UNIVERSIDAD DE BUENOS AIRES
FACULTAD DE CIENCIAS EXACTAS Y NATURALES
MAESTRÍA EN ESTADÍSTICA MATEMÁTICA

Tesis de Maestría

Autor

Ing. Ignacio José Stivala

TÍTULO DE LA TESIS: Modelo de predicción de compra de tarjeta de crédito

Tesis presentada para optar al Título de Magister de la Universidad de Buenos Aires en Estadística Matemática

Directora: Dra. Daniela A. Rodriguez
Codirectora: Mgt. Ludmila V. Venturini
Lugar de trabajo: Buenos Aires

Fecha de presentación del ejemplar: 10/04/2023
Fecha de Defensa: 21/06/2023

FIRMA DEL MAESTRANDO

RESUMEN

El trabajo muestra el desarrollo completo de un modelo de clasificación binario, aplicado a un problema real dentro de la industria bancaria, que presenta gran cantidad de datos (2.380.000 registros y 1.400 variables) y gran desbalanceo (1,8 %). La variable respuesta es si el cliente compra o no una tarjeta de crédito, y el parámetro de interés la probabilidad de dicha compra. Se desarrolló en un contexto macro económico de alta inflación, requiriendo trabajar con variables monetarias. Las etapas desarrolladas son armado de base, limpieza y preprocesamiento, selección de variables/reducción de dimensión, aplicación de algoritmos, evaluación y selección del modelo final. Se utilizan diversas técnicas con el objetivo de obtener la mejor predicción: regresión logística, Lasso, Ridge, Partial Least Squares-Discriminant Analysis (PLS-DA), Random Forest, Gradient Boosting Tree y Light Gradient Boosting Machine (light GBM). Para la explicación de las variables se utilizan SHapley Additive exPlanations (SHAP). Además, se muestra un análisis que permite decidir si es conveniente trabajar con un modelo global o dos modelos distintos separando al universo por una variable independiente.

Palabras claves: Clasificación binaria, desbalanceo, selección de variables, reducción de dimensión, regresión logística, PLS-DA, Random Forest, light GBM, SHAP.

ABSTRACT

Thesis title: Credit card purchase prediction model.

This work shows the complete development of a binary classification model, applied to a real problem in the banking industry, which presents a large amount of data (2,380,000 records and 1,400 variables) and a large imbalance (1.8 %). The response is whether or not the customer buys a credit card, and the interest variable is the probability of that purchase. It was developed in a macroeconomic context of high inflation, requiring work with monetary variables. The stages developed are collecting data, cleaning and pre-processing, variable selection/dimension reduction, application of algorithms, evaluation and selection of the final model. Various techniques are used in order to obtain the best prediction: logistic regression, Lasso, Ridge, Partial Least Squares-Discriminant Analysis (PLS-DA), Random Forest, Gradient Boosting Tree and Light Gradient Boosting Machine (light GBM). SHapley Additive exPlanations (SHAP) are used for the explanation of the variables. In addition, an analysis is shown that allows deciding if it is convenient to work with a global model or two different models splitting the universe by an independent variable.

Keywords: Binary classification, imbalance data, variable selection, dimension reduction, logistic regression, PLS-DA, Random Forest, light GBM, SHAP.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

1.1. Descripción del problema

El problema surge dentro de una industria bancaria que desea optimizar la gestión de ventas de tarjetas de crédito. En esta situación, el banco realiza diferentes campañas comerciales, categorizadas por distintos perfiles de clientes, productos y canales de gestión. A grandes rasgos, el proceso global de una campaña consta de una definición de la misma, armado de la base, evaluación crediticia, enriquecimiento de datos, gestión y seguimiento.

La problemática en particular, se enfoca en la venta de tarjetas de crédito para clientes y busca optimizar la gestión del ejecutivo comercial, quién dispone de gran cantidad de datos y poco tiempo. Por este motivo, a las etapas del proceso mencionado, se le agrega un modelado para obtener una predicción de compra. Así el ejecutivo podrá tener este dato al momento de gestionar la campaña, pudiendo optimizar su trabajo y comunicarse primero con los clientes más propensos a comprar la tarjeta de crédito.

Por otro lado, el banco posee modelos de predicción de compra actualmente en producción, sin embargo, los mismos han sido desarrollados hace más de 10 años. Por dicho motivo, realizar un modelo nuevo permitiría una mejora en los siguientes puntos:

- Incorporar variables que antes no existían.
- Corregir variables monetarias nominales que han quedado desactualizadas a causa de inflación.
- Utilizar un lenguaje de programación nuevo, con amplia variedad de herramientas.
- Mejorar la capacidad predictiva de los modelos actuales.

1.2. Antecedentes sobre el tema

Los modelos de regresión logística [1] han sido ampliamente utilizados en el ámbito bancario al otorgar una buena predicción de una variable binaria [2], combinado con una fácil interpretación del modelo. Este último punto es muy importante, dado que dichos modelos son muchas veces aplicados por personas vinculadas con el negocio que no necesariamente tienen un alto conocimiento estadístico. Consecuentemente, es una gran ventaja la relativamente simple explicación que tienen los coeficientes, cuando el modelo no presenta interacción entre sus variables.

Por otra parte, la gran capacidad que tienen las empresas de recabar información y almacenar muchas variables, fuerzan a contemplar otras herramientas para el modelado que utilicen toda la información disponible para obtener mejores predicciones de la variable de interés. Esta disponibilidad de información se encuentra acompañada del gran desarrollo producido en la ultima década de nuevos algoritmos de predicción cuyo objetivo se vincula más con su poder predictivo que con su interpretabilidad [3], y que contemplan un alto número de registros u observaciones, como así también un alto número de variables.

1.3. Modelos actuales

Actualmente se disponen de dos modelos completamente productivos desarrollados hace más de 10 años, cada uno con sus variables y parámetros particulares. Debido a este tiempo, dichos modelos han perdido capacidad predictiva, evidenciando una notoria pérdida de estabilidad. Además, muchas de sus variables han tenido cambios en su distribución respecto al desarrollo, algunas dejaron de estar operativas y otras han sufrido una desactualización por inflación, ya que son valores monetarios nominales.

Los dos modelos parten de dividir al universo original por una variable: si el cliente cobra sus haberes en el banco o no, en adelante la llamaremos *HA*. Al momento del desarrollo, el perfil del cliente era muy distinto discriminando por dicha variable, cobrar el sueldo en el banco representa un perfil mucho más comprador. La razón de realizar dos modelos es solo para obtener una mejor predicción, no por un motivo de negocio o proceso. Además, al ser un solo producto, tarjetas de crédito, los dos vectores de predicciones calculados (*HA* y *NO HA*) se unifican en uno solo, que se envía a gestión. Para ambos casos, la técnica utilizada corresponde a regresión logística sin interacción.

A continuación, se muestran métricas calculadas en una campaña correspondiente al primer trimestre de 2022 (*1Q22*), que evidencian el estado de ambos modelos. En primer lugar, la Figura 1.1 muestra la estabilidad por ventiles: se calcula la probabilidad de compra estimada acumulada por ventil, para la base de desarrollo y para la base en *1Q22*, y se comparan las distribuciones. Ambos gráficos muestran corrimientos respecto al desarrollo. Se calculó el Ks (Kolmogorov Smirnov), como la mayor distancia entre ambas distribuciones. Para el modelo *HA*, el Ks vale 17,8 % (ventil 14) y para *NO HA* 24,7 % (ventil 10).

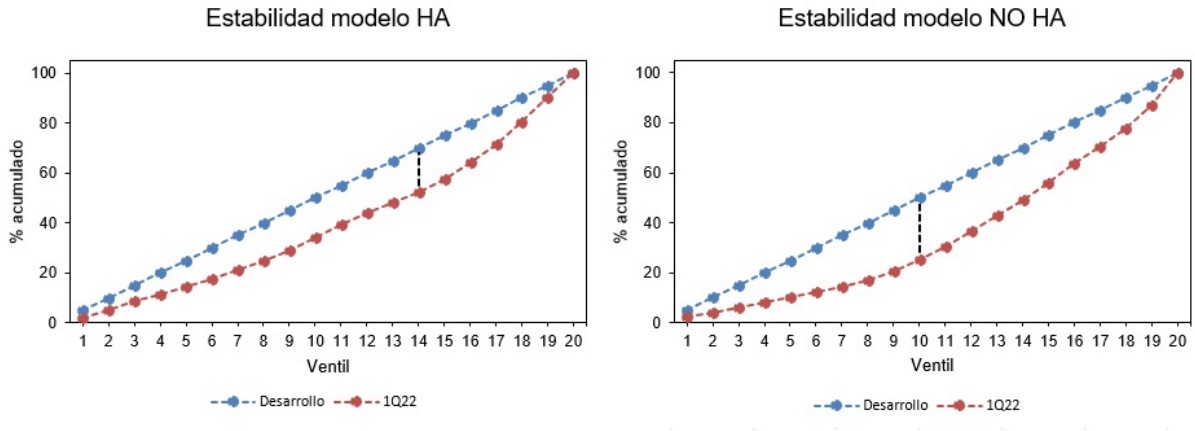

Figura 1.1: Estabilidad de modelos actuales.

En la Figura 1.2, se muestra el ordenamiento: se generan deciles de la probabilidad de compra estimada, luego se ordena de mayor a menor y para cada grupo se calcula la variable respuesta real ($\bar{y}$ por decil). Aquí, un buen modelo mostraría un orden decreciente. Sin embargo, el modelo *HA* no ordena. Respecto a *NO HA*, tiene desordenamiento en algunos deciles. Otro punto interesante a resaltar, la diferencia del target en ambos universos: 3,89 % de clientes compran tarjetas de crédito en *HA* y 0,96 % en *NO HA*.

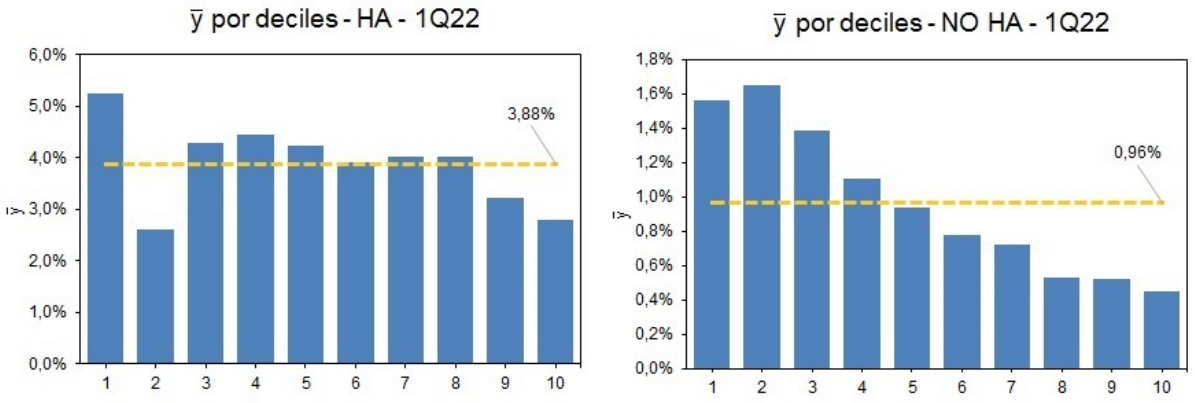

Figura 1.2: Ordenamiento por deciles de modelos actuales.

La Figura 1.3, compara la estimación del modelo ($\hat{y}$) contra target real ($\bar{y}$). El caso ideal (color rojo) debería ser una diagonal. En ambos modelos no es así.

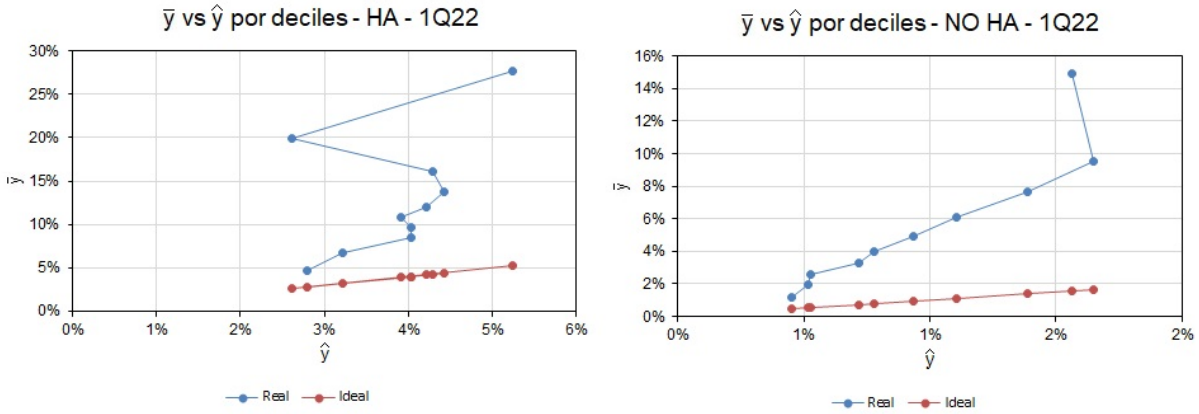

Figura 1.3: Target real vs probabilidad de compra estimada de modelos actuales.

Por último, se introduce una variable que consta de agrupar los deciles y calcular $\bar{y}$ por grupos. Corresponde a tomar los valores de Figura 1.2, agrupar los 2 deciles más bajos y llamarlo *G1*, los 3 deciles siguientes *G2* y los 5 más altos *G3*.

