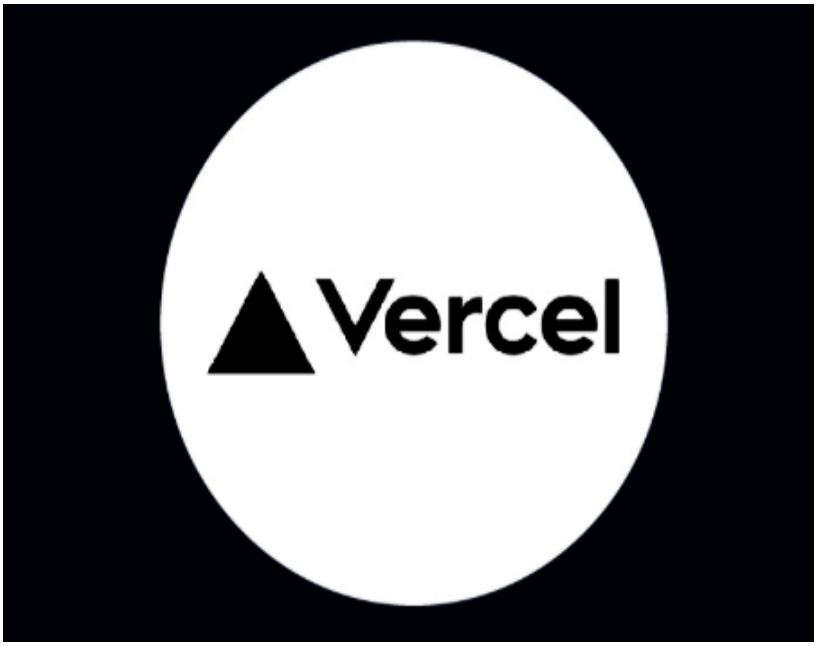
Predicción de Abandono de Clientes (Churn)

Trabajo Final: Modelo Predictivo



Unit 25 . Applied Machine Learning

Hecho por: Rodrigo Barrero

PORTADA

Nombre Alumno / DNI	Rodrigo Barrero Martín/51518191P
Título del Programa	3ºPD CYBER SECURITY / COMPUTER SCIENCE / DATA SCIENCE
Nº Unidad y Título	UNIT 25. APPLIED MACHINE LEARNING
Año académico	2025/2026
Profesor de la unidad	Rabindranath Andujar
Título del Assignment	AB Final
Día de emisión	30/09/25
Día de entrega	20/1/2026
Nombre IV y fecha	
Declaración del estudiante	**Certifico que la presentación del assignment es completamente mi propio trabajo y entiendo completamente las consecuencias del plagio. Entiendo que hacer una declaración falsa es una forma de mala práctica.** **Fecha: 20/1/2026** **Firma del alumno:** 

