Gestión de carteras con optimización de medidas clásicas de performance

by

Óscar Rodríguez Gallego

A thesis submitted in conformity with the requirements
for the MSc in Economics, Finance and Computer Science

Universidad de Huelva & International University of Andalusia

November 2018

Gestión de carteras con optimización de medidas clásicas de performance

Óscar Rodríguez Gallego

Máster en Economía, Finanzas y Computación

José Manuel Bravo Caro
Universidad de Huelva y Universidad Internacional de Andalucía

2018

Abstract

The purpose of this paper is to study different investment strategies with EURO STOXX50 shares, from 2003 to 2017 and the progression in this period of time. The strategies have been divided in passive and active investing. On the one hand, passive investing could be a ETF like EURO STOXX50 or a balanced portfolio. On the other hand, the optimized portfolios with Sharpe ratio, Treynor ratio and Jensen ratio by least squares. The optimized portfolios would obtain better results in economic growth or economic stability.

JEL classification: G11, G12, G15, G17

Key words: EURO STOXX50, portfolio, pricing, invest, simulation, Sharpe, Treynor, Jensen, optimize, python, least squares, programming.

Resumen

El trabajo se centra en el estudio de diferentes estrategias de inversión con las acciones del índice EURO STOXX50 desde el año 2003 hasta 2017, con el objetivo de conocer el comportamiento de cada una de las estrategias para este escenario. Éstas se dividen en estrategias de gestión pasiva y de gestión activa. Entre las primeras tenemos una cartera que replica los movimientos del índice (mediante un ETF), y una que otorga el mismo peso para cada uno de los activos. Entre las segundas, se han optimizado los índices de Sharpe, Treynor y Jensen con una técnica de mínimos cuadrados, ofreciendo mejores resultados cuando el índice aumenta o cuando permanece estable.

JEL classification: G11, G12, G15, G17

Key words: EURO STOXX50, cartera, precio, inversión, simulación, Sharpe, Treynor, Jensen, optimizar, python, minimos cuadrados, programación.

Tabla de Contenidos

Lista de Figuras y Tablas

1 Introducción

En el mundo de la inversión existen multitud de estrategias a la hora de elegir los activos de tu cartera y/o elegir el momento adecuado. Esto sumado a la poca formación financiera de muchos inversores y a la falta de escrúpulos de algunos agentes y especuladores hacen que la inversión en bolsa sea considerada más aleatoria de lo que ya es (Gómez-Bezares, 1993).

No podemos olvidar de que los precios de las acciones siguen un paseo aleatorio, existen tantas probabilidades de que aumente la cotización como de que disminuya. Precisamente por esto, se dice que el mercado de valores funciona de forma corecta. Exceptuando casos en los que se tenga información privilegiada (Blanchard, Amighini & Giavazzi, 1991). Aun así, durante las últimas décadas se ha investigado mucho en el campo de la teoría de carteras con el objetivo no de encontrar alguna técnica infalible e innovadora con la que ganar dinero ilimitadamente, sino de aumentar la probabilidad de éxito en las inversiones bursátiles.

Según Brealey, Myers & Allen (1980) solo unos cuantos administradores de fondos pueden generar rendimientos superiores al mercado por unos años, dado que los estadísticos pueden batir al mercado en determinadas ocasiones, pero a largo plazo la situación se vuelve mucho más difícil y estos dejan de funcionar.

La gran variedad de técnicas a la hora de conformar tu cartera de inversión, hace que la opción de comprar un activo que siga un índice cada vez sea más atractiva. Esto sumado a los altos costes que suponen las comisiones por tener una cartera diversificada, hacen que este tipo de productos sean más demandados. A los fondos cotizados se les denominan ETF[1] (las siglas de Exchange Traded Funds), muchos de estos fondos replican los movimientos de un índice. Gracias a ellos, un inversor que no posea unos fondos suficientes para tener una cartera diversificada, puede comprar algunos de estos ETF y no arriesgarse a tener una cartera con pocas acciones.

[1] La operativa de estos activos es muy similar a la de las acciones, dado que operan el mismo tiempo que ellas y poseen la misma fiscalidad.

Entre los inversores el concepto de diversificación está muy arraigado, pero son pocos los que realmente lo practican en su cartera de inversión. En Europa, gracias a la formación de la zona euro se ha adoptado un enfoque más generalista y esta diversificación de la que hablamos se está produciendo con más intensidad, gracias a la moneda única. Además, el auge de los fondos de inversión permite a pequeños inversores diversificar sin la necesidad de grandes sumas de dinero (Lamothe, 1999).

Los ETF que nos interesan en este estudio son los que replican los índices[2], en concreto, los que replican al índice EURO STOXX50 (generalmente, se trata del índice más usado a la hora de analizar el mercado accionarial de la zona euro). El índice fue creado en 1998 y fue desarrollado por la sociedad Dow-Jones. El cálculo del índice se desarrolla a partir de una ponderación de sus componentes a través de su capitalización en bolsa. En la Figura 1 se puede observar la evolución de su cotización en los últimos años.

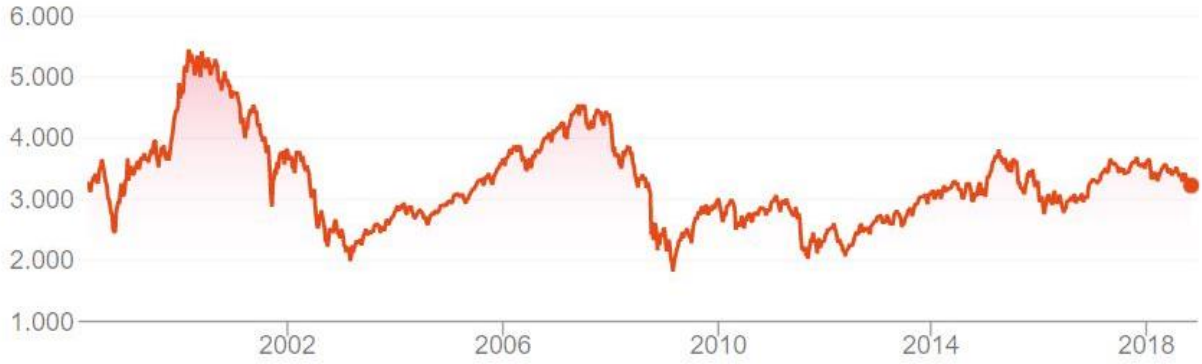

Figura 1. Evolución histórica de la cotización del índice EURO STOXX50 desde 1998 hasta 2018. Fuente: Google Finance.

Desde su creación, el índice ha sufrido numerosos cambios tanto en su composición (cada año se estudian las 50 empresas europeas con mayor capitalización, y si alguna de las que estaban dentro del selectivo es superada por otra que no aparece en él, se realizará el cambio oportuno) como en el peso de cada uno de sus activos.

[2] Además de los ETF referenciados al índice, existen CFD (Contracts For Difference) que simulan las cotizaciones del índice.

2 Objetivos

El propósito de este trabajo es observar los resultados de aplicar medidas clásicas de performance en las circunstancias actuales y con las herramientas computacionales que disponemos (así como el volumen de datos que se generan), para tener en cuenta estas estrategias de gestión activa de nuestro patrimonio o no. Para tener una fuente de comparación de estas técnicas de optimización, usaremos una estrategia de gestión pasiva de seguir el índice EURO STOXX50 mediante un ETF y otra de comprar todos los activos del índice con la misma proporción dentro de la cartera.

3 Revisión del estado del arte

En el año 1990, el Nobel de economía se concedió a tres economistas que estudiaron la Teoría de carteras, concretamente para Harry Markowitz, Miller y William Sharpe. El primero de ellos, es considerado el creador de la teoría con su trabajo de 1952. Sharpe por su parte simplificó el modelo y dio paso a un modelo de valoración de activos llamado CAPM[3]. Otro premio Nobel de Economía consiguió introducir avances en esta materia, introduciendo nuevos elementos en la teoría de carteras, es el caso de James Tobin al incluir la tasa libre de riesgo (Gómez-Bezares, 1993).

La teoría trata de buscar la combinación de activos, que, dado un riesgo maximice la rentabilidad de la cartera. Naturalmente, esta definición deja en el aire cuál es el riesgo que esté dispuesto a asumir el inversor, por tanto, las carteras óptimas son numerosas. Para identificar estas carteras, se desarrolló la llamada frontera eficiente, la cual nos permite visualizar cuáles son las carteras que maximizan la rentabilidad para cada riesgo. Posteriormente, a esta frontera se le ha añadido el concepto de activo libre de riesgo, que permite reducir el riesgo de la cartera combinando los títulos con y sin riesgo.

Hasta ahora tenemos dos medidas para las carteras, la rentabilidad y el riesgo, pero no son suficientes a la hora de compararlas entre sí, necesitamos una medida que sea capaz de unificar ambas y ofrecernos un único dato. A raíz de este problema se crearon las medidas clásicas de

[3] Capital Asset Pricing Model.

performance, que relacionan la rentabilidad con el riesgo de formas diferentes. Entre las medidas más famosas podemos encontrar el índice de Sharpe, el índice de Treynor y el Índice de Jensen. El índice de Sharpe relaciona la rentabilidad de la cartera con el riesgo total de esta y un activo libre de riesgo. En cambio, los otros dos índices sustituyen este riesgo total de la cartera por un riesgo sistemático, es decir, el riesgo que no se puede eliminar a través de la diversificación.

En lo relativo a los estudios realizados sobre la teoría de carteras son numerosos y de muy diversos tipos, en este apartado nos centraremos en los trabajos relacionados con la materia que sean similares a este (en los que se hayan utilizado varias medidas de *performance* u optimización de alguna de ellas).

Uno de los estudios interesantes sobre la materia es el realizado por Barrera (2014) en el que estudia las diferentes medidas de *performance* o eficiencia financiera que se han aportado al campo desde su creación. En él, comprueba y compara las diferentes medidas basadas en la teoría de carteras y en los modelos que salen de ella. Además, concluye en que es posible batir al mercado al usar estas herramientas en algunos escenarios y, por tanto, la gestión activa de la cartera quedaría justificada, en estos casos.

Otro de los estudios es el realizado por Johnson (2000) que estudia métodos alternativos de valoración de riesgo de las carteras. Entre estos métodos se encuentran la teoría de valores extremos, traking error y simulaciones de Monte Carlo. Concluye que, si la cartera no tiene activos no lineales como opciones, recomienda los análisis con retornos históricos, sin embargo, si existen activos en la cartera que sean no lineales recomienda el uso del método de simulación de Monte Carlo.

El último estudio que presentamos es el desarrollado en Conti, Simó y Rodríguez (2005), que estudian la teoría de carteras adicionando patrones de selección obtenidos mediante redes neuronales artificiales (RNA) para el mercado bursátil venezolano. En el estudio comparan ambas técnicas para el año 2002 y 2003. En cuanto a los resultados, las técnicas de redes neuronales artificiales ofrecen unos resultados homogéneos al de las medidas de *performance*, lo que nos demuestra que, bajo este escenario, este sistema de redes es eficiente para la valoración de carteras. El estudio recomienda el uso de técnicas de inteligencia artificial y de investigación de operaciones computacionales aplicables en el campo de las finanzas. Además, señala que uno

de los aspectos más importantes en la teoría de carteras es la diversificación para lograr reducir el riesgo total de la cartera.

4 Metodología

En este apartado se comentará la estructura de los datos, el escenario de inversión que se ha planteado para simular las estrategias y las diferentes estrategias que se han llevado a cabo.

4.1 Datos

La estructura de los datos corresponde a los denominados paneles de datos, un conjunto de datos forma un panel de datos cuando varios individuos son observados en diferentes momentos en el tiempo (Crespo, 2006).

Los datos han sido extraídos de la API (de sus siglas en inglés de Interfaz de Programación de Aplicaciones) de Yahoo!. Estos datos transcurren desde principios de 2002 hasta mediados del año 2018[4] y corresponden a los precios de cierre[5] de cada día de las acciones del índice europeo EURO STOXX50 (también se ha incluido el propio índice, dado que posteriormente necesitaremos la rentabilidad del mismo, la varianza y la covarianza con cada una de las acciones. Al ser un índice tenemos la problemática de que cada serie tiene una fecha inicial diferente, y en ocasiones, ésta puede no tener la longitud suficiente para extraer la suficiente información de ella. Por ello, las acciones de las empresas Amadeus y Koninklijke Ahold han sido retiradas de nuestro índice europeo, en su lugar se han incluido dos empresas que han estado presente en el índice EURO STOXX 50 hasta septiembre de 2018 (E.ON y Deutsche Bank) que tienen una serie temporal con mayor trayectoria. Una vez hemos realizado estos cambios observamos que las acciones con menor número de datos son Inditex, Airbus y Deutsche Bank que comienzan su serie a lo largo del año 2001, por tanto, el horizonte temporal para este trabajo comenzará en 2002.

---

[4] El primer día es el 2 de enero de 2002, dado que el 1 de enero las bolsas europeas permanecen cerradas, y el fin de los datos es el día 1 de julio de 2018.

[5] Se han utilizado los precios de cierre ajustados únicamente por splits y no por dividendos. Aunque el precio ajustado por dividendos refleje mejor la realidad de la rentabilidad de esa acción, se trata de un precio que no se da en la práctica y, por tanto, no es útil para la simulación de compra-venta que se va a realizar en este trabajo.

Las acciones presentes en nuestra base de datos se pueden agrupar en 8 países de la Unión Europea. El país con mayor representación es Francia con 18 empresas (L'Oreal, Orange, Danone, L'Air Liquide, Airbus, BNP Paribas, Essilor, Engie, Sanofi, Louis Vuitton, Safran, Schneider Electric, AXA, Vivendi, Total, Kering, Societe Generale y Vinci). El segundo país con más empresas es Alemania con 16 empresas (Adidas, Daimler, BASF, BMW, Fresenius, Bayern, Deutsche Telekom, Allianz, Deutsche Post, Volkswagen, Munich RE, SAP, Siemens, Linde, Deutsche Bank y E.ON). En tercer lugar, lo comparten Holanda y España con 5 empresas (de Holanda ASML, ING, Philips, Unilever y URW; y de España BBVA, Iberdrola, Inditex, Telefónica y Banco Santander). Por su parte Italia tiene a 3 empresas (Eni, Enel e Intensa Sanpaolo). Por último, Finlandia, Irlanda y Bélgica tienen una única empresa cada una (Nokia, CRH y Anheuser-Busch, respectivamente).

Una vez hemos obtenido los datos, el siguiente paso será la limpieza de los mismos. Dado que tenemos series que pertenecen a distintas bolsas de valores, los días en los que cotizan las acciones son diferentes. Por tanto, tenemos celdas vacías a lo largo de nuestro horizonte temporal. Para arreglar este problema las celdas que están vacías han sido sustituidas por su valor anterior inmediato, debido a que sí un mercado de valores no opera un determinado día el precio de sus activos sigue siendo el de cierre del último día en activo. Ahora que tenemos el panel de datos completo, identificamos qué datos corresponden a cada uno de los años para su posterior estudio, estableciendo así el número de días de cada año y el dato de las series en la que se cambia de año.

Además de las cotizaciones, necesitamos el rendimiento de un activo libre de riesgo para calcular las funciones que vamos a optimizar. Desde inicios del año 2002 hasta finales de noviembre de 2011, este activo libre de riesgo es el bono alemán a un año. Pero a partir de diciembre de ese mismo año se ha cambiado este activo de referencia por el bono estadounidense a uno año, debido a que durante ese mes el bono alemán a un año empezó a presentar tasas de rendimiento negativas. En el escenario planteado, el inversor no se plantea comprar un activo con

rendimiento negativo para reducir su riesgo[6], por tanto, hasta finalizar el horizonte temporal, éste será el activo de referencia libre de riesgo.

4.2 Escenario de inversión

4.2.1 Descripción general

En este momento debemos plantear las condiciones de nuestro escenario para su posterior simulación. Nuestro inversor posee una suma de 100.000€ y su bróker posee unas comisiones de para la compra y venta de un 2% del valor de la operación[7]. A su vez, pretende conservar unos límites máximos de peso de los activos de su cartera de un 2,5% para tener una cartera totalmente diversificada.

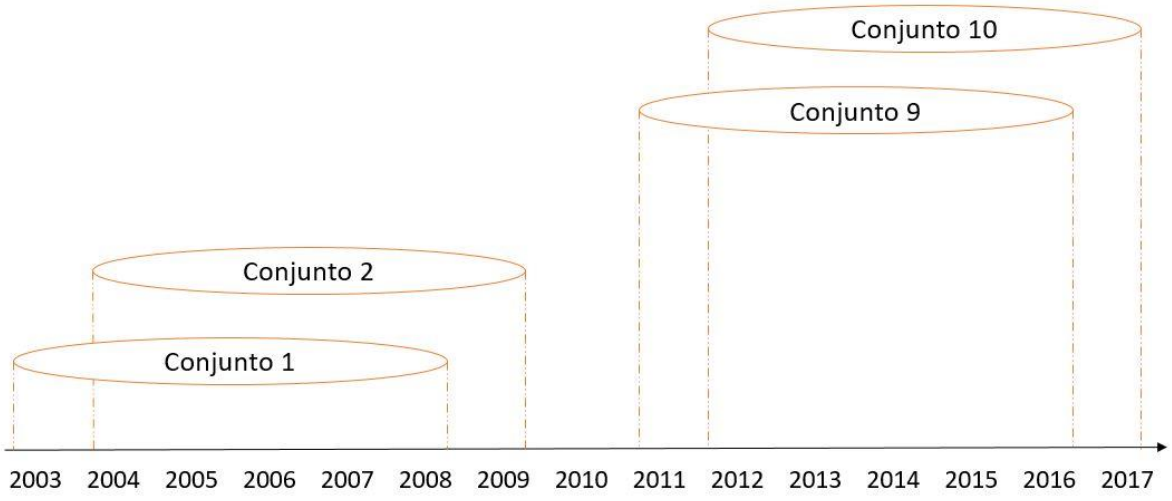

Figura 2. Ventanas deslizantes de la simulación. Fuente: Elaboración propia.

En la Figura 2 observamos la distribución de los dos primeros conjuntos de simulaciones y de los dos últimos. En el caso de la compra en estrategias no optimizadas, se simula que el inversor ha invertido en cada uno de los días hábiles del primer año del conjunto (con los mismos fondos), y vende en el último año del conjunto, por tanto, el número de simulaciones dentro de cada conjunto es igual al número de días hábiles en el primer año. En el caso de las optimizadas, cada

[6] Con objeto de simplificar el código no se ha planteado la problemática del tipo de cambio del dólar respecto al euro.

[7] Se ha establecido una única comisión (compra-venta) un poco más elevada que la media para no tener que incluir los diversos tipos de comisiones que existen.

año se cambian las acciones que tenemos en cartera. Por tanto, vendemos las acciones iniciales que tenemos y compramos las nuevas acciones que mantendremos durante un año completo, así hasta llegar al último año de la simulación. Es decir, en cada una de los conjuntos dibujados en la Figura 2, hay multitud de simulaciones, cada una para un día diferente. Con ello, conseguimos mayor volumen de datos y resultados.

En el escenario simularemos que un inversor esté dispuesto a invertir la misma cantidad (100.000€) durante seis años en reiteradas ocasiones, para tener la mayor cantidad de datos posibles, empezando desde inicios del 2003[8]. El primer paso es calcular cuántos títulos podrá comprar de cada activo en cada momento. Para ello, dividimos el total de fondos entre el precio del activo en ese momento y multiplicamos el resultado por el peso correspondiente a ese activo dentro de la cartera. Siendo *j* el día que calculamos los títulos e *i* el activo correspondiente.

$$\boldsymbol{titulos_j = \frac{fondos}{precio_j} \times w_i} \qquad [1]$$

El segundo paso es hallar la cantidad de fondos que se van a usar para la compra de los títulos de cada activo, dado que el 100% no se usará debido a que en el cálculo anterior donde sacábamos el número de títulos debemos truncar ese número a un entero[9]. A partir de este punto las estrategias siguen diferentes caminos, por tanto, no entraremos en cada una de ellas por el momento.

Una vez terminamos nuestras posiciones en el mercado, procederemos a calcular el resultado de las inversiones. Para clarificar los resultados mostraremos varias comparaciones entre estrategias usando histogramas, gráficos de su evolución en el tiempo y una tabla resumen de los datos que sean comparables.

---

[8] En las estrategias de optimización se indicará por qué se ha dejado 2002 sin simular.

