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

Machine Learning para Finanzas
Problem Set 2

Alumna: Paola Nuñez
Profesor: Lionel Modi

---

1 Introducción

Este trabajo presenta la clase `BalancedPortfolio`, que implementa un esquema de selección de portafolio basado en el modelo de cinco factores de Fama–French [1]. Para estimar las exposiciones de cinco activos se emplea una regresión Lasso entrenada con el conjunto de datos de 2017. A continuación, se busca minimizar el error cuadrático medio (MSE) respecto a una exposición uniforme entre factores, resolviendo el problema de optimización mediante dos enfoques: búsqueda en grilla y algoritmo de punto interior. La implementación cumple con las convenciones de docstrings de NumPy y sigue el estilo PEP 8.

1.1 Modelo de cinco factores de Fama–French

La versión extendida del CAPM propuesta por Fama y French (2015) añade cuatro factores adicionales al exceso de retorno del mercado. La ecuación de valuación para el activo i en la fecha t es:

$$R_{it} - R_{ft} = \alpha_i + \beta_i \left(R_{mt} - R_{ft}\right) + s_i \,\mathrm{SMB}_t + h_i \,\mathrm{HML}_t + r_i \,\mathrm{RMW}_t + c_i \,\mathrm{CMA}_t + \varepsilon_{it}\,. \tag{1}$$

donde:

- $R_{it} - R_{ft}$ es el exceso de retorno del activo i sobre la tasa libre de riesgo R_{ft}.
- $R_{mt} - R_{ft}$ es el exceso de retorno del mercado.
- SMB_t (*Small Minus Big*) mide la prima de tamaño: la diferencia de retorno entre carteras de acciones pequeñas y grandes.
- HML_t (*High Minus Low*) mide la prima de valor: la diferencia de retorno entre carteras de alto y bajo book-to-market.
- RMW_t (*Robust Minus Weak*) mide la prima de rentabilidad: la diferencia de retorno entre carteras de firmas con alta y baja rentabilidad operativa.
- CMA_t (*Conservative Minus Aggressive*) mide la prima de inversión: la diferencia de retorno entre carteras de firmas con baja y alta tasa de inversión.
- α_i es el intercepto, que se restringe a cero bajo la hipótesis de mercados eficientes.
- ε_{it} es el término de error.

En la formulación teórica estos cinco "factores" no son variables de estado observables, sino *portfolios miméticos* ("mimicking portfolios") que replican exposiciones a variables ocultas. Junto al portafolio de mercado y al activo libre de riesgo, dichos portfolios abarcan el conjunto multifactorial eficiente. La Ecuación 1 sugiere así la construcción de factores que permitan captar el efecto de esas variables de estado en los retornos esperados *sin* identificarlas directamente.

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 2 está organizada como sigue:

- `ps/ps2.py`: Implementa la clase `BalancedPortfolio`, que incluye la regresión Lasso y los dos métodos de resolución numérica (búsqueda en grilla con `get_allocations` y algoritmo de punto interior).
- `ps/ps2_test.py`: Script principal de testing: importa los datos, calibra el modelo, reporta ponderaciones y exposiciones, y genera los gráficos usados en este informe.
- `ps/ps2_logs.txt`: Registro de logs generados al ejecutar `ps/ps2_test.py`.
- `mlfin/utils.py`: Contiene la función `get_allocations()`, que genera la grilla de ponderaciones factibles.
- `mlfin/printing.py`: Incluye utilidades para configurar y formatear el logging.

3 Documentación

3.1 Implementación del modelo de cinco factores

La clase `BalancedPortfolio` está definida en `ps/ps2.py`. A continuación se describen sus componentes principales:

3.1.1 Constructor (`__init__`)

El constructor recibe como parámetro:

- `asset_returns` (`pd.DataFrame`): retornos diarios de los activos, con fechas en el índice y activos en las columnas.

Internamente:

- Se hace una copia local de los datos en `self.asset_returns`.
- Se inicializa el sistema de logging (`mlfin.printing.setup_logging`) y se registran en debug las primeras filas del DataFrame y el conteo de fechas disponibles.

