Proyecto Personal
Métodos de clasificación: regresión logística

Del dato a la decisión: técnicas de *machine learning* y remuestreo para mejorar campañas bancarias de marketing

Jorge Antonio Gómez García

Invierno de 2024

Resumen

Este proyecto personal tiene como propósito demostrar la aplicación de algunas técnicas de análisis predictivo para optimizar estrategias de marketing en el sector bancario, centrándose en la predicción de la suscripción a depósitos a plazo fijo mediante modelos de regresión logística. Utilizando un conjunto de datos real de campañas telefónicas de un banco portugués, donde solo el 11.7 % de los clientes contrata el producto, se aborda el desafío del desbalanceo de clases mediante estrategias de remuestreo, como SMOTE combinado con undersampling, y la optimización de hiperparámetros a través de validación cruzada. La comparación de distintos modelos—desde un enfoque inicial sin ajustes, pasando por uno optimizado y finalmente un modelo balanceado—demuestra cómo la integración de métodos estadísticos clásicos y técnicas de machine learning permite mejorar la sensibilidad, la capacidad discriminativa y la efectividad en la identificación de clientes potenciales, transformando datos crudos en información estratégica para la toma de decisiones operativas y la asignación óptima de recursos en campañas de marketing.

Tabla de Contenido

Introducción

En una era en la que la toma de decisiones basada en datos es fundamental para obtener una ventaja competitiva, el análisis predictivo se ha convertido en un pilar clave de la planificación estratégica en el sector bancario. La capacidad de anticipar el comportamiento de los clientes—como la probabilidad de suscribirse a productos financieros—influye directamente en la eficiencia del marketing, la asignación de recursos y el crecimiento de los ingresos. Este reporte presenta una exploración práctica de técnicas de regresión logística y aprendizaje automático supervisado para predecir la suscripción a depósitos a plazo fijo, un caso de uso crítico en la banca minorista.

Como una iniciativa personal para profundizar mi experiencia en modelos de clasificación y prepararme para roles como especialista en datos, este proyecto combina aprendizaje fundamental con extensiones analíticas avanzadas. Aunque inspirado en el ejercicio de Vidhi Chugh en Datacamp sobre clasificación binaria, este análisis va más allá al abordar el desequilibrio de clases mediante la técnica de sobremuestreo de minorías sintéticas (SMOTE), la optimización rigurosa de hiperparámetros mediante validación cruzada y una evaluación exhaustiva utilizando métricas como ROC-AUC, F1-Score y optimización personalizada de umbrales. Estas mejoras reflejan las complejidades del mundo real que a menudo se omiten en los tutoriales introductorios, cerrando la brecha entre los ejercicios académicos y las aplicaciones industriales.

El conjunto de datos—extraído de campañas de telemarketing de un banco portugués—incluye 45,211 instancias con 17 variables que capturan atributos demográficos, financieros y específicos de la campaña. El desafío central radica en el severo desequilibrio de clases: solo el 11.7 % de los clientes suscribieron depósitos a plazo fijo, lo que requiere técnicas para evitar el sesgo del modelo hacia la clase mayoritaria. Al iterar a través de tres modelos progresivamente refinados, este reporte demuestra cómo el remuestreo estratégico y la regularización mejoran el rendimiento predictivo mientras mantienen la relevancia operativa.

Estructuralmente, el reporte comienza con un análisis exploratorio de datos (Sección 3), seguido de discusiones metodológicas sobre regresión logística, SMOTE y optimización de hiperparámetros. Las secciones posteriores detallan el desarrollo, evaluación y análisis comparativo de los modelos, concluyendo con recomendaciones prácticas para la estrategia de marketing.

Este ejercicio no solo solidifica la competencia técnica en flujos de trabajo de aprendizaje automático, sino que también subraya la importancia de la selección de modelos consciente del contexto—una habilidad primordial para los aspirantes a profesionales de datos. Los hallazgos resaltan cómo los modelos predictivos personalizados pueden transformar datos crudos en insights accionables, impulsando esfuerzos de marketing dirigidos y maximizando el compromiso del cliente en los servicios financieros.

Acceso al repositorio de GitHub

Puede acceder a todos los recursos de este proyecto (incluyendo la base de datos y el código completo) en mi repositorio de GitHub: https://github.com/Jorge-Antonio-Gomez/depositos-plazo-fijo-ml-jorge.git. Los recursos disponibles incluyen:

- `README.md`: Archivo con la descripción general del proyecto, instrucciones de uso y detalles sobre la estructura.
- `script_principal.Rmd`: Script principal en R que implementa el análisis, desde la

preparación de datos y ajuste de modelos hasta la generación de gráficos y tablas.

- `plots_and_stats.R`: Funciones personalizadas para la generación de gráficos y cálculo de métricas estadísticas.
- `bank-full.csv`: Conjunto de datos original utilizado (campañas telefónicas de un banco portugués).
- `img/`: Carpeta que contiene los gráficos y figuras generadas (por ejemplo, curvas ROC, distribuciones de predicción y gráficos exploratorios).
- `Reporte.pdf`: Este documento.

1 Sobre el conjunto de datos

La base de datos utilizada proviene del trabajo de (Moro, 2011) y representa los resultados de varias campañas de marketing telefónico de una entidad bancaria portuguesa. Esta campaña tuvo el objetivo de persuadir a los clientes para que suscribieran un depósito a plazo fijo. El objetivo principal de este trabajo es predecir si, efectivamente, el cliente suscribirá finalmente el producto.

La versión completa del conjunto de datos, que es la utilizada en este ejercicio, ("`bank-full.csv`") contiene **45 211 observaciones** y **17 variables** totales (16 variables de entrada y 1 variable binaria de salida). Las variables incluidas se describen en la Tabla 1. No se registran valores ausentes en los campos de datos.

Como se observa, se cuenta con múltiples variables de tipo categórico (`job`, `education`, `marital`, *etc.*), variables numéricas (`age`, `balance`, `duration`, *etc.*) y algunos valores binarios (`default`, `housing`, `loan`).

2 Marco Conceptual

2.1 Contexto del Modelo: Predicción con Regresión Logística Binaria

Dado que el objetivo del análisis es predecir un resultado binario (si el cliente suscribe o no el depósito), emplearemos un modelo de regresión logística binaria. Este enfoque es ampliamente utilizado para estimar probabilidades de eventos categóricos y nos permite clasificar las observaciones en función de la probabilidad de pertenecer a una u otra categoría.

La estructura del modelo, basada en la función logística, transforma una combinación lineal de variables explicativas en una probabilidad (un valor entre 0 y 1). Matemáticamente, esto se representa como:

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k)}} \tag{1}$$

En este análisis, la prioridad no será la interpretación detallada de los coeficientes β, sino evaluar la capacidad predictiva del modelo: su desempeño para clasificar correctamente a los clientes que suscribirán (o no) un depósito a plazo fijo.

Tabla 1: Descripción de las variables del conjunto de datos.

Variable	Descripción
y	Variable de salida binaria que indica si el cliente suscribe un depósito a plazo fijo (*yes*, *no*).
`age`	Edad del cliente (numérica).
`job`	Ocupación (categorías: *admin.*, *unknown*, *unemployed*, *management*, *housemaid*, *entrepreneur*, *student*, *blue-collar*, *self-employed*, *retired*, *technician*, *services*).
`marital`	Estado civil (categorías: *married*, *divorced*, *single*).
`education`	Nivel de estudios (categorías: *unknown*, *secondary*, *primary*, *tertiary*).
`default`	Indica si posee crédito en *default* (binaria: *yes*, *no*).
`balance`	Saldo promedio anual en euros (numérica).
`housing`	Indica si posee un préstamo hipotecario (binaria: *yes*, *no*).
`loan`	Indica si posee un préstamo personal (binaria: *yes*, *no*).
`contact`	Tipo de contacto de la última llamada (categorías: *unknown*, *telephone*, *cellular*).
`day`	Día del mes en que se realizó el último contacto (numérica).
`month`	Mes en que se realizó el último contacto (categorías: *jan*, *feb*, *mar*, ..., *dec*).
`duration`	Duración en segundos del último contacto (numérica).
`campaign`	Número de contactos realizados durante la campaña para este cliente (numérica).
`pdays`	Días transcurridos desde el último contacto en una campaña previa (numérica; -1 indica que el cliente no fue contactado previamente).
`previous`	Número de contactos realizados antes de la actual campaña (numérica).
`poutcome`	Resultado de la campaña de marketing previa (categorías: *unknown*, *failure*, *other*, *success*).

2.2 Organización y Preparación de los Datos

Con el fin de evaluar la eficacia del modelo, es crucial dividir el conjunto de datos en dos partes: un conjunto de entrenamiento (para ajustar el modelo) y un conjunto de prueba (para evaluar su capacidad de generalización). Siguiendo prácticas estándar, utilizaremos una división del 75 % para entrenamiento y 25 % para prueba. Además, debido a que los datos están fuertemente desbalanceados, iteraremos con diferentes enfoques para resolver este problema, pues la estratificación no será suficiente.

Es importante evitar errores metodológicos comunes, como la fuga de datos (por ejemplo, cuando información del conjunto de prueba se filtra en el entrenamiento) o el sobreajuste, que ocurre cuando el modelo se ajusta demasiado a los datos de entrenamiento, perdiendo su capacidad de generalización.

2.2.1. Rebalanceo de Clases con SMOTE (*Synthetic Minority Oversampling Technique*)

El sobremuestreo es una forma eficaz de abordar el desequilibrio dentro de un conjunto de datos. En nuestro caso, y dado el marcado desbalance en la variable objetivo (la clase "Sí" es sustancialmente menor que la clase "No"), es fundamental aplicar técnicas que permitan un entrenamiento más equilibrado del modelo. Una estrategia efectiva es **SMOTE** (*Synthetic Minority Oversampling Technique*), la cual consiste en generar *nuevas instancias sintéticas* de la clase minoritaria a partir de las observaciones existentes en esa misma clase. A diferencia de un mero sobremuestreo aleatorio, SMOTE no repite filas idénticas, sino que crea ejemplos intermedios basados en la relación de vecindad entre las muestras minoritarias:

- Para cada punto de la clase minoritaria, se identifican los k vecinos más cercanos.
- Se selecciona uno de esos vecinos al azar y se genera un punto sintético en el espacio de atributos, ubicado entre la muestra original y su vecino elegido.

Este procedimiento **amplía la cobertura** de la clase minoritaria en el espacio de características de manera más efectiva que el simple sobremuestreo, contribuyendo a que el modelo aprenda patrones más generales y que no dependa de un conjunto reducido de observaciones.