Plagio

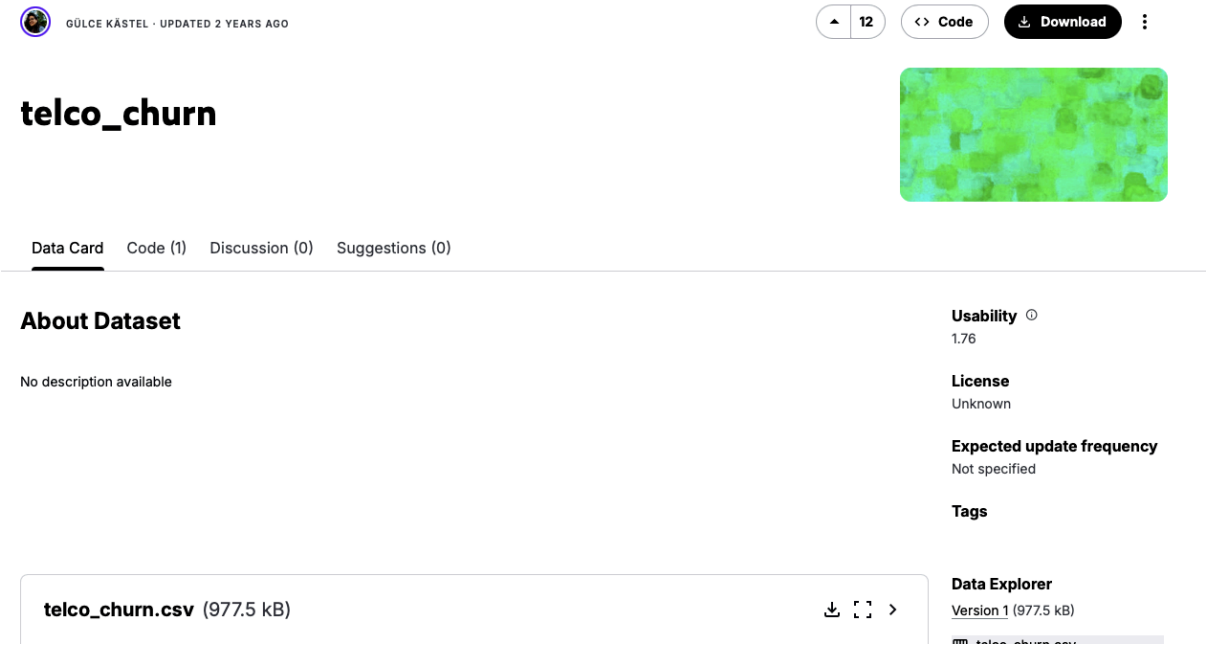
El plagio es una forma particular de hacer trampa. El plagio debe evitarse a toda costa y los alumnos que infrinjan las reglas, aunque sea inocentemente, pueden ser sancionados. Es su responsabilidad asegurarse de comprender las prácticas de referencia correctas. Como alumno de nivel universitario, se espera que utilice las referencias adecuadas en todo momento y mantenga notas cuidadosamente detalladas de todas sus fuentes de materiales para el material que ha utilizado en su trabajo, incluido cualquier material descargado de Internet. Consulte al profesor de la unidad correspondiente o al tutor del curso si necesita más consejos.

Índice:

1. Definición del problema

1.1 Contexto del Problema

La retención de clientes es crucial en telecomunicaciones porque conseguir nuevos clientes cuesta entre 5 y 7 veces más que mantener a los existentes. Este proyecto utiliza un CSV (Telco_Churn.csv) de 7,043 clientes de una empresa de telecomunicaciones con información demográfica, contractual y de consumo para predecir qué clientes abandonarán el servicio.



1.2 Hipótesis Inicial

Los clientes con mayor probabilidad de abandonar tienen contratos mensuales (no anuales), poca antigüedad en la empresa, y cargos mensuales elevados.

1.3 Valor para el Usuario Final

El sistema está diseñado para los departamentos de marketing y retención. Les permite identificar clientes en riesgo, priorizar los recursos limitados, y diseñar campañas de fidelización específicas. Reducir el churn aunque sea un poco, tiene un gran impacto en la rentabilidad.

2. Fundamentación Teórica

2.1 Algoritmo Elegido: Regresión Logística

He elegido la regresión logística por varias razones. Es ideal para clasificación binaria (abandona o no abandona), crea probabilidades interpretables (no solo sí o no), es fácil de explicar a stakeholders no técnicos, y funciona bien como modelo base antes de probar algoritmos más complejos.

2.2 Conexión con el Ejercicio de Excel

La Regresión Logística funciona igual que la red neuronal que hicimos en Excel. Aprende pesos para cada variable de entrada, tiene un bias que ajusta el punto de decisión, minimiza el error comparando predicciones con valores reales, y también usa la función sigmoide para convertir números en probabilidades entre 0 y 1.

La diferencia es que en Excel teníamos capas ocultas, mientras que la Regresión Logística es más simple: sólo tiene una capa.

