UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería del Software

Gestión de cartera inteligente mediante técnica Deep Learning

Smart portfolio management trough Deep Learning technique

Realizado
David Ruiz Rueda

Tutorizado por
Francisco López Valverde

Departamento
Lenguajes y Ciencias de la Computación

Málaga, junio de 2021

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADUADO EN INGENIERÍA DEL SOFTWARE

Gestión de cartera inteligente mediante técnica Deep Learning

Smart portfolio management trough Deep Learning technique

Realizado por
David Ruiz Rueda

Tutorizado por
Francisco López Valverde

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, JUNIO DE 2021

Fecha defensa: julio de 2021

Resumen

En este Trabajo Final de Grado se realiza una gestión de cartera inteligente mediante el uso de técnica Deep Learning, la Inteligencia Artificial trata de predecir los valores futuros de cada activo, otorgando un peso en la cartera a cada uno de estos activos que se modifica de manera dinámica con el paso del tiempo, maximizando de esta forma los beneficios obtenidos, así mismo se realiza una gestión de capital mediante control del riesgo a través de técnicas derivadas de Machine Learning para predecir volatilidades futuras y así poder realizar una gestión de capital eficiente, disminuyendo el riesgo y aumentando el beneficio de la cartera, también se muestran los resultados que respaldan el uso de estas técnicas en comparación a no usarlas, además se realiza un estudio previo del estado del arte el cual nos situa en el contexto de las más novedosas técnicas aplicadas a la gestión de cartera.

Palabras clave: Gestión de cartera, Aprendizaje Profundo, Gestión de capital

Abstract

In this Final Degree Project an intelligent portfolio management is carried out, through the use of Deep Learning technique, Artificial Intelligence tries to predict the future values of each asset, giving a weight in the portfolio to each of these assets that is modified from dynamically over time, thus maximizing the benefits obtained, likewise capital management is carried out through a risk control using techniques derived from Machine Learning to predict future volatilities and thus be able to carry out efficient capital management, reducing the risk and increasing the portfolio. The results that support the use of these techniques are also shown compared to not using it, in addition to carrying out a prior study of the state of the art which places us in the context of the most innovative techniques applied to management of wallet.

Key words: Deep Learning, Portfolio Management, Capital Management

Índice

1 Introducción

1.1. Contexto

La aplicación de la Inteligencia Aritifical (IA), está abriéndose camino con fuerza en distintas áreas, el mundo de las finanzas no es una excepción, ya hay diversos estudios para la predicción del comportamiento de los mercados basadas en técnicas de IA, estos estudios tradicionalmente han estado más enfocados a la estrategias de compra venta de valores, como podemos ver en los siguientes libros de J.M. Werming y Y.Hilpish respectivamente [1],[2], tratando el problema individualmente y no sobre un conjunto de activos, aunque ya podemos encontrar algunos estudios que hacen referencia a la gestión de carteras, en el siguiente artículo científico de H.Yuna, M.Lee, Y.S.Kang y J.Seok [3], podemos ver una muestra de ello. La gestión de carteras trata el problema de administrar el activo de una manera eficiente adaptándose a los cambios de mercado, no solo se trata de comprar o vender un producto, sino que dirige las operaciones sobre varios activos y además tiene en cuenta el riesgo subyacente de hacer estas inversiones logrando un equilibrio entre rentabilidad y riesgo asumido. Con este Trabajo Fin de Grado (TFG) se busca resolver este problema de una manera eficiente, mediante el uso de técnicas de Deep Learning capaces de adaptarse a los cambios, aprender de ellos, maximizar los beneficios, y además hacer una correcta gestión de los riesgos mediante técnicas de Machine Learning. Se enfocará a la gestión a productos de fondos indexados o por sus siglas en inglés ETF, estos productos replican un conjunto de valores y traen consigo una serie de ventajas como son diversificación de riesgo y menores costes de transacción [3].

1.2. Motivación

Estamos viviendo una nueva revolución y estamos en los inicios de ella, la IA ha llegado para quedarse y el no aplicarla pronto supondrá una clara desventaja, la aplicación de IA puede

llevarnos a un nuevo nivel de eficiencia en el siempre complicado mundo de las finanzas, “ This is the most exciting time to adopt a disruptive technology that will transform how everyone invest for generations”, cita el profesor M.L de Prado en uno de sus muchos libros de Machine Learning (ML) aplicadas a las finanzas [4].

Cada día se avanza más en la técnica en diferentes campos, en el mundo de las finanzas en el pasado no era complicado que un inversor aficionado pudiera sacar rentabilidad de los mercados. Pero cada día más estos últimos se están volviendo más eficientes, por lo que no está dejando cabida a la inversión por entretenimiento. Y ya hay autores que ven con claridad cuál es el futuro próximo sino el presente de las inversiones, “ One day in the near future, ML will dominate finance, science will curtail guessing, and investing will not mean gambling ”, nuevamente cabe la pena mencionar a M.L de Prado [4].

1.3. Objetivos

Se fija el objetivo del desarrollo de un aplicación en lenguaje Python capaz de gestionar carteras de manera inteligente, mediante técnicas Deep Learning (DL) y gestión del riesgo mediante técnicas de ML a través de lenguaje R embebido en la aplicación Python. Los objetivos generales y funcionalidades a alto nivel son los que se detallan a continuación.

1.3.1. Maximización de beneficios mediante técnicas de Deep Learning

Dada una lista de fondos indexados, la IA gestionará la cartera maximizando los beneficios, mediante el uso de técnicas de DL.

1.3.2. Gestión de capitales mediante control del riesgo a través de predicción de volatilidades

La aplicación dispondrá de códido R embebido en el lenguaje Python para analizar la volatilidad mediante técnicas derivadas de ML y así hacer una gestión de capitales conforme al riesgo predecido.

1.3.3. Análisis de resultados

Dispondrá a su vez de un sistema que arroje los resultados obtenidos por la IA, para poder comprobar su eficacia, la visualización de estos datos podrá hacerse mediante la librería de Pyplot y/o mediante el uso de la herramienta Tableau.

1.4. Estructura del documento

En el Capítulo 1 se realiza la introducción como hemos visto,donde se habla de la motivación, el contexto y los objetivos de este TFG. Posterioremente se trata el Estado del arte en el Capítulo 2, haciendo un repaso cronológico desde los orígenes hasta la más avanzadas técnicas empleadas hoy día en la gestión de carteras, para pasar a repasar la metodología de trabajo usada en el Capítulo 3 y las fases de la ingeniería del software aplicadas al proyecto en el Capítulo 4, 5 y 6 para el desarrollo del programa. En el Capítulo 7 se analizan los resultados obtenidos para comprobar la eficacia del algoritmo, por último se realizan una serie de conclusiones y se señalan las líneas futuras de investigación en el Capítulo 8. Se adjuntan en los apéndices los distintos manuales de intalación y usuario.

2

Estado del arte

2.1. Descripción del problema

Podemos resumir la gestión de carteras como la asignación de un conjunto de activos de número N, en el que pretendemos lograr el máximo beneficio minimizando los riesgos.

$$W_{\{w_1,\ldots,w_n\}} = argmax \left\{ \sum_k w_k E(r_k) - \lambda\sigma \left(\sum_k w_k E(r_k) \right) \right\} \tag{1}$$

En el artículo científico antes mencionado [3] , resumen el problema en esta Ecuación.1, w_k es el peso asignado al activo k-ésimo, r_k es el retorno del activo k-ésimo, σ es un método para evaluar el riesgo, λ es un peso para el parámetro riesgo y $E(r_k)$ es el retorno esperado y sujeto a $\sum_k w_k = 1$ y $0 \leq w_k \leq 1$.

2.2. Literatura previa

Si retornamos a los primeros estudios matemáticos referentes a la gestión de cartera debemos mencionar el que ha sido y sigue siendo un referente, H.M. Markowitz. En estos estudios [5],[6]. H.M. Markowitz utiliza la varianza como método del cálculo del riesgo y la media de retorno esperado de cada activo para realizar un balance riesgo/beneficio, lo que hoy día nos parece intuitivo de haberlo oído en innumerables ocasiones,y que se ha convertido en una frase hecha “a mayor beneficio mayor riesgo”, tiene sus orígenes en este estudio. En el nos muestra como diversificar las inversiones, es decir realizar inversiones en un conjunto de activos no altamente correlacionados y no en un activo o en un conjunto altamente correlacionado, conseguimos disminuir el riesgo considerablemente, tratando de maximizar los beneficios. Podemos observar con mayor facilidad este problema en la siguiente imagen.

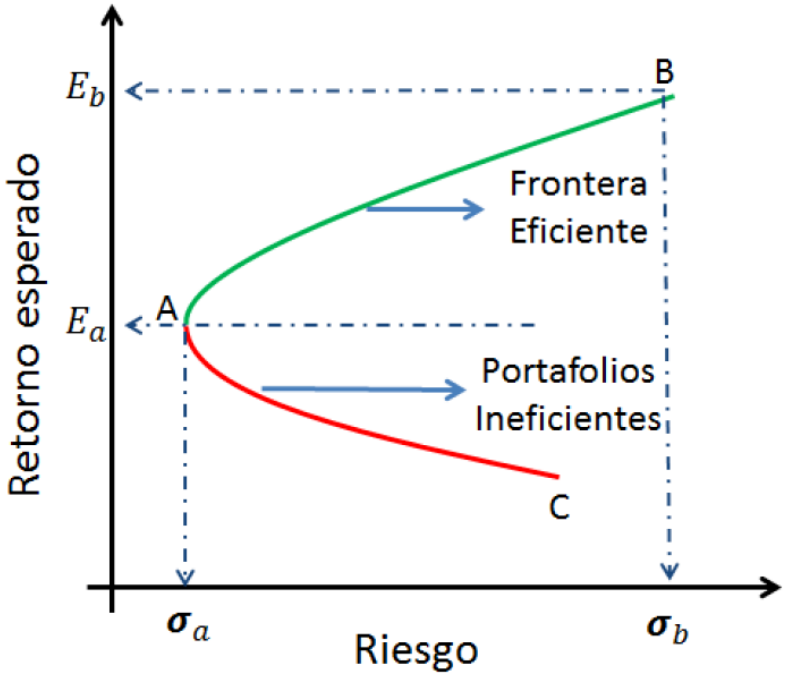


Figura 1: Frontera de la eficiencia. Fuente: [7]

En la Figura.1 podemos observar como al aumentar el retorno esperado E_b aumenta la desviación estandar σ_b para carteras eficientes, es decir aumenta el riesgo proporcialmente al beneficio, en carteras ineficientes no ocurre esto. En la Ecuación.2 extraida de los estudios vistos anteriormente [5],[6], se presenta la que es la clave para el cálculo del riesgo, como puede observarse una mayor correlación entre los activos implica un mayor riesgo.

$$\sigma_p^2 = \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq i} w_i w_j \sigma_i \sigma_j \rho_{ij} \tag{2}$$

Donde σ es la desviación estandar, w_i es el peso del activo i, y ρ_{ij} es el coeficiente de correlación entre dos activos de la cartera.

Posteriores estudios han trabajado sobre esta línea, añadiéndole diversas mejoras, así mismo podemos observar trabajos como el de C.B Kalayci, O.Ertenlice, M.A. Akbay [8], que han sintetizado esta serie de avances sobre el trabajo previo de Markowitz de la optimización de carteras, mediante la varianza media o por sus siglas en ingles (MVPO), realizando en él una clasificación de técnicas para resolver el problema de gestión de carteras.

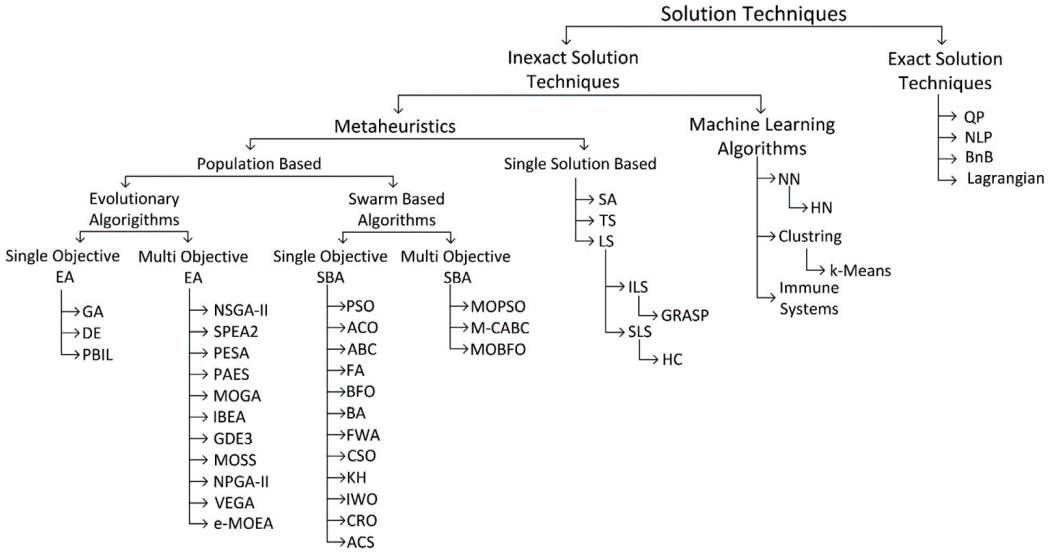


Figura 2: Clasificación de distintas aproximaciones para resolver el problema de la gestión de cartera. Fuente [8]

En la Figura.2 se observan la multitud de aproximaciones para dar solución al problema de MVPO a través de distintas técnicas.

