



UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

DEPARTAMENTO
DE MATEMÁTICA

Predicción de churn en clientes Telco: evaluación de cuatro enfoques de modelado

Aplicaciones de la matemática en la ingeniería (MAT281)

Integrantes:
Pablo Farias, Pablo Leiva, Benjamín Tondreau,
Joaquin Inzunza, William Nuñez

2^{do} semestre 2025

Contenidos

- 1 Acerca del problema**
- 2 Objetivos**
- 3 Descripción del dataset**
- 4 Metodología**
 - Modelo 1:
 - Modelo 2:
 - Modelo 3:
 - Modelo 4:
- 5 Resultados finales**

¿Qué problema queremos resolver?

- La empresa pierde ingresos cuando un cliente se va.
- Retener clientes es menos costoso que conseguir nuevos clientes.
- Queremos predecir *churn*¹ antes de que ocurra.

¹En este contexto, “churn” se refiere a la pérdida de un cliente, es decir, cuando da de baja el servicio o se cambia a otra compañía.

Objetivo del proyecto

- Construir 4 modelos de clasificación.
- Comparar su desempeño en la predicción de *churn*.
- Extraer conclusiones útiles para la empresa.

Dataset: Telco Customer Churn

- Cada fila del dataset es un cliente.
- El dataset cuenta con 7043 clientes.
- variable objetivo ***Churn*** (binaria: Sí o No).
- Variables:
 - Demográficas (género, pareja, dependientes).
 - Servicios contratados (internet, teléfono, TV, etc).
 - Facturación (meses en la compañía, cobro mensual, total pagado).

Dataset: Telco Customer Churn

Pequeña muestra del conjunto de datos con algunas columnas representativas

customerID	tenure	MonthlyCharges	Churn
7590-VHVEG	1	29.85	No
3668-QPYBK	2	53.85	Yes
9237-HQITU	2	70.70	Yes
1452-KIOVK	22	89.10	No

Desafíos del dataset

- Clases **desbalanceadas** (menos clientes que se van que los que se quedan).
- Algunas variables vienen como texto y hay que limpiarlas.
- Muchas variables categóricas (Sí/ No, tipo de contrato, tipo de internet, etc).

Flujo del trabajo

- Limpieza y preparación de datos.
- División en entrenamiento y prueba.
- Entrenamiento de 4 modelos.
- Evaluación en conjunto de prueba.
- Análisis de sobreajuste y métricas.

Preprocesamiento

- Eliminamos *CustomerID*.
- *TotalCharges* es variable numérica, se trataron espacios y vacíos.
- Escalamos numéricas (*tenure*, *MontlyCharges*, *TotalCharges*. con **StandardScaler** en regresión logística.
- Categóricas:
 - Algunas con **LabelEncoder** (binarias: Yes/No).
 - Otras con **one-hot encoding** (*get_dummies*) para Random Forest y XGBoost.