[9] Por ejemplo, sí calculamos el número de títulos de Adidas que podemos comprar el primer día según una estrategia en la que a Adidas le corresponda un 2% de los fondos, nos arrojará un resultado de 141,40620. Pero no podemos comprar una parte decimal de una acción, por tanto, el número de títulos se calculará por truncamiento (141).

4.2.2 Cálculos de rentabilidad y riesgo del escenario

Antes de empezar a estudiar cada una de las estrategias de inversión, debemos aclarar varios conceptos que serán vitales para su entendimiento.

Primero debemos pasar las cotizaciones a valores que nos expresen el rendimiento de la serie. Para ello, realizamos lo que viene expresado en la Ecuación 2 para cada una de las observaciones de nuestras series temporales (se trata de la tasa de variación entre el precio de un periodo adelantado respecto a su anterior).

$$\boldsymbol{R = \frac{P_1 - P_0}{P_0}} \qquad [2]$$

A la hora de evaluar una cartera de inversión debemos tener claro los conceptos de rentabilidad y riesgo en ésta. Por lo que respecta a la rentabilidad, debemos calcularla en un primer momento para cada uno de los activos por medio de la Ecuación 3, siendo *d* el número de días que contiene un año y *j* el día en que se está calculando la rentabilidad, para cada uno de los activos (*i*).

$$\boldsymbol{\overline{R_i} = \sum_{j=1}^{d} R_{i,j} \cdot \frac{1}{n}} \qquad [3]$$

Una vez tenemos la rentabilidad media de cada uno de los activos que se presentan en la cartera, pasamos a la Ecuación 4 en la que calculamos la rentabilidad media de la cartera. En esta ecuación ponderamos cada una de las rentabilidades por el peso que presentan en la cartera (*w*). Debido a que nuestros datos son diarios, hemos multiplicado el sumatorio por el número de días que tiene el año en el que estemos calculando su rentabilidad media anual. Siendo *n* el número total de activos (50).

$$\boldsymbol{E[R_p] = \sum_{i=1}^{n} w_i \cdot \overline{R_i} \cdot d} \qquad [4]$$

Markowitz (1952) plantea un modelo con el objetivo de buscar la combinación de activos que maximice la utilidad esperada. Se trata de maximizar la Ecuación 3 sujeta a varias restricciones. La primera de las restricciones es que la suma de los pesos debe ser igual a 1 (Ecuación 5).

$$\sum_{i=1}^{n} w_i = 1 \qquad [5]$$

La otra restricción es que todos los pesos deben ser mayores a 0 o nulos (Ecuación 6).

$$w_i \geq 0 \qquad i = 1, 2, \ldots, n \qquad [6]$$

En cuanto a las medidas de riesgo, Markowitz (1952) plantea un modelo a diferencia del anterior, su objetivo es minimizar. La ecuación que representa esta idea es la especificada en la Ecuación 7 (en nuestro caso hemos añadido *d*). En esta ecuación, Sharpe (1970) nos hace ver que la medida de riesgo depende de las varianzas de la rentabilidad de los títulos, de sus coeficientes de correlación y del peso de los activos en la cartera.

$$Var[R_p] = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \cdot cov(\overline{R_i}, \overline{R_j}) \cdot d \qquad [7]$$

Tanto la rentabilidad de la cartera como la varianza de ésta se han calculado para cada año, con el objeto de poder utilizar posteriormente estos datos. En lugar de usar la Ecuación 7 como medida de riesgo, se ha usado la raíz cuadrada de ésta para mayor simplicidad a la hora de comparar los datos.

Este planteamiento de Markowitz es el punto de partida de la teoría de carteras, dando paso a mejoras cómo las de Tobin (1958), en la que amplía el modelo incluyendo el activo libre de riesgo (r_f), gracias a éste las posibles carteras son fruto de las combinaciones entre los activos de riesgo y el activo libre de riesgo, siendo a partir de este momento mucho más claro ver cuáles son las combinaciones de activos de la frontera eficiente según el riesgo que queramos adoptar.

Una vez conocemos lo que hemos utilizado para medir el rendimiento de la cartera a un año (Ecuación 4) y lo que usamos como medida de riesgo (la raíz cuadrada de la Ecuación 7), podemos desgranar un poco más esta última medida. Dentro del riesgo podemos separarlo en dos partes, uno es el riesgo sistemático y el otro es el riesgo diversificable. En el primero, lo que mide es la sensibilidad del título ante las variaciones que se producen en el mercado, este a menudo se denomina beta (β, Ecuación 8). El segundo, sin embargo, se puede eliminar teniendo

una cartera lo suficientemente diversificada, dado que corresponde al riesgo de cada uno de los activos.

$$\boldsymbol{\beta_i} = \frac{\boldsymbol{cov}(\overline{\boldsymbol{R_i}}, \boldsymbol{R_{mercado}})}{\boldsymbol{var}(\boldsymbol{R_{mercado}})} \quad [8]$$

El valor de la beta nos puede indicar cuál es la relación del activo con el mercado, por ejemplo, en caso de que la beta sea igual a 1, nos indica que la acción se mueve exactamente igual que el mercado, sí el mercado sube un 2%, la acción también lo hará en la misma medida. En el caso de que la beta sea negativa, nos indica que el activo se mueve en dirección contraría al mercado, algo inusual en la práctica. Sí la beta es nula, los movimientos del mercado no afectan de ninguna manera al activo en cuestión. En el trabajo, para el cálculo de la beta de cada activo, se ha utilizado una regresión entre la rentbilidad del título y la del índice.

En las estrategias que analizaremos posteriormente se optimizaran diferentes índices que combinan rentabilidad y riesgo, y en algunos de ellos en lugar de presentar la raíz cuadrada de la Ecuación 7 como medida de este último, se utilizará la beta de cartera (Ecuación 9). Gracias a este cambio en la medida del riesgo en lugar de calcular el riesgo total, se obtiene el riesgo sistemático.

$$\boldsymbol{\beta_{cartera}} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{w_i} \cdot \boldsymbol{\beta_i} \quad [9]$$

Sí la cartera está bien diversificada, el riesgo de ésta depende mayormente del riesgo sitemático, es decir, del que es relativo al mercado, dado que los precios de los diferentes títulos no se mueven al unísono. En muchas ocasiones, las caídas de las cotizaciones de algunos activos son compensadas con las subidas de otros, dado que es un factor de riesgo que afecta únicamente al activo en cuestión. La diversificación perfecta surgiría cuando dos acciones se correlacionan negativamente, esto no ocurre en la realidad, a pesar de esto, se consigue reducir una parte del riesgo total de la cartera. Aunque sea positiva no todos los inversiores pueden planteársela, debido a que mantener un elevado número de activos diferentes requiere unos fondos mínimos y un tiempo mayor (Brealey, Myers & Allen, 1980).

4.3 Estrategias sin optimización

Las primeras estrategias que vamos a estudiar son la cartera que sigue la evolución del índice EURO STOXX y la cartera con diversificación equilibrada, que consiste en usar el mismo peso para cada activo.

Esta estrategia se basa en replicar el índice con la compra de un fondo cotizado (ETF) que simule los movimientos diarios de éste (ver Figura 1). Actualmente, se trata de la estrategia pasiva más usada, dado que permite tener un activo ya diversificado y sin los costes que ocasionaría comprar cada uno de los activos contenidos en él.

En la inversión indexada, los inversores no realizan previsiones sobre las acciones, sino que se intenta diseñar una cartera que replique lo mejor posible los cambios en el índice (Lamothe, 1999). Una de las ventajas de la gestión indexada es su bajo coste en diferentes conceptos como comisiones, bid-offer spreads, etc. Para L.L. Martín (1993) el ahorro es de 1,34% en el caso americano.

En la simulación realizada hemos supuesto que el inversor invierte en el año 0 y hasta que no pasan 6 años no realiza ninguna acción con la inversión, dado que únicamente se preocupa por la entrada y la salida de su inversión una vez pasados seis años.

Otra estrategia en la que no se hace optimización y que no implica un análisis previo de ningún tipo es la cartera diversificada de forma equilibrada. Aunque posee los mismos activos que la estrategia anterior, no tienen el mismo peso dado que el índice pondera a los componentes por su capitalización (con un máximo de un 10% del índice) y en esta estrategia todos ponderan al 2%.

4.4 Estrategias con optimización

En este apartado se estudiarán las carteras que han surgido a partir de analizar diferentes índices que se utilizan para medir la rentabilidad y riesgo de una cartera. La característica más importante de éstas es que a lo largo de los seis años de inversión el peso de los activos que contiene la cartera (w_i) va modificándose.

Para explicar como cambia el peso de los activos (w_i) debemos indicar primero como calculamos este vector. Nuestro simulador tiene como inicio el año 2003, o lo que es lo mismo, dejamos un

año sin simular. Esto se debe a que para tener los primeros pesos de los activos del año 2003 usamos los datos del año anterior, es decir, maximizamos una función para el año 2002 y usamos esos pesos para la inversión del año 2003.

Para la optimización se ha usado la librería de Python llamada scipy, concretamente optimize.minimize. En ella, establecemos la función a minimizar (aunque en nuestro caso queremos maximizar, una forma de realizarlo por este método es minimizando la función en negativo) que será el negativo del índice de Sharpe, de Treynor o de Jensen. Para optimizar la función se usarán los pesos de los activos dentro de la cartera (w_i). También debemos incluir las restricciones, por un lado, indicaremos que la suma de los pesos debe ser igual a 1 (Ecuación 5) y que el rango por el cual los pesos (w_i) se pueden mover es de 0 a 0,25 (establecido previamente por el inversor, para asegurar una cartera diversificada). Este proceso nos dará un vector de pesos diferentes por cada una de las funciones que hemos optimizado, los cuáles usaremos para el primer año de la inversión.

Cuando ha pasado un año de la entrada en el mercado, volvemos a realizar la optimización, pero en lugar de hacerla para el año previo a la primera compra, se hace para los nuevos datos que hemos obtenido durante el año. Este proceso se repetirá hasta llegar al último año de la inversión. Además, es reseñable el efecto de las comisiones en estas estrategias, que aparecen cada vez que cambia la composición de la cartera (tanto para la venta como para la compra), que hacen mermar los beneficios[10].

Ahora presentaremos las medidas clásicas de *performance* que se han utilizado en este trabajo. Se trata de sintetizar el modelo de la Ecuación 4 (rentabilidad) con la raíz de la Ecuación 7 o directamente con la Ecuación 9 (medidas del riesgo). Debido a que las rentabilidades no son comparables en activos que tienen diferente riesgo.

El primer índice lo presentó Sharpe (1963 y 1966) y parte de la idea del modelo de Markowitz e intenta dar una medida para poder comparar diferentes carteras, dada una rentabilidad media de

[10] En el sistema de simulación cada año vende la totalidad de las acciones y a continuación compra las acciones que el modelo indique que son las óptimas para ese año. Este sistema no representa totalmente la realidad, dado que se podría haber realizado la venta únicamente de las acciones que se quisieran desprender de la cartera. Pero se trata de un sistema simplificado de la realidad, que no refleja 100% la realidad, pero se aproxima con el objetivo de comprobar la eficacia de las estrategias.

la cartera, una rentabilidad libre de riesgo (r_f) y una medida de riesgo de la cartera. En ese momento presenta la fórmula que observamos en la Ecuación 10. Debemos indicar que este índice responde al riesgo total de la cartera, por lo que puede ser útil en carteras diversificadas (Gómez-Bezares, 1993).

$$\boldsymbol{S_i = \frac{E[R_p] - r_f}{Var[R_p]}} \quad [10]$$

Según Treynor (1965) la relación que existe entre las rentabilidades de los títulos y el mercado no es directa, sino que influyen simultáneamente en todos ellos. Por ello, Treynor elimina el denominador del índice de Sharpe y en su lugar introduce el riesgo sistemático de la cartera, es decir, la beta (Padilla, 2004).

$$\boldsymbol{T_i = \frac{E[R_p] - r_f}{\beta_{cartera}}} \quad [11]$$

El índice de Treynor, supone que la beta es un buen indicador del riesgo de la cartera, por tanto, utiliza el riesgo sistñematico únicamente, asumiendo la verificación del CAPM (Lamothe, 1999).

Por último, Jensen (1968 y 1969) da un nuevo enfoque en el que calcula la diferencia entre la rentabilidad del título o cartera con el activo libre de riesgo y la rentabilidad que debería de haber obtenido según el CAPM (Capital Asset Pricing Model).

$$\boldsymbol{J_i = \left(E[R_p] - r_f\right) - \beta_{cartera} \cdot (\overline{R_{mercado}} - r_f)} \quad [12]$$

Como conclusión de este apartado, para todos estos índices en los que se han unido los conceptos de rentabilidad y riesgo, nuestro objetivo es maximizalos.

5 Resultados

Una vez realizadas todas las técnicas anteriores, es necesario comparar los resultados obtenidos para poder obtener una conclusión lo más clara posible.

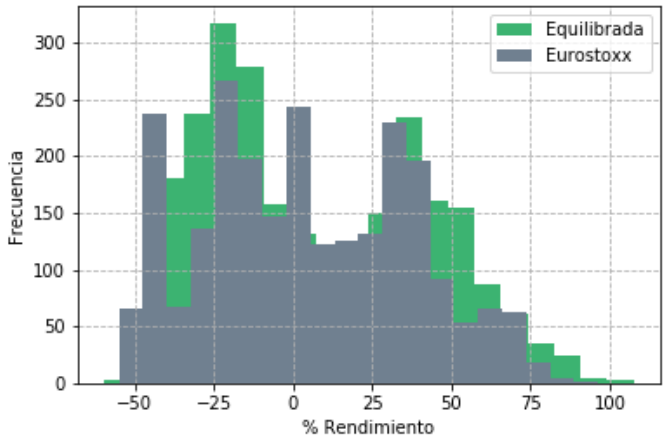

Figura 3. Histograma de las estrategias sin optimización. Fuente: Elaboración propia.

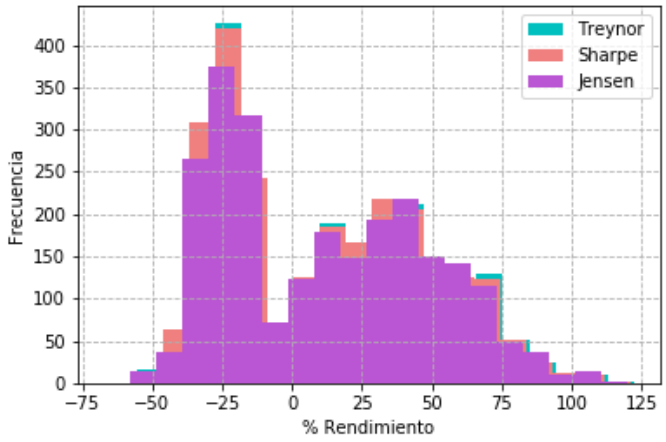

Figura 4. Histograma de las estrategias con optimización. Fuente: Elaboración propia.

Si comparamos la Figura 3 con la Figura 4 se puede observar que se ha penalizado el uso excesivo de cambios en la cartera por medio de las comisiones (pico alrededor del -25% en la Figura 4). Aunque sean estrategias con una mayor rentabilidad, en los años en los que no es posible obtener ganancias (debido a que la optimización del año anterior no ha sido suficiente, o bien, no había posibilidad de obtenerlas), las pérdidas se ven agravadas por las comisiones. Este hecho se ve mucho más claro en la Figura 5. En él se observa que en las últimas simulaciones de 2003-2008, en la totalidad de las simulaciones de 2004-2009 y de las simulaciones de 2005-2010 el rendimiento de las simulaciones sin optimización es más alto que en las optimizadas, debido a los problemas que hemos comentado anteriormente. Sin embargo, una vez terminados los efectos de la crisis en las cotizaciones bursátiles, estas estrategias optimizadas funcionan mucho mejor que las demás, dejando así una rentabilidad global media superior al finalizar el estudio. Durante los últimos años no ha existido una diferencia notoria en cuanto a rendimientos, pero al tener

comisiones en las estrategias de optimización, podríamos concluir que en el caso de no existir estas comisiones los resultados podrían haber sido superiores.

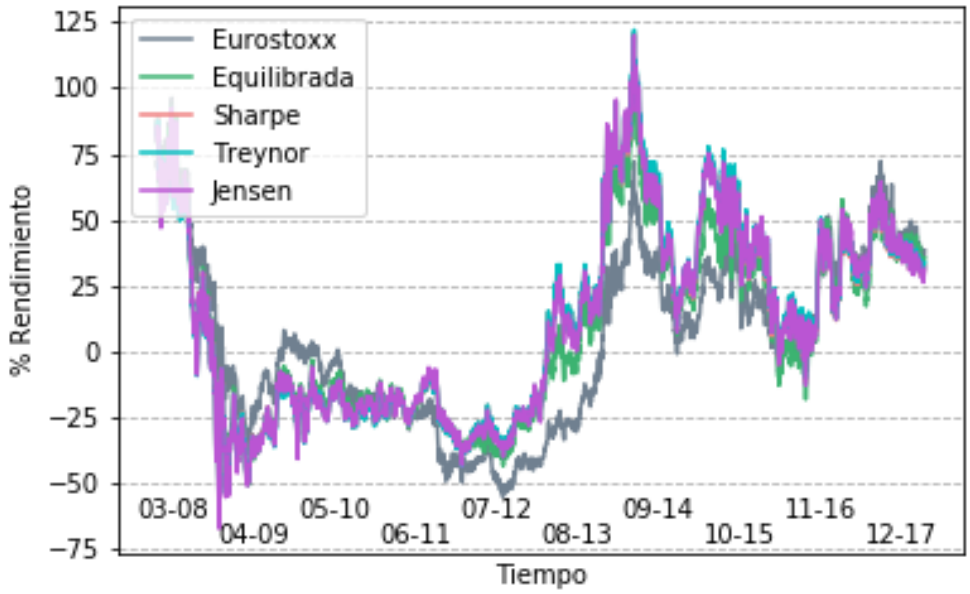

Figura 5. Evolución histórica de las carteras a lo largo del tiempo. En el eje de abscisas, aparecen los años de las diferentes inversiones a 6 años. Fuente: Elaboración propia.

Una forma diferente de visualizar los resultados obtenidos es con la elaboración de un histograma acumulativo, para conocer más características de nuestras estrategias (ver Figura 6). Gracias a este gráfico podemos observar que en la estrategia que sigue el EURO STOXX50 posee un porcentaje cercano al 20% de valores que son inferiores al -40% de rentabilidad. Sin embargo, en el resto de estrategias empiezan a existir rendimientos a partir del -40%. Una vez los rendimientos están entre el -20% y cercanos a 0, las estrategias son muy parejas y no se distancian hasta llegar a la nula rentabilidad. Como hemos visto en la Figura 4, apenas existen rendimientos cercanos al 0% en las estrategias de optimización, ayudándonos con la Figura 8 podemos afirmar que las carteras con optimización reducen nuestros resultados bajo condiciones desfavorables del mercado (entre los años 2008 y cercano a 2010, ver Figura 1) debido principalmente al excesivo coste en comisiones, sin embargo, en condiciones en las que el mercado se mantiene estable o incluso en alza, estas técnicas mejoran los resultados.

Además, también se puede ver que las medianas se podrían agrupar en dos grupos, las que están entorno al 1% (EURO STOXX50 y la cartera equilibrada) y las que están alrededor del 10% (las optimizadas). Una vez vamos avanzando hacia el rendimiento máximo, las estrategias sin optimizar llegan con antelación a su máximo, y posteriormente, lo hacen las optimizadas casi a la par.

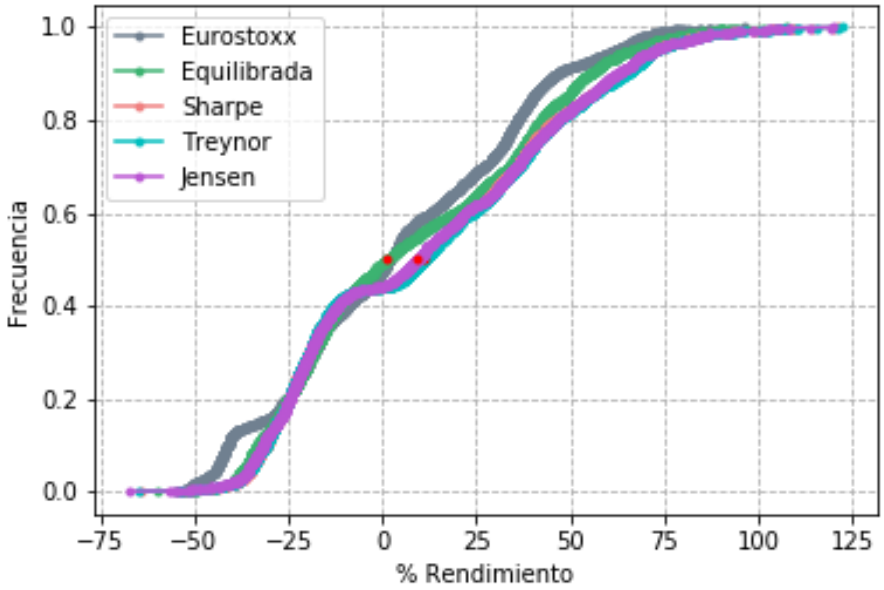

Figura 6. Histograma acumulativo de las estrategias estudiadas. Cada uno de los puntos rojos muestra el valor de su mediana. Fuente: Elaboración propia.