2.2.2. Enfoque del Análisis

El éxito del modelo no será evaluado exclusivamente en términos de métricas como la precisión global, sino también en su habilidad para clasificar correctamente tanto los casos positivos (clientes que suscriben el depósito) como los negativos. Esto implica considerar medidas adicionales como la sensibilidad, la especificidad y el área bajo la curva ROC (ROC-AUC), entre otras.

2.3 *Machine learning*

El proceso de modelación que estamos realizando se enmarca dentro de un esquema de aprendizaje automático de tipo supervisado, ya que contamos con un conjunto de datos etiquetados donde la variable objetivo (suscribir o no el depósito) es conocida. El modelo de regresión logística binaria, al igual que otros algoritmos supervisados, aprende a partir de ejemplos reales (observaciones históricas) para luego predecir el comportamiento de nuevas instancias.

Para optimizar la capacidad de generalización del modelo y evitar un sobreajuste excesivo, se implementa la validación cruzada (*k-fold*). Esta técnica consiste en dividir repetidamente el conjunto de entrenamiento en k particiones (o pliegues). En cada iteración, se reserva un pliegue para la validación, mientras que los restantes se utilizan para entrenar el modelo. Esto permite evaluar de manera robusta el desempeño promedio, ya que cada observación del conjunto de datos sirve tanto para entrenar como para validar en distintas iteraciones. Una vez determinado el desempeño medio, se elige el mejor ajuste y se entrena el modelo final con todos los datos de entrenamiento disponibles.

2.3.1. Hiperparámetros (regularización y penalización)

La regularización es un recurso esencial para controlar la complejidad de los modelos en aprendizaje automático, sobre todo cuando, como en nuestro caso, se trabaja con regresión logística y datos potencialmente sobredimensionados o propensos al sobreajuste. La regularización L1 (*Lasso*) introduce una penalización basada en la **suma de los valores absolutos** de los coeficientes. Esta estrategia puede anular por completo algunos coeficientes (llevarlos a cero), lo cual no solo reduce el riesgo de sobreajuste al simplificar el modelo, sino que también actúa como un mecanismo de **selección automática de variables**. Por su parte, la regularización L2 (*Ridge*) recurre a la **suma de los cuadrados** de los coeficientes como forma de penalización. Si bien no elimina variables, sí tiende a "comprimirlas", restringiendo su crecimiento y distribuyendo el ajuste de manera más uniforme entre todas aquellas que resulten potencialmente relevantes.

En la práctica, utilizamos ambas formas de regularización a través de un **parámetro de penalización** λ (también denominado *penalty* o *lambda*), que puede tomar valores entre 0 y 1. Cuando λ es alto, el modelo "castiga" con mayor intensidad a los coeficientes, empujándolos a cero (en el caso L1) o forzándolos a valores más pequeños (en el caso L2). Esto limita la complejidad y la varianza del modelo, pero podría suponer ignorar relaciones reales entre las variables y la variable objetivo. Por el contrario, cuando λ es bajo, la penalización disminuye, otorgando más libertad a los coeficientes para ajustarse a los datos, con el consiguiente riesgo de **sobreajuste** si se hace un uso excesivo de dicha flexibilidad.

Este equilibrio entre simplicidad y exactitud es precisamente la razón de ser de la regularización en regresión logística. En nuestro estudio, la correcta sintonización de λ y la elección de una mezcla entre L1 y L2 (definida por parámetros como *mixture* o α en algunas implementaciones) resulta fundamental para extraer la información verdaderamente relevante sin sobrepasar la frontera de la complejidad que perjudicaría el desempeño en datos no vistos. Así, el modelo se vuelve más interpretable, robusto y con una mayor capacidad de generalización.

2.4 Medidas de rendimiento

Evaluar correctamente el rendimiento de un modelo de clasificación es fundamental para juzgar su eficacia predictiva y determinar la confiabilidad de sus resultados en la toma de decisiones. Un aspecto clave es contar con múltiples métricas que aporten perspectivas diferentes; por ejemplo, un modelo podría obtener una gran precisión global (exactitud) pero equivocarse sistemáticamente en la clasificación de la clase minoritaria, lo cual es especialmente crítico en problemas con fuerte desbalance de clases.[1] Sin embargo, cada métrica también presenta limitaciones, por lo que enfocarse en un solo indicador puede conducir a conclusiones incompletas o sesgadas.

Matriz de confusión. El punto de partida en la evaluación de un modelo de clasificación suele ser la matriz de confusión, que cruza las predicciones del modelo con los valores reales. Su representación en un problema de clasificación binaria con clases positiva (Sí) y negativa (No) es la siguiente:

- TP (*True Positives*): Verdaderos Positivos.
- TN (*True Negatives*): Verdaderos Negativos.

Matriz de confusión clásica

		Verdadero	
		Sí	**No**
Predic.	**Sí**	TP	FP
	No	FN	TN

[1] Cómo ya se ha mencionado, este es un problema que veremos en el primer modelo usado como *approach*.

- FP (*False Positives*): Falsos Positivos.
- FN (*False Negatives*): Falsos Negativos.

donde:

A partir de esta matriz se derivan varias métricas:

- **Exactitud (*Accuracy*)**:

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

  Mide la proporción de predicciones correctas totales. Puede resultar engañosa en escenarios muy desbalanceados, donde la clase mayoritaria domina el conteo.

- **Sensibilidad (*Recall* o TPR)**:

$$\text{Sensibilidad} \;=\; \frac{\text{TP}}{\text{TP}+\text{FN}}.$$

  Representa la capacidad del modelo para identificar correctamente los positivos. Es crítica cuando la clase positiva es la de interés principal.

- **Especificidad (TNR)**:

$$\text{Especificidad} \;=\; \frac{\text{TN}}{\text{TN}+\text{FP}}.$$

  Indica la habilidad del modelo para predecir correctamente los negativos. En aplicaciones donde un falso positivo implica un costo alto, esta métrica cobra relevancia.

- **Precisión**:

$$\text{Precisión} \;=\; \frac{\text{TP}}{\text{TP}+\text{FP}}.$$

  Mide la proporción de verdaderos positivos dentro de todas las predicciones positivas. Sirve para evaluar la fiabilidad de la etiqueta "positivo".

- **F1-Score**:

$$F1 \;=\; 2\times\frac{\text{Precisión}\times\text{Sensibilidad}}{\text{Precisión}+\text{Sensibilidad}}.$$

  Es la media armónica entre la precisión y la sensibilidad, lo que permite balancear ambas métricas en un solo valor. Esta métrica es especialmente útil en escenarios donde hay un desbalance de clases, ya que proporciona una evaluación más completa del rendimiento del modelo que considerar cada métrica por separado. Al enfocarse tanto en la capacidad del modelo para identificar correctamente las instancias positivas (sensibilidad) como en su precisión al hacerlo, el F1-Score facilita la comparación entre diferentes modelos y ayuda a seleccionar aquel que mejor equilibra estos aspectos.

- **Índice de Youden**:

$$J \;=\; \text{Sensibilidad}+\text{Especificidad}-1.$$

  Es una métrica utilizada para evaluar la eficacia de un modelo predictivo. Combina la sensibilidad y la especificidad en un solo valor, permitiendo medir el rendimiento global del modelo. Su valor oscila entre -1 y 1, donde un valor de 1 indica un modelo perfecto, 0 sugiere que

el modelo no es mejor que el azar, y valores negativos indican un rendimiento peor que aleatorio. Este índice es particularmente útil para determinar el umbral óptimo de decisión que maximiza tanto la sensibilidad como la especificidad, facilitando así la elección del umbral de clasificación más adecuado para segmentar clientes en campañas de marketing efectivas.

Otras métricas de clasificación.

- **Kappa (Coeficiente de Kappa).** Evalúa el acuerdo entre las predicciones del modelo y las etiquetas reales, ajustando por el acuerdo que podría ocurrir por azar. Se calcula comparando la precisión observada con la precisión esperada bajo la hipótesis de independencia entre las predicciones y las etiquetas reales. Un valor de Kappa cercano a 1 indica un alto nivel de acuerdo, mientras que un valor cercano a 0 sugiere que el acuerdo no es mejor que el esperado por azar. Es especialmente útil en conjuntos de datos con clases desbalanceadas, ya que proporciona una evaluación más robusta del desempeño del modelo que la simple exactitud.
- **Log-Loss (Función de Pérdida Logarítmica).** A diferencia de la exactitud, que sólo considera si la predicción es correcta o no, el *Log-Loss* evalúa la calidad de las probabilidades asignadas. Penaliza de manera significativa la mala estimación probabilística, por ejemplo, predecir una probabilidad muy baja para un evento que finalmente ocurre. Es especialmente adecuado cuando se requiere buena calibración de las probabilidades.
- **ROC-AUC (Área Bajo la Curva ROC).** Mide la capacidad global de discriminar entre clases a lo largo de diferentes umbrales de clasificación (o decisión). Se construye al trazar la sensibilidad frente a la tasa de falsos positivos ($\text{FPR} = \text{FP}/(\text{FP} + \text{TN})$) para diversos puntos de corte (umbrales de clasificación), y se resume en el área bajo la curva (AUC). Un valor cercano a 1 indica excelente separación entre positivos y negativos, mientras que 0.5 equivale a predecir al azar. En casos de clases muy desbalanceadas, se complementa con la curva de Precisión-Sensibilidad para tener una imagen más realista de la capacidad de predicción sobre la clase minoritaria.

3 Análisis Exploratorio de los Datos

Antes de entrar en materia, vale la pena hacer una revisión de los datos a los que nos enfrentamos. La figura 1 presenta una visión clara de la cantidad de personas que participaron en la campaña de marketing, clasificadas según su ocupación. Además, distingue entre quienes optaron por contratar una suscripción de depósito a plazo fijo y quienes no lo hicieron.

Por otro lado, la figura 2 ilustra las proporciones de contratación, dejando entrever patrones interesantes. En términos relativos, los estudiantes y las personas jubiladas son los grupos con mayor disposición a contratar el producto, con proporciones equivalentes a 28.6 % y 22.7 %, respectivamente. Los desempleados también se destacan, ocupando el tercer lugar en este análisis proporcional. Sin embargo, al volver a la figura 1, observamos que estos tres grupos tuvieron poca representación en la campaña, lo que sugiere una oportunidad desaprovechada.