Esta variable, si bien no aporta información nueva, corresponde a una de las métricas más importantes del problema, ya que la mira el negocio. Por lo tanto, un modelo óptimo será aquel que maximice la relación entre $\bar{y}_{G1}$ vs $\bar{y}_{G3}$ (llamado *G1-G3*) y de $\bar{y}_{G1}$ vs $\bar{y}_{G2}$ (llamado *G1-G2*). De esta manera, se asegura que el grupo *G1* represente al 20 % de la base con mayor propensión a la compra. El interés en esta métrica radica en que a los ejecutivos no se les provee la información de la probabilidad estimada de compra por cliente, sino el grupo.

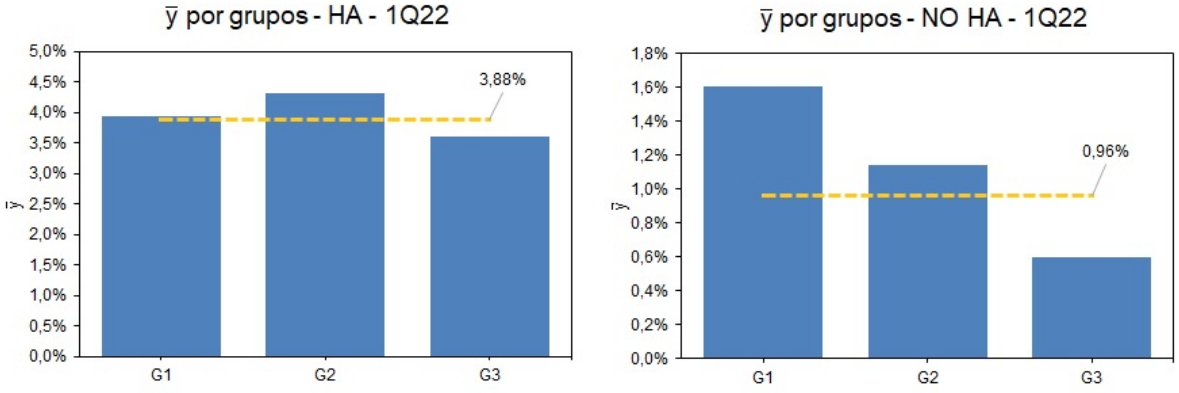

Figura 1.4: Ordenamiento por grupos de modelos actuales.

Capítulo 2

Armado y preprocesamiento de base

2.1. Adquisición de datos

El armado del data set tiene como fuente al data warehouse del banco. Aquí se dispone de todo tipo de información financiera y de la relación cliente-banco a través de distintos canales. Al tener información histórica por cliente, para algunas variables se toma el dato más reciente y para otras se realizan agrupados temporales, tomando como período los últimos 12 meses. De esta manera se generaron 1.417 variables.

2.2. Ventanas temporales

La campaña comercial correspondiente a la venta de tarjetas de crédito, tiene una duración de 4 meses. Terminado el período se vuelve a repetir todo el proceso global pudiendo hacerse alguna modificación, como por ejemplo una política crediticia. Un cambio en este punto, puede tener consecuencias directas en la base, ya que se modificaría el perfil del universo. Por este motivo, para no tener un data train (base de entrenamiento) sesgado a una sola campaña, y consecuentemente a un perfil temporal de cliente, se decide armar una base compuesta de 3 campañas, con un período anual de julio 2021 a julio 2022. De aquí se arma el data train y test. Por último se toma una campaña de julio 2022 a octubre 2022 (fuera de ventana), que se utilizará de validación para el modelo final.

2.3. Descripción de variables

Respecto a las 1.417 variables, representan información de tipo: demográfica, socioeconómica, saldos de deudas, de sistema financiero, consumos y formas de pago de tarjetas de crédito, consumos de tarjeta de débito, navegación web, tenencia de productos bancarios, uso de distintos canales del banco, ofertas de productos, ventas anteriores, transaccionalidad e información del cliente.

2.4. Muestreo y armado de bases

Como se comentó en la sección anterior, la base de entrenamiento y testeo parte de una ventana temporal anual tomando 3 campañas:

- Campaña A: julio 2021 a octubre 2021. Cantidad de registros: 624.000.
- Campaña B: noviembre 2021 a febrero 2022. Cantidad de registros: 852.000.
- Campaña C: marzo 2022 a junio 2022. Cantidad de registros: 904.000.

Para tener igual representación de cada campaña, se toma una muestra aleatoria de cada una de 600.000 registros. Luego se unifica obteniendo una base de 1.800.000 registros. Para generar train y test (base de entrenamiento y testeo), se divide en 70 % y 30 %.

Por otro lado, a la base de validación no se le realiza un muestreo, quedando con 780.000 registros.

2.5. Columnas sin información

A continuación, se muestra la limpieza de la base y tratamiento de variables. Como es común en este tipo de proyectos, el proceso es largo e iterativo. En cada paso se decidió explorar las variables con la idea de entender y adoptar los umbrales de corte óptimos.

En este primer paso se logró pasar de 1.417 columnas a 993. Al generar el data set, y debido a la gran cantidad de campos, muchos de ellos eran incorrectos. Por ende, luego de analizar uno a uno, se corrigieron formatos y se procedió a eliminar los restantes. Los motivos de eliminación son los siguientes: columnas con 100 % nulos, variables sin sentido de negocio (por ejemplo un campo que tenía como fuente una campaña obsoleta) y columnas duplicadas.

2.6. Registros duplicados

En particular, no se evidenciaron filas duplicadas pero sí clientes. Esto es normal, ya que un cliente puede aparecer en varias campañas, especialmente al haber diferencia temporal. Se decidió hacer un muestreo con el objetivo de no repetir el cliente y así conservar independencia. Esto se realizó de la siguiente manera. Recordando que la base train-test, se armó en base a 3 campañas A, B y C, se analizó si cada cliente aparecía en más de una. En caso afirmativo, se sampleó una de ellas aleatoriamente conservando un solo registro por cliente. La cantidad de filas disminuyó de 1.800.000 a 1.060.000.

2.7. Preprocesamiento de variables categóricas

2.7.1. Campos con una categoría

Se pasó de 993 a 980 columnas al eliminar variables con una sola categoría.

2.7.2. Campos con nulos

Debido al proceso en que se generaron las variables muchas de ellas podían tener registros nulos, entendiendo a esto como registros sin información (missing) para algún cliente. Al analizar su porcentaje, se detectó que 179 de 321 campos categóricos tenían al menos un registro nulo. Se realizaron gráficos bivariados entre cada variable categórica y la variable respuesta, con el objetivo de definir cuál es el umbral para eliminar campos con nulos. Con este criterio, se tomó un 80 %, pasando de 980 a 961 columnas.

Además, se recodificaron nulos como 0 en variables binarias, se mantuvieron 3 variables con nulos mayor a 80 % que presentan alta cantidad de tarjetas de crédito compradas en las pocas categorías con datos y se imputó como otro grupo a los registros nulos de las demás variables mantenidas.

2.7.3. Campos con alta asociación

Para detectar variables con alta asociación entre sí, se calculó el coeficiente de Cramer (Cramer's V) [4] entre campos categóricos. Al analizar los valores obtenidos, se deciden eliminar campos con un valor superior a 0,80. Se pasó de 961 a 873 variables.

2.8. Preprocesamiento de variables numéricas

2.8.1. Campos sin varianza

El primer paso fue calcular las variables numéricas con varianza cero y eliminarlas. Se obtienen 761 campos.

2.8.2. Campos con nulos

Se analizó el porcentaje de nulos (registros sin información), con una estrategia similar a la usada para variables categóricas. Se realizaron gráficos bivariados agrupando en deciles a las numéricas. Así se definió un umbral de 60 % para eliminar campos con nulos, pasando a un total 743. Los nulos restantes se imputan por mediana.

2.8.3. Campos con alta correlación

Se calcula el coeficente de correlación de Pearson entre variables numéricas. Luego de analizar los resultados se eliminan los campos con un valor mayor a 0,80. Se obtienen 565 variables.

2.8.4. Campos con outliers

El criterio tomado para definir un rango sin outlier es el siguiente:

$$(q1 - 3\,IQR; q3 + 3\,IQR)$$

- *q1*: primer cuantil de la variable.
- *q3*: tecer cuantil de la variable.
- IQR: rango intercuantil $(q3 - q1)$.

Se analizó en primer instancia las variables numéricas enteras. Allí se evidenció que solo 17 de 174 variables no tienen outliers. A continuación, se analizó variable a variable para detectar si había algún error en la carga del dato, corroborando que los valores tengan sentido. Luego se decidió poner un tope superior. Por otro lado, no fue necesario un tope inferior, ya que estas variables comenzaban en el valor 0 y presentaban asimetría a derecha.

Respecto a las numéricas decimales, solo 15 de 191 no poseen outliers. Al analizar la distribución de las mismas, se vió gran asimetría. Además, muchas de ellas representan valores monetarios nominales, altamente sensibles a la inflación. Por lo tanto, con motivo de corregir el problema de outliers e independizarse de montos nominales, se transforman todas estas variables a percentiles. Este tipo de transformaciones es usual en estos modelos. Se remarca que hay variables monetarias nominales en los modelos actuales, de manera que constituye una de las mejoras.

Finalmente, la base preprocesada tiene 1.060.000 filas y 565 columnas.

Capítulo 3

Selección de variables/reducción de dimensión

En este sección se analizan diversas herramientas y técnicas de selección de variables. Se destacan tres motivos para realizarlo, primero la alta cantidad de variables hacen que el data set sea muy pesado, pudiendo extender los tiempo de ajuste y búsqueda de hiperparámetros. Segundo, hay riesgo de exceder la memoria del servidor en algún etapa computacionalmente costosa, como estandarizaciones o ajustes de algoritmos. Por último, desde un punto de vista de modelado, no aporta valor incluir variables que no agreguen información a la hora de predecir.

3.1. Importancia de Random Forest

El principal método utilizado consiste en realizar ajustes mediante Random Forest [5] y obtener la importancia de las variables. Se remarca que este método se basa en la cantidad de veces que se selecciona a una variable como corte en un nodo, por ende, los campos categóricos no son incluidos como Dummies sino como ordinales, lo que no genera un problema ya que no se busca obtener una predicción en esta etapa. Además, tampoco es necesario realizar una búsqueda y optimización de hiperparámetros, se utilizan los valores:

- *n_estimators = 100*. Corresponde a la cantidad de árboles realizados.
- *max_depth = 5*. Corresponde a la capas o niveles del árbol.

Con intención de reducir la varianza del método, se repite el proceso 4 veces y se promedian los valores de importancia obtenidos.

En la Figura 3.1 se puede ver como se estabiliza la importancia en alrededor de 130 variables. Se toman las primeras 100, que corresponden a un 97 % de importancia acumulada. Finalmente, se corrobora que estas 100 variables tengan algún sentido de negocio para el problema.

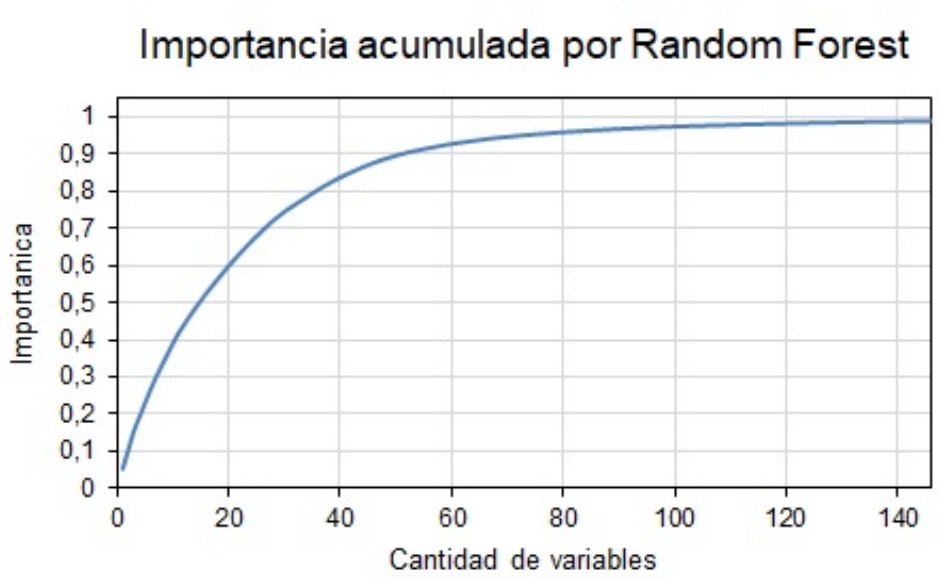

Figura 3.1: Importancia de variables de Random Forest.

3.2. Partial Least Squares-Discriminant Analysis (PLS-DA)

A continuación, se realiza un ajuste de PLS-DA [6]. Se obtiene la matriz de componentes (565 variables originales x 565 componentes de PLS) y se analiza el peso de cada celda.

El método se usa a modo confirmatorio del análisis realizado por Random Forest. Para ello se toman las variables del primer componente y se las ordena de manera decreciente por el valor absoluto del peso. Luego, se confirma que las variables descartadas por Random Forest no aparezcan entre las primeras 100 por PLS-DA. Esto se repite para las primeras 10 componentes. En la Tabla 3.1, se describe este procedimiento.

Tabla 3.1: Selección variables PLS-DA

Procedimiento selección variables PLS-DA
1. Tomar la columna del primer componente con las 565 variables originales.
2. Ordenar de manera decreciente por valor absoluto del peso.
3. Tomar las primeras 100 variables originales según el orden.
4. Comparar con las 465 variables descartadas por Random Forest.
5. Registrar diferencias.
6. Repetir en las primeras 10 componentes.

De esta manera, se confirmaron los resultados obtenidos por ambos métodos, encontrando coincidencia en las 100 variables seleccionadas.

