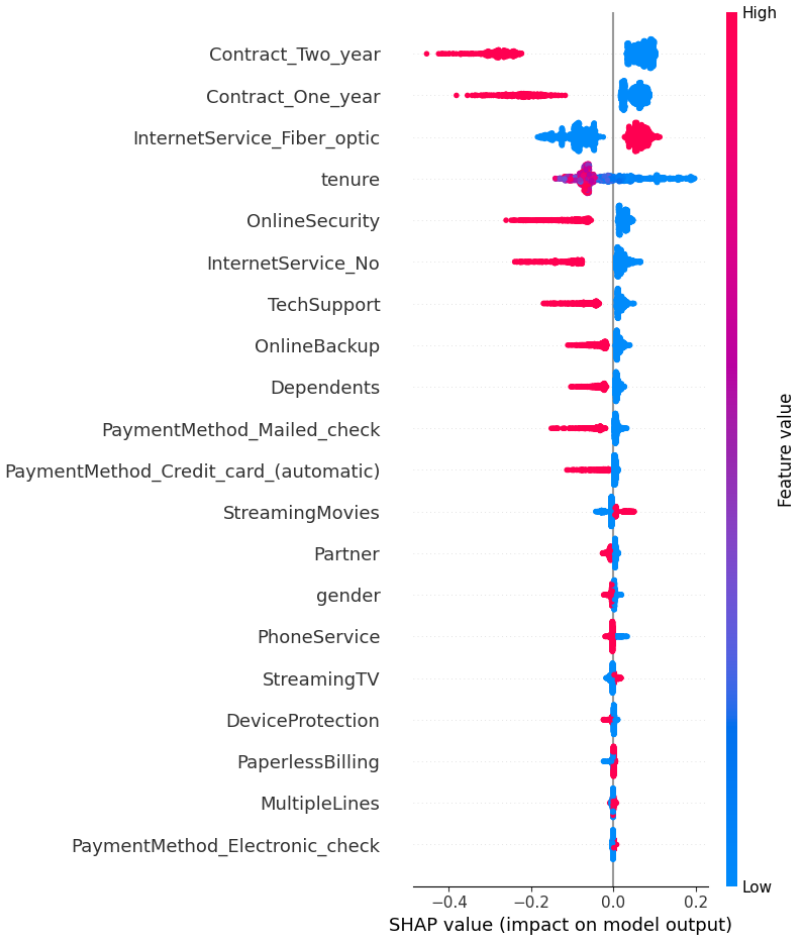

Figura 53: Valores SHAP del modelo LightGBM. Cada punto representa una observación. *Fuente: Elaboración propia.*

En el gráfico, se observa que valores bajos de TENURE (color azul) están fuertemente asociados a valores SHAP positivos, lo que indica una mayor probabilidad de abandono. Lo mismo ocurre con la ausencia de contratos largos (CONTRACT_ONE_YEAR, CONTRACT_TWO_YEAR) y la presencia de INTERNETSERVICE_FIBER_OPTIC, confirmando que estos factores aumentan el riesgo de churn. Por el contrario, clientes con larga permanencia y contratos extendidos tienden a mostrar una baja propensión a abandonar el servicio.

LightGBM también evidencia el papel de los servicios adicionales en la retención de clientes. Características como TECHSUPPORT, ONLINEBACKUP y ONLINESECURITY presentan un patrón

donde su ausencia incrementa la probabilidad de churn. Estos resultados coinciden con lo observado en modelos como XGBoost, reforzando la consistencia del patrón encontrado.

En conjunto, el modelo destaca tanto la duración de la relación contractual como el acceso a servicios digitales de soporte como factores clave en la predicción de abandono.

Interpretación del modelo CatBoost

CatBoost es un algoritmo de *gradient boosting* desarrollado por Yandex, optimizado para trabajar eficientemente con variables categóricas sin requerir preprocesamiento como *one-hot encoding*. Utiliza ordenamientos y estadísticas acumuladas para reducir el *target leakage* y manejar adecuadamente el sobreajuste, siendo especialmente útil en contextos con muchas variables categóricas y relaciones complejas.

En el presente análisis, el modelo CatBoost identificó como variables más influyentes a CONTRACT_TWO_YEAR, INTERNETSERVICE_NO, TENURE y CONTRACT_ONE_YEAR, todas asociadas a la relación contractual del cliente y la disponibilidad de servicios digitales. La Tabla 23 presenta la importancia relativa de cada variable.

Variable	Importancia
CONTRACT_TWO_YEAR	30.48
INTERNETSERVICE_NO	15.07
TENURE	14.17
CONTRACT_ONE_YEAR	11.76
INTERNETSERVICE_FIBER_OPTIC	8.43
ONLINESECURITY	7.26
TECHSUPPORT	4.44
DEPENDENTS	2.74
PAYMENTMETHOD_MAILED_CHECK	2.05
PAYMENTMETHOD_CREDIT_CARD_ (AUTOMATIC)	1.40
ONLINEBACKUP	1.49
PAYMENTMETHOD_ELECTRONIC_CHECK	0.46
PARTNER	0.16
GENDER	0.07
MULTIPLELINES	0.00
PAPERLESSBILLING	0.00
STREAMINGMOVIES	0.00
SENIORCITIZEN	0.00
STREAMINGTV	0.00
DEVICEPROTECTION	0.00
PHONESERVICE	0.00

Tabla 23: Importancia de variables en CatBoost. *Fuente: Elaboración propia.*

Los valores SHAP obtenidos para este modelo (Figura 54) confirman las observaciones anteriores. Variables como CONTRACT_TWO_YEAR y TENURE generan un fuerte efecto en la probabilidad predicha: los valores altos (en rojo) tienden a alejar la predicción del abandono, mientras que los bajos (en azul) aumentan dicha probabilidad. Esto refuerza la idea de que contratos prolongados y relaciones de larga duración tienen un efecto protector frente al churn.