2.3. Avances relacionados con la gestión de cartera

Como hemos visto anteriormente hay multitud de estudios relacionados con la optimización de gestión de carteras, pero hay que reseñar algunos de ellos en concreto dado el ámbito de este TFG.

2.3.1. Reducción de dimensiones del conjunto de datos

Cabe mencionar la reducción de dimensiones en el preprocesado de datos para la optimización de carteras. La reducción de estos datos son herramientas útiles en el preprocesado de análisis de datos cuantitavos, con el objetivo de preservar las características principales, se demuestran unos resultados prometedores del análisis de componente principales (PCA), en el trabajo de H.A. Tayali y S.Tolun [9], para la reducción del ruido en el preprocesado de datos en el estudio de R.Wang, F.Nie, R.Hong, X.Chang, X.Yang y W.Yu [10] y multitud de estudios en otras áreas, como podemos observar en el trabajo de F.Nie, D.Xu, X.Li y S.Xiang [11], donde realizan el estudio de reducción de dimensiones a través de etiquetas virtuales de regresión

y en el de los autores antes mencionados S.Tolun, H.A. Tayoli en este nuevo estudio [12], de minería de datos para problemas de ubicación, que dan peso a la utilidad del PCA.

Cuando aplicamos una PCA a un conjunto de datos, lo que conseguimos es una reducción de las dimensiones de este conjunto sin perder datos representativos, es decir eliminamos datos redundantes, hay que señalar que no se eliminan columnas enteras sino que se realiza una nueva configuración con los datos más interesantes, estos datos serán los que tienen mayor desviación, se utiliza para este fin el método de la covarianza o el método de las correlaciones, en el caso que nos concierne, el de la covarianza es el más aceptado. Como resultado final se obtiene un conjunto de datos reducidos pero que mantiene las características del conjunto original manteniendo la máxima varianza de este último, uno de los beneficios principales además de la disminución del tamaño es la de evitar el temido overfitting.

¿ Pero como saber a cuántas componentes reducir el conjunto de datos original ? , en los artículos científicos de R.Gauojoux y C.Seoihe [13] y en el trabajo de C.D.Sigg y J.M.Buhmann [14], podemos observar que el 99 % la varianza fue convenientemente simplificada a dos componentes principales. Esto satisface la aproximación usual de no seguir extrayendo características cuando se alcanza un 90 % de la varianza original extraida del libro de E.Alpaydin [15].

2.3.2. Agrupamiento del conjunto de datos

Otra de las mejoras realizadas sobre los estudios previos de Markowitz [5],[6], es el agrupamiento de datos. En el artículo científico de C. lorio, G. Frassio, A. D´Ambrosio y R.Siciliano [16], y en el también artículo científico de D. León, A.Aragón, J.Sandoval, G.Hernández, A.Arévalo y J.Niño [17], donde tratan el problema del agrupamiento de manera directa y como objeto principal de estos estudios, ambos muestran unos resultados prometedores, además se logran volatilidades menores en comparación con el modelo propuesto clásico MVPO como se extrae del estudio antes mencionado [17].

Así mismo no es difícil entender el concepto intuitivamente, si realizamos una agrupación de los activos por la correlación que tienen entre ellos, agrupando los que tienen una alta correlación, será más fácil poder controlar el riesgo dada la medida vista en la Ecuación.2. Ya que la desviación estandar es directamente proporcional al índice de correlación entre los activos. Así mismo es bastante intuitivo entender porqué elementos altamente correlacionados generan un mayor riesgo. Supongamos una cartera altamente correlacionada entre sus productos, esto sig-

nifica que si tenemos pérdidas en alguno de los productos es muy probable que los restantes productos también tengan pérdidas, por lo que el riesgo es alto al no haber una diversicación efectiva, aunque tengamos un número N alto de estos productos.Supongamos ahora que estos productos no están correlacionados, el hecho de que un producto o activo sufra pérdidas no supondrá que el resto deba seguir el mismo camino. Por lo tanto con la agrupación de datos según su correlación podremos observar el número de grupos diferenciados que se crean y si este número no es lo suficientemente alto, tendrémos que replantear los activos incluidos en la cartera por otros en lo que exista una diversificación real.

No es el único beneficio que trae la agrupación de datos, además de la observancia de los distintos grupos que se forman, el agrupamiento de datos permite suavizar los datos atípicos. Supongamos ahora que dentro de un grupo de estos valores altamente correlacionados uno de ellos sufre lo que se denomina un dato atípico "outliers", esto podría suponer un desequilibrio o reacción excesiva en una estrategia de inversión, ya que estos datos afectarían más de lo que cabría esperar al resultado, pero al estar agrupado en un conjunto se produce un efecto de suavizado, los restantes activos harán converger el resultado del grupo a la media, evitando así giros bruscos o inesperados.

2.4. Machine learning en las finanzas

ML es un campo de la Inteligencia Artificial (IA), donde se trata de que los algoritmos aprendan, se entiende este aprendizaje como la capacidad de mejorar el rendimiento con la experiecia, como se extrae del artículo de la Wikipedia [18] y donde podemos encontrar más información al respecto. Este aprendizaje tiene una amplia gama de aplicaciones en el campo de las finanzas como son cálculo del riesgo, créditos, calcular solvencia de empresas etc.

Las aproximaciones a ML se pueden dividir fundamentalmente en cuatro tipos de algoritmos, aprendizaje supervisado, aprendizaje no supervisado, aprendizaje mixto y aprendizaje por refuerzo, como puede verse en el artículo antes citado [18] y en el que además se realizan subclasificaciones de estos tipos principales.

2.4.1. Machine Learning aplicado a las finanzas

Podemos ver en el excelente trabajo de M.L de Prado, *Advances in Financial Machine Learning* [4], donde habla de la aplicación de ML a las finanzas, sus peculiaridades y que los distingue del ML estandar, también se hace eco de una serie de errores muy comunes en este campo y da solución a estos.

El tipo de dato y su estructura sobre el que podemos trabajar en el área financiera es una de las características de ML aplicado a las finanzas, que podemos extraer del libro antes citado [4], en el se explica de manera breve y sencilla estos tipos de datos y la estructura típica de los mismos.

M.L de Prado nos recalca en su libro [4], que uno de los problemas más comunes, que hacen que los algoritmos no funcionen en la práctica en el área de las finanzas, es el overfitting, el sobreentrenamiento hace que los algoritmos se adapten muy bien a los datos proporcionados pero con una nueva entrada de datos fallarán estrepitosamente, nos explica que es una de las razones fundamentales de porque ML estándar no funciona en la práctica en el mundo de las finanzas.

2.4.2. Estudios recientes por áreas en el campo de las finanzas

Las técnicas ML se han aplicado a las finanzas tradicionalmente para distintos tipos de cometidos, como pueden ser:

Cálculo del riesgo : Como podemos ver en un reciente estudio de A. Fenerich et al. [19], en el que se hace recopilación de trabajos anteriores y se proponen mejoras en el preprocesado de datos, también se añade un índice de solvencia del cliente, en contraste con la puntuación crediticia que se usa tradicionalmente.

Concesión de créditos : Un ejemplo de esto lo tenemos en el reciente artículo de M.J.A Garzón, J. Arroyo, A. Caparrini y M.J.S Vargas [20], en el hablan de las reticencias de los bancos a usar ML, ya que a veces se percibe como una caja negra, pero ellos demuestran que se puede aplicar técnicas ML de manera transparente y precisa [20], además se introducen en el novedoso mercado del crédito P2P con técnicas bien conocidas como el modelo de regresión logistica entre otros.

Seguros de vida : En el artículo científicio de A.S Krah, Z.Nilolic y R.Korn [21], tenemos una muestra de ML aplicado al cálculo de los seguro de vida, en el introducen el método de mínimos cuadrados de Monte Carlo (LSCM), señalan es su estudio [21], que la idea principal de este método es realizar una selección inteligente de simulaciones para obtener una función dependiente del riesgo de pérdidas.

2.5. Deep learning en las finanzas

2.5.1. Deep Learning aplicado a las finanzas

El uso DL está en pleno auge, ya existen estudios de aplicación de estas técnicas a la gestión de carteras como los artículos de J.B. Heaton, N.G. Polson y J.H. Witte [22] y de T. Fischer y C. Krauss [23], en el primero usan la ténica de codificación automática, mientras en el segundo de estos estudios se aplican redes de memoria larga a corto plazo o por sus siglas en inglés (LSTM) a la gestión de carteras . " Los algoritmos de Deep Learning pueden explotar y detectar interacciones que son invisibles para las teorías exixstentes sobre finanzas ", como se recoge del artículo antes mencionado [22]. Los humanos no tenemos una capacidad innata para predecir el comportamiento de un valor, en este sentido las herramientas de DL pueden ser de utilidad. El uso de estas herramientas requiere cierta configuración inicial, son claves las funciones de activación, el número de capas así como adoptar una función de coste para la minimización de los errores.

En algunos de estos estudios se encuentran cuestiones muy interesantes, dada la amplia variedad de técnicas DL y ML, de entre las cuales escoger. Hay algunas estructuras que parecen adaptarse mejor que otras a las series temporales, es el caso de las redes neuronales recurrentes (RNN), en este artículo web de C.OLah [24], encontramos una introducción gentil a un tipo específico de estas redes y antes mencionadas LSTM. Podemos así realizar la siguiente afirmación, las RNN se han mostrado útiles para la predicción de series temporales en el área de las finanzas, que se ha extraido del estudio ya mencionado [23], en el artículo científico de Y.Ma, R.Han y W.Wang [25], demuestran que hay técnicas ML que parecen tener mejor rendimiento que estas técnicas DL.

Pero la predicción del comportamiento de valores no es un tema trivial, además del alto nivel de ruido y a la generalmente aceptada idea de eficiencia de los mercados, es complicado

superar al mercado. Aun así el mercado financiero está lleno de anomalías bien conocidas, como se derivan de los estudios de H.Jacobs [26] y de J. Green, J.R.M Hand y X.F.Zhang [27].

2.5.2. Estudios recientes por áreas en el campo de las finanzas

Cómo hemos visto ya anteriormente en la sección 2.4.2 , la IA se utiliza en el campo de las finanzas para diversos cometidos, tradicionalmente se ha venido usando más ML para estos cometidos ya que sus algoritmos tienen más transparencia que los de DL, dando la posibilidad de la intervención humana cosa que los algoritmos de DL al tratarse de cajas negras no permiten, por lo que hay reticencias a la hora de usar este tipo de algoritmos en estos cometidos, aun así podemos encontrar algunos ejemplos:

Cálculo del riesgo : Como podemos ver en el estudio de P.M. Addo, D. Guegan y B. Hassani [28], donde construyen clasificadores binarios basados en modelos de DL y de ML en datos reales para predecir la probabilidad de incumplimiento del préstamo, también se incide en la necesidad de la regulación de este tipo de algoritmos por parte de las autoridades, como conclusión final no ven un mejor rendimiento en los algoritmos de DL con respecto a los de ML.

Concesión de créditos : En el estudio de Van-Sang Ha y Ha-Nam Nguyen [29], en el ellos construyen un modelo de calificación crediticia, basado en el aprendizaje profundo y la selección de características para evaluar la puntuación de crédito del solicitante, a partir de estas características, uno de los objetivos del estudio es la selección adecuada de características, llegan a la conclusión de que los algoritmos de DL si son efectivos para la concesión de créditos y que pueden resultar mas eficientes computacionalmente y suponer un ahorro para los prestamistas.

