Modelo opción de pago

**veco-opeco-modelo-opcion-pagos**

Operación Ecosistemas

**Autores**Juan Pablo Jaramillo Castrillon

**Fecha de última revisión**06/02/2025

Contenido

[Historial de revisiones 3](#_Toc174094160)

[Introducción 4](#_Toc174094161)

[Objetivo 4](#_Toc174094162)

[Impactos esperados 4](#_Toc174094163)

[Impactos al negocio 4](#_Toc174094164)

[Impactos técnicos 5](#_Toc174094165)

[Estructura del proyecto 5](#_Toc174094166)

[Insumos 5](#_Toc174094167)

[Salidas 6](#_Toc174094168)

[Diagrama conceptual 7](#_Toc174094169)

[Diagrama de capa de datos 8](#_Toc174094170)

[Diagrama de flujo 9](#_Toc174094171)

[Modelo entidad relación (MER) 10](#_Toc174094172)

[Proyecto 11](#_Toc174094173)

[Requerimientos 11](#_Toc174094174)

[Instalación 11](#_Toc174094175)

[Instanciación 11](#_Toc174094176)

[Periodicidad 12](#_Toc174094177)

[Soporte técnico 12](#_Toc174094178)

[Errores comunes 12](#_Toc174094179)

# Historial de revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha (dd/mm/yyyy) | Versión | Descripción/Cambio realizado | Autor y/o participantes |
| 06/02/2025 | 1.0 | Creación del documento | Juan Pablo Jaramillo Castrillón |

# Introducción

Bancolombia ofrece un amplio portafolio de productos de crédito para diversos segmentos de clientes, con obligaciones que generalmente se pagan en cuotas mensuales. Aunque la mayoría de los clientes cumplen con sus pagos, un porcentaje cae en mora, lo que activa estrategias de gestión desde el primer día de incumplimiento. Para evitar que estas obligaciones lleguen a procesos de judicialización o castigo, el banco ofrece opciones de pago que buscan aliviar la carga financiera de los clientes en mora.

Estas opciones incluyen ampliaciones de plazo, reducciones de cuota, renegociaciones de tasas de interés y reestructuraciones de crédito. Sin embargo, no todos los clientes tienen acceso a ellas y solo se pueden preaprobar hasta tres opciones por obligación y por mes. Además, una vez aplicada una opción de pago, deben transcurrir entre tres y cuatro meses antes de poder optar por otra.

Dado el alto volumen de clientes en mora, Bancolombia emplea un sistema de priorización basado en la exposición de la deuda y la probabilidad de pago. No obstante, el banco busca anticiparse y predecir, con un mes de antelación, la probabilidad de que un cliente acepte una opción de pago. Incorporar esta información en el sistema de priorización permitiría una gestión más eficiente, enfocándose en clientes con alta probabilidad de aceptación y contribuyendo a desacelerar el índice de cartera vencida.

# Objetivo

El objetivo es mejorar la eficiencia en la gestión de cartera en mora de Bancolombia mediante la incorporación de un modelo predictivo que anticipe, con un mes de anticipación, la probabilidad de que un cliente acepte una opción de pago. Esto permitirá optimizar la priorización de clientes dentro del sistema de gestión, enfocando los esfuerzos en aquellos con mayor disposición a regularizar su deuda. Como resultado, se busca reducir el índice de cartera vencida, minimizar costos asociados a la judicialización y mejorar la recuperación del capital expuesto.

# Impactos esperados

Los impactos esperados de la implementación del modelo predictivo en la gestión de cartera en mora de Bancolombia incluyen:

1. **Mayor eficiencia en la gestión de cartera vencida**: Se optimizarán los esfuerzos del banco al priorizar clientes con alta probabilidad de aceptar opciones de pago, mejorando la efectividad de las estrategias de cobranza.
2. **Reducción del índice de cartera vencida**: Al incrementar la cantidad de clientes que regularizan su deuda antes de alcanzar niveles críticos de mora, se desacelerará el crecimiento de la cartera en mora.
3. **Disminución de costos operativos**: Se reducirá el gasto en procesos de judicialización y castigo de deuda, así como en gestiones de cobranza innecesarias para clientes con baja probabilidad de pago.
4. **Mejora en la recuperación del capital**: Al ofrecer opciones de pago estratégicas, el banco podrá recuperar un mayor porcentaje del dinero prestado, minimizando el riesgo de pérdida de capital.
5. **Optimización del uso de recursos**: Se asignarán de manera más eficiente los recursos humanos y tecnológicos en la gestión de cobranza, enfocándose en los clientes con mayor potencial de pago.
6. **Mejora en la experiencia del cliente**: Los clientes recibirán ofertas de pago más adecuadas a su situación financiera, lo que puede fortalecer la relación con el banco y fomentar su fidelización

Los impactos esperados del proyecto pueden clasificarse en dos categorías: impactos al negocio e impactos técnicos.