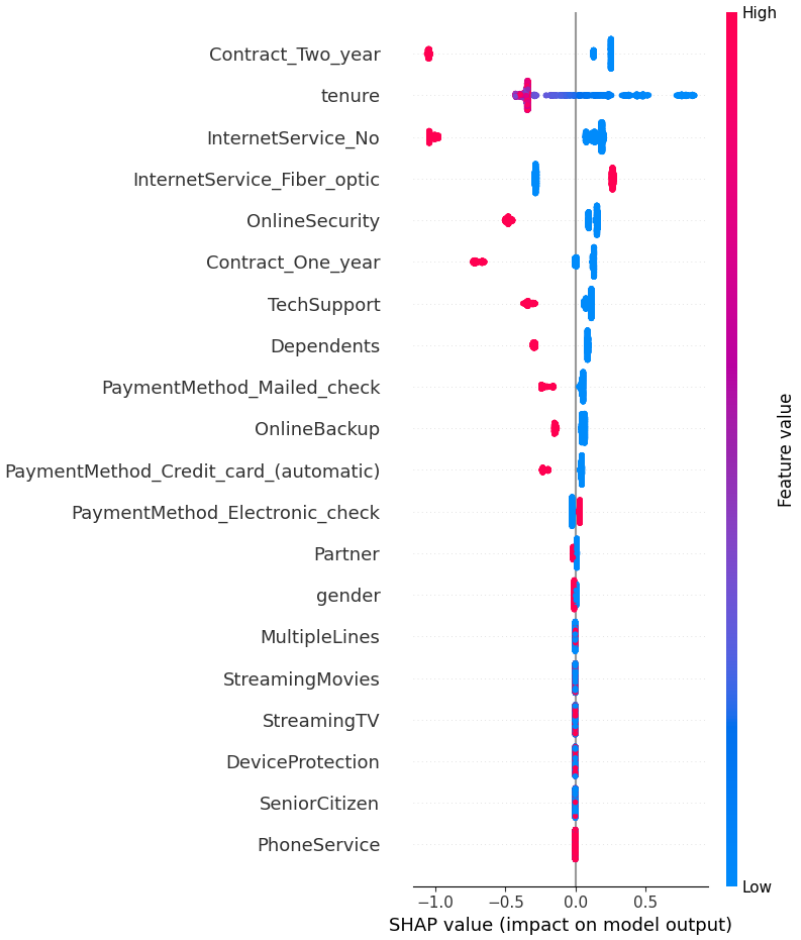

Figura 54: Valores SHAP del modelo CatBoost. Cada punto representa una observación. *Fuente: Elaboración propia.*

Destaca también la relevancia de la variable INTERNETSERVICE_NO, que al igual que en otros modelos boosting, actúa como un fuerte diferenciador entre usuarios con servicios contratados y aquellos que no acceden a internet, probablemente representando perfiles de menor riesgo de abandono.

En contraste, variables como PHONESERVICE, STREAMINGTV, SENIORCITIZEN y DEVICEPROTECTION presentaron una importancia nula en este modelo, lo que podría deberse a que su efecto está completamente capturado por otras variables más influyentes o que su contribución es irrelevante dentro del árbol de decisiones construido por CatBoost.

En resumen, CatBoost refuerza los patrones detectados en modelos anteriores, acentuando el rol de la relación contractual y la duración del servicio como los principales factores explicativos del abandono.

Interpretación del modelo MLP

El Perceptrón Multicapa (MLP) corresponde a una red neuronal artificial de arquitectura completamente conectada y supervisada. En este estudio se configuró con un número limitado de capas y neuronas para evitar sobreajuste, dada la naturaleza tabular del conjunto de datos.

Debido a la estructura no lineal y al carácter de caja negra de los MLP, su interpretabilidad directa es limitada. Por esta razón, se recurre a herramientas de interpretación post-hoc como los valores SHAP, los cuales permiten estimar la contribución marginal de cada variable sobre la predicción final. En este caso, se utilizó SHAP DeepExplainer, adaptado para modelos basados en redes neuronales.

La Figura 55 muestra las interacciones SHAP entre las variables TENURE y DEPENDENTS, seleccionadas como las dos variables más relevantes dentro de la red. El eje horizontal representa el valor de interacción SHAP y cada punto una observación, codificada por color según el valor de la variable (azul: bajo, rojo: alto).