Modelos

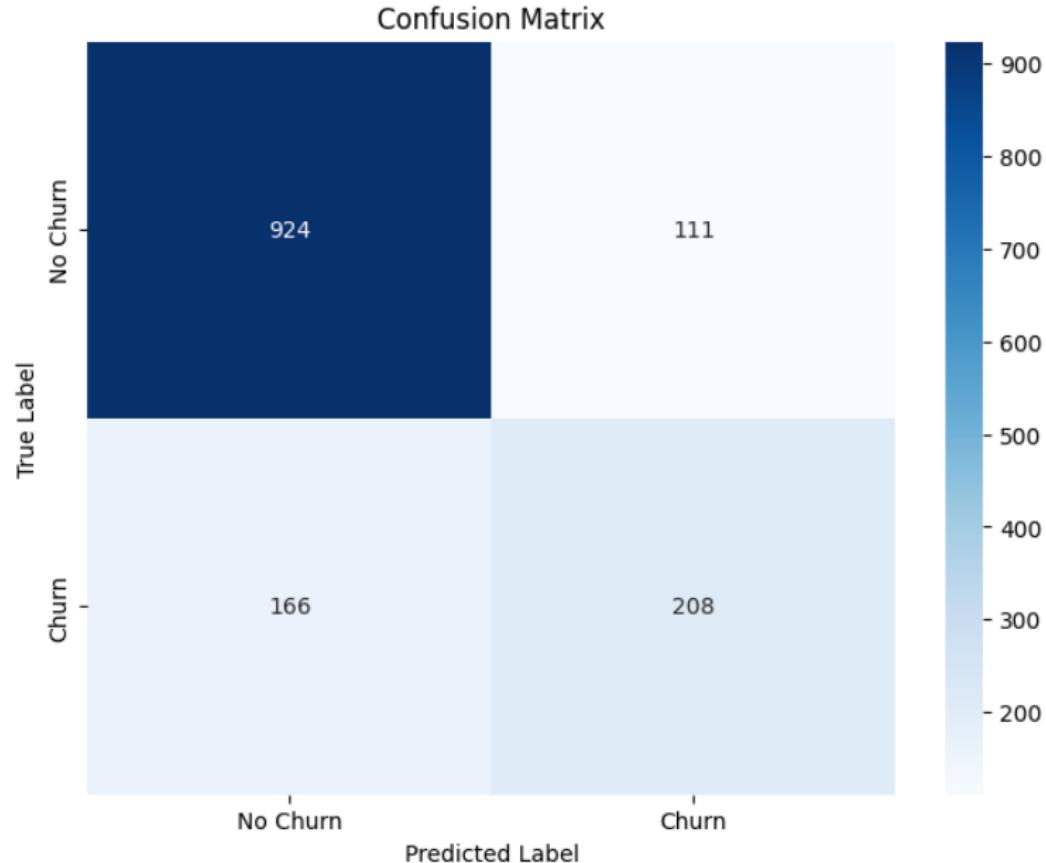
- 1 Modelo 1: **Regresión Logística**: Modelo lineal, interpretable.
- 2 Modelo 2: **Random Forest**: Muchos árboles de decisión, *class_weight='balanced'*.
- 3 Modelo 3: **XGBoost**: Ensamble de árboles (gradient boosting). Usa *scale_pos_weight* para el desbalance del **churn**.
- 4 Modelo 4: **Red neuronal**: perceptrón multicapa que combina las variables de forma no lineal y aprende pesos para aproximar la probabilidad de **churn**.

Regresión Logística

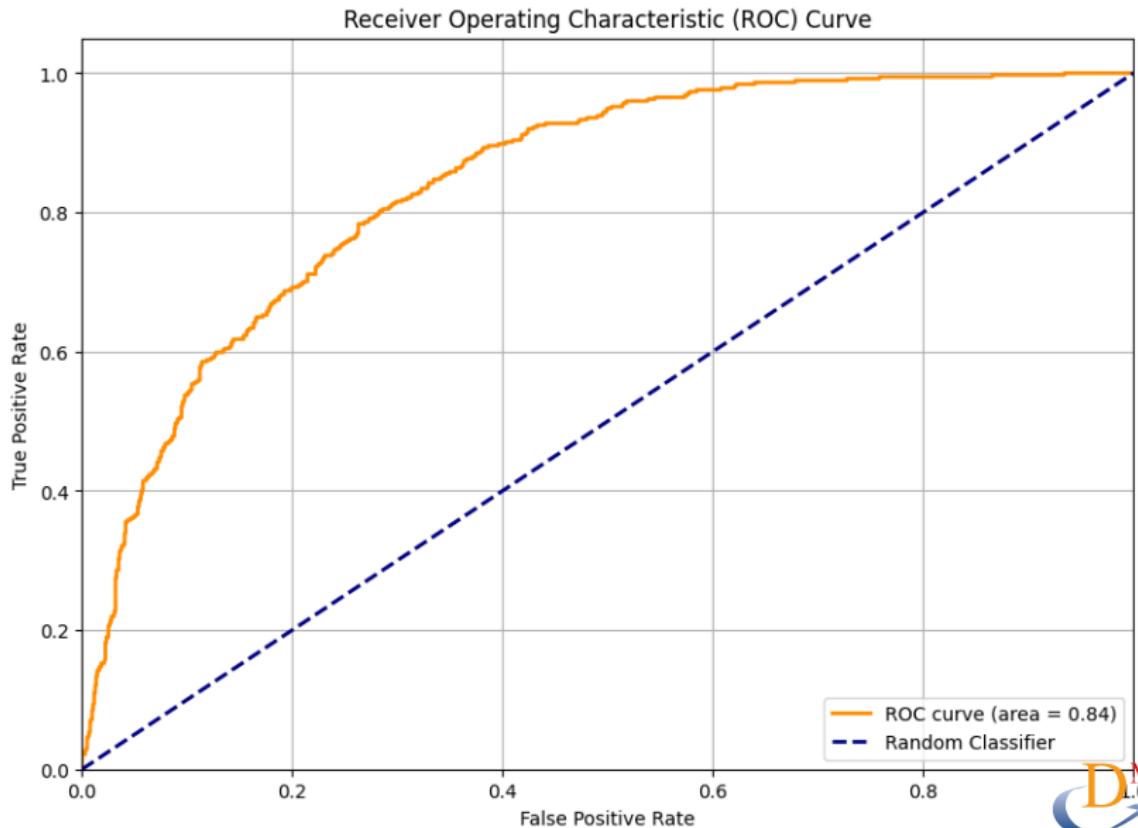
¿Qué hace?