Para aclarar las conclusiones que necesitamos para finalizar el estudio, se ha elaborado una tabla de resultados en la que se han recogido los siguientes datos para cada una de las estrategias: rentabilidad media, desviación típica, mediana, porcentaje de resultados positivos, índice de Sharpe, de Treynor y de Jensen.

La estrategia con mayor rentabilidad es la optimizada con Treynor con un 11,981%, sin embargo, también es la que posee mayor desviación (37,0653%). Por su parte, la de menor variabilidad es la que sigue el índice EURO STOXX50. En cuanto a la mediana, ya hemos clarificado que podríamos separarlas en dos grupos (optimizadas, entorno al 10%; y no, entorno al 1,5%). Pero es interesante conocer que en las carteras no optimizadas la rentabilidad media y la mediana se distancian en casi 2,5 puntos y 7 puntos. Algo que nos indica que, aunque la rentabilidad media de todo el recorrido sea de un 4,36% y un 8,33%, respectivamente, la mitad de ocasiones hubiésemos tenido una rentabilidad menor a 1,92% y a 1,42%, respectivamente.

Debido a esta diferencia entre la mediana y la media de la rentabilidad de algunas carteras, nos preguntamos cuáles son las probabilidades de tener éxito con nuestras estrategias, es decir, cuántas veces tenemos rendimientos positivos. Para dar respuesta a esto, se ha añadido la columna de resultados positivos de la Tabla 1. En ella, podemos encontrar de que la estrategia con la mayor probabilidad nos habría dado pérdidas es la equilibrada, a pesar de poseer una rentabilidad media superior a la de EURO STOXX50. En cuanto a las de mayor probabilidad de

éxito, la optimizada con Sharpe es la ganadora con un 56,16% de tener rendimientos positivos, a pesar de no ser la que tiene mayor rentabilidad media ni mayor mediana.

Por último, se han calculado los índices de Sharpe, de Treynor y de Jensen para comprobar que sus carteras optimizadas son las mejores[11]. Gracias a ellas, estas estrategias no sólo son comparables entre sí, además, nos ayudaran a encontrar una mejor cartera. Para ello, necesitamos un volumen amplio de datos para poder calcular los promedios de las rentabilidades y sus correspondientes medidas de riesgo (Gómez-Bezares, 1993). A vista de los resultados, la optimización ha sido llevada a cabo con éxito.

Tabla 1. Resultados obtenidos por medio de la simulación. Fuente: Elaboración propia a partir de los datos recogidos en Python.

Cartera	Rentabilidad media(%)	Desviación típica media(%)	Mediana	Resultados positivos(%)	Índice de Sharpe	Índice de Treynor	Índice de Jensen
EURO STOXX50	4,3624	32,8247	1,9226	52,59	0,3578	0,0354	0
Equilibrada	8,3344	34,4087	1,4264	50,85	0,6653	0,1049	0,0645
Optimizada con Sharpe	11,594	36,6402	10,8378	56,16	1,0203	0,1797	0,1263
Optimizada con Treynor	11,981	37,0653	11,1766	56,12	1,02	0,1797	0,1264
Optimizada con Jensen	11,3788	36,6942	9,3004	55,83	1,0086	0,1764	0,128

6 Conclusiones

A vista de los resultados expuestos podemos afirmar que, bajo el escenario planteado en el estudio, las estrategias de composición de activos en una cartera bajo criterios de optimización de las medidas clásicas de *performance* funcionan bien bajo entornos favorables. Durante los años en los que tanto el índice cómo los mercados europeos se han mantenido estables o al alza, la rentabilidad de estas carteras ha sido superior al resto de estrategias planteadas. Sin embargo, durante épocas de incertidumbre económica dónde las cotizaciones bursátiles tienden a caer precipitadamente, no se consigue mejorar los resultados de la estrategia pasiva (seguir al índice

[11] En las fórmulas de estos índices aparece el activo libre de riesgo (r_f). Aunque la inversión total es a 6 años, a la hora de calcular cada índice para cada estrategia, se ha realizado año por año, y posteriormente, se ha calculado su promedio. Por tanto, el activo libre de riesgo es el bono a 1 año.

EURO STOXX50). Y no sólo eso, sino que los resultados bajo estas circunstancias son peores en estas carteras.

Cómo anunciábamos en la introducción del trabajo, no existe una estrategia que te haga ganar dinero en bolsa de forma regular y bajo cualquier circunstancia, pero el hecho de basar nuestra inversión en la optimización de medidas clásicas de valoración de carteras con los datos pasados más inmediatos, no ha sido tan infructuosa como se podría imaginar. De hecho, se ha conseguido una rentabilidad media mejor, una mediana mayor y una probabilidad de éxito superior en estas carteras.

El mayor problema de utilizar técnicas de optimización para mejorar tu cartera es que para tener una rentabilidad superior a la del mercado te exige que cambies más a menudo la composición de tu cartera, lo que hace que la parte destinada al pago de comisiones aumente a una gran velocidad. Sin embargo, si establecemos unos límites a la hora de maximizar, en forma de restricciones, se pueden disminuir estos costes lo máximo posible (en nuestro caso, al indicarle un límite de peso máximo para un activo en la cartera de 2,5%, lo que hacía obligatorio que poseyéramos siempre como mínimo 40 tipos de activos).

Actualmente, se siguen multitud de estrategias bursátiles, unas más lógicas que otras, pero basar tu estrategia de inversión en los datos pasados sigue siendo una opción rentable y gracias a la gran cantidad de datos que disponemos y al gran volumen que se genera cada día, este tipo de herramientas serán útiles durante más tiempo. Gracias a la optimización vía computacional se pueden comparar multitud de activos en muy poco tiempo y comprobar con exactitud si funcionan actualmente las diferentes medidas de rendimiento y riesgo de las carteras en los mercados actuales.

Bibliografía

Barrera, J. (2009). Medidas de evaluación: performance de títulos, carteras o fondos de inversión. *Pensamiento crítico*, vol. 10, marzo, 29–42.

Blanchard, O., Amighini, A. & Giavazzi, F. (1991). *Macroeconomics: A European Perspective*. (5ª ed.) Londres: Pearson.

Brealey, R.A., Myers, S.C. & Allen, F. (1980). *Principles of Corporate Finance*. (9ª ed.) Nueva York: McGraw-Hill.

Conti, D., Simó C. & Rodríguez, A. (2005). Teoría de cartera de inversión para la diversificación del riesgo: enfoque clásico y uso de redes neuronales artificiales. *Revista Ciencia e Ingienería*, vol. 26, 1, 35–42.

Crespo, R. & Ortiz, S. (2006). *Los fondos de inversión a examen. Un análisis empírico con datos de panel*. Madrid: Vision Net.

Gómez-Bezares, F. (1993). *Gestión de carteras (Eficiencia, teoría de cartera, CAPM, APT*). (3ª ed.) Bilbao: Desclée de Brouwer.

Jensen, M.C. (1968). The performance of mutual funds in the period 1945-1964. *Journal of Finance*, vol. 23, 2, 386–416.

Jensen, M.C. (1969). Risk, the pricing of capital assets, and the evaluation of investment portfolios. *Journal of business*, vol. 42, 2, 167–247.

Johnson, C.A. (2000). Métodos de evaluación del riesgo para portafolios de inversión. *Documentos de Trabajo Banco Central de Chile*, 67, marzo.

Lamothe, P. (1999). *Gestión de carteras de acciones internacionales*. Madrid: Pirámide.

Markowitz, H.F. (1952). Portfolio Selection. *Journal of Finance* 7, 77–91.

Martín, L.L. (1993). The evolution of passive versus active equity management. *Journal of Investing*, 2,(1), 17-20.

Matplotlib.org (2018). Mathplotlib: Python plotting – Matplotlib 3.0.2 documentation. [online] Disponible en: https://matplotlib.org/index.html [Último acceso: 12 Nov. 2018]

Padilla, N. (2004). *Análisis multicriterio aplicado a la selección de carteras*. Huelva: Servicios de Publicaciones Universidad de Huelva.

Scipy.org (2018). Scypy v1.1.0 Reference Guide. [online] Disponible en: https://docs.scipy.org/doc/scipy/reference/index.html [Último acceso: 12 Nov. 2018]

Sharpe, H.F. (1963). A Simplified Model for Portfolio Analysis. *Management Science*, vol. 9, 2, 277–293.

Sharpe, H.F. (1966). Mutual fund performance. *Journal of business*, vol. 39, 1, 119–138.

Sharpe, H.F. (1970). *Portfolio theory and capital markets*. Nueva York: McGraw-Hill.

Tobin, J. (1958). Liquidity Preference as Behaviour Toward Risk. *Review of Economic Studies*, vol. 26, 1, 65–86.

Treynor, J.L. (1965). How to Rate Management of Investment Funds. *Harvard Business Review*, vol. 43, 1, 63–75.

Apéndice

El código seguido para la elaboración del trabajo es el siguiente:

```
#Importamos los módulos a utilizar
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from pandas_datareader import data as pdr
import fix_yahoo_finance as yf
import mpld3
import statistics as stats
import scipy.optimize as optimize
from tabulate import tabulate
import statsmodels.api as sm


pd.core.common.is_list_like = pd.api.types.is_list_like
yf.pdr_override() #arreglamos el data reader de Yahoo
lista = ['ADS.DE','DAI.DE','BAS.DE','BMW.DE','FRE.DE','BAYN.DE','DTE.DE',
         'ALV.DE','DPW.DE','VOW.DE','MUV2.DE','SAP.DE','SIE.DE','LIN.DE',
            'ASML.AS','INGA.AS','PHIA.AS','UNA.AS','URW.AS',
            'OR.PA','ORA.PA','BN.PA','AI.PA','AIR.PA','BNP.PA','EI.PA',
            'SAN.PA','MC.PA','SAF.PA','SU.PA','CS.PA','VIV.PA','FP.PA',
              'KER.PA', 'GLE.PA','DG.PA', 'ENGI.PA',
              'ENI.MI','ENEL.MI','ISP.MI',
            'BBVA.MC','IBE.MC','ITX.MC','TEF.MC','SAN.MC',
            'ABI.BR','CRG.IR','NOKIA.HE',
            'DBK.DE','EOAN.DE',
            '^STOXX50E']
start_date = "2002-01-02" #fecha de inicio
end_date = "2018-07-01" #fecha final

df=pd.DataFrame() #formato panel de datos
for asset in lista: #extraer los datos
    df_asset = pdr.get_data_yahoo(asset, start=start_date, end=end_date)
                                        ["Close"]
    df_asset = df_asset.to_frame(name=asset)
    df = pd.concat([df_asset, df], axis=1, sort=False)

df = df.dropna()
 #(precio_siguiente - precio_anterior) / precio_anterior
df_change = df.pct_change().dropna()
#extraemos el indice del dataframe para despues poder calcular la beta
indice = df.filter(items=['^STOXX50E'])
#extraemos el indice del dataframe para despues poder calcular la beta
indice_change = df_change.filter(items=['^STOXX50E'])
df.pop('^STOXX50E') #eliminamos el indice del dataframe de los precios
#eliminamos el indice del dataframe de las variaciones
df_change.pop('^STOXX50E')
```

```
#2002
#año 2002 completo (se suma un dia al de cierre dado que no lo incluye)
df_change2002 = df_change.iloc[0:245]
indice_change2002 = indice_change.iloc[0:245]
dias2002 = 245 #días del año 2002
rf2002 = .0335452 #promedio del 2002 del bono libre de riesgo a 1 año
#2003
df_change2003 = df_change.iloc[245:497]
indice_change2003 = indice_change.iloc[245:497]
dias2003 = 252
rf2003 = .0218835
#2004
df_change2004 = df_change.iloc[497:754]
indice_change2004 = indice_change.iloc[497:754]
dias2004 = 257
rf2004 = .0213645
#2005
df_change2005 = df_change.iloc[754:1006]
indice_change2005 = indice_change.iloc[754:1006]
dias2005 = 252
rf2005 = .0223160
#2006
df_change2006 = df_change.iloc[1006:1257]
indice_change2006 = indice_change.iloc[1006:1257]
dias2006 = 251
rf2006 = .0326670
#2007
df_change2007 = df_change.iloc[1257:1490]
indice_change2007 = indice_change.iloc[1257:1490]
dias2007 = 233
rf2007 = .0412950
#2008
df_change2008 = df_change.iloc[1490:1735]
indice_change2008 = indice_change.iloc[1490:1735]
indice_change2008_2 = indice_change.iloc[1610:1854]
dias2008 = 245
rf2008 =.0368123
#2009
df_change2009 = df_change.iloc[1735:1980]
indice_change2009 = indice_change.iloc[1735:1980]
indice_change2009_2 = indice_change.iloc[1854:2100]
dias2009 = 245
rf2009 = .0085205
#2010
df_change2010 = df_change.iloc[1980:2227]
indice_change2010 = indice_change.iloc[1980:2227]
dias2010 = 247
rf2010 = .0059499
#2011
df_change2011 = df_change.iloc[2227:2471]
indice_change2011 = indice_change.iloc[2227:2471]
dias2011 = 244
rf2011 = .0083021
#2012
df_change2012 = df_change.iloc[2471:2713]
```

```
indice_change2012 = indice_change.iloc[2471:2713]
dias2012 = 252
rf2012 = .0016717
#2013
df_change2013 = df_change.iloc[2713:2956]
indice_change2013 = indice_change.iloc[2713:2956]
dias2013 = 243
rf2013 = .0012336
#2014
df_change2014 = df_change.iloc[2956:3199]
indice_change2014 = indice_change.iloc[2956:3199]
dias2014 = 243
rf2014 = .0011042
#2015
df_change2015 = df_change.iloc[3199:3439]
indice_change2015 = indice_change.iloc[3199:3439]
dias2015 = 240
rf2015 = .0030520
#2016
df_change2016 = df_change.iloc[3439:3680]
indice_change2016 = indice_change.iloc[3439:3680]
dias2016 = 241
rf2016 =.0060029
#2017
df_change2017 = df_change.iloc[3680:3927]
indice_change2017 = indice_change.iloc[3680:3927]
dias2017 = 247
rf2017 =  .0119481


weights = []
#volvemos a poner la lista sin el indice,
#pero inversamente ordenada para usarla en el cálculo de la beta
#está inversa dado que hemos usado antes el comando iloc
#para extraer la información de los dataframe
#y en ellos el último elemento incorporado pasa siempre
#a la primera posición
lista  = ['EOAN.DE','DBK.DE','NOKIA.HE','CRG.IR','ABI.BR','SAN.MC',
'TEF.MC','ITX.MC','IBE.MC','BBVA.MC',
'ISP.MI','ENEL.MI','ENI.MI','DG.PA','GLE.PA','KER.PA','FP.PA','VIV.PA',
'CS.PA','SU.PA','SAF.PA',
'MC.PA','SAN.PA','ENGI.PA','EI.PA','BNP.PA','AIR.PA','AI.PA','BN.PA',
'ORA.PA','OR.PA','URW.AS',
'UNA.AS','PHIA.AS','INGA.AS','ASML.AS','LIN.DE','SIE.DE','SAP.DE',
'MUV2.DE','VOW.DE','DPW.DE',
'ALV.DE','DTE.DE','BAYN.DE','FRE.DE','BMW.DE','BAS.DE','DAI.DE','ADS.DE']



n_assets = len(lista)
#Formato de pesos
for i in range(n_assets):
    weights.append(1/n_assets)
w = np.array(weights)
```

```
fondos = 100000
limite=0.025
comision_compra=1.02
comision_venta=0.98
```

La siguiente parte del código debe ser replicada para cada año, aquí sólo he mostrada la del año 2002, con objeto de no repetir parte del código, pero sería hasta el 2015 (inclusive).

```
#2002
#cálculo de la beta
beta2002 = []
for accion in lista:
    x2002 = sm.add_constant(indice_change2002)
    model2002 = sm.OLS(df_change2002[accion],x2002)
    results2002 = model2002.fit()
    beta2002 = np.append(beta2002,results2002.params[1])
#cálculos previos
#rendimiento esperado de cada activo (media)
r2002 = np.array(np.mean(df_change2002))
#rendimiento esperado del indice (media)
rindice2002 = np.array(np.mean(indice_change2002))
C2002 = np.cov(df_change2002.transpose()) #matriz varianzas-covarianzas
#peso del indice 100% para la rentabilidad de la cartera de mercado
w_indice=[1]
#funciones necesarias
def mu2002(w,r2002):
    '''Rentabilidad cartera anual'''
    return sum(w * r2002 * dias2002)
def sigma2002(w, C2002):
    '''Desviación típica cartera anual'''
    return np.dot(w,np.dot(C2002,w.T)) ** (1/2) * dias2002 ** (1/2)
def mercado2002(w,beta2002):
    '''Beta de la cartera anual'''
    return sum(w * beta2002)
def sharpe2002(w):
    '''Ratio de Sharpe'''
    return (mu2002(w,r2002) - rf2002) / sigma2002(w,C2002)
def neg_sharpe2002(w):
    '''Ratio de Sharpe negativo'''
    return -sharpe2002(w)
def treynor2002(w):
    '''Indice de Treynor'''
    return (mu2002(w,r2002) - rf2002) / mercado2002(w,beta2002)
def neg_treynor2002(w):
    '''Indice de Treynor negativo'''
    return -treynor2002(w)



def jensen2002(w):
    '''Indice de Jensen'''
    return (mu2002(w,r2002) - rf2002) - (mu2002(w_indice,rindice2002) -
               rf2002) * mercado2002(w,beta2002)
def neg_jensen2002(w):
    '''Indice de Jensen negativo'''
```

```
    return -jensen2002(w)


##########SHARPE#####################
#Programación secuencial por mínimos cuadrados
#Optimización discreta (float), con restricciones (apply_sum:constraint)
#con función objetivo (sharpe en negativo, dado que minimizamos)
def apply_sum_constraint(inputs):
    total = 1 - np.sum(inputs)
    return total
my_constraints = ({'type': 'eq', "fun": apply_sum_constraint })
result = optimize.minimize(neg_sharpe2002,
                       w,
                       method='SLSQP',
                       bounds=((0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite)),
                       options={'disp': True},
                       constraints=my_constraints)
opt_s_w2002 = result["x"]


#############TREYNOR##################
#Programación secuencial por mínimos cuadrados
#Optimización discreta (float), con restricciones (apply_sum:constraint)
#con función objetivo (treynor en negativo, dado que minimizamos)
def apply_sum_constraint(inputs):
    total = 1 - np.sum(inputs)
    return total
my_constraints = ({'type': 'eq', "fun": apply_sum_constraint })
result = optimize.minimize(neg_treynor2002,
                       w,
                       method='SLSQP',
                       bounds=((0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite)),

                       options={'disp': True},
                       constraints=my_constraints)
```

```
opt_t_w2002 = result["x"]


###############JENSEN####################
#Programación secuencial por mínimos cuadrados
#Optimización discreta (float), con restricciones (apply_sum:constraint)
#con función objetivo (jensen en negativo, dado que minimizamos)
def apply_sum_constraint(inputs):
    total = 1 - np.sum(inputs)
    return total
my_constraints = ({'type': 'eq', "fun": apply_sum_constraint })
result = optimize.minimize(neg_jensen2002,
                       w,
                       method='SLSQP',
                       bounds=((0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite), (0, limite), (0, limite), (0, limite),
(0, limite), (0, limite),(0, limite), (0, limite), (0, limite),
(0, limite), (0, limite)),

                       options={'disp': True},
                       constraints=my_constraints)
opt_j_w2002 = result["x"]
```