3.1.2 Método `get_balanced_portfolio`

Este método calcula las ponderaciones óptimas de un portafolio para que la exposición a cada uno de los cinco factores sea lo más parecida a 1/n, siendo n la cantidad de factores. Recibe como parámetro:

- `factor_returns` (`pd.DataFrame`): retornos diarios de los factores que tiene como columnas a los retornos de los factores y fechas en el índice.

En el interior del método se lleva a cabo un algoritmo con los siguientes pasos:

1. Se alinean fechas entre activos y factores con `pd.merge` y se eliminan filas con `NaN`.
2. Para cada activo, se ajusta un modelo `LassoCV(cv=5)` contra la matriz de factores X y se obtienen los coeficientes resultantes son las exposiciones $\beta_{i,j}$.
3. Se construye la matriz de *loadings* $B \in \mathbb{R}^{n_{\text{factores}} \times n_{\text{activos}}}$, escalada a puntos porcentuales.
4. Se define el vector objetivo de exposición uniforme $t = (1/n_{\text{factores}}, \ldots, 1/n_{\text{factores}})$.
5. **Búsqueda en grilla:** se itera sobre todas las asignaciones factibles de pesos con `get_allocations`, se evalúa el MSE en cada una de ellas y se guarda la resolución óptima en los atributos `weights_grilla` y `exposure_grilla`.

6. **Optimización convexa:** resuelve

$$\min_{w \geq 0,\ \sum w = 1} \|Bw - t\|^2$$

usando `cvxpy` con el solver `CLARABEL`. Almacena los resultados en los atributos `weights_conv` y `exposure_conv`.

7. Registra en el log (info/debug) las ponderaciones y exposiciones óptimos encontrados por ambos métodos, asigna un atributo `results` que contiene un diccionario con los resultados de ambas optimizaciones.

8. Considerando ambas evaluaciones de la función objetivo en el óptimo, se elige el mejor modelo y se retornan las ponderaciones y exposición a los factores resultantes.

3.2 Pruebas y validación

El módulo `ps2_test` contiene la rutina de *testing* y generación de gráficos:

- Lee los archivos CSV de precios y factores [1]
- Calcula los retornos diarios de los ETFs y los convierte en exceso de retorno sobre el libre de riesgo.
- Filtra los datos para el año 2017 y selecciona los activos 'BOND','SUSA','DNL','XLF' y 'XSLV'.
- Instancia `BalancedPortfolio` usando como input para la clase los excesos de retorno obtenidos previamente y llama a `get_balanced_portfolio` para obtener ponderaciones y exposiciones.
- Imprime en pantalla los resultados para ambos métodos (grilla y convexa) con formato de porcentaje.
- Genera tres tipos de gráficos:
  1. Evolución acumulada (base 100) de ETFs vs. factores.
  2. Superficie 3D y contorno de la función score vs. pesos (primeros dos activos).
  3. Series "Real vs. Fitted" para el activo cuyo signo de exposición resultante es diferente del propuesto en el enunciado (e.g., `XSLV`).
- De este modo se valida numéricamente (MSE) y visualmente (gráficos) la correcta implementación del modelo.

4 Resultados

Para un análisis preliminar, en la Figura 1 se aprecia que el factor *Market minus RF* alcanza niveles acumulados muy superiores a los demás (picos del orden de miles frente a decenas), lo que genera una escala desequilibrada respecto al resto de los factores. Este desbalance puede sesgar la penalización en Lasso, que actúa sobre magnitudes absolutas de los coeficientes. Como ejercicio futuro, convendría *estandarizar* cada factor (media cero y desviación unitaria) antes de la regresión. Sin embargo, en adelante, se resuelve el ejercicio de acuerdo a lo solicitado en el enunciado. Con ello, para resolver numéricamente el problema de minimizar el MSE de la exposición uniforme, se aplicaron los dos métodos mencionados previamente:

1. **Búsqueda en grilla**, que explora combinaciones discretas y evita atraparse en mínimos locales.
2. **Optimización convexa**, que busca la solución óptima en el interior del dominio continuo $\{w \geq 0,\ \sum w = 1\}$. Además, debido a la continuidad, tiene la ventaja de encontrar una solución al menos tan buena como la del primer método (en términos de la evaluación final de la función objetivo) si se explora el mismo espacio de soluciones.

