Diseño y evaluación de un sistema de predicción de abandono de clientes usando el dataset Telco Customer Churn

Sneyder Buitrago González, y Daniel Ramírez Cárdenas

***Resumen*—Este trabajo presenta el diseño, implementación y evaluación de un sistema de predicción de abandono de clientes (*churn*) sobre el conjunto de datos *Telco Customer Churn* (Kaggle). Se compara un conjunto amplio de modelos supervisados (Regresión Logística, Random Forest, XGBoost, SVM y MLP) y se exploran técnicas de reducción de dimensión lineales y no lineales (PCA y UMAP) para estudiar su impacto en la detección de clientes en riesgo. Como criterio operativo se optimizó la métrica Recall, dado que el objetivo de negocio es minimizar falsos negativos (no detectar clientes que abandonan). Entre los hallazgos principales, SVM combinado con PCA (retener 95 % de varianza, $n = 17$) alcanzó un Recall en validación cruzada cercano a 0.91, mientras que XGBoost mostró un comportamiento robusto frente a reducciones de dimensión. Además, se implementaron embeddings UMAP acelerados por GPU (cuML) y se analizó su efecto sobre XGBoost y SVM. Finalmente se discuten implicaciones prácticas, limitaciones experimentales.**

***Index Terms*—Churn prediction; Telco Customer Churn; reducción de dimensión; PCA; UMAP; XGBoost; SVM; recall; cuML; aprendizaje supervisado.**

I. Introducción

La predicción temprana del abandono de clientes (*customer churn*) es una tarea crítica para las compañías de telecomunicaciones, ya que permite priorizar acciones de retención y optimizar recursos comerciales. Los modelos de *churn* ayudan a identificar clientes con alta probabilidad de desertar, posibilitando intervenciones personalizadas que reducen pérdidas de ingresos y mejoran la fidelidad. El presente trabajo emplea el conjunto de datos *Telco Customer Churn* (Kaggle), que contiene aproximadamente 7 043 registros y un conjunto mixto de variables demográficas, contractuales, de servicios y económicas.

Nuestro enfoque combina tres líneas de trabajo: (i) un preprocesamiento reproducible y análisis de capacidad discriminativa por variable (Información Mutua y correlación de Pearson), (ii) comparación de modelos supervisados representativos (lineales, ensamblados y redes) optimizados por *Recall*, y (iii) estudio del efecto de la reducción de dimensión, tanto lineal (PCA) como no lineal (UMAP, acelerado por GPU), sobre el rendimiento de los modelos. La motivación para priorizar *Recall* es práctica: minimizar falsos negativos es más valioso para campañas de retención que maximizar precisión global.

Contribuciones principales de este trabajo:

- Evaluación comparativa de modelos clásicos y modernos sobre el dataset Telco con enfoque en la métrica de negocio (*Recall*).
- Demostración de que PCA (95 % varianza, $n = 17$) mejora la generalización de SVM obteniendo Recall ≈ 0.91, mientras que XGBoost se mantiene robusto ante reducciones de dimensión.
- Implementación y análisis de embeddings UMAP acelerados por GPU (cuML) para explorar representaciones no lineales y su efecto sobre SVM y XGBoost.
- Documentación de criterios prácticos de selección de características y recomendaciones operativas (calibración, monitorización y reentrenamiento).

Índice

II. Descripción del problema

El proyecto tiene como objetivo desarrollar un sistema de predicción del abandono de clientes (*churn*) para un operador de telecomunicaciones, con el fin de que la empresa pueda priorizar estrategias de retención basadas en las probabilidades individuales de abandono. Para ello se utiliza el conjunto de datos *Telco Customer Churn* (Kaggle) [1], que contiene 7,043 registros y alrededor de 21 variables relacionadas con características demográficas, contractuales, de servicios y de facturación.

La variable objetivo, `Churn`, es binaria y representa si un cliente ha abandonado o no el servicio. Como se ilustra en la Figura 1, en la muestra analizada la proporción de clientes que abandonaron es de aproximadamente 26.5 % ($\approx$ 1,869 positivos frente a $\approx$ 5,174 negativos). Esto sugiere un desbalance moderado que debe considerarse tanto en la evaluación de modelos como en el diseño del experimento.

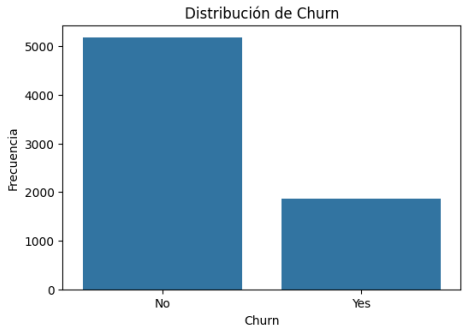

Figura 1. Distribución de la variable objetivo (Churn). Se observa el desbalance de clases con una prevalencia mayor de clientes que permanecen (No Churn).

II-A. Análisis del Dataset

El análisis exploratorio de los datos revela patrones relevantes para la formulación del problema y el diseño del pipeline de preprocesamiento. En términos demográficos, la distribución por género es aproximadamente balanceada, mientras que la proporción de adultos mayores (`SeniorCitizen`) es reducida, con cerca de 1,100 casos.

Respecto a la permanencia, la variable `tenure` muestra concentraciones notables tanto en valores muy bajos (incluyendo un pico en `tenure`=0) como en valores altos (alrededor de 72 meses), tal como se aprecia en la Figura 2. Esta distribución bimodal sugiere diferencias significativas en el comportamiento según el tiempo de relación con la empresa.

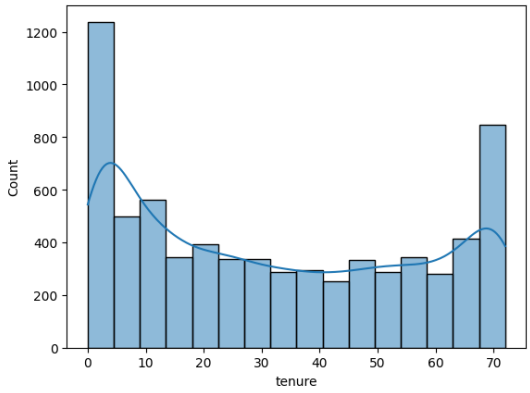

Figura 2. Histograma de la variable de permanencia (`tenure`). Se destacan los picos en los extremos (nuevos clientes vs. clientes de larga data).