Red Neuronal_Rodrigo Barrero.xlsx — Abrir con

	Sample Data				Weights					Weight Adjusment		
Input 1	Input 2	Input 3	Output	Weight 1	Weight 2	Weight 3	Bias	Output	W1 Change	W2 Change	W3 Change
1	0	0	1	71%	76%	83%	0,71	0,67	7,3%	0,0%	0,0%
1	0	1	1	78%	76%	83%	1,61	0,83	2,3%	0,0%	2,3%
0	1	1	0	81%	76%	85%	1,61	0,83	0,0%	-11,6%	-11,6%
1	1	1	1	81%	64%	74%	2,19	0,90	0,9%	0,9%	0,9%
0	0	1	0	82%	65%	75%	0,75	0,68	0,0%	0,0%	-14,8%
0	1	0	0	82%	65%	60%	0,65	0,66	0,0%	-14,8%	0,0%
1	1	0	1	82%	51%	60%	1,32	0,79	3,5%	3,5%	0,0%
0	0	0	0	85%	54%	60%	0,00	0,00	0,0%	0,0%	0,0%
1	0	0	1	85%	54%	60%	0,85	0,70	6,3%	0,0%	0,0%
1	0	1	1	91%	54%	60%	1,51	0,82	2,7%	0,0%	2,7%
0	1	1	0	94%	54%	63%	1,17	0,76	0,0%	-13,8%	-13,8%
1	1	1	1	94%	40%	49%	1,83	0,86	1,6%	1,6%	1,6%
0	0	1	0	96%	42%	50%	0,50	0,62	0,0%	0,0%	-14,6%
0	1	0	0	96%	42%	36%	0,42	0,60	0,0%	-14,4%	0,0%
1	1	0	1	96%	27%	36%	1,23	0,77	4,0%	4,0%	0,0%
0	0	0	0	100%	31%	36%	0,00	0,00	0,0%	0,0%	0,0%
1	0	1	1	100%	31%	36%	1,35	0,79	3,4%	0,0%	3,4%
0	1	1	0	103%	31%	39%	0,71	0,67	0,0%	-14,8%	-14,8%
1	1	1	1	103%	17%	24%	1,44	0,81	3,0%	3,0%	3,0%
0	0	1	0	106%	20%	27%	0,27	0,57	0,0%	0,0%	-13,9%
0	1	0	0	106%	20%	13%	0,20	0,55	0,0%	-13,6%	0,0%
1	1	0	1	106%	6%	13%	1,12	0,75	4,6%	4,6%	0,0%
0	0	0	0	110%	11%	13%	0,00	0,00	0,0%	0,0%	0,0%
1	0	0	1	110%	11%	13%	1,10	0,75	4,7%	0,0%	0,0%

3. Ingeniería Y Análisis de Datos

Análisis Exploratorio (EDA)

3.1 Distribución del Churn

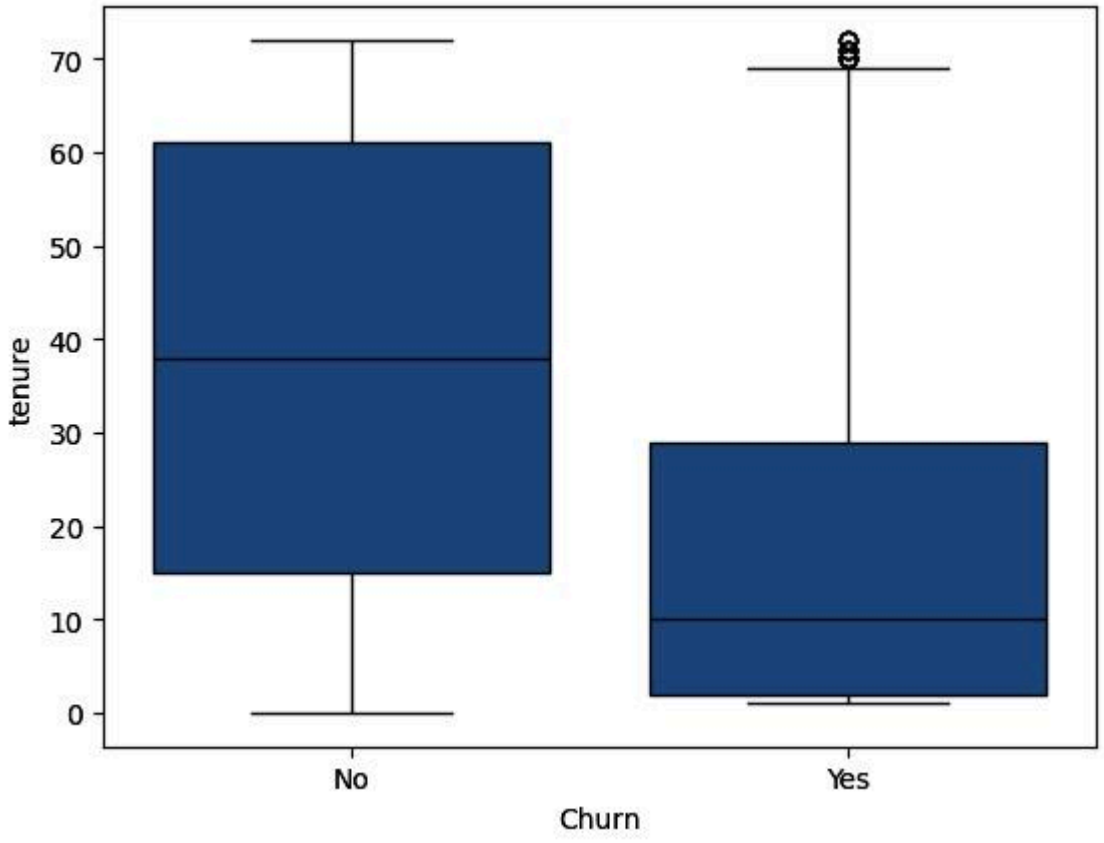
Se observa un desbalance moderado de clases, el 73.46% de los clientes permanecen (No) mientras que 26.54% abandonan (Yes).