2.6. Estudios recientes en la gestión de cartera

2.6.1. Estudios recientes aplicando Machine Learning

En lo referente a ML aplicado a la gestión de carteras podemos encontrarnos con estas recientes investigaciones de W.Chei, H.Zhang, M.K. Mehlawat y L. Jia [30], utilizan una novedo-

sa construcción de cartera realizando un modelo híbrido entre ML y los modelos de varianza-media, en una primera etapa predicen el valor de la acción mediante técnicas ML y en una segunda etapa realizan la selección de carteras, para ello usan la técnica extreme gradient boosting (XGBoost), con un algoritmo firefly mejorado para la optimización de parámetros de XGBoost, XGBoost está recibiendo mucha atención últimamente, ejemplo de esto es el artículo publicado por J.Nobre y R.F. Neves [31], donde demuestran que es posible conseguir grandes retornos mediante la técnica XGBoost usando esta como un clasificador binario, o la menos reciente pero no por ello menos importante de S.Dey, Y. Kumar, S.Saha y S. Basak [32], donde utilizan la técnica para clasificación de tendencias de mercado, también hay que mencionar que tiene un menor coste computacional que otras técnicas y ha mostrado buenos rendimientos como en los estudios anteriormente mencionados [31],[32], para un entendimiento más profundo de la técnica XGBoost podemos dirigirnos a este artículo web [33].

En otro reciente estudio, realizado en forma de artículo científico por los autores P.Jain y S.Jain [34], hacen una comparativa de la técnica ML, Hierarchical Risk Parity (HRP), introducida por M.L de Prado [35], con otros métodos tradicionales de asignación de carteras. En el HRP se aplica teoría de grafos y técnicas ML para construir una cartera diversificada. En el citado articulo [34], hacen además un estudio del rendimiento cuando se especifica erróneamente la covarianza, que recordemos es usada como indicador del riesgo. Como conclusión de este estudio se extrae que la técnica HRP, supera en la mayoría de universos a las técnicas tradicionales y en los peores casos no tiene un rendimiento considerado inferior. También se llega a la conclusión de que la técnica HRP se vuelve más fuerte cuanto más activos hay en cartera, y se deja esta línea de investigación pendiente ya que ellos realizaron el estudio con 10 activos. La ventaja principal de HRP estriba en el que no hay que realizar la inversión de la matriz de covarianza, como si es necesario en otros métodos, cuando las carteras son extensas los errores de estimación son mayores. El algoritmo se basa en tres etapas, en una primera etapa se realiza un agrupamiento de los activos por su correlación, en un segunda etapa se realiza la cuasi diagonalización de la matriz de covarianza, el objetivo es lograr una representación diagonal de la matriz de covarianza con altas correlaciones colocadas cerca una de la otra, por último los pesos se distribuyen utilizando la matriz de varianza inversa entre subgrupos, como puede verse en el citado anteriormente estudio [34].

Otro artículo científico que cabe mencionar es el realizado por los autores F.D.Paiva, R.T.N. Cardoso, G.P. Hanaoka y W.M. Duarte [36]. En este estudio se emplea un clasificador fusionado con las técnicas support Vector Machine (SVM) y varianza-media, el algoritmo fue diseñado para dividir los activos en dos grupos, un grupo que tenía potencial de conseguir un retorno mayor o igual al esperado y otro que no cumplía estas características. Aunque el método SVM presenta inconvenientes como carecer de una probabilidad de predicción de precios, estudios en el campo de las finanzas como el de Q.Wen, Z.Yang, Y.Song y P.Jia [37], sugieren su alto potencial para resolver este tipo de problemas. El método SVM es capaz de resolver problemas de estimación de regresiones, series temporales, densidad de estimación y reconocimiento de patrones, de acuerdo al estudio de V.N. Vapnik [38], al cual hay que mencionar como uno de los más prominentes científicos de la materia, como recoje el artículo científico antes mencionado [37].”La idea básica del método SVM es mapear vectores de entrada en un gran espacio por medio de un elemento de mapeo no lineal definido”,de acuerdo con Vapnik [38].En las conclusiones del estudio se deriva que la técnica SVM tiene unos resultados significativos aunque indican que para futuras líneas de investigación sería necesario ampliar el origen de los activos ya que todos pertenecían al índice Brasileño Ibovespa.

Por último hay que mencionar el estudio realizado por Y.Ma, R.Han y W.Wang y ya mencionado [25], en el se emplean técnicas ML como son Random Forest (RF), Support Vector Regression (SVR) y otras técnicas DL que no cabe mencionar en este apartado. Cabe destacar la técnica RF fusionada con la varianza-media, ya que es la que arroja mejores resultados. RF es un modelo no paramétrico y no lineal, que fue propuesto en 1995 por Ho, en la conferencia de análisis de documentos y reconocimiento [39]. Según Breiman [40], el modelo evita el overfitting dado que siempre converge. Los parámetros principales de RF son el número de arboles de decisión, la máxima profundidad del arbol, el mínimo numero de muestras requeridos para dividir un nodo interno, el número mínimo de muestras para llegar al nodo hoja y el número de características considereadas cuando miramos por la mejor división, como se deduce del estudio que estamos tratando [25]. En las conclusiones de este estudio se estrae que RF fue la que obtuvo mejor rendimiento por encima de las técnicas de DL incluidas y que conlleva ventajas como la antes mencionada de sortear el sobreentrenamiento. Cómo línea fu-

turas de investigación señalan ampliar estos estudios ya que utilizaron como datos de entrada simplemente los retornos históricos.

2.6.2. Estudios recientes aplicando Deep Learning

En el estudio realizado por Y.Ma, R.Han y W.Wang [25], además de las técnicas ML que ya vimos en el capítulo anterior, también basaron su investigación en técnicas DL, en concreto usaron técnicas LSTM, Deep Multilayer Perceptron (DMLP) y Convolutional Neural Network (CNN). DMLP es un tipo clásico de red neuronal artificial, la cual difiere del Multilayer Perceptron (MLP), principalmente porque contiene más capas escondidas , DMLP ofrece normalmente un rendimiento mejor que MLP como se extrae de los estudios de L.O. Orimoloye, M-C. Sung, T.Ma y J.E.v. johnson [41], donde hacen una comparativa de redes neuronales y de R.Singh y S.Srivastava [42], donde también realizan una comparativa. Aunque en todos estos estudios se muestran unos resultados significativos, no existe un consenso en cuanto al método óptimo, lo que también hace pensar que el overfitting puede estar involucrado en estos resultados y en la práctica no se obtengan tan buenos beneficios como los esperados. En el estudio que estamos tratando [25], la DMLP se compone de tres capas, una capa de entrada de datos, una escondida y otra capa de salida de datos. La DMLP fue entrenada con un gradiente descendente estocástico para evitar el problema del overfitting, hay que destacar que como función de activación se uso la función de unidad linear rectificada o por sus siglas (ReLU), ya que fue recomendada en el estudio ya mencionado [41]. En cuanto a la CNN empleada, que es un tipo de red neuronal novedoso introducida por Y.LeCun y Y.Bengio [43] . Las CNN se usan a menudo en Vision por Computador y en procesado de imágenes, pero en años más recientes algunos investigadores las han usado en predicción de precios en el mercado de valores, teniendo unos resultados promotedores como son los estudios de E.Hoseinzade y S.Haratizadeh [44] y de O.B. Sezer y A.M.Ozbayoglu [45]. La estructura típica de CNN consta de sucesivas capas convolucionales y de agrupación, seguida por capas totalmente conectadas, al estar tratando el tipo de dato de series temporales se utiliza una dimensión para la predición del precio, también usan con esta técnica el el gradiente descendente estocástico para el entrenamiento de la red y la función ReLU como función de activación, en las conclusiones obtenidas ya mencionadas en el apartado de ML, las técnicas ML fueron netamente superiores a las técnicas DL.

Otro estudio que cabe la pena mencionar es el realizado por F.Soleymani y E. Paquet [46], en este estudio introducen un marco de trabajo llamado DeepBreath, esta metodología combina un codificador automático apilado restringido (CAAR) y CNN.El CAAR esta inspirado por el libro de T.Wong y Z.Luo [47], y es empleado para la reducción de dimensiones y para la selección de características, asegurando que la información más importante queda retenida y no se pierde en este proceso, mientras la CNN es empleada para el aprendizaje y hacer cumplir la política de asignación de recursos en orden de maximizar los beneficios. El marco de trabajo consiste en estrategias fuera de línea y en línea, donde la primera se usa para el entrenamiento de la CNN y la segunda es donde se realiza la variación en la asignación de recursos conforme avanzan las series temporales.El número características del vector de entrada utilizadas en el estudio que estamos tratando [46], eran de 11, en las cuales se encontraban valores del precio como el de cierre, alto, bajo y indicadores financieros, el algoritmo extrae las características mas descorrelatadas mientras se está optimizando en el proceso de aprendizaje. Por su parte los codificadores automáticos son un tipo de red neuronal que reconstruye su salida en base a la entrada de datos, y que se adapta muy bien a las series temporales, formadas por dos partes llamadas codificador y decodificador. El codificador reduce las dimensiones de los datos de entrada, mientras el decodificador reconstruye los datos de entrada desde la capa escondida intermedia [47] , del estudio de Y.LeCun, Y.Bengio y G. Hinton [48], se extrae que los codificadores automáticos son capaces de aprender la reducción de dimensiones no linales, eliminando redundancias y correlaciones mientras extrae carácterisicas altamente informativas.En la Figura. 3 podemos ver una imagen de la estructura del codificador automático empleado en este estudio.

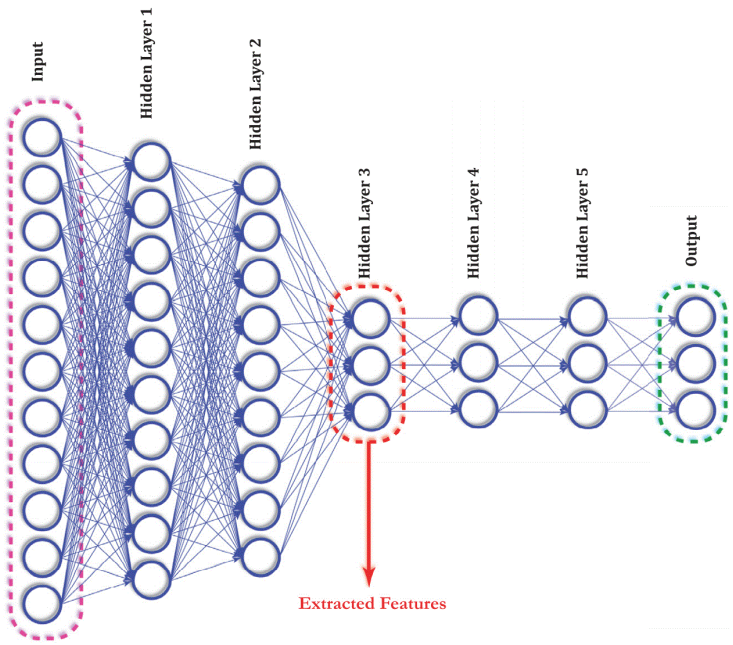


Figura 3: Estructura del codificador automático. Fuente [46]

Se llega a la conclusión en este estudio, que la complejidad computacional es lineal con el incremento del número de activos financieros, lo cual es una buena noticia y además quedo demostrada la eficiencia del sistema , dejan como futura línea de investigación introducir el factor exógeno de las redes sociales.

3

Metodología de trabajo

3.1. Metodología

Se usa una metodología ágil, en concreto SCRUM, pero modificada al desarrollo con una sola persona y enfocada a la ingeniería del software como se verá en el Capítulo 4, en cada sprint se ha ido añadiendo funcionalidades al programa , se ha optado por esta metodología para facilitar la revisión parcial del proyecto por parte del tutor, así como su adecuación al programa que presentará varias funcionalidades. Estas funcionalidades coincidirán con sprint (entregables) que el tutor irá validando y/o haciendo peticiones de mejoras, siguiendo cada uno de ellos las fases tradicionales de SCRUM. Podemos dividir la metodología empleada en cinco fases, que son las que encontramos a continuación.

