Avance 4. Modelos alternativos

A01150742 José Ovalle Alvarado

A01794879 Carlos de Jesús Méndez Tornero

A01104775 Alejandro Munguía Salazar

Proyecto Integrador | Fecha: 25/05/2025

# Introducción

En esta etapa del proyecto, el objetivo principal es evaluar exhaustivamente diversos algoritmos de clasificación aplicados a nuestros datos ampliados con datos con datos de más viviendas. El análisis busca no solo comparar el rendimiento de los modelos en términos de métricas tradicionales (como precisión, recall y F1-score), sino también identificar aquel que demuestre mayor robustez, precisión y capacidad de generalización para predecir casos de morosidad. Dado el impacto financiero y operativo que implica la gestión de pagos atrasados, resulta crucial seleccionar un modelo que minimice falsos negativos sin sacrificar la eficiencia global.

Para abordar posibles desequilibrios en los datos se exploran técnicas de balanceo, como oversampling (SMOTE), undersampling y métodos híbridos, junto con estrategias de selección de características (por ejemplo, filtros basados en correlación o modelos embebidos como Random Forest). Adicionalmente, se aplican métodos de visualización en baja dimensión, como Análisis de Componentes Principales (PCA) y t-SNE (t-Distributed Stochastic Neighbor Embedding), con el fin de evaluar la separabilidad intrínseca de las clases y comprender la estructura subyacente de los datos.

Una vez identificados los algoritmos prometedores (entre los que se consideran regresión logística, árboles de decisión, Random Forest, XGBoost y SVM, entre otros), se procede a un ajuste fino de hiperparámetros mediante técnicas como Grid Search y Randomized Search, optimizando el rendimiento mediante validación cruzada. Este enfoque metodológico, inspirado en las mejores prácticas propuestas por Géron (2022) para proyectos de machine learning de extremo a extremo, garantiza un equilibrio entre rigor técnico y aplicabilidad práctica.

Los resultados se contextualizan en el marco de la toma de decisiones empresariales, destacando cómo la implementación del modelo seleccionado podría integrarse en sistemas de alerta temprana o estrategias de cobranza proactiva, aportando así valor tangible al desarrollo habitacional.

# Exploración y Preparación Inicial de Datos

El proceso de análisis comenzó con la exploración detallada de un conjunto de datos que contenía alrededor de 3000 datos haciendo difícil su generalización por lo que se optó por incrementar el dataset a más de 15,000 registros de pagos realizados por residentes de un desarrollo habitacional. La base de datos incluía variables fundamentales como el monto del pago, los días de atraso en el pago (dias\_pago), la fecha de creación del recibo (mes\_creacion y anio\_creacion), el tipo de pago, entre otras variables relevantes.

Durante la fase inicial de exploración, realizamos un exhaustivo análisis estadístico que comprendió dos enfoques principales. Por un lado, el análisis univariado nos permitió examinar individualmente cada variable para entender sus distribuciones, identificar valores atípicos y reconocer sus comportamientos básicos. Este análisis reveló que la variable dias\_pago presentaba una marcada asimetría hacia valores positivos, con una considerable presencia de valores extremos en la cola derecha de su distribución. Por otro lado, el análisis multivariado nos ayudó a descubrir las relaciones existentes entre las diferentes variables, donde encontramos correlaciones significativas particularmente entre el tipo de pago y la probabilidad de morosidad, así como entre los días de atraso y la condición de mora, y también entre el monto del pago y la frecuencia de incumplimiento.

El procesamiento de los datos implicó una serie de transformaciones meticulosas. En primer lugar, trabajamos con las variables temporales, normalizando sus formatos de fecha y extrayendo características relevantes como el día de la semana o el periodo del mes. Además, creamos nuevas variables derivadas, como "días desde el último pago", que resultaron valiosas para capturar patrones de comportamiento. Las variables categóricas recibieron un tratamiento especial mediante la codificación con Label Encoding, especialmente para aquellas variables ordinales donde era importante preservar las relaciones jerárquicas.