Hasta esta parte debería ser replicada para cada uno de los años de estudio de las medidas de *performance*.
La siguiente parte del código pertenece a la simulación de carteras:

```
def cartera(inicio,fin,w1,w2,w3,w4,w5,dias1,dias2,dias3,dias4,dias5):
    rendimiento=[]
    for dia in range(inicio, fin+1):
        #año 1
        titulos_0=int((fondos / df.iloc[dia,0]) * w1[0])
        titulos_1=int((fondos / df.iloc[dia,1]) * w1[1])
        titulos_2=int((fondos / df.iloc[dia,2]) * w1[2])
        titulos_3=int((fondos / df.iloc[dia,3]) * w1[3])
        titulos_4=int((fondos / df.iloc[dia,4]) * w1[4])
        titulos_5=int((fondos / df.iloc[dia,5]) * w1[5])
        titulos_6=int((fondos / df.iloc[dia,6]) * w1[6])
        titulos_7=int((fondos / df.iloc[dia,7]) * w1[7])
        titulos_8=int((fondos / df.iloc[dia,8]) * w1[8])
        titulos_9=int((fondos / df.iloc[dia,9]) * w1[9])
        titulos_10=int((fondos / df.iloc[dia,10]) * w1[10])
        titulos_11=int((fondos / df.iloc[dia,11]) * w1[11])
        titulos_12=int((fondos / df.iloc[dia,12]) * w1[12])
        titulos_13=int((fondos / df.iloc[dia,13]) * w1[13])
        titulos_14=int((fondos / df.iloc[dia,14]) * w1[14])
        titulos_15=int((fondos / df.iloc[dia,15]) * w1[15])
        titulos_16=int((fondos / df.iloc[dia,16]) * w1[16])
```

```
titulos_17=int((fondos / df.iloc[dia,17]) * w1[17])
titulos_18=int((fondos / df.iloc[dia,18]) * w1[18])
titulos_19=int((fondos / df.iloc[dia,19]) * w1[19])
titulos_20=int((fondos / df.iloc[dia,20]) * w1[20])
titulos_21=int((fondos / df.iloc[dia,21]) * w1[21])
titulos_22=int((fondos / df.iloc[dia,22]) * w1[22])
titulos_23=int((fondos / df.iloc[dia,23]) * w1[23])
titulos_24=int((fondos / df.iloc[dia,24]) * w1[24])
titulos_25=int((fondos / df.iloc[dia,25]) * w1[25])
titulos_26=int((fondos / df.iloc[dia,26]) * w1[26])
titulos_27=int((fondos / df.iloc[dia,27]) * w1[27])
titulos_28=int((fondos / df.iloc[dia,28]) * w1[28])
titulos_29=int((fondos / df.iloc[dia,29]) * w1[29])
titulos_30=int((fondos / df.iloc[dia,30]) * w1[30])
titulos_31=int((fondos / df.iloc[dia,31]) * w1[31])
titulos_32=int((fondos / df.iloc[dia,32]) * w1[32])
titulos_33=int((fondos / df.iloc[dia,33]) * w1[33])
titulos_34=int((fondos / df.iloc[dia,34]) * w1[34])
titulos_35=int((fondos / df.iloc[dia,35]) * w1[35])
titulos_36=int((fondos / df.iloc[dia,36]) * w1[36])
titulos_37=int((fondos / df.iloc[dia,37]) * w1[37])
titulos_38=int((fondos / df.iloc[dia,38]) * w1[38])
titulos_39=int((fondos / df.iloc[dia,39]) * w1[39])
titulos_40=int((fondos / df.iloc[dia,40]) * w1[40])
titulos_41=int((fondos / df.iloc[dia,41]) * w1[41])
titulos_42=int((fondos / df.iloc[dia,42]) * w1[42])
titulos_43=int((fondos / df.iloc[dia,43]) * w1[43])
titulos_44=int((fondos / df.iloc[dia,44]) * w1[44])
titulos_45=int((fondos / df.iloc[dia,45]) * w1[45])
titulos_46=int((fondos / df.iloc[dia,46]) * w1[46])
titulos_47=int((fondos / df.iloc[dia,47]) * w1[47])
titulos_48=int((fondos / df.iloc[dia,48]) * w1[48])
titulos_49=int((fondos / df.iloc[dia,49]) * w1[49])
comision = comision_compra
compra_0= comision * titulos_0  * df.iloc[dia,0]
compra_1= comision * titulos_1  * df.iloc[dia,1]
compra_2= comision * titulos_2  * df.iloc[dia,2]
compra_3= comision * titulos_3  * df.iloc[dia,3]
compra_4= comision * titulos_4  * df.iloc[dia,4]
compra_5= comision * titulos_5  * df.iloc[dia,5]
compra_6= comision * titulos_6  * df.iloc[dia,6]
compra_7= comision * titulos_7  * df.iloc[dia,7]
compra_8= comision * titulos_8  * df.iloc[dia,8]
compra_9= comision * titulos_9  * df.iloc[dia,9]
compra_10= comision * titulos_10  * df.iloc[dia,10]
compra_11= comision * titulos_11  * df.iloc[dia,11]
compra_12= comision * titulos_12  * df.iloc[dia,12]
compra_13= comision * titulos_13  * df.iloc[dia,13]
compra_14= comision * titulos_14  * df.iloc[dia,14]
compra_15= comision * titulos_15  * df.iloc[dia,15]
compra_16= comision * titulos_16  * df.iloc[dia,16]
compra_17= comision * titulos_17  * df.iloc[dia,17]
compra_18= comision * titulos_18  * df.iloc[dia,18]
compra_19= comision * titulos_19  * df.iloc[dia,19]
compra_20= comision * titulos_20  * df.iloc[dia,20]
```

```
compra_21= comision * titulos_21  * df.iloc[dia,21]
compra_22= comision * titulos_22  * df.iloc[dia,22]
compra_23= comision * titulos_23  * df.iloc[dia,23]
compra_24= comision * titulos_24  * df.iloc[dia,24]
compra_25= comision * titulos_25  * df.iloc[dia,25]
compra_26= comision * titulos_26  * df.iloc[dia,26]
compra_27= comision * titulos_27  * df.iloc[dia,27]
compra_28= comision * titulos_28  * df.iloc[dia,28]
compra_29= comision * titulos_29  * df.iloc[dia,29]
compra_30= comision * titulos_30  * df.iloc[dia,30]
compra_31= comision * titulos_31  * df.iloc[dia,31]
compra_32= comision * titulos_32  * df.iloc[dia,32]
compra_33= comision * titulos_33  * df.iloc[dia,33]
compra_34= comision * titulos_34  * df.iloc[dia,34]
compra_35= comision * titulos_35  * df.iloc[dia,35]
compra_36= comision * titulos_36  * df.iloc[dia,36]
compra_37= comision * titulos_37  * df.iloc[dia,37]
compra_38= comision * titulos_38  * df.iloc[dia,38]
compra_39= comision * titulos_39  * df.iloc[dia,39]
compra_40= comision * titulos_40  * df.iloc[dia,40]
compra_41= comision * titulos_41  * df.iloc[dia,41]
compra_42= comision * titulos_42  * df.iloc[dia,42]
compra_43= comision * titulos_43  * df.iloc[dia,43]
compra_44= comision * titulos_44  * df.iloc[dia,44]
compra_45= comision * titulos_45  * df.iloc[dia,45]
compra_46= comision * titulos_46  * df.iloc[dia,46]
compra_47= comision * titulos_47  * df.iloc[dia,47]
compra_48= comision * titulos_48  * df.iloc[dia,48]
compra_49= comision * titulos_49  * df.iloc[dia,49]
comision= comision_venta
venta_0= comision * titulos_0  * df.iloc[dia+dias1,0]
venta_1= comision * titulos_1  * df.iloc[dia+dias1,1]
venta_2= comision * titulos_2  * df.iloc[dia+dias1,2]
venta_3= comision * titulos_3  * df.iloc[dia+dias1,3]
venta_4= comision * titulos_4  * df.iloc[dia+dias1,4]
venta_5= comision * titulos_5  * df.iloc[dia+dias1,5]
venta_6= comision * titulos_6  * df.iloc[dia+dias1,6]
venta_7= comision * titulos_7  * df.iloc[dia+dias1,7]
venta_8= comision * titulos_8  * df.iloc[dia+dias1,8]
venta_9= comision * titulos_9  * df.iloc[dia+dias1,9]
venta_10= comision * titulos_10  * df.iloc[dia+dias1,10]
venta_11= comision * titulos_11  * df.iloc[dia+dias1,11]
venta_12= comision * titulos_12  * df.iloc[dia+dias1,12]
venta_13= comision * titulos_13  * df.iloc[dia+dias1,13]
venta_14= comision * titulos_14  * df.iloc[dia+dias1,14]
venta_15= comision * titulos_15  * df.iloc[dia+dias1,15]
venta_16= comision * titulos_16  * df.iloc[dia+dias1,16]
venta_17= comision * titulos_17  * df.iloc[dia+dias1,17]
venta_18= comision * titulos_18  * df.iloc[dia+dias1,18]
venta_19= comision * titulos_19  * df.iloc[dia+dias1,19]
venta_20= comision * titulos_20  * df.iloc[dia+dias1,20]
venta_21= comision * titulos_21  * df.iloc[dia+dias1,21]
venta_22= comision * titulos_22  * df.iloc[dia+dias1,22]
venta_23= comision * titulos_23  * df.iloc[dia+dias1,23]
venta_24= comision * titulos_24  * df.iloc[dia+dias1,24]
```

```
        venta_25= comision * titulos_25  * df.iloc[dia+dias1,25]
        venta_26= comision * titulos_26  * df.iloc[dia+dias1,26]
        venta_27= comision * titulos_27  * df.iloc[dia+dias1,27]
        venta_28= comision * titulos_28  * df.iloc[dia+dias1,28]
        venta_29= comision * titulos_29  * df.iloc[dia+dias1,29]
        venta_30= comision * titulos_30  * df.iloc[dia+dias1,30]
        venta_31= comision * titulos_31  * df.iloc[dia+dias1,31]
        venta_32= comision * titulos_32  * df.iloc[dia+dias1,32]
        venta_33= comision * titulos_33  * df.iloc[dia+dias1,33]
        venta_34= comision * titulos_34  * df.iloc[dia+dias1,34]
        venta_35= comision * titulos_35  * df.iloc[dia+dias1,35]
        venta_36= comision * titulos_36  * df.iloc[dia+dias1,36]
        venta_37= comision * titulos_37  * df.iloc[dia+dias1,37]
        venta_38= comision * titulos_38  * df.iloc[dia+dias1,38]
        venta_39= comision * titulos_39  * df.iloc[dia+dias1,39]
        venta_40= comision * titulos_40  * df.iloc[dia+dias1,40]
        venta_41= comision * titulos_41  * df.iloc[dia+dias1,41]
        venta_42= comision * titulos_42  * df.iloc[dia+dias1,42]
        venta_43= comision * titulos_43  * df.iloc[dia+dias1,43]
        venta_44= comision * titulos_44  * df.iloc[dia+dias1,44]
        venta_45= comision * titulos_45  * df.iloc[dia+dias1,45]
        venta_46= comision * titulos_46  * df.iloc[dia+dias1,46]
        venta_47= comision * titulos_47  * df.iloc[dia+dias1,47]
        venta_48= comision * titulos_48  * df.iloc[dia+dias1,48]
        venta_49= comision * titulos_49  * df.iloc[dia+dias1,49]
        restante = (fondos - compra_0 - compra_1 - compra_2 - compra_3
- compra_4 - compra_5 - compra_6 - compra_7 - compra_8 - compra_9
- compra_10 - compra_11 - compra_12 - compra_13 - compra_14 - compra_15
- compra_16 - compra_17 - compra_18 - compra_19 - compra_20 - compra_21
- compra_22 - compra_23 - compra_24 - compra_25 - compra_26 - compra_27
- compra_28 - compra_29 - compra_30- compra_31 - compra_32 - compra_33
- compra_34 - compra_35 - compra_36 - compra_37 - compra_38 - compra_39
- compra_40 - compra_41 - compra_42 - compra_43 - compra_44 - compra_45
- compra_46 - compra_47 - compra_48 - compra_49)
        resultado = (restante + venta_0 + venta_1 + venta_2 + venta_3
+ venta_4 + venta_5 + venta_6+ venta_7 + venta_8 + venta_9 + venta_10
+ venta_11 + venta_12 + venta_13 + venta_14+ venta_15 + venta_16
+ venta_17 + venta_18 + venta_19 + venta_20 + venta_21+ venta_22
+ venta_23 + venta_24 + venta_25 + venta_26 + venta_27 + venta_28
+ venta_29 + venta_30 + venta_31 + venta_32 + venta_33 + venta_34
+ venta_35 + venta_36 + venta_37 + venta_38 + venta_39 + venta_40
+ venta_41 + venta_42 + venta_43 + venta_44 + venta_45 + venta_46
+ venta_47 + venta_48 + venta_49 )
        #año 2
        titulos_0=int((resultado / df.iloc[dia+dias1,0]) * w2[0])
        titulos_1=int((resultado / df.iloc[dia+dias1,1]) * w2[1])
        titulos_2=int((resultado / df.iloc[dia+dias1,2]) * w2[2])
        titulos_3=int((resultado / df.iloc[dia+dias1,3]) * w2[3])
        titulos_4=int((resultado / df.iloc[dia+dias1,4]) * w2[4])
        titulos_5=int((resultado / df.iloc[dia+dias1,5]) * w2[5])
        titulos_6=int((resultado / df.iloc[dia+dias1,6]) * w2[6])
        titulos_7=int((resultado / df.iloc[dia+dias1,7]) * w2[7])
        titulos_8=int((resultado / df.iloc[dia+dias1,8]) * w2[8])
        titulos_9=int((resultado / df.iloc[dia+dias1,9]) * w2[9])
        titulos_10=int((resultado / df.iloc[dia+dias1,10]) * w2[10])
```

```
titulos_11=int((resultado / df.iloc[dia+dias1,11]) * w2[11])
titulos_12=int((resultado / df.iloc[dia+dias1,12]) * w2[12])
titulos_13=int((resultado / df.iloc[dia+dias1,13]) * w2[13])
titulos_14=int((resultado / df.iloc[dia+dias1,14]) * w2[14])
titulos_15=int((resultado / df.iloc[dia+dias1,15]) * w2[15])
titulos_16=int((resultado / df.iloc[dia+dias1,16]) * w2[16])
titulos_17=int((resultado / df.iloc[dia+dias1,17]) * w2[17])
titulos_18=int((resultado / df.iloc[dia+dias1,18]) * w2[18])
titulos_19=int((resultado / df.iloc[dia+dias1,19]) * w2[19])
titulos_20=int((resultado / df.iloc[dia+dias1,20]) * w2[20])
titulos_21=int((resultado / df.iloc[dia+dias1,21]) * w2[21])
titulos_22=int((resultado / df.iloc[dia+dias1,22]) * w2[22])
titulos_23=int((resultado / df.iloc[dia+dias1,23]) * w2[23])
titulos_24=int((resultado / df.iloc[dia+dias1,24]) * w2[24])
titulos_25=int((resultado / df.iloc[dia+dias1,25]) * w2[25])
titulos_26=int((resultado / df.iloc[dia+dias1,26]) * w2[26])
titulos_27=int((resultado / df.iloc[dia+dias1,27]) * w2[27])
titulos_28=int((resultado / df.iloc[dia+dias1,28]) * w2[28])
titulos_29=int((resultado / df.iloc[dia+dias1,29]) * w2[29])
titulos_30=int((resultado / df.iloc[dia+dias1,30]) * w2[30])
titulos_31=int((resultado / df.iloc[dia+dias1,31]) * w2[31])
titulos_32=int((resultado / df.iloc[dia+dias1,32]) * w2[32])
titulos_33=int((resultado / df.iloc[dia+dias1,33]) * w2[33])
titulos_34=int((resultado / df.iloc[dia+dias1,34]) * w2[34])
titulos_35=int((resultado / df.iloc[dia+dias1,35]) * w2[35])
titulos_36=int((resultado / df.iloc[dia+dias1,36]) * w2[36])
titulos_37=int((resultado / df.iloc[dia+dias1,37]) * w2[37])
titulos_38=int((resultado / df.iloc[dia+dias1,38]) * w2[38])
titulos_39=int((resultado / df.iloc[dia+dias1,39]) * w2[39])
titulos_40=int((resultado / df.iloc[dia+dias1,40]) * w2[40])
titulos_41=int((resultado / df.iloc[dia+dias1,41]) * w2[41])
titulos_42=int((resultado / df.iloc[dia+dias1,42]) * w2[42])
titulos_43=int((resultado / df.iloc[dia+dias1,43]) * w2[43])
titulos_44=int((resultado / df.iloc[dia+dias1,44]) * w2[44])
titulos_45=int((resultado / df.iloc[dia+dias1,45]) * w2[45])
titulos_46=int((resultado / df.iloc[dia+dias1,46]) * w2[46])
titulos_47=int((resultado / df.iloc[dia+dias1,47]) * w2[47])
titulos_48=int((resultado / df.iloc[dia+dias1,48]) * w2[48])
titulos_49=int((resultado / df.iloc[dia+dias1,49]) * w2[49])
comision = comision_compra
compra_0= comision * titulos_0  * df.iloc[dia+dias1,0]
compra_1= comision * titulos_1  * df.iloc[dia+dias1,1]
compra_2= comision * titulos_2  * df.iloc[dia+dias1,2]
compra_3= comision * titulos_3  * df.iloc[dia+dias1,3]
compra_4= comision * titulos_4  * df.iloc[dia+dias1,4]
compra_5= comision * titulos_5  * df.iloc[dia+dias1,5]
compra_6= comision * titulos_6  * df.iloc[dia+dias1,6]
compra_7= comision * titulos_7  * df.iloc[dia+dias1,7]
compra_8= comision * titulos_8  * df.iloc[dia+dias1,8]
compra_9= comision * titulos_9  * df.iloc[dia+dias1,9]
compra_10= comision * titulos_10  * df.iloc[dia+dias1,10]
compra_11= comision * titulos_11  * df.iloc[dia+dias1,11]
compra_12= comision * titulos_12  * df.iloc[dia+dias1,12]
compra_13= comision * titulos_13  * df.iloc[dia+dias1,13]
compra_14= comision * titulos_14  * df.iloc[dia+dias1,14]
```

```
compra_15= comision * titulos_15  * df.iloc[dia+dias1,15]
compra_16= comision * titulos_16  * df.iloc[dia+dias1,16]
compra_17= comision * titulos_17  * df.iloc[dia+dias1,17]
compra_18= comision * titulos_18  * df.iloc[dia+dias1,18]
compra_19= comision * titulos_19  * df.iloc[dia+dias1,19]
compra_20= comision * titulos_20  * df.iloc[dia+dias1,20]
compra_21= comision * titulos_21  * df.iloc[dia+dias1,21]
compra_22= comision * titulos_22  * df.iloc[dia+dias1,22]
compra_23= comision * titulos_23  * df.iloc[dia+dias1,23]
compra_24= comision * titulos_24  * df.iloc[dia+dias1,24]
compra_25= comision * titulos_25  * df.iloc[dia+dias1,25]
compra_26= comision * titulos_26  * df.iloc[dia+dias1,26]
compra_27= comision * titulos_27  * df.iloc[dia+dias1,27]
compra_28= comision * titulos_28  * df.iloc[dia+dias1,28]
compra_29= comision * titulos_29  * df.iloc[dia+dias1,29]
compra_30= comision * titulos_30  * df.iloc[dia+dias1,30]
compra_31= comision * titulos_31  * df.iloc[dia+dias1,31]
compra_32= comision * titulos_32  * df.iloc[dia+dias1,32]
compra_33= comision * titulos_33  * df.iloc[dia+dias1,33]
compra_34= comision * titulos_34  * df.iloc[dia+dias1,34]
compra_35= comision * titulos_35  * df.iloc[dia+dias1,35]
compra_36= comision * titulos_36  * df.iloc[dia+dias1,36]
compra_37= comision * titulos_37  * df.iloc[dia+dias1,37]
compra_38= comision * titulos_38  * df.iloc[dia+dias1,38]
compra_39= comision * titulos_39  * df.iloc[dia+dias1,39]
compra_40= comision * titulos_40  * df.iloc[dia+dias1,40]
compra_41= comision * titulos_41  * df.iloc[dia+dias1,41]
compra_42= comision * titulos_42  * df.iloc[dia+dias1,42]
compra_43= comision * titulos_43  * df.iloc[dia+dias1,43]
compra_44= comision * titulos_44  * df.iloc[dia+dias1,44]
compra_45= comision * titulos_45  * df.iloc[dia+dias1,45]
compra_46= comision * titulos_46  * df.iloc[dia+dias1,46]
compra_47= comision * titulos_47  * df.iloc[dia+dias1,47]
compra_48= comision * titulos_48  * df.iloc[dia+dias1,48]
compra_49= comision * titulos_49  * df.iloc[dia+dias1,49]
comision= comision_venta
venta_0= comision * titulos_0  * df.iloc[dia+dias1+dias2,0]
venta_1= comision * titulos_1  * df.iloc[dia+dias1+dias2,1]
venta_2= comision * titulos_2  * df.iloc[dia+dias1+dias2,2]
venta_3= comision * titulos_3  * df.iloc[dia+dias1+dias2,3]
venta_4= comision * titulos_4  * df.iloc[dia+dias1+dias2,4]
venta_5= comision * titulos_5  * df.iloc[dia+dias1+dias2,5]
venta_6= comision * titulos_6  * df.iloc[dia+dias1+dias2,6]
venta_7= comision * titulos_7  * df.iloc[dia+dias1+dias2,7]
venta_8= comision * titulos_8  * df.iloc[dia+dias1+dias2,8]
venta_9= comision * titulos_9  * df.iloc[dia+dias1+dias2,9]
venta_10= comision * titulos_10  * df.iloc[dia+dias1+dias2,10]
venta_11= comision * titulos_11  * df.iloc[dia+dias1+dias2,11]
venta_12= comision * titulos_12  * df.iloc[dia+dias1+dias2,12]
venta_13= comision * titulos_13  * df.iloc[dia+dias1+dias2,13]
venta_14= comision * titulos_14  * df.iloc[dia+dias1+dias2,14]
venta_15= comision * titulos_15  * df.iloc[dia+dias1+dias2,15]
venta_16= comision * titulos_16  * df.iloc[dia+dias1+dias2,16]
venta_17= comision * titulos_17  * df.iloc[dia+dias1+dias2,17]
venta_18= comision * titulos_18  * df.iloc[dia+dias1+dias2,18]
```

```
        venta_19= comision * titulos_19  * df.iloc[dia+dias1+dias2,19]
        venta_20= comision * titulos_20  * df.iloc[dia+dias1+dias2,20]
        venta_21= comision * titulos_21  * df.iloc[dia+dias1+dias2,21]
        venta_22= comision * titulos_22  * df.iloc[dia+dias1+dias2,22]
        venta_23= comision * titulos_23  * df.iloc[dia+dias1+dias2,23]
        venta_24= comision * titulos_24  * df.iloc[dia+dias1+dias2,24]
        venta_25= comision * titulos_25  * df.iloc[dia+dias1+dias2,25]
        venta_26= comision * titulos_26  * df.iloc[dia+dias1+dias2,26]
        venta_27= comision * titulos_27  * df.iloc[dia+dias1+dias2,27]
        venta_28= comision * titulos_28  * df.iloc[dia+dias1+dias2,28]
        venta_29= comision * titulos_29  * df.iloc[dia+dias1+dias2,29]
        venta_30= comision * titulos_30  * df.iloc[dia+dias1+dias2,30]
        venta_31= comision * titulos_31  * df.iloc[dia+dias1+dias2,31]
        venta_32= comision * titulos_32  * df.iloc[dia+dias1+dias2,32]
        venta_33= comision * titulos_33  * df.iloc[dia+dias1+dias2,33]
        venta_34= comision * titulos_34  * df.iloc[dia+dias1+dias2,34]
        venta_35= comision * titulos_35  * df.iloc[dia+dias1+dias2,35]
        venta_36= comision * titulos_36  * df.iloc[dia+dias1+dias2,36]
        venta_37= comision * titulos_37  * df.iloc[dia+dias1+dias2,37]
        venta_38= comision * titulos_38  * df.iloc[dia+dias1+dias2,38]
        venta_39= comision * titulos_39  * df.iloc[dia+dias1+dias2,39]
        venta_40= comision * titulos_40  * df.iloc[dia+dias1+dias2,40]
        venta_41= comision * titulos_41  * df.iloc[dia+dias1+dias2,41]
        venta_42= comision * titulos_42  * df.iloc[dia+dias1+dias2,42]
        venta_43= comision * titulos_43  * df.iloc[dia+dias1+dias2,43]
        venta_44= comision * titulos_44  * df.iloc[dia+dias1+dias2,44]
        venta_45= comision * titulos_45  * df.iloc[dia+dias1+dias2,45]
        venta_46= comision * titulos_46  * df.iloc[dia+dias1+dias2,46]
        venta_47= comision * titulos_47  * df.iloc[dia+dias1+dias2,47]
        venta_48= comision * titulos_48  * df.iloc[dia+dias1+dias2,48]
        venta_49= comision * titulos_49  * df.iloc[dia+dias1+dias2,49]
        restante = (resultado - compra_0 - compra_1 - compra_2 - compra_3
- compra_4 - compra_5 - compra_6 - compra_7 - compra_8 - compra_9
- compra_10 - compra_11 - compra_12 - compra_13 - compra_14 - compra_15
- compra_16 - compra_17 - compra_18 - compra_19 - compra_20 - compra_21
- compra_22 - compra_23 - compra_24 - compra_25 - compra_26 - compra_27