3.1.1. Inicio

Se estudia y analiza el proyecto identificando los objetivos principales, dividiendo el proyecto en sprints, estos sprints concuerdan con los objetivos de este TFG, en cada uno de estos objetivos se han identificado una serie de Sprints, que podemos ver a continuación:

Objetivo 1 : Maximización de beneficios mediante técnicas de Deep Learning, se identificaron los siguientes sprints o historias de usuario:

- Recogida y almacenamiento de datos de mercado desde proveedor externo con código Python.
- Cálculo de indicadores de mercado

- Reducción de dimensiones del conjunto de datos
- Agrupación del conjunto de datos por correlaciones
- Preparación del vector de entrada de datos
- Creación del modelo de IA, entrenamiento, guardar y cargar el modelo

Objetivo 2 : Gestión del riesgo mediante control de volatilidades, en cuanto a este objetivo se identificaron los siguientes sprints:

- Inyección de código R en el código Python
- Paso de datos desde Python a código R
- Cálculo de volatilidad con R
- Paso de resultado de código R a Python
- Restricción del capital a invertir según volatilidad calculada

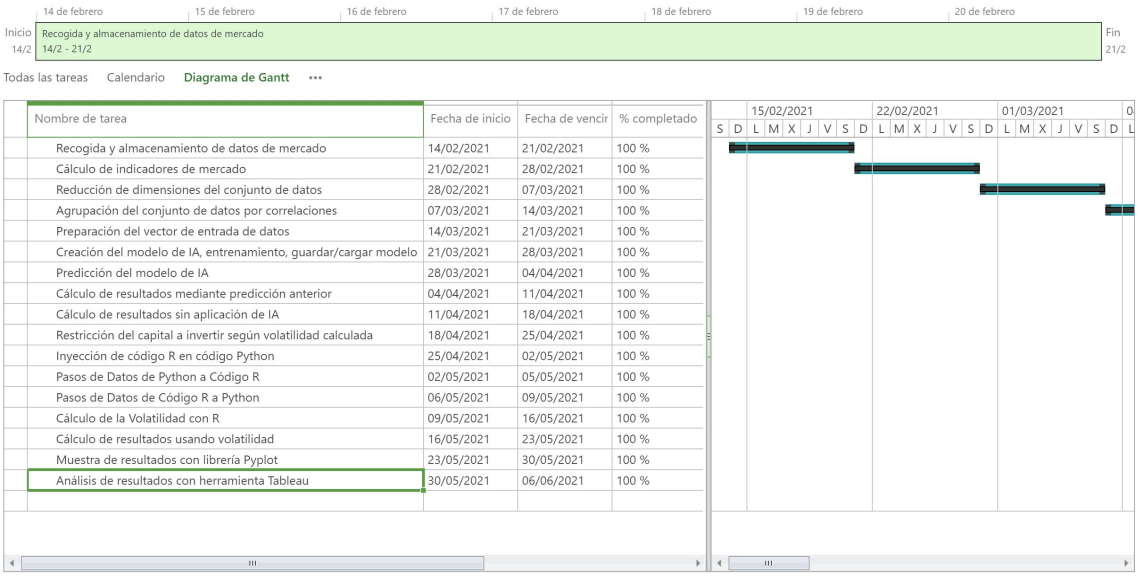
Objetivo 3 : Análisis de resultados, se extrayeron los siguientes sprints:

- Predicción del modelo de IA
- Cálculo de resultados mediante predicción anterior
- Cálculo de resultados sin aplicación de IA
- Muestra de resultados con librería Pyplot
- Análisis de resultados con herramienta Tableau

3.1.2. Planificación y estimación :

Se realiza un estudio de prioridades de los sprints, teniendo en cuenta su orden de importancia y el hecho de que hay sprints que dependen de otros. Se realiza la siguiente ordenación según criterios anteriores y se establece la estimación general de cada sprint en una semana, pudiendo variar esta aproximación en defecto y en exceso. Podemos ver en la Figura.4 un

imagen en detalle y en la Figura.5 una imagen general del diagrama de Gantt con los sprints desarrollados.



Nombre de tarea	Fecha de inicio	Fecha de vencir	% completado
Recogida y almacenamiento de datos de mercado	14/02/2021	21/02/2021	100 %
Cálculo de indicadores de mercado	21/02/2021	28/02/2021	100 %
Reducción de dimensiones del conjunto de datos	28/02/2021	07/03/2021	100 %
Agrupación del conjunto de datos por correlaciones	07/03/2021	14/03/2021	100 %
Preparación del vector de entrada de datos	14/03/2021	21/03/2021	100 %
Creación del modelo de IA, entrenamiento, guardar/cargar modelo	21/03/2021	28/03/2021	100 %
Predicción del modelo de IA	28/03/2021	04/04/2021	100 %
Cálculo de resultados mediante predicción anterior	04/04/2021	11/04/2021	100 %
Cálculo de resultados sin aplicación de IA	11/04/2021	18/04/2021	100 %
Restricción del capital a invertir según volatilidad calculada	18/04/2021	25/04/2021	100 %
Inyección de código R en código Python	25/04/2021	02/05/2021	100 %
Pasos de Datos de Python a Código R	02/05/2021	05/05/2021	100 %
Pasos de Datos de Código R a Python	06/05/2021	09/05/2021	100 %
Cálculo de la Volatilidad con R	09/05/2021	16/05/2021	100 %
Cálculo de resultados usando volatilidad	16/05/2021	23/05/2021	100 %
Muestra de resultados con librería Pyplot	23/05/2021	30/05/2021	100 %
Análisis de resultados con herramienta Tableau	30/05/2021	06/06/2021	100 %

Figura 4: Diagrama de Gantt en detalle, realizado con Microsoft Project

Figura 5: Diagrama de Gantt plano general, realizado con Microsoft Project

3.1.3. Implementación :

Una vez realizadas las fases anteriores se procede a implementar cada sprint, los sprints se dividieron en funcionalidades básicas, y a su vez fueron ordenados convenientemente por prioridades como hemos visto anteriomente, según su prioridad se realiza la implementación de cada uno de ellos, en el Capitulo 6, se profundiza más sobre este apartado, en esta fase juega un papel fundamental el diseño que se realizó en la fase de planificación y estimación, cada funcionalidad básica o sprint cuenta con unos requisitos y casos de uso asociados a estos requisitos.

3.1.4. Revisión y Lanzamiento :

Se revisa y valida el sprint, sino cumple con las exigencias se vuelve al punto 3 con las modificaciones pertinentes de la entrevista con el tutor, implementando así un bucle que se ejecuta de manera iterativa hasta el éxito de la ejecución de cada historia de usuario, momento en el cual se continua con el siguiente sprint. En la revisión se comprueba y validan los requisitos que el sprint implementa, también se incluye en esta fase, la etapa de pruebas para la detección de fallos y que es llevada a cabo antes de la revisión con el tutor. Una vez se finalizaron todas las historias de usuario y obtuvieron el visto bueno, se procede al envío de los entregables, la cual es la denominada fase de lanzamiento.

4

Análisis de Requisitos

4.1. Análisis de Requisitos

Se realizó un estudio de las funcionalidades que debía recoger la aplicación, se analizaron y se priorizaron, se estudió la posible incongruencia entre requisitos. Este paso coincide con la iniciación en la metodología de trabajo. Ya vimos en el capítulo de metodología de trabajo la extracción de sprints en base a objetivos, de esos sprints podemos sintetizar una lista de requisitos funcionales (RF) y no funcionales (RNF). Hay que hacer notar que al tratarse de código Python puro, ya que el TFG está centrado en la ciencia de datos y no en la interfaces gráficas de usuario, la lista de requisitos está adaptada a las exigencias de este TFG.

4.2. Listado de requisitos

4.2.1. Lista de requisitos funcionales

1. Recogida y almacenamiento de datos de mercado, que se obtendrán de un proveedor externo y que posteriormente se almacenarán en un fichero local

2. Extracción de los indicadores de mercado, de los datos obtenidos en el apartado anterior

3. Preprocesamiento de datos adecuado que constará de los siguientes puntos:

    a) Reducción de dimensiones del conjunto de datos

    b) Agrupación del conjunto de datos por correlaciones

    c) Preparación de un vector de entrada de datos

4. Disposición de un modelo de IA

    a) Se guardará este modelo cuando se cree una instancia del mismo
    b) Se podrá cargar este modelo cuando se haya guardado previamente
5. Cálculo dinámico de la volatilidad
6. Restricción del capital invertido según volatilidad calculada
    a) Cantidad a invertir inversamente proporcial a la votalidad
7. Predicción de los precios de los valores mediante el modelo de IA
    a) Se realizará una predicción para los valores agrupados según su sector
    b) Se realizará una predicción para los valores individualmente
8. Asignación de pesos a cada activo
    a) Cálculo del peso que se debe asignar a cada grupo por sectores
    b) Cálculo del peso de cada activo dentro de cada grupo
    c) Normalizado de todos los cálculos anteriores
9. Cálculo de resultados con los pesos anteriomente asignados
10. Cálculo de resultados sin aplicación de IA
    a) Asignación de pesos ecualizada a todos los activos
    b) Cálculo de resultados con esta asignación de pesos
11. Muestra de los resultados mediante gráficas
    a) Muestra de resultados aplicando el modelo de IA
    b) Muestra de resultados sin aplicación del modelo de IA

4.2.2. Lista de requisitos no funcionales

1. Construcción principalmente en lenguaje Python
2. Se dispondrá de código R inyectado en el código Python

3. Realización de intercambios de datos mediante:

    a) Código R y lenguaje Python

    b) Lenguaje Python y código R

4. Cálculo de la volatilidad mediante código R

5. Uso de la herramienta Tableau para el análisis de resultados

6. El sistema será eficiente, se podrá realizar lo siguiente:

    a) Salvar los datos obtenidos de mercado

    b) Salvar los modelos de IA elaborados

    c) Guardar en csv los resultados obtenidos

7. Modelos de IA construidos con la librería TensorFlow

8. Modelos ML construidos con la librería SciKit-Learn

4.3. Priorización de la lista de requisitos funcionales

Se realiza un análisis de priorización de RF, para realizar este análisis se tienen en cuenta los objetivos de este TFG, se priorizan estos objetivos y se extraen tres niveles, **prioridad máxima** (Nivel 1), **prioridad media** (Nivel 2) y **prioridad baja** (Nivel 3). Dentro de estos tres niveles se realiza una segunda ordenación por prioridad y por dependencia, es decir hay requisitos que para poder ejecutarlos necesitamos ejecutar antes otra serie de estos.

Nivel 1: Comprenden la siguiente lista de requisitos arriba mencionados.

- 7.a, para cumplir con este requisito es necesario realizar antes los siguientes RF:
    - 1
    - 2
    - 3.a
    - 3.b

        - 3.c
        - 4.a
        - 4.b
    - 7.b, para cumplir con este requisito hay que realizar antes los siguientes RF:
        - 1
        - 2
        - 3.a
        - 3.c

Nivel 2: Comprenden la siguiente lista de RF, deben haberse realizado los RF de Nivel 1 con anterioridad.

- 5
- 6.a

Nivel 3: Comprenden la siguiente lista de RF, deben haberse relizado con anterioridad los RF de Nivel 1 y Nivel 2.

- 11.a, para cumplir con este RF es necesario realizar antes los siguientes RF:
    - 8a
    - 8b
    - 8c
    - 9
- 11.b, para cumplir con este RF hay que realizar anteriormente estos RF:
    - 10a
    - 10b

5
Diseño

5.1. Integración de las distintas tecnologías utilizadas

5.1.1. Vista general

Se hace una primera aproximación de los distintos componentes de la aplicación, que podemos visualizar en la Figura.6, una breve explicación de cada uno de ellos, sus interacciones y sus correspondencias con los componentes en detalle.

Gestión de Carteras : Componente **desarrollado para este TFG**, corresponde con el componente **TradingSystem** interacciona con el resto de los componentes accediendo a sus interfaces para poder hacer uso de distintos servicios que iremos viendo en los distintos componentes, este componente puede dividirse en otros dos (en una visión general del sistema), el componente **Nucleo** y el componente **Archivos Locales**, el primero de ellos se encarga de dar la funcionalidad al sistema, el segundo de ellos se encarga del almacenamiento local de ficheros en distintos formatos, en este último se almaceran de manera local los siguientes archivos:

1. Los modelos generados de IA, estén entrenados o no.

2. Los archivos con la información financiera de cada ETF, en formato de Data Frames.

3. Los resultados generados, en formato csv.

Inteligencia Artificial : Componente **externo** para el desarrollo del modelo de la IA,corresponde con el componente **TensorFlow**, se usan sus interfaces para la creación del moddelo de IA, guardado de este, entrenamiento, predicción etc.

Ciencia de Datos: Componente **externo**, se corresponde con el componente **sci-kit**, para uso de funcionalidades de ciencias de datos, tales como análisis de componentes principales, normalizado de datos etc.

Código R : Componente **externo**, se corresponde con el componente **rpy2**, para inyección de código R en el código Python.

Indicadores de Mercado : Componente **externo**, se corresponde con componente **ta**, para la extracción de indicadores de mercado de los datos de mercado.

Lector Base de Datos: Componente **externo**, se corresponde con componente **pandas**, para lectura de datos de mercado de la BBDD de *Yahoo Finance*

BBDD: Componente **externo**, se corresponde con el componente **YahooFinance**, para el acceso a datos de mercado.

