Universidad Torcuato Di Tella
Maestría en Economía & Finanzas
Segundo Trimestre 2025

Machine Learning para Finanzas
Problem Set 3

Alumna: Paola Nuñez | Profesor: Lionel Modi

---

1 Introducción

Este trabajo presenta la clase `AnalistaDeRiesgo`, que implementa un flujo para estimar la probabilidad de default (PD) de bancos y cuantificar los activos en riesgo del sistema. Se usan datos históricos etiquetados de la FDIC para entrenar y seleccionar modelos, y un panel de bancos *actuales* para inferir la probabilidad de default y marcar entidades en riesgo según distintos criterios de umbral. La implementación sigue PEP 8 y docstrings estilo NumPy.

2 Estructura del proyecto

El código fuente completo está disponible en el siguiente repositorio de GitHub: `https://github.com/nunezpaola/MLforfinance`. Desde la carpeta raíz, la resolución del Problem Set 3 está organizada como sigue:

- `ps/ps3.py`: clase `AnalistaDeRiesgo`. Incluye el preprocesamiento, selección por `GridSearchCV`, elección de umbral para definir cuándo una probabilidad es lo *suficientemnte alta* como para indicar riesgo de default y cálculo de métricas (entidades y activos en riesgo).
- `ps/ps3_test.py`: módulo de test. Carga datos, define modelos y grillas, corre la clase un loop de métodos de umbral y genera gráficos.
- `ps/ps3_logs.txt`: logs INFO en terminal obtenidos al correr el módulo de test
- `mlfin/plotting.py`: utils para gráficos.
- `mlfin/printing.py`: helpers de logging.
- `data/central_bank_data.h5`: base de datos con información histórica y actual de entidades financiera usada para entrenar el modelo.

3 Documentación

3.1 Clase `AnalistaDeRiesgo`

La clase se disponibiliza en el archivo `ps/ps3.py` y cuenta con métodos públicos y privados:

Constructor `__init__(modelos_y_grids, scoring="roc_auc", random_state=42).` Toma como inputs una lista de pares (estimador, grilla), métrica para cross-validation (CV) y semilla.

`load_data(df_train, default_col).` Valida y almacena el `pd.dataFrame` histórico etiquetado y el nombre de la columna objetivo (`defaulter`).

`_make_preprocessor(features, scale).` Pipeline numérico con `SimpleImputer(strategy="median")` y, si corresponde (SVC/KNN), `StandardScaler`. Se inserta en un `ColumnTransformer`.

`_wrap_grid(grid, "clf").` Prefija los hiperparámetros con `"clf__"` para usarlos dentro del `Pipeline` de scikit-learn.

`_fit_and_select(X, y, features).` Para cada modelo: construye `Pipeline`, hace `GridSearchCV` (5x5, `scoring="roc_auc"`), guarda el mejor y devuelve el de mayor `score`.

`pick_threshold(y_true, y_proba, method, ...).` Selector de umbral:

- **fixed:** umbral fijo (por defecto 0.5). Se elige si resulta indiferente cuántas/cuales son las entidades que defaultean y no incorpora el tradeoff entre subestimar un default (riesgo sistémico) o sobreestimar uno.
- **youden:** maximiza la cantidad de verdaderos positivos respecto de los falsos (TPR − FPR) en la ROC.
- **f1:** maximiza F1 (balance precisión/recall).
- **cost:** minimiza pérdida esperada según el costo de un falso positivo y el costo de un máximo negativo(C_{FP}, C_{FN}) y permite diferenciar entre estos. En este trabajo, además se pondera por *activos actuales*.
- **target_rate:** se utiliza en caso de querer estimar riesgo sistémico. Es decir, estamos pensando que, después de que defaulteen cierta cantidad (es decir, target_rate, en puntos porcentuales) de entidades se puede generar riesgo sistémico y buscamos la cantidad de activos a cubrir en tal caso.