- compra_28 - compra_29 - compra_30- compra_31 - compra_32 - compra_33
- compra_34 - compra_35 - compra_36 - compra_37 - compra_38 - compra_39
- compra_40 - compra_41 - compra_42 - compra_43 - compra_44 - compra_45
- compra_46 - compra_47 - compra_48 - compra_49)
        resultado = (restante + venta_0 + venta_1 + venta_2 + venta_3
+ venta_4 + venta_5 + venta_6+ venta_7 + venta_8 + venta_9 + venta_10
+ venta_11 + venta_12 + venta_13 + venta_14+ venta_15 + venta_16
+ venta_17 + venta_18 + venta_19 + venta_20 + venta_21+ venta_22
+ venta_23 + venta_24 + venta_25 + venta_26 + venta_27 + venta_28
+ venta_29 + venta_30 + venta_31 + venta_32 + venta_33 + venta_34
+ venta_35 + venta_36 + venta_37 + venta_38 + venta_39 + venta_40
+ venta_41 + venta_42 + venta_43 + venta_44 + venta_45 + venta_46
+ venta_47 + venta_48 + venta_49 )
        #año 3
        titulos_0=int((resultado / df.iloc[dia+dias1,0]) * w3[0])
        titulos_1=int((resultado / df.iloc[dia+dias1,1]) * w3[1])
        titulos_2=int((resultado / df.iloc[dia+dias1,2]) * w3[2])
        titulos_3=int((resultado / df.iloc[dia+dias1,3]) * w3[3])
        titulos_4=int((resultado / df.iloc[dia+dias1,4]) * w3[4])
```

```
titulos_5=int((resultado / df.iloc[dia+dias1,5]) * w3[5])
titulos_6=int((resultado / df.iloc[dia+dias1,6]) * w3[6])
titulos_7=int((resultado / df.iloc[dia+dias1,7]) * w3[7])
titulos_8=int((resultado / df.iloc[dia+dias1,8]) * w3[8])
titulos_9=int((resultado / df.iloc[dia+dias1,9]) * w3[9])
titulos_10=int((resultado / df.iloc[dia+dias1,10]) * w3[10])
titulos_11=int((resultado / df.iloc[dia+dias1,11]) * w3[11])
titulos_12=int((resultado / df.iloc[dia+dias1,12]) * w3[12])
titulos_13=int((resultado / df.iloc[dia+dias1,13]) * w3[13])
titulos_14=int((resultado / df.iloc[dia+dias1,14]) * w3[14])
titulos_15=int((resultado / df.iloc[dia+dias1,15]) * w3[15])
titulos_16=int((resultado / df.iloc[dia+dias1,16]) * w3[16])
titulos_17=int((resultado / df.iloc[dia+dias1,17]) * w3[17])
titulos_18=int((resultado / df.iloc[dia+dias1,18]) * w3[18])
titulos_19=int((resultado / df.iloc[dia+dias1,19]) * w3[19])
titulos_20=int((resultado / df.iloc[dia+dias1,20]) * w3[20])
titulos_21=int((resultado / df.iloc[dia+dias1,21]) * w3[21])
titulos_22=int((resultado / df.iloc[dia+dias1,22]) * w3[22])
titulos_23=int((resultado / df.iloc[dia+dias1,23]) * w3[23])
titulos_24=int((resultado / df.iloc[dia+dias1,24]) * w3[24])
titulos_25=int((resultado / df.iloc[dia+dias1,25]) * w3[25])
titulos_26=int((resultado / df.iloc[dia+dias1,26]) * w3[26])
titulos_27=int((resultado / df.iloc[dia+dias1,27]) * w3[27])
titulos_28=int((resultado / df.iloc[dia+dias1,28]) * w3[28])
titulos_29=int((resultado / df.iloc[dia+dias1,29]) * w3[29])
titulos_30=int((resultado / df.iloc[dia+dias1,30]) * w3[30])
titulos_31=int((resultado / df.iloc[dia+dias1,31]) * w3[31])
titulos_32=int((resultado / df.iloc[dia+dias1,32]) * w3[32])
titulos_33=int((resultado / df.iloc[dia+dias1,33]) * w3[33])
titulos_34=int((resultado / df.iloc[dia+dias1,34]) * w3[34])
titulos_35=int((resultado / df.iloc[dia+dias1,35]) * w3[35])
titulos_36=int((resultado / df.iloc[dia+dias1,36]) * w3[36])
titulos_37=int((resultado / df.iloc[dia+dias1,37]) * w3[37])
titulos_38=int((resultado / df.iloc[dia+dias1,38]) * w3[38])
titulos_39=int((resultado / df.iloc[dia+dias1,39]) * w3[39])
titulos_40=int((resultado / df.iloc[dia+dias1,40]) * w3[40])
titulos_41=int((resultado / df.iloc[dia+dias1,41]) * w3[41])
titulos_42=int((resultado / df.iloc[dia+dias1,42]) * w3[42])
titulos_43=int((resultado / df.iloc[dia+dias1,43]) * w3[43])
titulos_44=int((resultado / df.iloc[dia+dias1,44]) * w3[44])
titulos_45=int((resultado / df.iloc[dia+dias1,45]) * w3[45])
titulos_46=int((resultado / df.iloc[dia+dias1,46]) * w3[46])
titulos_47=int((resultado / df.iloc[dia+dias1,47]) * w3[47])
titulos_48=int((resultado / df.iloc[dia+dias1,48]) * w3[48])
titulos_49=int((resultado / df.iloc[dia+dias1,49]) * w3[49])
comision = comision_compra
compra_0= comision * titulos_0  * df.iloc[dia+dias1+dias2,0]
compra_1= comision * titulos_1  * df.iloc[dia+dias1+dias2,1]
compra_2= comision * titulos_2  * df.iloc[dia+dias1+dias2,2]
compra_3= comision * titulos_3  * df.iloc[dia+dias1+dias2,3]
compra_4= comision * titulos_4  * df.iloc[dia+dias1+dias2,4]
compra_5= comision * titulos_5  * df.iloc[dia+dias1+dias2,5]
compra_6= comision * titulos_6  * df.iloc[dia+dias1+dias2,6]
compra_7= comision * titulos_7  * df.iloc[dia+dias1+dias2,7]
compra_8= comision * titulos_8  * df.iloc[dia+dias1+dias2,8]
```

```
        compra_9= comision * titulos_9  * df.iloc[dia+dias1+dias2,9]
        compra_10= comision * titulos_10  * df.iloc[dia+dias1+dias2,10]
        compra_11= comision * titulos_11  * df.iloc[dia+dias1+dias2,11]
        compra_12= comision * titulos_12  * df.iloc[dia+dias1+dias2,12]
        compra_13= comision * titulos_13  * df.iloc[dia+dias1+dias2,13]
        compra_14= comision * titulos_14  * df.iloc[dia+dias1+dias2,14]
        compra_15= comision * titulos_15  * df.iloc[dia+dias1+dias2,15]
        compra_16= comision * titulos_16  * df.iloc[dia+dias1+dias2,16]
        compra_17= comision * titulos_17  * df.iloc[dia+dias1+dias2,17]
        compra_18= comision * titulos_18  * df.iloc[dia+dias1+dias2,18]
        compra_19= comision * titulos_19  * df.iloc[dia+dias1+dias2,19]
        compra_20= comision * titulos_20  * df.iloc[dia+dias1+dias2,20]
        compra_21= comision * titulos_21  * df.iloc[dia+dias1+dias2,21]
        compra_22= comision * titulos_22  * df.iloc[dia+dias1+dias2,22]
        compra_23= comision * titulos_23  * df.iloc[dia+dias1+dias2,23]
        compra_24= comision * titulos_24  * df.iloc[dia+dias1+dias2,24]
        compra_25= comision * titulos_25  * df.iloc[dia+dias1+dias2,25]
        compra_26= comision * titulos_26  * df.iloc[dia+dias1+dias2,26]
        compra_27= comision * titulos_27  * df.iloc[dia+dias1+dias2,27]
        compra_28= comision * titulos_28  * df.iloc[dia+dias1+dias2,28]
        compra_29= comision * titulos_29  * df.iloc[dia+dias1+dias2,29]
        compra_30= comision * titulos_30  * df.iloc[dia+dias1+dias2,30]
        compra_31= comision * titulos_31  * df.iloc[dia+dias1+dias2,31]
        compra_32= comision * titulos_32  * df.iloc[dia+dias1+dias2,32]
        compra_33= comision * titulos_33  * df.iloc[dia+dias1+dias2,33]
        compra_34= comision * titulos_34  * df.iloc[dia+dias1+dias2,34]
        compra_35= comision * titulos_35  * df.iloc[dia+dias1+dias2,35]
        compra_36= comision * titulos_36  * df.iloc[dia+dias1+dias2,36]
        compra_37= comision * titulos_37  * df.iloc[dia+dias1+dias2,37]
        compra_38= comision * titulos_38  * df.iloc[dia+dias1+dias2,38]
        compra_39= comision * titulos_39  * df.iloc[dia+dias1+dias2,39]
        compra_40= comision * titulos_40  * df.iloc[dia+dias1+dias2,40]
        compra_41= comision * titulos_41  * df.iloc[dia+dias1+dias2,41]
        compra_42= comision * titulos_42  * df.iloc[dia+dias1+dias2,42]
        compra_43= comision * titulos_43  * df.iloc[dia+dias1+dias2,43]
        compra_44= comision * titulos_44  * df.iloc[dia+dias1+dias2,44]
        compra_45= comision * titulos_45  * df.iloc[dia+dias1+dias2,45]
        compra_46= comision * titulos_46  * df.iloc[dia+dias1+dias2,46]
        compra_47= comision * titulos_47  * df.iloc[dia+dias1+dias2,47]
        compra_48= comision * titulos_48  * df.iloc[dia+dias1+dias2,48]
        compra_49= comision * titulos_49  * df.iloc[dia+dias1+dias2,49]
        comision= comision_venta
        venta_0= comision * titulos_0  * df.iloc[dia+dias1+dias2+dias3,0]
        venta_1= comision * titulos_1  * df.iloc[dia+dias1+dias2+dias3,1]
        venta_2= comision * titulos_2  * df.iloc[dia+dias1+dias2+dias3,2]
        venta_3= comision * titulos_3  * df.iloc[dia+dias1+dias2+dias3,3]
        venta_4= comision * titulos_4  * df.iloc[dia+dias1+dias2+dias3,4]
        venta_5= comision * titulos_5  * df.iloc[dia+dias1+dias2+dias3,5]
        venta_6= comision * titulos_6  * df.iloc[dia+dias1+dias2+dias3,6]
        venta_7= comision * titulos_7  * df.iloc[dia+dias1+dias2+dias3,7]
        venta_8= comision * titulos_8  * df.iloc[dia+dias1+dias2+dias3,8]
        venta_9= comision * titulos_9  * df.iloc[dia+dias1+dias2+dias3,9]
        venta_10= comision * titulos_10  * df.iloc[dia+dias1+dias2+dias3,
10]
        venta_11= comision * titulos_11  * df.iloc[dia+dias1+dias2+dias3,
```

```
11]
        venta_12= comision * titulos_12  * df.iloc[dia+dias1+dias2+dias3,
12]
        venta_13= comision * titulos_13  * df.iloc[dia+dias1+dias2+dias3,
13]
        venta_14= comision * titulos_14  * df.iloc[dia+dias1+dias2+dias3,
14]
        venta_15= comision * titulos_15  * df.iloc[dia+dias1+dias2+dias3,
15]
        venta_16= comision * titulos_16  * df.iloc[dia+dias1+dias2+dias3,
16]
        venta_17= comision * titulos_17  * df.iloc[dia+dias1+dias2+dias3,
17]
        venta_18= comision * titulos_18  * df.iloc[dia+dias1+dias2+dias3,
18]
        venta_19= comision * titulos_19  * df.iloc[dia+dias1+dias2+dias3,
19]
        venta_20= comision * titulos_20  * df.iloc[dia+dias1+dias2+dias3,
20]
        venta_21= comision * titulos_21  * df.iloc[dia+dias1+dias2+dias3,
21]
        venta_22= comision * titulos_22  * df.iloc[dia+dias1+dias2+dias3,
22]
        venta_23= comision * titulos_23  * df.iloc[dia+dias1+dias2+dias3,
23]
        venta_24= comision * titulos_24  * df.iloc[dia+dias1+dias2+dias3,
24]
        venta_25= comision * titulos_25  * df.iloc[dia+dias1+dias2+dias3,
25]
        venta_26= comision * titulos_26  * df.iloc[dia+dias1+dias2+dias3,
26]
        venta_27= comision * titulos_27  * df.iloc[dia+dias1+dias2+dias3,
27]
        venta_28= comision * titulos_28  * df.iloc[dia+dias1+dias2+dias3,
28]
        venta_29= comision * titulos_29  * df.iloc[dia+dias1+dias2+dias3,
29]
        venta_30= comision * titulos_30  * df.iloc[dia+dias1+dias2+dias3,
30]
        venta_31= comision * titulos_31  * df.iloc[dia+dias1+dias2+dias3,
31]
        venta_32= comision * titulos_32  * df.iloc[dia+dias1+dias2+dias3,
32]
        venta_33= comision * titulos_33  * df.iloc[dia+dias1+dias2+dias3,
33]
        venta_34= comision * titulos_34  * df.iloc[dia+dias1+dias2+dias3,
34]
        venta_35= comision * titulos_35  * df.iloc[dia+dias1+dias2+dias3,
35]
        venta_36= comision * titulos_36  * df.iloc[dia+dias1+dias2+dias3,
36]
        venta_37= comision * titulos_37  * df.iloc[dia+dias1+dias2+dias3,
37]
        venta_38= comision * titulos_38  * df.iloc[dia+dias1+dias2+dias3,
38]
```

```
        venta_39= comision * titulos_39  * df.iloc[dia+dias1+dias2+dias3,
39]
        venta_40= comision * titulos_40  * df.iloc[dia+dias1+dias2+dias3,
40]
        venta_41= comision * titulos_41  * df.iloc[dia+dias1+dias2+dias3,
41]
        venta_42= comision * titulos_42  * df.iloc[dia+dias1+dias2+dias3,
42]
        venta_43= comision * titulos_43  * df.iloc[dia+dias1+dias2+dias3,
43]
        venta_44= comision * titulos_44  * df.iloc[dia+dias1+dias2+dias3,
44]
        venta_45= comision * titulos_45  * df.iloc[dia+dias1+dias2+dias3,
45]
        venta_46= comision * titulos_46  * df.iloc[dia+dias1+dias2+dias3,
46]
        venta_47= comision * titulos_47  * df.iloc[dia+dias1+dias2+dias3,
47]
        venta_48= comision * titulos_48  * df.iloc[dia+dias1+dias2+dias3,
48]
        venta_49= comision * titulos_49  * df.iloc[dia+dias1+dias2+dias3,
49]
        restante = (resultado - compra_0 - compra_1 - compra_2 - compra_3
- compra_4 - compra_5 - compra_6 - compra_7 - compra_8 - compra_9
- compra_10 - compra_11 - compra_12 - compra_13 - compra_14 - compra_15
- compra_16 - compra_17 - compra_18 - compra_19 - compra_20 - compra_21
- compra_22 - compra_23 - compra_24 - compra_25 - compra_26 - compra_27
- compra_28 - compra_29 - compra_30- compra_31 - compra_32 - compra_33
- compra_34 - compra_35 - compra_36 - compra_37 - compra_38 - compra_39
- compra_40 - compra_41 - compra_42 - compra_43 - compra_44 - compra_45
- compra_46 - compra_47 - compra_48 - compra_49)
        resultado = (restante + venta_0 + venta_1 + venta_2 + venta_3
+ venta_4 + venta_5 + venta_6+ venta_7 + venta_8 + venta_9 + venta_10
+ venta_11 + venta_12 + venta_13 + venta_14+ venta_15 + venta_16
+ venta_17 + venta_18 + venta_19 + venta_20 + venta_21+ venta_22
+ venta_23 + venta_24 + venta_25 + venta_26 + venta_27 + venta_28
+ venta_29 + venta_30 + venta_31 + venta_32 + venta_33 + venta_34
+ venta_35 + venta_36 + venta_37 + venta_38 + venta_39 + venta_40
+ venta_41 + venta_42 + venta_43 + venta_44 + venta_45 + venta_46
+ venta_47 + venta_48 + venta_49 )
        #año 4
        titulos_0=int((resultado / df.iloc[dia+dias1,0]) * w4[0])
        titulos_1=int((resultado / df.iloc[dia+dias1,1]) * w4[1])
        titulos_2=int((resultado / df.iloc[dia+dias1,2]) * w4[2])
        titulos_3=int((resultado / df.iloc[dia+dias1,3]) * w4[3])
        titulos_4=int((resultado / df.iloc[dia+dias1,4]) * w4[4])
        titulos_5=int((resultado / df.iloc[dia+dias1,5]) * w4[5])
        titulos_6=int((resultado / df.iloc[dia+dias1,6]) * w4[6])
        titulos_7=int((resultado / df.iloc[dia+dias1,7]) * w4[7])
        titulos_8=int((resultado / df.iloc[dia+dias1,8]) * w4[8])
        titulos_9=int((resultado / df.iloc[dia+dias1,9]) * w4[9])
        titulos_10=int((resultado / df.iloc[dia+dias1,10]) * w4[10])
        titulos_11=int((resultado / df.iloc[dia+dias1,11]) * w4[11])
        titulos_12=int((resultado / df.iloc[dia+dias1,12]) * w4[12])
        titulos_13=int((resultado / df.iloc[dia+dias1,13]) * w4[13])
```

```
        titulos_14=int((resultado / df.iloc[dia+dias1,14]) * w4[14])
        titulos_15=int((resultado / df.iloc[dia+dias1,15]) * w4[15])
        titulos_16=int((resultado / df.iloc[dia+dias1,16]) * w4[16])
        titulos_17=int((resultado / df.iloc[dia+dias1,17]) * w4[17])
        titulos_18=int((resultado / df.iloc[dia+dias1,18]) * w4[18])
        titulos_19=int((resultado / df.iloc[dia+dias1,19]) * w4[19])
        titulos_20=int((resultado / df.iloc[dia+dias1,20]) * w4[20])
        titulos_21=int((resultado / df.iloc[dia+dias1,21]) * w4[21])
        titulos_22=int((resultado / df.iloc[dia+dias1,22]) * w4[22])
        titulos_23=int((resultado / df.iloc[dia+dias1,23]) * w4[23])
        titulos_24=int((resultado / df.iloc[dia+dias1,24]) * w4[24])
        titulos_25=int((resultado / df.iloc[dia+dias1,25]) * w4[25])
        titulos_26=int((resultado / df.iloc[dia+dias1,26]) * w4[26])
        titulos_27=int((resultado / df.iloc[dia+dias1,27]) * w4[27])
        titulos_28=int((resultado / df.iloc[dia+dias1,28]) * w4[28])
        titulos_29=int((resultado / df.iloc[dia+dias1,29]) * w4[29])
        titulos_30=int((resultado / df.iloc[dia+dias1,30]) * w4[30])
        titulos_31=int((resultado / df.iloc[dia+dias1,31]) * w4[31])
        titulos_32=int((resultado / df.iloc[dia+dias1,32]) * w4[32])
        titulos_33=int((resultado / df.iloc[dia+dias1,33]) * w4[33])
        titulos_34=int((resultado / df.iloc[dia+dias1,34]) * w4[34])
        titulos_35=int((resultado / df.iloc[dia+dias1,35]) * w4[35])
        titulos_36=int((resultado / df.iloc[dia+dias1,36]) * w4[36])
        titulos_37=int((resultado / df.iloc[dia+dias1,37]) * w4[37])
        titulos_38=int((resultado / df.iloc[dia+dias1,38]) * w4[38])
        titulos_39=int((resultado / df.iloc[dia+dias1,39]) * w4[39])
        titulos_40=int((resultado / df.iloc[dia+dias1,40]) * w4[40])
        titulos_41=int((resultado / df.iloc[dia+dias1,41]) * w4[41])
        titulos_42=int((resultado / df.iloc[dia+dias1,42]) * w4[42])
        titulos_43=int((resultado / df.iloc[dia+dias1,43]) * w4[43])
        titulos_44=int((resultado / df.iloc[dia+dias1,44]) * w4[44])
        titulos_45=int((resultado / df.iloc[dia+dias1,45]) * w4[45])
        titulos_46=int((resultado / df.iloc[dia+dias1,46]) * w4[46])
        titulos_47=int((resultado / df.iloc[dia+dias1,47]) * w4[47])
        titulos_48=int((resultado / df.iloc[dia+dias1,48]) * w4[48])
        titulos_49=int((resultado / df.iloc[dia+dias1,49]) * w4[49])
        comision = comision_compra
        compra_0= comision * titulos_0  * df.iloc[dia+dias1+dias2+dias3,0]
        compra_1= comision * titulos_1  * df.iloc[dia+dias1+dias2+dias3,1]
        compra_2= comision * titulos_2  * df.iloc[dia+dias1+dias2+dias3,2]
        compra_3= comision * titulos_3  * df.iloc[dia+dias1+dias2+dias3,3]
        compra_4= comision * titulos_4  * df.iloc[dia+dias1+dias2+dias3,4]
        compra_5= comision * titulos_5  * df.iloc[dia+dias1+dias2+dias3,5]
        compra_6= comision * titulos_6  * df.iloc[dia+dias1+dias2+dias3,6]
        compra_7= comision * titulos_7  * df.iloc[dia+dias1+dias2+dias3,7]
        compra_8= comision * titulos_8  * df.iloc[dia+dias1+dias2+dias3,8]
        compra_9= comision * titulos_9  * df.iloc[dia+dias1+dias2+dias3,9]
        compra_10= comision * titulos_10  * df.iloc[dia+dias1+dias2+dias3,
10]
        compra_11= comision * titulos_11  * df.iloc[dia+dias1+dias2+dias3,
11]
        compra_12= comision * titulos_12  * df.iloc[dia+dias1+dias2+dias3,
12]
        compra_13= comision * titulos_13  * df.iloc[dia+dias1+dias2+dias3,
13]
```

```
        compra_14= comision * titulos_14  * df.iloc[dia+dias1+dias2+dias3,
14]
        compra_15= comision * titulos_15  * df.iloc[dia+dias1+dias2+dias3,
15]
        compra_16= comision * titulos_16  * df.iloc[dia+dias1+dias2+dias3,
16]
        compra_17= comision * titulos_17  * df.iloc[dia+dias1+dias2+dias3,
17]
        compra_18= comision * titulos_18  * df.iloc[dia+dias1+dias2+dias3,
18]
        compra_19= comision * titulos_19  * df.iloc[dia+dias1+dias2+dias3,
19]
        compra_20= comision * titulos_20  * df.iloc[dia+dias1+dias2+dias3,
20]
        compra_21= comision * titulos_21  * df.iloc[dia+dias1+dias2+dias3,
21]
        compra_22= comision * titulos_22  * df.iloc[dia+dias1+dias2+dias3,
22]
        compra_23= comision * titulos_23  * df.iloc[dia+dias1+dias2+dias3,
23]
        compra_24= comision * titulos_24  * df.iloc[dia+dias1+dias2+dias3,
24]
        compra_25= comision * titulos_25  * df.iloc[dia+dias1+dias2+dias3,
25]
        compra_26= comision * titulos_26  * df.iloc[dia+dias1+dias2+dias3,
26]
        compra_27= comision * titulos_27  * df.iloc[dia+dias1+dias2+dias3,
27]
        compra_28= comision * titulos_28  * df.iloc[dia+dias1+dias2+dias3,
28]
        compra_29= comision * titulos_29  * df.iloc[dia+dias1+dias2+dias3,
29]
        compra_30= comision * titulos_30  * df.iloc[dia+dias1+dias2+dias3,
30]
        compra_31= comision * titulos_31  * df.iloc[dia+dias1+dias2+dias3,
31]
        compra_32= comision * titulos_32  * df.iloc[dia+dias1+dias2+dias3,
32]
        compra_33= comision * titulos_33  * df.iloc[dia+dias1+dias2+dias3,
33]
        compra_34= comision * titulos_34  * df.iloc[dia+dias1+dias2+dias3,
34]
        compra_35= comision * titulos_35  * df.iloc[dia+dias1+dias2+dias3,
35]
        compra_36= comision * titulos_36  * df.iloc[dia+dias1+dias2+dias3,
36]
        compra_37= comision * titulos_37  * df.iloc[dia+dias1+dias2+dias3,
37]
        compra_38= comision * titulos_38  * df.iloc[dia+dias1+dias2+dias3,
38]
        compra_39= comision * titulos_39  * df.iloc[dia+dias1+dias2+dias3,
39]
        compra_40= comision * titulos_40  * df.iloc[dia+dias1+dias2+dias3,
40]
```

```
        compra_41= comision * titulos_41  * df.iloc[dia+dias1+dias2+dias3,
41]
        compra_42= comision * titulos_42  * df.iloc[dia+dias1+dias2+dias3,
42]
        compra_43= comision * titulos_43  * df.iloc[dia+dias1+dias2+dias3,
43]
        compra_44= comision * titulos_44  * df.iloc[dia+dias1+dias2+dias3,
44]
        compra_45= comision * titulos_45  * df.iloc[dia+dias1+dias2+dias3,
45]
