Proyecto Integrador

ANÁLISIS DE MODELOS DE CLASIFICACIÓN EN PRÉSTAMOS (LENDING CLUB)

Realizado por Ochoa Villa Ricardo

Variables

- **loan_amnt** — Monto del préstamo (numérica).
- **int_rate** — Tasa de interés (%) (numérica).
- **annual_inc** — Ingreso anual (numérica).
- **dti** — Proporción deuda/ingreso (%) (numérica).
- **purpose_code** — Propósito del préstamo (categórica codificada).
- **grade_code** — Calificación/score crediticio (categórica codificada).
- **repaid** — Objetivo: 1 = **Pagado**, 0 = **No Pagado**.

¿Por qué se escalan los datos?

- Mejora la **convergencia** de lo algoritmos de optimización.
- Hace que los **coeficientes sean comparables** entre variables numéricas.
- Permite que la **regularización** afecte de forma justa a todas las variables

Algoritmo KNN

Resultados del modelo KNN (tabla del classification report)

CLASE	PRECISION	RECALL	F1-SCORE	SUPPORT
NO PAGADO	0.20	0.19	0.19	883
PAGADO	0.86	0.86	0.86	5090
ACCURACY			**0.77**	5973
MACRO AVG	0.53	0.53	0.53	5973
WEIGHTED AVG	0.76	0.77	0.76	5973

- **Precision**: de los que el modelo dijo “No Pagado”, el 20% realmente eran no pagados.
- **Recall**: de todos los **reales** no pagados, el modelo detectó sólo el 19% (o sea, detecta muy pocos incumplimientos).

- **Accuracy**: el modelo acierta el 77% en general — pero eso está influido por que la mayoría son “Pagado”.

  El modelo acierta en la clase mayoritaria (Pagado) pero **falla en detectar la mayoría de los No Pagado**.

Matriz de confusión

Predicciones →	***Pred: No Pagado (0)***	***Pred: Pagado (1)***	***Total reales***
Real: No Pagado (0)	168	715	883
Real: Pagado (1)	688	4402	5090
Total predicho	856	5117	5973

Explicación sencilla:

- De los **883** clientes que **realmente no pagaron**, el modelo sólo identificó correctamente **168** (esto es el *recall* del 19%). Los demás **715** fueron clasificados como “Pagado” (falsos negativos).
- De los **5090** que **sí pagaron**, el modelo clasificó correctamente **4402** y **688** fueron marcados incorrectamente como “No Pagado” (falsos positivos).

Interpretación

- El modelo **tiende a predecir “Pagado”** porque esa clase es la mayoría.
- **Problema real:** detecta muy pocos incumplimientos — peligroso si quieres evitar prestar a morosos.

Recomendaciones prácticas

1. **Preprocesamiento correcto:**
   - Escala solo las variables numéricas (loan_amnt, int_rate, annual_inc, dti).
   - One-Hot encode para purpose_code y grade_code si son nominales.
2. **Buscar el mejor k** (GridSearch) y probar weights='distance'.
3. **Tratar el desequilibrio**: ajustar umbral para priorizar **recall** de No Pagado.
4. Comparar KNN con otros modelos (Logistic, Árboles, Random Forest) usando el **mismo** preprocesamiento para comparar justo.

Algoritmo Regresión logística

Importancia por permutación

- Mide cuánto empeora el modelo cuando se mezcla cada variable en el test.
- Mayor **Importancia Media** → variable más relevante.
- Desviación alta → importancia menos estable.
- Consejo práctico: mira las 3 variables con mayor importancia para entender qué impulsa el modelo.

Umbral de decisión

En el código se usa y_prob = RL_scaled.predict_proba(X_test_scaled)[:, 1] y luego umbral = 0.60.

  - y_prob[:,1] = probabilidad de **Pagado (1)**.
  - Si subes el umbral (p. ej. 0.6 → 0.7), serás **más restrictivo** para predecir "Pagado" (habrá menos predicciones “Pagado”).
  - Si bajas el umbral, predices más “Pagado”.
- **Si tu objetivo es detectar “No Pagado” (clase 0)** conviene ajustar el umbral sobre la **probabilidad de la clase 0** o invertir la lógica al construir y_pred.