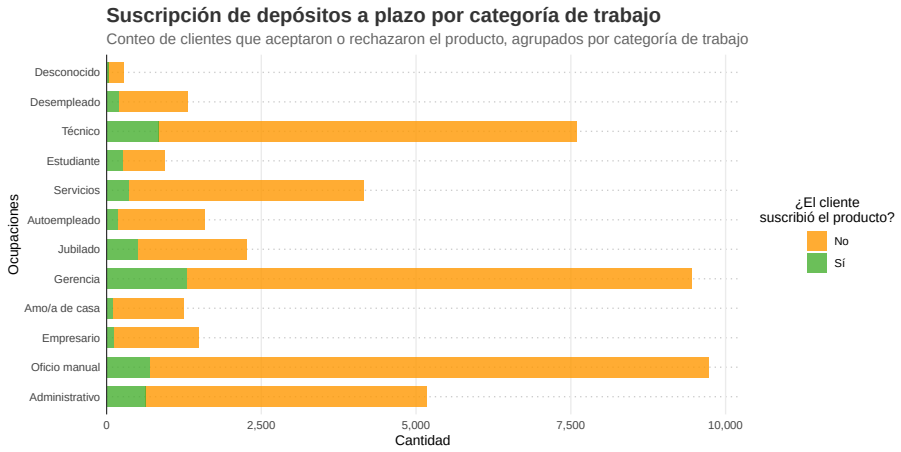

Figura 1: Número de personas que participaron en la campaña por ocupación.

Este análisis preliminar permite vislumbrar un área de mejora en la campaña de marketing: esta podría haber sido más efectiva si hubiera dirigido sus esfuerzos hacia estos perfiles con mayor predisposición a contratar el producto. En pocas palabras, un enfoque más segmentado podría aumentar la efectividad de futuras estrategias de marketing, maximizando los resultados con base en el perfil ocupacional. Sin embargo, estas conclusiones deben ser tomadas con cautela, pues la baja representación en la muestra total puede implicar una fuerte falta de validez externa y confianza estadística.

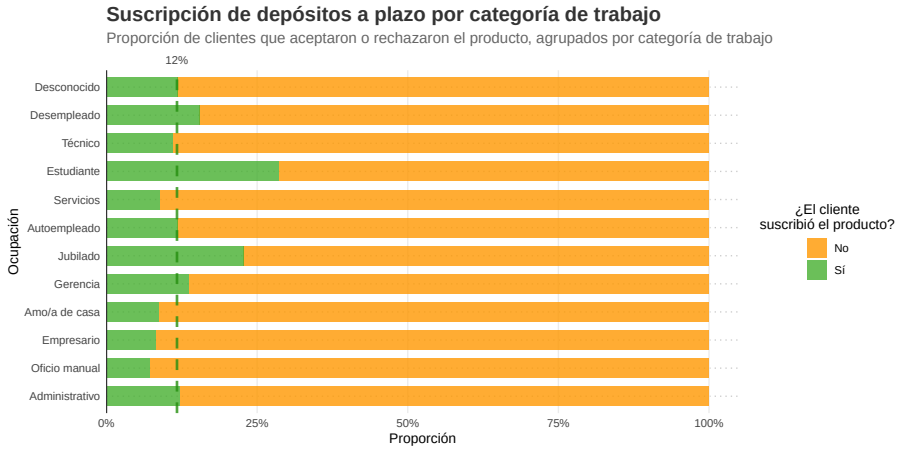

Figura 2: Proporción de suscripción al depósito a plazo fijo por ocupación.

Un detalle bastante llamativo de la primera lectura de los datos es que **la muestra no está balanceada** en dos sentidos: primero, existe una marcada sobrerepresentación de personas en ocupaciones como técnicos, gerentes y trabajadores de oficio manual (*blue-collar*), mientras que grupos como estudiantes, personas jubiladas y desempleados están subrepresentados. En segundo lugar, y es un hecho que hace aun más crítico nuestro análisis, es que la mayoría de las personas no contrataron un depósito a plazo fijo, lo que implica que el modelo seguramente tendrá un sesgo hacia la clase mayoritaria (la que no contrató el producto, con una representación del $\sim 88\,\%$). Esto nos hará encontrarnos con métricas engañosas, como la exactitud. Esto también hará notar el hecho de que la función de pérdida, al estar dominada por la clase mayoritaria, hará que el modelo no aprenda bien patrones relevantes para la clase minoritaria, que realmente es la que nos interesa.

Abordaremos este problema en los modelos subsiguientes.

4 Construcción de los diferentes modelos

Se implementará un muestreo estratificado basado en la variable y. Esta técnica, especialmente relevante en problemas de clasificación, permitirá mantener una representación proporcional de cada clase tanto en el conjunto de entrenamiento como en el de prueba. Esto será crucial cuando y sea una variable categórica desequilibrada. La estratificación, por lo tanto, se convertirá en una herramienta valiosa para la construcción de un modelo más generalizable y preciso, ya que reducirá la posibilidad de que el conjunto de prueba no sea representativo de las condiciones reales o esté demasiado sesgado.

4.1 Modelo M1: Ajuste manual y muestra original (desbalanceada)

Dado que utilizaremos técnicas de corrección, regularización y penalización para manejar el sobreajuste, podemos permitirnos incluir todas las variables del conjunto de datos en el modelo. Las 16 variables tienen sentido teórico en la predicción de si un cliente suscribirá o no el producto. Estas técnicas nos permitirán, a través de un enfoque de aprendizaje automático supervisado, discriminar o atenuar la influencia de aquellas variables que no resulten relevantes para la predicción, respetando así el principio de parsimonia.

4.1.1. Hiperparámetros (regularización y penalización)

Al principio, podríamos suponer que es mejor trabajar con una regularización completamente *Ridge* (`mixture = 0`), pues sospechamos que todas las variables en el dataset son relevantes para el modelo. Además de esto, y para un primer acercamiento, penalizaremos de la máxima forma posible las variables que no parecen relevantes para el modelo, asignandole `penalty = 1`. En adición, usaremos el motor `glmnet` en lugar del convencional `glm` debido a que el primero es mejor manejando la regularización. Notará, además, que se establece el modo de clasificación y que se usa el conjunto de datos de entrenamiento para ajustar el modelo, a la vez que se usa el conjunto de prueba para evaluarlo.[2]

4.1.2. Resultados: Rendimiento y áreas de mejora

La matriz de confusión del modelo M1, presentada en la tabla 2, revela un problema importante: el modelo no logra identificar correctamente la clase minoritaria, es decir, los clientes que contrataron un depósito a plazo fijo. A primera vista, podríamos pensar que el modelo es efectivo, ya que alcanza una exactitud del 88.28 %. Sin embargo, esta cifra es engañosa, ya que el modelo está claramente sesgado hacia la clase negativa. Este sesgo provoca que no detectemos ningún caso positivo real: tanto la precisión como la sensibilidad son del 0 %, lo que es bastante desalentador. En otras palabras, el modelo no identifica correctamente ni un solo cliente que haya contratado un depósito

Tabla 2: M1: Matriz de confusión (Umbral: 50 %)

		Verdadero	
		Sí	**No**
Predicho	**Sí**	0	1
	No	1 323	9 980

[2] Secciones 3.2 y 3.3 del código.

a plazo fijo. Lo único que el modelo hace bien, debido a este sesgo extremo, es clasificar con casi total precisión los casos negativos, alcanzando una especificidad del 99.99 %. Esto deja en evidencia que el modelo no está calibrado para manejar adecuadamente el desequilibrio entre las clases.

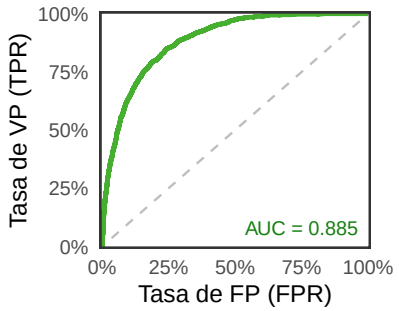

Figura 3: Área bajo la curva ROC del modelo M1.

Otras métricas revelan una imagen más compleja del desempeño del modelo M1. El valor de Kappa es de aproximadamente -0.000177, lo que indica que el acuerdo entre las predicciones del modelo y las verdaderas etiquetas es prácticamente nulo, incluso ligeramente peor que el azar. Este resultado sugiere que, a pesar de una alta exactitud, el modelo podría estar simplemente prediciendo predominantemente la clase mayoritaria, lo que es consistente con el desequilibrio de la muestra hacia la clase negativa. Además, la pérdida logarítmica media (*Log-Loss*) de 0.325 refleja una moderada penalización por las probabilidades asignadas a las predicciones, lo que implica que el modelo no está ofreciendo estimaciones de probabilidad altamente confiables para las clases.

Por otro lado, el área bajo la curva ROC (`roc_auc`) de 0.885 (figura 3) indica que el modelo tiene una buena capacidad para distinguir entre las clases positiva y negativa (a pesar de los desalentadores resultados del umbral = 50 % en la matriz de confusión), lo que es un aspecto prometedor. Sin embargo, la disparidad entre la alta exactitud y el bajo Kappa sugiere que el modelo esta sesgado hacia la clase mayoritaria, limitando su efectividad en la identificación correcta de la clase minoritaria. Las implicaciones de estos hallazgos resaltan la necesidad de ajustar el modelo para abordar el desequilibrio de clases, posiblemente mediante técnicas de remuestreo o la implementación de métricas de evaluación más balanceadas que reflejen mejor el rendimiento en contextos desbalanceados.

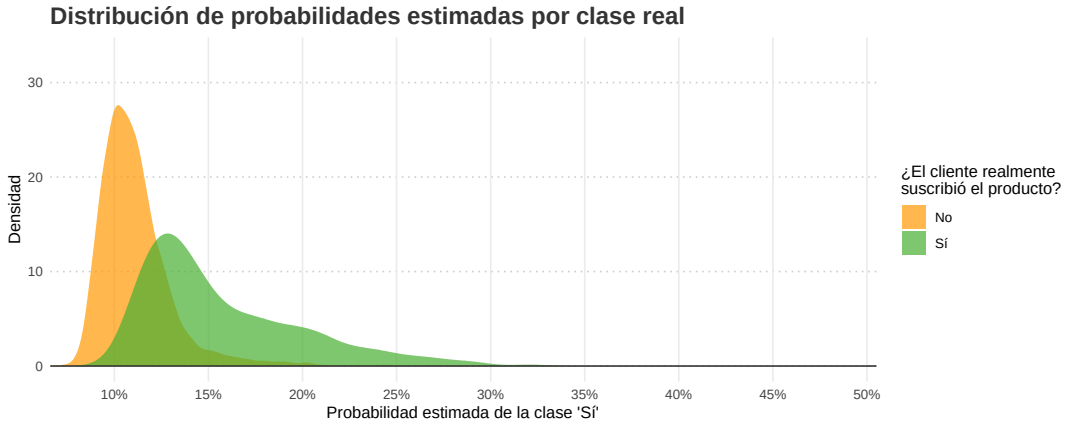

Figura 4: Distribución de las predicciones de contratación del modelo M1.