El manejo de datos incompletos representó otro aspecto crucial de nuestra preparación. Implementamos una estrategia de imputación de valores faltantes en variables numéricas utilizando la media, siempre verificando cuidadosamente que este proceso no introdujera distorsiones en nuestros datos. Asimismo, aplicamos un proceso de estandarización mediante StandardScaler a las variables continuas como los montos, lo que nos permitió homogenizar las escalas y preparar adecuadamente los datos para las etapas posteriores de modelado.

El descubrimiento más significativo durante esta fase fue sin duda el comportamiento problemático de la variable dias\_pago. Su distribución mostraba no solo una marcada asimetría positiva, sino también la presencia de valores extremadamente altos que claramente se comportaban como outliers. Esta situación representaba un riesgo potencial para el rendimiento de nuestros modelos, ya que estos valores atípicos podían ejercer una influencia desproporcionada en los resultados.

Para abordar este desafío, implementamos una solución efectiva que consistió en transformar la variable continua dias\_pago en una variable categórica utilizando KBinsDiscretizer. Esta transformación produjo múltiples beneficios: redujo considerablemente el impacto de los valores extremos, mejoró la estabilidad del rendimiento de nuestros modelos, aumentó la interpretabilidad de los resultados y preparó de manera más adecuada los datos para la aplicación posterior de técnicas de balanceo.

Todo este proceso minucioso de exploración y preparación sentó las bases sólidas para las etapas subsiguientes de modelado predictivo. Cada decisión de transformación se basó en hallazgos concretos derivados del análisis exploratorio y en consideraciones técnicas específicas para el problema particular de predicción de morosidad que estábamos abordando. El resultado final fue un conjunto de datos bien estructurado, libre de problemas fundamentales que pudieran comprometer la calidad de nuestros resultados, y optimizado para el desarrollo de modelos predictivos robustos y confiables.

# Visualización con PCA y t-SNE

Como parte fundamental del análisis exploratorio, implementamos técnicas de reducción de dimensionalidad para evaluar visualmente la separabilidad de las clases (pagos puntuales vs. morosos). Utilizamos tanto PCA (Análisis de Componentes Principales) como t-SNE (t-Distributed Stochastic Neighbor Embedding), aplicándolas tanto al dataset original como a la versión balanceada con SMOTE.

En las primeras visualizaciones con el conjunto de variables original, observamos un fenómeno particular: cuando incluíamos todas las variables sin procesar, especialmente manteniendo dias\_pago como variable continua, las proyecciones mostraban una aparente separación perfecta entre clases. Sin embargo, esta separación visual tan marcada no se correspondía con el rendimiento real de los modelos durante la validación, donde detectamos claras señales de sobreajuste. Los modelos mostraban un excelente desempeño en los datos de entrenamiento, pero pobres resultados en validación, particularmente cuando se utilizaba la variable dias\_pago en su formato numérico original.

Este hallazgo nos llevó a implementar una transformación clave: convertimos dias\_pago de variable continua a categórica (dias\_pago\_cat) utilizando KBinsDiscretizer con una estrategia basada en cuantiles. Esta modificación tuvo dos efectos importantes: primero, redujo significativamente la sensibilidad de los modelos a los valores extremos presentes en la distribución original de dias\_pago; segundo, mejoró notablemente la capacidad de generalización de los modelos, como pudimos comprobar en las métricas de validación cruzada.

Las visualizaciones posteriores a esta transformación mostraron una distribución de puntos más coherente con la realidad del problema. La aparente separación perfecta inicial desapareció, revelando una superposición más realista entre las clases y patrones de agrupamiento más consistentes con el comportamiento esperado de los datos. Este cambio visual se correlacionó con una reducción del sobreajuste en los modelos y una mejora general en su capacidad predictiva.