- Asume relación lineal entre las variables log-odds de churn.
- Devuelve una probabilidad de churn por cliente.
- Entrenado con `solver='liblinear'` y datos escalados.

Regresión Logística



Regresión Logística



Regresión Logística

Resultados:

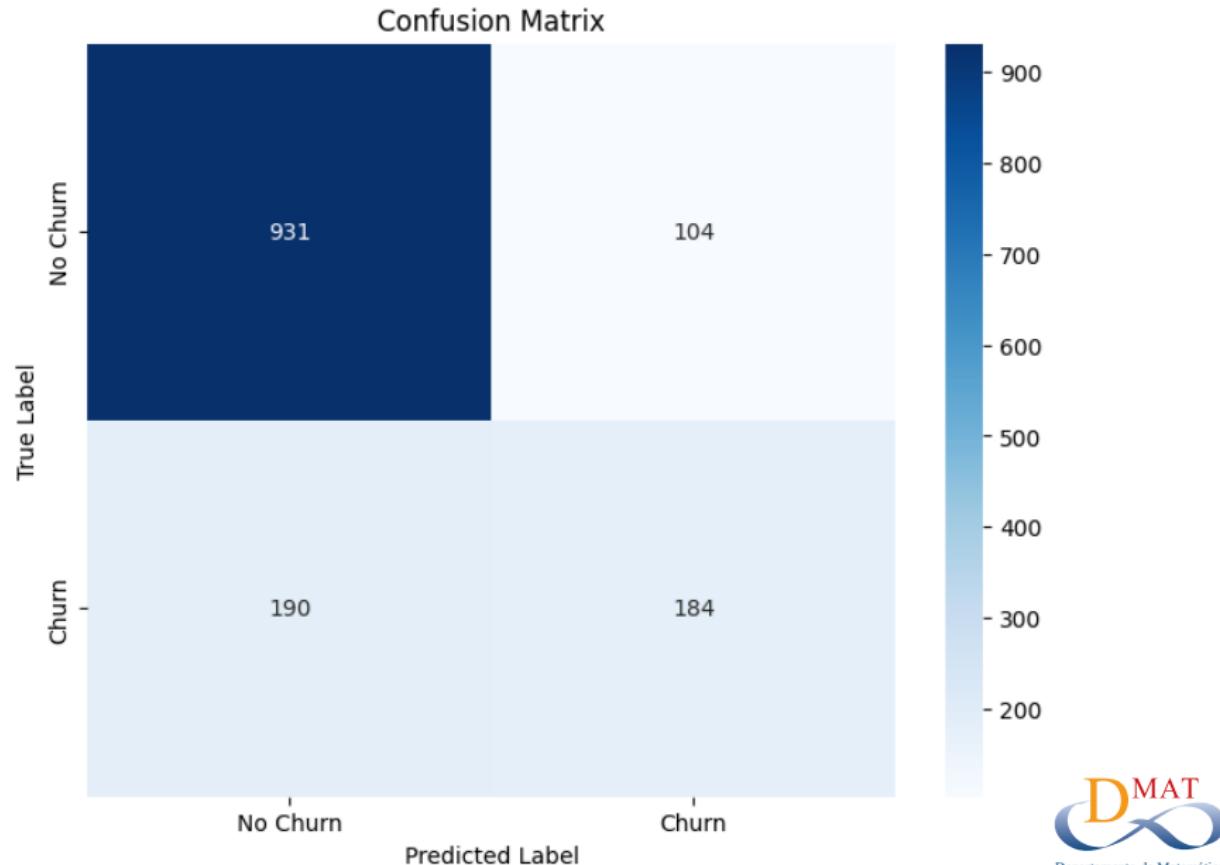
- Accuracy: 0.8034
- Precisión (churn): 0.6520
- Recall (churn): 0.5561
- F1-score (churn): 0.6003
- ROC-AUC: 0.8422