Veamos una vista genérica de la interacción de los componentes de la aplicación en la Figura. 6

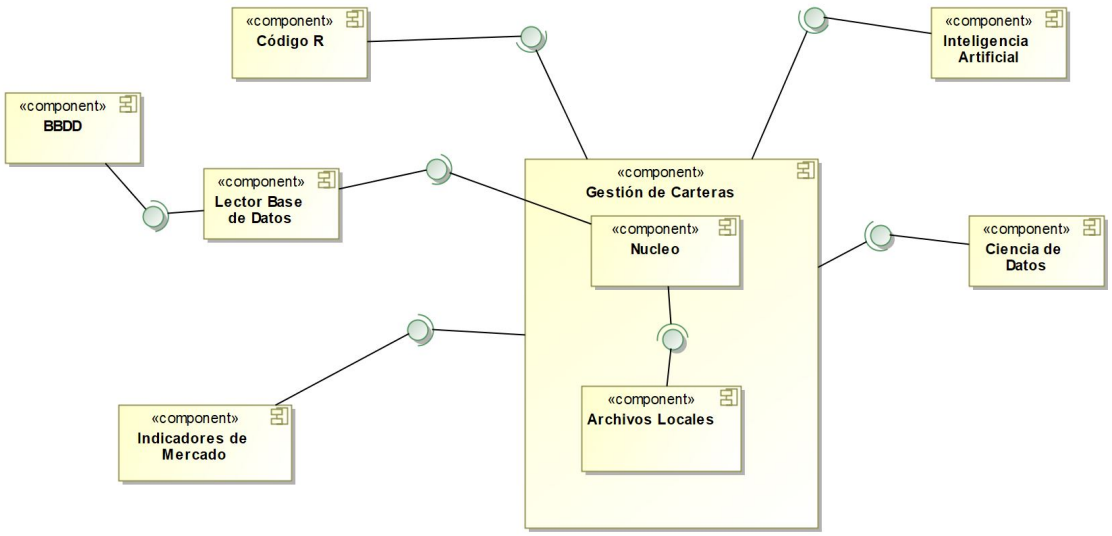


Figura 6: Diagrama de componentes, vista general

5.1.2. Vista en detalle

Para facilitar la comprensión de la integración de las distintas tecnologías en la aplicación, se describe la interacción de las distintas librerías externas con la aplicación de manera detallada.

Componente pandas : Para el acceso a los datos de mercado se usa en concreto el método *DataReader()* de **pandas_datareader**, que acceder a la BBDD de yahoo finance. Para el guardado local y posterior carga de datos se usan las funciones *pickle* de la clase **pandas**, estas interfaces son usadas por la clase **ETF** para el acceso de datos de mercado.

Componente ta : La clase **Momentun** es usada para los indicadores de momento, en concreto se usan los métodos *ROCIndicator(),RSIIndicator() y KAMAIndicator()* y la clase **Volatility** que usa los métodos *AverageTrueRange() y UlcerIndex()* como indicadores de volatilidad. La Clase **Cluster** usa estos métodos para extraer estos indicadores de las series temporales.

Componente sci-kit : La clase **PCA** es usada para la reducción de dimensiones de datos, en concreto se usan los métodos *fit() y transform()* son usados y la clase **MinMaxScaler** que usa los métodos *fit() y transform()* para la normalización de datos entre 0 y 1. La clase **InputDataInGroup** y **InputDataTotalGroup** son las que usan estos métodos.

Componente rpy2 : La clase **robjects** es usada y la clase **r** que usa el método *assign()* para poder inyectar código R en lenguaje Python. Para ello basta con introducir el código R dentro de la clase *r* para ello se usa como argumento el código R embebido entre comillas. La clase **Portfolio** usa este componente para la inyección de código R, en distintos métodos de esta clase.

Componente TensorFlow : La clase **Layers** es usada para las capas del modelo, utiliza los métodos *Dense() y LSTM()*, el primero de ellos para configurar la capa última y la clase **Sequencial** que usa el método *add(), compile(), summary(), fit(), save(), load_model() y predict()* para implementar y ejecutar los métodos en una red neuronal recurrente. La clase **ModelAI** es la que utiliza este componente .

Veamos a continuación el **diagrama de componentes en detalle** en la Figura.7, donde se observa la interacción entre componentes interna, es decir los componentes creados específicamente para esta aplicación.

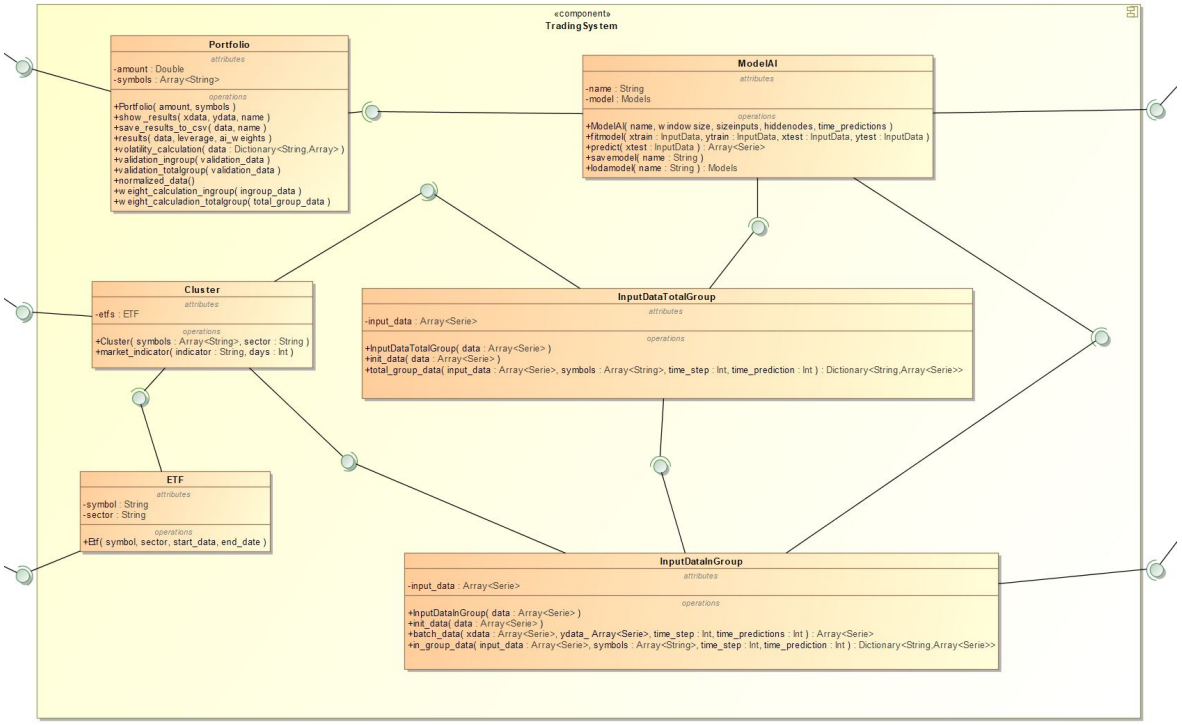


Figura 7: Diagrama de componentes en detalle,interacción interna

En el siguiente diagrama se muestra **diagrama de componentes** en la Figura.8, donde se observa la interacción entre componentes interna y la interacción con otros componentes externos.

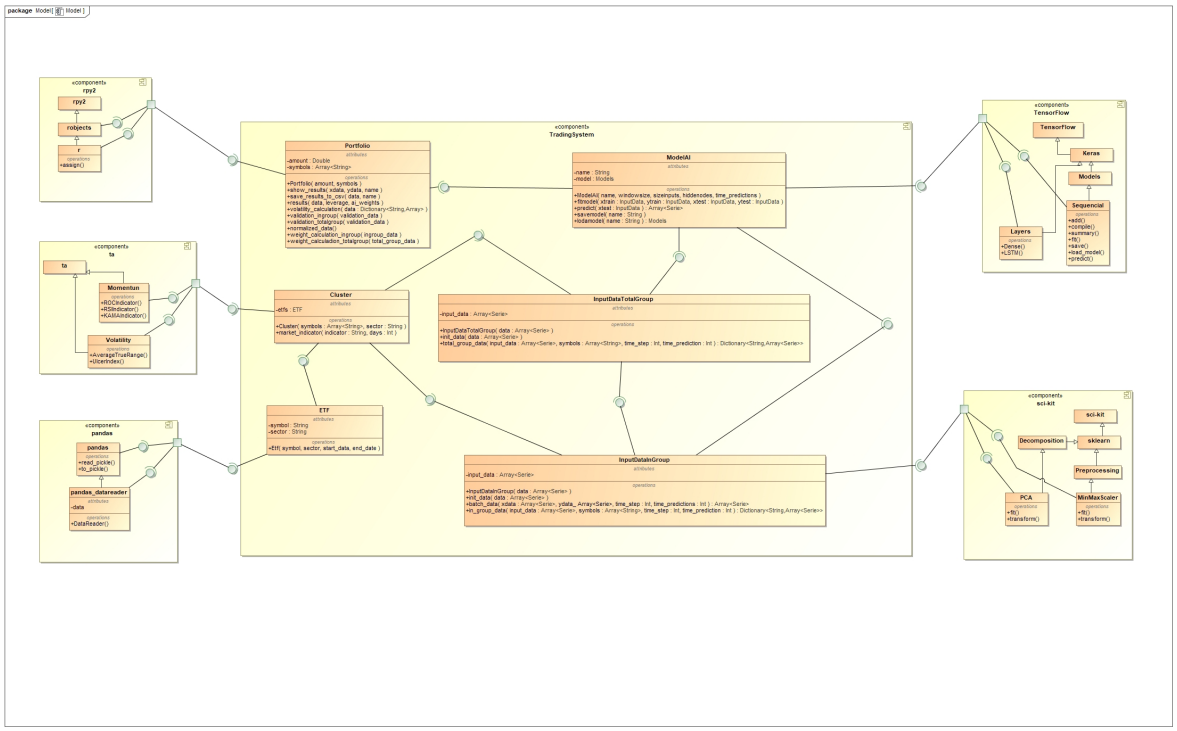


Figura 8: Diagrama de componentes, interacción externa

5.2. Tecnologías usadas

Se realiza una enumeración y breve explicación de las técnologías que se usaron y con que objetivo en este proyecto :

1.Lenguaje Python : Lenguaje multipropósito que cada día gana más adeptos, es muy usado para ciencias de datos por lo que se adaptaba a la perfección a los requerimientos del proyecto, además la facilidad de uso y la gran cantidad de información en internet facilitó el desarrollo del proyecto. Se usa este lenguaje como el principal para la construcción de la aplicación, en concreto para la consecución de los objetivos que podemos ver en la sección 1.3.1 y sección 1.3.3

2.Lenguaje R: Otro lenguaje muy usado para ciencias de datos, que cuenta con infinidad de librerías y que es de código abierto, además tanto Python como R permiten la integración de uno con el otro. Este lenguaje es usado para la consecución del objetivo definido en la sección 1.3.2, de este proyecto.

3.Librería rpy2: Librearía de inyección de código R en lenguaje Python. Se usó para la inyección de código R en lenguaje Python y para el paso de información desde Python a R y viceversa.

4.Librería pyplot: Librearía para la visualización de datos con Python. Usados en el sistema para mostrar los resultados obtenidos.

4.Librería pandas: Librería para guardado local y carga de datos. Se usa en la aplicación para acceder a la BBDD de Yahoo Finance,para guardar y cargar datos de mercado, de esta manera se evita tener que volver a descarlo de la BBDD.

5.Librería sci-kit(sklearn): Librería de Machine Learning, cómoda de usar. Esta librería se implementa para el preprocesamiento de datos de esta aplicación, en concreto para la reducción de las dimensiones del conjunto de datos mediante la ejecución de la PCA y de un normalizado de datos previo.

6.Librería ta: Librería para indicadores de mercado. Esta librería se implementa para el preprocesamiento de datos de esta aplicación, en concreto para la extracción de indicadores de mercado del conjunto de datos inicial.

7.Librería TensorFlow y Keras : Librerías de Deep Learning, esta tiene una curva de aprendizaje más inclinada pero sin duda los resultados son de gran calidad. Esta librería es usada para la construcción del modelo de IA en este proyecto.

8.Yahoo Finance (proveedor datos externo): BBDD para la recogida de información sobre los activos financieros ETF´s. Usada en la aplicación para la obtención de los datos de mercado de cada ETF mediante la librería pandas de Python.

9.Overleaf(Latex) : Editor de texto online de latex, permite un desarrollo cómodo así como profesional en la elaboración de PDF´s. Es utilizada para la elaboración de la memoria del TFG.

10.Pycharm: Entorno de desarrollo para el lenguaje Python, permite un modo ciencia de datos para visualizar de una manera más cómoda los datos, al estilo de Rstudio el IDE para

el lenguaje R. Es usada en este TFG para la elaboración de los modulos, clases, funciones y scripts de la aplicación.

11.Jupiter Notebooks: Permite la ejecución online o versión escritorio de código Python, así como muchos otros, facilita el uso de diferentes librerías como puede ser Tensor Flow. Se usa en este proyecto para la prueba y entrenamiento de modelos de IA.

12.Tableau: Programa para visualización y análisis de datos. Usada en la memoria del TFG para mostrar y analizar resultados obtenidos.

13.Microsoft Project: Empleado para la planificación de tareas. Se usa en el proyecto para la planificación en la ejecución de tareas, llevar un control de la ejecución en tiempo de cada tarea.