Churn	proportion
No	0.73463
Yes	0.26537

dtype: float64

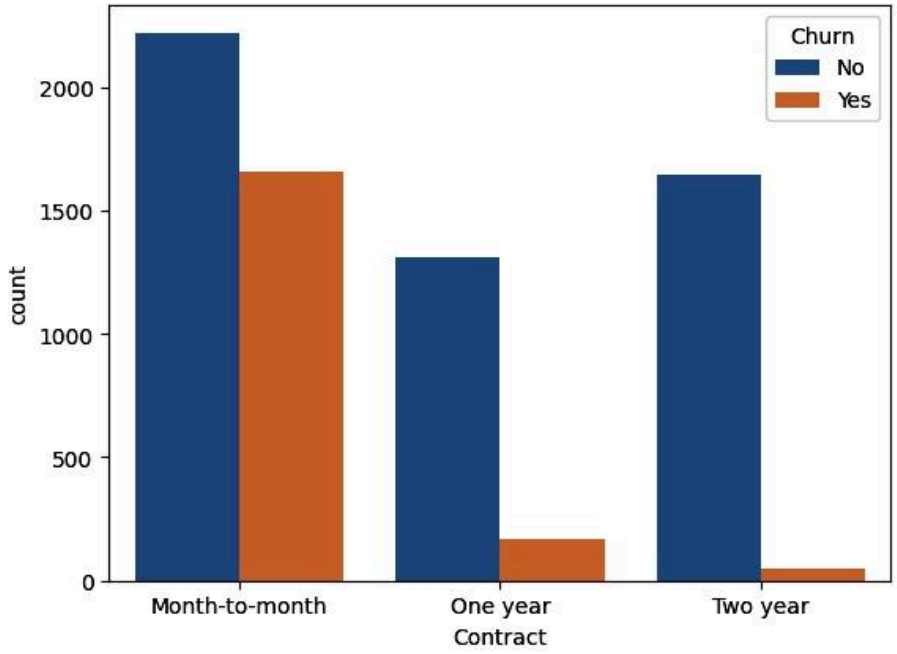
3.2 Antigüedad vs Churn

El boxplot muestra claramente que los clientes que abandonan tienen mucha menos antigüedad (mediana ~10 meses) comparado con los que se quedan (mediana ~38 meses). Los primeros meses son críticos para la retención.



3.3 Tipo de Contrato vs Churn

Los contratos mensuales (Month-to-month) presentan una tasa de abandono dramáticamente superior comparado con contratos de uno o dos años, confirmando la hipótesis inicial.



3.4 Preprocesamiento

Limpieza

Convertí TotalCharges a numérico ya que tenía espacios en blanco, también eliminé 11 registros con valores faltantes (0.2% del total), y quité la columna customerID.

Transformaciones

Para la variable Churn convertí "Yes"→1, "No"→0. Para las variables categóricas apliqué one-hot encoding (Contract, PaymentMethod, etc.) con drop_first=True para evitar la multicolinealidad.

División Y Escalado

Utilicé 80% entrenamiento, 20% test (con random_state=42 para reproducibilidad). Apliqué StandardScaler para transformar variables numéricas a media=0, desviación=1. Importante: el scaler se ajustó solo en train para evitar data leakage.