3.3. Lasso

Como último método se realiza un ajuste de regresión logística con penalización L1 (Lasso) [7]. Aquí es necesario hacer la búsqueda de un hiperparámetro $tuning\_parameter = \lambda$, que influye directamente en el término de penalización, afectando a la cantidad de variables seleccionadas.

El procedimiento seguido consistió en separar la base conformada por las campañas A, B y C, 75 % en *train* y 25 % en *test*. Se armó una grilla de λ y se ajustó para *train*. Luego se realizó la predicción en *test*, calculando el área bajo la curva de Característica Operativa del Receptor (tasa de verdaderos positivos TPR y tasa de falsos positivos FPR): ROC AUC. Se eligió el punto en donde se estabiliza el valor de ROC AUC, ver Figura 3.2. Corresponde a un $\lambda = 200$, 39 variables y ROC AUC de 0,77.

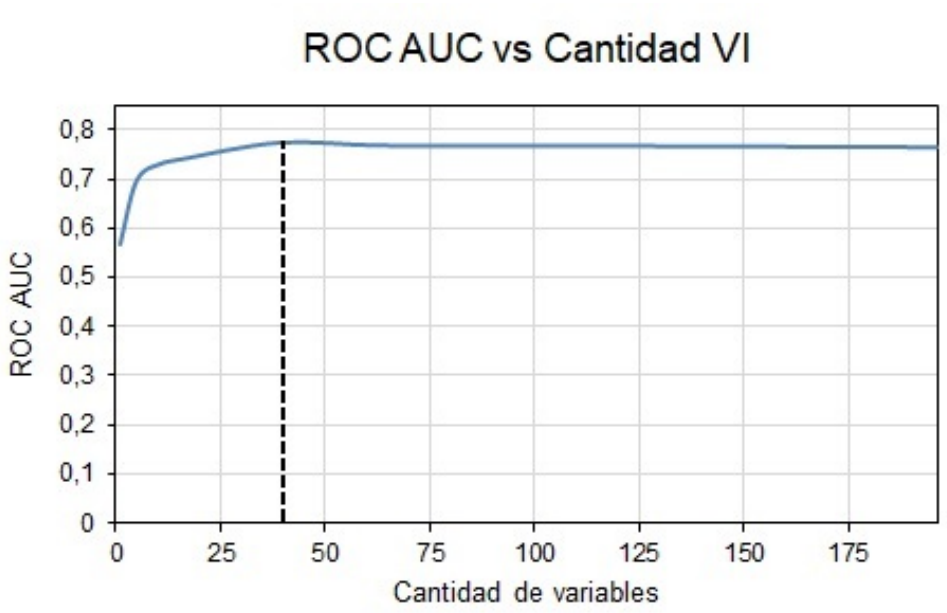

Figura 3.2: Lasso - ROC AUC en test contra cantidad de variables.

Se comparan estas 39 variables seleccionadas por Lasso con las 100 obtenidas en los pasos anteriores. Aquí se vió que 16 de las 39 obtenidas por Lasso, no fueron elegidas por Random Forest y confirmadas por PLS-DA. Finalmente, se decide incorporarlas, obteniendo 116 variables finales para modelar con las 1.060.000 filas.

Capítulo 4

Ajuste y evaluación a distintos universos

Como se ha mencionado en la introducción, actualmente se poseen dos modelos que surgen de separar al universo por una variable, si el cliente cobra sus haberes en el banco o no (*HA* y *NO HA*). La pregunta que interesa responder en esta sección es si actualmente vale la pena tener dos modelos. Es decir, si hay una mejora notable en la predicción que justifique realizar y mantener productivos ambos.

La manera en la que se analizará consiste en ajustar dos modelos con bases de entrenamiento distintas y evaluar su performance en mismos data test. Por un lado, un modelo entrenado con el universo *total* y por el otro, solo con los clientes *HA*. Se aclara que no se ajustó a la muestra *NO HA*, ya que ella corresponde a un 92 % del universo *total*. Con lo cuál el perfil realmente distinto al *total*, es la minoría *HA*.

Por último, se resalta que la base está muy desbalanceada: 1,85 % de la variable respuesta en el universo *total*.

4.1. Armado de bases para distintos universos

En la Figura 4.1, se muestra un esquema del universo para facilitar la explicación. El ajuste 1, corresponde a la base train *HA* y el ajuste 2, a tomar la base train *total* (*HA* y *NO HA*). Se generan muestras test con el mismo criterio, en dónde se predice y evalúan métricas: test *HA*, test *NO HA* y test *total* (*HA* y *NO HA*).

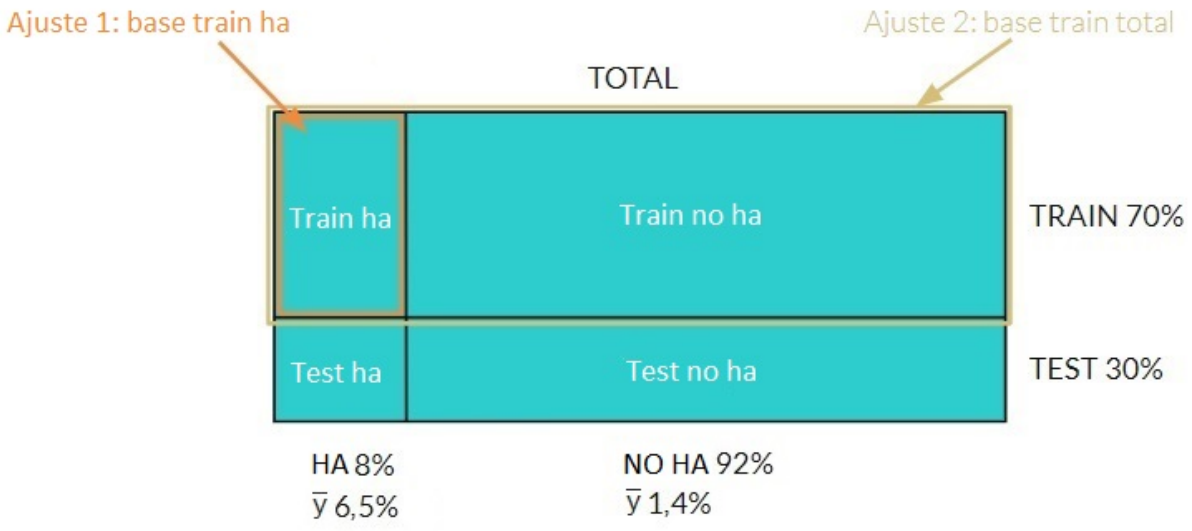

Figura 4.1: Armado de bases para ajuste y evaluación.

4.2. Ajustes de Modelos

El método principal utilizado es mediante regresión logística, probando con muchas variantes. Luego se ajustó con un Random Forest.

4.2.1. Regresión logística

Entre las diferentes alternativas que se probaron destacan balancear la base de entrenamiento en su respuesta con sub-muestreo, balancear la base de entrenamiento respecto a la variable *HA* con sub-muestreo, buscar umbrales óptimos para clasificar 0 y 1, realizar distintas interacciones entre variables y colocar menor cantidad de variables.

4.2.2. Random Forest

Luego de realizar todas las variantes con regresión logística, se ajustó un Random Forest optimizando los hiperparámetros por grillado y validación cruzada k fold ($k = 4$).

4.3. Evaluación

En el anexo, se muestra el detalle de las métricas obtenidas en todos los ajustes relizados: Tabla 6.1, Tabla 6.2, Tabla 6.3, Tabla 6.4 y Tabla 6.5.

La manera de interpretar las tablas es la siguiente:

- Base train: base usada para entrenar el modelo. El universo *total* ó *HA*.
- Base pred: base donde se realiza la predicción y calculan métricas.

- Balanceo target: la distribución de la variable respuesta en la base train. Hay variaciones al realizar distintos sub-muestreos.
- Umbral: threshold a partir del cual el se clasifica como 1 (comprador) a la probabilidad predicha.
- F1: métrica en base pred, $F1 = 2\frac{precision\ recall}{precision+recall}$.
- G1-G3: métrica de negocio en base pred. Representa la relación de la variable respuesta entre el grupo predicho más propenso a compra ($\bar{y}_{G1}$) y el menos propenso ($\bar{y}_{G3}$), $\bar{y}_{G1}/\bar{y}_{G3}$.
- AUC: métrica ROC AUC en base pred, área acumulada bajo la curva de tasa de verdaderos positivos *TPR* y tasa de falsos positivos *FPR*.

A continuación, se realiza un resumen de las métricas obtenidas para la mejor alternativa, regresión logística simple con todas las variables y balanceo del target en la base de entrenamiento al 29,5 % con umbral óptimo (punto de K_s máximo). Se van comparando las métricas de predicción en cada data test, para el modelo ajustado con la base train *total* contra el ajustado con la base train *HA*.

Tabla 4.1: Métricas de mejor regresión logística en test *total*.

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	29,5 %	0,31	61,7 %	6,38	73,2 %
train total	**test total**	**1,9 %**	**0,31**	**9,2 %**	**17,09**	**73,5 %**
train ha	train total	29,5 %	0,60	45,4 %	2,09	63,9 %
train ha	**test total**	**1,9 %**	**0,60**	**15,3 %**	**3,62**	**63,7 %**

- Predicción en test total - Tabla 4.1: *AUC* de 73, 5 % contra 63, 7 % y *G1-G3* de 17, 09 contra 3, 62. Se concluye que el modelo entrenado en train *HA* es un perfil particular que no puede generalizar a toda la base.

Tabla 4.2: Métricas de mejor regresión logística en test *HA*.

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train ha	61,0 %	0,31	79,4 %	1,93	60,9 %
train total	**test ha**	**6,5 %**	**0,31**	**15,2 %**	**6,77**	**60,8 %**
train ha	train ha	61,0 %	0,60	74,8 %	2,08	68,3 %
train ha	**test ha**	**6,5 %**	**0,60**	**20,6 %**	**6,89**	**67,9 %**

- Predicción en test *HA* - Tabla 4.2: *AUC* de 60, 8 % contra 67, 9 % y *G1-G3* de 6, 77 contra 6, 89. Se concluye que el modelo entrenado con train *total*, tiene resultados aceptables para el perfil *HA*. Si bien la métrica de predicción cae (7 puntos menos de *AUC*), al mirar la métrica de negocio (*G1-G3)*, la diferencia es mínima.

Tabla 4.3: Métricas de mejor regresión logística en test *NO HA*.

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train no ha	24,5 %	0,31	54,3 %	6,17	70,3 %
train total	**test no ha**	**1,4 %**	**0,31**	**7,5 %**	**13,70**	**70,8 %**
train ha	train no ha	24,5 %	0,60	27,3 %	2,97	57,2 %
train ha	**test no ha**	**1,4 %**	**0,60**	**10,6 %**	**4,42**	**57,0 %**

- Predicción en test *NO HA* - Tabla 4.3: *AUC* de $70,8\,\%$ contra $57,0\,\%$ y *G1-G3* de $13,70$ contra $4,42$. Se concluye que el modelo entrenado con train *HA* es un perfil particular que no generaliza bien a *NO HA*.

La conclusión es realizar un solo modelo *total*. Se mostró que la ganancia en *G1-G3* que se puede obtener al hacer un modelo particular para *HA* es mínima y por lo tanto no amerita tener dos modelos. Por otro lado, los resultados mostrados no están atados al algoritmo de regresión logística, ya que con Random Forest se obtiene un análisis similar (ver métricas en Tabla 6.6).

Capítulo 5

Ajuste y evaluación a universo final

5.1. Armado de bases

Tomada la decisión de hacer un modelo total que incluya ambos perfiles *HA* y *NO HA*, se realizan más cantidad de ajustes incorporando otros algoritmos. Además, se agrega la base de validación (fuera de ventana) correspondiente a otro período temporal. Esto permitirá tener un mejor acercamiento a la capacidad de generalización de los distintos modelos planteados. Respecto al data set de entrenamiento, se prueban dos bases, la original y un sub-muestreo rebalanceando la variable target al 30 %.

A continuación, un esquema que muestra dichas bases finales, todas con 116 variables.

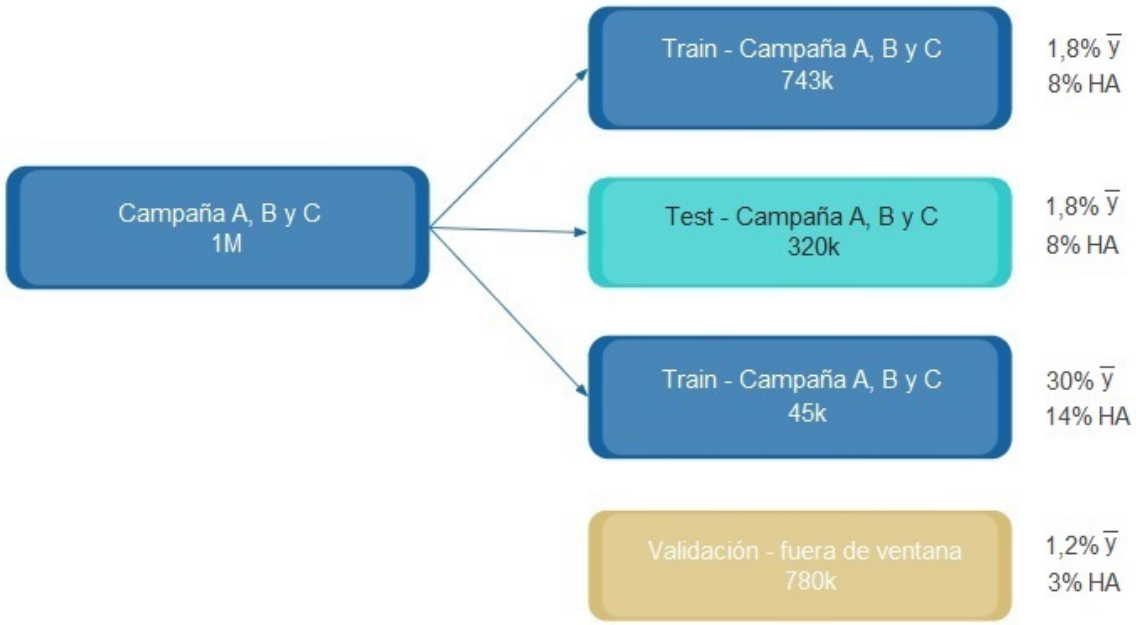

Figura 5.1: Esquema de bases finales para ajuste de modelo total.