`get_report(df_predict, features, umbral_riesgo, threshold_kwargs).`

1. Selecciona el mejor `Pipeline` por CV sobre histórico.
2. Predice probabilidad de default en los bancos *actuales*.
3. Elige umbral:
   - **fixed**: usa el valor numérico.
   - **cost**: minimiza $\sum_i \left[\mathbb{1}_{p_i \geq t}(1-p_i)C_{FP} + \mathbb{1}_{p_i<t} p_i C_{FN}\right] \cdot \text{activos}_i$ con p_i de *actuales*.
   - **target_rate**: percentil de p_i de *actuales*.
   - **youden/f1**: calcula el umbral sobre histórico (predicciones in-sample).
4. Etiqueta riesgo ($p_i \geq t$) y computa:
   - % de entidades marcadas.
   - % de activos en riesgo y **activos en riesgo (USD B)**.
   - Total de activos del sistema (USD B).
5. Devuelve un diccionario con nombre del modelo ganador, AUC de CV, hiperparámetros, umbral usado y métricas.

3.2 Pruebas y validación

El módulo de test se encuentra en `ps/ps3_test.py`. A continuación, se detalla el proceso que se lleva a cabo:

- Carga `df_train` (`bank_defaults_FDIC`) y `df_predict` (`regulated_banks`).
- Define 3 clasificadores y sus grillas: `KNN`, `SVC` y `DecisionTree`, con las grillas especificadas en el enunciado.
- Corre un loop para evaluar los métodos de umbral: `fixed, youden, f1, cost, target_rate` (con $C_{FN} : C_{FP} = 20 : 1$ y `target_rate` de 10%).
- Gráficos:
  1. **ROC** (obtenido de `mlfin.plotting.plot_roc_curve`).
  2. **Calibración** in-sample.
  3. **Importancia de features**
  4. **Histograma de probabilidades de default estimadas** en entidades financieras actuales con líneas de corte por método.

3.3 Variables utilizadas

En la estimación se emplean los siguientes regresores:

Columna	Descripción
`log_TA`	Logaritmo de activos totales (tamaño del banco).
`NI_to_TA`	Resultado neto / TA (rentabilidad).
`Equity_to_TA`	Patrimonio / TA (capitalización).
`NPL_to_TL`	Préstamos en mora / TL (calidad de cartera).
`REO_to_TA`	Real estate owned / TA (activos adjudicados).
`ALLL_to_TL`	Provisiones por pérdidas / TL.
`core_deposits_to_TA`	Depósitos "core" (estables) / TA.
`brokered_deposits_to_TA`	Depósitos intermediados / TA (volátiles).
`liquid_assets_to_TA`	Activos líquidos de alta calidad / TA.
`loss_provision_to_TL`	Carga de provisiones del período / TL.
`NIM`	Net Interest Margin (margen financiero).
`assets_growth`	Crecimiento % de activos (variación).

Notas: TA = *Total Assets*, TL = *Total Loans*.

4 Resultados

El modelo ganador por validación cruzada es **SVC (RBF, C=10)**, con $\text{AUC}_{CV} \approx 0.986$. La calibración insample es globalmente buena (Fig. 1), con leve sobreestimación en probabilidades muy bajas y leve subestimación en la cola alta.