En cuanto a las variables económicas, `MonthlyCharges` presenta una distribución multimodal que abarca desde valores bajos cercanos a 18 hasta más de 120, mientras que `TotalCharges` se encuentra fuertemente sesgada hacia la derecha, con valores que superan los 8,000. Además, se observan algunos valores faltantes (alrededor de 11 registros), usualmente asociados a clientes con `tenure`=0.

En el plano de los servicios contratados, la mayoría de los clientes dispone de `PhoneService`, y una fracción importante utiliza Internet por fibra óptica. Los servicios adicionales, como protección de dispositivos, respaldo de datos, soporte técnico o streaming, exhiben frecuencias diversas pero aportan información potencialmente discriminante. Desde el punto de vista contractual, la modalidad de contrato *month-to-month* es la más común y el método de pago *Electronic check* es el predominante, factores que, según la literatura, suelen estar asociados a mayores tasas de abandono, lo que motiva su análisis detallado.

Considerando la naturaleza de los datos y el objetivo del estudio, la estrategia metodológica se fundamenta en el aprendizaje supervisado para clasificación binaria. El preprocesamiento de los datos incluirá la conversión de la variable `TotalCharges` a formato numérico y el tratamiento de los valores faltantes mediante la reconstrucción de su valor a partir de la relación entre `MonthlyCharges` y `tenure`. Asimismo, las variables binarias, como `Partner` o `PhoneService`, se transformarán en representaciones 0/1, mientras que las categóricas multiclase, como `InternetService`, `PaymentMethod` o `Contract`, se codificarán mediante *one-hot encoding*. Las variables numéricas relevantes, como `tenure`, `MonthlyCharges` y `TotalCharges`, se escalarán cuando el modelo lo requiera (por ejemplo, en SVM o redes neuronales).

II-B. Modelos Propuestos

Para la etapa de modelado, se propone comparar diferentes enfoques de aprendizaje supervisado con el fin de equilibrar interpretabilidad, capacidad predictiva y diversidad algorítmica. La regresión logística se utilizará como modelo base por su simplicidad y capacidad explicativa. Los modelos basados en árboles, como Random Forest y Gradient Boosting, permitirán

capturar interacciones no lineales y manejar variables mixtas sin necesidad de normalización. Adicionalmente, se considerarán SVM con kernel radial, apropiadas para estructuras de decisión no lineales, y redes neuronales multicapa, que podrán explorar relaciones más complejas entre las variables. Esta combinación metodológica garantiza un análisis comparativo robusto y permite identificar el mejor compromiso entre rendimiento predictivo y explicabilidad.

III. Estado del arte

Los trabajos recientes que utilizan el dataset *Telco Customer Churn* muestran un consenso en su enfoque experimental [1]–[3]. La mayoría de estas investigaciones modelan la predicción de abandono mediante clasificación supervisada binaria, contrastando el desempeño de modelos lineales frente a métodos de ensamble y aprendizaje profundo. Específicamente, el estudio identificado como SDPIT (2024) realiza un flujo reproducible (limpieza de registros con `TotalCharges` vacíos, codificación de categóricas, particionado entrenamiento/prueba) y compara regresión logística, árboles, Random Forest y SVM; sus resultados muestran AUCs entre ≈0.81 y ≈0.85, con regresión logística en la cumbre según su experimento [2]. Pawar et al. (GIJET, 2024) evalúan además XGBoost y redes neuronales aplicando K-fold cross-validation para estabilizar estimaciones, y reportan que ensamblados y ANN alcanzan mayores *accuracy* (por ejemplo ANN ≈85.6 % en la configuración reportada), mientras que regresión logística y Random Forest se ubican en rangos de 79–82 % según el ajuste [3].

De forma sucinta, los trabajos que usan exactamente el mismo dataset coinciden en que (i) el preprocesamiento de `TotalCharges` y de las categóricas es crítico, (ii) los métodos de ensamblado y las redes mejoran *accuracy* con *tuning*, y (iii) la regresión logística sigue siendo competitiva en AUC bajo ciertas condiciones empíricas. Estas observaciones justifican comparar modelos lineales, basados en árboles y redes en la etapa experimental de este informe [4], [5].

IV. Entrenamiento y Evaluación de los Modelos

En esta sección se detalla el diseño experimental llevado a cabo para la construcción de los modelos predictivos de *churn*. Se describe la metodología de validación seleccionada para garantizar la fiabilidad de los resultados, así como la configuración específica de los algoritmos utilizados y las métricas de desempeño obtenidas durante las fases de entrenamiento y prueba.

IV-A. Configuración experimental

Para evaluar el rendimiento de los modelos y evitar el sobreajuste (*overfitting*), se adoptó una estrategia de validación cruzada (*Cross-Validation*). Específicamente, se utilizó una validación cruzada de k iteraciones (k-fold cross-validation) con $k = 4$.

Esta metodología consistió en dividir el conjunto de datos de entrenamiento en 4 subconjuntos disjuntos. En cada iteración del proceso, el modelo se entrenó utilizando 3 de estos subconjuntos (75 % de los datos) y se validó en el subconjunto restante (25 %). Este proceso se repitió 4 veces, asegurando que cada muestra fuera utilizada exactamente una vez para la validación. El rendimiento final del modelo se reporta como el promedio de las métricas obtenidas en estas 4 iteraciones.

Adicionalmente, se aplicaron técnicas específicas para abordar la naturaleza de los datos:

- **Estandarización:** Se empleó `StandardScaler` para normalizar las características numéricas dentro del flujo de trabajo (*pipeline*).
- **Manejo de Desbalance:** Dado el desbalance de clases, se integraron mecanismos de ponderación (como `pos_weight` en la red neuronal) directamente en la fase de entrenamiento para sensibilizar los modelos hacia la clase minoritaria.