5.2. Ajuste de Modelos y evaluación

En esta sección se muestran los distintos algoritmos evaluados y las métricas obtenidas en la bases train, test y validación (oow - out of window). Las tablas presentadas tienen los siguientes campos:

- Base pred: base donde se realiza la predicción y calculan métricas.
- Balanceo target: la distribución de la variable respuesta en cada base. El data train puede tener variaciones al realizar distintos sub-muestreos.
- Umbral: threshold a partir del cual el se clasifica como 1 (comprador) a la probabilidad predicha.
- F1: métrica en base pred, $F1 = 2\frac{precision\ recall}{precision+recall}$.
- AUC: métrica ROC AUC en base pred, área acumulada bajo la curva de tasa de verdaderos positivos *TPR* y tasa de falsos positivos *FPR*.
- Gini: métrica Gini en cada base calculada con la curva de Lorenz, relación entre área de curva de modelo y área de línea de igualdad vs área de modelo perfecto.
- Ks: métrica Ks (Kolmogorov Smirnov) en cada base, máxima distancia entre distribuciones acumuladas de $y = 1$ (compra) y $y = 0$ (no compra).
- G1-G3: métrica de negocio en base pred. Representa la relación de la variable respuesta entre el grupo predicho más propenso a compra ($\bar{y}_{G1}$) y el menos propenso ($\bar{y}_{G3}$), $\bar{y}_{G1}/\bar{y}_{G3}$.
- Ks estabilidad: métrica Ks (Kolmogorov Smirnov), máxima distancia entre la distribución de probabilidades obtenida para la base train y la base oow.

5.2.1. Regresión logística

Regresión logística simple, se analizó la significancia de los p-values de los coeficientes, manteniendo finalmente 42 variables. El método es uno de los peores en cuanto a capacidad de predicción, no se obtiene gran discriminación entre *G1-G3*, pero es de los más estables (Ks estabilidad bajo). Por otro lado, el tiempo de ajuste es muy rápido, alrededor de 5 minutos.

Tabla 5.1: Métricas regresión logística.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,40	58,50 %	79,40 %	40,61 %	43,14 %	5,37	-
test	1,87 %	0,40	10,01 %	78,90 %	55,34 %	42,69 %	12,03	10,50 %
oow	1,24 %	0,40	6,77 %	80,10 %	57,75 %	44,18 %	13,92	8,05 %

5.2.2. Regresión logística con penalización L2 (Ridge)

Regresión logística simple con penalización L2 (Ridge) [8], en este caso se debe seleccionar un hiperparámetro *tuning_parameter* $= \lambda$, que influye directamente en el término de penalización. Se realiza por grillado mediante validación cruzada k fold ($k = 4$), obteniendo $\lambda = 2,59$.

Las métricas de predicción mejoran respecto al caso anterior, *AUC* sube 2 puntos y *G1-G3* tiene un incremente significativo para oow. Por el lado negativo, empeora la estabilidad (Ks estabilidad alto). El tiempo de ajuste es rápido (10 minutos), ya que hay un solo hiperparámetro a evaluar.

Tabla 5.2: Métricas regresión logística con penalización L2.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,41	61,34 %	82,50 %	44,88 %	48,91 %	6,78	-
test	1,87 %	0,41	11,08 %	81,60 %	60,56 %	46,90 %	17,63	11,52 %
oow	1,24 %	0,41	9,81 %	82,40 %	62,08	46,36 %	20,51	20,44 %

5.2.3. Random Forest

Se ajusta un Random Forest optimizando los hiperparámetros *n_estimators* y *max_depth* por dos métodos distintos (ambos por validación cruzada k fold ($k = 4$)).

En primera instancia mediante un grillado, ver Tabla 5.3. Los resultados muestran un sobreajuste (*AUC* es $94,80\,\%$ en train y cae a $81,50\,\%$ en test), y una estabilidad en niveles similares al ajuste Ridge. Por otro lado, el tiempo de ajuste fue mucho mayor, alrededor de 3 horas.

Los hiperparámetros elegidos:

- *n_estimators = 1200*. Corresponde a la cantidad de de árboles realizados.
- *max_depth = 15*. Corresponde a la capas o niveles del árbol.

Tabla 5.3: Métricas Random Forest obtenidas por grillado.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,37	81,99 %	94,80 %	62,07 %	74,60 %	29,51	-
test	1,87 %	0,37	10,59 %	81,50 %	60,36 %	46,61 %	17,09	12,08 %
oow	1,24 %	0,37	11,03 %	82,00 %	61,20 %	45,88 %	18,51	21,74 %

El segundo método de optimización (Tabla 5.4) se realizó con la librería Optuna [9] disponible en Python. Esta, permite hacer de manera automatizada una búsqueda inteligente al minimizar una función de pérdida e ir iterando distintos valores de hiperparámetros. Así logra reducir los tiempos notablemente. Además, no se le pasa como input una grilla de valores sino su rango y tipo para cada hiperparámetro.

Al ajustar, se focalizó en reducir el sobreajuste observado en la Tabla 5.3. Se observan métricas de predicción buenas, sin overfitting y con un tiempo de ajuste muy rápido (10 minutos). Por otra parte, los resultados de estabilidad son malos. Los hiperparámetros seleccionados, *n_estimators = 224* y *max_depth = 10*.

Tabla 5.4: Métricas Random Forest obtenidas por Optuna.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,39	66,38 %	85,90 %	49,62 %	52,30 %	9,11	-
test	1,87 %	0,39	11,03 %	81,30 %	59,90 %	46,14 %	16,25	12,60 %
oow	1,24 %	0,39	13,54 %	82,00 %	61,27 %	45,55 %	18,92	29,71 %

5.2.4. Gradient Boosting Tree (GBT)

En el caso del algoritmo GBT [10] [11], se realizó la búsqueda de los hiperparámetros *n_estimators*, *max_depth* y *learning_rate* mediante un grillado con validación cruzada k fold ($k = 4$). Primero se hizo un grillado grueso y luego uno fino utilizando dos funciones de la librería sklearn de Python: RandomizedSearchCV y GridSearchCV. Sin embargo, los tiempos de ajuste fueron muy largos (alrededor de 12 horas).

Los hiperparámetros seleccionados valen:

- *n_estimators = 1560*. Corresponde a la cantidad de de árboles realizados.
- *max_depth = 5*. Corresponde a la capas o niveles del árbol.
- *learning_rate = 0,5*. Corresponde a la tasa de aprendizaje.

Respecto a los resultados, se observa un gran sobreajuste (AUC es $100,00\,\%$ en train y cae a $78,80\,\%$ en test). La estabilidad no es buena (Ks estabilidad alto). El overfitting es consecuencia de los hiperparámetros seleccionados, y por ende podría mejorarse. Pero dado que los tiempos de ajustes fueron muy largos, no se optó por tratar de mejorar el método.

Tabla 5.5: Métricas GBT.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,20	100,00 %	100,00 %	63,24	99,72 %	100,00	-
test	1,87 %	0,20	8,87 %	78,80 %	55,38 %	41,43 %	12,09	19,92 %
oow	1,24 %	0,20	7,86 %	78,30 %	54,67 %	40,72 %	11,43	26,14 %

5.2.5. Light Gradient Boosting Machine (light GBM)

El algoritmo lightGBM [12], se enmarca dentro de la familia de árboles generados por ensambles de pequeños aprendizajes similar a GBT, desarrollado por Microsoft en 2017 y con el foco puesto en reducir los tiempos de ajustes sin perder capacidad predictiva. Actualmente, el método es muy popular en la comunidad de Data Science y es ampliamente utilizado en competencias como las organizadas por Kaggle [13].

En particular, parte del algoritmo GBT y focaliza en que lo más pesado computacionalmente es el escaneo por cada variable de todos los posibles puntos de corte (*split points*). Por ende, previo a este paso, propone realizar dos técnicas novedosas que permiten reducir tiempos: *Gradient-based One-Side Sampling (GOSS)* y *Exclusive Feature Bundling (EFB)*.

La primera, *GOSS*, consiste en excluir una proporción de datos. La selección la realiza calculando los gradientes de todos los puntos y manteniendo aquellos de mayor valor. Además, en los datos con menores gradientes, se realiza un sampleo para elegir cuales mantener y cuales descartar.

La segunda técnica, *EFB*, busca hacer una reducción de dimensión para que el algoritmo tenga que procesar una cantidad de variables menor a la original. La idea radica en juntar variables que sean mutuamente excluyentes, ya que, según se argumenta en [12], raramente muchas variables toman valores diferentes a cero simultáneamente.

La búsqueda de hiperparámetros se realizó mediante la librería Optuna por validación cruzada k fold ($k = 4$), obteniendo tiempos de ajustes relativamente rápidos (30 minutos). A continuación, los hiperparámetros seleccionados y su explicación:

- *n_estimators = 10000*. Cantidad de árboles realizados.
- *learning_rate = 0,01*. Tasa de aprendizaje.
- *num_leaves = 800*. Cantidad de nodos del árbol.
- *max_depth = 5*. Cantidad de capas o niveles del árbol.
- *min_data_in_leaf = 1000*. Cantidad de datos mínimos a ser considerados para agregar un nuevo nodo.
- *lambda_l1 = 15*. Parámetro de regularización L1.
- *lambda_l2 = 10*. Parámetro de regularización L2.
- *min_gain_to_split = 2,9*. Valor mínimo de función de pérdida que debe obtenerse en un nodo para añadirlo.
- *bagging_fraction = 0,9*. Porcentaje de datos a samplear sin reemplazo al entrenar cada nodo.
- *bagging_freq = 1*. Frecuencia de realización de sampleo de datos. En este caso, se realiza cada 1 iteración.
- *feature_fraction = 0,3*. Porcentaje de variables a samplear al entrenar cada árbol.

En la Tabla 5.6 se muestran los resultados. Se ve que de todos los modelos presentados hasta ahora, este posee las mejores métricas de predicción ($AUC = 82{,}80\,\%$) y de negocio (*G1-G3 = 22,24*). Por otro lado, la estabilidad no es buena (Ks estabilidad alto). Dado

que fue el método con mayor capacidad predictiva, se decidió realizar diferentes pruebas con motivo de mejorar la estabilidad.

Tabla 5.6: Métricas lightGBM.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,41	62,36 %	83,20 %	45,83 %	48,87 %	7,11	-
test	1,87 %	0,41	11,37 %	82,10 %	61,37 %	47,31 %	19,28	12,52 %
oow	1,24 %	0,41	12,75 %	82,80 %	62,77 %	47,02 %	22,24	29,42 %

De esa manera, el mejor resultado obtenido se muestra en la Tabla 5.7. Corresponde al lightGBM pero ajustado con 22 variables. Las mismas corresponden a las de mayor $mean(SHAP\_Value)$ [14]. Consiste en promediar los SHAP Values entre todos los individuos por variable, lo que permite orientar con cual quedarnos, para luego ajustar y corroborar evaluando métricas. En la sección 5.3.2 se detalla más sobre el tema. Por otro lado, se utilizó la base de ajuste sin balancear (original).

Los resultados muestran que las métricas de predicción se mantienen (*AUC = 82,40 %* y *G1-G3 = 21,75*), mejorando la estabilidad. Este es el modelo elegido y sobre el que se hará un refinamiento final, mostrado en la siguiente sección 5.3. Los hiperparámetros seleccionados son los siguientes:

- *n_estimators = 186*
- *learning_rate = 0,45*
- *num_leaves = 1540*
- *max_depth = 6*
- *min_data_in_leaf = 667*
- *lambda_l1 = 0*
- *lambda_l2 = 40*
- *min_gain_to_split = 0,71*
- *bagging_fraction = 0,87*
- *bagging_freq = 1*
- *feature_fraction = 0,53*

Tabla 5.7: Métricas lightGBM con 22 variables.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	1,85 %	0,11	17,75 %	83,40 %	63,96 %	50,11 %	22,80	-
test	1,87 %	0,11	16,04 %	81,30 %	59,97 %	45,70 %	17,41	0,09 %
oow	1,24 %	0,11	2,21 %	82,40 %	62,20 %	47,24 %	21,75	20,71 %

5.2.6. PLS-DA y Random Forest

El método consistió en ajustar en primera instancia PLS-DA y utilizar las primeras k componentes de su matriz de loadings como input para un Random Forest.

Esto se hizo de la siguiente manera, se fue iterando ajustando PLS-DA para distintos valores de k y luego ajustando un Random Forest seleccionando los hiperparámetros por grillado con validación cruzada, para cada matriz de loadings. Así, se comparó el mejor valor obtenido de *ROC AUC* para cada k. Los resultados se muestran en la Figura 5.2, se eligió $k = 23$. Los hiperparámetros de Random Forests para este punto son *n_estimators* $= 500$ y *max_depth* $= 5$.

