Instituto Tecnológico y de Estudios Superiores de Monterrey

EGADE Business School

Finanzas Corporativas

Profesor: Dr. Alberto Méndez Morales

"Portafolio Eficiente"

Optimización mediante Algoritmos Genéticos

Javier Durán Vega

30 de Noviembre de 2025

Introducción y Justificación de la Metodología

El objetivo inicial del trabajo propuesto es entender cómo se puede diversificar el riesgo mediante la metodología de Markowitz para generar portafolios de inversión eficientes, así como optimizar la cartera dada las cotas de riesgo y rendimientos deseados.

Si bien se pretende aplicar estos conocimientos de manera general, las herramientas tradicionales como Excel no siempre contemplan opciones de mejora u optimización inteligente para problemas de gran escala. Dado que esto excede el enfoque habitual, me permito desarrollar este trabajo como aporte para cubrir las deficiencias de cómputo que podría tener Excel al momento de generar los posibles escenarios. El problema recae en el tiempo que Excel tarda en procesar la información; más allá de que la herramienta no esté optimizada, nos referimos a los cálculos de combinatoria como un proceso que es computacionalmente costoso (un problema de complejidad NP en ciertos contextos).

Dado que existen infinitas posibilidades en un espacio continuo para encontrar estos escenarios, resulta eficiente utilizar una técnica de cómputo avanzado como las metaheurísticas. En este caso puntual usaré un **Algoritmo Genético (AG)** que nos permitirá obtener combinaciones eficientes, convergiendo en soluciones óptimas en unas cuantas generaciones o "épocas" [1].

Para fines prácticos se pondrá el repositorio con el código en GitHub, así como el link con la aplicación en Streamlit para su consulta a posteriori.

Recolección y Procesamiento de Datos

Para la recolección de datos se hace uso de la plataforma Yahoo Finance mediante la librería `yfinance`; esto nos permite recabar datos directamente de dicha plataforma y obtener los valores de los tickers, similar a lo realizado con Excel pero de forma automatizada.

El uso de Python nos permite usar selectores para eficientar la descarga. Primero, definimos la temporalidad y los rangos de fechas. Por otro lado, contamos con un botón en la interfaz desarrollada que nos permite agregar más tickers dinámicamente; de esta forma no nos limitamos a una cartera estática de 5 activos, sino que podemos ampliarla o reducirla según sea necesario.

El resultado es una matriz con el histórico de precios de cierre ajustados, información con la cual podemos generar las matrices de promedios, correlación, desviación estándar y los valores de Sharpe, métricas fundamentales para este ejercicio.

Construcción de Matrices Fundamentales

Una vez obtenidos los precios de cierre históricos, transformamos los datos en métricas para evaluar el comportamiento de cada activo.

El primer cálculo es la matriz de rendimientos porcentuales. Para cada ticker, calculamos el cambio porcentual entre días consecutivos usando la fórmula:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{1}$$

En Python, este cálculo se realiza de manera vectorizada sobre todo el DataFrame:

```
returns = close_prices.pct_change().dropna()
```

El método `pct_change()` aplica la fórmula a cada celda y `dropna()` elimina la primera fila vacía. Con esta matriz, calculamos el Promedio de rendimientos diarios y la desviación estándar muestral (Volatilidad):

$$\mu_i = \frac{1}{n}\sum_{t=1}^{n} R_{i,t} \tag{2}$$

$$\sigma_i = \sqrt{\frac{1}{n-1}\sum_{t=1}^{n}(R_{i,t} - \mu_i)^2} \tag{3}$$

Se observa el uso de $n-1$ en el denominador (Corrección de Bessel), equivalente a la función `DESVEST.M` (STDEV.S) de Excel. En Python:

```
mean_returns = returns.mean()       # Vector de promedios
volatility = returns.std(ddof=1)    # ddof=1 equivale a STDEV.S
```

Matriz de Correlación

La matriz de correlación mide la relación lineal entre los rendimientos. Los valores oscilan entre -1 y 1. Una correlación cercana a cero indica que los activos se mueven de manera independiente, lo cual es deseable para la diversificación.

$$\rho_{i,j} = \frac{Cov(R_i, R_j)}{\sigma_i \cdot \sigma_j} \tag{4}$$

```
correlation = returns.corr()
```