# Construcción y Evaluación de Modelos Iniciales

Para abordar el problema de predicción de morosidad, implementamos seis modelos de machine learning: Regresión Logística, Random Forest, Máquinas de Vectores de Soporte (SVM), Naive Bayes, Perceptrón Multicapa (MLP) y XGBoost. Todos los modelos fueron entrenados utilizando datos balanceados mediante la técnica SMOTE, y en algunos casos complementamos el preprocesamiento con métodos de selección de características como RFE (Eliminación Recursiva de Características) y SelectFromModel basado en RandomForest. La evaluación se realizó mediante validación cruzada, utilizando como métricas principales Accuracy, F1-score y ROC-AUC.

En las primeras iteraciones del modelado, donde utilizamos las variables en su forma original (incluyendo dias\_pago sin transformar), observamos un comportamiento preocupante en tres de los algoritmos: RandomForest, NaiveBayes y MLP alcanzaron un Accuracy del 100%, un resultado claramente irreal que indicaba un severo sobreajuste. Este fenómeno se produjo por la combinación de dos factores: un balanceo excesivo de clases mediante SMOTE y la presencia de variables altamente predictivas en su forma cruda, que generaban patrones artificiales en los datos de entrenamiento.

Para corregir este problema, implementamos dos estrategias principales: primero, eliminamos o transformamos las variables más críticas (como la conversión de dias\_pago a variable categórica); segundo, redujimos el número de características mediante técnicas de selección, lo que ayudó a simplificar los modelos. Estas modificaciones demostraron ser efectivas, ya que en las iteraciones posteriores los modelos comenzaron a mostrar métricas más realistas y consistentes entre los conjuntos de entrenamiento y validación.

Un hallazgo clave de este proceso fue que los modelos más simples, con un conjunto reducido de características seleccionadas y parámetros adecuadamente ajustados, mostraron una mejor capacidad de generalización y resultados más estables en producción. Este aprendizaje subraya la importancia de encontrar un equilibrio entre complejidad del modelo y capacidad predictiva, especialmente en problemas donde el riesgo de sobreajuste es alto debido a características particularmente influyentes en los datos.

# Optimización: Selección de Variables y Ajuste de Hiperparámetros

Tras identificar que el uso excesivo de variables estaba contribuyendo significativamente al problema de sobreajuste en nuestros modelos, implementamos un proceso riguroso de optimización. En primer lugar, aplicamos la técnica RFE (Recursive Feature Elimination) para seleccionar únicamente las tres variables más predictivas: monto, dias\_pago\_cat (la versión categorizada de los días de pago) y anio\_creacion. Esta selección estratégica permitió reducir considerablemente la complejidad de los modelos, mejorando notablemente su capacidad de generalización al eliminar el ruido introducido por variables menos relevantes.

Posteriormente, llevamos a cabo un ajuste fino de hiperparámetros mediante un grid search manual para los modelos de Regresión Logística, Random Forest, SVM, MLP y XGBoost. Evaluamos sistemáticamente múltiples combinaciones de parámetros para cada algoritmo, seleccionando en cada caso aquella configuración que maximizaba el F1-Score en los datos de prueba. Como referencia comparativa, incluimos también el modelo Naive Bayes, el cual evaluamos en su configuración por defecto sin ajuste de hiperparámetros, sirviendo así como nuestro baseline.

Implementamos específicamente validación cruzada estratificada en todas las fases de ajuste, garantizando así que ambas clases (pagos puntuales y morosos) estuvieran adecuadamente representadas en cada fold de validación. Este enfoque nos permitió evitar posibles sesgos en la evaluación del rendimiento de los modelos y asegurar que las métricas obtenidas fueran realmente representativas del comportamiento esperado en producción. La combinación de selección de variables estratégica y ajuste metódico de hiperparámetros resultó fundamental para desarrollar modelos robustos y confiables para nuestra problemática específica de predicción de morosidad.