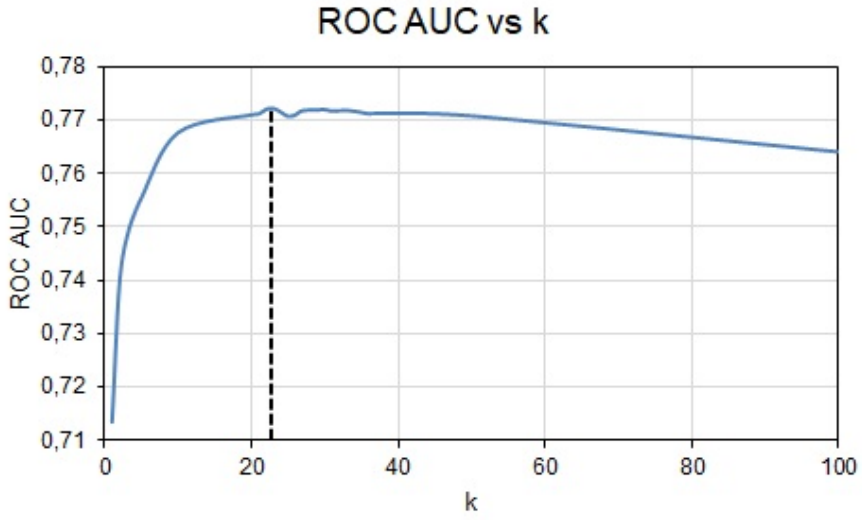

Figura 5.2: ROC AUC contra cantidad de componentes de matriz de loadings PLS-DA.

En la Tabla 5.8, se muestran los resultados de las evaluaciones. Si bien el proceso fue rápido (30 minutos), las métricas de predicción, negocio y estabilidad no son buenas. Además, se pierde mucha interpretabilidad al ingresar variables ficticias (loadings) al Random Forest. Para el banco, este aspecto es muy importante, y la opción elegita (lightGBM) permite interpretar las variables de una manera aceptable, similar a cualquier otro método de ensamble de árboles.

Tabla 5.8: Métricas PLS con Random Forest.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	30,08 %	0,36	48,08 %	78,90 %	39,76 %	42,76 %	5,11	-
test	1,87 %	0,36	9,93 %	77,60 %	52,77 %	40,56 %	9,92	11,38 %
oow	1,24 %	0,36	10,53 %	77,90 %	53,40 %	40,41 %	9,63	29,06 %

5.3. Refinamiento de modelo elegido

5.3.1. Ajustes finales

Luego de realizar la comparación de los modelos y elegir al algoritmo lightGBM con 22 variables, se procedió a hacer un análisis más fino, mirando en detalle cada una de ellas. Se vieron los siguientes puntos:

- Se controló la fuente de las variables, asegurando que sigan estando disponibles a futuro. Al ser un modelo productivo con corridas periódicas esto es fundamental.
- Se realizaron gráficos bivariados entre cada variable y la respuesta, y se calcularon SHAP Values. Ver sección 5.3.2.
- Se agruparon categorías en algunas variables categóricas.
- Se eliminaron variables que no tenían sentido de negocio.
- Se eliminaron variables que no se podía explicar su comportamiento dentro del modelo.

Por ende, el modelo final paso de 22 a 15 variables independientes, de las cuales se tiene garantía que estarán disponibles a futuro y permiten una explicación a nivel negocio-modelo. Además se mejora notablemente la estabilidad. Los hiperparámetros seleccionados valen:

- *n_estimators = 81*
- *learning_rate = 0,21*
- *num_leaves = 140*
- *max_depth = 9*
- *min_data_in_leaf = 658*
- *lambda_l1 = 5*
- *lambda_l2 = 30*
- *min_gain_to_split = 8,27*
- *bagging_fraction = 0,56*
- *bagging_freq = 1*
- *feature_fraction = 0,58*

A continuación, se muestran las métricas en detalle, Tabla 5.9. Se ve que los valores de predicción (*AUC = 81,40 %*) y negocio (*G1-G3 = 17,49*) son buenos. Mejora la estabilidad respecto al modelo con 22 variables (Ks estabilidad disminuye 6 puntos en oow).

Tabla 5.9: Métricas finales lightGBM con 15 variables.

Base pred	**Balanceo target**	**Umbral**	**F1**	**AUC**	**Gini**	**Ks**	**G1-G3**	**Ks Estabilidad**
train	1,85 %	0,09	15,78 %	81,30 %	60,05 %	46,39 %	16,65	-
test	1,87 %	0,09	14,37 %	79,10 %	55,84 %	41,52 %	12,36	0,17 %
oow	1,24 %	0,09	12,50 %	81,40 %	60,27 %	45,48 %	17,49	14,56 %

La Figura 5.3 muestra un ordenamiento por grupos muy bueno. Gran resultado en la métrica de negocio.

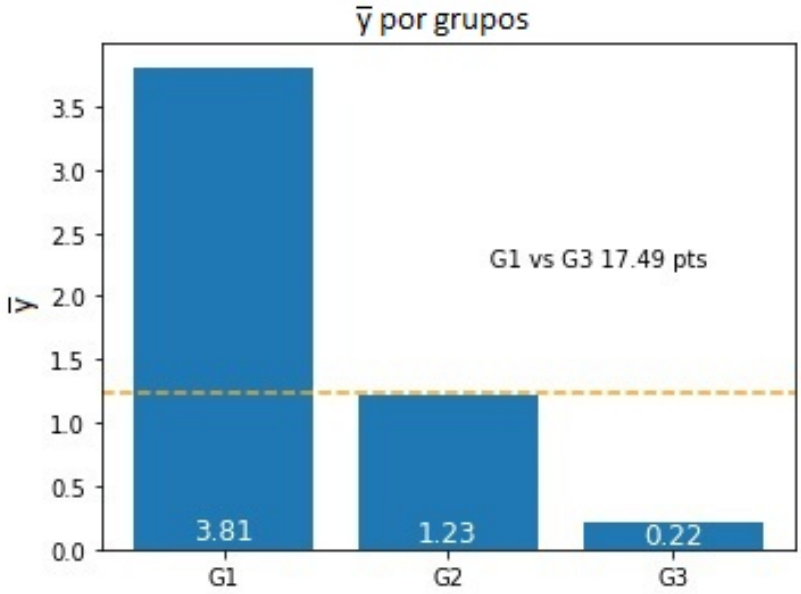

Figura 5.3: Target promedio por grupos.

Se ve un buen ordenamiento por deciles.

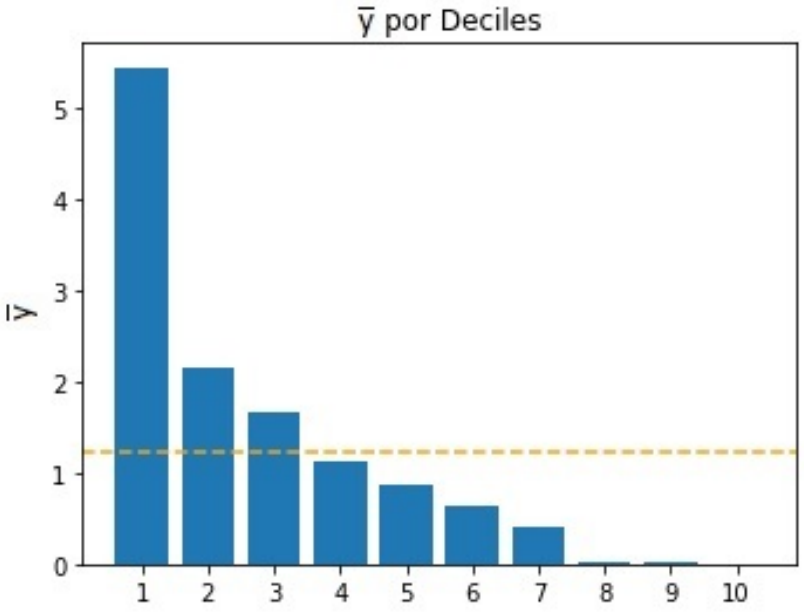

Figura 5.4: Target promedio por deciles.

Se ve un buen ordenamiento por ventiles.

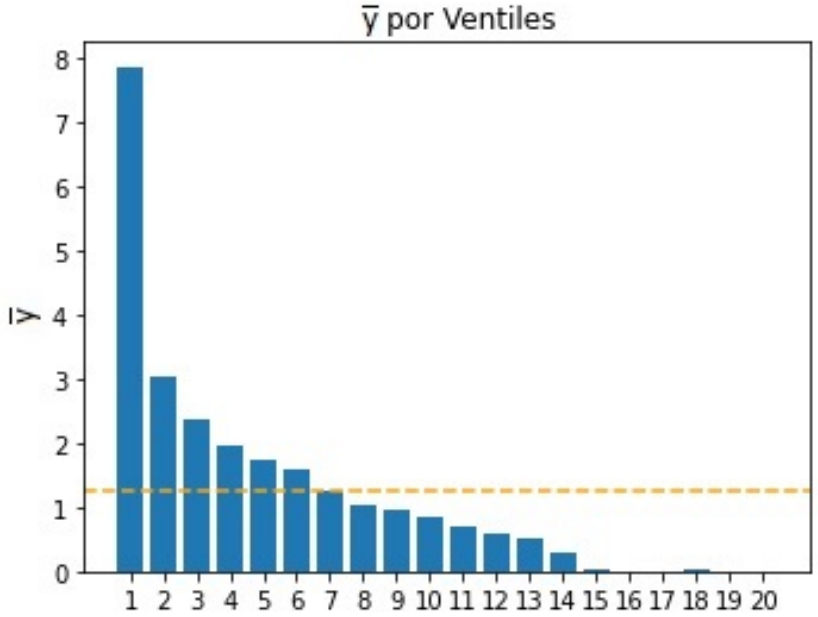

Figura 5.5: Target promedio por ventiles.

Se compara la predicción del target contra el valor real. Los puntos están cercanos a la diagonal.

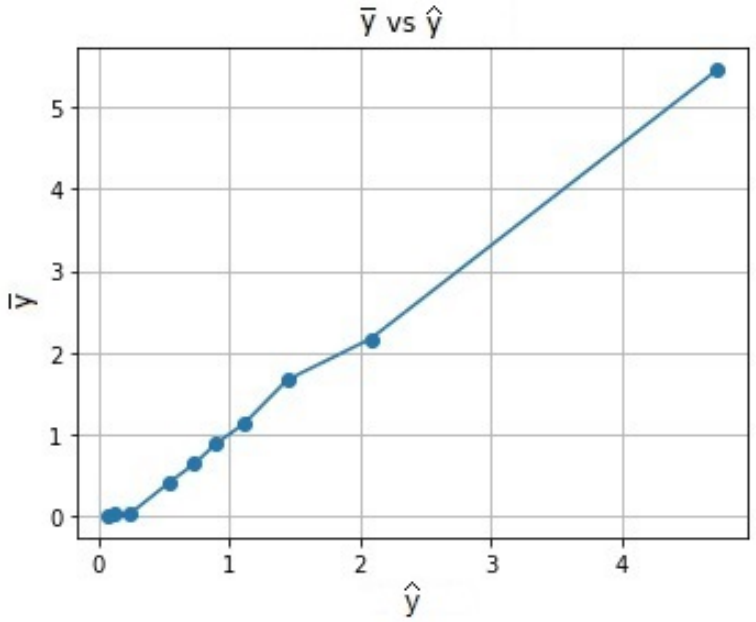

Figura 5.6: Target promedio contra target promedio predicho por deciles.

Se muestran las curvas ROC y Gini.

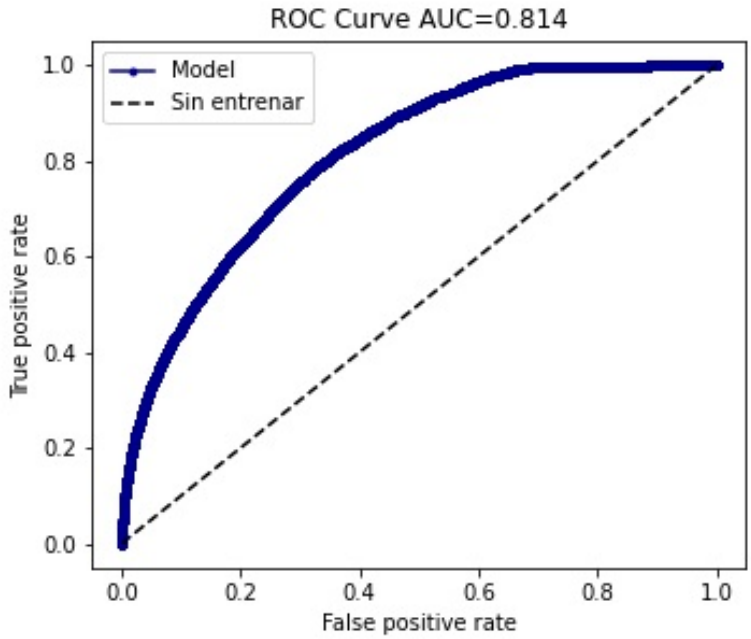

Figura 5.7: Curva ROC AUC.