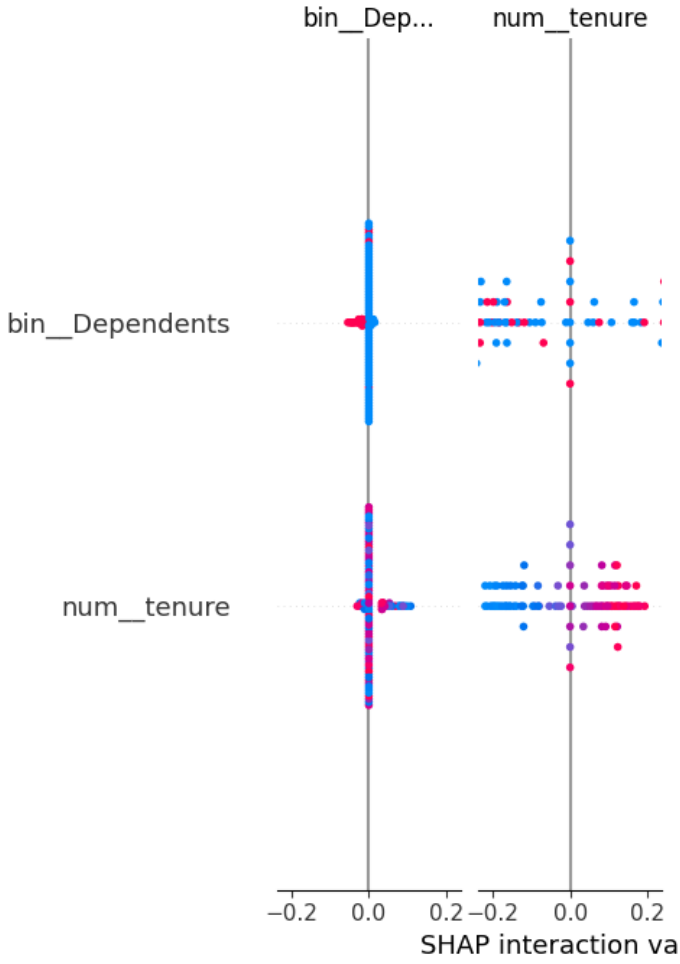

Figura 55: Interacción SHAP entre TENURE y DEPENDENTS en el modelo MLP. *Fuente: Elaboración propia.*

Se observa que ambas variables presentan una relación inversa respecto al riesgo de abandono: valores altos de TENURE tienden a reducir la probabilidad de churn, mientras que la presencia de DEPENDENTS (i.e., tener personas a cargo) también se asocia a menor propensión al abandono. Esto es coherente con la hipótesis de que una mayor estabilidad temporal y familiar se correlaciona con un vínculo más duradero con el proveedor.

Sin embargo, en comparación con modelos más interpretables como Regresión Logística o Árbol de Decisión, los patrones generados por el MLP resultan más difusos y difíciles de traducir a reglas explícitas de negocio. Este modelo se comporta como una caja negra, capaz de capturar relaciones

complejas, pero a expensas de perder explicabilidad directa. En contextos donde la trazabilidad de las decisiones es un requisito clave, esta limitación puede afectar su aplicabilidad práctica.

10.5 Fase de comparación de modelos

Definición de criterios y análisis de robustez

Una vez entrenados todos los modelos considerados, es necesario establecer una metodología rigurosa para compararlos y seleccionar aquellos que presentan un mejor desempeño general. Esta comparación no puede limitarse a una única métrica, dado que cada modelo presenta ventajas y desventajas dependiendo del criterio evaluado. Por esto, se optó por realizar un análisis multicriterio, ponderando distintos aspectos relevantes definidos previamente en el marco teórico. Los criterios de evaluación se organizaron en tres dimensiones principales:

- **Desempeño predictivo:** Esta dimensión busca capturar la capacidad del modelo para identificar correctamente los casos positivos de abandono. Las métricas seleccionadas fueron: **Recall**, **F1-Score**, **ROC-AUC** y **PR-AUC**. Estas permiten evaluar no solo la proporción de abandonos correctamente identificados, sino también el equilibrio general entre sensibilidad y precisión, y el rendimiento en escenarios con clases desbalanceadas. Se privilegió esta dimensión debido a la importancia práctica de minimizar los falsos negativos (clientes que abandonan y no fueron detectados).

Recall	F1	AUC-ROC	PR-AUC
KNN	XGBoost	MLP	Gradient Boosting
MLP	Random Forest	KNN	MLP
Decision Tree	Cat Boost	LightGBM	Cat Boost
XGBoost	MLP	Gradient Boosting	LightGBM
Cat Boost	LightGBM	XGBoost	Regresión Logística
Naive Bayes	Gradient Boosting	Random Forest	KNN
Random Forest	Regresión Logística	Cat Boost	XGBoost
Regresión Logística	Naive Bayes	Regresión Logística	Random Forest
Gradient Boosting	KNN	Naive Bayes	Naive Bayes
LightGBM	Decision Tree	Decision Tree	Decision Tree

Tabla 24: Ranking por desempeño predictivo. *Fuente: Elaboración propia.*

- **Eficiencia computacional:** Se incluyeron tiempos de entrenamiento y testeo como aproximación a los costos de implementación operativa de cada modelo. Si bien estos no impactan directamente en la capacidad predictiva, son relevantes al momento de escalar soluciones en entornos productivos.