        compra_46= comision * titulos_46  * df.iloc[dia+dias1+dias2+dias3,
46]
        compra_47= comision * titulos_47  * df.iloc[dia+dias1+dias2+dias3,
47]
        compra_48= comision * titulos_48  * df.iloc[dia+dias1+dias2+dias3,
48]
        compra_49= comision * titulos_49  * df.iloc[dia+dias1+dias2+dias3,
49]
        comision= comision_venta
        venta_0= comision * titulos_0  * df.iloc[dia+dias1+dias2+dias3
+dias4,0]
        venta_1= comision * titulos_1  * df.iloc[dia+dias1+dias2+dias3
+dias4,1]
        venta_2= comision * titulos_2  * df.iloc[dia+dias1+dias2+dias3
+dias4,2]
        venta_3= comision * titulos_3  * df.iloc[dia+dias1+dias2+dias3
+dias4,3]
        venta_4= comision * titulos_4  * df.iloc[dia+dias1+dias2+dias3
+dias4,4]
        venta_5= comision * titulos_5  * df.iloc[dia+dias1+dias2+dias3
+dias4,5]
        venta_6= comision * titulos_6  * df.iloc[dia+dias1+dias2+dias3
+dias4,6]
        venta_7= comision * titulos_7  * df.iloc[dia+dias1+dias2+dias3
+dias4,7]
        venta_8= comision * titulos_8  * df.iloc[dia+dias1+dias2+dias3
+dias4,8]
        venta_9= comision * titulos_9  * df.iloc[dia+dias1+dias2+dias3
+dias4,9]
        venta_10= comision * titulos_10  * df.iloc[dia+dias1+dias2+dias3
+dias4,10]
        venta_11= comision * titulos_11  * df.iloc[dia+dias1+dias2+dias3
+dias4,11]
        venta_12= comision * titulos_12  * df.iloc[dia+dias1+dias2+dias3
+dias4,12]
        venta_13= comision * titulos_13  * df.iloc[dia+dias1+dias2+dias3
+dias4,13]
        venta_14= comision * titulos_14  * df.iloc[dia+dias1+dias2+dias3
+dias4,14]
        venta_15= comision * titulos_15  * df.iloc[dia+dias1+dias2+dias3
+dias4,15]
        venta_16= comision * titulos_16  * df.iloc[dia+dias1+dias2+dias3
+dias4,16]
        venta_17= comision * titulos_17  * df.iloc[dia+dias1+dias2+dias3
+dias4,17]
```

```
        venta_18= comision * titulos_18  * df.iloc[dia+dias1+dias2+dias3
+dias4,18]
        venta_19= comision * titulos_19  * df.iloc[dia+dias1+dias2+dias3
+dias4,19]
        venta_20= comision * titulos_20  * df.iloc[dia+dias1+dias2+dias3
+dias4,20]
        venta_21= comision * titulos_21  * df.iloc[dia+dias1+dias2+dias3
+dias4,21]
        venta_22= comision * titulos_22  * df.iloc[dia+dias1+dias2+dias3
+dias4,22]
        venta_23= comision * titulos_23  * df.iloc[dia+dias1+dias2+dias3
+dias4,23]
        venta_24= comision * titulos_24  * df.iloc[dia+dias1+dias2+dias3
+dias4,24]
        venta_25= comision * titulos_25  * df.iloc[dia+dias1+dias2+dias3
+dias4,25]
        venta_26= comision * titulos_26  * df.iloc[dia+dias1+dias2+dias3
+dias4,26]
        venta_27= comision * titulos_27  * df.iloc[dia+dias1+dias2+dias3
+dias4,27]
        venta_28= comision * titulos_28  * df.iloc[dia+dias1+dias2+dias3
+dias4,28]
        venta_29= comision * titulos_29  * df.iloc[dia+dias1+dias2+dias3
+dias4,29]
        venta_30= comision * titulos_30  * df.iloc[dia+dias1+dias2+dias3
+dias4,30]
        venta_31= comision * titulos_31  * df.iloc[dia+dias1+dias2+dias3
+dias4,31]
        venta_32= comision * titulos_32  * df.iloc[dia+dias1+dias2+dias3
+dias4,32]
        venta_33= comision * titulos_33  * df.iloc[dia+dias1+dias2+dias3
+dias4,33]
        venta_34= comision * titulos_34  * df.iloc[dia+dias1+dias2+dias3
+dias4,34]
        venta_35= comision * titulos_35  * df.iloc[dia+dias1+dias2+dias3
+dias4,35]
        venta_36= comision * titulos_36  * df.iloc[dia+dias1+dias2+dias3
+dias4,36]
        venta_37= comision * titulos_37  * df.iloc[dia+dias1+dias2+dias3
+dias4,37]
        venta_38= comision * titulos_38  * df.iloc[dia+dias1+dias2+dias3
+dias4,38]
        venta_39= comision * titulos_39  * df.iloc[dia+dias1+dias2+dias3
+dias4,39]
        venta_40= comision * titulos_40  * df.iloc[dia+dias1+dias2+dias3
+dias4,40]
        venta_41= comision * titulos_41  * df.iloc[dia+dias1+dias2+dias3
+dias4,41]
        venta_42= comision * titulos_42  * df.iloc[dia+dias1+dias2+dias3
+dias4,42]
        venta_43= comision * titulos_43  * df.iloc[dia+dias1+dias2+dias3
+dias4,43]
        venta_44= comision * titulos_44  * df.iloc[dia+dias1+dias2+dias3
+dias4,44]
        venta_45= comision * titulos_45  * df.iloc[dia+dias1+dias2+dias3
```

```
+dias4,45]
        venta_46= comision * titulos_46  * df.iloc[dia+dias1+dias2+dias3
+dias4,46]
        venta_47= comision * titulos_47  * df.iloc[dia+dias1+dias2+dias3
+dias4,47]
        venta_48= comision * titulos_48  * df.iloc[dia+dias1+dias2+dias3
+dias4,48]
        venta_49= comision * titulos_49  * df.iloc[dia+dias1+dias2+dias3
+dias4,49]
        restante = (resultado - compra_0 - compra_1 - compra_2 - compra_3
- compra_4 - compra_5 - compra_6 - compra_7 - compra_8 - compra_9
- compra_10 - compra_11 - compra_12 - compra_13 - compra_14 - compra_15
- compra_16 - compra_17 - compra_18 - compra_19 - compra_20 - compra_21
- compra_22 - compra_23 - compra_24 - compra_25 - compra_26 - compra_27
- compra_28 - compra_29 - compra_30- compra_31 - compra_32 - compra_33
- compra_34 - compra_35 - compra_36 - compra_37 - compra_38 - compra_39
- compra_40 - compra_41 - compra_42 - compra_43 - compra_44 - compra_45
- compra_46 - compra_47 - compra_48 - compra_49)
        resultado = (restante + venta_0 + venta_1 + venta_2 + venta_3
+ venta_4 + venta_5 + venta_6+ venta_7 + venta_8 + venta_9 + venta_10
+ venta_11 + venta_12 + venta_13 + venta_14+ venta_15 + venta_16
+ venta_17 + venta_18 + venta_19 + venta_20 + venta_21+ venta_22
+ venta_23 + venta_24 + venta_25 + venta_26 + venta_27 + venta_28
+ venta_29 + venta_30 + venta_31 + venta_32 + venta_33 + venta_34
+ venta_35 + venta_36 + venta_37 + venta_38 + venta_39 + venta_40
+ venta_41 + venta_42 + venta_43 + venta_44 + venta_45 + venta_46
+ venta_47 + venta_48 + venta_49 )
        #año 5
        titulos_0=int((resultado / df.iloc[dia+dias1,0]) * w5[0])
        titulos_1=int((resultado / df.iloc[dia+dias1,1]) * w5[1])
        titulos_2=int((resultado / df.iloc[dia+dias1,2]) * w5[2])
        titulos_3=int((resultado / df.iloc[dia+dias1,3]) * w5[3])
        titulos_4=int((resultado / df.iloc[dia+dias1,4]) * w5[4])
        titulos_5=int((resultado / df.iloc[dia+dias1,5]) * w5[5])
        titulos_6=int((resultado / df.iloc[dia+dias1,6]) * w5[6])
        titulos_7=int((resultado / df.iloc[dia+dias1,7]) * w5[7])
        titulos_8=int((resultado / df.iloc[dia+dias1,8]) * w5[8])
        titulos_9=int((resultado / df.iloc[dia+dias1,9]) * w5[9])
        titulos_10=int((resultado / df.iloc[dia+dias1,10]) * w5[10])
        titulos_11=int((resultado / df.iloc[dia+dias1,11]) * w5[11])
        titulos_12=int((resultado / df.iloc[dia+dias1,12]) * w5[12])
        titulos_13=int((resultado / df.iloc[dia+dias1,13]) * w5[13])
        titulos_14=int((resultado / df.iloc[dia+dias1,14]) * w5[14])
        titulos_15=int((resultado / df.iloc[dia+dias1,15]) * w5[15])
        titulos_16=int((resultado / df.iloc[dia+dias1,16]) * w5[16])
        titulos_17=int((resultado / df.iloc[dia+dias1,17]) * w5[17])
        titulos_18=int((resultado / df.iloc[dia+dias1,18]) * w5[18])
        titulos_19=int((resultado / df.iloc[dia+dias1,19]) * w5[19])
        titulos_20=int((resultado / df.iloc[dia+dias1,20]) * w5[20])
        titulos_21=int((resultado / df.iloc[dia+dias1,21]) * w5[21])
        titulos_22=int((resultado / df.iloc[dia+dias1,22]) * w5[22])
        titulos_23=int((resultado / df.iloc[dia+dias1,23]) * w5[23])
        titulos_24=int((resultado / df.iloc[dia+dias1,24]) * w5[24])
        titulos_25=int((resultado / df.iloc[dia+dias1,25]) * w5[25])
        titulos_26=int((resultado / df.iloc[dia+dias1,26]) * w5[26])
```

```
        titulos_27=int((resultado / df.iloc[dia+dias1,27]) * w5[27])
        titulos_28=int((resultado / df.iloc[dia+dias1,28]) * w5[28])
        titulos_29=int((resultado / df.iloc[dia+dias1,29]) * w5[29])
        titulos_30=int((resultado / df.iloc[dia+dias1,30]) * w5[30])
        titulos_31=int((resultado / df.iloc[dia+dias1,31]) * w5[31])
        titulos_32=int((resultado / df.iloc[dia+dias1,32]) * w5[32])
        titulos_33=int((resultado / df.iloc[dia+dias1,33]) * w5[33])
        titulos_34=int((resultado / df.iloc[dia+dias1,34]) * w5[34])
        titulos_35=int((resultado / df.iloc[dia+dias1,35]) * w5[35])
        titulos_36=int((resultado / df.iloc[dia+dias1,36]) * w5[36])
        titulos_37=int((resultado / df.iloc[dia+dias1,37]) * w5[37])
        titulos_38=int((resultado / df.iloc[dia+dias1,38]) * w5[38])
        titulos_39=int((resultado / df.iloc[dia+dias1,39]) * w5[39])
        titulos_40=int((resultado / df.iloc[dia+dias1,40]) * w5[40])
        titulos_41=int((resultado / df.iloc[dia+dias1,41]) * w5[41])
        titulos_42=int((resultado / df.iloc[dia+dias1,42]) * w5[42])
        titulos_43=int((resultado / df.iloc[dia+dias1,43]) * w5[43])
        titulos_44=int((resultado / df.iloc[dia+dias1,44]) * w5[44])
        titulos_45=int((resultado / df.iloc[dia+dias1,45]) * w5[45])
        titulos_46=int((resultado / df.iloc[dia+dias1,46]) * w5[46])
        titulos_47=int((resultado / df.iloc[dia+dias1,47]) * w5[47])
        titulos_48=int((resultado / df.iloc[dia+dias1,48]) * w5[48])
        titulos_49=int((resultado / df.iloc[dia+dias1,49]) * w5[49])
        comision = comision_compra
        compra_0= comision * titulos_0  * df.iloc[dia+dias1+dias2+dias3
+dias4,0]
        compra_1= comision * titulos_1  * df.iloc[dia+dias1+dias2+dias3
+dias4,1]
        compra_2= comision * titulos_2  * df.iloc[dia+dias1+dias2+dias3
+dias4,2]
        compra_3= comision * titulos_3  * df.iloc[dia+dias1+dias2+dias3
+dias4,3]
        compra_4= comision * titulos_4  * df.iloc[dia+dias1+dias2+dias3
+dias4,4]
        compra_5= comision * titulos_5  * df.iloc[dia+dias1+dias2+dias3
+dias4,5]
        compra_6= comision * titulos_6  * df.iloc[dia+dias1+dias2+dias3
+dias4,6]
        compra_7= comision * titulos_7  * df.iloc[dia+dias1+dias2+dias3
+dias4,7]
        compra_8= comision * titulos_8  * df.iloc[dia+dias1+dias2+dias3
+dias4,8]
        compra_9= comision * titulos_9  * df.iloc[dia+dias1+dias2+dias3
+dias4,9]
        compra_10= comision * titulos_10  * df.iloc[dia+dias1+dias2+dias3
+dias4,10]
        compra_11= comision * titulos_11  * df.iloc[dia+dias1+dias2+dias3
+dias4,11]
        compra_12= comision * titulos_12  * df.iloc[dia+dias1+dias2+dias3
+dias4,12]
        compra_13= comision * titulos_13  * df.iloc[dia+dias1+dias2+dias3
+dias4,13]
        compra_14= comision * titulos_14  * df.iloc[dia+dias1+dias2+dias3
+dias4,14]
        compra_15= comision * titulos_15  * df.iloc[dia+dias1+dias2+dias3
```

```
+dias4,15]
        compra_16= comision * titulos_16  * df.iloc[dia+dias1+dias2+dias3
+dias4,16]
        compra_17= comision * titulos_17  * df.iloc[dia+dias1+dias2+dias3
+dias4,17]
        compra_18= comision * titulos_18  * df.iloc[dia+dias1+dias2+dias3
+dias4,18]
        compra_19= comision * titulos_19  * df.iloc[dia+dias1+dias2+dias3
+dias4,19]
        compra_20= comision * titulos_20  * df.iloc[dia+dias1+dias2+dias3
+dias4,20]
        compra_21= comision * titulos_21  * df.iloc[dia+dias1+dias2+dias3
+dias4,21]
        compra_22= comision * titulos_22  * df.iloc[dia+dias1+dias2+dias3
+dias4,22]
        compra_23= comision * titulos_23  * df.iloc[dia+dias1+dias2+dias3
+dias4,23]
        compra_24= comision * titulos_24  * df.iloc[dia+dias1+dias2+dias3
+dias4,24]
        compra_25= comision * titulos_25  * df.iloc[dia+dias1+dias2+dias3
+dias4,25]
        compra_26= comision * titulos_26  * df.iloc[dia+dias1+dias2+dias3
+dias4,26]
        compra_27= comision * titulos_27  * df.iloc[dia+dias1+dias2+dias3
+dias4,27]
        compra_28= comision * titulos_28  * df.iloc[dia+dias1+dias2+dias3
+dias4,28]
        compra_29= comision * titulos_29  * df.iloc[dia+dias1+dias2+dias3
+dias4,29]
        compra_30= comision * titulos_30  * df.iloc[dia+dias1+dias2+dias3
+dias4,30]
        compra_31= comision * titulos_31  * df.iloc[dia+dias1+dias2+dias3
+dias4,31]
        compra_32= comision * titulos_32  * df.iloc[dia+dias1+dias2+dias3
+dias4,32]
        compra_33= comision * titulos_33  * df.iloc[dia+dias1+dias2+dias3
+dias4,33]
        compra_34= comision * titulos_34  * df.iloc[dia+dias1+dias2+dias3
+dias4,34]
        compra_35= comision * titulos_35  * df.iloc[dia+dias1+dias2+dias3
+dias4,35]
        compra_36= comision * titulos_36  * df.iloc[dia+dias1+dias2+dias3
+dias4,36]
        compra_37= comision * titulos_37  * df.iloc[dia+dias1+dias2+dias3
+dias4,37]
        compra_38= comision * titulos_38  * df.iloc[dia+dias1+dias2+dias3
+dias4,38]
        compra_39= comision * titulos_39  * df.iloc[dia+dias1+dias2+dias3
+dias4,39]
        compra_40= comision * titulos_40  * df.iloc[dia+dias1+dias2+dias3
+dias4,40]
        compra_41= comision * titulos_41  * df.iloc[dia+dias1+dias2+dias3
+dias4,41]
        compra_42= comision * titulos_42  * df.iloc[dia+dias1+dias2+dias3
+dias4,42]
```

```
        compra_43= comision * titulos_43  * df.iloc[dia+dias1+dias2+dias3
+dias4,43]
        compra_44= comision * titulos_44  * df.iloc[dia+dias1+dias2+dias3
+dias4,44]
        compra_45= comision * titulos_45  * df.iloc[dia+dias1+dias2+dias3
+dias4,45]
        compra_46= comision * titulos_46  * df.iloc[dia+dias1+dias2+dias3
+dias4,46]
        compra_47= comision * titulos_47  * df.iloc[dia+dias1+dias2+dias3
+dias4,47]
        compra_48= comision * titulos_48  * df.iloc[dia+dias1+dias2+dias3
+dias4,48]
        compra_49= comision * titulos_49  * df.iloc[dia+dias1+dias2+dias3
+dias4,49]
        comision= comision_venta
        venta_0= comision * titulos_0  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,0]
        venta_1= comision * titulos_1  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,1]
        venta_2= comision * titulos_2  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,2]
        venta_3= comision * titulos_3  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,3]
        venta_4= comision * titulos_4  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,4]
        venta_5= comision * titulos_5  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,5]
        venta_6= comision * titulos_6  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,6]
        venta_7= comision * titulos_7  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,7]
        venta_8= comision * titulos_8  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,8]
        venta_9= comision * titulos_9  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,9]
        venta_10= comision * titulos_10  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,10]
        venta_11= comision * titulos_11  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,11]
        venta_12= comision * titulos_12  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,12]
        venta_13= comision * titulos_13  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,13]
        venta_14= comision * titulos_14  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,14]
        venta_15= comision * titulos_15  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,15]
        venta_16= comision * titulos_16  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,16]
        venta_17= comision * titulos_17  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,17]
        venta_18= comision * titulos_18  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,18]
        venta_19= comision * titulos_19  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,19]
```

```
        venta_20= comision * titulos_20  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,20]
        venta_21= comision * titulos_21  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,21]
        venta_22= comision * titulos_22  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,22]
        venta_23= comision * titulos_23  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,23]
        venta_24= comision * titulos_24  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,24]
        venta_25= comision * titulos_25  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,25]
        venta_26= comision * titulos_26  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,26]
        venta_27= comision * titulos_27  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,27]
        venta_28= comision * titulos_28  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,28]
        venta_29= comision * titulos_29  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,29]
        venta_30= comision * titulos_30  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,30]
        venta_31= comision * titulos_31  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,31]
        venta_32= comision * titulos_32  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,32]
        venta_33= comision * titulos_33  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,33]
        venta_34= comision * titulos_34  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,34]
        venta_35= comision * titulos_35  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,35]
        venta_36= comision * titulos_36  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,36]
        venta_37= comision * titulos_37  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,37]
        venta_38= comision * titulos_38  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,38]
        venta_39= comision * titulos_39  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,39]
        venta_40= comision * titulos_40  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,40]
        venta_41= comision * titulos_41  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,41]
        venta_42= comision * titulos_42  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,42]
        venta_43= comision * titulos_43  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,43]
        venta_44= comision * titulos_44  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,44]
        venta_45= comision * titulos_45  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,45]
        venta_46= comision * titulos_46  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,46]
        venta_47= comision * titulos_47  * df.iloc[dia+dias1+dias2+dias3
```

```
+dias4+dias5,47]
        venta_48= comision * titulos_48  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,48]
        venta_49= comision * titulos_49  * df.iloc[dia+dias1+dias2+dias3
+dias4+dias5,49]
        restante = (resultado - compra_0 - compra_1 - compra_2 - compra_3
- compra_4 - compra_5 - compra_6 - compra_7 - compra_8 - compra_9
- compra_10 - compra_11 - compra_12 - compra_13 - compra_14 - compra_15
- compra_16 - compra_17 - compra_18 - compra_19 - compra_20 - compra_21
- compra_22 - compra_23 - compra_24 - compra_25 - compra_26 - compra_27
- compra_28 - compra_29 - compra_30- compra_31 - compra_32 - compra_33
- compra_34 - compra_35 - compra_36 - compra_37 - compra_38 - compra_39
- compra_40 - compra_41 - compra_42 - compra_43 - compra_44 - compra_45
- compra_46 - compra_47 - compra_48 - compra_49)
        resultado = (restante + venta_0 + venta_1 + venta_2 + venta_3
+ venta_4 + venta_5 + venta_6+ venta_7 + venta_8 + venta_9 + venta_10
+ venta_11 + venta_12 + venta_13 + venta_14+ venta_15 + venta_16
+ venta_17 + venta_18 + venta_19 + venta_20 + venta_21+ venta_22
+ venta_23 + venta_24 + venta_25 + venta_26 + venta_27 + venta_28
+ venta_29 + venta_30 + venta_31 + venta_32 + venta_33 + venta_34
+ venta_35 + venta_36 + venta_37 + venta_38 + venta_39 + venta_40
+ venta_41 + venta_42 + venta_43 + venta_44 + venta_45 + venta_46
+ venta_47 + venta_48 + venta_49 )
        rendimiento = np.append(rendimiento, resultado)
    return rendimiento
```