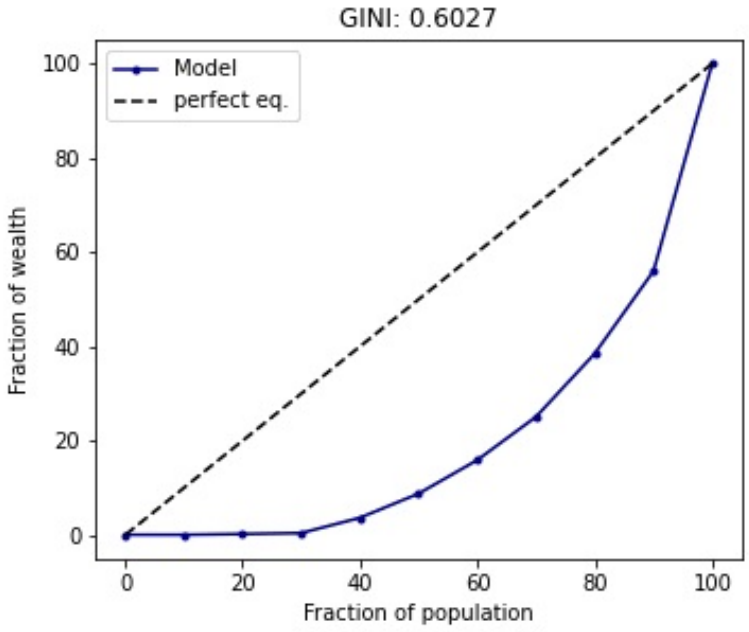

Figura 5.8: Curva Gini.

La Figura 5.9 muestra la curva Gain. Corresponde al porcentaje de $y = 1$ acumulado por decil.

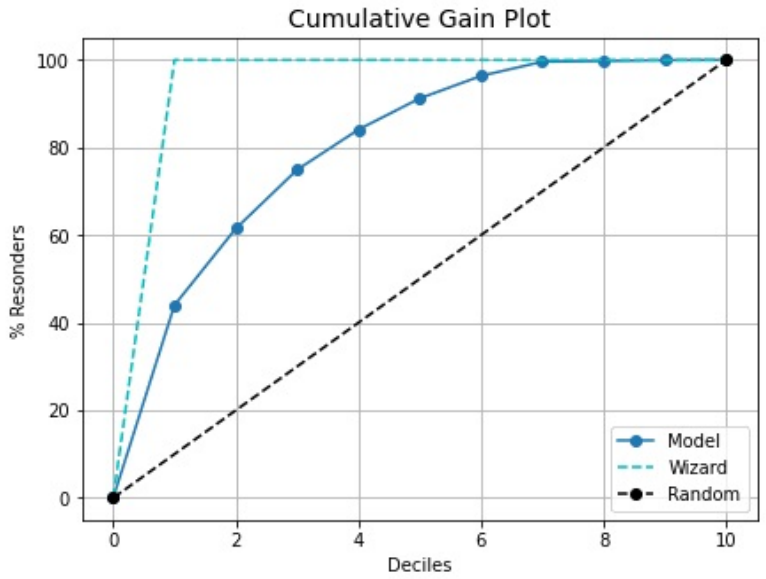

Figura 5.9: Curva Gain.

La Figura 5.10 muestra la curva Lift. Corresponde la proporción entre Gain ($y = 1$ acumulado por decil) comparado con un modelo aleatorio.

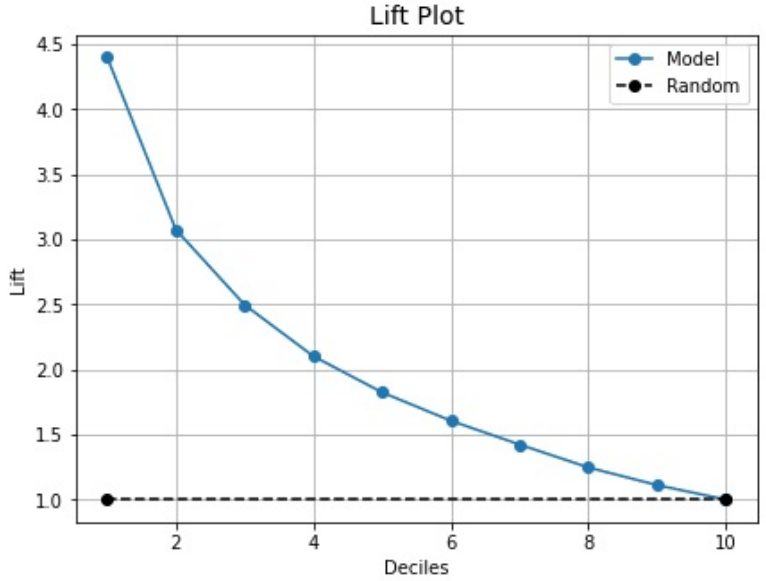

Figura 5.10: Curva Lift.

El gráfico Ks por decil, indicando su valor en donde es máximo.

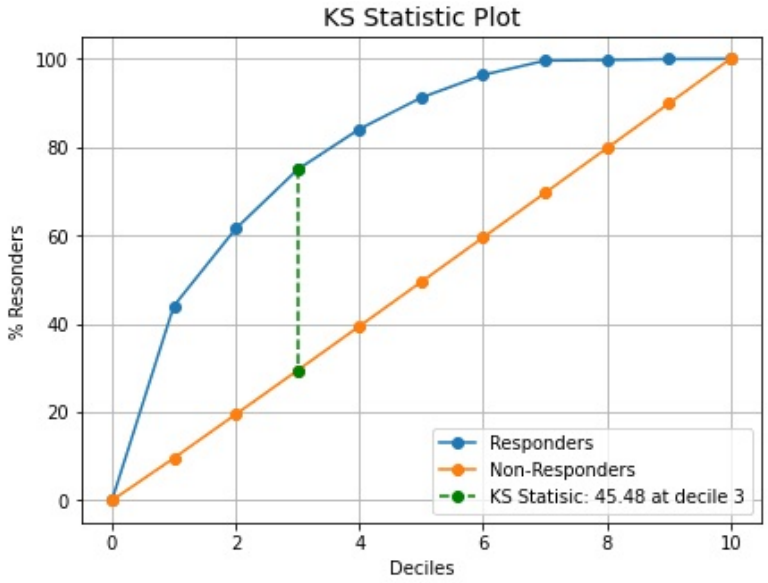

Figura 5.11: Curva KS.

A continuación se muestra la matriz de confusión y métricas de clasificación. De todas maneras, estas métricas no son de gran interés para el problema, ya que no se busca clasificar como compra o no compra, sino obtener la probabilidad con objetivo de dar un orden de prioridad.

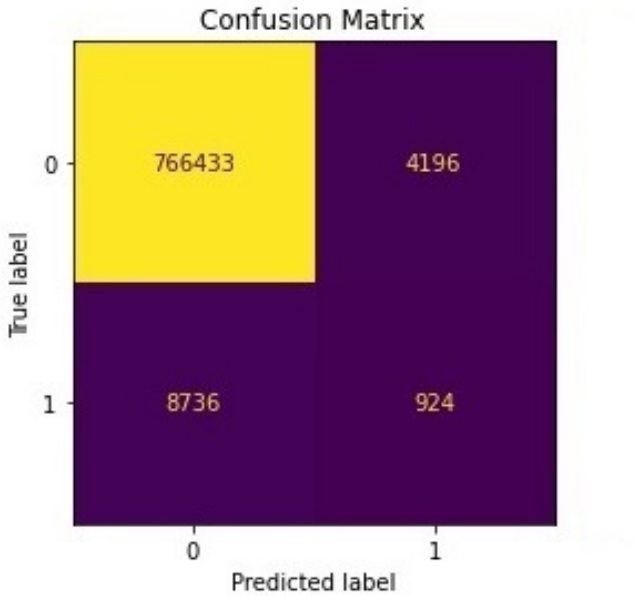

Figura 5.12: Matriz de confusión.

Classification Report

	precision	recall	f1-score	support
0	0.99	0.99	0.99	770629
1	0.18	0.10	0.13	9660
accuracy			0.98	780289
macro avg	0.58	0.55	0.56	780289
weighted avg	0.98	0.98	0.98	780289

Figura 5.13: Métricas de clasificación

Respecto al umbral de corte, se buscó el punto óptimo entre recall y precision.

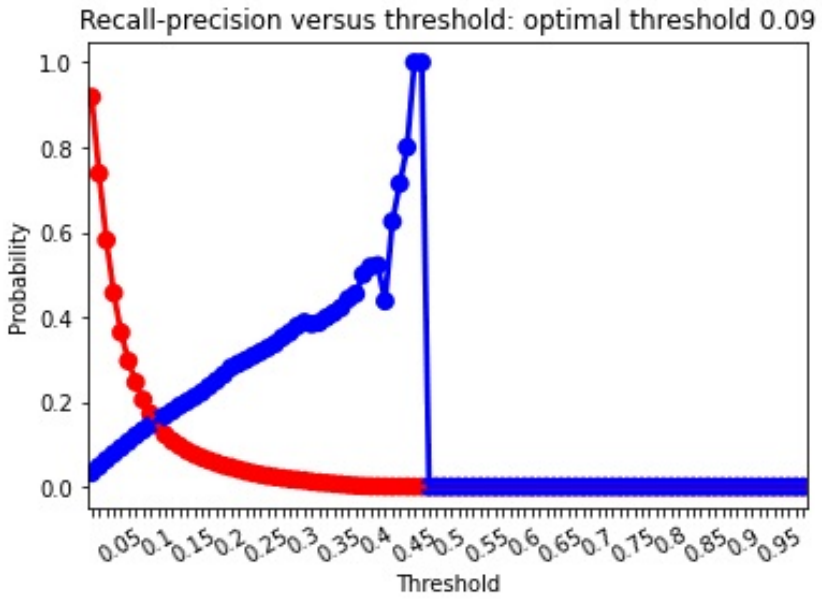

Figura 5.14: Umbral óptimo.

En los próximos dos gráficos se muestra el Ks de la estabilidad, comparando las bases train y test (Figura 5.15), y train y validación (Figura 5.16).

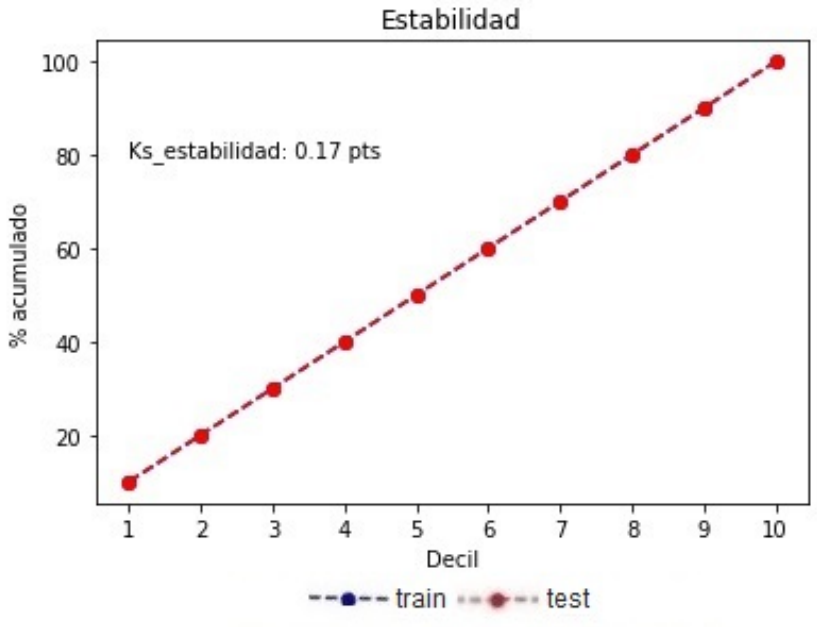

Figura 5.15: Ks estabilidad entre base train y base test.

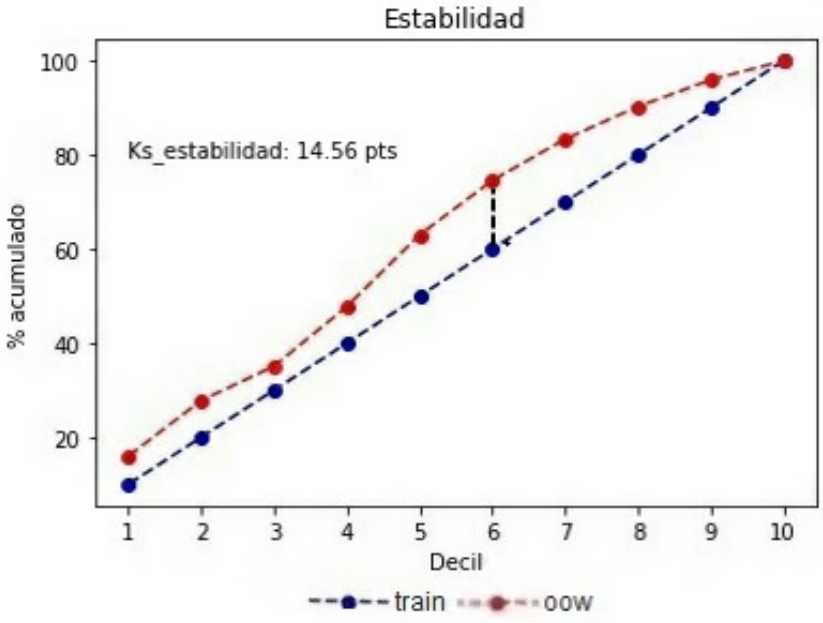

Figura 5.16: Ks estabilidad entre base train y base validación (oow).

5.3.2. Explicación de variables

En esta sección se explican las variables incluidas en el modelo final, en total 15. En la Tabla 5.10 se las describe, junto a su tipo.

Tabla 5.10: Descripción de variables.