La Figura 2 ilustra la superficie y las curvas de nivel de la función score en los dos primeros activos. En la Tabla 1 se muestran los coeficientes de exposición estimados por LassoCV, y en la Tabla 2 las ponderaciones y exposiciones óptimas obtenidas con ambos métodos. Es de notar que la búsqueda en grilla reproduce exactamente el benchmark de ponderaciones del enunciado, y las exposiciones coinciden en signo y magnitud salvo para CMA, donde el signo obtenido es negativo.

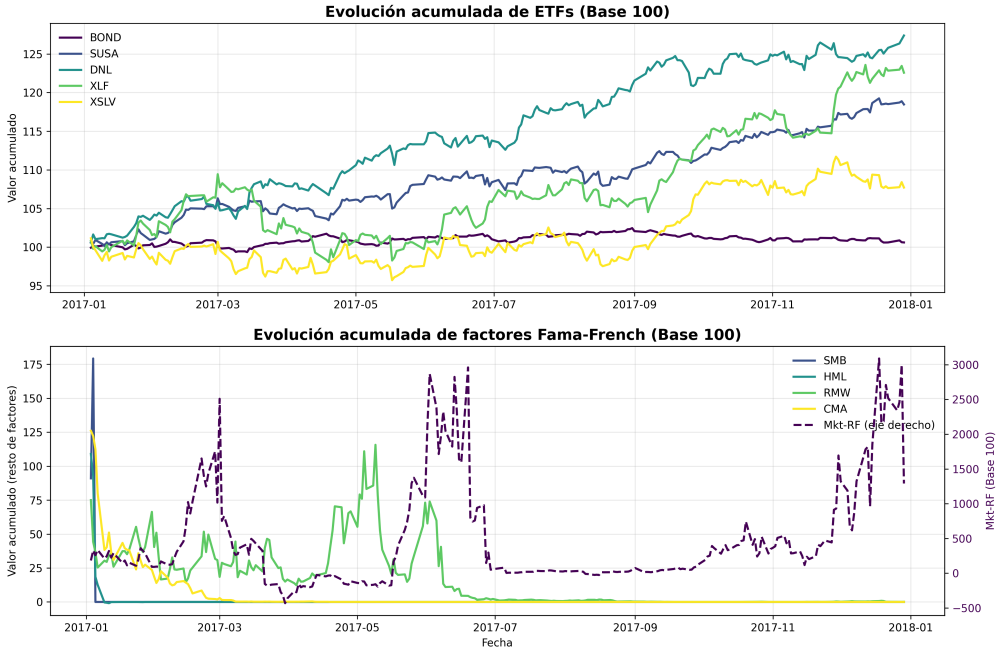

Figure 1: Evolución acumulada de ETFs (arriba) y de factores Fama–French (abajo).

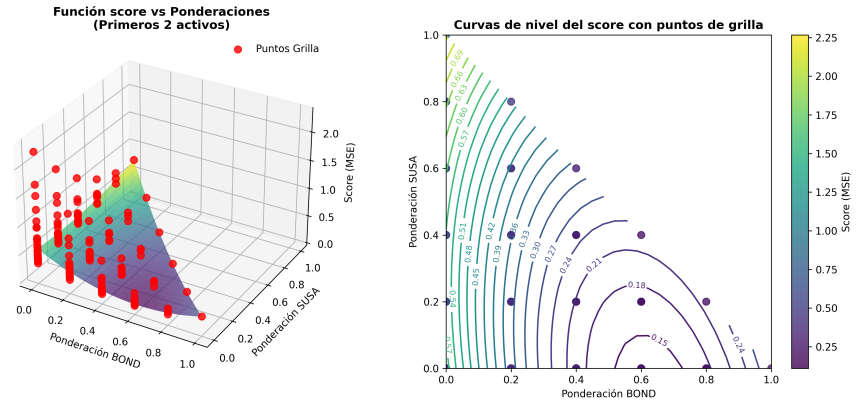

Figure 2: Superficie y curvas de nivel de la función score vs. ponderaciones (dos activos).