14.MagicDraw: Entorno de desarrollo para la elaboración de diagramas de distintos tipos. Empleado para la realización del diagrama de clases de este proyecto y para el diagrama de componentes

15.Sketch: Herramienta de diseño gráfico. Fue usada para la elaboración de gráficos para la memoria.

16.RStudio: IDE para desarrollo de código R. Fue usado para pruebas de código en R, en el cálculo de la volatilidad.

5.3. Diseño y Arquitectura

En el siguiente paso se define el diseño del modelo y la arquitectura, al no ser objeto de este TFG el uso de interfaz gráfica, se optó por una arquitectura modular. El diseño del modelo se centró en estos módulos que crean diferentes funcionalidades al conjunto del programa, que están pensados para poder ser reutilizados y para evitar la repetición de código. A continuación hacemos un breve repaso a la funcionalidad de cada clase diseñada.

Trading System : Este modulo integra todas las clases diseñadas para esta aplicación, podríamos decir que es el equivalente al main, aunque se modificó este nombre para darle más sentido dada la funcionalidad del programa, las distintas clases interaccionan con los diferentes servicios externos, en este caso librerías.

Portfolio : Esta clase tiene como finalidad la gestión de la cartera, en ella se refleja el activo metálico disponible, los símbolos de los activos con los cuales contamos en cartera, muestra y guarda los resultados obtenidos por el sistema. Hace uso principalmente de la clase ModelAI, de la cual obtiene las predicciones sobre los distintos activos.

ModelAI : En esta clase se encuentra la implementación del modelo de IA, se pueden crear distintos modelos que se guardarán una vez se hayan entrenado, ya que el entrenamiento es muy costoso computacionalmente, si se desea entrenar de nuevo un mismo modelo es posible, para poder almacenar distintos modelos esta clase dispone de un atributo "name". Esta clase interacciona además de con la clase anteriormente vista, con la clase **InputDataInGroup** y **InputDataTotalGroup**, que les proporcionan las entradas de datos por argumento al modelo de IA, en la primera y segunda etapa respectivamente.

InputDataInGroup : Esta clase es la que lleva la carga de parte del preprocesamiento de datos, es clave para el éxito de la aplicación, ya que se encarga de realizar el agrupamiento de datos para el paso como argumento al modelo de IA para la primera etapa, es decir la etapa en la cual los datos de los ETFs están agrupados por sectores, además de con la clase anteriormente vista interacciona con la clase **Cluster**, ya que ella es la que le proporciona los datos para su posterior preprocesado.

InputDataTotalGroup : Esta clase junto con la anterior se encargan del preprocesamiento de datos, en ella se agrupan los datos, para dar la entrada a la segunda etapa del modelo de IA, veremos con más detalle los agrupamientos de esta clase y la anterior en el capítulo. 6 (Implementacion), además de con la clase anteriormente vista **ModelAI** interacciona con la clase **Cluster**, ya que ella es la que le proporciona los datos para su posterior preprocesado.

Cluster : Esta clase agrupa los distintos ETFs por su sector, la correlación entre activos en un mismo sector es alta por lo que se elige este método para el agrupamiento de activos, esta clase hace uso de la clase **ETF** para la obtención de los datos.

ETF : Esta clase almacena el símbolo, el sector y las series temporales de datos de cada activo, interacciona con la clase vista anteriormente, podríamos considerarla como la unidad básica de almacenamiento de datos del sistema, ya que **Cluster**, **InputDataInGroup** y **InputDataTotalGroup** también realizan estas funciones pero con un mayor procesamiento de datos, que es más acusado en las dos últimas clases mencionadas.

A continuación se muestra el **diagrama de clases en detalle** del diseño de la aplicación, enfocándose en el módulo Trading System en la Figura.9, donde podemos observar la interacción interna entre las clases que han sido creadas para desarrollar esta aplicación.

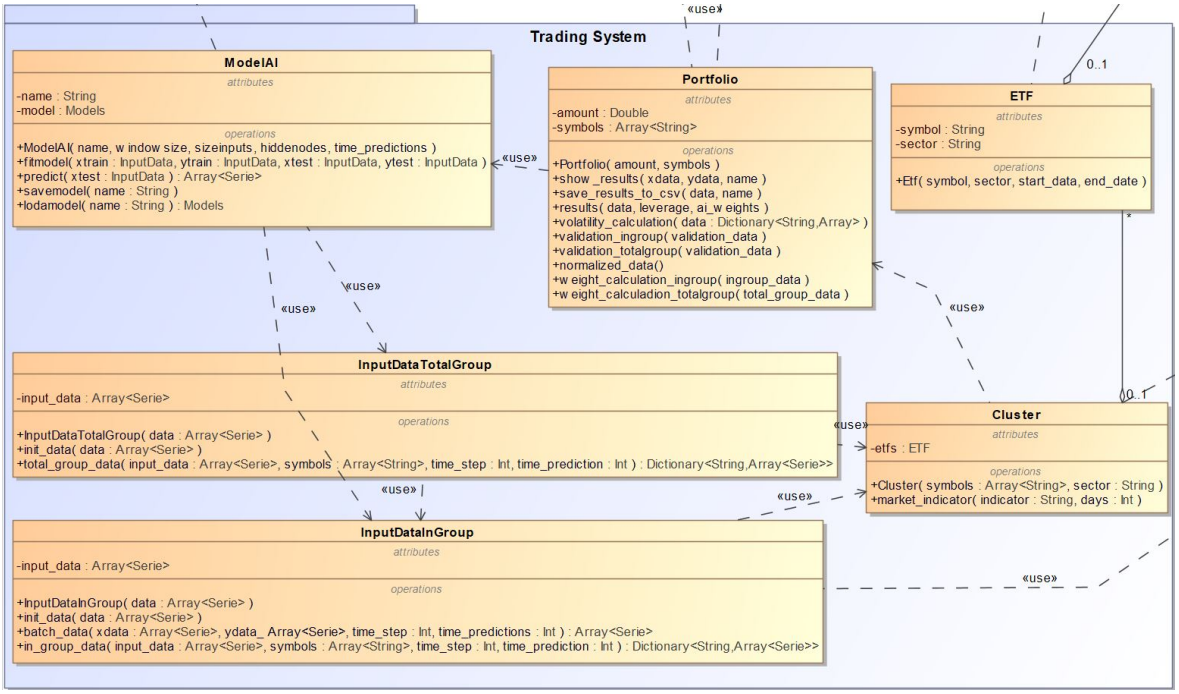


Figura 9: Diagrama de clases en detalle, interacción interna

En la siguiente Figura.10, podemos observar el uso de las distintas librerías externas "«use»", y la interacción entre las distintas clases, en el propio sistema desarrollado en el paquete Trading System.

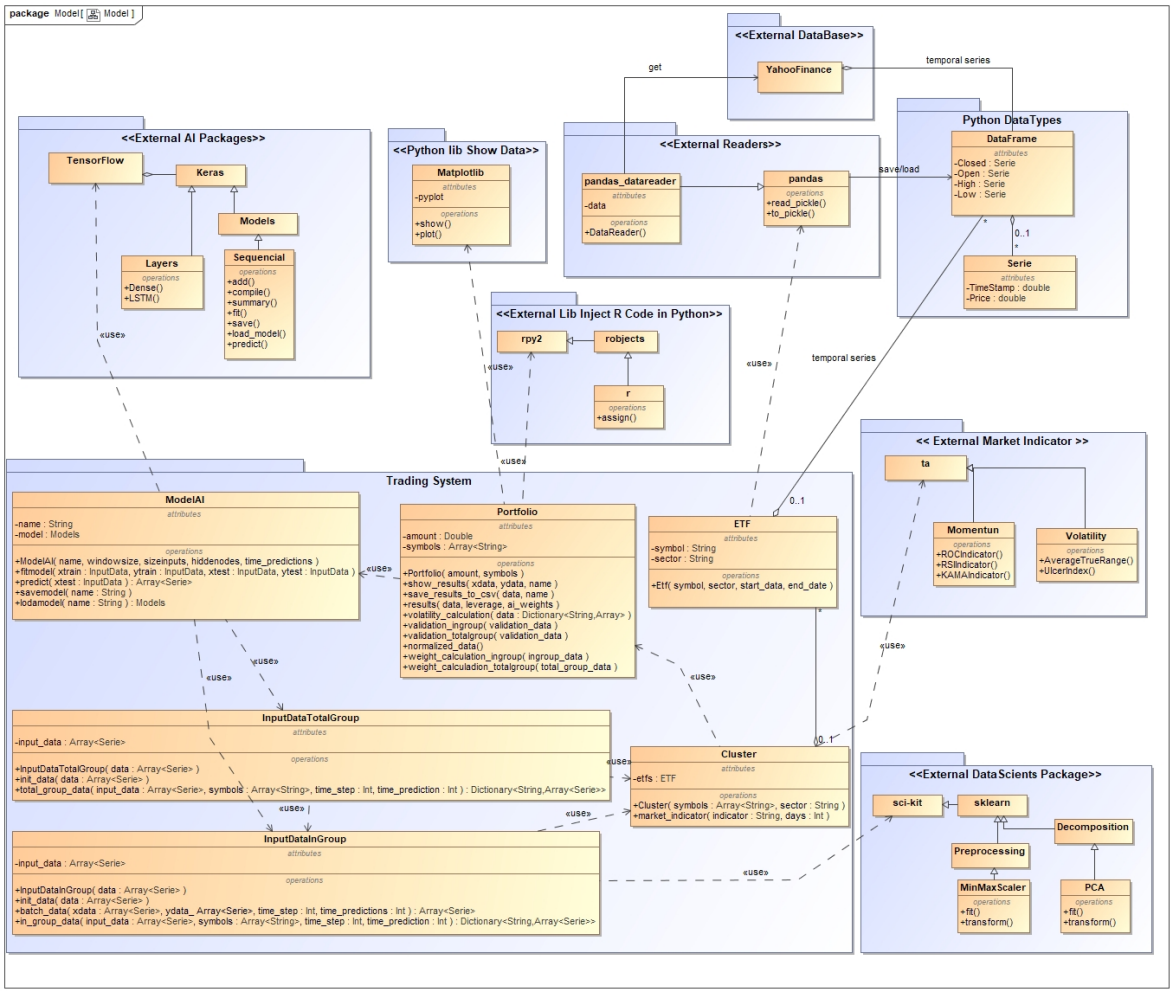


Figura 10: Diagrama de clases, interacción externa

6 Implementación

6.1. Metodología

Para el desarrollo de la implementación se sigue la metodología mencionada en la sección 3.1, adaptada a la ingeniería de software y la planificación de los sprints que puede verse en la sección 3.1.2, donde se detalla cada sprint y las fechas para su desarrollo.

6.2. Estructura general del sistema

Para el modelo de IA, se ha pensado en una estructura de dos etapas, en una primera etapa el modelo realiza una predicción por grupos, estos grupos están formados por activos correlacionados, ya que pertenecen al mismo sector, en la segunda etapa el modelo de IA hace una predicción de cada activo dentro del grupo. En la imagen de la Figura.11, podemos observar un esquéma general de la estructura del modelo del sistema.

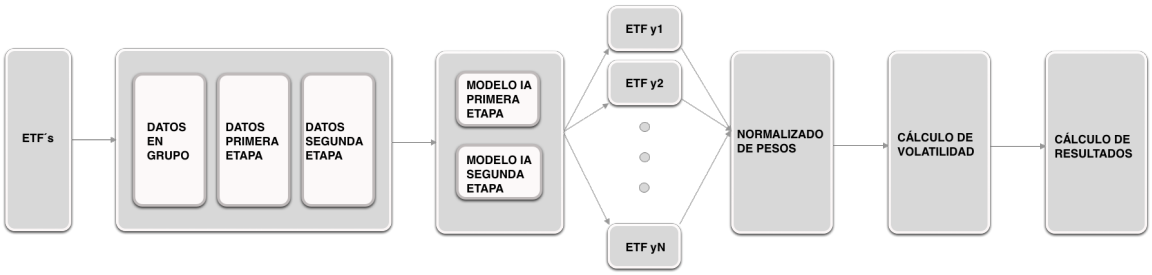


Figura 11: Estructura general del sistema

6.3. Preprocesamiento de datos

Una de las fases más importantes de la implementación es el preprocesado de datos y la preparación del vector de entrada de datos, que se puede considerar parte de este procesamien-

to. Vamos ahora a analizar esta estructura con más detenimiento y como se han dispuesto estos vectores de entrada de datos sobre las dos etapas del modelo de IA.