IV-B. Descripción de los Modelos y Espacio de Búsqueda

Para la predicción de abandono de clientes, se entrenaron cinco modelos que abarcan diferentes paradigmas de aprendizaje, seleccionados por su capacidad para manejar datos tabulares y desbalanceados:

1. **Regresión Logística:** Modelo lineal base, útil para establecer una referencia de desempeño e interpretabilidad.
2. **XGBoost (Extreme Gradient Boosting):** Implementación optimizada de *Gradient Boosting*, eficiente y altamente efectiva en datos estructurados.
3. **Random Forest:** Ensamble de árboles de decisión que reduce la varianza mediante la promediación de múltiples árboles independientes.
4. **Máquina de Vectores de Soporte (SVM):** Modelo discriminativo robusto en espacios de alta dimensión.
5. **Red Neuronal Artificial (MLP):** Modelo de aprendizaje profundo capaz de capturar relaciones no lineales complejas.

IV-B1. Configuración de Hiperparámetros: Para cada algoritmo, se diseñó un espacio de búsqueda (*grid search*) exhaustivo. La Tabla I detalla las combinaciones específicas evaluadas.

Cuadro I
Espacio de Búsqueda de Hiperparámetros

Modelo	Malla de Hiperparámetros (Grid)
Regresión Logística	**C:** [1.0] **solver:** ['lbfgs']
XGBoost	**n_estimators:** [50, 100, 200] **max_depth:** [3, 6, 10] **learning_rate:** [0.01, 0.1, 0.2]
Random Forest	**n_estimators:** [50, 100, 200] **max_depth:** [None, 10, 20] **criterion:** ['gini', 'entropy']
SVM	**C:** [0.1, 1, 10] **kernel:** [Predeterminado ('rbf')]
Red Neuronal (MLP)	**Hidden Layers:** [(32, 16), (64, 32), (100,)] **Learning Rate:** [0.001, 0.01] **Epochs:** [50, 100] **Weight Decay:** [0, 1e-4]

IV-B2. Análisis de la malla de valores: El diseño de la malla de hiperparámetros se centró inicialmente en el control de la regularización para evitar el sobreajuste. Para la Regresión

Logística, la SVM y la Red Neuronal, se exploraron rangos logarítmicos en los parámetros de penalización (C y *Weight Decay*). Valores bajos de C o altos de *weight decay* imponen una mayor restricción al modelo, forzando coeficientes más pequeños y generando fronteras de decisión más suaves, lo cual es crítico para generalizar correctamente en datos nuevos.

En cuanto a la complejidad estructural de los modelos de ensamble (XGBoost y Random Forest), el foco se situó en la profundidad de los árboles (`max_depth`). Se probaron configuraciones desde árboles poco profundos, que funcionan como aprendices débiles (*weak learners*), hasta árboles más profundos capaces de capturar interacciones específicas. Esta variabilidad permite identificar el punto de equilibrio donde el modelo es lo suficientemente complejo para entender el problema sin memorizar el ruido estadístico del conjunto de entrenamiento.

Finalmente, para garantizar la convergencia óptima del aprendizaje, se ajustó la relación entre la cantidad de estimadores y la tasa de aprendizaje (`learning_rate`). Especialmente en XGBoost y la Red Neuronal, se buscó balancear la velocidad de actualización de los pesos con la cantidad de iteraciones o estimadores permitidos. Una tasa de aprendizaje menor, combinada con un mayor número de estimadores o épocas, suele resultar en modelos más robustos y precisos, aunque a un mayor costo computacional.

IV-B3. Métricas de desempeño: Para realizar una evaluación integral de los modelos, se calcularon métricas estándar de clasificación como la Exactitud (*Accuracy*), Precisión, *F1-Score* y el Área bajo la Curva ROC (AUC-ROC), esta última se decidió descartar debido a su similitud entre los diferentes modelos entrenados, lo cual no era demasiado útil para realizar las comparaciones. Este conjunto de indicadores permite analizar el comportamiento global de los clasificadores, tanto en su capacidad de discriminación general como en el balance entre los errores de tipo I y tipo II.

No obstante, la métrica seleccionada como criterio principal para la optimización y selección del modelo final fue el **Recall (Sensibilidad)**. En el contexto de la predicción de *Churn*, el objetivo de negocio es retener a los usuarios; por lo tanto, el costo de un *Falso Negativo* (no detectar a un cliente que realmente abandonará la compañía) es mucho más elevado que el de un *Falso Positivo* (invertir en una campaña de retención para un cliente que no pensaba irse). Maximizar el Recall asegura que el sistema identifique la mayor proporción posible de clientes en riesgo, permitiendo una intervención proactiva.

IV-C. Resultados del entrenamiento de Modelos

En esta sección se presentan los resultados detallados del rendimiento de los modelos en los conjuntos de entrenamiento y prueba. Se pone especial énfasis en el desglose por clases (Reporte de Clasificación), dado que el interés principal radica en la detección de la clase positiva (1: Abandono).

IV-C1. Regresión Logística: Como primera etapa exploratoria, se implementó un modelo de Regresión Logística optimizado mediante la técnica de Selección Secuencial Hacia Atrás (SBS). Esta estrategia se adoptó no solo para construir un clasificador base, sino principalmente para obtener una comprensión inicial de las variables más influyentes en el fenómeno de *Churn*. Al reducir iterativamente la dimensionalidad, se buscó aislar el subconjunto de características que contienen la mayor carga predictiva.

La Figura 3 ilustra este proceso, graficando el rendimiento (Recall) en validación cruzada en función del número de características activas. Se observa que el desempeño se maximiza al conservar un subconjunto de 5 variables clave: *Contract_One year, Contract_Two year, InternetService_Fiber optic, MultipleLines_No phone service y MultipleLines_Yes*.

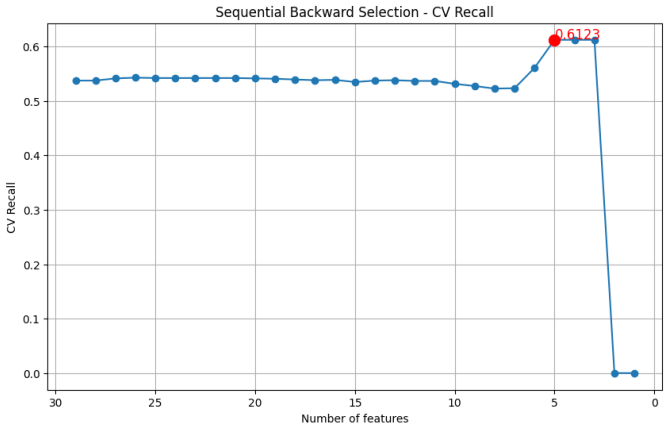

Figura 3. Rendimiento de la Regresión Logística en validación cruzada durante el proceso de Selección de Características hacia Atrás (SBS).