Table 1: Coeficientes de exposición estimados (LassoCV)

Factor	BOND	SUSA	DNL	XLF	XSLV
Mkt–RF	-0.0774	0.9499	0.6487	1.1592	0.7057
SMB	0.0000	-0.0591	0.0000	-0.0746	0.7328
HML	-0.1225	-0.1111	-0.3039	1.0366	0.3546
RMW	0.0000	0.0849	0.0000	-0.0997	0.2525
CMA	0.0000	0.0480	0.1993	-0.4939	-0.1738

Finalmente, la Figura 3 compara la serie real de XSLV en exceso de retorno con el ajuste *fitted* de Lasso. De este modo, se confirma gráficamente que la pendiente es negativa frente a CMA, lo que sugiere como ejercicio futuro reconsiderar dos cuestiones:

- *Penalización Lasso:* dado que tiende a forzar a cero las exposiciones menos relevantes y, en

[1] `selected_etfs.csv` y `F-F_Research_Data_5_Factors_2x3_daily.csv`, respectivamente.

Table 2: Ponderaciones y exposiciones óptimas para 2017

Método	**BOND**	**SUSA**	**DNL**	**XLF**	**XSLV**
Ponderaciones (%)					
Grilla	60.00	0.00	0.00	0.00	40.00
Convexa	61.00	0.00	4.00	1.60	33.40
Exposiciones (%)					
Grilla	23.60	29.30	6.80	10.10	–7.00
Convexa	23.30	24.40	4.80	8.30	–5.80

ocasiones, puede estimar un signo diferente del parámetro estructural si esto reduce el MSE penalizado.

- *Muestra limitada a 2017:* los outliers o períodos con baja dispersión pueden alterar la pendiente real del spread estimada.

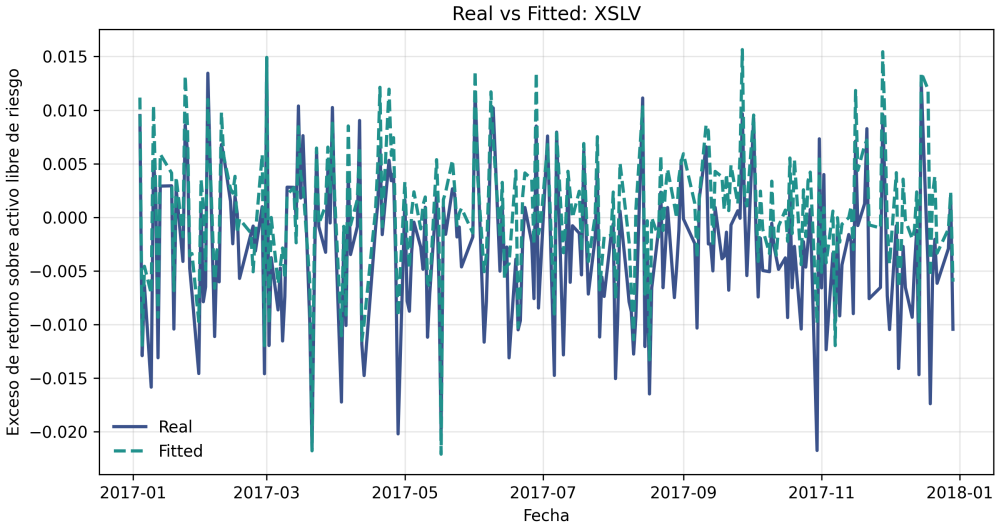

Figure 3: Real vs. fitted para XSLV (exceso de retorno).

References

[1] Fama, E. F. & French, K. R. (2015). A five-factor asset pricing model. *Journal of Financial Economics*, 116(1), 1–22. 10.1016/j.jfineco.2014.10.010.

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

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