Variable	**Tipo**	**Descripción**
telefono	Binaria	Si tiene un teléfono móvil validado
boni	Categórica	Bonificación en la oferta de producto
antiguedad	Numérica entera	Meses de antigüedad como cliente
cajero	Numérica entera	Indicador de actividad en cajeros automáticos
digital	Binaria	Si el cliente usa los canales de contactos digitales del banco
tipo_cliente	Categórica	Categoría según perfil de cliente
edad	Numérica entera	Edad en años
score_cred	Numérica continua	Score crediticio de bureau
ca	Binaria	Indicador de tenencia caja de ahorro
oferta_tc	Categórica	Categoría de nueva oferta de tarjeta de crédito y relación con posesión actual
prefijo_tel	Categórica	Agrupado de prefijos de número telefónico
log_online	Numérica entera	Cantidad de logeos en home banking
ha	Binaria	Si el cliente cobra sus haberes en el banco
tipo_pago	Categórica	Tipo de pago de tarjetas de crédito activas
score_default	Numérica continua	Score crediticio respecto a esperanza de pago

Mediante los SHAP (SHapley Additive exPlanation) Values [14] se explica el comportamiento de las variables en el modelo. Esta técnica fue presentada en el año 2017 como una propuesta unificada para la interpretación de predicciones de modelos y corresponde a la contribución marginal de cada variable en la probabilidad final, para cada individuo.

La propuesta consiste en definir un modelo *explanation model* $g(x')$ más simple que el original $f(x)$, donde se mapean los *inputs* $x \to x'$ en *simplified inputs* y se cumple $f(x) \approx g(x')$. El modelo simplificado es:

$$g(x') = \phi_0 + \sum_i \phi_i x'_i$$

Donde:

- ϕ_0 es la respuesta del modelo sin variables.
- ϕ_i es la contribución a la respuesta de cada variable.

Para que el modelo simplificado cumpla ciertas propiedades deseables (ver en [14]), los ϕ_i deben ser los Shapley Values [15]. Estos surgen en Teoría de Juegos y su cálculo consiste en tomar al modelo elegido, y hacer 2^p ajustes (p cantidad total de variables)

con todos los posibles conjuntos de las variables. Luego, por individuo, se realizan las 2^p predicciones y por variable se mira la contribución marginal, es decir, cuánto cambia la predicción al agregar la variable a un conjunto dónde no estaba presente. Finalmente, se calcula el SHAP Value, como una media ponderada de todas contribuciones de la variable. Se remarca que el SHAP Value se calcula a nivel individuo, posibilitando una intrerpretabilidad local. Por otro lado, es válido realizar agregados de los individuos, lo que permite también una interpretabilidad global.

Por último, el principal inconveniente del método es el alto costo computacional, debido a tener que entrenar 2^p modelos. Por eso, los autores propusieron diferentes *Explainer* (*KernelExplainer*, *LinearExplainer*, entre otros). Algunos de ellos, como *TreeExplainer* específico para árboles [16], fueron publicados años más tarde. Básicamente, se enfocan en reducir los tiempos de computo a través de estimar los SHAP Values, muestrear los conjuntos de manera eficiente y operar en otro espacio y transformar.

Respecto al modelo de la tesis, se muestran dos gráficos. La Figura 5.17 corresponde al promedio de los SHAP Value calculado para cada individuo, por variable. La primer información que se obtiene, es que hay dos variables *telefono* y *boni* que son las más importantes. Hay un salto respecto a la variable *antiguedad* y luego se va decreciendo casi linealmente en importancia.

Por otro lado, se puede ver el comportamiento de algunas variables:

- *telefono*, *cajero*, *digital*, *log_online* y *ha* tienen un impacto positivo. A mayor valor mayor probabilidad, lo cual tiene sentido.
- Respecto a *boni*, si bien indica un signo negativo, esto puede confundir al ser una variable categórica. Se ve el detalle en la Figura 5.18.
- La variable *edad*, tiene impacto negativo (a mayor edad menor probabilidad de compra), tiene sentido de negocio. Lo mismo pasa con *score_cred*, una variable muy usada en el rubro, y que suele indicar menor probabilidad a mayor score. Esto está relacionado con un perfil de cliente con poca mora, muy conservativo y poco propenso a adquirir productos crediticios indiscriminadamente. Por ende, más difícil venderle una tarjeta de crédito.

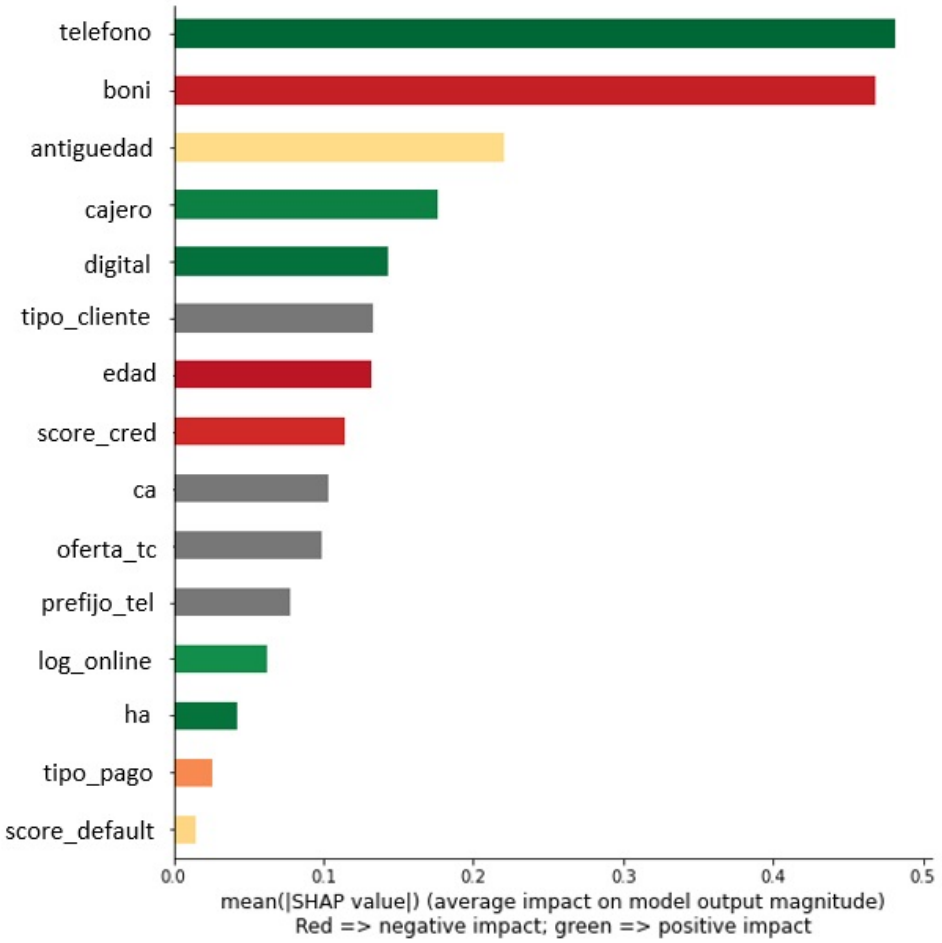

Figura 5.17: Promedio de SHAP Value por variable.

La Figura 5.18 permite hacer un análisis más profundo por variable, entendiendo el comportamiento de las mismas en el modelo.

- *telefono*, poseer dato de teléfono corresponde a mayor probabilidad. Sin el dato, no se puede contactar al cliente.
- *boni*, el modelo discrimina según la categoría de la variable. Con bonificación aumenta la probabilidad.
- *antiguedad*, la tendencia general de negocio es que a menor antigüedad como cliente, mayor perfil comprador.
- *cajero*, mayor uso de cajero, mayor probabilidad.
- *digital*, un cliente digital es más propenso a comprar productos.
- *tipo_cliente*, variable categórica que separa por perfil.
- *edad*, un cliente más joven tiende a ser más comprador.
- *score_cred*, esta es una variable con una tendencia conocida. A menor score, perfil más arriesgado y más comprador.
- *ca*, poseer caja de ahorro corresponde a mayor probabilidad.

- *oferta_tc*, esta es una variable categórica que separa por el tipo de oferta.
- *prefijo_tel*, puede ser pensada como una variable de ubicación. Es categórica.
- *log_online*, a mayor actividad en Home Banking mayor probabilidad de compra.
- *ha*, cobrar los haberes en el banco incrementa la probabilidad.
- *tipo_pago*, es una variable categórica que representa distintos perfiles respecto a afrontar sus pagos de deudas.
- *score_default*, otra variable conocida. Si bien tiene poco peso, la tendencia es que un cliente que no paga sus deudas es más comprador.

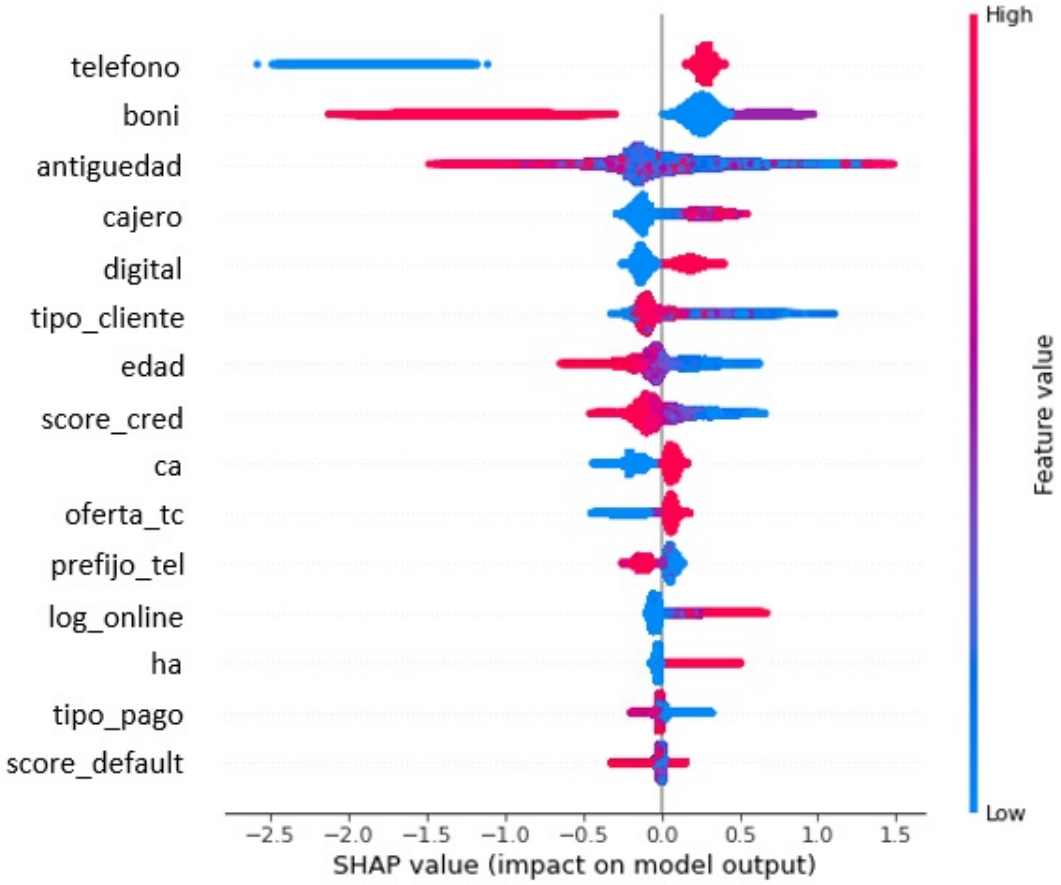

Figura 5.18: SHAP Value por variable.

5.3.3. Comparación con modelo actual

En esta sección se muestran las métricas del modelo actual comparado con la nueva propuesta desarrollada. Como se comentó en capítulos previos, se disponen de dos modelos (*HA* y *NO HA*). Luego de hacer la predicción en cada uno de estos, se unifica en un solo vector de probabilidades estimadas y con esta base conformada se definen los grupos *G1*, *G2* y *G3* que luego van a gestión. Esto es así ya que comercialmente es un solo producto, tarjetas de crédito, que debe vender el ejecutivo.

Entonces, la comparación se realiza con esta base de gestión unificada (actual) contra la del nuevo modelo, para la ventana de validación (oow). En la Tabla 5.11, se muestra que el nuevo modelo tiene mayor capacidad predictiva (diferencia en Ks de *15,6*) y una mejora notable de la métrica de negocio *G1-G3* (*3,56* veces).

Tabla 5.11: Métricas modelo actual contra nuevo.

Modelo	**Base pred**	**Ks**	**G1-G3**
Actual	oow	29,80 %	4,91
Nuevo	oow	45,48 %	17,49

En la siguiente figura, se compara el porcentaje de target acumulado por decil en ambos modelos. Para el decil 3, el nuevo modelo acumula un 75 % del target contra un 59 % del modelo actual.

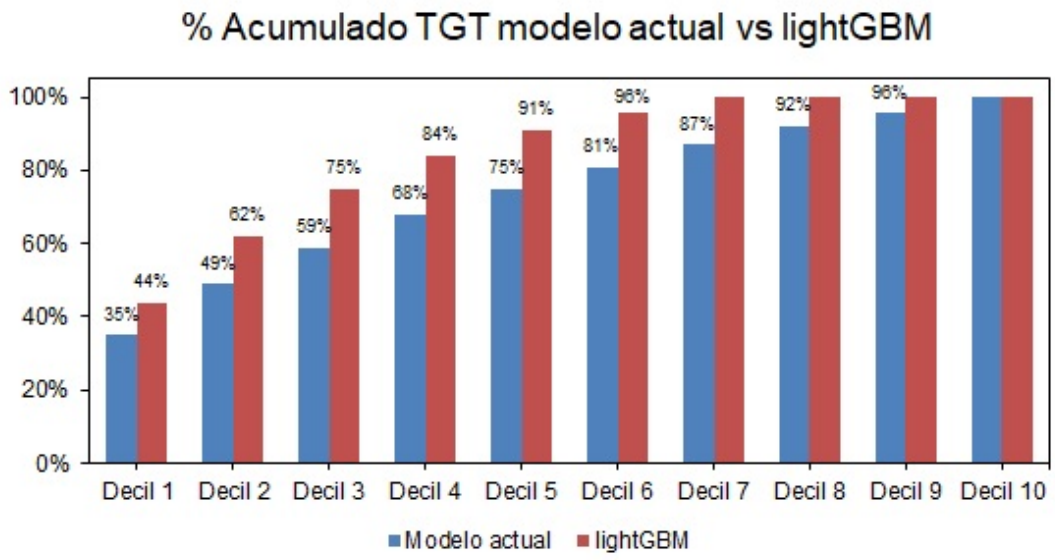

Figura 5.19: Acumulación del target en modelo actual contra nuevo.