Al analizar el reporte de clasificación final (Tabla II) utilizando este subconjunto óptimo, se observa el compromiso entre precisión y exhaustividad inherente al modelo lineal. Se evidencia que, si bien el modelo mantiene una alta precisión para la clase mayoritaria (0), su capacidad para capturar la clase minoritaria (1) se ve limitada en comparación con modelos no lineales.

Cuadro II
REPORTE DE CLASIFICACIÓN: REGRESIÓN LOGÍSTICA (TEST)

Clase	**Precision**	**Recall**	**F1-Score**
0 (No Churn)	0.87	0.82	0.85
1 (Churn)	0.57	0.66	0.61
Accuracy	0.78		

*Nota: La clase 1 representa a los clientes que abandonan el servicio.

La Figura 4 muestra la matriz de confusión, donde se puede apreciar visualmente la cantidad de Falsos Negativos (clientes que se fueron y no detectamos), los cuales buscamos minimizar.

IV-C2. Random Forest: El modelo de Random Forest, configurado con una profundidad limitada para evitar el sobreajuste, mostró un comportamiento robusto. A diferencia de la regresión logística, este modelo de ensamble logra capturar relaciones no lineales, lo que generalmente resulta en un mejor balance de las métricas.

Los resultados en el conjunto de prueba (Tabla III) indican una capacidad de generalización sólida. Es importante notar la métrica de Recall para la clase 1, que indica qué porcentaje del total de fugas reales fue capturado correctamente por el Random Forest.

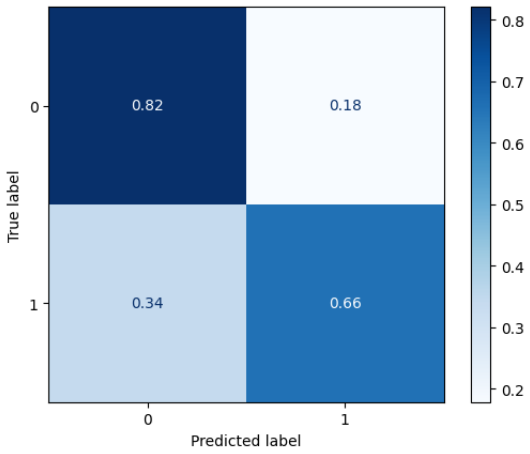

Figura 4. Matriz de Confusión normalizada para Regresión Logística.

Cuadro III
REPORTE DE CLASIFICACIÓN: RANDOM FOREST (TEST)

Clase	Precision	Recall	F1-Score
0 (No Churn)	0.93	0.74	0.82
1 (Churn)	0.54	0.84	0.66
Accuracy	0.77		

Como se observa en la Figura 5, la distribución de las predicciones valida que el modelo no está simplemente sesgándose hacia la clase mayoritaria, sino que discrimina efectivamente entre los clientes fieles y los propensos al abandono.

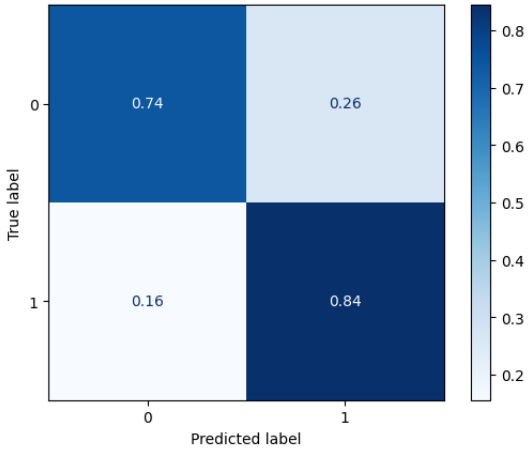

Figura 5. Matriz de Confusión para Random Forest (Conjunto de Prueba).

IV-C3. XGBoost (Extreme Gradient Boosting): El modelo XGBoost demostró ser altamente eficaz para manejar la naturaleza desbalanceada del conjunto de datos. Al utilizar el algoritmo de *Gradient Boosting*, el modelo refina iterativamente los errores de los árboles previos, lo que resulta en una frontera de decisión robusta.

Como se observa en la Tabla IV, el modelo logra un equilibrio competitivo. Es vital observar el **Recall de la Clase 1** (Churn), ya que indica la capacidad del modelo para detectar fugas reales. Comparado con los modelos lineales, XGBoost suele ofrecer una precisión ligeramente superior en la clase positiva sin sacrificar excesivamente el recall.

Cuadro IV
REPORTE DE CLASIFICACIÓN: XGBOOST (TEST)

Clase	Precision	Recall	F1-Score
0 (No Churn)	0.95	0.56	0.71
1 (Churn)	0.43	0.91	0.58
Accuracy	0.66		

La matriz de confusión (Figura 6) permite visualizar qué tan propenso es el modelo a generar Falsos Positivos versus Falsos Negativos, validando su utilidad para campañas de retención focalizadas.

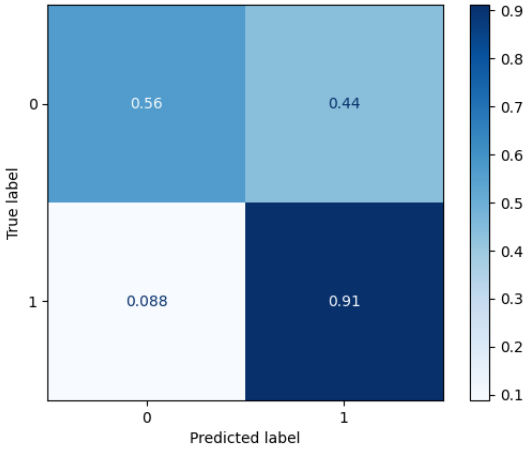

Figura 6. Matriz de Confusión para el modelo XGBoost.