3.5 Entrenamiento Del Modelo

Paso 1: Crear el modelo

python

```
model = LogisticRegression(max_iter=1000)
```

¿Qué hace este código?

Crea un espacio vacío de tipo Regresión Logística. El parámetro max_iter=1000 le da 1000 intentos para aprender los patrones de los datos.

Paso 2: Entrenar el modelo

python

```
model.fit(X_train, y_train)
```

¿Qué hace este código?

Le muestra al modelo los datos de entrenamiento. X_train contiene las características de los clientes (antigüedad, tipo de contrato, cargos, etc.) y y_train contiene las respuestas correctas (si abandonaron o no). El modelo aprende los patrones que llevan al churn.

Evaluación del Modelo

Paso 3: Hacer Predicciones

python

```
pythony_pred = model.predict(X_test)
```

¿Qué hace este código?

El modelo toma los datos de clientes que nunca ha visto antes (X_test) y predice si van a abandonar o no. Es el "examen final" del modelo.

Paso 4: Evaluar el rendimiento

```
python
print("Accuracy:", accuracy_score(y_test,
y_pred))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

¿Qué hace este código?

Compara las predicciones del modelo (`y_pred`) con lo que realmente pasó (`y_test`) y calcula las métricas de rendimiento para saber qué tan bien funciona el modelo.

Salida:

```
Accuracy: 0.787 (78.7%)
[[915 118]
 [181 193]]
```

Interpretación:

- **Accuracy 78.7%:** El modelo acierta aproximadamente 79 de cada 100 predicciones
- **Matriz de confusión:** Muestra los aciertos y errores. Los 181 casos en la segunda fila (falsos negativos) son clientes que se fueron pero el modelo predijo que se quedarían
- **Recall 0.52:** Solo detectamos al 52% de los clientes que realmente abandonan, lo cual es muy insuficiente.

3.6 Análisis Crítico de Errores

El modelo presenta tres problemas principales que limitan su rendimiento. Primero, el recall es sólo del 52%, lo que significa que de 374 clientes que realmente abandonan, solo detectamos 193. Se nos escapan 181 clientes, cuando en la industria se busca un recall superior al 70%. Segundo, el modelo se confunde con clientes que tienen características mixtas: por ejemplo, un contrato mensual (señal negativa) combinado con alta antigüedad (señal positiva) lo lia, igual que un contrato anual (señal positiva) con cargos muy altos (señal negativa). Tercero, faltan variables críticas que el modelo no conoce, como la satisfacción del cliente, la calidad del servicio que recibe, las ofertas de la competencia o su historial de llamadas al soporte técnico. Sin estos datos, el modelo nunca podrá alcanzar un buen rendimiento.

3.7 Verificación de Hipótesis

La hipótesis inicial quedó confirmada: los contratos mensuales y la poca antigüedad sí predicen el churn de forma contundente. Sin embargo, la relación con los altos cargos ha sido más complicada de lo que me esperaba. A veces indican compromiso con la empresa (el cliente tiene muchos servicios contratados), y no siempre insatisfacción por los altos precios. Como descubrimiento nuevo e inesperado, los métodos de pago automáticos reducen significativamente el churn, algo que no había pensado en la hipótesis original.

3.8 Honestidad de los Resultados

Un recall del 52% no es suficiente para llevar esto a producción. En la industria se busca superar el 70%. Sin embargo, prefiero ser honesto con estos resultados porque demuestran algo importante: predecir el comportamiento humano es prácticamente imposible, y la calidad y cantidad de datos disponibles limita el rendimiento máximo que cualquier modelo puede alcanzar.

4. Arquitectura Y Despliegue

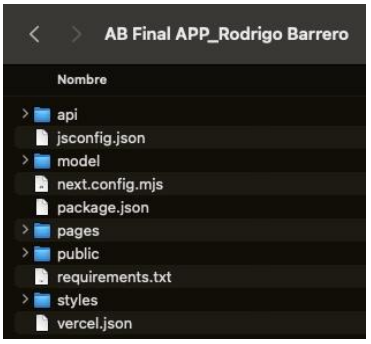
4.1 Diagrama del Sistema

Usuario → Web (Vercel) → API Backend → Modelo ML → Predicción → Usuario

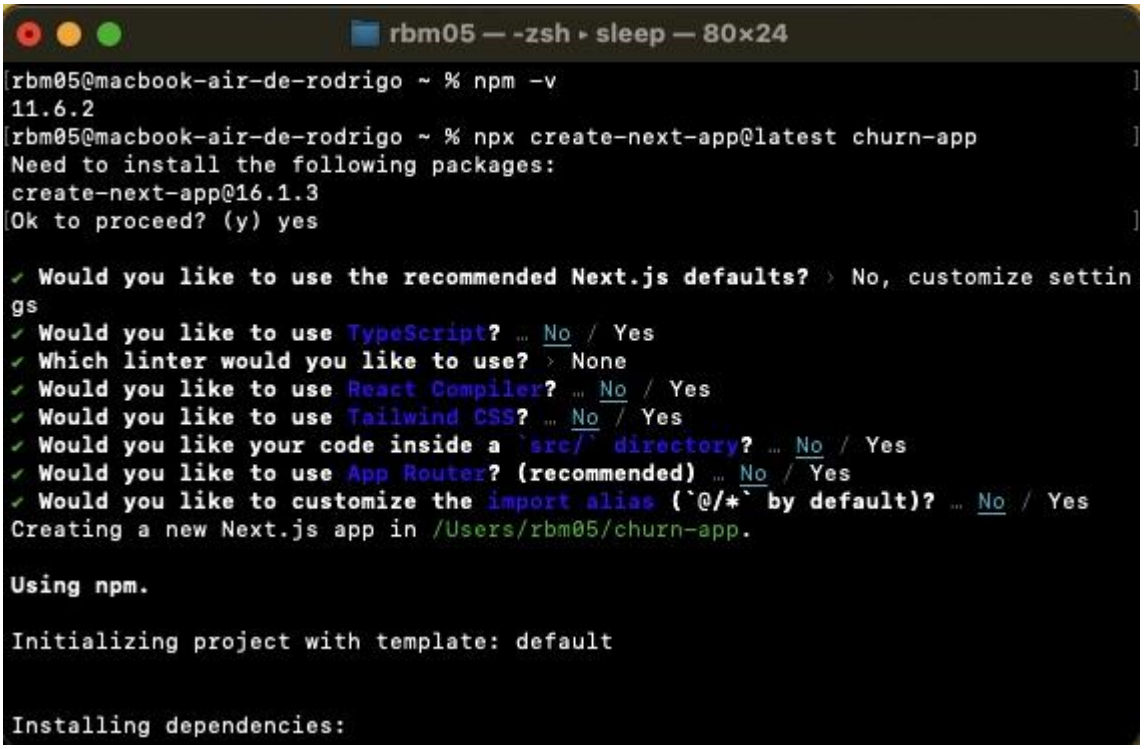
4.2 Componentes

El frontend usa Next.js con React (formulario para ingresar datos), el backend es una API Python en /api/predict, y el modelo son archivos model.pkl y scaler.pkl serializados con joblib.

La Estructura de carpetas de mi proyecto, incluye los archivos del modelo (model.pkl) y del escalador (scaler.pkl), así como la configuración de Next.js y las dependencias necesarias.



Durante el proceso de creación del proyecto usando Next.js con create-next-app. Se configuró el framework React con las opciones recomendadas para la web en Vercel.

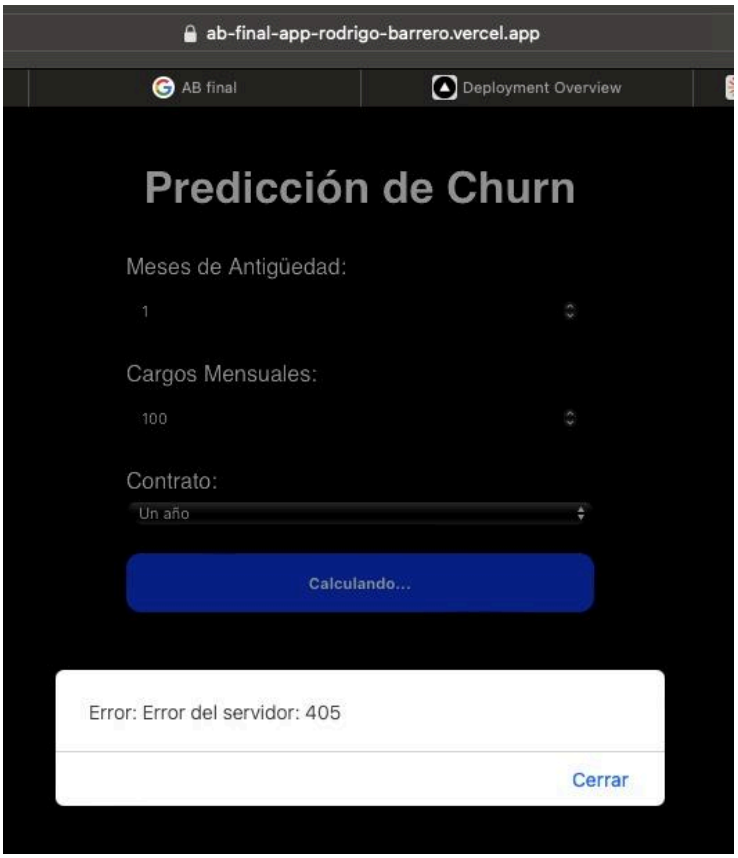


4.3 Flujo de Comunicación