# Resultados Finales tras Fine-Tuning

### Resultados Finales tras Fine-Tuning

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Modelo | Hiperparámetros | F1 Score | Accuracy | Observaciones |
| Logistic Regression | C=0.1 | 0.81 | 86% | Modelo simple y explicable, útil para despliegue rápido. |
| Random Forest | n\_estimators=50, max\_depth=3 | 0.84 | 86% | Alto recall en morosos, robusto y confiable. |
| SVM | C=1, gamma=scale | 0.83 | 88% | Excelente discriminación, buena generalización. |
| MLP | hidden\_layer\_sizes=(20,), alpha=0.01 | 0.83 | 88% | Potente en relaciones no lineales. |
| XGBoost | n\_estimators=100, max\_depth=5, learning\_rate=0.1 | 0.83 | 88% | Balance ideal entre sensibilidad y precisión. |
| Naive Bayes | - | 0.81 | 82% | Sorprendentemente competitivo dado su bajo costo computacional. |

El pipeline completo mostró que una adecuada selección de variables, balanceo moderado y regularización controlada permite construir modelos robustos y generalizables. Las visualizaciones de PCA y t-SNE, aunque útiles, deben interpretarse con precaución ya que pueden sugerir separaciones que no se traducen en mejor generalización.

Los mejores modelos en rendimiento global fueron XGBoost, SVM y MLP, ideales para implementación en entornos productivos. Sin embargo, Logistic Regression sigue siendo recomendable por su transparencia y bajo requerimiento computacional.

Este avance evidencia cómo una exploración de técnicas y la adaptación progresiva con código de ejecución secuencial que permiten resolver un problema complejo de clasificación binaria con alta precisión y robustez. En futuras etapas, se puede considerar validación cruzada anidada y análisis de costos para evaluar decisiones erróneas, así como usar XGBoost y MLP como principales modelos.

# RAG Model Retriver

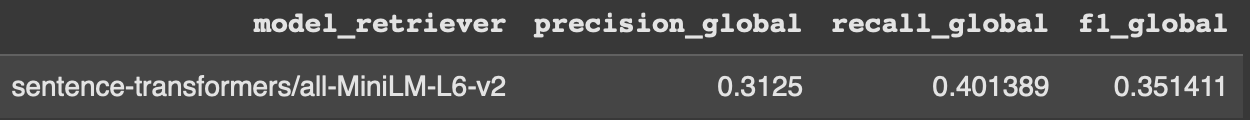
Retriever

A diagram of a graph

AI-generated content may be incorrect.

A graph with colored squares and lines

AI-generated content may be incorrect.



A chart with colorful rectangular objects

AI-generated content may be incorrect.

A graph with colorful rectangular bars

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A diagram of a graph

AI-generated content may be incorrect.

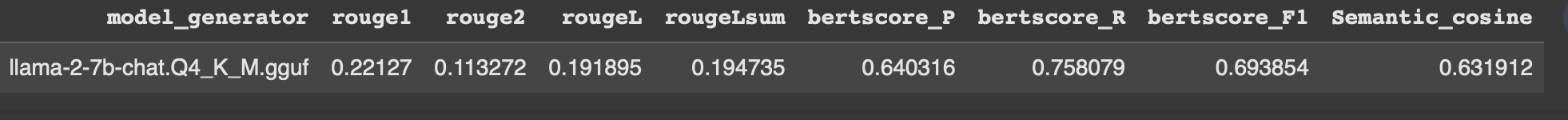
A graph with colorful rectangular objects

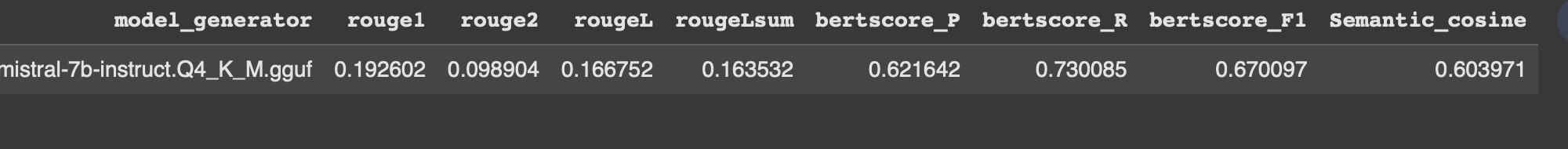
AI-generated content may be incorrect.