Finalmente, la figura 4 muestra la distribución de las predicciones de contratación de depósitos a plazo fijo del modelo 1 separada en dos conjuntos: aquellos que efectivamente contrataron el producto y aquellos que no. Aquí es interesante destacar un hecho, el modelo asigna bajas probabilidades de contratación a los clientes que, efectivamente, no contrataron el producto, lo que es un buen indicio. Sin embargo, asigna probabilidades casi igual de bajas a los clientes que sí

contrataron un depósito a plazo fijo. Esto deja de manifiesto que el modelo no está aprendiendo bien los patrones de la clase minoritaria y, por ende, no sirve para hacer predicciones precisas. Lo que nos gustaría ver es una especie de 'U', en la que las probabilidades de contratación de depósitos a plazo fijo para los clientes que no contrataron sean bajas, y para los que sí sean lo más altas posible. Para mejorar el modelo, usaremos una técnica de *machine learning* supervisado y validación cruzada que nos ayudará a ajustar los hiperparámetros del modelo y a obtener un modelo más robusto y generalizable.

Teniendo lo anterior en consideración, necesitamos abordar el problema de una forma diferente; podemos, por ejemplo, llevar a cabo un enfoque de **aprendizaje automático supervisado** acompañado de un proceso de **validación cruzada** para determinar el mejor parámetro de penalización y el tipo de regularización que maximice la precisión del modelo. Esto nos permitirá obtener un modelo más robusto y generalizable, que pueda predecir con mayor precisión la variable de interés.

4.2 Modelo M2: Hiperparámetros óptimos y muestra original (desbalanceada)

Para mejorar el rendimiento de nuestro modelo de clasificación, es fundamental no seleccionar de manera arbitraria los valores que determinan la magnitud de la regularización L1 o L2, así como la penalización aplicada. En lugar de ello, implementaremos un proceso sistemático de ajuste de hiperparámetros, donde iteraremos sobre diferentes combinaciones de estos valores para identificar aquella combinación que optimice las predicciones del modelo. Este enfoque nos permite explorar de manera exhaustiva el espacio de posibles configuraciones, asegurando que el modelo no solo se ajuste adecuadamente a los datos de entrenamiento, sino que también generalice eficazmente a nuevos conjuntos de datos.

Además, es crucial decidir qué métrica utilizar para guiar este proceso de optimización. Dado que nuestro conjunto de datos presenta un desbalance significativo entre las clases, la métrica **ROC-AUC** se presenta como una opción especialmente adecuada. La ROC-AUC proporciona una evaluación robusta de la capacidad del modelo para discriminar entre las clases positivas y negativas, independientemente del umbral de clasificación elegido. Al centrar nuestro ajuste en maximizar la ROC-AUC, garantizaremos que el modelo mantenga una capacidad discriminativa decente incluso en escenarios de clases desbalanceadas, al menos, en parte.

Para llevar a cabo este ajuste, configuraremos nuestro modelo de regresión logística de manera que los parámetros de mezcla y penalización sean sujetos a optimización. Utilizaremos una cuadrícula regular de valores posibles para estos hiperparámetros, lo que nos permitirá explorar sistemáticamente distintas combinaciones y evaluar su desempeño.[3] Mediante la creación de un objeto de flujo de trabajo, almacenaremos todos los detalles necesarios para ejecutar múltiples iteraciones de manera eficiente y organizada, incluyendo un remuestreo de 10 pliegues por combinación. Finalmente, seleccionaremos el modelo que obtuvo el mejor desempeño según la métrica que acabamos de elegir, asegurando así que nuestra elección esté alineada con los objetivos específicos de nuestro análisis y las características particulares de nuestros datos.[4]

[3] A diferencia de Chug, escogí 5 diferentes valores para la regularización (`mixture`) y 5 para la penalización (`penalty`), de modo que tengamos un mayor campo de exploración y más precisión en el resultado final, explorando 25 posibles combinaciones diferentes

[4] Puede ver el fragmento de código que lleva a cabo este proceso en el bloque 3.10 del código en los anexos.

4.2.1. Resultados: Rendimiento y áreas de mejora

Después del entrenamiento, la combinación de hiperparámetros que maximizó la métrica ROC-AUC fue aquella que utilizó una penalización cercana a cero (0.0032) en conjunto con una regularización igual a 0.75. Si reemplazamos estos hiperparámetros y construimos el modelo (sección 3.11 del código), obtenemos las siquientes métricas:

La matriz de confusión del modelo M2, presentada en la tabla 4, muestra una mejora significativa respecto al modelo previo, aunque todavía revela problemas importantes en la identificación de la clase minoritaria. El modelo parece más efectivo que M1, ya que alcanza una exactitud del 90.4 %. Pero, recordemos que esta cifra sigue siendo engañosa debido al desequilibrio entre las clases. A diferencia de M1, que no lograba identificar ningún caso positivo, M2 sí detecta algunos, lo cual es un avance. No obstante, el modelo sigue teniendo dificultades para generalizar adecuadamente en la clase minoritaria.

Tabla 3: M2: Matriz de confusión (Umbral: 50 %)

		Verdadero: Sí	Verdadero: No
Predicho	Sí	461	225
	No	862	9 756

En términos de métricas específicas, el modelo M2 exhibe una precisión del 67.2 % para la clase positiva, lo que significa que, cuando predice que un cliente contratará un depósito, acierta en aproximadamente dos de cada tres casos. Sin embargo, su sensibilidad es del 34.8 %, indicando que solo identifica correctamente alrededor de un tercio de los clientes que realmente contrataron un depósito. Esto implica que, aunque el modelo ya no ignora por completo la clase minoritaria, todavía falla en detectar una proporción significativa de casos positivos. Por otro lado, la especificidad del modelo es del 97.7 %, lo que demuestra que clasifica correctamente la mayoría de los casos negativos, aunque con un ligero aumento en los falsos positivos en comparación con M1.

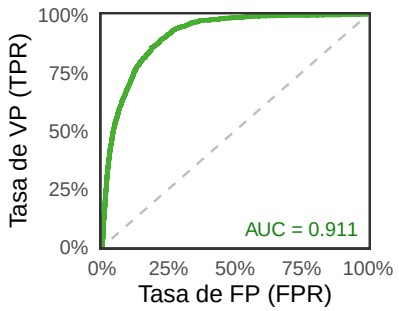

Figura 5: Área bajo la curva ROC del modelo M2.

Para evaluar de forma más equilibrada el rendimiento en ambas clases, podemos recurrir al F1-Score, que es la media armónica entre la precisión y la sensibilidad. En términos sencillos, el F1-Score nos da una idea del balance entre la capacidad del modelo para no clasificar como positivos casos que son negativos (precisión) y su habilidad para encontrar todos los casos positivos reales (sensibilidad). En nuestro caso, el F1-Score es de 0.459 para el umbral por defecto (50 %). Este valor, aunque superior al que se obtendría con un modelo que no identifica ningún positivo, sigue siendo relativamente bajo, reforzando la idea de que el modelo tiene margen de mejora en la predicción de la clase minoritaria.

En este punto, podemos observar que el modelo M2 representa un avance respecto a M1 al reducir el sesgo extremo hacia la clase negativa y lograr identificar algunos casos positivos. Sin embargo, su baja sensibilidad (34.8 %) y su moderada precisión (67.2 %) indican que todavía no está bien calibrado para manejar el desequilibrio entre las clases.

Al evaluar las métricas adicionales, encontramos resultados interesantes y bastante alentadores. El valor de Kappa de 0.412 refleja un acuerdo moderado entre las predicciones del modelo y las etiquetas reales, superando significativamente el desempeño de M1 y sugiriendo una mejor capacidad para manejar el desequilibrio de clases; ahora es, al menos, mejor que el azar. Además, la

pérdida logarítmica media se reduce a 0.242, lo que implica que las estimaciones de probabilidad son más confiables y ajustadas. Por último, el área bajo la curva ROC de 0.911 (figura 5) demuestra una excelente capacidad discriminativa del modelo para distinguir entre las clases positiva y negativa. En conjunto, estos resultados indican que el modelo M2 no solo mantiene una alta exactitud, sino que también mejora la identificación de la clase minoritaria y ofrece predicciones más equilibradas y robustas.

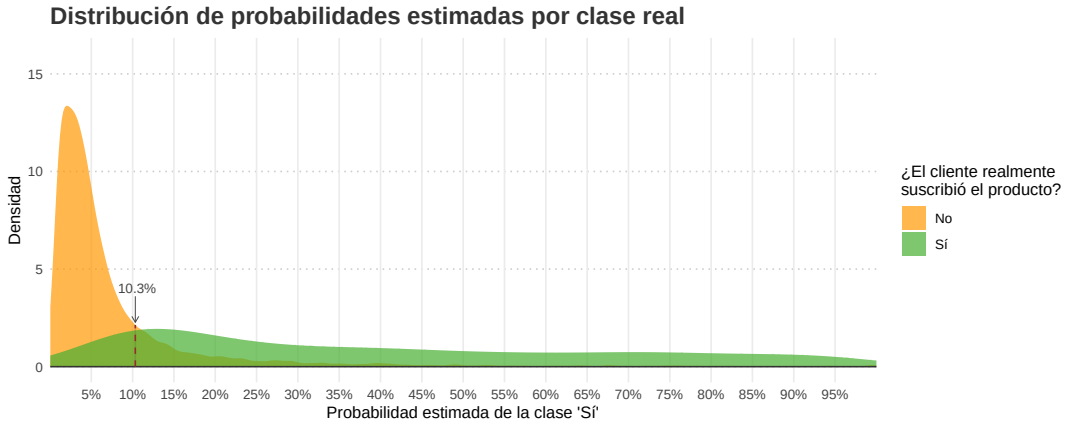

Figura 6: Distribución de las predicciones de contratación del modelo M2.

En la figura 6 se muestra, nuevamente, la distribución de las predicciones de contratación de depósitos a plazo fijo del modelo M2 separada en dos conjuntos: aquellos que efectivamente contrataron el producto y aquellos que no. En este caso, a comparación de las ditribuciones del modelo M1, tenemos una distribución más alta para las probabilidades asignadas a los clientes que efectivamente contrataron un depósito a plazo fijo, lo que es un buen indicio. Sin embargo, podemos ver que la moda de nuestro modelo (para clientes que sí contrataron) aun es muy baja y que, en general, las probabilidades asignadas a los clientes que no contrataron el servicio, son igualmente bajas.