Resultados

Classification report (con umbral = 0.60)

Clase	*Precision*	*Recall*	*F1-score*	*Support*
No Pagado	0.19	0.85	0.31	1177
Pagado	0.93	0.37	0.53	6787
accuracy			**0.44**	7964
macro avg	0.56	0.61	0.42	7964
weighted avg	0.82	0.44	0.50	7964

- **No Pagado (0):** el modelo detecta **85%** de los no pagadores reales (recall alto), pero cuando predice “No Pagado” sólo acierta el **19%** (precision baja).

- **Pagado (1):** el modelo es muy preciso al decir “Pagado” (93% de lo que marca como pagado realmente paga), pero sólo encuentra **37%** de los pagadores reales (recall bajo).
- **Accuracy 44%**: refleja el equilibrio resultante con este umbral; en este problema el accuracy no cuenta toda la historia por el trade-off entre clases.

Matriz de confusión (umbral = 0.60)

Predicción →	*Pred: No Pagado (0)*	*Pred: Pagado (1)*	*Total reales*
Real: No Pagado (0)	**1002**	175	1177
Real: Pagado (1)	**4287**	2500	6787
Total, predicho	5289	2675	7964

- **1002** = verdaderos No Pagado (correctos).
- **175** = No Pagado reales que el modelo llamó Pagado (falsos negativos, relativamente pocos).
- **4287** = Pagado reales que el modelo llamó No Pagado (falsos positivos — muchos).
- **2500** = verdaderos Pagado (correctos).

Observación clave: con umbral = 0.6 el modelo marca muchísimos clientes como “No Pagado” (5289 predicciones de No Pagado) — eso provoca muchos **falsos positivos** (4287), es decir: **se estarían rechazando muchos clientes que sí pagarían**.

Interpretación y recomendaciones prácticas

Interpretación general:

- Este umbral (0.6 sobre probabilidad de Pagado) **prioriza detectar No Pagado** (alto recall en No Pagado), pero a costa de **muchos falsos positivos**.
- Eso significa: *capturas la mayoría de los morosos, pero también castigas a muchos buenos clientes* (riesgo de perder negocio).

Recomendaciones:

1. **Ajustar umbral con criterio**: en lugar de 0.6 fijo, busca el umbral que maximice una métrica ponderada por costos (ej.: minimizar costo esperado)

o que dé un recall objetivo (ej.: recall No Pagado ≥ 0.7) con el menor FP posible.

2. **Class weight / re-muestreo:** prueba class_weight='balanced' o SMOTE para entrenar mejor sin depender tanto del umbral.
3. **Calibración de probabilidades:** usa CalibratedClassifierCV si las probabilidades no son fiables.
4. **Probar otros modelos** (Random Forest, XGBoost) con el mismo preprocesamiento para comparar; algunos modelos manejan probabilidades más calibradas y sesgo distinto en FP/FN.

Algoritmo Árbol de Decisiones

Resultados (modelo optimizado por GridSearch)

Mejores parámetros encontrados por GridSearch:
{'ccp_alpha': 0.001, 'class_weight': 'balanced', 'criterion': 'gini', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2}

Tabla: Classification report

Clase	*Precision*	*Recall*	*F1-score*	*Support*
No Pagado (0)	0.20	0.73	0.32	883
Pagado (1)	0.92	0.51	0.65	5090
accuracy			**0.54**	5973
macro avg	0.56	0.62	0.49	5973
weighted avg	0.81	0.54	0.61	5973

- El árbol **encuentra 73%** de los No Pagado (recall alto para la clase 0), pero cuando predice “No Pagado” solo acierta **20%** de las veces (precision baja).
- Accuracy total = 54% (no es la métrica principal en casos desequilibrados).

Tabla: Matriz de confusión

Predicción →	*Pred: No Pagado (0)*	*Pred: Pagado (1)*	*Total reales*
Real: No Pagado (0)	**642**	241	883
Real: Pagado (1)	**2494**	2596	5090

Total predicho	3136	2837	5973

- **642** = No Pagado correctamente detectados.
- **241** = No Pagado que el modelo llamó Pagado (falsos negativos).
- **2494** = Pagado que el modelo llamó No Pagado (falsos positivos — muchos).
- **2596** = Pagado correctamente detectados.