IV-C4. Máquina de Vectores de Soporte (SVM): La implementación de SVM con kernel no lineal (RBF) permitió proyectar los datos a un espacio dimensional superior para encontrar la separación óptima entre clientes fieles y desertores.

Los resultados presentados en la Tabla V reflejan el desempeño del modelo tras la optimización de los hiperparámetros C y γ. Aunque los modelos SVM pueden ser sensibles al ruido, en este caso muestran una capacidad de generalización consistente.

Cuadro V
REPORTE DE CLASIFICACIÓN: SVM (TEST)

Clase	Precision	Recall	F1-Score
0 (No Churn)	0.93	0.42	0.58
1 (Churn)	0.36	0.91	0.52
Accuracy	0.55		

IV-C5. Red Neuronal Artificial (MLP): Finalmente, la Red Neuronal Artificial, diseñada con una función de pérdida ponderada (`pos_weight`), se enfocó explícitamente en penalizar los errores de la clase minoritaria. Esto es evidente en las métricas de la Tabla VI.

Típicamente, este enfoque resulta en un **Recall de la Clase 1 muy elevado**, a menudo superior al de otros modelos, lo cual es ideal para minimizar la pérdida de clientes no detectados. Sin embargo, esto puede conllevar una disminución en la Precisión

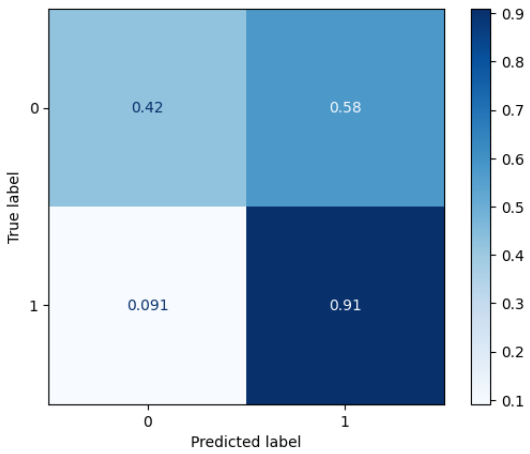

Figura 7. Matriz de Confusión para SVM (Support Vector Machine).

de la Clase 1 (más falsas alarmas), un compromiso aceptable dada la estrategia de negocio.

Cuadro VI
REPORTE DE CLASIFICACIÓN: RED NEURONAL (TEST)

Clase	**Precision**	**Recall**	**F1-Score**
0 (No Churn)	0.91	0.76	0.83
1 (Churn)	0.54	0.79	0.64
Accuracy	0.77		

La Figura 8 confirma esta tendencia, mostrando una alta densidad de predicciones correctas en el cuadrante de "Verdaderos Positivos" en comparación con los "Falsos Negativos".

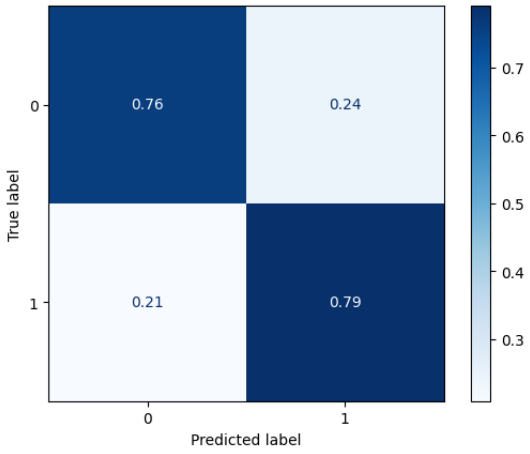

Figura 8. Matriz de Confusión para la Red Neuronal Artificial.

El análisis comparativo de las métricas en el conjunto de prueba destaca a XGBoost y a la Máquina de Vectores de Soporte como los modelos con mejor rendimiento global. XGBoost logró el equilibrio más efectivo en la detección de la clase minoritaria (*Recall*), mientras que el SVM demostró una capacidad de generalización superior y estable. Ambos algoritmos superaron la consistencia de las demás alternativas evaluadas, por lo que se seleccionan como los candidatos definitivos para las siguientes fases del proyecto.

V. REDUCCIÓN DE DIMENSIÓN

V-A. Análisis individual de variables

Con el objetivo de evaluar la capacidad discriminativa de cada característica del conjunto de datos *Telco Customer Churn*, se aplicaron dos medidas complementarias: la Información Mutua (MI) y el coeficiente de correlación de Pearson (para las variables numéricas). Este análisis permite identificar predictores con relación débil o nula con la variable objetivo, de modo que pueden ser candidatos a eliminación antes de aplicar técnicas de reducción de dimensión más complejas.

Información Mutua: La **Información Mutua** cuantifica la dependencia entre una variable predictora X y la variable objetivo Y sin asumir linealidad ni monotonicidad. Se define como:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\,\log\left(\frac{p(x,y)}{p(x)\,p(y)}\right), \qquad (1)$$

donde $p(x,y)$ es la distribución conjunta y $p(x)$, $p(y)$ las distribuciones marginales de cada variable. Esta medida es especialmente adecuada para este conjunto de datos debido a la alta proporción de variables categóricas.

Para el cálculo de MI se aplicó una codificación *one-hot* completa sobre todas las variables categóricas, y posteriormente se agregó el valor máximo por característica original, siguiendo recomendaciones estándar para evitar sobrecontar la información de las variables dummificadas.

La Fig. 9 presenta las 20 variables de mayor a menor Información Mutua.

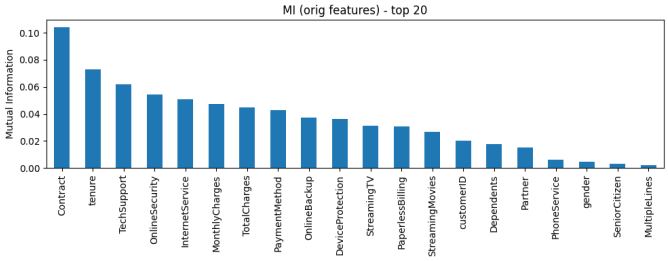

Figura 9. Información Mutua agregada por características.