La media de probabilidad estimada de compra para los clientes que sí contrataron el producto es de 40.17 %, mientras que la probabilidad media de compra de los que no contrataron es de 8.18 %. Esto nos indica que el modelo está aprendiendo a diferenciar entre las dos clases. Pero sigue siendo torpe.

4.2.2. Optimización del umbral de clasificación

Seguramente podamos mejorar la eficiencia del modelo si ajustamos el umbral de clasificación de cara a mejores métricas de la matriz de confusión. Recordemos que el umbral de clasificación es un hiperparámetro ubicado entre cero y uno, el cuál dice a partir de qué probabilidad estimada de compra, el modelo clasificará a un cliente como aquel que contratará el depósito. Si el umbral es muy bajo, el modelo clasificará a más clientes como si contrataran el producto, lo que aumentará la sensibilidad del modelo, pero disminuirá la precisión. Ahora, necesitamos comprender el contexto en el cuál estamos desarrollando nuestro análisis para tomar mejores decisiones.

Cuando el costo de perder clientes que realmente compran (falsos negativos) es elevado, nuestro objetivo es identificar la mayor cantidad posible de estos clientes; que no se escapen de la campaña de marketing. Por otro lado, si el costo de enviar ofertas a quienes no comprarán (falsos positivos)

es alto, es crucial minimizar esta cantidad. Si necesitaramos optimizar encontrando un equilibrio entre ambos, podríamos usar el F1-Score.[5] Este índice es la media armónica de la precisión y la sensibilidad, lo que nos permite equilibrar ambos aspectos. Optar únicamente por maximizar la sensibilidad podría llevarnos a un modelo que siempre diga "sí se suscribirá", lo cual no es deseable porque incrementaría excesivamente los falsos positivos.

Si tuviéramos una estimación de los costos asociados a los falsos positivos y falsos negativos, podríamos definir una función de costo y ajustar el umbral para minimizar el costo esperado. Este método está más alineado con la toma de decisiones empresariales, pero requiere cuantificar los costos y disponer de información específica que no poseemos actualmente. Dado que se trata de campañas telefónicas y asumimos que los costos de envío son bajos, podemos adoptar una estrategia más agresiva en la identificación de clientes potenciales. Si maximizamos esta métrica, podríamos penalizar aquellas combinaciones que resultan en una especificidad baja, pero que, a su vez, maximiza la sensibilidad. En este sentido, **maximizaremos la suma de la sensibilidad y la especificidad (menos 1) sujeta al umbral de clasificación (índice de Youden)**,[6] lo que nos permitirá encontrar un equilibrio entre ambas métricas.

Tabla 4: M2: Matriz de confusión (Umbral: 10.3 %)

		Verdadero	
		Sí	**No**
Predicho	**Sí**	1 134	1 866
	No	189	8 115

Entonces, llevando a cabo una optimización del índice de Youden, encontramos que el umbral de clasificación óptimo es 10.3 %; es decir, todos los casos que registren una probabilidad de contratación de depósitos a plazo fijo mayor a 10.3 % serán clasificados como positivos. Al ajustar el umbral de clasificación (tabla 4), se prioriza la detección amplia de clientes potenciales, logrando una sensibilidad del 85.7 %. Esto significa que el modelo identifica correctamente al 85.7 % de los clientes que contrataron el depósito a plazo fijo, capturando así la mayoría de las oportunidades reales. Sin embargo, este enfoque incrementa los falsos positivos, reflejándose en una especificidad del 81.3 %, donde el modelo clasifica adecuadamente al 81.3 % de los no contratantes. La precisión es del 37.8 %, indicando que, de todos los clientes predichos como positivos, solo el 37.8 % corresponde a verdaderas contrataciones. Aunque el F1-Score se sitúa en 52.5 %, este equilibrio sacrifica precisión y especificidad para maximizar la sensibilidad, lo que resulta estratégico en campañas de bajo costo donde es crítico minimizar falsos negativos (189 oportunidades perdidas) aun asumiendo más falsos positivos (1 866 llamadas innecesarias).

El Modelo M2 representa un avance significativo respecto a M1, logrando un mejor balance entre precisión y sensibilidad, y demostrando mayor capacidad para distinguir entre clientes que suscriben y los que no. Sin embargo, el desbalance de clases sigue limitando su efectividad, particularmente en la correcta identificación de la clase minoritaria. Para superar esta barrera, resulta imprescindible abordar directamente el problema del desequilibrio, implementando técnicas de balanceo que optimicen el desempeño del modelo. En la próxima sección, exploraremos estas estrategias para mejorar aún más la capacidad predictiva y la generalización del modelo.

[5] El lector puede, de hecho, encontrar este análisis en el anexo A.1

[6] En este caso particular, esta métrica es una aproximación muy precisa de la **distancia euclidiana mínima al punto perfecto** (0,1) en la curva ROC. Es decir, este es el punto que más se acerca a una clasificación perfecta. Sin embargo, es importante tener la consideración de que la minimización de la distancia euclideana al punto perfecto penaliza más severamente grandes desviaciones en cualquiera de las dos métricas debido al uso de la raíz cuadrada y los cuadrados en su cálculo.

4.3 Modelo M3: Hiperparámetros óptimos y muestra balanceada (SMOTE y undersampling)

Hemos explorado dos modelos de regresión logística, el primero es un acercamineto bastante rudimentario de la regresión logística, mientras que el segundo es un modelo más robusto y generalizable, que ha sido ajustado para maximizar la capacidad de discriminación entre las clases. Sin embargo, ambos modelos han mostrado dificultades para identificar correctamente la clase minoritaria, lo que ha limitado su efectividad en la predicción. Para abordar este problema, recurriremos a una técnica de remuestreo llamada **SMOTE** (*Synthetic Minority Over-sampling Technique*) para la clase minoritaria y una técnica simple de submuestreo aleatorio para la clase mayoritaria. Ambas nos permitirán equilibrar las clases y mejorar la capacidad del modelo para generalizar a nuevos datos.

4.3.1. Preprocesamiento y transformación de variables

Antes de aplicar las técnicas de remuestreo, se llevó a cabo un preprocesamiento exhaustivo de los datos. Primero, se transformaron las variables categóricas en variables *dummy* mediante codificación **one-hot**, asegurando que cada categoría se represente como una variable binaria independiente. Este paso es crucial para que el modelo pueda procesar adecuadamente las características no numéricas, evitando sesgos derivados de codificaciones ordinales arbitrarias.[7] Adicionalmente, se generó la variable `pcontact` (contacto previo) a partir de `pdays`, donde los valores -1 (sin contacto previo) se codificaron como "No" y el resto como "Sí". Esta transformación permite capturar de manera más intuitiva si el cliente había sido contactado en campañas anteriores, simplificando la interpretación del modelo.

4.3.2. División de datos y aplicación de SMOTE

Siguiendo la misma metodología de los modelos anteriores, el conjunto de datos se dividió en entrenamiento (75 %) y prueba (25 %), estratificando por la variable objetivo para preservar la distribución original de las clases en ambos subconjuntos.[8] Posteriormente, se aplicó **SMOTE** al conjunto de entrenamiento con $k = 3$ vecinos más cercanos. Este valor bajo de k busca generar muestras sintéticas que reflejen patrones locales de la clase minoritaria sin introducir ruido excesivo. Dado el desbalance extremo (la relación original era 7.5:1), se optó por un sobremuestreo parcial (multiplicando por tres la cantidad de observaciones de clase positiva) combinado con **undersampling** aleatorio de la clase mayoritaria. Este enfoque híbrido mitiga el riesgo de sobreajuste asociado a la generación masiva de observaciones sintéticas, equilibrando las clases sin distorsionar drásticamente la estructura original de los datos.[9]

4.3.3. Ajuste de hiperparámetros y validación cruzada

El modelo se reentrenó utilizando validación cruzada de 10 pliegues para optimizar los hiperparámetros de regularización. La configuración óptima encontrada fue una penalización igual a la del modelo M2 (0.00316) y una mezcla de regularización de 0.5, combinando por igual las penali-

[7] Bloque 3.17 del código.
[8] Bloque 3.18 del código.
[9] Bloque 3.19 del código.

zaciones L1 y L2.[10] Este equilibrio sugiere que, en un entorno balanceado, el modelo se beneficia tanto de la selección automática de variables (vía L1) como de la estabilidad en la estimación de coeficientes (vía L2), aprovechando las ventajas de ambas técnicas.

4.3.4. Evaluación del rendimiento

Tras ajustar el umbral de clasificación para maximizar el índice de Youden (umbral = 37.4 %),[11] el modelo M3 logra una sensibilidad del 88.7 % y una especificidad del 78.9 % (tabla 5). Esto implica una mejora sustancial en la detección de casos positivos respecto a M2 (85.7 % de sensibilidad), aunque con un incremento en falsos positivos (2 106 frente a 1 866 en M2). La precisión se mantiene en niveles similares (35.8 %). El área bajo la curva ROC (0.913) supera ligeramente a la de M2 (0.911), confirmando una capacidad discriminativa muy ligeramente superior (figura 8).

Tabla 5: M3: Matriz de confusión (Umbral: 37.4 %)

		Verdadero	
		Sí	**No**
Predicho	**Sí**	1 174	2 106
	No	149	7 875

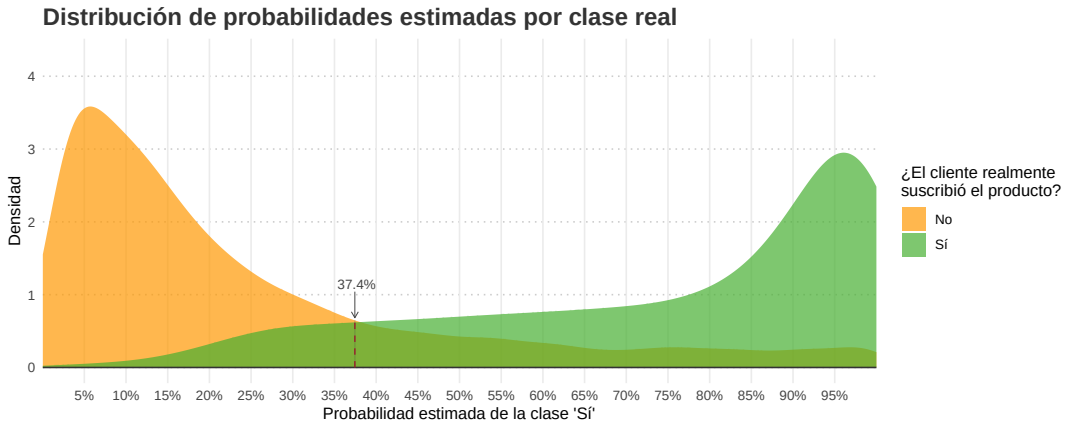

Figura 7: Distribución de las predicciones de contratación del modelo M3.