Y este es el simulador para la cartera que sigue al índice EURO STOXX50:

```
def cartera_indice(inicio,fin,dias1,dias2,dias3,dias4,dias5):
    rendimiento=[]
    for dia in range(inicio, fin+1):
        #año 1
        titulos_0=int((fondos / indice.iloc[dia,0]))
        comision = comision_compra
        compra_0= comision * titulos_0  * indice.iloc[dia,0]
        venta_0= comision * titulos_0  * indice.iloc[dia+dias1+dias2+dias3
+dias4+dias5,0]
        restante = (fondos - compra_0)
        resultado = (restante + venta_0 )
        rendimiento = np.append(rendimiento, resultado)
    return rendimiento
```

Una vez tenemos los simuladores de carteras, introducimos los datos para el simulador:

```
inicio1 =246
inicio2 =498
inicio3 =755
inicio4 =1007
inicio5 =1258
inicio6 =1491
inicio7 =1736
inicio8 =1981
inicio9 =2228
inicio10=2472
fin1=497
fin2=754
```

```
fin3=1006
fin4=1257
fin5=1490
fin6=1735
fin7=1980
fin8=2227
fin9=2471
fin10=2713

equilibrada = np.array(cartera(inicio1,fin1,w,w,w,w,w,dias2003,dias2004,
dias2005,dias2006,dias2007))#03
equilibrada = np.append(equilibrada, cartera(inicio2,fin2,w,w,w,w,w,
dias2004,dias2005,dias2006,dias2007,dias2008))#04
equilibrada = np.append(equilibrada, cartera(inicio3,fin3,w,w,w,w,w,
dias2005,dias2006,dias2007,dias2008,dias2009))#05
equilibrada = np.append(equilibrada, cartera(inicio4,fin4,w,w,w,w,w,
dias2006,dias2007,dias2008,dias2009,dias2010))#06
equilibrada = np.append(equilibrada, cartera(inicio5,fin5,w,w,w,w,w,
dias2007,dias2008,dias2009,dias2010,dias2011))#07
equilibrada = np.append(equilibrada, cartera(inicio6,fin6,w,w,w,w,w,
dias2008,dias2009,dias2010,dias2011,dias2012))#08
equilibrada = np.append(equilibrada, cartera(inicio7,fin7,w,w,w,w,w,
dias2009,dias2010,dias2011,dias2012,dias2013))#09
equilibrada = np.append(equilibrada, cartera(inicio8,fin8,w,w,w,w,w,
dias2010,dias2011,dias2012,dias2013,dias2014))#10
equilibrada = np.append(equilibrada, cartera(inicio9,fin9,w,w,w,w,w,
dias2011,dias2012,dias2013,dias2014,dias2015))#11
equilibrada = np.append(equilibrada, cartera(inicio10,fin10,w,w,w,w,w,
dias2012,dias2013,dias2014,dias2015,dias2016))#12

eurostoxx = np.array(cartera_indice(inicio1,fin1,dias2003,dias2004,
dias2005,dias2006,dias2007))#03
eurostoxx = np.append(eurostoxx, cartera_indice(inicio2,fin2,dias2004,
dias2005,dias2006,dias2007,dias2008))#04
eurostoxx = np.append(eurostoxx, cartera_indice(inicio3,fin3,dias2005,
dias2006,dias2007,dias2008,dias2009))#05
eurostoxx = np.append(eurostoxx, cartera_indice(inicio4,fin4,dias2006,
dias2007,dias2008,dias2009,dias2010))#06
eurostoxx = np.append(eurostoxx, cartera_indice(inicio5,fin5,dias2007,
dias2008,dias2009,dias2010,dias2011))#07
eurostoxx = np.append(eurostoxx, cartera_indice(inicio6,fin6,dias2008,
dias2009,dias2010,dias2011,dias2012))#08
eurostoxx = np.append(eurostoxx, cartera_indice(inicio7,fin7,dias2009,
dias2010,dias2011,dias2012,dias2013))#09
eurostoxx = np.append(eurostoxx, cartera_indice(inicio8,fin8,dias2010,
dias2011,dias2012,dias2013,dias2014))#10
eurostoxx = np.append(eurostoxx, cartera_indice(inicio9,fin9,dias2011,
dias2012,dias2013,dias2014,dias2015))#11
eurostoxx = np.append(eurostoxx, cartera_indice(inicio10,fin10,dias2012,
dias2013,dias2014,dias2015,dias2016))#12

sharpe = np.array(cartera(inicio1,fin1,opt_s_w2002,opt_s_w2003,
opt_s_w2004,opt_s_w2005,opt_s_w2006,dias2003,dias2004,dias2005,dias2006,
dias2007))#03
sharpe = np.append(sharpe, cartera(inicio2,fin2,opt_s_w2003,opt_s_w2004,
```

```
opt_s_w2005,opt_s_w2006,opt_s_w2007,dias2004,dias2005,dias2006,
dias2007,dias2008))#04
sharpe = np.append(sharpe, cartera(inicio3,fin3,opt_s_w2004,opt_s_w2005,
opt_s_w2006,opt_s_w2007,opt_s_w2008, dias2005,dias2006,dias2007,dias2008,
dias2009))#05
sharpe = np.append(sharpe, cartera(inicio4,fin4,opt_s_w2005,opt_s_w2006,
opt_s_w2007,opt_s_w2008,opt_s_w2009, dias2006,dias2007,dias2008,
dias2009,dias2010))#06
sharpe = np.append(sharpe, cartera(inicio5,fin5,opt_s_w2006,opt_s_w2007,
opt_s_w2008,opt_s_w2009,opt_s_w2010, dias2007,dias2008,dias2009,
dias2010,dias2011))#07
sharpe = np.append(sharpe, cartera(inicio6,fin6,opt_s_w2007,opt_s_w2008,
opt_s_w2009,opt_s_w2010,opt_s_w2011, dias2008,dias2009,dias2010,
dias2011,dias2012))#08
sharpe = np.append(sharpe, cartera(inicio7,fin7,opt_s_w2008,opt_s_w2009,
opt_s_w2010,opt_s_w2011,opt_s_w2012, dias2009,dias2010,dias2011,
dias2012,dias2013))#09
sharpe = np.append(sharpe, cartera(inicio8,fin8,opt_s_w2009,opt_s_w2010,
opt_s_w2011,opt_s_w2012,opt_s_w2013, dias2010,dias2011,dias2012,
dias2013,dias2014))#10
sharpe = np.append(sharpe, cartera(inicio9,fin9,opt_s_w2010,opt_s_w2011,
opt_s_w2012,opt_s_w2013,opt_s_w2014, dias2011,dias2012,dias2013,
dias2014,dias2015))#11
sharpe = np.append(sharpe, cartera(inicio10,fin10,opt_s_w2011,opt_s_w2012,
opt_s_w2013,opt_s_w2014,opt_s_w2015, dias2012,dias2013,dias2014,
dias2015,dias2016))#12

treynor = np.array(cartera(inicio1,fin1,opt_t_w2002,opt_t_w2003,
opt_t_w2004,opt_t_w2005,opt_t_w2006, dias2003,dias2004,dias2005,dias2006,
dias2007))#03
treynor = np.append(treynor, cartera(inicio2,fin2,opt_t_w2003,opt_t_w2004,
opt_t_w2005,opt_t_w2006,opt_t_w2007, dias2004,dias2005,dias2006,
dias2007,dias2008))#04
treynor = np.append(treynor, cartera(inicio3,fin3,opt_t_w2004,opt_t_w2005,
opt_t_w2006,opt_t_w2007,opt_t_w2008, dias2005,dias2006,dias2007,
dias2008,dias2009))#05
treynor = np.append(treynor, cartera(inicio4,fin4,opt_t_w2005,opt_t_w2006,
opt_t_w2007,opt_t_w2008,opt_t_w2009, dias2006,dias2007,dias2008,
dias2009,dias2010))#06
treynor = np.append(treynor, cartera(inicio5,fin5,opt_t_w2006,opt_t_w2007,
opt_t_w2008,opt_t_w2009,opt_t_w2010, dias2007,dias2008,dias2009,
dias2010,dias2011))#07
treynor = np.append(treynor, cartera(inicio6,fin6,opt_t_w2007,opt_t_w2008,
opt_t_w2009,opt_t_w2010,opt_t_w2011, dias2008,dias2009,dias2010,
dias2011,dias2012))#08
treynor = np.append(treynor, cartera(inicio7,fin7,opt_t_w2008,opt_t_w2009,
opt_t_w2010,opt_t_w2011,opt_t_w2012, dias2009,dias2010,dias2011,
dias2012,dias2013))#09
treynor = np.append(treynor, cartera(inicio8,fin8,opt_t_w2009,opt_t_w2010,
opt_t_w2011,opt_t_w2012,opt_t_w2013, dias2010,dias2011,dias2012,
dias2013,dias2014))#10
treynor = np.append(treynor, cartera(inicio9,fin9,opt_t_w2010,opt_t_w2011,
opt_t_w2012,opt_t_w2013,opt_t_w2014, dias2011,dias2012,dias2013,
dias2014,dias2015))#11
treynor = np.append(treynor, cartera(inicio10,fin10,opt_t_w2011,
```

```
opt_t_w2012,opt_t_w2013,opt_t_w2014,opt_t_w2015, dias2012,dias2013,dias201
4,
dias2015,dias2016))#12

jensen = np.array(cartera(inicio1,fin1,opt_j_w2002,opt_j_w2003,
opt_j_w2004,opt_j_w2005,opt_j_w2006, dias2003,dias2004,dias2005,dias2006,
dias2007))#03
jensen = np.append(jensen, cartera(inicio2,fin2,opt_j_w2003,opt_j_w2004,
opt_j_w2005,opt_j_w2006,opt_j_w2007, dias2004,dias2005,dias2006,
dias2007,dias2008))#04
jensen = np.append(jensen, cartera(inicio3,fin3,opt_j_w2004,opt_j_w2005,
opt_j_w2006,opt_j_w2007,opt_j_w2008, dias2005,dias2006,dias2007,
dias2008,dias2009))#05
jensen = np.append(jensen, cartera(inicio4,fin4,opt_j_w2005,opt_j_w2006,
opt_j_w2007,opt_j_w2008,opt_j_w2009, dias2006,dias2007,dias2008,
dias2009,dias2010))#06
jensen = np.append(jensen, cartera(inicio5,fin5,opt_j_w2006,opt_j_w2007,op
t_j_w2008,opt_j_w2009,opt_j_w2010, dias2007,dias2008,dias2009,dias2010,dia
s2011))#07
jensen = np.append(jensen, cartera(inicio6,fin6,opt_j_w2007,opt_j_w2008,op
t_j_w2009,opt_j_w2010,opt_j_w2011, dias2008,dias2009,dias2010,dias2011,dia
s2012))#08
jensen = np.append(jensen, cartera(inicio7,fin7,opt_j_w2008,opt_j_w2009,op
t_j_w2010,opt_j_w2011,opt_j_w2012, dias2009,dias2010,dias2011,dias2012,dia
s2013))#09
jensen = np.append(jensen, cartera(inicio8,fin8,opt_j_w2009,opt_j_w2010,op
t_j_w2011,opt_j_w2012,opt_j_w2013, dias2010,dias2011,dias2012,dias2013,dia
s2014))#10
jensen = np.append(jensen, cartera(inicio9,fin9,opt_j_w2010,opt_j_w2011,op
t_j_w2012,opt_j_w2013,opt_j_w2014, dias2011,dias2012,dias2013,dias2014,dia
s2015))#11
jensen = np.append(jensen, cartera(inicio10,fin10,opt_j_w2011,opt_j_w2012,
opt_j_w2013,opt_j_w2014,opt_j_w2015, dias2012,dias2013,dias2014,dias2015,d
ias2016))#12
```