6.3.1. Vectores de entrada de datos

Primera etapa: Corresponde a la predicción para datos agrupados por correlación (sector), en esta etapa la entrada de datos corresponderá con la agrupación de las PCA de cada ETF perteneciente a un grupo en concreto, es decir primero se realiza un preprocesado de datos obteniendose la PCA de cada ETF de manera independiente, una vez obtenidas se agrupan, estas PCA no son obtenidas desde el precio, sino de los indicadores de mercado, tambíen en esta entrada de datos tenemos un dato de entrada referente al precio del conjunto de los ETF, la salida será la predicción de este precio conjunto. El vector de entrada de datos X queda como se aprecian en la ecuación 3, en esta primera etapa.

$$X_{primera\_etapa} = \begin{bmatrix} pc_{0,t} & \dots & pc_{0,t-s} \\ pc_{1,t} & \dots & pc_{1,t-s} \\ \dots & & \dots \\ pc_{k,t} & \dots & pc_{k,t-s} \\ P_{t,N} & \dots & P_{t-s,N} \end{bmatrix} \tag{3}$$

Donde $pc_{k,t}$ es la PCA del activo késimo en el tiempo t , $P_{t,N}$ es el precio agrupado de los ETF, en el tiempo t del grupo de activos N, s es el número de muestras seleccionadas.

Segunda etapa: Para esta segunda etapa, preparamos el vector de manera complementaria a la anterior, es decir si antes hemos usado las PCA de cada activo de manera individual, ahora usaremos una PCA del grupo de activos y el precio usado ahora será el de cada activo de manera individual. El vector de entrada de datos X queda como se aprecian en la ecuación 4, en esta segunda etapa.

$$X_{segunda\_etapa} = \begin{bmatrix} PC_{N,t} & \dots & PC_{N,t-s} \\ P_{t,K} & \dots & P_{t-s,K} \end{bmatrix} \tag{4}$$

Donde $PC_{N,t}$ es la PCA del grupo de ETF´s N en el tiempo t , $P_{t,K}$ es el precio del késimo ETF, en el tiempo t, s es el número de muestras seleccionadas.

Indicadores de mercado : Para la obtención de las PCA´s de cada ETF, se usaron ocho indicadores de mercado, de los cuales se extrajeron dos componentes principales, seis de estos indicadores son de tipo momento y los otros dos de tipo volatilidad, a continuación podemos ver la lista de estos indicadores de mercado :

- Indicadores de Momento
    - Momento a 3 meses
    - Momento a 6 meses
    - Momento a 9 meses
    - Momento a 12 meses
    - Indicador RSI
    - Indicador Kama
- Indicadores de Volatilidad
    - Indicador Average True Range
    - Indicador Ulcer Index

6.3.2. Formato del vector de entrada de datos

Lo primero a tener en cuenta es la dimensión temporal, que se estableció en el día como unidad mínima, se ha seleccionado el mes natural para identificar el número de muestras que debe tener esto vector, por tanto el número de muestras queda con $s = 22$, ya que hay que tener en cuenta que Sábados y Domingos se encuentran los mercados cerrados, tendremos una predicción de tres días. Por tanto para el formato del vector X se empaquetaron muestras de 22 días, para el formato del vector Y el empaquetado fue de tres días, en la siguiente imagen de la Figura.12, podemos observar este empaquetamiento.

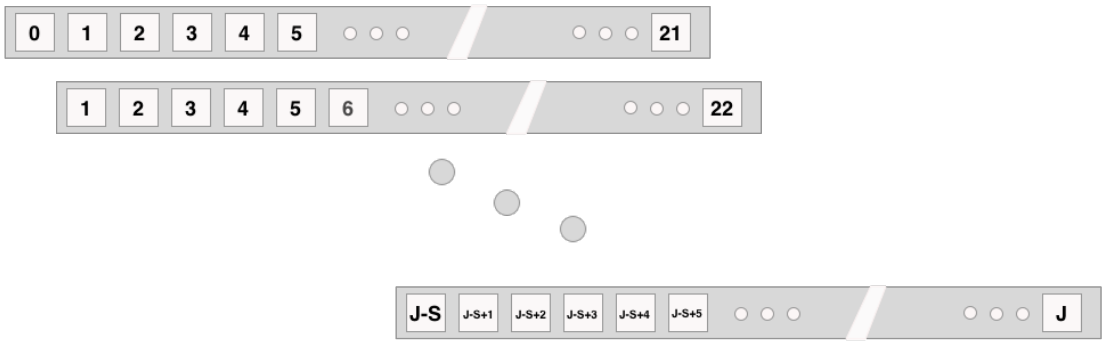


Figura 12: Formato del vector de entrada de datos

6.3.3. Partición del conjunto de los datos

Los datos una vez empaquetados como hemos visto anteriormente, se particionan en dos subconjuntos, el primer conjunto para el entrenamiento de los modelos de IA, y el segundo subconjunto destinado al test y que es de donde obtenemos los resultados finales de los modelos. El tamaño para el entrenamiento de los datos se estableció en el $65\,\%$, siendo el restante destinado al test. Se estableció la fecha de inicio en el 01-01-2010 y la fecha fin 01-01-2020 para el tamaño del conjunto total de los datos. Del segundo conjunto realizamos una nueva partición, esta vez con el objetivo de la validación de los modelos, se toma aproximadamente un $10\,\%$ del conjunto test para validación, es importante remarcar que este conjunto de datos no ha sido entrenado y tiene por objeto medir la robustez de los modelos.

6.3.4. Validación de los modelos de IA

TensorFlow tiene sus propios métodos de validación de modelos, pero normalmente estos se basan en medición de los errores entre la estimación y el resultado real, para este sistema se ha diseñado un algoritmo de validación, dado el objeto de este TFG lo que nos interesa es identificar señales de compra, para ello se diseña un sistema de validación donde se detectan verdaderos positivos, es decir señales de compra que realmente fueron una predicción acertada. Así en la ecuación.5, se detalla la fórmula utilizada para la validación de los modelos de IA.

$$V_i = (TP_i/NT_i) * 100 \tag{5}$$

Donde TP_i es el número de True Positives para el ETF i, es decir el número de señales de compras que dió la IA y que resultaron predicciones acertadas, NT_i es el número total de predecciones sobre los que se realiza la validación de los modelos para el activo i, V_i es por tanto la validación obtenida en tantos por ciento para el activo i.

En la siguiente Tabla. 6.3.4, podemos observar la precisión de estos modelos, podemos observar el porcentaje que obtuvieron en la validación cada modelo de IA, en el periodo de validación comprendido entre 15-04-2019 y 01-01-2020.

Modelo de IA	Porcentaje de precisión
Grupo Salud	64.61 %
Grupo Tecnología	64.61 %
Grupo Bienes raíces	66.15 %
Grupo Grandes capitalizaciones	66.15 %
RXL	60.0 %
PTH	63.07 %
VHT	63.07 %
IYH	58.46 %
IGM	61.53 %
VGT	61.53 %
BJK	70.76 %
PTF	64.61 %
FRI	73.84 %
IYR	56.92 %
URE	56.92 %
VNQ	64.61 %
DDM	64.61 %
JDK	61.53 %
SPY	64.61 %
SPLG	63.07 %

6.4. Modelo de IA

6.4.1. Estructura en dos etapas del modelo de IA

Esquema general de la estructura del modelo de IA, en la Figura.13, podemos observar la entrada de datos en gris para la primera etapa, el resultado obtenido que son los pesos para cada grupo de ETF´s, en la segunda etapa en gris también la entrada de datos y el resultado en blanco que son los pesos para cada ETF, en azul se procede al normalizado de datos, donde la suma de los pesos de cada ETF pertenecientes a unn grupo, no podrá superar el valor del peso asignado por la primera etapa para ese grupo. Así mismo se descartan valores negativos, ya que la gestión de carteras está centrada en una estrategia a largo y no a corto plazo.

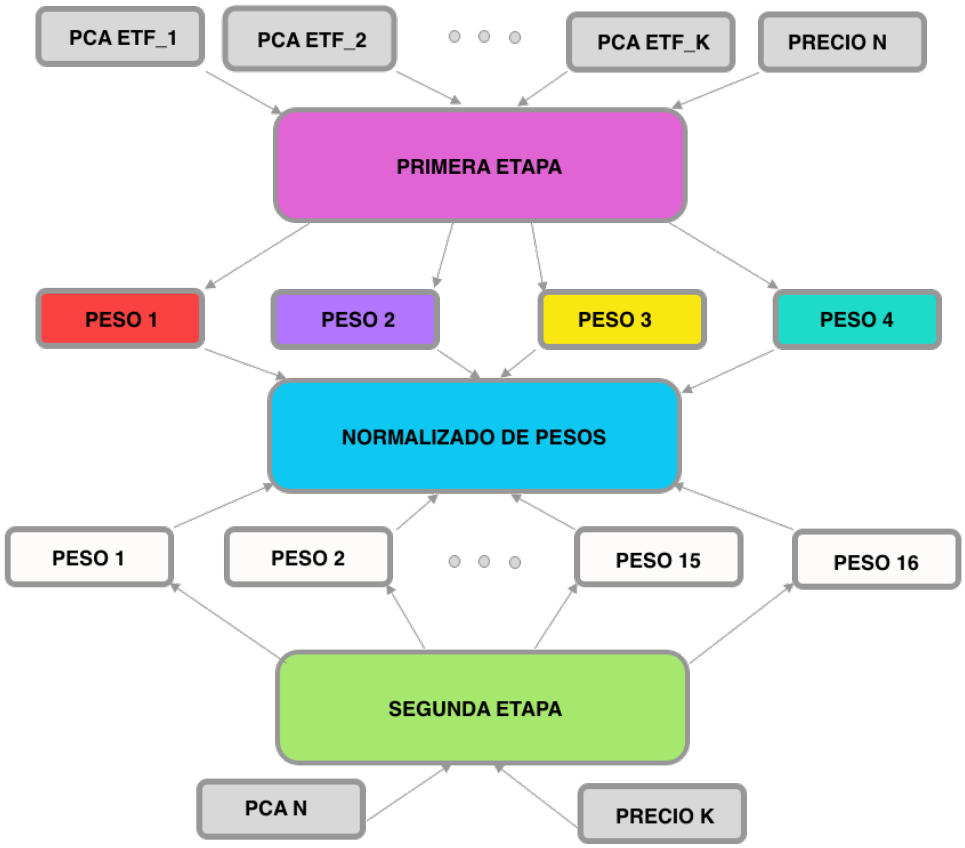


Figura 13: Estructura general del modelo de IA

6.4.2. Modelo computacional de IA

Según visto en el Capítulo Estado del Arte, una de las IA clasificadas como Deep Learning que mejor se adaptan a las series temporales son las redes LSTM, un tipo de red neuronal recurrente (RNN), que almacena memoria de corto y largo plazo, adaptándose bien a los cambios que se van produciendo en los mercados financieros. Es por esto que se va a implementar este tipo de RNN para la optimización de la cartera.

6.4.3. Estructura interna de la RNN

Esta RNN va a estar compuesta de tres capas y una última capa *Dense*, las dos primeras tienen como función de activación la tangente hiperbólica, que tiene como Rango los valores de la recta real entre $[-1$ y $1]$, la tercera capa tiene como función de activación la función sigmoide, el uso de estas funciones de activación y en este orden tiene su origen en investigaciones previas como la vista en el Capítulo Estado del Arte [3], y por el buen resultado que se ha experimentado en este TFG, en comparación con otras configuraciones. El número de nodos escondidos se estableció en un nivel ligéramente superior al tamaño de ventana, siendo estos $30, 25, 25$, también este número se basa en estudios previos [3] y en la propia experiencia de buenos resultados acaecidos.

La primera de las capas contiene 22 nodos, ya que la matriz de entrada de datos tiene 22 componentes o muestras, como podemos observar con más detalle en la sección 6.3.2, después de esta primera capa de entrada de datos se dispone de otras dos capas que estan totalmente conectadas entre ellas, para dar como salida un único resultado, por lo que la última capa, la capa *Dense* tiene un único nodo, esto único nodo nos arrojará como resultado la predicción del precio para el próximo día.

En la Figura. 14 podemos observar con mayor facilidad la estructura de la RNN.