Random Forest

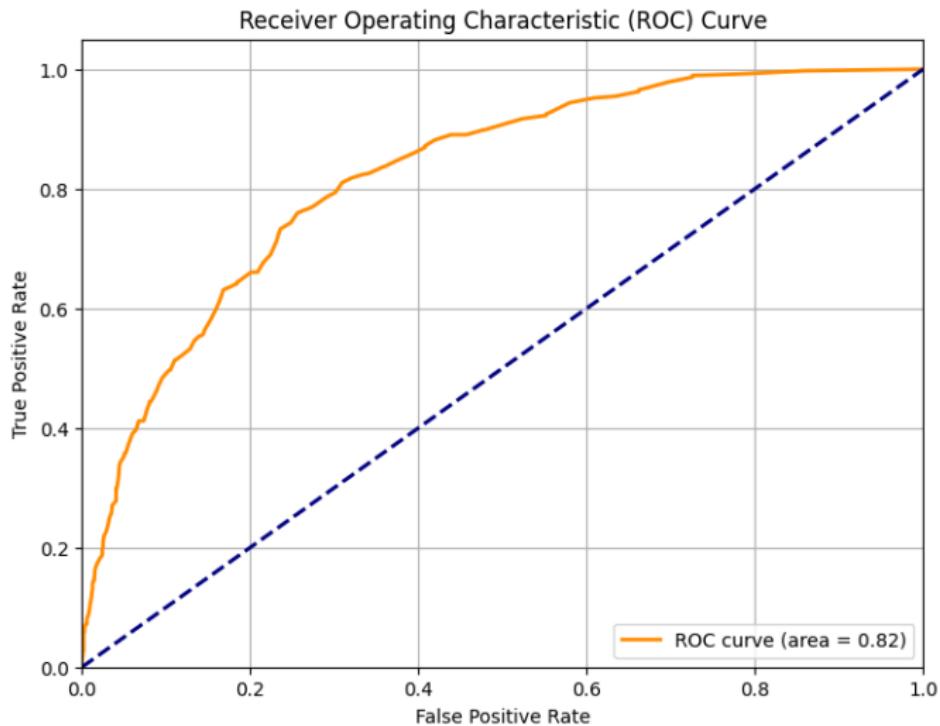
¿Qué hace?

- Entrena **100 árboles** de decisión.
- Cada árbol ve subconjuntos de variables y datos.
- Se usa *class_weight='balanced'* para compensar que hay menos churn que no-churn.

Random Forest



Random Forest



Random Forest

Resultados:

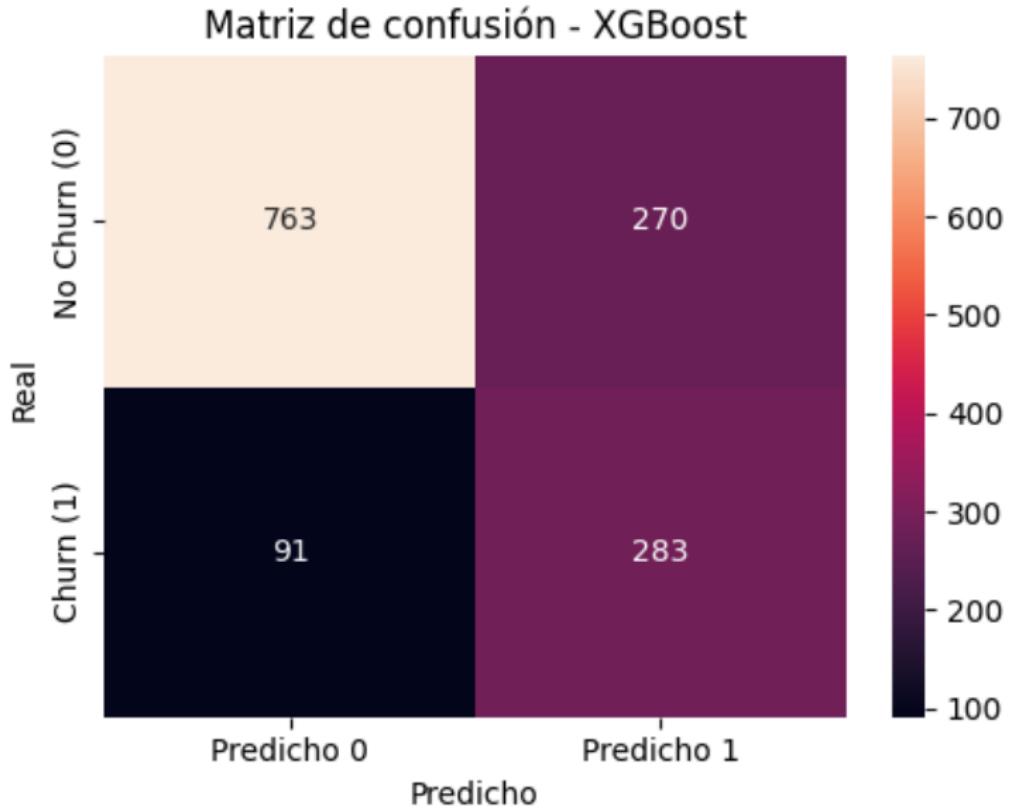
- Accuracy: 0.7913
- Clase 1 (churn):
 - Presición: 0.64
 - Recall: 0.49
 - F1: 0.56
- ROC-AUC: 0.8224