## Impactos al negocio

* **Tomar decisiones más acertadas:** Contar con bases de datos consolidadas bajo permanente actualización nos permite obtener una visión integral y granular del cliente y el producto, comprender sus preferencias, necesidades y comportamientos; lo que impulsa una toma de decisiones más informada, estratégica y altamente personalizada.
* **Mejorar la experiencia del cliente:** Al comprender la información es posible ofrecer experiencias más personalizadas, tanto en productos y servicios como en el soporte postventa, aumentando significativamente la satisfacción del cliente.
* **Sincronización de metas:** Al democratizar la información, hacemos visibles los indicadores clave y permitimos realizar ajustes oportunos en función del comportamiento del mercado y del público objetivo. Esto garantiza que todas las operaciones estén alineadas con los objetivos estratégicos del negocio.
* **Informes precisos y detallados:** Simplifica el cumplimiento normativo y minimiza riesgos generando informes precisos y detallados que satisfacen los requisitos de reguladores, fortaleciendo la confianza en la organización y el producto.

## Impactos técnicos

El proyecto incorpora prácticas de MLOps para asegurar la reproducibilidad, automatización y mantenimiento del modelo. Esto incluye:

* **Versionamiento de Modelos:** Uso de estrategias para guardar y comparar versiones del modelo antes del despliegue.
* **Automatización de Pipeline:** Implementación de pipelines de entrenamiento y despliegue para reducir intervenciones manuales.
* **Monitoreo de Desempeño:** Evaluación continua del modelo para detectar derivas en los datos y degradación del rendimiento.
* **Uso de Pipelines:** Asegura un flujo de trabajo limpio y reproducible en el preprocesamiento de datos y entrenamiento del modelo.
* **Optimización de Código:** Implementación de GridSearchCV para la búsqueda de hiperparámetros, garantizando mejor rendimiento.
* **Despliegue en Entornos Productivos:** Configuración para facilitar la integración con sistemas de producción.
* **Modularidad del Código:** Estructura orientada a clases y métodos que permite actualizaciones sin afectar el flujo de trabajo completo.

# Estructura del proyecto

Para una mejor comprensión de cómo funciona el proyecto, a continuación, se detallan sus componentes principales:

## Insumos

Se emplean las bases de datos puras del producto:

* **prueba\_op\_base\_pivot\_var\_rpta\_alt\_enmascarado\_trtest:** Basede datos que contiene la variable respuesta y las principales características de la gestión, de resultados del pago y de características de las opciones de pago habilitadas para el cliente en el mes de gestión o de evaluación de la variable
* **prueba\_op\_probabilidad\_oblig\_base\_hist\_enmascarado\_completa:** Información asociada con los resultados de los modelos analíticos existentes para la cobranza de Bancolombia.
  + **Alerta temprana:** describe la probabilidad que un cliente en su obligación entre en mora en su próxima fecha de pago
  + **Auto cura:** Describe la probabilidad de que un cliente que esté en mora en su obligación pague por sí solo, o sin ninguna gestión directa (llamada)
  + **Propensión de pago:** Describe la probabilidad de que un cliente que esté en mora en su obligación haga un pago a su obligación.
* **prueba\_op\_master\_customer\_data\_enmascarado\_completa**: Información asociada a las características generales del cliente o demográficas de forma mensual.
* **prueba\_op\_maestra\_cuotas\_pagos\_mes\_hist\_enmascarado\_completa:** Información que describe el comportamiento de pagos del cliente en sus obligaciones a lo largo del tiempo.