Tiempo de entrenamiento	Tiempo de testeo
XGBoost	Decision Tree
Naive Bayes	Naive Bayes
MLP	Logistic Regression
Decision Tree	KNN
Logistic Regression	XGBoost
Random Forest	LightGBM
CatBoost	Random Forest
Gradient Boosting	MLP
KNN	CatBoost
LightGBM	Gradient Boosting

Tabla 25: Ranking por eficiencia computacional (ordenados de más rápido a más lento). *Fuente: Elaboración propia.*

- **Interpretabilidad:** Finalmente, se incorporó una medida subjetiva de interpretabilidad, asociada a la capacidad del modelo para ser comprendido por usuarios no expertos. La siguiente tabla presenta la clasificación adoptada.

Modelo	Puntaje	Justificación
Regresión Logística	3	Modelo lineal con coeficientes directamente interpretables.
Decision Tree	3	Visualizable como un conjunto de reglas lógicas.
Naive Bayes	3	Basado en probabilidades condicionales comprensibles.
KNN	2	Basado en proximidad; comprensible pero difícil de explicar decisiones específicas.
Random Forest	2	Ensamble de árboles; interpretabilidad parcial mediante importancia de variables.
Gradient Boosting	2	Requiere técnicas post-hoc (SHAP, etc.) para explicación.
XGBoost / LightGBM / CatBoost	1	Alta complejidad; difícil de explicar sin herramientas específicas.
MLP (Red Neuronal)	0	Caja negra con múltiples capas y pesos; baja interpretabilidad.

Tabla 26: Puntaje de interpretabilidad asignado a cada modelo. *Fuente: Elaboración propia.*

Para capturar la contribución de cada métrica en una única valoración global, se implementó un sistema de ponderaciones variables. Esto permite simular distintos escenarios de evaluación, donde un usuario podría priorizar por ejemplo interpretabilidad sobre la precisión, o viceversa.

Cada modelo j recibe un puntaje calculado como:

$$\text{Puntaje}_j = \sum_{i=1}^{n} w_i \cdot m_{ij}$$

donde:

- m_{ij}: es el valor de la métrica i (normalizada) para el modelo j.
- w_i: es el peso asignado a esa métrica en la configuración correspondiente.
- n: es el numero total de métricas.

Robustez comparativa de modelos frente a cambios de criterio

Con el objetivo de evaluar la solidez de cada modelo frente a distintas formas de priorizar los criterios de evaluación, se llevó a cabo un análisis de sensibilidad que simula múltiples escenarios de decisión. Para ello, se generaron todas las combinaciones posibles de pesos aplicados a las métricas activas, en incrementos de 5 % y manteniendo la suma total igual a 100 %. Esto dio lugar a una malla exhaustiva de más de 230.000 configuraciones distintas, cada una representando un perfil de usuario con preferencias distintas respecto al balance entre desempeño, eficiencia e interpretabilidad.

Formulación matemática

El análisis de robustez se basó en la combinación de siete métricas de evaluación, agrupadas en tres dimensiones principales:

- **Desempeño predictivo:** *Recall*, *F1-Score*, *ROC-AUC* y *PR-AUC*.
- **Eficiencia computacional:** tiempo de entrenamiento (*Tiempo_train*) y tiempo de inferencia o testeo (*Tiempo_test*).
- **Interpretabilidad:** puntaje asignado de acuerdo con la capacidad del modelo para ser comprendido por usuarios no técnicos.

Cada una de estas métricas fue considerada como un criterio independiente dentro del sistema multicriterio. Con el fin de simular distintos perfiles de priorización, se generaron todas las combinaciones posibles de pesos w_i aplicados a las métricas activas, en incrementos del 5 %, manteniendo la suma total igual a 1.

De esta forma, cada vector de pesos $w \in \mathbb{R}^n$ (con $n = 7$) representa una configuración específica de prioridades, y la totalidad de combinaciones posibles conforma una matriz $W \in \mathbb{R}^{k \times n}$, donde $k = 230{,}230$ corresponde al número total de configuraciones generadas:

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{17} \\ w_{21} & w_{22} & \dots & w_{27} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k1} & w_{k2} & \dots & w_{k7} \end{bmatrix}, \quad \text{con } \sum_{i=1}^{7} w_{ji} = 1,\ w_{ji} \in \{0, 0{,}05, 0{,}10, \dots, 1\}.$$

Sea $M \in \mathbb{R}^{n \times m}$ la matriz de desempeño, donde cada columna m_j contiene los valores normalizados de las siete métricas para el modelo j, y m es el número total de modelos comparados.

El puntaje final de cada modelo bajo una configuración de pesos w se calcula mediante el producto punto:

$$P_j = w^\top m_j = \sum_{i=1}^{7} w_i \, m_{ij}.$$

La multiplicación completa entre las matrices W y M genera una matriz $P \in \mathbb{R}^{k \times m}$ de puntajes finales, donde cada fila representa una configuración de preferencias distinta y cada columna un modelo evaluado:

$$P = W \cdot M.$$

Cada fila de P permite identificar el modelo con el mayor puntaje en esa configuración, y la distribución de estos resultados determina la frecuencia con que cada modelo resulta óptimo bajo las

distintas ponderaciones.

Dimensión	Métricas evaluadas
Desempeño predictivo	Recall, F1-Score, ROC-AUC, PR-AUC
Eficiencia computacional	Tiempo de entrenamiento, Tiempo de testeo
Interpretabilidad	Puntaje de interpretabilidad

Tabla 27: Métricas incluidas en la formulación multicriterio. Fuente: Elaboración propia.