XGBoost

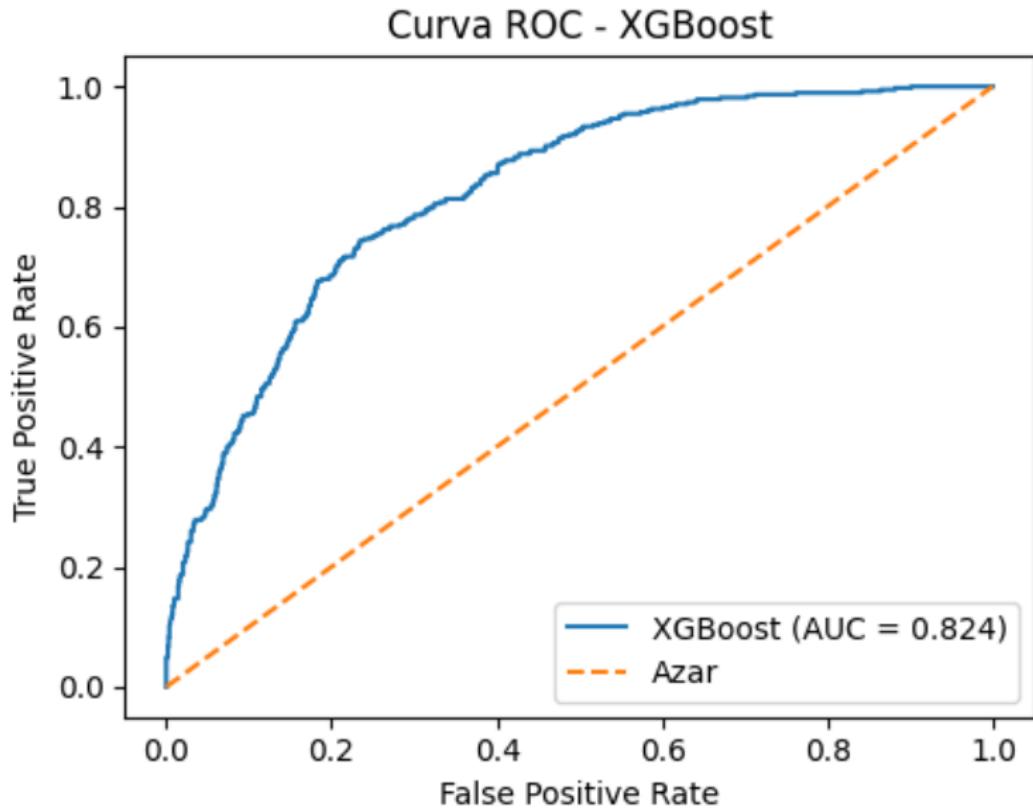
¿Qué hace?

- Es un modelo de aprendizaje automático basado en árboles de decisión.
- Construye muchos árboles pequeños de forma secuencial.
- Cada nuevo árbol se entrena para corregir los errores de los árboles anteriores.
- La predicción final es la suma de las contribuciones de todos los árboles.
- Utiliza una tasa de aprendizaje y regularización para evitar el sobreajuste.
- Es especialmente eficaz en datos tabulares y captura relaciones no lineales e interacciones entre variables.