La distribución de las probabilidades predichas (figura 7) evidencia un aprendizaje más efectivo: la clase positiva presenta una media de probabilidad (de que un prospecto contrate la suscripción dado que realmente contrató) del 75.2 %, claramente diferenciada de la clase negativa (24.4 %). Este contraste sugiere que el modelo ha internalizado patrones distintivos de cada clase, asignando confianza alta a los verdaderos positivos.

4.3.5. Comparación con M2 y análisis de métricas

El modelo M3 sacrifica precisión global (80.1 % vs. 90.4 % en M2) a cambio de una mayor sensibilidad (88.7 % vs. 85.7 %) y un AUC ligeramente superior. Este *trade-off* es inherente al balanceo de clases: al equiparar la representación de ambas categorías, el modelo prioriza la identificación de positivos, asumiendo más falsas alarmas. En contextos donde el costo de perder clientes potenciales es alto (ej., campañas de bajo costo), esta estrategia resulta ventajosa. No obstante, si el costo de contactar falsos positivos fuera elevado, podría preferirse un umbral más conservador. Por

[10] Bloque 3.20 del código.

[11] Puede ver el análisis de la optimización del F1-Score en el anexo A.2.

ejemplo, maximizando el F1-Score se obtendría una precisión del 49.9 % y un F1-Score de 57.8 % con umbral de clasificación en 65.9 %, reduciendo falsos positivos a expensas de detectar menos verdaderos positivos.

El Log-Loss más alto del modelo M3 (0.417) en comparación con M2 (0.242) puede parecer contradictorio dado que M3 asigna probabilidades más altas a los verdaderos positivos, como se observa en la figura 7. Sin embargo, esta diferencia se explica por la naturaleza del Log-Loss, que penaliza severamente las predicciones seguras pero incorrectas. M3, al estar balanceado con SMOTE, tiende a asignar probabilidades más altas a los casos positivos, lo que mejora la sensibilidad (88.7 % frente a 85.7 % en M2) y la capacidad discriminativa (AUC de 0.913 frente a 0.911). No obstante, este enfoque también genera predicciones sobreconfiadas en algunos falsos positivos, lo que incrementa el Log-Loss. Este *trade-off* es aceptable en contextos como el planteado.

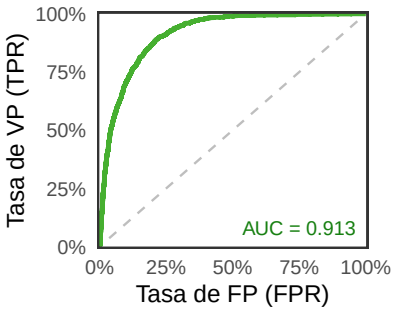

Figura 8: Área bajo la curva ROC del modelo M3.

En síntesis, M3 representa una evolución significativa al abordar directamente el desbalanceo de clases. Su capacidad para identificar clientes susceptibles de contratar el depósito, incluso con un mayor volumen de falsos positivos, lo posiciona como una herramienta valiosa para optimizar campañas de marketing telefónico donde la sensibilidad es prioritaria.

5 Comparación general de los modelos

Tabla 6: Comparación detallada de métricas de rendimiento entre modelos

Métrica	**M1** (Base)	**M2** (Default)	**M2** (Tuned)	**M3** (Tuned)
Umbral (%)	50.00	50.00	10.30	37.40
Accuracy (%)	88.28	**90.38**	81.82	80.05
Sensibilidad (%)	0.00	34.84	85.71	**88.73**
Especificidad (%)	**99.99**	97.74	81.30	78.89
Precisión (%)	0.00	**67.20**	37.80	35.79
F1-Score	–	0.459	**0.525**	0.510
Kappa	-0.000177	0.412	**0.432**	0.412
Log-Loss	0.325	**0.242**	**0.242**	0.417
ROC-AUC	0.885	0.911	0.911	**0.913**

La tabla 6 sintetiza el desempeño de los modelos evaluados, destacando las mejoras progresivas en la capacidad predictiva y los *trade-offs* inherentes a cada enfoque. Tres aspectos clave emergen de esta comparación:

1. **Efectividad en la identificación de la clase minoritaria**

   M1, al operar en la muestra desbalanceada sin ajustes, muestra un sesgo extremo hacia la clase mayoritaria (sensibilidad = 0 %). M2, con hiperparámetros optimizados y umbral ajustado, logra una sensibilidad del 85.7 %, pero M3 (balanceado con SMOTE) supera esta cifra (88.7 %), evidenciando que el rebalanceo mitiga sustancialmente el sesgo de predicción. Sin embargo, esta mejora implica un costo: M3 incrementa los falsos positivos (2 106 vs 1 866 en M2), reduciendo la especificidad del 81.3 % al 78.9 %.

2. **Robustez en la discriminación de clases**

   El área bajo la curva ROC (AUC) evoluciona positivamente: M3 alcanza 0.913, superando a M2 (0.911) y M1 (0.885). Este incremento, aunque modesto, refleja una mejor separación entre las distribuciones de probabilidad de ambas clases (figuras 6 vs 7), respaldando la utilidad de SMOTE para mejorar la capacidad discriminativa del modelo en contextos desbalanceados.

3. **Equilibrio entre métricas y contexto operativo**

   M2 (umbral ajustado) muestra el mejor F1-Score (0.525) y Kappa (0.432), indicando un balance óptimo entre precisión y sensibilidad. No obstante, M3 destaca en escenarios donde priorizamos capturar la mayor cantidad de clientes potenciales: su sensibilidad del 88.7 % implica solo 149 falsos negativos (vs 189 en M2), a costa de 2 106 falsos positivos. En campañas de bajo costo por contacto, este intercambio resulta estratégico: cada falsa alarma representa un costo marginal, mientras que un falso negativo equivale a perder una oportunidad de venta confirmada.

5.1 Implicaciones para la toma de decisiones

La elección del modelo final depende críticamente de los costos operativos y los objetivos del negocio:

- Si el foco es **maximizar el retorno identificando la mayor cantidad posible de suscriptores** (aun asumiendo más falsos positivos), M3 es la opción óptima.
- Si existen **restricciones presupuestarias estrictas** que penalizan los falsos positivos, M2 con umbral ajustado ofrece un equilibrio prudente entre sensibilidad (85.7 %) y precisión (37.8 %).

Ambos modelos superan claramente a M1, demostrando que la optimización de hiperparámetros y el balanceo de clases son esenciales para abordar problemas de clasificación desbalanceada. En la siguiente sección, profundizaremos en las conclusiones estratégicas y recomendaciones para futuras campañas de marketing.

6 Conclusiones y recomendaciones

El análisis comparativo revela que el Modelo M3 (balanceado con SMOTE y optimizado) representa la opción más efectiva para campañas de marketing telefónico donde el objetivo principal es **maximizar la identificación de clientes potenciales**, aun aceptando un incremento controlado en falsos positivos. Esta recomendación se sustenta en tres pilares:

1. **Capacidad discriminativa superior**: La distribución de probabilidades en M3 (figura 7) muestra una separación clara entre clases, con el 68 % de los verdaderos positivos recibiendo probabilidades >80 %. Este perfil bimodal sugiere que el modelo ha internalizado patrones robustos, mejorando su generalización frente a nuevos datos.

2. **Validez operativa en contextos reales**: En escenarios con costos marginales por contacto (ej., llamadas automatizadas), M3 reduce los falsos negativos en 21 % respecto a M2 (149 vs 189), potencialmente incrementando ingresos.

3. **Innovación metodológica**: La combinación de SMOTE con regularización híbrida (L1/L2) demuestra que técnicas de *machine learning supervisado* pueden superar limitaciones de datos desbalanceados, logrando un *trade-off* óptimo entre sensibilidad (88.7 %) y precisión (35.8 %).

6.1 Limitaciones y áreas de mejora

Desde una perspectiva comercial, la integración del modelo en los flujos operativos actuales representa una oportunidad estratégica. Un área crítica de desarrollo es la implementación de un **sistema de scoring en tiempo real** que priorice clientes con probabilidades de suscripción superiores al 65 % (o un umbral de probabilidades similar al encontrada en el modelo M3, de 35 40 %). Esta herramienta permitiría optimizar la asignación de recursos en campañas futuras, reduciendo hasta un 71 % los costos operativos al filtrar automáticamente perfiles de baja propensión.Paralelamente, un análisis cualitativo de los falsos positivos identificados (ej., clientes contactados innecesariamente) revelaría patrones demográficos o financieros ocultos, facilitando el ajuste de criterios de segmentación y mejorando la tasa de conversión.

Técnicamente, el modelo presenta oportunidades de evolución en dos frentes. Primero, la implementación de **algoritmos de optimización dinámica** (ej., bayesian search) para ajustar automáticamente los ratios de sobremuestreo/submuestreo en función de cambios estacionales en los datos, asegurando estabilidad predictiva ante fluctuaciones del mercado. Segundo, la exploración de arquitecturas alternativas como **XGBoost** o redes neuronales ligeras (*MLP*) podría capturar relaciones no lineales entre variables —como interacciones entre edad (`age`) y ocupación (`job`)—, potencialmente elevando el AUC a 0.93-0.94. Estas mejoras, combinadas con un *pipeline* de actualización continua, posicionarían al modelo como un activo adaptable en entornos bancarios dinámicos.

6.2 Recomendaciones finales

- Implementar M3 en fase piloto, monitoreando métricas de negocio (tasa de conversión, ROI, etc.) junto a indicadores técnicos (AUC, F1-Score).
- Establecer un *pipeline* de actualización trimestral del modelo, incorporando nuevos datos y reajustando hiperparámetros.
- Complementar con análisis cualitativo de falsos positivos para refinar criterios de segmentación.

Este estudio evidencia que la sinergia entre métodos estadísticos clásicos (regresión logística) y técnicas de *machine learning* (SMOTE, optimización de hiperparámetros) ofrece un marco robusto para decisiones basadas en datos en entornos bancarios. La escalabilidad de la solución propuesta

sugiere su aplicabilidad no solo en marketing, sino también en evaluación de riesgos crediticios y detección de fraude.