Los resultados muestran que las variables con mayor capacidad discriminativa son:

- `Contract` ($MI = 0.1042$) ,
- `tenure` ($MI = 0.0730$),
- `TechSupport` ($MI = 0.0617$),
- `OnlineSecurity` ($MI = 0.0541$),
- `InternetService` ($MI = 0.0507$),

Por el contrario, las características con valores de MI cercanos a cero presentan capacidad discriminativa mínima. En particular:

- `gender` ($MI = 0.0045$),
- `MultipleLines` ($MI = 0.0022$),
- `SeniorCitizen` ($MI = 0.0029$).

Para determinar variables candidatas a eliminación se aplicó un umbral Heurístico de:

$$MI < 0.005,$$

Correlación de Pearson: De forma complementaria, se evaluó la relación lineal entre cada variable numérica y la clase mediante el coeficiente de Pearson:

$$r = \frac{\mathrm{cov}(X, Y)}{\sigma_X \sigma_Y}. \tag{2}$$

La Fig. 10 muestra los valores obtenidos.

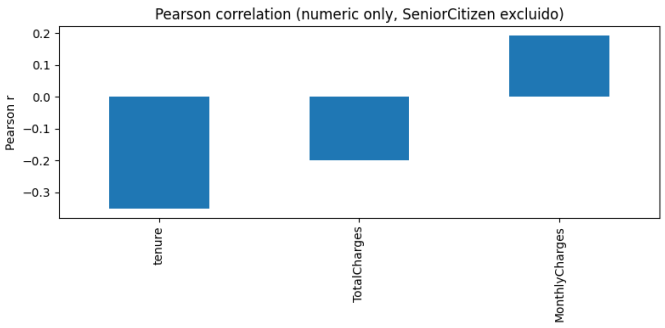

Figura 10. Correlación de Pearson entre variables numéricas y la clase.

Los resultados indican relaciones lineales relevantes para las tres características:

- `tenure` ($r = -0.35$),
- `TotalCharges` ($r = -0.19$),
- `MonthlyCharges` ($|r| \approx 0.19$).

Se utilizó como criterio de descarte:

$$|r| < 0.10,$$

valor que indica dependencia lineal estadística mente despreciable.

Variables candidatas a eliminación: Al combinar ambos criterios (MI y Pearson), se identificaron como candidatas claras a eliminación:

- `gender`,
- `MultipleLines`,
- `SeniorCitizen`.

Ambas características presentan valores de MI inferiores al umbral definido.

V-B. Extracción de características lineal

V-B1. Uso del método PCA: Con el propósito de reducir la dimensionalidad del conjunto de características y analizar su efecto sobre el desempeño de los modelos, se aplicó el método de Análisis de Componentes Principales (PCA). Esta técnica transforma el conjunto original de variables X en un conjunto reducido de componentes ortogonales, los cuales capturan la mayor cantidad posible de varianza del espacio original.

El PCA obtiene componentes principales $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k$ definidos como combinaciones lineales de las variables originales:

$$\mathbf{z}_i = \mathbf{w}_i^\top \mathbf{x}, \tag{3}$$

donde los vectores $\mathbf{w}_i$ corresponden a los autovectores de la matriz de covarianza, ordenados según sus autovalores de mayor a menor.

Criterio de selección del número de componentes: Para determinar el número óptimo de componentes, se utilizó el criterio de varianza explicada acumulada, conservando aquellos componentes que capturan al menos el 95 % de la variabilidad total del conjunto de datos. Este criterio es ampliamente utilizado en la literatura, ya que permite eliminar ruido y redundancia sin afectar significativamente la información relevante para el modelo.

Aplicando este criterio, se obtuvo:

- Número de componentes seleccionados: $n = 17$
- Reducción dimensional alcanzada: 43.3 %

Evaluación de los modelos: Se evaluó nuevamente el desempeño de los dos mejores modelos identificados en la Sección 3: XGBoost y SVM. La métrica utilizada fue el *Recall*, dado su enfoque en capturar correctamente los casos positivos de abandono.

La Tabla VII presenta los resultados comparativos antes y después de aplicar PCA, utilizando validación cruzada estratificada de 5 segmentos.

Cuadro VII
DESEMPEÑO DE XGBOOST Y SVM ANTES Y DESPUÉS DE PCA (RECALL).

Modelo	Recall Base	Recall PCA	Std PCA	Reducción
XGBoost	0.8257 ± 0.1029	0.8203	0.0564	43.3 %
SVM	0.9017 ± 0.0135	**0.9098**	0.0163	43.3 %

Análisis de resultados: Los resultados permiten extraer las siguientes observaciones:

- **XGBoost**: el recall pasa de 0.8257 a 0.8203, una variación mínima (0.5 %), lo que indica que el modelo es robusto a la reducción y no se beneficia particularmente del PCA.
- **SVM**: el recall aumenta de 0.9017 a 0.9098, evidenciando una mejora consistente. Debido a que las SVM son sensibles a la dimensionalidad y redundancia entre características, el PCA favorece la generalización.

Por lo tanto el PCA permitió reducir la dimensionalidad en un 43.3 % manteniendo al menos el 95 % de la varianza original. Mientras que XGBoost conserva prácticamente su desempeño, el modelo SVM muestra una mejora al trabajar en un espacio reducido y ortogonal.

V-C. Extracción de características no lineal

V-C1. Uso del método UMAP: El objetivo de esta sección es reducir la dimensionalidad del conjunto de datos mediante un método no lineal que pueda capturar relaciones complejas entre variables y, posteriormente, evaluar el impacto de dicha reducción sobre los dos mejores modelos identificados en la Sección 4. Para ello se empleó *Uniform Manifold Approximation and Projection* (UMAP), una técnica basada en teoría de grafos y geometría diferencial que preserva tanto las relaciones locales como globales del espacio original.

A diferencia de PCA, que proyecta linealmente los datos, UMAP modela la estructura del manifold subyacente mediante un grafo de vecinos y optimiza una representación de baja dimensión que preserva esa estructura. Esto permite capturar separaciones no lineales en los datos, lo cual puede resultar

ventajoso en tareas de clasificación como churn, donde las fronteras de decisión son complejas.