A black and white text

AI-generated content may be incorrect.

Metrics generator





1. Comparación de modelos
2. model\_configs = [
3. {
4. "name": "miniLM-semantico",
5. "faiss\_index\_path": "./faiss\_miniLM\_semantic",
6. "embeddings\_model": "all-MiniLM-L6-v2",
7. "splitter\_fn": lambda: SentenceTransformersTokenTextSplitter(chunk\_overlap=100, chunk\_size=1000),
8. },
9. {
10. "name": "miniLM-recursivo",
11. "faiss\_index\_path": "./faiss\_miniLM\_recursive",
12. "embeddings\_model": "all-MiniLM-L6-v2",
13. "splitter\_fn": lambda: RecursiveCharacterTextSplitter(chunk\_size=1000, chunk\_overlap=100),
14. },
15. {
16. "name": "mpnet-semantico",
17. "faiss\_index\_path": "./faiss\_mpnet\_semantic",
18. "embeddings\_model": "sentence-transformers/paraphrase-multilingual-mpnet-base-v2",
19. "splitter\_fn": lambda: SentenceTransformersTokenTextSplitter(chunk\_overlap=100, chunk\_size=1000),
20. },
21. {
22. "name": "mpnet-recursivo",
23. "faiss\_index\_path": "./faiss\_mpnet\_recursive",
24. "embeddings\_model": "sentence-transformers/paraphrase-multilingual-mpnet-base-v2",
25. "splitter\_fn": lambda: RecursiveCharacterTextSplitter(chunk\_size=1000, chunk\_overlap=100),
26. },
27. {
28. "name": "distiluse-semantico",
29. "faiss\_index\_path": "./faiss\_distiluse\_semantic",
30. "embeddings\_model": "sentence-transformers/distiluse-base-multilingual-cased-v2",
31. "splitter\_fn": lambda: SentenceTransformersTokenTextSplitter(chunk\_overlap=100, chunk\_size=1000),
32. },
33. {
34. "name": "distiluse-recursivo",
35. "faiss\_index\_path": "./faiss\_distiluse\_recursive",
36. "embeddings\_model": "sentence-transformers/distiluse-base-multilingual-cased-v2",
37. "splitter\_fn": lambda: RecursiveCharacterTextSplitter(chunk\_size=1000, chunk\_overlap=100),
38. },

A graph with different colored lines

AI-generated content may be incorrect.

A graph with colorful lines and text

AI-generated content may be incorrect.

A screenshot of a black screen

AI-generated content may be incorrect.

# A chart with numbers and colors AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.

# Fuentes:

Hernández-Sampieri, R., & Mendoza, C. (2023). Metodología de la investigación: Las rutas de la investigación cuantitativa, cualitativa y mixta. McGraw-Hill. <https://0-bc-vitalsource-com.biblioteca-ils.tec.mx/tenants/BIB_TECDEMTY/libraries?bookmeta_vbid=9786071520326>

Mukhiya, S., & Ahmed, U. (2020). Hands-On Exploratory Data Analysis with Python: Perform EDA Techniques to Understand, Summarize, and Investigate Your Data. Packt Publishing.

Visengeriyeva, L., Kammer, A., Bär, I., Kniesz, A., & Plöd, M. (2023). CRISP-ML(Q): The ML Lifecycle Process. INNOQ. <https://ml-ops.org/content/crisp-ml>

**MiCoto.** (2024). Manual de Usuario de la plataforma MiCoto. Documento interno (PDF).

**MiCoto.** (2024). Descripción funcional de MiCoto (dictado). Documento interno (PDF).

**MiCoto.** (2024). Base de datos de mensajes de usuarios (archivo CSV interno). Exportado desde la plataforma MiCoto para fines de análisis.

**MiCoto.mx.** (s.f.). *¿Quiénes somos?* Recuperado el 01/5/2025 de: <https://micoto.mx/#quienes-somos>