Capítulo 6

Conclusiones

En base a la realización de la presente tesis, se pueden concluir los siguientes puntos:

- Se desarrolló un nuevo modelo que es ampliamente mejor el actual, tanto en términos de predicción como de negocio.
- Se logró reducir de dos a un modelo, mostrando que esto no tiene consecuencias negativas en la capacidad de predicción y reduciendo a futuro los costos de mantener productivo un modelo.
- Se aplicaron técnicas de muestreo para armar la base de entrenamiento, lo que permitió generar un data set que conserve independencia entre sus observaciones.
- Se partieron de 1.417 variables. Luego se utilizaron tres métodos distintos de selección permitiendo pasar de 565 a 116. Finalmente se lograron encontrar las 15 más predictivas para el problema.
- Se incorporaron nuevas variables y se transformaron a percentiles las variables numéricas, para no sufrir desactualización de montos monetarios a futuro.
- Se ajustaron 6 distintos tipos de algoritmos al universo final, permitiendo encontrar el mejor modelo.
- Se utilizaron librerías modernas que permiten reducir los tiempos de ajuste y búsqueda de hiperparámetros.
- Se analizaron las variables finales, pudiendo comprender el comportamiento en el modelo y entendiendo su sentido en base al conocimiento del negocio.

Referencias

[1] A. J. DOBSON, *An introduction to generalized linear models*. Chapman & Hall, 2001.

[2] A. AGRESTI, *Categorical Data Analysis*. Wiley, 1990.

[3] T. Chen y C. Guestrin, «Xgboost: A scalable tree boosting system,» *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, págs. 785-794, 2016.

[4] H. Cramer, *Mathematical Methods of Statistics*. Princeton: Princeton University Press, 1946, cap. Chapter 21. The two-dimensional case, pág. 282.

[5] L. Breiman, «Random Forests,» *Machine Learning 45*, págs. 5-32, 2001.

[6] R. Rosipal y N. Krämer, «Overview and Recent Advances in Partial Least Squares,» *Lecture Notes in Computer Science, vol 3940*, págs. 5-32, 2016.

[7] R. Tibshirani, «Regression shrinkage and selection via the lasso,» *J R Stat Soc Series B*, págs. 267-288, 1996.

[8] A. Hoerl y R. Kennard, «Ridge regression: Biased estimation for nonorthogonal problems,» *Technometrics. 8*, págs. 27-51, 1970.

[9] T. Akiba, S. Sano, T. Yanase, T. Ohta y M. Koyama, «Optuna: A Next-generation Hyperparameter Optimization Framework,» en *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[10] J. H. Friedman, «Greedy function approximation: a gradient boosting machine,» *Annals of statistics*, págs. 1189-1232, 2001.

[11] T. Hastie, R. Tibshirani y J. Friedman, *The Elements of Statistical Learning*, second edition. 2009, págs. 337-384.

[12] G. Ke, Q. Meng, T. Finley et al., «LightGBM: A Highly Efficient Gradient Boosting Decision Tree,» *31st Conference on Neural Information Processing Systems, Long Beach, CA, USA.*, 2017.

[13] Dirección: `https://www.kaggle.com/competitions`.

[14] S. M. Lundberg y S.-I. Lee, «A Unified Approach to Interpreting Model Predictions,» en *Advances in Neural Information Processing Systems 30*, Curran Associates, Inc., 2017, págs. 4765-4774.

[15] L. S. Shapley, «A value for n-person games,» *Contributions to the Theory of Games*, vol. 2.28, págs. 307-317, 1953.

[16] S. M. Lundberg, G. Erion, H. Chen et al., «From local explanations to global understanding with explainable AI for trees,» *Nature Machine Intelligence*, vol. 2, n.º 1, págs. 2522-5839, 2020.

Anexo

Tabla 6.1: Métricas regresión logística simple - parte 1

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	1,9 %	0,50	0,0 %	1,03	50,0 %
train total	test total	1,9 %	0,50	0,0 %	1,02	50,0 %
train total	train ps	6,5 %	0,50	0,2 %	1,00	50,0 %
train total	test ps	6,5 %	0,50	0,0 %	0,99	50,0 %
train total	train no ps	1,4 %	0,50	0,0 %	1,01	50,0 %
train total	test no ps	1,5 %	0,50	0,0 %	1,00	50,0 %
train ps	train total	1,9 %	0,50	10,9 %	1,02	52,1 %
train ps	test total	1,9 %	0,50	6,1 %	0,99	52,1 %
train ps	train ps	6,5 %	0,50	0,3 %	1,01	50,0 %
train ps	test ps	6,5 %	0,50	0,1 %	1,02	50,0 %
train ps	train no ps	1,4 %	0,50	7,3 %	1,00	53,2 %
train ps	test no ps	1,5 %	0,50	7,2 %	1,00	53,1 %
train total	train total	29,5 %	0,31	61,7 %	6,38	73,2 %
train total	test total	1,9 %	0,31	9,2 %	17,09	73,5 %
train total	train ps	61,0 %	0,31	79,4 %	1,93	60,9 %
train total	test ps	6,5 %	0,31	15,2 %	6,77	60,8 %
train total	train no ps	24,5 %	0,31	54,3 %	6,17	70,3 %
train total	test no ps	1,4 %	0,31	7,5 %	13,70	70,8 %
train ps	train total	29,5 %	0,31	53,5 %	2,10	64,8 %
train ps	test total	1,9 %	0,31	5,4 %	3,63	64,6 %
train ps	train ps	61,0 %	0,31	80,0 %	2,08	62,7 %
train ps	test ps	6,5 %	0,31	15,6 %	6,83	61,8 %
train ps	train no ps	24,5 %	0,31	46,3 %	2,97	63,1 %
train ps	test no ps	1,4 %	0,31	4,1 %	4,44	62,9 %

Tabla 6.2: Métricas regresión logística simple - parte 2

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	29,5 %	0,60	51,4 %	6,38	63,1 %
train total	test total	1,9 %	0,60	5,4 %	17,09	62,8 %
train total	train ps	61,0 %	0,60	75,9 %	1,93	69,6 %
train total	test ps	6,5 %	0,60	20,5 %	6,77	67,8 %
train total	train no ps	24,5 %	0,60	45,5 %	6,17	62,4 %
train total	test no ps	1,4 %	0,60	4,2 %	13,70	62,4 %
train ps	train total	29,5 %	0,60	45,4 %	2,09	63,9 %
train ps	test total	1,9 %	0,60	15,3 %	3,62	63,7 %
train ps	train ps	61,0 %	0,60	74,8 %	2,08	68,3 %
train ps	test ps	6,5 %	0,60	20,6 %	6,89	67,9 %
train ps	train no ps	24,5 %	0,60	27,3 %	2,97	57,2 %
train ps	test no ps	1,4 %	0,60	10,6 %	4,42	57,0 %
train total	train total	22,2 %	0,21	55,8 %	9,68	74,5 %
train total	test total	1,9 %	0,21	9,9 %	16,51	72,4 %
train total	train ps	43,1 %	0,21	66,1 %	2,93	62,0 %
train total	test ps	6,5 %	0,21	15,7 %	7,11	62,3 %
train total	train no ps	14,1 %	0,21	42,5 %	9,48	69,4 %
train total	test no ps	1,4 %	0,21	7,9 %	13,34	68,7 %
train ps	train total	22,2 %	0,21	50,5 %	6,05	70,9 %
train ps	test total	1,9 %	0,21	7,0 %	8,69	69,9 %
train ps	train ps	43,1 %	0,21	66,6 %	3,09	63,1 %
train ps	test ps	6,5 %	0,21	16,1 %	7,16	63,2 %
train ps	train no ps	14,1 %	0,21	36,6 %	5,38	67,8 %
train ps	test no ps	1,4 %	0,21	5,4 %	6,61	67,7 %
train total	train total	22,2 %	0,44	51,6 %	9,68	68,4 %
train total	test total	1,9 %	0,44	15,2 %	16,50	62,9 %
train total	train ps	43,1 %	0,44	66,0 %	2,93	68,5 %
train total	test ps	6,5 %	0,44	21,2 %	7,11	68,1 %
train total	train no ps	14,1 %	0,44	24,1 %	9,48	56,6 %
train total	test no ps	1,4 %	0,44	9,9 %	13,34	56,2 %
train ps	train total	22,2 %	0,44	51,6 %	6,05	70,0 %
train ps	test total	1,9 %	0,44	8,6 %	8,70	67,9 %
train ps	train ps	43,1 %	0,44	66,6 %	3,08	69,3 %
train ps	test ps	6,5 %	0,44	21,3 %	7,16	68,1 %
train ps	train no ps	14,1 %	0,44	38,3 %	5,37	66,6 %
train ps	test no ps	1,4 %	0,44	6,6 %	6,60	66,0 %

Tabla 6.3: Métricas regresión logística con 10 interacciones de variable ha

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	29,5 %	0,31	61,8 %	6,37	73,3 %
train total	test total	1,9 %	0,31	9,2 %	17,02	73,5 %
train total	train ps	61,0 %	0,31	79,6 %	1,94	61,5 %
train total	test ps	6,5 %	0,31	15,2 %	6,82	61,0 %
train total	train no ps	24,5 %	0,31	54,4 %	6,17	70,4 %
train total	test no ps	1,4 %	0,31	7,5 %	13,66	70,8 %
train ps	train total	29,5 %	0,60	51,4 %	2,32	63,4 %
train ps	test total	1,9 %	0,60	5,6 %	3,96	63,2 %
train ps	train ps	61,0 %	0,60	75,9 %	2,07	69,5 %
train ps	test ps	6,5 %	0,60	20,6 %	6,76	67,9 %
train ps	train no ps	24,5 %	0,60	45,2 %	2,96	62,4 %
train ps	test no ps	1,4 %	0,60	4,3 %	4,43	62,4 %

Tabla 6.4: Métricas regresión logística con 74 interacciones de variable ha

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	29,5 %	0,31	61,8 %	6,37	73,3 %
train total	test total	1,9 %	0,31	9,2 %	17,02	73,5 %
train total	train ps	61,0 %	0,31	79,6 %	1,94	61,5 %
train total	test ps	6,5 %	0,31	15,2 %	6,82	61,0 %
train total	train no ps	24,5 %	0,31	54,4 %	6,17	70,4 %
train total	test no ps	1,4 %	0,31	7,5 %	13,66	70,8 %
train ps	train total	29,5 %	0,67	50,3 %	2,33	62,7 %
train ps	test total	1,9 %	0,67	5,5 %	3,96	62,0 %
train ps	train ps	61,0 %	0,67	71,8 %	2,08	69,7 %
train ps	test ps	6,5 %	0,67	22,0 %	6,67	66,9 %
train ps	train no ps	24,5 %	0,67	45,3 %	2,96	62,5 %
train ps	test no ps	1,4 %	0,67	4,3 %	4,42	62,3 %

Tabla 6.5: Métricas regresión logística simple con 30 variables

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	29,5 %	0,31	57,3 %	4,49	69,5 %
train total	test total	1,9 %	0,31	7,9 %	9,65	69,7 %
train total	train ps	61,0 %	0,31	76,5 %	1,58	52,5 %
train total	test ps	6,5 %	0,31	12,8 %	3,34	52,6 %
train total	train no ps	24,5 %	0,31	48,8 %	4,07	66,1 %
train total	test no ps	1,4 %	0,31	6,3 %	7,32	66,4 %
train ps	train total	29,5 %	0,60	48,8 %	1,56	60,1 %
train ps	test total	1,9 %	0,60	4,9 %	2,13	60,4 %
train ps	train ps	61,0 %	0,60	66,3 %	1,68	63,1 %
train ps	test ps	6,5 %	0,60	18,3 %	3,92	63,1 %
train ps	train no ps	24,5 %	0,60	45,1 %	2,19	61,8 %
train ps	test no ps	1,4 %	0,60	4,0 %	3,13	62,0 %

Tabla 6.6: Métricas Random Forest

Base train	**Base pred**	**Balanceo target**	**Umbral**	**F1**	**G1-G3**	**AUC**
train total	train total	22,2 %	0,37	78,5 %	21,36	84,9 %
train total	test total	1,9 %	0,37	10,6 %	17,50	72,1 %
train total	train ps	43,1 %	0,37	87,0 %	4,11	76,9 %
train total	test ps	6,5 %	0,37	15,3 %	7,40	61,3 %
train total	train no ps	14,1 %	0,37	74,4 %	22,30	82,3 %
train total	test no ps	1,4 %	0,37	8,6 %	13,76	67,6 %
train ps	train total	22,2 %	0,51	62,5 %	6,52	73,4 %
train ps	test total	1,9 %	0,51	9,8 %	10,64	69,8 %
train ps	train ps	43,1 %	0,51	97,1 %	4,36	95,5 %
train ps	test ps	6,5 %	0,51	18,4 %	7,55	67,0 %
train ps	train no ps	14,1 %	0,51	48,5 %	4,40	65,9 %
train ps	test no ps	1,4 %	0,51	7,4 %	7,84	65,7 %