Para permitir una exploración eficiente de múltiples configuraciones, se empleó la versión GPU de UMAP disponible en `cuML`, reduciendo drásticamente los tiempos de cómputo. Los hiperparámetros utilizados fueron los siguientes:

- **Número de componentes** $d \in \{3, 5, 10, 15\}$: Se seleccionaron valores que permiten explorar representaciones muy compactas ($d = 3$), representaciones intermedias ($d = 5$ y $d = 10$) y espacios algo más expresivos ($d = 15$). Estos rangos son comunes en la literatura para tareas de clasificación en las que se busca un equilibrio entre compresión e información útil.
- **Número de vecinos cercanos** $n\_neighbors = 50$: UMAP utiliza este parámetro para definir la escala de la estructura local del manifold. Valores elevados fomentan que el embedding preserve más estructura global, lo cual es favorable en problemas donde existen subpoblaciones amplias, como en churn, donde los patrones de abandono suelen agruparse en regiones amplias del espacio.
- **Distancia mínima** $min\_dist = 0.1$: Este hiperparámetro controla cuán "apretados" pueden estar los grupos en el embedding. Un valor moderado (0.1) evita clusters excesivamente compactos, permitiendo mantener separaciones útiles para modelos como SVM.
- **Número de épocas** $n\_epochs = 500$: Se seleccionó un número alto para asegurar una convergencia adecuada del proceso de optimización. En la práctica, valores entre 200 y 500 ofrecen buenos resultados, pero dada la disponibilidad de GPU, se optó por un valor robusto.
- **Método de construcción del grafo**: `nn_descent`. Este método es el estándar recomendado en implementaciones GPU por su eficiencia y capacidad para manejar conjuntos de datos medianos y grandes. Permite construir el grafo de vecinos aproximados de forma rápida y con bajo costo computacional.

La selección conjunta de estos hiperparámetros se realizó con el propósito de obtener embeddings estables, expresivos y adecuados para modelos clasificadores que dependen de separaciones en el espacio latente, como XGBoost y SVM. Cada embedding generado fue posteriormente evaluado mediante validación cruzada estratificada con $k = 5$, empleando *Recall* como métrica principal dada la relevancia de detectar clientes que abandonan.

Resultados: Los resultados obtenidos para las distintas configuraciones se presentan en la Tabla VIII. La tabla incluye el número de componentes usados, el *Recall* promedio y su desviación estándar para XGBoost y SVM, así como el tiempo de ejecución de UMAP GPU.

Cuadro VIII
RESULTADOS DE UMAP GPU CON DISTINTOS NÚMEROS DE COMPONENTES.

Comp.	Recall XGB	Std XGB	Recall SVM	Std SVM	Tiempo (s)
3	0.6263	0.0424	0.7366	0.0211	990.45
5	0.8182	0.0389	0.7607	0.0143	74.13
10	0.7493	0.0843	0.8108	0.0248	13.90
15	0.6190	0.0847	0.8122	0.0218	7.21

Los resultados muestran varios patrones relevantes:

- Para **XGBoost**, el mejor desempeño se obtiene con $d = 5$, alcanzando un *Recall* de 0.8182. Esto indica que un embedding moderadamente compacto preserva suficiente información discriminativa para este modelo.
- Para $d = 3$, el rendimiento cae considerablemente, lo cual era esperado debido a la pérdida significativa de información al proyectar los datos a un espacio tan reducido.
- Para **SVM**, los valores de *Recall* más altos se obtienen en $d = 10$ (0.8108) y $d = 15$ (0.8122). Esto es coherente con la sensibilidad del modelo a espacios donde las fronteras de decisión están mejor separadas.
- El tiempo de cómputo disminuye drásticamente conforme aumenta d: valores bajos requieren más trabajo de optimización debido a la mayor compresión, mientras que valores mayores facilitan al optimizador ubicar las muestras en un espacio más amplio.

En conjunto, los resultados indican que UMAP aporta una representación útil para ambos modelos, especialmente SVM, que muestra mejoras sostenidas a medida que aumenta la dimensionalidad del embedding no lineal. Esto confirma que la estructura latente capturada por UMAP contiene información relevante para la predicción del churn.

Visualización del embedding tridimensional: Para el caso $d = 3$, la Figura 11 presenta el embedding tridimensional obtenido. La visualización revela agrupamientos naturales, algunos de los cuales presentan mayor concentración de clientes con churn positivo. Esto sugiere la existencia de patrones latentes que UMAP logra capturar adecuadamente.

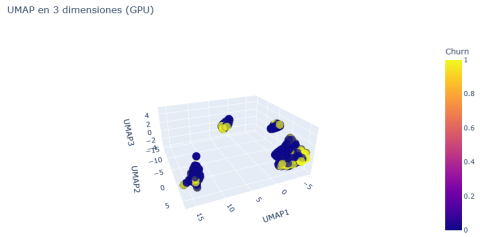

Figura 11. Embedding tridimensional generado con UMAP ($d = 3$).

Este análisis visual confirma que UMAP captura estructuras relevantes del conjunto de datos y permite identificar zonas con mayor probabilidad de churn, lo cual es coherente con los resultados obtenidos en la Tabla VIII.

VI. DISCUSIÓN Y CONCLUSIONES

VI-A. Discusión general de resultados

En este trabajo se evaluaron varias estrategias de modelado y reducción de dimensionalidad sobre el dataset *Telco Customer Churn*. A continuación se discuten los hallazgos más relevantes y su interpretación en el contexto del objetivo del proyecto (maximizar la detección de clientes con probabilidad de abandono, es decir, maximizar *Recall*):