XGBoost



XGBoost



XGBoost

Resultados modelo

- **Accuracy (exactitud):** $\approx 0,74$
- **Precision (clase churn):** $\approx 0,51$
- **Recall / Sensibilidad (clase churn):** $\approx 0,76$
- **F1-score (clase churn):** $\approx 0,61$
- **ROC-AUC:** $\approx 0,82$

Red neuronal

¿Qué hace?

- Toma todas las variables del cliente como entrada.
- Las pasa por varias capas de neuronas que combinan la información de forma no lineal.
- Aprende automáticamente los pesos para aproximar la probabilidad de **churn**.
- Entrega como salida una probabilidad de que el cliente se vaya de la compañía.

Red neuronal

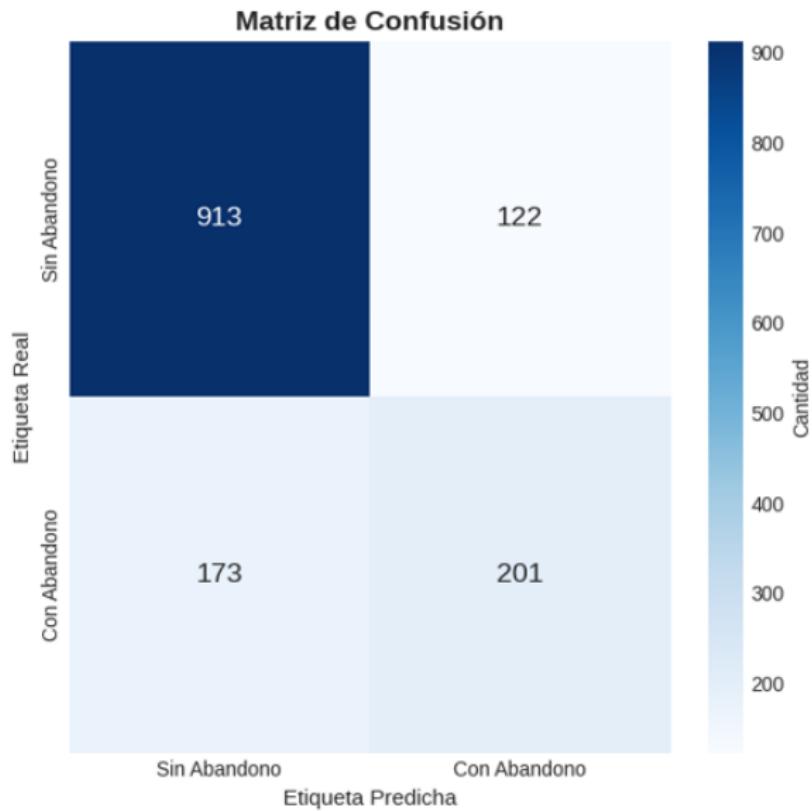
Características:

- La entropía cruzada es la función de costo más utilizada para entrenar redes neuronales en problemas de clasificación con clases mutuamente excluyentes.