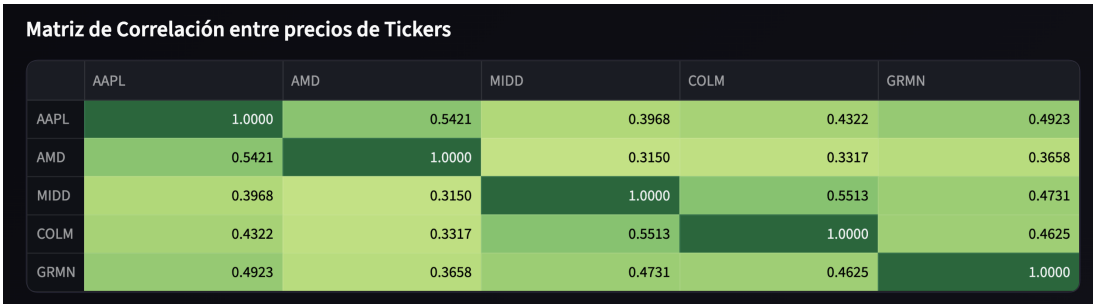

Figura 1: Mapa de calor de la Matriz de Correlación. Los tonos más oscuros indican una relación lineal positiva más fuerte entre los activos. Se busca combinar activos con correlaciones bajas (tonos claros) para maximizar la diversificación.

Matriz de Covarianza

Esta matriz contiene en su diagonal las varianzas diarias (σ_i^2) y en los demás elementos las covarianzas. Es esencial para calcular la volatilidad del portafolio:

$$Cov(R_i, R_j) = \frac{1}{n-1} \sum_{t=1}^{n} (R_{i,t} - \mu_i)(R_{j,t} - \mu_j) \tag{5}$$

```
covariance = returns.cov(ddof=1)
```

Figura 2: Mapa de calor de la Matriz de Covarianza (Diaria). La diagonal principal (tonos rojos/naranjas) muestra la varianza individual de cada activo, siendo AMD el de mayor magnitud, consistente con su riesgo.

Optimización de Portafolios y Frontera Eficiente

El Problema de Optimización

Un portafolio se define por un vector de pesos $\mathbf{w} = [w_1, ..., w_n]^T$ tal que $\sum w_i = 1$.

Rendimiento Esperado del Portafolio:

$$\mu_p = \mathbf{w}^T \boldsymbol{\mu} \tag{6}$$

Volatilidad del Portafolio:

$$\sigma_p = \sqrt{\mathbf{w}^T \Sigma \mathbf{w}} \tag{7}$$

En Python, implementamos estas operaciones matriciales y anualizamos los resultados (multiplicando el retorno por 252 y la volatilidad por $\sqrt{252}$):

```
def portfolio_return(weights, mean_returns):
    return np.dot(weights, mean_returns) * 252  # Anualizado

def portfolio_volatility(weights, cov_matrix):
    return np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights))) * np.
    sqrt(252)
```

Algoritmo Genético

Dado que el espacio de búsqueda es continuo, encontrar los pesos óptimos mediante fuerza bruta es ineficiente. Las metaheurísticas, específicamente los algoritmos genéticos inspirados en la evolución biológica, permiten evaluar la función de manera inteligente [1].

Representación del Individuo

El individuo en la población es un vector de pesos normalizado:

```
def create_individual(n_assets):
    weights = np.random.random(n_assets)
    return weights / weights.sum()  # Normalizar a 100%
```

Función Objetivo

Es el criterio a maximizar (Retorno, Sharpe) o minimizar (Volatilidad). Se añade un término de entropía $H(\mathbf{w})$ para premiar la diversificación y evitar soluciones donde se invierta todo en un solo activo a menos que sea estrictamente necesario.

$$f(\mathbf{w}) = \text{Métrica}(\mathbf{w}) + \lambda \cdot H(\mathbf{w}) \tag{8}$$

Análisis Detallado y Resultados

Para ilustrar la metodología, se seleccionó un portafolio compuesto por: **Apple (AAPL), Garmin (GRMN), AMD, Middleby (MIDD) y Columbia (COLM)**.

Se configuró una fecha desde abril 2021 hasta octubre 2025, utilizando datos **diarios**. Para el algoritmo genético se configuró una población de 200 individuos y 150 generaciones.

