Prueba Analítica
Modelo Opciones de Pago
Fecha: 28 de enero de 2025
Autora: Nicole Ramírez Ramírez

1. Introducción

En el ámbito financiero, la capacidad de entender y predecir el comportamiento de los clientes resulta crucial para optimizar estrategias de negocio y fortalecer la relación con los usuarios. En este contexto, el presente informe describe el desarrollo y la implementación de un modelo de clasificación basado en aprendizaje automático, diseñado para evaluar la probabilidad de que un cliente acepte opciones de pago propuestas por el Banco.

2. Metodología

A continuación, se presenta la metodología utilizada para resolver el problema propuesto.

2.1. Selección de Variables

Se tiene la hipótesis de que la probabilidad de que un cliente acepte o no una opción de pago está relacionada con la capacidad de pago del cliente, es decir, cuánto dinero tiene y su historial de pagos. Tomando en cuenta la hipótesis planteada anteriormente, se realiza la selección de variables.

La base de datos `prueba_op_master_customer_data_enmascarado_completa` recopila información demográfica de los clientes del Banco, la cual resulta clave para el desarrollo del modelo. Es importante destacar que, para cada cliente, se utilizó únicamente el registro más reciente. A continuación, se presenta la lista de variables seleccionadas, las cuales se consideran relevantes para la hipótesis planteada, ya que podrían estar asociadas con la capacidad de pago del cliente.

- `genero_cli`: Se considera que podría capturar información relevante para el modelo a desarrollar. Para los clientes sin información registrada se asigna la moda de los datos.
- `edad_cli`: La edad del cliente es una variable que puede reflejar aspectos clave relacionados con su capacidad de pago. Inicialmente, se identificaron registros con edades menores de 18 años y mayores de 90 años, los cuales fueron considerados atípicos. Para estos casos, se asignó el valor promedio de la edad dentro del conjunto de datos.
- `estado_civil`: El estado civil de una persona puede reflejar información relevante sobre su estabilidad económica. Sin embargo, dado que la variable cuenta con siete categorías, se decidió agruparlas en tres nuevas categorías: `SOLTERO`, `PAREJA` y `NO INFORMA`.

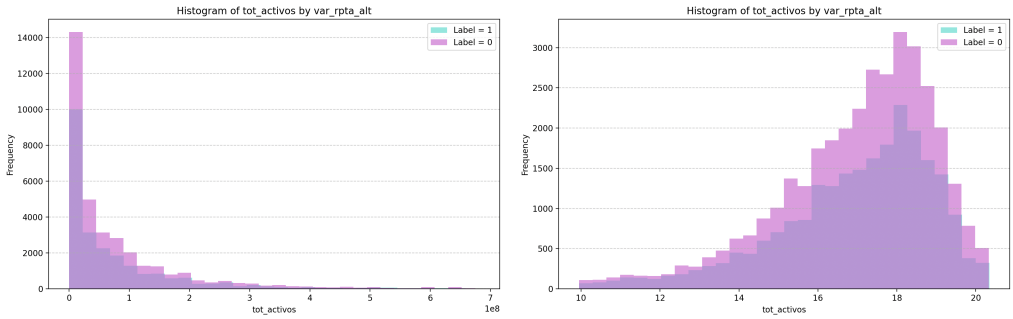
- `tipo_vivienda`: El tipo de vivienda puede reflejar el poder adquisitivo del cliente. Por ejemplo, una persona con vivienda propia podría tener una mayor capacidad económica. Durante el proceso de limpieza de esta variable, los clientes sin información registrada o con la etiqueta `NO INFORMA` fueron clasificados como `DESCONOCIDO`. Los demás registros conservaron sus etiquetas originales.

- `num_hijos` y `personas_dependientes`: Estas variables se transformaron en categóricas de la siguiente manera:

$$X_{ij} = \begin{cases} 1, & \text{Si el } i\text{-ésimo cliente tiene hijos} \\ 0, & \text{d.l.c.} \end{cases}$$

$$X_{ij} = \begin{cases} 1, & \text{Si el } i\text{-ésimo cliente tiene dependientes} \\ 0, & \text{d.l.c.} \end{cases}$$

- `tot_activos` y `total_ing`: Al estudiar la dinámica de estas variables, se evidencia que hay un gran número de ceros (ver Figura 1a), por lo que se decide eliminar estos registros y luego de esto identificar outliers.