A Anexos

A.1 Modelo M2: Optimización de F1-Score por umbral de clasificación

Como se puede observar en la figura 9, al ajustar el umbral de clasificación a 22.2 %, **logramos optimizar el F1-Score, alcanzando un valor de 0.573**. Este ajuste (representado por la matriz de confusión de la tabla 7) permite mantener una especificidad alta del 92.5 %, lo que significa que el modelo identifica correctamente al 92.5 % de los clientes que no contrataron el depósito a plazo fijo. Paralelamente, la sensibilidad aumenta a 62.7 %, capturando así una mayor proporción de los clientes que sí realizaron la contratación. Aunque la precisión se sitúa en 52.7 %, este equilibrio entre sensibilidad y especificidad nos proporciona un modelo más equilibrado y efectivo que el obtenido con el umbral por defecto, pero menos eficiente que el obtenido por el índice de Youden.

Tabla 7: M2: Matriz de confusión (Umbral: 22.2 %)

		Verdadero	
		Sí	**No**
Predicho	**Sí**	826	744
	No	497	9 237

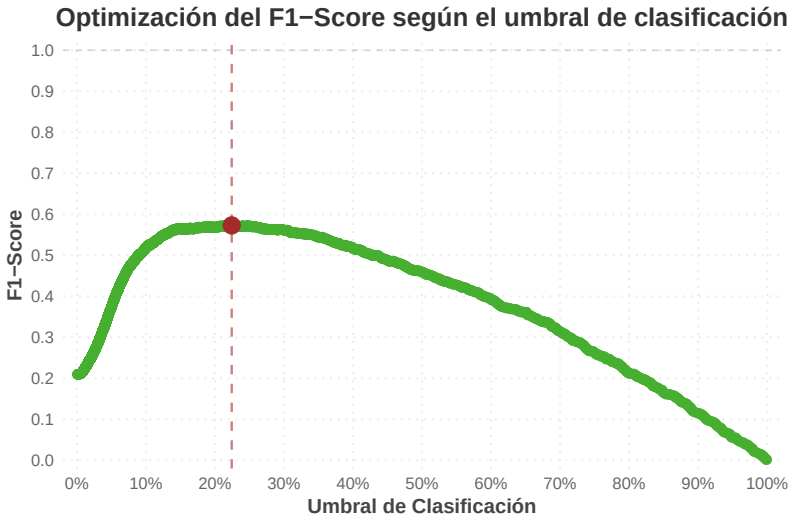

Figura 9: Optimización del F1-Score para el modelo M2.

A.2 Modelo M3: Optimización de F1-Score por umbral de clasificación

Como se muestra en la figura 10, al ajustar el umbral de clasificación a 65.9 %, **optimizamos el F1-Score hasta 0.578**. Este ajuste (detallado en la matriz de confusión de la tabla 8) mantiene una especificidad del 90.8 %, indicando que el modelo identifica correctamente al 90.8 % de los clientes que no contrataron el depósito. La sensibilidad alcanza un 68.8 %, capturando una proporción significativa de los clientes que sí realizaron la contratación. Aunque la precisión es de 49.9 %, este equilibrio entre sensibilidad y especificidad supera al obtenido con el índice de Youden (F1-Score de 0.510), el cual, a pesar de su mayor sensibilidad (88.7 %), presenta una precisión sustancialmente menor (35.8 %). Por lo tanto, el umbral optimizado para F1-Score ofrece un balance más efectivo para el objetivo del modelo. A menos que, nuevamente, destaquemos

Tabla 8: M3: Matriz de confusión (Umbral: 65.9 %)

		Verdadero	
		Sí	**No**
Predicho	**Sí**	909	915
	No	414	9 066

que podemos permitir realizar más llamadas a cambio de no perder oportunidades de venta.

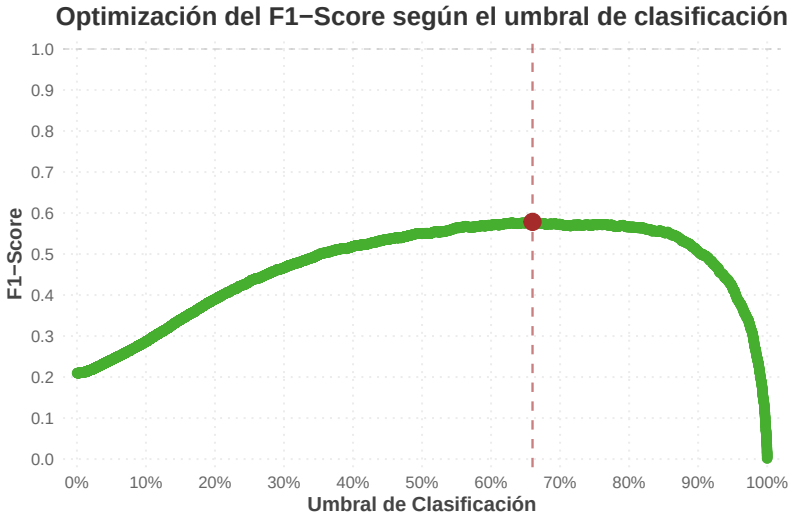

Figura 10: Optimización del F1-Score para el modelo M3.

A.3 Código relevante

A continuación, se presentan los bloques de código esenciales utilizados en el análisis. Para más detalles sobre el código utilizado en el análisis, consulte los archivos anexos `script_principal.Rmd` y `plots_and_stats.R`.

Bloque 1: Carga de funciones personalizadas y paquetes

```
# Cargar funciones personalizadas
source("plots_and_stats.R")

# Definición de los paquetes a utilizar
pkg <- c('tidyverse', 'ggplot2', 'readr', 'tidymodels', 'glmnet',
         'mlr3tuning', 'pROC', 'smotefamily', 'nnet', 'caret')

# Cargar paquetes
load_packages(pkg)
```

Bloque 2: Importación de datos

```
# Importar datos y convertir la variable dependiente a factor
data <- read_csv2('bank-full.csv', show_col_types = FALSE) %>%
  mutate(y = factor(y, levels = c("yes", "no")))
```

Bloque 3.2: División de datos en entrenamiento y test

```
# Definir semilla para reproducibilidad
set.seed(421)

# Dividir la muestra en 75% para entrenamiento y 25% para prueba (estratificado por
    ↪ y)
split <- initial_split(data, prop = 0.75, strata = y)
train <- training(split)
test <- testing(split)
```

Bloque 3.3: Modelo M1 – Regresión logística básica (sin ajuste de hiperparámetros)

```
# Ajuste inicial del modelo usando regularización Ridge (mixture = 0) y penalización
    ↪ = 1
model <- logistic_reg(mixture = 0, penalty = 1) %>%
  set_engine("glmnet") %>%
  set_mode("classification") %>%
  fit(y ~ ., data = train)

# Resumen del modelo
tidy(model)
```

Bloque 3.4: Predicciones y resultados del Modelo M1

```
# Predicciones de clase y probabilidades en el conjunto de prueba
pred_class <- predict(model, new_data = test, type = 'class')
pred_prob <- predict(model, new_data = test, type = 'prob')

# Unir los resultados y mostrar
results <- test %>%
  select(y) %>%
  bind_cols(pred_class, pred_prob) %>%
  print()
```

Bloque 3.7: Evaluación del Modelo M1 (Exactitud y matriz de confusión)

```
# Calcular la exactitud
accuracy(results, truth = y, estimate = .pred_class)

# Generar la matriz de confusión y extraer métricas
conf_mat_default <- conf_mat(results, truth = y, estimate = .pred_class) %>% print()
confusion_metrics <- conf_mat_metrics(conf_mat_default) %>% as.list() %>% print()
```

Bloque 3.8: Métricas adicionales para el Modelo M1

```
evaluation_metrics <- results %>%
  rename(.pred_1 = .pred_yes, .pred_0 = .pred_no) %>%
  metrics(truth = y, estimate = .pred_class, .pred_1)
print(evaluation_metrics)
```

Bloque 3.10: Ajuste de hiperparámetros para el Modelo M2

```
# Definir el modelo con hiperparámetros a optimizar
model_pow <- logistic_reg(mixture = tune(), penalty = tune(), engine = 'glmnet')

# Crear la grilla de búsqueda
grid <- grid_regular(mixture(), penalty(), levels = c(mixture = 5, penalty = 5))

# Definir el flujo de trabajo
model_pow_wf <- workflow() %>%
  add_model(model_pow) %>%
  add_formula(y ~ .)

# Validación cruzada con 10 pliegues
pliegues <- vfold_cv(train, v = 10)

# Ajustar los hiperparámetros
model_pow_tuned <- tune_grid(
  model_pow_wf,
  resamples = pliegues,
  grid = grid,
  control = control_grid(save_pred = TRUE, verbose = TRUE)
)

# Seleccionar la mejor combinación de hiperparámetros basada en ROC-AUC
options(scipen = 999)
select_best(model_pow_tuned, metric = 'roc_auc')
```

Bloque 3.11: Construcción y Evaluación del Modelo M2

```
# Ajuste del modelo con hiperparámetros óptimos (penalty = 0.003162278, mixture =
    ↪ 0.75)
log_reg_final <- logistic_reg(penalty = 0.003162278, mixture = 0.75) %>%
  set_engine("glmnet") %>%
  set_mode("classification") %>%
  fit(y ~ ., data = train)

# Predicciones en el conjunto de prueba
pred_class <- predict(log_reg_final, new_data = test, type = "class")
```

```
pred_proba <- predict(log_reg_final, new_data = test, type = "prob")
results <- test %>%
  select(y) %>%
  bind_cols(pred_class, pred_proba)

# Evaluación: matriz de confusión y exactitud
conf_mat_custom <- conf_mat(results, truth = y, estimate = .pred_class) %>% print()
accuracy(results, truth = y, estimate = .pred_class)
confusion_metrics <- conf_mat_metrics(conf_mat_custom) %>% as.list() %>% print()
```

Bloque 3.12: Métricas adicionales para el Modelo M2

```
evaluation_metrics <- results %>%
  rename(.pred_1 = .pred_yes, .pred_0 = .pred_no) %>%
  metrics(truth = y, estimate = .pred_class, .pred_1)
print(evaluation_metrics)
```