## Modelos

1. Modelos Utilizados

Los modelos aplicados en el análisis incluyen:

1. Logistic Regression (Regresión Logística)
2. Random Forest Classifier (Clasificador de Bosques Aleatorios)
3. Decision Tree Classifier (Clasificador de Árboles de Decisión)
4. XGBoost Classifier
5. LightGBM Classifier

2. Evaluación y Selección de Modelos

2.1. Logistic Regression

* Ventajas: Simple de implementar y fácil de interpretar.
* Desventajas: Limitado en su capacidad para capturar relaciones no lineales.
* Conclusión: Adecuado para problemas lineales y como modelo base para comparación.

2.2. Random Forest Classifier

* Ventajas: Robusto al sobreajuste, maneja bien datos no lineales y proporciona una buena precisión.
* Desventajas: Menos interpretable y más costoso computacionalmente.
* Conclusión: Excelente opción para problemas complejos con datos no lineales.

2.3. Decision Tree Classifier

* Ventajas: Fácil de interpretar y no requiere preprocesamiento de datos.
* Desventajas: Propenso al sobreajuste, especialmente con árboles profundos.
* Conclusión: Útil para interpretabilidad y como componente en modelos de ensamble.

2.4. XGBoost Classifier

* Ventajas: Alta precisión, eficiente computacionalmente y maneja bien datos no lineales.
* Desventajas: Menos interpretable y requiere ajuste de hiperparámetros.
* Conclusión: Uno de los mejores modelos para problemas de clasificación complejos, especialmente en competiciones de machine learning.

2.5. LightGBM Classifier

* Ventajas: Alta precisión, rápido en entrenamiento y eficiente computacionalmente.
* Desventajas: Menos interpretable y requiere ajuste de hiperparámetros.
* Conclusión: Similar a XGBoost, pero con ventajas adicionales en términos de velocidad y eficiencia.

3. Proceso de Calibración y Evaluación

Para optimizar y evaluar los modelos, se utilizaron los siguientes enfoques:

* GridSearchCV: Se implementó GridSearchCV para realizar una búsqueda exhaustiva de los mejores hiperparámetros de cada modelo, utilizando validación cruzada para garantizar la robustez de los resultados.
* Métricas de Evaluación: Se analizaron los modelos mediante métricas como precisión, precision weighted, recall weighted y F1-score weighted para seleccionar el más eficiente.

4. Despliegue y Ejecución

El modelo seleccionado se integró en el flujo de trabajo siguiendo estos pasos:

1. Guardado del Modelo: Se almacena el mejor modelo en el disco utilizando pickle para su reutilización.
2. Carga del Modelo: El modelo guardado se recupera desde el disco con joblib.
3. Predicción: Se realizan predicciones sobre los datos de prueba y se obtienen las probabilidades de la clase positiva.
4. Subida de Resultados: Los resultados de las predicciones se cargan en el sistema Sparky para su análisis y posterior uso.

## Salidas

En la siguiente sección se exponen las fuentes de información que se emplearon para el desarrollo:

* **proceso\_ecosistemas.opeco\_prediccion\_pagos:** Para cada combinación cliente-obligación entregada en el archivo de calificación de la muestra fuera de tiempo, se debe predecir la variable respuesta llamada (var\_rpta\_alt), que tomará un valor de 0 o 1. El archivo deberá tener el siguiente formato.
* **proceso\_ecosistemas.opeco\_prediccion\_pagos\_resultado:** La información en la base de datos se almacenará en una tabla que contendrá cada combinación única de cliente-obligación, junto con la predicción de la variable respuesta y su probabilidad.

## Diagrama conceptual

El diagrama ejemplifica la estructura principal y el flujo de ejecución del proyecto.

Partiendo de la estructura del orquestador 2.0, el proyecto cuenta con:

* **.venv:** Carpeta donde se almacena el entorno virtual dispuesto para este proyecto.
* **logs:** Carpeta donde se almacena los logs resultados de las diferentes ejecuciones.
* **static:** Carpeta donde se almacenan los archivos estáticos (.sql, .json, etc) necesarios para la ejecución del proyecto.
* **sql:** Carpeta donde se almacenan los .sql que dan vida a las tablas resultantes.
* **sql / 000\_create\_tables:** Carpeta que contiene los archivos sql que crean las tablas resultantes y empleada en el proyecto como maestro de validaciones y logs.
* **sql / 001\_first\_time:** Carpeta que contiene los archivo sql para llenar las tablas con la información histórica.
* **sql / 002\_recurrent:** Carpeta que contiene los deltas necesarios para ejecutar de forma periódica el proyecto.
* **config.json:** Archivo con los parámetros necesarios para la ejecución del proyecto.
* **ejecution.**py: Codigo donde se instancia la ejecucion del proyecto, este ejecuta el maestro de validaciones, los pasos definidos en etl.py, guarda los diferentes logs.
* **etl.py:** Codigo que contiene los pasos definidos para transformaciones y/o ejecuciones.
* **.PypipEnv.bat:** Ejecutable que despliega dentro del proyecto el entorno virtual y los requerimientos necesarios para la ejecución.
* **PyRun.bat:** Ejecutable para realizar la respectiva ejecución del proyecto.

El proceso se inicia con la carga de un archivo de configuración (config.json) y scripts SQL. Estos son procesados por un script de Python (ejecucion.py) que sigue las instrucciones definidas en etl.py. Una vez finalizada la ejecución, se genera un log detallado. Si la ejecución es exitosa, los datos se cargan en la zona de aterrizaje (LZ) y se actualiza un tablero en Power BI.

## Diagrama de flujo

El diagrama representa la ejecución interna del proyecto y los cruces de información realzados.

Diagrama

Descripción generada automáticamente

## Modelo entidad relación (MER)

En la imagen se puede observar las diferentes relaciones entre las tablas resultantes de la ejecución de la ETL.

Tabla

Descripción generada automáticamente con confianza media

# Proyecto

Para realizar la configuración y ejecución del proyecto es necesario cumplir tanto con conocimientos técnicos como con los permisos y programas asociados al entorno analítico.

## Requerimientos

Para la ejecución del proyecto, es necesario contar con:

* Licencia LZ Administrador.
* Python versión 3.9.12.
* Acceso a las tablas insumo detalladas en la sección anterior.
* Acceso a la zona de resultados de escritura detallada en la sección anterior.

## Instalación

Para realizar la ejecución en equipos locales se cuenta con dos alternativas:

* Descargar el proyecto del repositorio oficial en Azure Devops, en un archivo comprimido y descomprimirlo en la carpeta donde se desea trabajar.
* Realizar una clonación al proyecto en el repositorio oficial de Azure Devops [veco-opeco-bnplexperience - Repos (visualstudio.com)](https://grupobancolombia.visualstudio.com/Vicepresidencia%20de%20Innovaci%C3%B3n%20y%20%20Transformaci%C3%B3n%20Digital/_git/veco-opeco-bnplexperience).

git clone https://grupobancolombia.visualstudio.com/Vicepresidencia%20de%20Innovaci%C3%B3n%20y %20Transformaci%C3%B3n%20Digital/\_git/veco-opeco-bnplexperience

## Instanciación

Al contar con el proyecto en local se realiza el siguiente proceso para obtener los resultados:

* **Configurar el entorno virtual:** Ejecutar el archivo “.PyPipEnv.bat”, este se encuentra encargado de crear e instalar los requerimientos para el correcto funcionamiento del proyecto.
* **Realizar la ejecución:** Ejecutar el archivo “PyRun.bat” o correr directamente el módulo “ejecutar.py”

Para que la ejecución sea exitosa, es fundamental que los datos de entrada estén completos y actualizados.

## Periodicidad

La ejecución puede ser realizada bajo múltiples escenarios:

* **Automática:** Siendo optativo a ser calendarizado, se espera una ejecución diaria.
* **A demanda:** En caso de no estar calendarizado se realizará una ejecución cada x tiempo (negociado con el área).

Si han pasado más de 15 días desde la última actualización (configurable en el config), se recomienda revisar y actualizar los datos antes de ejecutar el proceso.

# Soporte técnico

Ante un incidente o error con las bases de datos resultantes, por favor escalar un requerimiento al siguiente grupo “OPERACION BAAS Y OPEN FINANCE ANALYTIC”

## Errores comunes

* El proceso es una ejecución de módulos de sql, el error esperado se puede deber a cambios en los nombres de los campos de las nuevas fcrs de clientes, los cuales reemplazaron la master customer data.
* Al pasar un periodo de tiempo de ejecución superior a 15 días (delta dentro del config) se debe modificar este previo a la ejecución.