(a) Datos originales. (b) Datos con $\log(1+x)$.

Figura (1) Histograma de `tot_activos`.

Para identificar outliers, primero se aplicó la transformación $\log(1+x)$ a todos los registros con el objetivo de normalizarlos parcialmente (ver Figura 1b). Posteriormente, utilizando un valor de *alpha* definido se calculan dos cuantiles para determinar los límites inferior y superior para la detección de outliers.

- `tot_pasivos`: Al igual que en las variables mencionadas en el ítem anterior, se identifica un gran número de ceros. Sin embargo, no se eliminan los registros con valor cero, ya que es importante capturar la información sobre si un cliente tiene deudas. Por lo tanto, se realiza un mapeo de la siguiente forma:

$$X_{ij} = \begin{cases} 1, & \text{Si el } i\text{-ésimo cliente tiene pasivos mayores a cero} \\ 0, & \text{d.l.c.} \end{cases}$$

- tot_patrimonio: Al analizar el comportamiento de esta variable, se identificaron registros con un valor de cero que no aportan información relevante al modelo. Por ello, se decidió trabajar únicamente con los registros que presentan un patrimonio mayor a cero.

- segm: El segmento asignado al cliente dentro del Banco se considera una fuente clave de información para evaluar su capacidad de pago.

- region_of: En esta variable, los registros con las etiquetas `DIRECCIÓN GENERAL`, `BANCO` y `BANCO DE COLOMBIA` fueron reasignados a la etiqueta `ANTIOQUIA`. Los demás registros conservaron sus etiquetas originales.

- egresos_mes: Al estudiar el comportamiento de esta variable, se decidió aplicar un proceso de bucketización para simplificar los datos continuos en cinco intervalos, creando así una variable discreta que puede resultar útil para el modelo.

La base de datos prueba_op_base_pivot_var_rpta_alt_enmascarado_trtest recopila información sobre la variable objetivo. De esta base se consideran importantes las siguientes variables.

- producto: El tipo de producto asociado a cada cliente es un factor relevante a considerar. Sin embargo, dado que existen 24 etiquetas posibles en los registros, se opta por agruparlas en seis nuevas categorías, con el objetivo de simplificar el modelo y reducir su complejidad.

- aplicativo: Al igual que el tipo de producto, el tipo de aplicativo se considera un factor importante, ya que los datos relacionados están equilibrados. Esto sugiere que podría proporcionar información valiosa para el modelo.

Por otro lado, esta base de datos da un punto de partida importante, ya que dice la fecha desde la cual se tiene información para la variable objetivo Y.

La base de datos prueba_op_probabilidad_oblig_base_hist_enmascarado_completa recopila información sobre diversos modelos desarrollados previamente por el Banco. De esta base se seleccionan tres variables que representan probabilidades asignadas a cada cliente, las cuales se consideran clave y relevantes para el modelo. Estas variables son:

- prob_propension: Probabilidad de hacer un pago el próximo mes.

- prob_alrt_temprana: Probabilidad de entrar en mora el próximo mes.

- prob_auto_cura: Probabilidad de que un cliente con una mora menor a 15 días se ponga al día sin ningún tipo de gestión directa o interacción humana.

Con el objetivo de identificar patrones temporales, como tendencias en el comportamiento de pago de los clientes, se ha decidido implementar ventanas móviles. A continuación, se describe la metodología utilizada para construir estas ventanas:

La estrategia consiste en considerar los datos de los tres meses previos al mes en el cual se desea realizar la predicción. Dado que la base de datos `prueba_op_base_pivot_var_rpta_alt_enmascarado_trtest` cuenta con información de la variable objetivo (Y) desde agosto de 2023 (202308), es necesario incluir datos desde mayo de 2023 (202305) hasta diciembre de 2023 (202312). Esto permite predecir el comportamiento de pago para el primer mes con datos de la variable de respuesta disponible, agosto de 2023, hasta el mes objetivo, enero de 2024 (202401). En la Figura 2 se pueden ver de manera gráfica las seis ventanas creadas.