1. **Rendimiento de los modelos:** Entre los modelos evaluados, SVM y XGBoost resultaron ser los algoritmos con mejor comportamiento según la métrica de *Recall*. En particular, la SVM alcanzó un *Recall* en validación cruzada de aproximadamente 0.9017 en el espacio original y mejoró a 0.9098 tras aplicar PCA (17 componentes, 95 % varianza). XGBoost mostró un *Recall* base de 0.8257 y se mantuvo estable alrededor de 0.82 con PCA. Estos resultados indican que:
   - SVM es especialmente sensible a la redundancia y la escala de las variables; PCA le aporta una representación más ortogonal que mejora su capacidad de detección.
   - XGBoost, por su naturaleza de ensamble y manejo intrínseco de interacciones, es robusto a la reducción lineal de dimensión y no se benefició claramente de PCA.
2. **Análisis de variables (MI & Pearson):** El análisis individual de variables identificó predictores con alta capacidad discriminativa (por ejemplo `Contract`, `tenure`, `TechSupport`, `OnlineSecurity`) y características con información limitada (`gender`, `MultipleLines`). Aunque se aplicó un umbral heurístico para MI ($MI < 0.005$) y para Pearson ($|r| < 0.10$ en la versión revisada), la decisión final incorpora criterios estadísticos y conocimiento del dominio: se recomienda eliminar `gender` y `MultipleLines` por su baja contribución empírica, pero `SeniorCitizen` se mantiene dada su relevancia demográfica documentada en la literatura.
3. **Reducción no lineal (UMAP GPU):** UMAP produjo embeddings útiles pero con comportamiento variable según la dimensión:
   - Para SVM, embeddings con $d = 10$ y $d = 15$ alcanzaron *Recall* $\approx$ 0.81, consistentes con una buena separación no lineal; sin embargo, estos valores fueron inferiores al *Recall* logrado por SVM con PCA (≈0.91) en nuestro experimento.
   - Para XGBoost, el mejor resultado con UMAP fue en $d = 5$ (*Recall* $\approx$ 0.818), similar al rendimiento del modelo en el espacio original.
   - Coste computacional: UMAP (particularmente con d pequeño y n_epochs alto) puede ser costoso; la implementación GPU (cuML) y parámetros como `build_algo=nn_descent`, disminuir la muestra o reducir `n_epochs` son opciones que permiten experimentación práctica sin sacrificar demasiada calidad.

VI-B. Comparación con la Sección 3 (estado del arte)

La Sección 3 describe trabajos previos que usan el mismo dataset y reportan métricas como AUC y accuracy. Por ejemplo, dichos estudios muestran AUCs en el rango ≈ 0.81–0.85 y valores de accuracy alrededor de 79–85 % (según [2], [3]). Al comparar nuestros resultados con esos trabajos es importante considerar lo siguiente:

- **Métrica diferente:** Los trabajos citados frecuentemente reportan AUC o accuracy, mientras que en este proyecto se priorizó *Recall* por su relevancia de negocio. Por tanto, la comparación directa no es homogénea. Sin embargo, la capacidad de alcanzar un *Recall* alto (SVM con PCA $\approx$ 0.91) es coherente con enfoques que priorizan detección de positivos, y no contradice que AUCs reportados en la literatura se ubiquen en rangos similares cuando se optimizan métricas globales.
- **Posible superioridad práctica:** Algunos estudios reportan que ensamblados (XGBoost, Random Forest) y redes pueden alcanzar accuracy/AUC competitivos; en nuestro caso, XGBoost ofrece buen equilibrio y SVM logra mayor *Recall* tras PCA. Esto sugiere que, dependiendo de la métrica objetivo (AUC vs Recall), distintos trabajos pueden priorizar modelos distintos, lo cual coincide con la diversidad de conclusiones en [2], [3].
- **Reproducibilidad y diferencias experimentales:** Variaciones en preprocesamiento (tratamiento de `TotalCharges`, codificación de categorías, manejo de outliers), en selección de muestras y en la métrica de evaluación explican parte de las diferencias entre resultados. Nuestro flujo se documentó íntegramente (thresholds, criterios de eliminación, seed, folds), lo que facilita comparaciones reproducibles.

VI-C. Limitaciones del estudio

- **Comparabilidad de métricas:** Como se mencionó, la diferencia en métricas reportadas respecto a otros estudios (AUC vs Recall) dificulta comparaciones directas; recomendamos publicar ambas métricas para facilitar futuras comparaciones.
- **Sensibilidad a la muestra y aleatoriedad:** Algunas técnicas no lineales (UMAP) y búsqueda de hiperparámetros pueden ser sensibles a `random_state`; se aconseja fijar seeds y documentar experimentos para garantizar reproducibilidad.
- **Costo computacional:** UMAP con parámetros agresivos (d pequeño, `n_epochs` alto) puede ser costoso; la solución GPU reduce tiempos pero introduce dependencias en el entorno de ejecución.

VI-D. Conclusión final

El estudio demuestra que, con un pipeline bien documentado y una selección cuidadosa de métricas alineadas al objetivo de negocio, es posible alcanzar un *Recall* alto (≈0.91) en la predicción de churn sobre el dataset Telco Customer Churn. PCA demostró ser una técnica eficaz y barata computacionalmente para mejorar la generalización de SVM; UMAP aporta embeddings no lineales útiles pero con mayor coste y comportamiento dependiente de la dimensión elegida. En balance, recomendamos como solución práctica priorizar **SVM + PCA** (u un ensemble con XGBoost) para implementación en pruebas piloto, complementando la solución con calibración, monitorización y reentrenamiento periódico.

REFERENCIAS

[1] I. S. Data, "Telco customer churn dataset," https://www.kaggle.com/datasets/blastchar/telco-customer-churn, 2017, wA_Fn-UseC_-Telco-Customer-Churn.csv. Consultado en octubre de 2025.

[2] Highlights in Science, Engineering and Technology (SDPIT) authors, “Telco customer churn prediction,” https://www.researchgate.net/publication/381544028_Telco_Customer_Churn_Prediction, 2024, estudio experimental usando el dataset WA_Fn-UseC_-Telco-Customer-Churn (Kaggle). Consultado en octubre de 2025.
[3] D. Pawar, Y. Sabla, N. Tayal, and S. Nainan, “Predicting telecom customer churn: An in-depth evaluation of machine learning algorithms,” *Global International Journal of Engineering and Technology (GIJET)*, 2024, estudio experimental usando WA_Fn-UseC_-Telco-Customer-Churn (Kaggle). Consultado en octubre de 2025.
[4] A. Barsotti *et al.*, “A decade of churn prediction techniques in the telco industry: survey and perspectives,” *SN Computer Science*, 2024.
[5] W. Verbeke, D. Martens, C. Mues, and B. Baesens, “Social network analysis for customer churn prediction,” *Applied Soft Computing*, vol. 14, pp. 431–446, 2014.