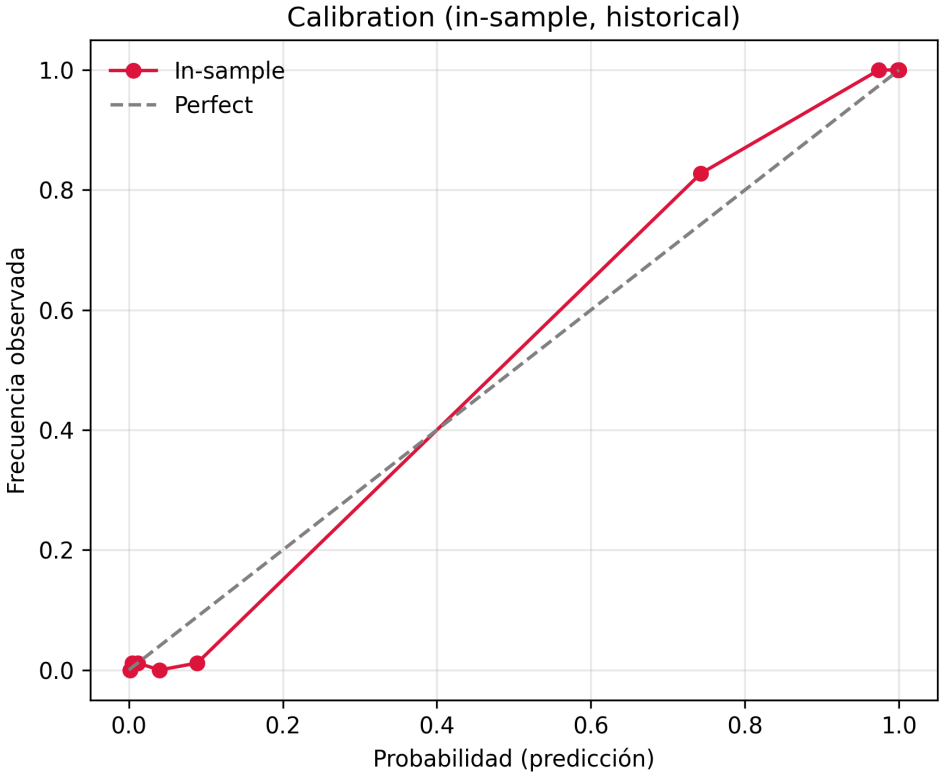

Figure 1: Curva de calibración (insample, histórico). La línea ideal (gris) indica que la estimación de probabilidad de default es igual a la observada.

En términos de *drivers*, la Fig. 2 muestra que el tamaño (`log_TA`) domina ampliamente, seguido por `NPL_to_TL` y `Equity_to_TA`. El resto de ratios tienen incidencia virtualmente baja. Por otro lado, en la Figura 3 se reporta la distribución de la estimación de probabilidades de default y los distintos *thresholds evaluados*.

Los resultados numéricos por método de umbral son:

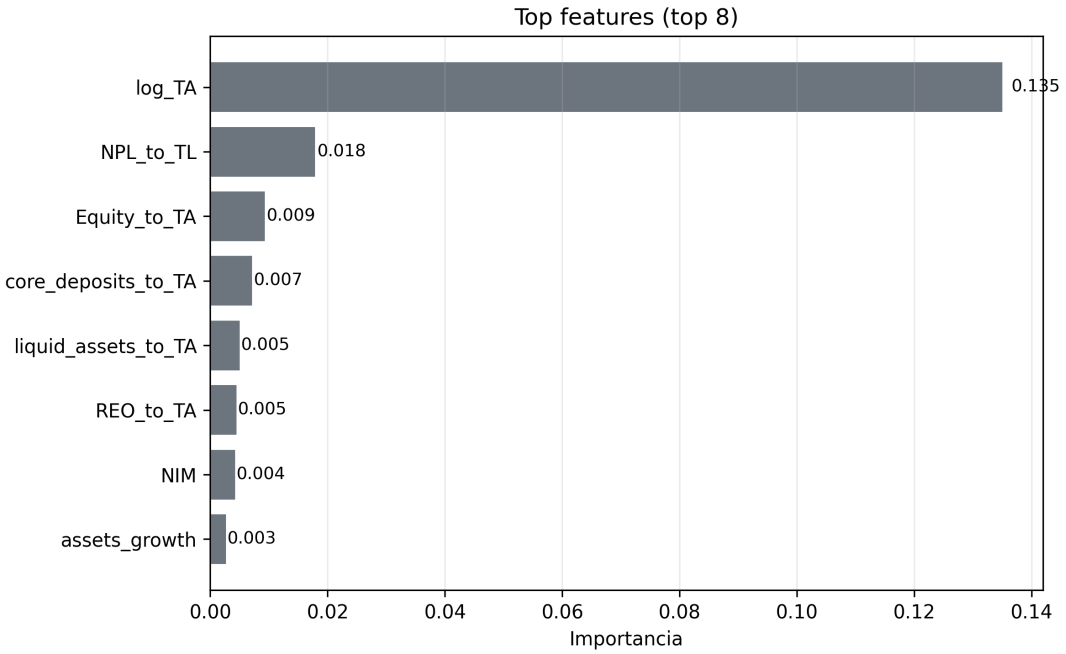

Figure 2: Importancias (top 8). El tamaño explica gran parte de la separabilidad; calidad de cartera y capital le siguen.