Procedimiento aplicado

En cada configuración de evaluación:

1. Se calculó el puntaje final de cada modelo mediante el producto entre la fila de pesos y la matriz de métricas.

2. Se identificó el modelo con el mayor puntaje.

3. Se registró al modelo ganador para esa configuración.

Este procedimiento permitió analizar cuántas veces cada modelo fue el mejor bajo distintos escenarios, lo que ofrece una medida de ribustez comparativa. Es decir, permite distinguir entre modelo que son altamente dependientes de una configuración específica y aquellos que ofrecen buen rendimiento bajo múltiples perfiles de evaluación.

Resultados del análisis de robustez

A partir de las más de 230.000 combinaciones de ponderaciones generadas, se calculó el puntaje final de cada modelo para cada configuración. Luego, se identificó qué modelo obtuvo el mejor desempeño en cada caso. Este análisis permitió construir una distribución de victorias que refleja qué tan robusto es cada modelo ante cambios en las prioridades de evaluación.

La tabla 28 resume la cantidad de veces que cada modelo fue el mejor (Top 1) y su frecuencia relativa sobre el total de combinaciones.

Modelo	Veces como Top 1	%
MLP	79.487	34,53
XGBoost	54.616	23,72
KNN	54.268	23,57
Decision Tree	20.403	8,86
Regresión Logística	11.944	5,19
Gradient Boosting	4.635	2,01
Random Forest	2.950	1,28
Naive Bayes	1.760	0,76
CatBoost	167	0.007

Tabla 28: Cantidad de veces que cada modelo resultó como Top 1 en las configuraciones evaluadas. *Fuente: Elaboración propia.*

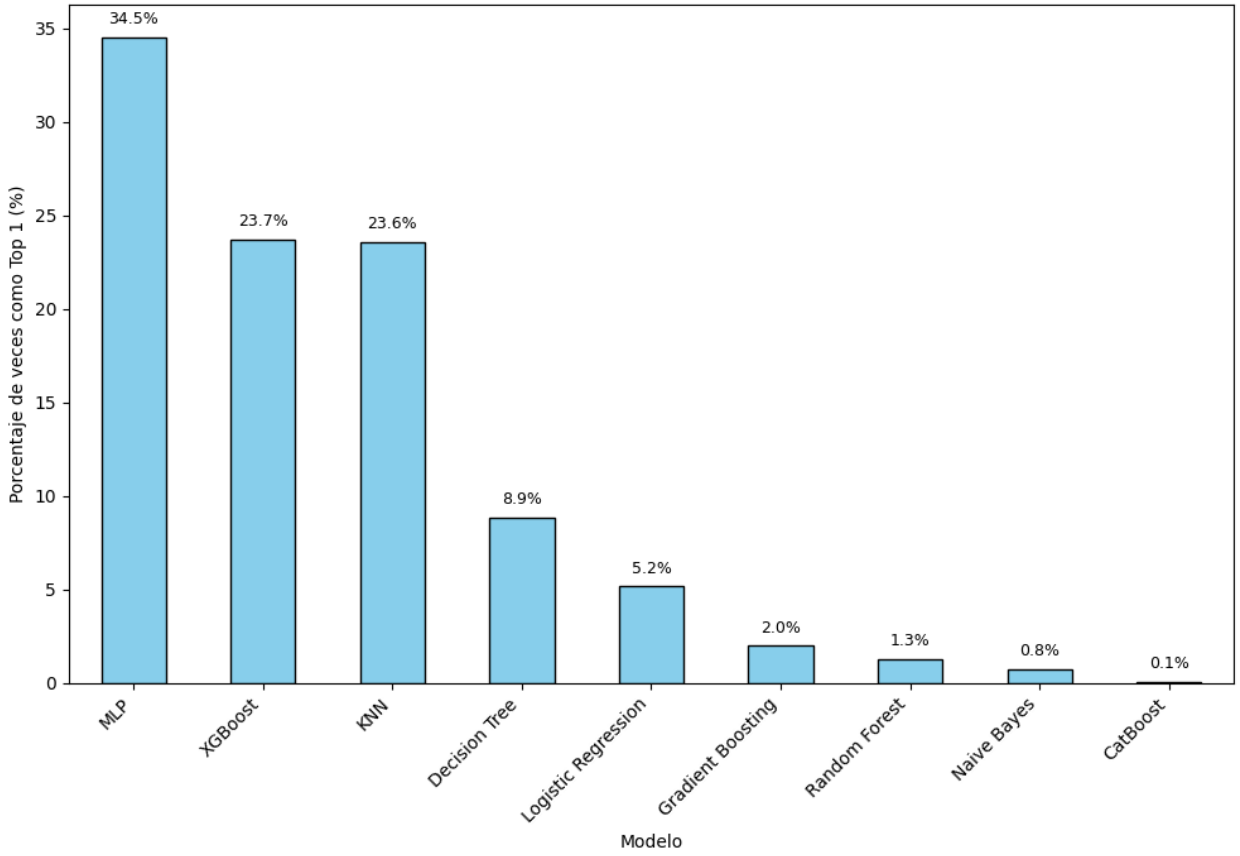

Figura 56: Frecuencia porcentual de cada modelo como Top 1 bajo todas las combinaciones. *Fuente: Elaboración propia.*

El modelo MLP (red neuronal) resultó ser el más robusto, obteniendo el primer lugar en más de un tercio de todas las combinaciones posibles de prioridades. Este resultado es coherente con lo observado en las tablas de desempeño predictivo (Tabla 24) donde el MLP alcanzó valores sobresalientes en Recall, AUC-ROC y PR AUC. Su capacidad para capturar patrones complejos a través de múltiples capas no lineales le permitió adaptarse a configuraciones en las que se privilegiaba la sensibilidad por sobre otros criterios.