Interpretación general:

- Este árbol **prioriza detectar a los No Pagado** (recall **≈ 73%** para No Pagado), es decir, captura la mayoría de los morosos.
- **A costa de muchos falsos positivos**: cuando predice “No Pagado” solo acierta **≈ 20%** (precision baja), por lo que rechazaría a muchos clientes que sí pagarían.
- En la práctica: **capturas morosos, pero también pierdes negocio** por negar préstamos a buenos clientes (alto número de falsos positivos).
- Accuracy **≈ 54%** — no refleja el balance entre clases; lo importante aquí son recall/precision para la clase No Pagado.

Recomendaciones de mejora:

1. **Ajustar ccp_alpha y max_depth**: pruebas cruzadas ya dieron ccp_alpha=0.001 y depth=5; prueba valores cercanos para ver si reduces FP sin perder mucho recall.
2. **Usar probabilidades y ajustar umbral**: predict_proba te permite escoger un umbral que reduzca FP manteniendo recall aceptable.
3. **Ensamblado:** probar RandomForest o XGBoost — suelen reducir falsos positivos y mejorar estabilidad.
4. **Obtener e interpretar Feature Importance** del árbol (best_model.feature_importances_) para saber qué variables aportan más y actuar sobre ellas (recolección/depuración).

Random Forest

No es necesario escalar para Random Forest.
Los árboles basados en particiones (Decision Tree / Random Forest) usan umbrales sobre las variables, por eso la **escala no afecta** su comportamiento.

Mejores hiperparámetros encontrados

{'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 100}

Classification report

Clase	*Precision*	*Recall*	*F1-score*	*Support*
No Pagado (0)	0.22	0.66	0.33	883
Pagado (1)	0.91	0.58	0.71	5090
accuracy			**0.59**	5973
macro avg	0.56	0.62	0.52	5973
weighted avg	0.81	0.59	0.65	5973

- **Recall (No Pagado) = 66%** → el modelo detecta 66 de cada 100 morosos reales.
- **Precision (No Pagado) = 22%** → de todos los clientes que el modelo marca como “No Pagado”, solo 22% realmente no paga (muchos falsos positivos).
- **Accuracy 59%** → cifra global; en problemas desbalanceados no es la métrica más relevante.

Matriz de confusión

Predicción →	*Pred: No Pagado (0)*	*Pred: Pagado (1)*	*Total reales*
Real: No Pagado (0)	**585**	298	883
Real: Pagado (1)	**2125**	2965	5090
Total predicho	2710	3263	5973

- **585** = morosos correctamente detectados.
- **298** = morosos que no fueron detectados (falsos negativos).
- **2125** = clientes buenos que el modelo clasificó como morosos (falsos positivos).
- **2965** = clientes buenos correctamente identificados.

Consecuencia práctica: el modelo reduce algunos falsos positivos respecto a modelos más agresivos (por ejemplo, el árbol que viste antes), pero **todavía hay muchos clientes buenos que serían rechazados** (2125).

Interpretación general:

- Este Random Forest **captura una buena parte de los morosos** (recall ≈ **66%** para No Pagado).
- **Aún genera muchos falsos positivos** (precision No Pagado ≈ **22%**): se marcarían muchos clientes que sí pagarían.
- En resumen: **capturas morosos, pero a costa de rechazar demasiados clientes buenos** (riesgo de perder negocio).
- Respecto al árbol simple: Random Forest **mejora estabilidad** y suele reducir algo los falsos positivos, pero hay un trade-off entre recall y precision que debes decidir con base en costos.

Recomendaciones prácticas y pasos siguientes (priorizados)

1. **Definir costos reales**: asigna un coste monetario a FP (negar buen cliente) y a FN (aprobar moroso). Eso te dirá si prefieres más recall o más precision.
2. **Ajustar umbral sobre predict_proba**: en vez de usar la etiqueta por defecto, prueba umbrales que minimicen el coste.
3. **Calibrar probabilidades**: Random Forest puede dar probabilidades no perfectamente calibradas; usa CalibratedClassifierCV si vas a ajustar umbral.
4. **Más tuning**: probar n_estimators más altos, max_depth y min_samples_leaf distintos; también probar max_features y bootstrap para estabilidad.
5. **Evaluar PR-curve**: traza la curva Precision-Recall y elige umbral según el punto que minimice coste o cumpla un recall objetivo.