Figura 3: Evolución histórica de los precios de cierre ajustados (Abril 2021 - Octubre 2025). Se observa la tendencia alcista general y la alta volatilidad de AMD en comparación con la estabilidad relativa de GRMN.

Estadísticas Individuales

A partir de los datos históricos diarios, obtenemos las siguientes métricas base:

Ticker	Promedio Diario (%)	Volatilidad Diaria (%)	Sharpe (aprox)
AAPL	0.1110	2.0165	0.8741
AMD	0.1578	3.4091	0.7346
MIDD	0.0401	2.6240	0.2423
COLM	-0.0113	2.2620	-0.0794
GRMN	0.0739	1.9566	0.5997

Tabla 1: Estadísticas de rendimientos diarios por ticker

Se observa que **AMD** presenta el mayor rendimiento promedio diario (0,1578 %), pero a costa de la mayor volatilidad del grupo (3,41 %). En contraste, **GRMN** ofrece la menor volatilidad (1,96 %) con un rendimiento positivo moderado. **COLM** muestra un rendimiento negativo en este periodo, lo que sugiere que restará eficiencia al portafolio a menos que su correlación con otros activos sea muy baja.

Matriz de Covarianza

La matriz de covarianza (varianza diaria en la diagonal) confirma los niveles de riesgo individuales:

	AAPL	**AMD**	**MIDD**	**COLM**	**GRMN**
AAPL	**0.000407**	0.000373	0.000210	0.000197	0.000194
AMD	0.000373	**0.001162**	0.000282	0.000256	0.000244
MIDD	0.000210	0.000282	**0.000689**	0.000327	0.000243
COLM	0.000197	0.000256	0.000327	**0.000512**	0.000205
GRMN	0.000194	0.000244	0.000243	0.000205	**0.000383**

Tabla 2: Matriz de covarianza muestral de los rendimientos diarios

Resultados de la Optimización

Tras ejecutar el algoritmo genético, se obtuvieron tres portafolios característicos. Los valores presentados a continuación están **anualizados** para facilitar la interpretación financiera:

Métrica	**Máx. Retorno**	**Mín. Volatilidad**	**Óptimo (Sharpe)**
Retorno Anual	37.35 %	17.06 %	28.85 %
Volatilidad Anual	48.05 %	26.45 %	30.75 %
Sharpe Ratio	0.736	0.569	0.873