Le siguen XGBoost y KNN, ambos con un rendimiento similar cercano al 24 % de victorias. En el caso de XGBoost, su éxito puede atribuirse a su excelente desempeño predictivo combinado con eficiencia computacional razonable (Tabla 25), lo que lo hace competitivo en escenarios donde se equilibran precisión y costos. El KNN, por su parte, mostró valores muy altos de Recall, lo que le otorgó una ventaja significativa en configuraciones donde la sensibilidad tenía mayor peso.

Por otro lado, modelos como Árbol de Decisión y Regresión Logística lograron un rendimiento moderado. Si bien estos no destacaron en desempeño predictivo puro, sí presentan ventajas claras en interpretabilidad (Tabla 26) y tiempo de entrenamiento, lo que los hizo competitivos en configuraciones que valoraban la transparencia del modelo o los costos computacionales.

En el extremo opuesto, CatBoost y Naive Bayes mostraron bajo rendimiento relativo, con menos del 1 % de combinaciones ganadas. En el caso de CatBoost, su baja interpretabilidad (puntaje 1 en la Tabla 26) y su rendimiento moderado en métricas predictivas redujeron su robustez general. Naive Bayes, si bien es altamente interpretable (puntaje 3), mostró resultados débiles en métricas como F1 y PR AUC, lo que lo penalizó fuertemente en configuraciones con énfasis en rendimiento.

Este análisis permite concluir que el MLP y XGBoost no solo son fuertes en escenarios específicos, sino que mantienen un rendimiento competitivo de manera transversal. Por el contrario, modelos como Naive Bayes o CatBoost pueden ser adecuados solo en situaciones muy particulares (por

ejemplo, con fuertes restricciones de interpretabilidad o conjuntos de datos muy simples).

En suma, la robustez observada en este análisis ofrece evidencia para priorizar ciertos modelos en la selección final, dependiendo de los objetivos operacionales de la organización. Un análisis más detallado de estos resultados se presenta en la siguiente sección, donde se discuten posibles compensaciones (trade-offs) y se recomienda un conjunto reducido de modelos para validación en producción.

11 Conclusiones

El presente estudio tuvo como objetivo evaluar comparativamente distintos modelos de *machine learning* aplicados a la predicción de abandono de clientes en el sector telecomunicaciones, utilizando como base el dataset IBM Telco Customer Churn. La metodología se estructuró siguiendo el enfoque CRISP-DM, abarcando desde la comprensión del negocio hasta la evaluación exhaustiva de los modelos entrenados.

Durante el desarrollo del proyecto, se entrenaron múltiples algoritmos supervisados, incluyendo técnicas lineales, modelos de árboles, métodos de ensamble y redes neuronales. Cada uno de ellos fue evaluado mediante un conjunto de métricas previamente definidas, agrupadas en tres dimensiones clave: desempeño predictivo (Recall, F1-Score, ROC AUC, PR AUC), eficiencia computacional (tiempo de entrenamiento y testeo) e interpretabilidad.

Para capturar la diversidad de preferencias que pueden existir en la práctica, se implementó un análisis de robustez que consideró más de 230.000 combinaciones de ponderaciones posibles entre las métricas. Este enfoque permitió simular múltiples escenarios de evaluación, desde aquellos centrados en la sensibilidad predictiva hasta otros que priorizan la explicabilidad o el tiempo de respuesta.

Los resultados de este análisis revelaron que el modelo **MLP (red neuronal)** fue el más robusto, obteniendo el mejor puntaje en más del 34 % de las configuraciones. Le siguieron de cerca XGBoost y KNN, ambos con rendimientos superiores al 23 %, posicionándose como modelos altamente competitivos y versátiles. Estos tres modelos demostraron adaptarse bien a diferentes combinaciones de criterios, lo que sugiere una alta estabilidad en entornos donde los objetivos del negocio pueden variar.

Por otro lado, modelos como la Regresión Logística y el Árbol de Decisión ofrecieron ventajas claras en términos de interpretabilidad y eficiencia, pero sacrificaron precisión en métricas claves, lo que limitó su desempeño global. Finalmente, modelos como CatBoost, Naive Bayes y Random Forest presentaron bajo rendimiento en el análisis de robustez, siendo superados en la mayoría de los escenarios.

En función de estos hallazgos, se concluye que los modelos basados en redes neuronales (MLP) y métodos de ensamble avanzados (XGBoost) son las alternativas más sólidas para abordar el problema de *churn* en telecomunicaciones, especialmente cuando se prioriza la capacidad predictiva y la estabilidad del modelo frente a distintos enfoques de evaluación. No obstante, en contextos donde la interpretabilidad sea un requisito crítico, la Regresión Logística o los Árboles de Decisión pueden seguir siendo opciones válidas.