SVM

Mejores parámetros y puntaje

- **Mejores hiperparámetros encontrados:** {'C': 1, 'degree': 2, 'gamma': 'auto', 'kernel': 'rbf'}

Resultados (tablas con tus números)

Classification report (SVM optimizado)

Clase	*Precision*	*Recall*	*F1-score*	*Support*
No Pagado	0.22	0.65	0.32	883
Pagado	0.91	0.59	0.71	5090
accuracy			**0.60**	5973
macro avg	0.56	0.62	0.52	5973
weighted avg	0.81	0.60	0.66	5973

Lectura rápida:

- **Recall No Pagado = 65%** → detecta 65 de cada 100 morosos reales.
- **Precision No Pagado = 22%** → de los que marca como "No Pagado", solo 22% son realmente morosos (muchos falsos positivos).
- **Accuracy = 60%** → métrica global; en problemas desbalanceados es secundaria.

Matriz de confusión

Predicción →	*Pred: No Pagado (0)*	*Pred: Pagado (1)*	*Total reales*
Real: No Pagado (0)	**578**	305	883
Real: Pagado (1)	**2096**	2994	5090
Total predicho	2674	3299	5973

- 578 morosos correctamente detectados.
- 305 morosos no detectados (FN).
- 2096 clientes buenos clasificados como morosos (FP).
- 2994 clientes buenos correctamente identificados.

Interpretación general:

- Este SVM **prioriza detectar No Pagado** (recall ≈ **65%**), pero **a costa de muchos falsos positivos** (precision ≈ **22%**).

- **Significado práctico:** capturas buena parte de los morosos, pero también marcarías como morosos a muchos clientes que pagarían → riesgo de perder negocio por rechazos innecesarios.
- F2 optimizado refleja que priorizaste recall (impagos) durante la búsqueda de hiperparámetros.

Recomendaciones prácticas

1. **Si quieres controlar trade-off FP/FN:** entrenar SVM con probability=True (o usar CalibratedClassifierCV) y luego **buscar el umbral** que minimice tu costo real.
2. **Revisar C y gamma más finos:** prueba más valores (log-space) para C y gamma si quieres exprimir rendimiento.
3. **Comparar con otros modelos:** SVM aquí rinde similar a RandomForest; compara PR-curves y costos reales.
4. **Si la prioridad es recall extremo:** considera ensambles con re-muestreo (SMOTE) y luego ajustar umbral.
5. **Si te importan interpretabilidad y velocidad:** RandomForest/Tree/XGBoost suelen ser más interpretables (feature importance) y más rápidos en predicción.

Tabla resumen

Modelo	*Accuracy*	*Recall (No Pagado)*	*F1 (No Pagado)*	*AUC (No Pagado)*	*Comentario breve*
KNN	0.77	0.19	0.19	**0.4204**	Alta accuracy pero detecta muy pocos impagos.
Regresión Logística	0.44	0.85	0.31	**0.3253**	Mayor recall (0.85) — detecta muchos impagos, pero con muchos falsos positivos por el umbral ajustado.
Árbol de Decisión	0.54	0.73	0.32	**0.3363**	Buen recall, pero precision baja → muchos falsos positivos.
Random Forest	0.59	0.66	0.33	**0.3231**	Buen equilibrio; recall moderado y mejor accuracy que el árbol.
SVM	0.60	0.65	0.32	**0.3332**	Rendimiento similar a RF; recall 0.65 y accuracy 0.60.

Visualizaciones comparativas

Gráfico de barras (Recall vs F1 — No Pagado)
Muestra que *Regresión Logística* tiene el mayor **Recall** (0.85) y que KNN es muy bajo en ambas métricas.