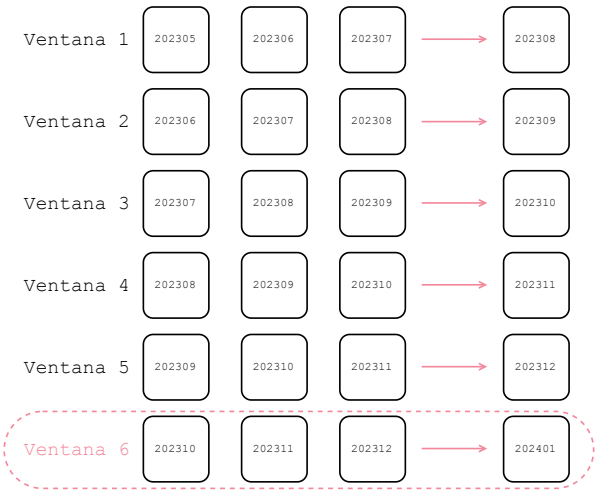


Figura (2) Ventanas Móviles.

Finalmente, en la base de datos `prueba_op_maestra_cuotas_pagos_mes_hist_enmascarado_completa` se recopila información sobre los pagos de las obligaciones, a continuación se presentan las variables seleccionadas:

- `pago_total`: Valor del pago total realizado por el cliente en el mes.

Dado que en esta base de datos se cuenta con información de todo el 2023 se decide crear las mismas ventanas móviles presentadas en la Figura 2 con el fin de capturar patrones temporales en el pago total de las obligaciones para cada cliente.

2.2. Preprocesamiento

Tras la selección de todas las variables, se genera una base de datos que consolida la información correspondiente a cada cliente y obligación en cada una de las ventanas. Las ventanas, numeradas de la 1 a la 5, junto con la variable objetivo, se estratifican con el propósito

de crear conjuntos de entrenamiento, validación y prueba. Esta estratificación asegura un balance adecuado tanto en la distribución de las ventanas como en la proporción de unos y ceros de la variable objetivo dentro de los conjuntos generados.

Posteriormente, se lleva a cabo la imputación de datos faltantes utilizando la estrategia *most frequent*. A continuación, se realizan las transformaciones necesarias a las variables, ajustándolas según su naturaleza.

Las transformaciones se implementan con la librería *scikit-learn* y se diferencian según el tipo de variable: numéricas o categóricas.

En el caso de las variables numéricas, se aplican dos transformaciones principales. Primero, se realiza una normalización con *StandardScaler* para ajustar los datos a una media de 0 y una varianza de 1. Después, para aquellas variables que presentan distribuciones asimétricas o sesgadas, se utiliza la transformación $\log(1 + x)$ con el fin de aproximar los datos a una distribución normal. Por otro lado, las variables categóricas se transforman mediante *OneHotEncoder*, lo que permite representar las categorías en un formato binario adecuado para su procesamiento.

2.3. Selección de Modelo

Random Forest y XGBoost son los modelos más utilizados para trabajar con datos tabulares. Por esta razón, se empleará un GridSearch para la selección de hiperparámetros, optimizando el rendimiento del modelo en función del F1 Score.

2.3.1. Random Forest

El modelo RandomForestClassifier fue definido con un estado aleatorio fijo para garantizar la reproducibilidad de los resultados. Posteriormente, se estableció una rejilla de hiperparámetros para realizar una búsqueda exhaustiva mediante GridSearchCV.

Se utilizó la validación cruzada con tres particiones para evaluar el desempeño de las diferentes combinaciones de hiperparámetros en los datos de entrenamiento. La métrica utilizada para evaluar las combinaciones fue el F1 Score macro, que permite medir el equilibrio entre la precisión y el recall en el contexto de un conjunto de datos desbalanceado.