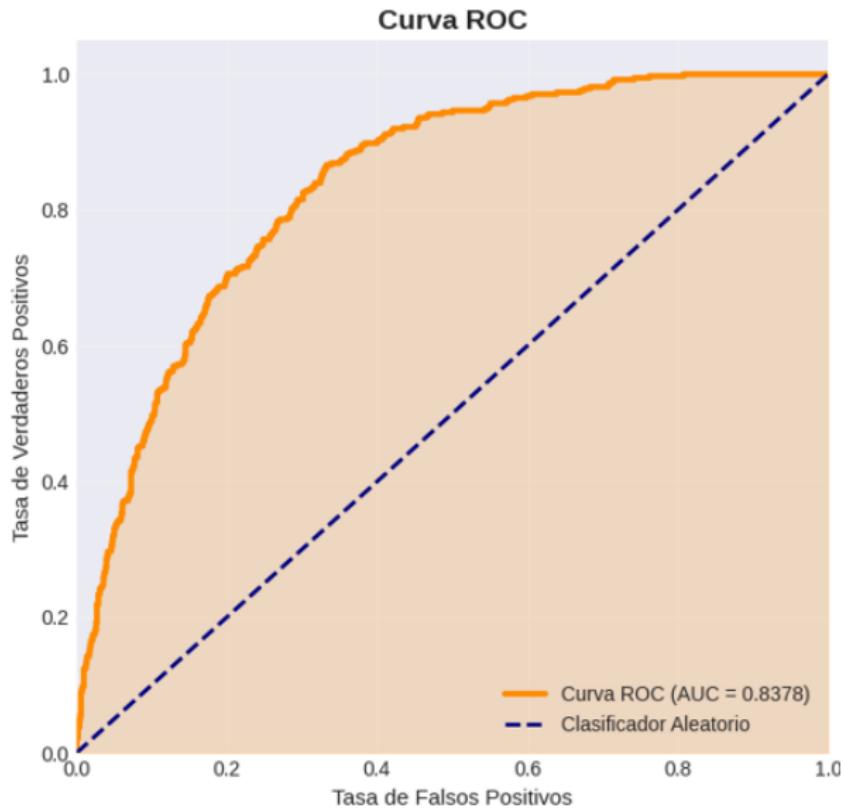
$$L(f(x), y) = \text{CE}(f(x), y) = - \sum_k y_k \log(f_k(x))$$

- 4 capas ocultas con arquitectura: 1218 → 64 → 32 → 16 neuronas.
- Batch Normalization y Dropout para regularización
- Early stopping y reducción de learning rate
- Métricas completas de clasificación

Red neuronal



Red neuronal



Red neuronal

Resultados modelo

- **Accuracy (exactitud):** 0.7906 (79.06%)
- **Precision (clase churn):** 0.6223 (62.23%)
- **Recall / Sensibilidad (clase churn):** 0.5374 (53.74%)
- **F1-score (clase churn):** 0.5768 (57.68%)
- **ROC-AUC:** 0.8378 (83.78%)

Conclusiones sobre los modelos

- **Regresión logística:** mejor equilibrio global. Mayor accuracy (0.8034) y mejor ROC-AUC (0.8422), con F1 de churn ≈ 0.60 . Modelo simple e interpretable.

Conclusiones sobre los modelos

- **Regresión logística:** mejor equilibrio global. Mayor accuracy (0.8034) y mejor ROC-AUC (0.8422), con F1 de churn ≈ 0.60 . Modelo simple e interpretable.
- **Random Forest:** accuracy (0.7913) y ROC-AUC (0.8224) algo menores, con recall de churn más bajo (0.49). No mejora a la regresión logística, por lo que queda dominado.

Conclusiones sobre los modelos

- **Regresión logística:** mejor equilibrio global. Mayor accuracy (0.8034) y mejor ROC-AUC (0.8422), con F1 de churn $\approx 0,60$. Modelo simple e interpretable.
- **Random Forest:** accuracy (0.7913) y ROC-AUC (0.8224) algo menores, con recall de churn más bajo (0.49). No mejora a la regresión logística, por lo que queda dominado.
- **XGBoost:** menor accuracy ($\approx 0,74$) y precisión para churn ($\approx 0,51$), pero **mayor recall** ($\approx 0,76$) y ROC-AUC $\approx 0,82$. Detecta más clientes que se van, a costa de más falsos positivos.

Conclusiones sobre los modelos

- **Regresión logística:** mejor equilibrio global. Mayor accuracy (0.8034) y mejor ROC-AUC (0.8422), con F1 de churn $\approx 0,60$. Modelo simple e interpretable.
- **Random Forest:** accuracy (0.7913) y ROC-AUC (0.8224) algo menores, con recall de churn más bajo (0.49). No mejora a la regresión logística, por lo que queda dominado.
- **XGBoost:** menor accuracy ($\approx 0,74$) y precisión para churn ($\approx 0,51$), pero **mayor recall** ($\approx 0,76$) y ROC-AUC $\approx 0,82$. Detecta más clientes que se van, a costa de más falsos positivos.
- **Red neuronal:** resultados muy cercanos a la regresión logística (accuracy 0.7906, ROC-AUC 0.8378), sin una mejora clara pero con mayor complejidad.

Conclusión

En resumen, si buscamos un modelo equilibrado y fácil de interpretar, elegimos la **regresión logística**; si priorizamos detectar la mayor cantidad posible de clientes en riesgo, el **XGBoost** es el modelo más atractivo.