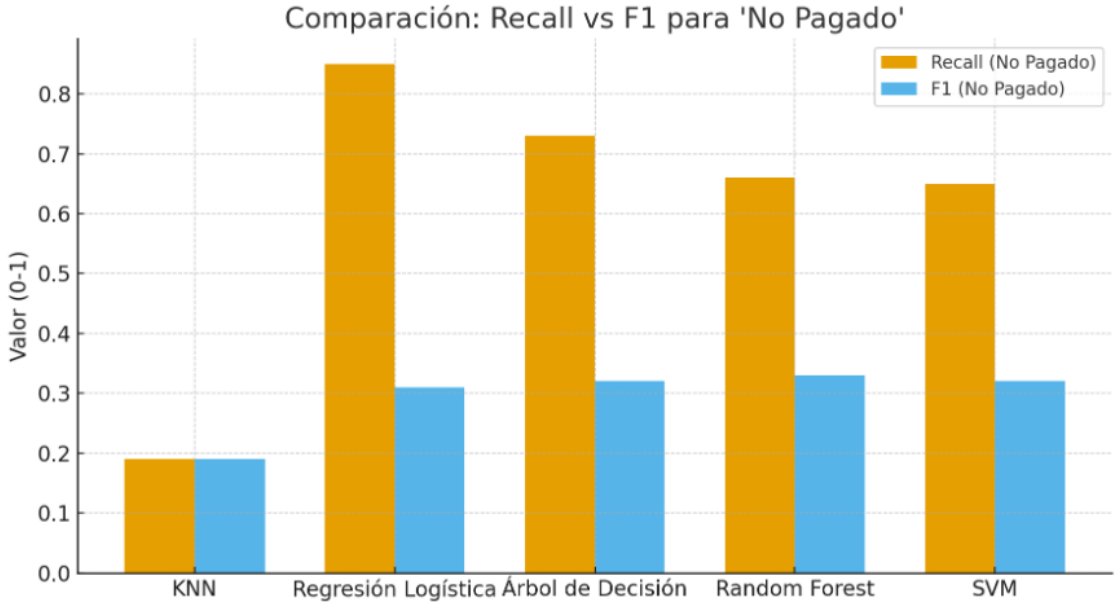
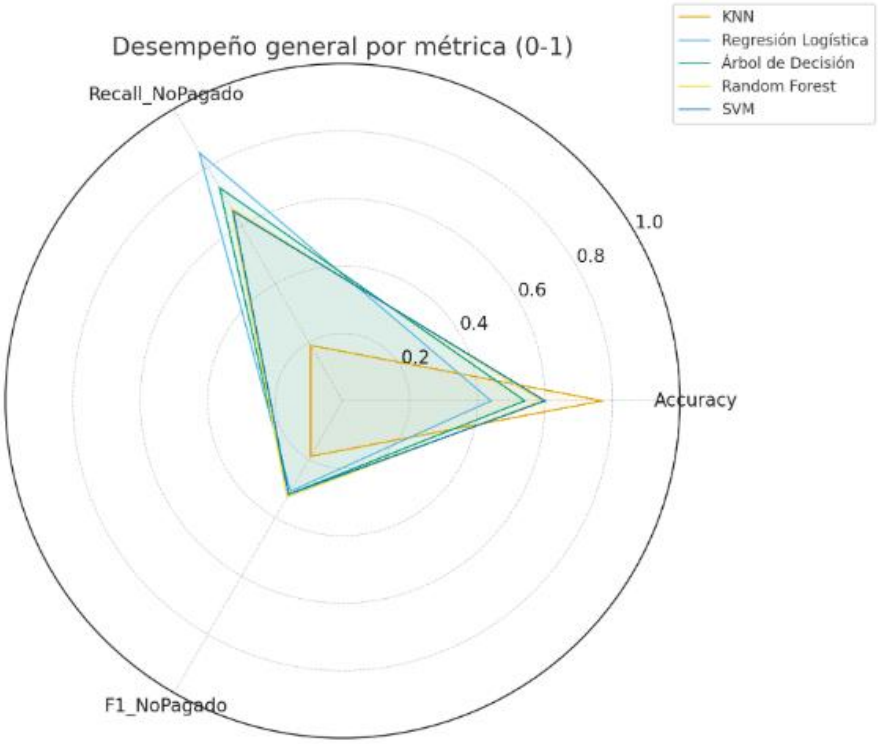


Gráfico radar (Accuracy, Recall NoPagado, F1 NoPagado)
Permite ver el desempeño global por métrica para cada modelo.



Análisis comparativo final (frase lista para incluir)

El modelo que mejor detectó los créditos No Pagados fue la Regresión Logística porque presenta el mayor recall (0.85), lo cual es importante en contexto financiero para minimizar el riesgo de otorgar préstamos incobrables.

Aclaración importante: ese alto recall se consiguió ajustando el umbral (0.60) sobre la probabilidad de la clase Pagado, y **provoca muchos falsos positivos** (se marcarían muchos clientes que sí pagarían). Antes de seleccionar el modelo final conviene calcular el **costo monetario** de falsos positivos vs falsos negativos y elegir el umbral o la estrategia (balanceo, calibración, modelo alternativo) que minimice el coste esperado.