Tras completar la búsqueda con GridSearchCV, se identificaron los mejores hiperparámetros para el modelo: 500 árboles y 20 muestras requeridas para dividir un nodo interno. El mejor modelo fue ajustado con estos hiperparámetros y luego evaluado en los conjuntos de validación y prueba.

Este modelo obtuvo un score de 0.54474 en el Leaderboard de Kaggle.

2.3.2. XGBoost

Aplicando una metodología similar a la utilizada en el modelo de Random Forest, se llevó a cabo una búsqueda de hiperparámetros para optimizar el modelo XGBoost. Este modelo es reconocido como uno de los métodos de referencia para el análisis de datos tabulares. En este caso, el XGBoost logró superar el baseline del modelo anterior, alcanzando un score de 0.57624.

Se identificaron los mejores hiperparámetros para el modelo: 400 estimadores y una tasa de aprendizaje de 0.05.

Es importante resaltar que ambos modelos fueron entrenados tomando en cuenta el desbalanceo que existe en la variable objetivo, esto se logró utilizando los parámetros 'class_weight': ["balanced"] y 'scale_pos_weight': [1.5] en los modelos Random Forest y XGBoost, respectivamente.

2.4. Trabajo Futuro

El desempeño del modelo obtenido es aceptable, considerando las limitaciones de tiempo y el conocimiento limitado del contexto de los datos proporcionados por el Banco. No obstante, existe un margen considerable de mejora. Explorar más tipos de modelos y realizar una optimización más exhaustiva de los hiperparámetros podría mejorar el desempeño actual, aunque es probable que estas mejoras no sean sustanciales. Para lograr un avance significativo en el desempeño, es esencial crear nuevas variables que aporten mayor capacidad discriminante. Por ejemplo, calcular una proporción entre los ingresos mensuales del cliente y la cuota mensual a pagar podría proporcionar información clave para enriquecer el modelo.

Asimismo, no todas las variables actualmente utilizadas son necesariamente relevantes para el desempeño del modelo. Identificar y seleccionar aquellas que realmente aportan valor es fundamental para reducir la complejidad del modelo y mejorar su eficiencia. Una selección cuidadosa de variables, combinada con el diseño de nuevas características relevantes, puede ser la clave para alcanzar un desempeño notablemente superior.

Finalmente, podemos hacer un mejor uso de las variables autoregresivas, por ejemplo, se puede utilizar `prob_auto_cura` en los últimos n meses para estimar si esta probabilidad crece o decrece en el tiempo.

3. Despliegue

El modelo presentado en este informe puede ser de gran utilidad para los empleados de Bancolombia encargados de ofrecer alternativas de pago a sus clientes. Este beneficio puede maximizarse si los empleados tienen la capacidad de interactuar con el modelo en tiempo real.

Por ejemplo, justo antes de una llamada, un empleado podría ajustar su guion de acuerdo con la probabilidad proporcionada por el modelo. Si esta probabilidad es baja, el asesor tendría la oportunidad de reflexionar cuidadosamente sobre los argumentos que presentará al cliente.

Asimismo, el asesor podría retroalimentar al modelo inmediatamente después de concluir la llamada. Esto permitiría una evaluación constante del modelo, facilitando la detección de cambios repentinos en sus tasas de acierto e, incluso, la implementación de estrategias de aprendizaje continuo (online learning).

Para alcanzar esta arquitectura ideal, es fundamental contar con la infraestructura tecnológica adecuada. En este caso, se propone que el modelo se ejecute en lotes de clientes utilizando la infraestructura existente del banco. Posteriormente, los resultados podrían ser expuestos mediante una API creada con herramientas como FastAPI. Esta API podría implementarse en la nube utilizando servicios como AWS EC2.

Si el objetivo es incorporar aprendizaje continuo (online learning), esta lógica de negocio también podría desarrollarse en la nube, utilizando microservicios web de AWS en colaboración con los equipos de ingeniería de datos y arquitectura de Bancolombia.