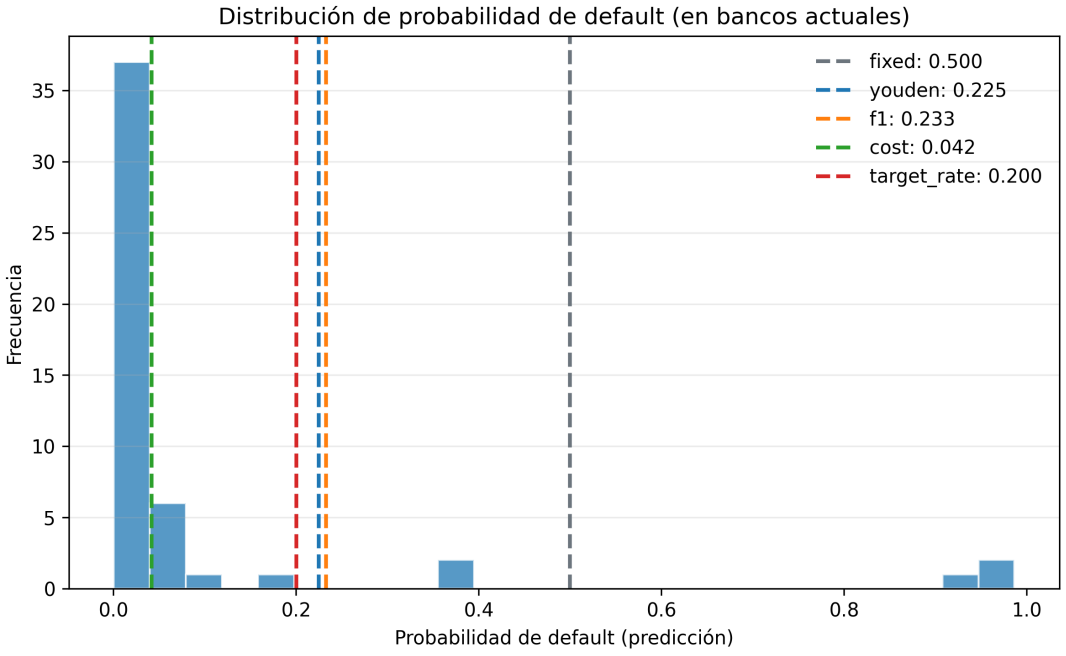

Figure 3: Distribución de PD en bancos actuales con umbrales por método.

Método	AUC (CV)	Umbral	% Ent. en riesgo	% Activos en riesgo	Activos en riesgo (USD B)
fixed (0.50)	0.986	0.500	6.0%	0.09%	0.00
youden	0.986	0.229	10.0%	57.27%	0.68
f1	0.986	0.222	10.0%	57.27%	0.68
cost (20:1)	0.986	0.036	26.0%	61.55%	0.73
target_rate 10%	0.986	0.217	10.0%	57.27%	0.68

Recordando que el total de activos del sistema es ∼USD 1.19 B, se desprenden dos conclusiones:

- **fixed 0.5**. Indica que el 6% de entidades tiene probabilidad de default *lo suficientemente alta* y captura apenas 0.09% de los activos totales bajo administración. Este método no internaliza ningún tradeoff entre falsos y verdaderos positivos, reportando una menor practicidad a la hora de tomar decisiones de política monetaria. Consecuentemente, podemos sostener que este umbral es muy alto para el riesgo sistémico que puede generar un default.
- **youden/f1/target_rate(10%)** reportan que el 10% de las entidades tienen probabilidades de default estimadas superiores al umbral de corte, y esto se corresponde con el 57.3% de los activos bajo administracion. Lo que implica un riesgo sistémico mucho más costoso de prevenir.

Interpretación. (i) El clasificador *rankea* muy bien (AUC alto) y las probabilidades estimadas son razonables (calibración cerca de la diagonal), con dos sesgos muestrales: conservador en la cola baja (prefiere falsas alarmas pequeñas) y algo optimista en la cola alta (probabilidad ligeramente subestimada). (ii) la importancia de las variables `log_TA` y `NPL_to_TL` sugiere que *tamaño* y *calidad de cartera* explican la mayor parte de la separación de riesgo. (iii) La concentración es extrema: con sólo 10% de entidades se cubre >50% de los activos totales en gestión.

References

[1] NumPy Documentation Guide, "Docstring Standard", `https://numpydoc.readthedocs.io/en/latest/format.html`. Último acceso: Julio 2025.

[2] PEP 8 – Style Guide for Python Code, `https://www.python.org/dev/peps/pep-0008/`. Último acceso: Julio 2025.