Este estudio entrega una base metodológica replicable para otros problemas de clasificación binaria con múltiples métricas y prioridades cambiantes, y deja abierta la posibilidad de futuras extensiones, como la incorporación de técnicas de AutoML, ajuste fino de hiperparámetros o análisis en escenarios reales con datos en producción.

12 Discusión

La comparación de modelos presentada en este estudio no se limitó a evaluar el rendimiento predictivo tradicional, sino que incorporó dimensiones adicionales que reflejan consideraciones prácticas reales, como la eficiencia computacional y la interpretabilidad. Esta visión ampliada responde a la necesidad de que los modelos predictivos no solo sean precisos, sino también viables de implementar y comprensibles para los distintos actores involucrados en la toma de decisiones.

Desde el punto de vista del desempeño predictivo, las métricas utilizadas (Recall, F1-Score, ROC AUC y PR AUC) permitieron evaluar con mayor profundidad el comportamiento de los modelos frente a un problema con clases desbalanceadas. En particular, la métrica recall cobró especial relevancia al tratarse de un contexto donde los falsos negativos, es decir, clientes que abandonan y no fueron detectados, tienen un alto costo comercial. Modelos como MLP y KNN mostraron valores particularmente elevados en estas métricas, lo que explica su buen posicionamiento en el análisis de robustez.

En cuanto a la eficiencia computacional, si bien no fue el foco principal del estudio, las métricas de tiempo de entrenamiento y testeo permitieron diferenciar modelos que, a igualdad de precisión, resultan más convenientes para aplicaciones en producción. Modelos como la regresión logística o el árbol de decisión destacaron por su baja complejidad y tiempos de ejecución reducidos, lo que puede justificar su elección en entornos con recursos limitados o con requerimientos de tiempo de respuesta estrictos.

Respecto a la interpretabilidad, la tabla de puntuación asignada a cada modelo permitió visibilizar una dimensión frecuentemente omitida en comparaciones cuantitativas. En contextos regulatorios o comerciales donde se requiere justificar las decisiones basadas en modelos, esta característi-

ca adquiere gran relevancia. Modelos como naive bayes, árbol de decisión y regresión logística sobresalieron en esta dimensión, mientras que otros como MLP, XGBoost y CatBoost presentan dificultades para ser comprendidos sin herramientas adicionales de interpretabilidad post-hoc.

El análisis de robustez permitió integrar todas estas dimensiones mediante un enfoque exhaustivo de ponderaciones variables. En vez de definir arbitrariamente una única combinación de pesos, se exploraron más de 230.000 configuraciones distintas, simulando distintos perfiles de decisión. Este enfoque evidenció qué modelos son estructuralmente más competitivos bajo distintas prioridades, y no solo en escenarios optimizados para su estructura interna.

Los resultados muestran que el modelo MLP no solo lidera en rendimiento predictivo, sino que también mantiene una alta competitividad en un espectro amplio de configuraciones, lo que lo posiciona como una opción robusta. Por su parte, XGBoost demostró ser un modelo versátil, con un buen equilibrio entre precisión y eficiencia. En contraste, modelos como CatBoost y random forest mostraron un desempeño más inconsistente, lo que podría deberse tanto a su sensibilidad a los hiperparámetros como a su menor capacidad de adaptación a escenarios con restricciones prácticas.

Este tipo de análisis aporta evidencia empírica para la toma de decisiones informada, permitiendo balancear precisión, costo e interpretabilidad según los requerimientos específicos del problema y de la organización.

13 Limitaciones

Si bien el presente estudio logró implementar una metodología exhaustiva de comparación entre modelos a través de múltiples métricas y ponderaciones, existen ciertas limitaciones que es necesario declarar.

En primer lugar, el análisis se basa en un único conjunto de datos (IBM Telco Customer Churn), lo cual restringe la generalización de los resultados a otras industrias o contextos. Aunque el dataset es ampliamente utilizado como benchmark, no contempla particularidades de clientes reales de empresas chilenas ni variaciones temporales.

En segundo lugar, la asignación de puntuación a la interpretabilidad se realizó de manera cualitativa y discreta. Aunque basada en la literatura, esta clasificación podría complementarse con métodos más objetivos, como medidas SHAP agregadas, análisis de complejidad del modelo, o encuestas a usuarios.

Además, si bien el análisis de robustez permitió evaluar múltiples configuraciones de evaluación mediante combinaciones de ponderaciones, este no consideró aspectos de incertidumbre en los datos ni sensibilidad frente a perturbaciones externas (por ejemplo, ruido artificial, datos faltantes o cambios en la distribución de entrada).

Por último, el estudio no incorporó criterios económicos o financieros directamente en la evaluación de los modelos, como el costo asociado a falsos positivos, el valor de retención de clientes o el impacto en campañas de fidelización. Estos elementos podrían ser relevantes al momento de trasladar los resultados a contextos de decisión operativa.

Estas limitaciones abren espacio para futuras líneas de trabajo orientadas a validar y extender los resultados obtenidos en este estudio.