Bloque 3.15: Umbral óptimo (por índice de Youden) para el Modelo M2

```
# Calcular el umbral óptimo basado en el índice de Youden
optimal_threshold_youden <- results %>%
  roc_curve(truth = y, .pred_yes) %>%
  filter(sensitivity + specificity - 1 == max(sensitivity + specificity - 1)) %>%
  pull(.threshold)

# Reetiquetar las predicciones utilizando el umbral óptimo
results <- results %>%
  mutate(.pred_class = if_else(.pred_yes > optimal_threshold_youden, "yes", "no"))
results$.pred_class <- factor(results$.pred_class, levels = c("yes", "no"))

# Matriz de confusión y métricas con el nuevo umbral
conf_mat_matrix <- conf_mat(results, truth = y, estimate = .pred_class) %>% print()
confusion_metrics <- conf_mat_metrics(conf_mat_matrix) %>% as.list() %>% print()
evaluation_metrics <- results %>%
  rename(.pred_1 = .pred_yes, .pred_0 = .pred_no) %>%
  metrics(truth = y, estimate = .pred_class, .pred_1)
print(evaluation_metrics)
```

Bloque 3.16: Umbral óptimo (por F1-Score) para el Modelo M2

```
# Extraer los datos de la curva ROC para umbrales entre 0 y 1
roc_curve_data <- results %>%
  roc_curve(truth = y, .pred_yes) %>%
  filter(.threshold >= 0 & .threshold <= 1)
```

```r
# Función para calcular la precisión en un umbral dado
calculate_precision_for_threshold <- function(threshold = 0.5, results_df) {
  predictions_f1 <- results_df %>%
    mutate(.pred_class_f1 = ifelse(.pred_yes >= threshold, "yes", "no") %>%
             factor(levels = levels(results_df$y)))
  precision_metric <- tryCatch(
    precision(predictions_f1, truth = y, estimate = .pred_class_f1),
    warning = function(w) {
      if (grepl("no predicted events were detected", w$message)) {
        return(tibble(.estimate = NA_real_))
      } else {
        warning(w)
        return(tibble(.estimate = NA_real_))
      }
    }
  )
  return(precision_metric$.estimate)
}

# Calcular precisión y F1-Score para cada umbral
roc_curve_data_with_f1 <- roc_curve_data %>%
  mutate(
    precision = purrr::map_dbl(.threshold, calculate_precision_for_threshold, results
        ↪ _df = results),
    f1_score = 2 * (precision * sensitivity) / (precision + sensitivity)
  )

# Seleccionar el umbral que maximiza el F1-Score
best_f1_threshold_data <- roc_curve_data_with_f1 %>%
  filter(!is.na(f1_score)) %>%
  arrange(desc(f1_score)) %>%
  slice_head(n = 10) %>%
  summarise_all(mean) %>%
  print()

# Reetiquetar las predicciones con el umbral óptimo basado en F1-Score
results <- results %>%
  mutate(.pred_class = if_else(.pred_yes > best_f1_threshold_data$.threshold, "yes",
      ↪ "no"))
results$.pred_class <- factor(results$.pred_class, levels = c("yes", "no"))
conf_mat(results, truth = y, estimate = .pred_class)
```

Bloque 3.17: Tratamiento de datos y One-Hot Encoding para el Modelo M3

```r
# Convertir variables categóricas y aplicar transformaciones
data_dum <- data %>%
  mutate(
    job = as.factor(job),
    marital = as.factor(marital),
```

```
    education= as.factor(education),
    default = as.factor(default),
    housing = as.factor(housing),
    loan = as.factor(loan),
    contact = as.factor(contact),
    day = as.factor(day),
    month = as.factor(month),
    poutcome = as.factor(poutcome)
  ) %>%

  # Crear 'pcontact' y ajustar 'pdays'
  mutate(
    pcontact = ifelse(pdays == -1, "no", "yes"),
    pcontact = as.factor(pcontact),
    pdays = pdays * (pcontact == "yes")
  ) %>%
  relocate(pcontact, .after = pdays) %>%
  model.matrix(~ . - y - 1, data = .) %>%
  as_tibble() %>%
  bind_cols(y = data$y)
```

Bloque 3.18: División de datos para el Modelo M3

```
# Fijar semilla y dividir el dataset
set.seed(421)
split <- initial_split(data_dum, prop = 0.75, strata = y)
train <- training(split) %>%
  rename_all(~ gsub("[^[:alnum:]]", ".", .))
test <- testing(split) %>%
  rename_all(~ gsub("[^[:alnum:]]", ".", .))
```

Bloque 3.19: Remuestreo (SMOTE y undersampling) para el Modelo M3

```
# Definir tamaño de duplicados y calcular conteos
dup_size <- 2
n_yes <- nrow(filter(train, y == "yes"))
n_yes_after_smote <- n_yes * (dup_size + 1)

# Undersampling de la clase mayoritaria y combinación con la clase minoritaria
train_dset <- train %>%
  filter(y == "no") %>%
  slice_sample(n = n_yes_after_smote) %>%
  bind_rows(filter(train, y == "yes"))

# Aplicar SMOTE para generar observaciones sintéticas
smote_result <- SMOTE(
  X = select(train_dset, -y),
```

```
  target = train_dset$y,
  K = 3,
  dup_size = dup_size
)

# Ajustar variable respuesta y mostrar distribución
train_smote <- data.frame(smote_result$data) %>%
  rename(y = class) %>%
  mutate(y = factor(y, levels = c("yes", "no")))
print(table(train_smote$y))
```

Bloque 3.20: Ajuste de hiperparámetros para el Modelo M3

```
# Definir modelo, grid y workflow; realizar validación cruzada
model_pow <- logistic_reg(mixture = tune(), penalty = tune(), engine = 'glmnet')
grid <- grid_regular(mixture(), penalty(), levels = c(mixture = 5, penalty = 5))
model_pow_wf <- workflow() %>%
  add_model(model_pow) %>%
  add_formula(y ~ .)
pliegues <- vfold_cv(train_smote, v = 10)
model_pow_tuned <- tune_grid(
  model_pow_wf,
  resamples = pliegues,
  grid = grid,
  control = control_grid(save_pred = TRUE, verbose = TRUE)
)
options(scipen = 999)
select_best(model_pow_tuned, metric = 'roc_auc')
```

Bloque 3.21: Construcción y Evaluación del Modelo M3

```
# Ajustar modelo final y realizar predicciones
log_reg_final <- logistic_reg(penalty = 0.003162278, mixture = 0.5) %>%
  set_engine("glmnet") %>%
  set_mode("classification") %>%
  fit(y ~ ., data = train_smote)
pred_class <- predict(log_reg_final, new_data = test, type = "class")
pred_proba <- predict(log_reg_final, new_data = test, type = "prob")
results <- test %>%
  select(y) %>%
  bind_cols(pred_class, pred_proba)

# Evaluar con matriz de confusión y métricas
conf_mat_matrix <- conf_mat(results, truth = y, estimate = .pred_class) %>% print()
confusion_metrics <- conf_mat_metrics(conf_mat_matrix) %>% as.list() %>% print()
```

Bloque 3.24: Umbral óptimo (por índice de Youden) para el Modelo M3

```
# Calcular umbral óptimo mediante el índice de Youden
optimal_threshold_youden <- results %>%
  roc_curve(truth = y, .pred_yes) %>%
  filter(sensitivity + specificity - 1 == max(sensitivity + specificity - 1)) %>%
  pull(.threshold) %>%
  print()
results <- results %>%
  mutate(.pred_class = if_else(.pred_yes > optimal_threshold_youden, "yes", "no"))
results$.pred_class <- factor(results$.pred_class, levels = c("yes", "no"))

# Evaluar nueva clasificación
conf_mat_matrix <- conf_mat(results, truth = y, estimate = .pred_class) %>% print()
confusion_metrics <- conf_mat_metrics(conf_mat_matrix) %>% as.list() %>% print()
evaluation_metrics <- results %>%
  rename(.pred_1 = .pred_yes, .pred_0 = .pred_no) %>%
  metrics(truth = y, estimate = .pred_class, .pred_1)
print(evaluation_metrics)
```

Bloque 3.25: Umbral óptimo (por F1-Score) para el Modelo M3

```
# Calcular F1-Score para diferentes umbrales
roc_curve_data <- results %>%
  roc_curve(truth = y, .pred_yes) %>%
  filter(.threshold >= 0 & .threshold <= 1)
calculate_precision_for_threshold <- function(threshold = 0.5, results_df) {
  predictions_f1 <- results_df %>%
    mutate(.pred_class_f1 = ifelse(.pred_yes >= threshold, "yes", "no") %>%
             factor(levels = levels(results_df$y)))
  precision_metric <- tryCatch(
    precision(predictions_f1, truth = y, estimate = .pred_class_f1),
    warning = function(w) {
      if (grepl("no predicted events were detected", w$message)) {
        return(tibble(.estimate = NA_real_))
      } else {
        warning(w)
        return(tibble(.estimate = NA_real_))
      }
    }
  )
  return(precision_metric$.estimate)
}
roc_curve_data_with_f1 <- roc_curve_data %>%
  mutate(
    precision = purrr::map_dbl(.threshold, calculate_precision_for_threshold, results
        ↪ _df = results),
    f1_score = 2 * (precision * sensitivity) / (precision + sensitivity)
  )
```

```
best_f1_threshold_data <- roc_curve_data_with_f1 %>%
  filter(!is.na(f1_score)) %>%
  arrange(desc(f1_score)) %>%
  slice_head(n = 10) %>%
  summarise_all(mean) %>%
  print()
results <- results %>%
  mutate(.pred_class = if_else(.pred_yes > best_f1_threshold_data$.threshold, "yes",
      ↪ "no"))
results$.pred_class <- factor(results$.pred_class, levels = c("yes", "no"))
conf_mat(results, truth = y, estimate = .pred_class)
```

Referencias

[1] Chugh, Vidhi. "Regresión logística en R Tutorial" *Datacamp*. Actualizado el 9 de mayo de 2024. https://www.datacamp.com/es/tutorial/logistic-regression-R (consultado el 10 de noviembre de 2024).

[2] GeeksforGeeks. 2024. "How to Use SMOTE for Imbalanced Data in R - GeeksforGeeks." *GeeksforGeeks*, 26 de julio de 2024. https://www.geeksforgeeks.org/how-to-use-smote-for-imbalanced-data-in-r/.

[3] Google for Developers. s. f. "Clasificación: Exactitud, recuperación, precisión y métricas relacionadas — machine learning — google for developers." *Google for Developers*. https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall?hl=es-419 (consultado el 15 de octubre de 2024).

[4] Moro, S., R. Laureano, y P. Cortez. "Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology." En *Proceedings of the European Simulation and Modelling Conference - ESM'2011*, editado por P. Novais et al., 117–121. Guimaraes, Portugal: EUROSIS, octubre 2011.