El usuario completa el formulario (meses antigüedad, cargos, contrato), luego el frontend valida los datos y envía POST a /api/predict. El backend carga el modelo, preprocesa los datos, predice, y devuelve JSON con probabilidad de churn. Finalmente, si todo estuviese perfecto, el frontend debería mostrar el resultado al usuario.

4.4 Problema: Error 405

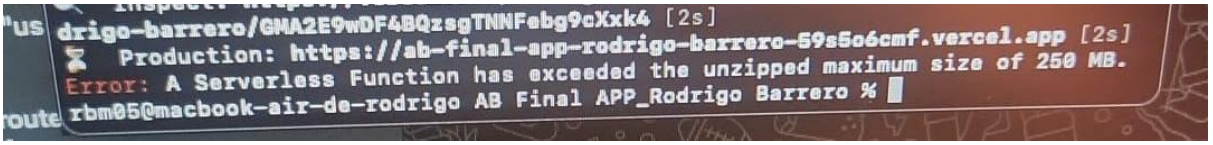
La aplicación no funciona. Al darle a predecir aparece el error 405 (Method Not Allowed).



4.5 Causas Probables

Quiero aclarar que no tengo del todo claro las causas que han hecho que mi trabajo falle, sino habría intentado solucionarlo de una mejor manera, pero después de una investigación, y de preguntarle a algunas Ia, él porque ha fallado mi proyecto, he identificado algunas de las posibles causas.

La configuración de vercel.json usa una versión anticuada y Vercel ha cambiado cómo maneja las APIs de Python, los archivos del modelo con todo incluido, son muy pesados y más de una vez me ha salido error en la terminal por superar los 250 MB de capacidad, el mapeo de rutas hacia el archivo Python puede estar mal configurado, y las versiones de las librerías en requirements.txt podrían ser incompatibles con el runtime de Vercel.



4.6 ¿Por Qué Persiste?

Creo que el problema no lo he podido solucionar porque Vercel tiene limitaciones importantes en su capa gratuita para Python serverless. Las librerías de machine learning como Scikit-learn, numpy y pandas generan una carpeta muy pesada, y además me faltó hacer un testing local más a fondo antes de crear la web.

4.7 Ciclo de Feedback (Diseño Teórico)

Aunque el sistema de feedback no está implementado por culpa del error 405, el diseño teórico sería el siguiente:

Cada vez que el modelo hace una predicción, se guardaría en una base de datos junto con las características del cliente, la probabilidad calculada y la fecha. Mensualmente se actualizaría esta base de datos con el resultado real (si el cliente efectivamente abandonó o no). Cuando se acumulen suficientes datos nuevos, se combinarían con el dataset histórico para reentrenar el modelo, se evaluaría si realmente mejora el rendimiento, y después se desplegaría la nueva versión. Esto permitiría que el modelo evolucione y se adapte en el negocio y en el comportamiento de los clientes.

5. Futuras Mejoras

Como ya ha quedado claro, este modelo necesita muchas mejoras y cambios de cara al futuro para poder llegar a ser funcional y útil, se me han ocurrido varias ideas sobre cómo podría hacerlo.

Para empezar, el modelo mejoraría mucho si pudiese añadir variables de satisfacción del cliente e incluir el historial de soporte técnico. También sería útil poder capturar datos temporales en vez de trabajar solo con una foto fija del cliente.

Desde el punto de vista del modelado, el siguiente paso sería probar algoritmos más potentes como Random Forest o XGBoost, los cuales pueden capturar relaciones no lineales entre variables. Además, habría que optimizar la toma de

decisiones considerando los costes reales del negocio y manejar mejor el desbalance de clases usando técnicas como SMOTE.