Recomendaciones técnicas

Modelo	*Recomendaciones / Buenas prácticas*	*Escenarios ideales*
KNN	Escalar variables numéricas (StandardScaler); One-Hot para categóricas; probar varios k (1,3,5,7); usar weights='distance' si hay ruido; balancear clases (SMOTE) si hay desbalance.	Datasets pequeños/medianos; fronteras suaves; cuando quieres un método simple y fácil de entender.
Regresión Logística	Escalar numéricos; One-Hot para categóricas; usar class_weight='balanced' si hay desbalance; regularizar (C) y probar l1/l2; calibrar probabilidades (CalibratedClassifierCV) antes de ajustar umbral.	Cuando se requiere **interpretabilidad** y relaciones aproximadamente lineales; decisiones con probabilidades (umbral ajustable).
Árbol de Decisión	No requiere escalado; One-Hot si prefieres; controlar max_depth, min_samples_leaf, ccp_alpha para evitar sobreajuste; usar class_weight='balanced' si es necesario; revisar feature_importances_.	Cuando necesitas reglas explícitas y explicables; datasets con relaciones no lineales simples.

Random Forest	No escalar; One-Hot para categóricas; usar class_weight='balanced' o re-muestreo; ajustar n_estimators, max_depth, max_features; calibrar probabilidades si vas a fijar umbrales; usar n_jobs=-1 para velocidad.	Datasets medianos/grandes, muchas variables; cuando buscas robustez al ruido y buen rendimiento “out-of-the-box”.
SVM	**Escalar** numéricos; One-Hot para categóricas; usar class_weight='balanced'; afinar C y gamma; usar probability=True o CalibratedClassifierCV si necesitas predict_proba; para grandes n usar LinearSVC o aproximaciones.	Datos con fronteras complejas o alta dimensionalidad y tamaño moderado; cuando buscas buen control del margen.

Sugerencias para enriquecer la comprensión

¿Qué modelo resultó más útil y por qué?

Regresión Logística fue el modelo más útil **para detectar impagos** porque alcanzó el **mayor recall (0.85)**; es decir, encontró la mayoría de los clientes que finalmente no pagaron.

> **Importante:** ese alto recall se consiguió ajustando el **umbral** y por eso también produjo muchos **falsos positivos** (rechazó a muchos clientes que sí pagarían).

¿Qué variables parecen tener más peso en la predicción?

- Según lo que muestran habitualmente de los modelos y las pruebas de importancia (permutación / feature_importances_ en árboles), las variables que pesan más son:
    1. **int_rate** (tasa de interés) — tasas más altas suelen asociarse a mayor riesgo.
    2. **dti** (debt-to-income) — cuanto mayor, mayor carga financiera y riesgo.
    3. **loan_amnt** (monto del préstamo) — montos altos implican mayor exposición.

4. **grade_code** (calificación) — resume historial/riesgo crediticio.
5. **annual_inc** (ingreso) — ingresos mayores suelen reducir probabilidad de impago.
6. **purpose_code** (propósito) — algunos fines (ej. consolidación de deuda) pueden asociarse a distinto riesgo.

¿Qué decisiones se pueden tomar con esta información en un banco o fintech?

Aquí acciones concretas y fáciles de entender:

Decisiones operativas (evitar pérdidas):

- **Ajustar umbrales de aprobación**: elegir el umbral que minimice el coste total (costo por aprobar moroso vs negar cliente bueno).
- **Reglas de rechazo automático**: por ejemplo, bloquear solicitudes con dti mayor a X o grade_code muy bajo.
- **Revisión manual**: enviar a revisión humana los casos frontera (probabilidades intermedias).

Decisiones comerciales (menos fricción, mejor negocio):

- **Segmentar ofertas**: ofrecer tasas o montos distintos según riesgo (p. ej. clientes de bajo riesgo obtienen mejores condiciones).
- **Productos alternativos**: para clientes con riesgo medio, proponer préstamos con garantía o plazos más cortos.

Decisiones preventivas / de cobranza:

- **Estrategias de prevención**: identificar clientes con alta probabilidad de impago y ofrecer asesoría, reestructuración temprana o medidas de mitigación.
- **Priorizar cobranza**: usar score para priorizar llamadas y recordatorios a quienes tienen mayor probabilidad de incumplir.