14 Anexos

Referencias

Ahn J.-H., Han S. P., y Lee Y.-S. (2006). Customer churn analysis: Churn determinants and mediation effects of partial defection in the korean mobile telecommunications service industry. *Telecommunications Policy*, 30:552–568.

Amin A., Adnan A., y Anwar S. (2023). An adaptive learning approach for customer churn prediction in the telecommunication industry using evolutionary computation and naïve bayes. *Applied Soft Computing*, 137:110103.

Athanassopoulos A. D. (2000). Customer satisfaction cues to support market segmentation and explain switching behavior. *Journal of Business Research*, 47(3):191–207.

Berson A., Smith S., y Thearling K. (2000). *Building Data Mining Applications for CRM*. McGraw-Hill Companies, New York.

Breiman L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Chawla N. V., Bowyer K. W., Hall L. O., y Kegelmeyer W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chen T. y Guestrin C. (2016). Xgboost: A scalable tree boosting system. En *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, p. 785–794, New York, NY, USA. Association for Computing Machinery.

Cover T. y Hart P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Dormann C. F., Elith J., Bacher S., Buchmann C., Carl G., Carré G., Marquéz J. R. G., Gruber B., Lafourcade B., Leitão P. J., Münkemüller T., McClean C., Osborne P. E., Reineking B., Schröder B., Skidmore A. K., Zurell D., y Lautenbach S. (2013). Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, 36(1):27–46.

Dorogush A. V., Ershov V., y Gulin A. (2018). Catboost: gradient boosting with categorical features support.

Fernández A., García S., Galar M., Prati R., Krawczyk B., y Herrera F. (2018). *Learning from Imbalanced Data Sets*.

Freedman D. A. (2005). *Statistical Models: Theory and Practice*. Cambridge University Press.

Friedman J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Ganesh J., Arnold M. J., y Reynolds K. E. (2000). Understanding the customer base of service providers: An examination of the differences between switchers and stayers. *Journal of Marketing*, 64(3):65–87.

Goodfellow I., Bengio Y., y Courville A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Gujarati D. N. y Porter D. C. (2009). *Econometría básica*. McGraw-Hill, Nueva York, 5 edición.

Hastie T., Tibshirani R., y Friedman J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edición.

He H. y Garcia E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Jain A., Duin R., y Mao J. (2000). Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:4–37.

James G., Witten D., Hastie T., y Tibshirani R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer, New York.

Japkowicz N. y Stephen S. (2002). The class imbalance problem: A systematic study1. *Intelligent Data Analysis*, 6(5):429–449.

Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., y Liu T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree.

Keramati A., Jafari R., Aliannejadi M., Ahmadian I., Mozaffari M., y Abbasi U. (2014). Improved churn prediction in telecommunication industry using data mining techniques. *Applied Soft Computing*, 24:994–1012.

Kuhn M. y Johnson K. (2013). *Applied Predictive Modeling*. Springer, New York, NY.

Kutner M. H., Nachtsheim C. J., Neter J., y Li W. (2005). *Applied Linear Statistical Models*. McGraw-Hill/Irwin, New York, 5 edición.

Little R. y Rubin D. (2019). *Statistical Analysis with Missing Data*. Wiley, third edición.

Lundberg S. y Lee S.-I. (2017). A unified approach to interpreting model predictions.

McGill R., Tukey J. W., y Larsen W. A. (1978). Variations of box plots. *The American Statistician*, 32(1):12–16.

Murphy K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

O'Brien R. (2007). A caution regarding rules of thumb for variance inflation factors. *Quality Quantity*, 41:673–690.

Panimalar S., Krishnakumar D., y Kumar D. (2024). Intensified customer churn prediction: Connectivity with weighted multi-layer perceptron and enhanced multipath back propagation: Englsih. *International Journal of Advanced Science and Computer Applications*, 4.

Rish I. (2001). An empirical study of the naïve bayes classifier. *IJCAI 2001 Work Empir Methods Artif Intell*, 3.

Sikri A., Jameel R., Idrees S., y Kaur H. (2024). Enhancing customer retention in telecom industry with machine learning driven churn prediction. *Scientific Reports*, 14.

Silverman B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London.

Swetha P. A. y Dayananda R. B. (2021). Customer churn prediction in telecommunication industry through machine learning based fine-tuned xgboost algorithm. En *Proceedings of the International Conference on Innovative Computing & Communication (ICICC)*. Available at SSRN.

Verbeke W., Dejaeger K., Martens D., Hur J., y Baesens B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218(1):211–229.

Wang C., Rao C., Hu F., Xiao X., y Goh M. (2024). Risk assessment of customer churn in telco using fclcnn-lstm model. *Expert Systems with Applications*, 248:123352.

Wirth R. y Hipp J. (2000). Crisp-dm: Towards a standard process model for data mining. En *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining (PADD)*, London, UK. Springer. Also available at `http://www.crisp-dm.org/`.

Zhang H. y Zhang W. (2024). Application of gwo-attention-convlstm model in customer churn prediction and satisfaction analysis in customer relationship management. *Heliyon*, 10(17):e37229.

Óskarsdóttir M., Bravo C., Verbeke W., Sarraute C., Baesens B., y Vanthienen J. (2017). Social network analytics for churn prediction in telco: Model building, evaluation and network architecture. *Expert Systems with Applications*, 85:204–220.