Para solucionar los problemas de la web, lo ideal sería cambiar el backend a Flask o FastAPI, que son más estables para este tipo de aplicaciones. También pienso que otras plataformas como Railway o Render en vez de Vercel habrían funcionado mejor para mi trabajo.

Finalmente, para tener un sistema completo en producción faltaría implementar una base de datos que permita el feedback loop entre predicciones y resultados reales, crear un dashboard de monitoreo para poder vigilar el rendimiento del modelo, y establecer un proceso de reentrenamiento automático cuando se acumulen nuevos datos.

6. Conclusiones

Este proyecto confirma que los contratos mensuales y la poca antigüedad son los principales indicadores de abandono de la gente. El modelo funciona decentemente con un 79% de precisión, pero tiene un problema importante, solo detecta a la mitad de los clientes que realmente se van, lo cual es insuficiente para usarlo en producción. Aun así, usar la regresión logística ha sido la decisión correcta porque me permite entender *por qué* predice cada cosa, no solo obtener un número.

La web ha fallado (error 405), y aunque es frustrante, no me ha quedado otra que intentar explicar el porqué ha fallado. En proyectos reales las cosas también se rompen. Lo importante de este trabajo para mí, no era tener un modelo perfecto, sino el aprender de todo el proceso, entender el problema, limpiar datos reales, entrenar e interpretar el modelo, y enfrentarme a los desafíos de llevarlo a la web. Prefiero un modelo imperfecto que entiendo completamente a uno con métricas perfectas que no sepa ni entender ni explicar.

7. Anexos

Enlace de la web: https://ab-final-app-rodrigo-barrero.vercel.app

Repositorio: Contiene notebook, código de la app, y configuraciones

8. Bibliografía

1. Vercel – AB Final App de Rodrigo Barrero Martins (página principal)
   BARRERO-MARTINS, R., *s. f.* AB Final App – Proyecto en Vercel. Disponible en: https://vercel.com/rodrigo-barrero-martins-projects/ab-final-app-rodrigo-barrero (Accedido: 14 enero 2026).
2. Google Colab – Notebook de Telco Churn (Drive)
   s. f. Google Colab – Notebook de Rodrigo Barrero Martins. Disponible en: https://colab.research.google.com/drive/1Myg3RDYDYPuYG2zJr6LEig-z1npSvW-8 (Accedido: 8 enero 2026).
3. YouTube – Vídeo sobre el proyecto
   s. f. YouTube: sxcKVwURuhk – Video relevante. Disponible en: https://www.youtube.com/watch?v=sxcKVwURuhk (Accedido: 17 enero 2026).
4. ChatGPT (sitio web principal)
   s. f. ChatGPT – Plataforma de OpenAI. Disponible en: https://chatgpt.com/ (Accedido: 19 enero 2026).
5. Kaggle – Dataset Telco Churn
   s. f. Telco Churn Dataset. Kaggle. Disponible en: https://www.kaggle.com/datasets/gncgulce/telco-churn(Accedido: 5 enero 2026).
6. Gemini (Google)
   s. f. Gemini de Google – Aplicación de IA. Disponible en: https://gemini.google.com/app (Accedido: 19 enero 2026).
7. Vercel – Logs del deployment AB Final App
   BARRERO-MARTINS, R., *s. f.* AB Final App – Logs en Vercel. Disponible en: https://vercel.com/rodrigo-barrero-martins-projects/ab-final-app-rodrigo-barrero/logs?selectedLogId=g92cb-1768867183260-5e8860bfdcd5&panelState=closed (Accedido: 20 enero 2026).