Tabla 3: Métricas anualizadas de los portafolios optimizados

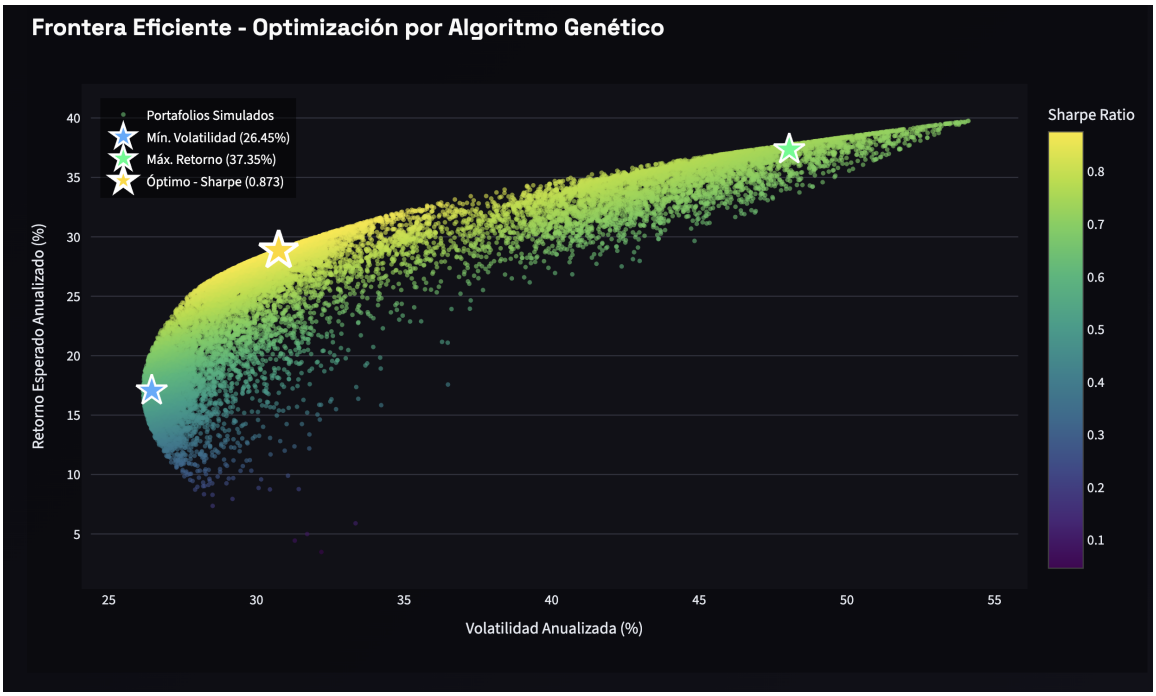

Figura 4: Visualización de la Frontera Eficiente generada por el Algoritmo Genético. La nube de puntos representa los portafolios simulados, y las estrellas destacan los tres escenarios óptimos seleccionados.

Composición de Pesos

- **Portafolio de Máximo Retorno:** La estrategia se inclina agresivamente hacia el activo de mayor rendimiento histórico.
  - AMD: 83.7 %
  - AAPL: 12.2 %
  - (Resto marginal en MIDD y GRMN)

  Esto genera un retorno anual del 37.35 %, pero con una volatilidad considerable del 48 %.

- **Portafolio de Mínima Volatilidad:** Busca minimizar la varianza global aprovechando las correlaciones.
  - GRMN: 33.2 % (Activo menos volátil)
  - AAPL: 30.3 %
  - COLM: 20.5 %

- MIDD: 11.3 %
- AMD: 4.6 %

Sorprendentemente, COLM se incluye con un 20 % a pesar de su rendimiento negativo, puramente por efectos de diversificación de varianza. La volatilidad baja a 26.45 %.

- **Portafolio Óptimo (Máximo Sharpe):** Ofrece la mejor relación riesgo-beneficio (Sharpe 0.873).
  - **AAPL: 54.0 %**
  - **AMD: 24.5 %**
  - **GRMN: 21.5 %**

  Este portafolio descarta totalmente a MIDD y COLM. Se apalanca en la solidez de AAPL y GRMN para mitigar la volatilidad, mientras mantiene casi un cuarto del portafolio en AMD para capturar crecimiento, logrando un retorno anual del 28.85 % con un riesgo controlado del 30 %.

Conclusiones

La implementación de algoritmos genéticos demuestra ser una alternativa superior a la búsqueda exhaustiva. El AG logró converger hacia una frontera eficiente clara (visible en la gráfica de dispersión con forma de hipérbola), identificando que la combinación óptima para este conjunto de activos reside mayoritariamente en el sector tecnológico (AAPL y AMD) balanceado con electrónica de consumo (GRMN).

A diferencia de los optimizadores clásicos, este método nos permitió observar cómo activos con rendimientos negativos (COLM) pueden tener utilidad técnica en un portafolio de mínima varianza, pero son descartados inmediatamente cuando el objetivo es maximizar la eficiencia (Sharpe), validando la teoría de Markowitz con datos reales.

Anexos

A. Disponibilidad del Código y Recursos

Con el objetivo de garantizar la reproducibilidad de los resultados expuestos en este reporte y fomentar la transparencia metodológica, se ha puesto a disposición pública todo el código fuente utilizado.

El repositorio incluye:

- Los scripts de extracción de datos vía API de Yahoo Finance.
- La lógica completa del Algoritmo Genético (selección, cruce y mutación).
- La interfaz gráfica desarrollada en **Streamlit** para la visualización interactiva.
- El archivo `requirements.txt` con las dependencias necesarias.

Repositorio Oficial en GitHub

`https://github.com/Cliftaine/portfolio_genetics`

Última actualización: 30 de Noviembre de 2025

B. Instrucciones de Ejecución

Para ejecutar la aplicación en un entorno local, se requiere tener instalado `Python 3.8+` y seguir los siguientes comandos en la terminal:

```
# Clonar el repositorio
git clone https://github.com/Cliftaine/portfolio_genetics.git

# Instalar dependencias
pip install -r requirements.txt

# Ejecutar la interfaz de Streamlit
streamlit run app.py
```

Referencias

[1] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.