Para generar los gráficos utilizados en el estudio, así como la tabla de resultados se ha usado la siguiente parte del código:

```
mpld3.enable_notebook()
#calculos anteriores
rendimiento_porcentual_cartera_eu = (((eurostoxx) - fondos) / fondos) *100
media_cartera_eu = stats.mean(rendimiento_porcentual_cartera_eu)
desv_cartera_eu = stats.stdev(rendimiento_porcentual_cartera_eu)
mediana_eu = np.median(rendimiento_porcentual_cartera_eu)

#histograma acumulativo
ordenado_eu = np.sort(rendimiento_porcentual_cartera_eu)
longitud_eu = np.arange(1, len(ordenado_eu)+1) / len(ordenado_eu)
plt.plot(ordenado_eu, longitud_eu, marker='.', label='Eurostoxx',
color='slategrey')
plt.plot(mediana_eu, 0.5, marker='.', color='r')

#calculos anteriores
rendimiento_porcentual_cartera_eq = (((equilibrada) - fondos) / fondos)
       *100
```

```
media_cartera_eq = stats.mean(rendimiento_porcentual_cartera_eq)
desv_cartera_eq = stats.stdev(rendimiento_porcentual_cartera_eq)
mediana_eq = np.median(rendimiento_porcentual_cartera_eq)

#histograma acumulativo
ordenado_eq = np.sort(rendimiento_porcentual_cartera_eq)
longitud_eq = np.arange(1, len(ordenado_eq)+1) / len(ordenado_eq)
plt.plot(ordenado_eq, longitud_eq, marker='.', label='Equilibrada',
 color='mediumseagreen')
plt.plot(mediana_eq, 0.5, marker='.', color='r')

#calculos anteriores
rendimiento_porcentual_cartera_s2 = (((sharpe) - fondos) / fondos) *100
media_cartera_s2 = stats.mean(rendimiento_porcentual_cartera_s2)
desv_cartera_s2 = stats.stdev(rendimiento_porcentual_cartera_s2)
mediana_s2 = np.median(rendimiento_porcentual_cartera_s2)

#histograma acumulativo
ordenado_s2 = np.sort(rendimiento_porcentual_cartera_s2)
longitud_s2 = np.arange(1, len(ordenado_s2)+1) / len(ordenado_s2)
plt.plot(ordenado_s2, longitud_s2, marker='.', label='Sharpe',
color='lightcoral')
plt.plot(mediana_s2, 0.5, marker='.', color='r')

#calculos anteriores
rendimiento_porcentual_cartera_t2 = (((treynor) - fondos) / fondos) *100
media_cartera_t2 = stats.mean(rendimiento_porcentual_cartera_t2)
desv_cartera_t2 = stats.stdev(rendimiento_porcentual_cartera_t2)
mediana_t2 = np.median(rendimiento_porcentual_cartera_t2)
#histograma acumulativo
ordenado_t2 = np.sort(rendimiento_porcentual_cartera_t2)
longitud_t2 = np.arange(1, len(ordenado_t2)+1) / len(ordenado_t2)
plt.plot(ordenado_t2, longitud_t2, marker='.', label='Treynor', color='c')
plt.plot(mediana_t2, 0.5, marker='.', color='r')
#calculos anteriores
rendimiento_porcentual_cartera_j2 = (((jensen) - fondos) / fondos) *100
media_cartera_j2 = stats.mean(rendimiento_porcentual_cartera_j2)
desv_cartera_j2 = stats.stdev(rendimiento_porcentual_cartera_j2)
mediana_j2 = np.median(rendimiento_porcentual_cartera_j2)

#histograma acumulativo
ordenado_j2 = np.sort(rendimiento_porcentual_cartera_j2)
longitud_j2 = np.arange(1, len(ordenado_j2)+1) / len(ordenado_j2)
plt.plot(ordenado_j2, longitud_j2, marker='.', label='Jensen',
color='mediumorchid')
plt.plot(mediana_j2, 0.5, marker='.', color='r')

plt.xlabel('% Rendimiento')
plt.ylabel('Frecuencia')
plt.legend()
plt.grid(linestyle='--')
plt.savefig('figura9.png')
plt.show()
```

```
#cálculo del porcentaje de ganancia
positivo_eu=[]
negativo_eu=[]
for i in range(0,len(rendimiento_porcentual_cartera_eu)):
    if rendimiento_porcentual_cartera_eu[i]>=0:
        positivo_eu.append(rendimiento_porcentual_cartera_eu[i])
    else:
        negativo_eu.append(rendimiento_porcentual_cartera_eu[i])
pos_eu=len(positivo_eu)/len(rendimiento_porcentual_cartera_eu)

positivo_eq=[]
negativo_eq=[]
for i in range(0,len(rendimiento_porcentual_cartera_eq)):
    if rendimiento_porcentual_cartera_eq[i]>=0:
        positivo_eq.append(rendimiento_porcentual_cartera_eq[i])
    else:
        negativo_eq.append(rendimiento_porcentual_cartera_eq[i])
pos_eq=len(positivo_eq)/len(rendimiento_porcentual_cartera_eq)

positivo_s2=[]
negativo_s2=[]
for i in range(0,len(rendimiento_porcentual_cartera_s2)):
    if rendimiento_porcentual_cartera_s2[i]>=0:
        positivo_s2.append(rendimiento_porcentual_cartera_s2[i])
    else:
        negativo_s2.append(rendimiento_porcentual_cartera_s2[i])
pos_s2=len(positivo_s2)/len(rendimiento_porcentual_cartera_s2)

positivo_t2=[]
negativo_t2=[]
for i in range(0,len(rendimiento_porcentual_cartera_t2)):
    if rendimiento_porcentual_cartera_t2[i]>=0:
        positivo_t2.append(rendimiento_porcentual_cartera_t2[i])
    else:
        negativo_t2.append(rendimiento_porcentual_cartera_t2[i])
pos_t2=len(positivo_t2)/len(rendimiento_porcentual_cartera_t2)

positivo_j2=[]
negativo_j2=[]
for i in range(0,len(rendimiento_porcentual_cartera_j2)):
    if rendimiento_porcentual_cartera_j2[i]>=0:
        positivo_j2.append(rendimiento_porcentual_cartera_j2[i])
    else:
        negativo_j2.append(rendimiento_porcentual_cartera_j2[i])
pos_j2=len(positivo_j2)/len(rendimiento_porcentual_cartera_j2)


#funciones necesarias
def sharpeindice(periodo,rindice,rf,dias):
    '''Ratio de Sharpe para el índice EuroStoxx50'''
    mu =sum(rindice * dias)
    sigma =np.var(periodo) ** (1/2) * dias ** (1/2)
    return (mu - rf) / sigma
def treynorindice(rindice,rf,dias):
    '''Indice de Treynor para el índice EuroStoxx50'''
```

```
    mu =sum(rindice * dias)
    return (mu - rf)
#Ratio de Sharpe
list_s_eu=[sharpeindice(indice_change2003,rindice2003,rf2003,dias2003),
sharpeindice(indice_change2004,rindice2004,rf2004,dias2004),
sharpeindice(indice_change2005,rindice2005,rf2005,dias2005),
          sharpeindice(indice_change2006,rindice2006,rf2006,dias2006),
sharpeindice(indice_change2007,rindice2007,rf2007,dias2007),
sharpeindice(indice_change2008,rindice2008,rf2008,dias2008),
          sharpeindice(indice_change2009,rindice2009,rf2009,dias2009),
sharpeindice(indice_change2010,rindice2010,rf2010,dias2010),
sharpeindice(indice_change2011,rindice2011,rf2011,dias2011),
          sharpeindice(indice_change2012,rindice2012,rf2012,dias2012),
sharpeindice(indice_change2013,rindice2013,rf2013,dias2013),
sharpeindice(indice_change2014,rindice2014,rf2014,dias2014),
          sharpeindice(indice_change2015,rindice2015,rf2015,dias2015)]
sha_eu=np.mean(list_s_eu)
list_s_eq=[sharpe2003(w),sharpe2004(w),sharpe2005(w),sharpe2006(w),
sharpe2007(w),sharpe2008(w),sharpe2009(w),sharpe2010(w),sharpe2011(w),
sharpe2012(w),
       sharpe2013(w),sharpe2014(w),sharpe2015(w)]
sha_eq=np.mean(list_s_eq)
list_s_s2=[sharpe2003(opt_s_w2003),sharpe2004(opt_s_w2004),
sharpe2005(opt_s_w2005),sharpe2006(opt_s_w2006),sharpe2007(opt_s_w2007),
sharpe2008(opt_s_w2008),sharpe2009(opt_s_w2009),sharpe2010(opt_s_w2010),
sharpe2011(opt_s_w2011),sharpe2012(opt_s_w2012),
       sharpe2013(opt_s_w2013),sharpe2014(opt_s_w2014),
sharpe2015(opt_s_w2015)]
sha_s2=np.mean(list_s_s2)
list_s_t2=[sharpe2003(opt_t_w2003),sharpe2004(opt_t_w2004),
sharpe2005(opt_t_w2005),sharpe2006(opt_t_w2006),sharpe2007(opt_t_w2007),
sharpe2008(opt_t_w2008),sharpe2009(opt_t_w2009),sharpe2010(opt_t_w2010),
sharpe2011(opt_t_w2011),sharpe2012(opt_t_w2012),
       sharpe2013(opt_t_w2013),sharpe2014(opt_t_w2014),
sharpe2015(opt_t_w2015)]
sha_t2=np.mean(list_s_t2)
list_s_j2=[sharpe2003(opt_j_w2003),sharpe2004(opt_j_w2004),
sharpe2005(opt_j_w2005),sharpe2006(opt_j_w2006),sharpe2007(opt_j_w2007),
sharpe2008(opt_j_w2008),sharpe2009(opt_j_w2009),sharpe2010(opt_j_w2010),
sharpe2011(opt_j_w2011),sharpe2012(opt_j_w2012),
       sharpe2013(opt_j_w2013),sharpe2014(opt_j_w2014),
sharpe2015(opt_j_w2015)]
sha_j2=np.mean(list_s_j2)
#Índice de Treynor
list_t_eu=[treynorindice(rindice2003,rf2003,dias2003),
treynorindice(rindice2004,rf2004,dias2004),treynorindice(rindice2005,
rf2005,dias2005),treynorindice(rindice2006,rf2006,dias2006),
    treynorindice(rindice2007,rf2007,dias2007),
treynorindice(rindice2008,rf2008,dias2008),
treynorindice(rindice2009,rf2009,dias2009),
treynorindice(rindice2010,rf2010,dias2010),
    treynorindice(rindice2011,rf2011,dias2011),
treynorindice(rindice2012,rf2012,dias2012),
treynorindice(rindice2013,rf2013,dias2013),
treynorindice(rindice2014,rf2014,dias2014),
```

```
    treynorindice(rindice2015,rf2015,dias2015)]
tre_eu=np.mean(list_t_eu)
list_t_eq=[treynor2003(w),treynor2004(w),treynor2005(w),treynor2006(w),
treynor2007(w),treynor2008(w),treynor2009(w),treynor2010(w),
treynor2011(w),treynor2012(w),
       treynor2013(w),treynor2014(w),treynor2015(w)]
tre_eq=np.mean(list_t_eq)
list_t_s2=[treynor2003(opt_s_w2003),treynor2004(opt_s_w2004),
treynor2005(opt_s_w2005),treynor2006(opt_s_w2006),
treynor2007(opt_s_w2007),treynor2008(opt_s_w2008),
treynor2009(opt_s_w2009),treynor2010(opt_s_w2010),
treynor2011(opt_s_w2011),treynor2012(opt_s_w2012),
       treynor2013(opt_s_w2013),treynor2014(opt_s_w2014),
treynor2015(opt_s_w2015)]
tre_s2=np.mean(list_t_s2)
list_t_t2=[treynor2003(opt_t_w2003),treynor2004(opt_t_w2004),
treynor2005(opt_t_w2005),treynor2006(opt_t_w2006),
treynor2007(opt_t_w2007),treynor2008(opt_t_w2008),
treynor2009(opt_t_w2009),treynor2010(opt_t_w2010),
treynor2011(opt_t_w2011),treynor2012(opt_t_w2012),
       treynor2013(opt_t_w2013),treynor2014(opt_t_w2014),
treynor2015(opt_t_w2015)]
tre_t2=np.mean(list_t_t2)
list_t_j2=[treynor2003(opt_j_w2003),treynor2004(opt_j_w2004),
treynor2005(opt_j_w2005),treynor2006(opt_j_w2006),
treynor2007(opt_j_w2007),treynor2008(opt_j_w2008),
treynor2009(opt_j_w2009),treynor2010(opt_j_w2010),
treynor2011(opt_j_w2011),treynor2012(opt_j_w2012),
       treynor2013(opt_j_w2013),treynor2014(opt_j_w2014),
treynor2015(opt_j_w2015)]
tre_j2=np.mean(list_t_j2)
#Índice de Jensen
list_j_eq=[jensen2003(w),jensen2004(w),jensen2005(w),jensen2006(w),
jensen2007(w),jensen2008(w),jensen2009(w),jensen2010(w),jensen2011(w),
jensen2012(w),
       jensen2013(w),jensen2014(w),jensen2015(w)]
jen_eq=np.mean(list_j_eq)
list_j_s2=[jensen2003(opt_s_w2003),jensen2004(opt_s_w2004),
jensen2005(opt_s_w2005),jensen2006(opt_s_w2006),jensen2007(opt_s_w2007),
jensen2008(opt_s_w2008),jensen2009(opt_s_w2009),jensen2010(opt_s_w2010),
jensen2011(opt_s_w2011),jensen2012(opt_s_w2012),
       jensen2013(opt_s_w2013),jensen2014(opt_s_w2014),
jensen2015(opt_s_w2015)]
jen_s2=np.mean(list_j_s2)
list_j_t2=[jensen2003(opt_t_w2003),jensen2004(opt_t_w2004),
jensen2005(opt_t_w2005),jensen2006(opt_t_w2006),jensen2007(opt_t_w2007),

jensen2008(opt_t_w2008),jensen2009(opt_t_w2009),jensen2010(opt_t_w2010),
jensen2011(opt_t_w2011),jensen2012(opt_t_w2012),
       jensen2013(opt_t_w2013),jensen2014(opt_t_w2014),
jensen2015(opt_t_w2015)]
jen_t2=np.mean(list_j_t2)
list_j_j2=[jensen2003(opt_j_w2003),jensen2004(opt_j_w2004),
jensen2005(opt_j_w2005),jensen2006(opt_j_w2006),jensen2007(opt_j_w2007),
jensen2008(opt_j_w2008),jensen2009(opt_j_w2009),jensen2010(opt_j_w2010),
```

```
jensen2011(opt_j_w2011),jensen2012(opt_j_w2012),
       jensen2013(opt_j_w2013),jensen2014(opt_j_w2014),
jensen2015(opt_j_w2015)]
jen_j2=np.mean(list_j_j2)


#tabla resumen
resumen =   [['Cartera','Rentabilidad media %', 'Desv.Tip. media %',
'Mediana', '% Resultado +', 'Ratio de Sharpe', 'Indice de Treynor',
'Indice de Jensen'],
             ['Eurostoxx', round(media_cartera_eu, 4),
round(desv_cartera_eu, 4), round(mediana_eu, 4), round(pos_eu , 4),
 round(sha_eu , 4), round(tre_eu , 4), 0],
             ['Equilibrada', round(media_cartera_eq, 4),
round(desv_cartera_eq, 4), round(mediana_eq, 4), round(pos_eq , 4),
round(sha_eq , 4), round(tre_eq , 4), round(jen_eq , 4)],
             ['Sharpe', round(media_cartera_s2, 4),
round(desv_cartera_s2, 4), round(mediana_s2, 4), round(pos_s2 , 4),
round(sha_s2 , 4), round(tre_s2 , 4), round(jen_s2 , 4)],
             ['Treynor', round(media_cartera_t2, 4),
round(desv_cartera_t2, 4), round(mediana_t2, 4), round(pos_t2 , 4),
round(sha_t2 , 4), round(tre_t2 , 4), round(jen_t2 , 4)],
             ['Jensen', round(media_cartera_j2, 4),
round(desv_cartera_j2, 4), round(mediana_j2, 4), round(pos_j2 , 4),
round(sha_j2 , 4), round(tre_j2 , 4), round(jen_j2 , 4)]]
print(tabulate(resumen, headers='firstrow', tablefmt='simple'))


#histogramas
plt.hist(rendimiento_porcentual_cartera_eq, bins=20, label='Equilibrada',
color='mediumseagreen')
plt.hist(rendimiento_porcentual_cartera_eu, bins=20, label='Eurostoxx',
color='slategrey')
plt.xlabel('% Rendimiento')
plt.ylabel('Frecuencia')
plt.legend()
plt.grid(linestyle='--')
plt.savefig('figura2.png')
plt.show()

#histogramas
plt.hist(rendimiento_porcentual_cartera_s2, bins=20, label='Sharpe',
color='lightcoral')
plt.hist(rendimiento_porcentual_cartera_t2, bins=20, label='Treynor',
color='c')
plt.hist(rendimiento_porcentual_cartera_j2, bins=20, label='Jensen',
color='mediumorchid')
plt.xlabel('% Rendimiento')
plt.ylabel('Frecuencia')
plt.legend()
plt.grid(linestyle='--')
plt.savefig('figura4.png')
plt.show()

#evolución de las carteras
```

```
plt.plot(rendimiento_porcentual_cartera_eu, label='Eurostoxx',
color='slategrey')
plt.plot(rendimiento_porcentual_cartera_eq, label='Equilibrada',
color='mediumseagreen')
plt.plot(rendimiento_porcentual_cartera_s2, label='Sharpe',
color='lightcoral')
plt.plot(rendimiento_porcentual_cartera_t2, label='Treynor', color='c')
plt.plot(rendimiento_porcentual_cartera_j2, label='Jensen',
color='mediumorchid')
plt.xticks([])
plt.text(50, -60, u'03-07', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(285, -70, u'04-08', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(520, -60, u'05-09', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(745, -70, u'06-10', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(980, -60, u'07-11', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(1205, -70, u'08-12', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(1440, -60, u'09-13', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(1665, -70, u'10-14', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(1900, -60, u'11-15', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(2125, -70, u'12-16', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.text(2360, -60, u'13-17', fontsize = 10, horizontalalignment='center',
verticalalignment='center')
plt.xlabel('Tiempo')
plt.ylabel('% Rendimiento')
plt.legend(loc=2)
plt.grid(linestyle='--')
plt.savefig('figura8.png')
plt.show()
```