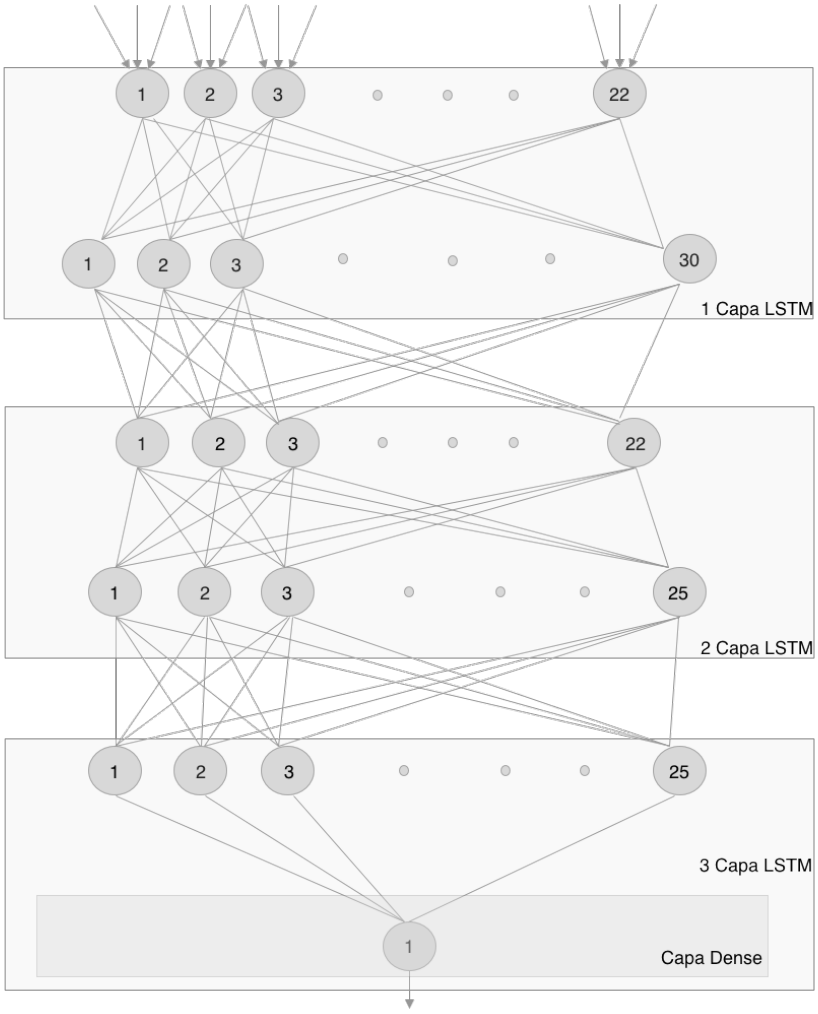


Figura 14: Esquema general interno de la RNN

6.4.4. Optimizador

Para el entrenamiento y ajuste de la IA, se optó por el optimizador *adam* un método de descenso de gradiente estocástico y por el error cuadrático medio como medida de ajuste, estos dos suelen ser los más recomendados por la comunidad y dieron excelentes resultados en este TFG en comparación con otros optimizadores y medidas de ajutes. Se estableció el entramiento en 100 épocas para todos los modelos de IA, ya que se comprobó que no se obtenía un mejor ajuste con mayores iteraciones y que esto perjudica la eficiencia del sistema, *TensorFlow* permite visualizar el ajuste obtenido según la época realizada gracias al parámetro *return_sequences*

6.4.5. Detalle de la entrada de datos

TensorFlow dispone del argumento *input_shape*, donde se especifican el tamaño de la ventana de datos, que como vimos en la sección 6.3.2 es de 22 para todos los modelos, otra de las dimensiones a especificar es el tamaño de las tuplas para la primera de las capas de la red LSTM de cada modelo de IA, esta dimensión es de 12 para los modelos grupales y 3 para los modelos individuales, en los modelos grupales este número se obtiene al sumar los precios de los datos de cada uno de los cuatro activos de cada grupo más las PCA,s correspondiente a de cada activo, que como recordamos según lo visto en la sección Preprocesamiento de Datos, tienen una dimensión de dos por cada activo y es el resultado de sintetizar los ocho indicadores de mercado para cada ETF, la salida en este modelo se corresponderá con una media de los precios de los ETF´s pertenecientes a ese grupo, en el caso de los modelos individuales la dimensión tres se obtiene de la suma del precio de ese activo ETF, con la PCA del grupo al que pertenece ese activo, la salida en este caso será el precio del ETF en concreto.

6.5. Estructura de la cartera

Se seleccionarion 16 activos ETF´s, de los que cada cuatro de estos pertenecen a un sector, creando cuatro particiones por correlación y dentro de estas, otras cuatro particiones materializados en cada ETF, como podemos observar en las siguientes tablas.

Salud
ProShares Ultra Health Care (RXL)
PowerShares DWA Healthcare Momentum ETF (PTH)
Vanguard Health Care ETF (VHT)
iShares US Healthcare ETF (IYH)

Tecnología
Shares North American Tech ETF (IGM)
Vanguard Information Technology ETF (VGT)
VanEck Vectors Gaming ETF (BJK)
PowerShares DWA Technology Momentum ETF (PTF)

Bienes raíces
First Trust S&P REIT ETF (FRI)
iShares US Real Estate ETF (IYR)
ProShares Ultra Real Estate (URE)
Vanguard Real Estate ETF (VNQ)

Grandes capitalizaciones
ProShares Ultra Dow30 (DDM)
iShares Morningstar Large-Cap ETF (JKD)
SPDR S&P 500 ETF (SPY)
SPDR Portfolio Large Cap ETF (SPLG)

6.6. Gestión de capitales mediante control del riesgo a través de la predicción de la volatilidad

Para el cálculo de la volatilidad, que es uno de los objetivos fijado en la sección 1.3.2, se usa código R inyectado en Python, mediante la librería rpy2. La forma de calcular la volatilidad utilizada es la siguiente, dado un tiempo t_0,t_{0-3} y t_{0-5}, cálculamos las pendientes de las rectas utilizando estos puntos. En la Figura.15, podemos ver una imagen de estas muestras temporales, tomada sobre la curva creada entre el eje de abscisas, donde se encuentra la dimensión temporal T y el eje de ordenadas, donde se encuentra la dimensión Precio P, este precio es una media del Precio de todos los activos de la cartera.

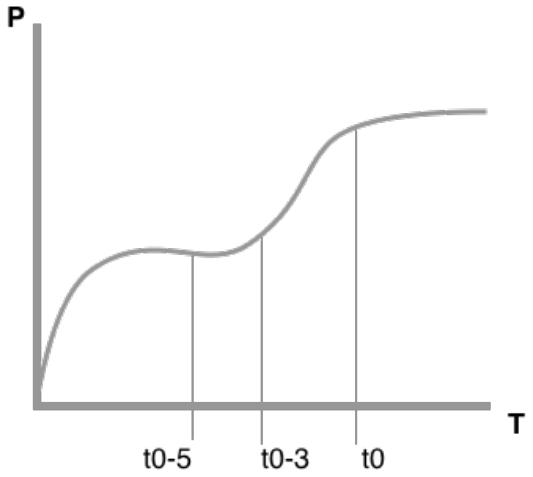


Figura 15: Muestras temporales

Con el objetivo de obtener una rectas de regresión, se realiza el cálculo de estas pendientes, estas rectas carecen del término independiente, tampoco este es necesario ya que lo que se busca es la predicción relativa del movimiento con respecto a la dimensión temporal. Una vez obtenidas estas pendientes se realiza un cálculo de la aceleración de las mismas, esto lo conseguimos mediante la dimensión temporal, es decir incremento de la pendiente por unidad de tiempo, por último se hace una comparación de la aceleración obtenida de la pendiente con mayor amplitud temporal (puntos t_0 y t_{0-5}), y la de menor amplitud (puntos t_0 y t_{0-3}), para tener una idea de si se está acelerando o decelerando la tendencia y en que sentido. Un aumento de las aceleraciones negativas será la manera de alertar sobre un volatilidad alta que perjudicará el rendimiento de la cartera.

En la Ecuación.6 podemos ver la fórmula que describe este operador.

$$R_{t0+1} = (t_0 - t_{0-3}/3) - (t_0 - t_{0-5}/5) \tag{6}$$

Este operador tiene el efecto de una segunda derivada, la cual es capaz de detecta cambios suaves de tendencia, al contrario que la primera derivada que solo es capaz de detectar cambios bruscos, por ello cuando el resultado es negativo, este operador esta detectando un cambio de tendencia, de tendencia positiva a negativa o bien una mayor aceleración de la tendencia negativa, en caso de resultado positivo la interpretación es la misma pero en sentido contrario.

En la siguiente Ecuación.7 podemos observar la cantidad de líquido disponible para invertir Q_{t0} sobre el montante original X , según la predicción de este operador.

$$Q_{t_0} = \begin{cases} 2X, & \text{si } R_{t_0+1} >= 0, \\ 0{,}5X, & \text{en otro caso.} \end{cases} \tag{7}$$

Como podemos observar cuando el operador es positivo permitimos el apalancamiento, y cuando es negativo restringimos la cantidad disponible a la mitad, este operador esta diseñado para evitar los crashs en la bolsa, donde todos los activos se verían afectados, ya que un crash de un solo sector sería identificado y prevenido por la IA. No se usa ningún umbral ya que esto podría ser motivo de overfitting, realizar un ajuste que se adaptaría muy bien a este sistema pero no a otro sistema.

7

Resultados

7.1. Resultados Obtenidos

7.1.1. Periodos temporales usados

Los periodos de pruebas fueron los comprendidos entre el 1/07/2016 y el 12/04/2019 para obtener los resultados con los distintos modelos, lo que supone el uso de 700 muestras, recordemos este periodo está fuera del entrenamiento de los modelos de IA, periodo que comprende desde el 01/01/2010 hasta el 30/06/2016 y de la validación de datos periodo desde 15/04/2019 hasta 01/01/2020. La cartera está compuesta por dieciséis ETF´s, de los cuales estos se pueden agrupar en cuatro grupos de cuatro activos cada uno, agrupándose por sectores. Se ha añadido una medida de riesgo a los resultados obtenidos, en concreto el indicador Sharpe Ratio (SR), este ha sido cálculado con la siguiente Ecuación.8 .

$$SR = R_p - R_f/\sigma_p \tag{8}$$

Donde σ es la desviación estandar del retorno de la cartera, R_p es el retorno de la cartera, y R_f es el retorno libre de riesgo. El retorno libre de riesgo se estimó en base a los bonos del Estado a 3 meses EEUU, conocidos como T-Bill, se tomó SR y este bono por ser los más ampliamente reconocidos por los inversores, se fijó esta tasa en el 1.25 %, esta tasa lleva años en niveles muy inferiores, rondando el 0 %, se hizo la media entre el valor más bajo y el más alto del periodo temporal para añadir un margen de seguridad debido a los tipos tan bajos actuales.

7.1.2. Comparación de resultados

Para poder comparar la eficiencia real o no del sistema, usando IA y usando gestión de capitales se muestran diferentes resultados, que resultan de la combinación de aplicar los modelos de IA y la gestión de capitales, o de no aplicar estos a la gestión de cartera, originando cuatro resultados de gestiones de carteras diferentes y que son las siguientes.

1. Resultado sin aplicar modelo de IA y sin gestión de capitales
2. Resultado aplicando modelo de IA y sin gestión de capilates
3. Resultando sin aplicar modelo de IA y aplicando gestión de capitales
4. Resultado aplicando modelo de IA y gestión de capilates

En la siguiente Tabla.7.1.2, podemos ver de manera esquematizada los cuatro sistemas de gestión de carteras originados dadas las diferentes combinaciones posibles.

Sistema Gestión de Cartera	Uso de modelo IA	Uso de gestión de capitales	Sección
1	No	No	7.1.3
2	Si	No	7.1.4
3	No	Si	7.1.5
4	Si	Si	7.1.6

Dados estos sistemas de gestión, se muestran los resultados en las siguientes secciones 7.1.3, 7.1.4, 7.1.5 y 7.1.6. Además de estos resultados se añade un análisis comparatorio entre uso/no uso de la IA y entre uso/no uso de la gestión de capitales.

En la Tabla.7.1.2 vemos el análisis comparativa generado de usar/no usar IA.

Análisis comparativo	Uso/No Uso de modelo de IA	Uso de gestión de capitales	Sección
1	Si	No	7.1.7

En la Tabla.7.1.2 vemos el análisis comparativa generado de usar/no usar gestión de capitales.

Análisis comparativo	Uso de modelo de IA	Uso/ No uso de gestión de capitales	Sección
2	No	Si	7.1.8

Recordemos que la aplicación del modelo de IA es un objetivo que podemos ver en la sección 1.3.1, la aplicación de gestión de capitales tiene su origen en el objetivo fijado en la sección 1.3.2, y la propia muestra de resultados es otro objetivo definido en la seccción 1.3.3.

Nota: Para la obtención de estos resultados se utilizó la misma metodología y la misma inversión inicial en todos los sistemas, para obtener unos resultados lo más comparables posibles.

7.1.3. Resultados Sistema de Gestión de Cartera 1

Para la obtención de estos resultados se ha ponderedo la cartera con el mismo peso para cada activo ETF, es decir cada uno de ellos con un peso de $1/16$ y con una estrategia *Buy and Hold*, es decir comprar y mantener. Con esta estrategia se obtuvo un beneficio total de $42891{,}81\$$ de una inversión inicial de $100000\$$, lo que supone una rentabilidad en el periodo del $42{,}89\,\%$ y una rentabilidad anual de $16{,}18\,\%$.

En la siguiente Figura.16 podemos observar la rentabilidad mensual obtenida en la parte inferior y en la superior la acumulada.

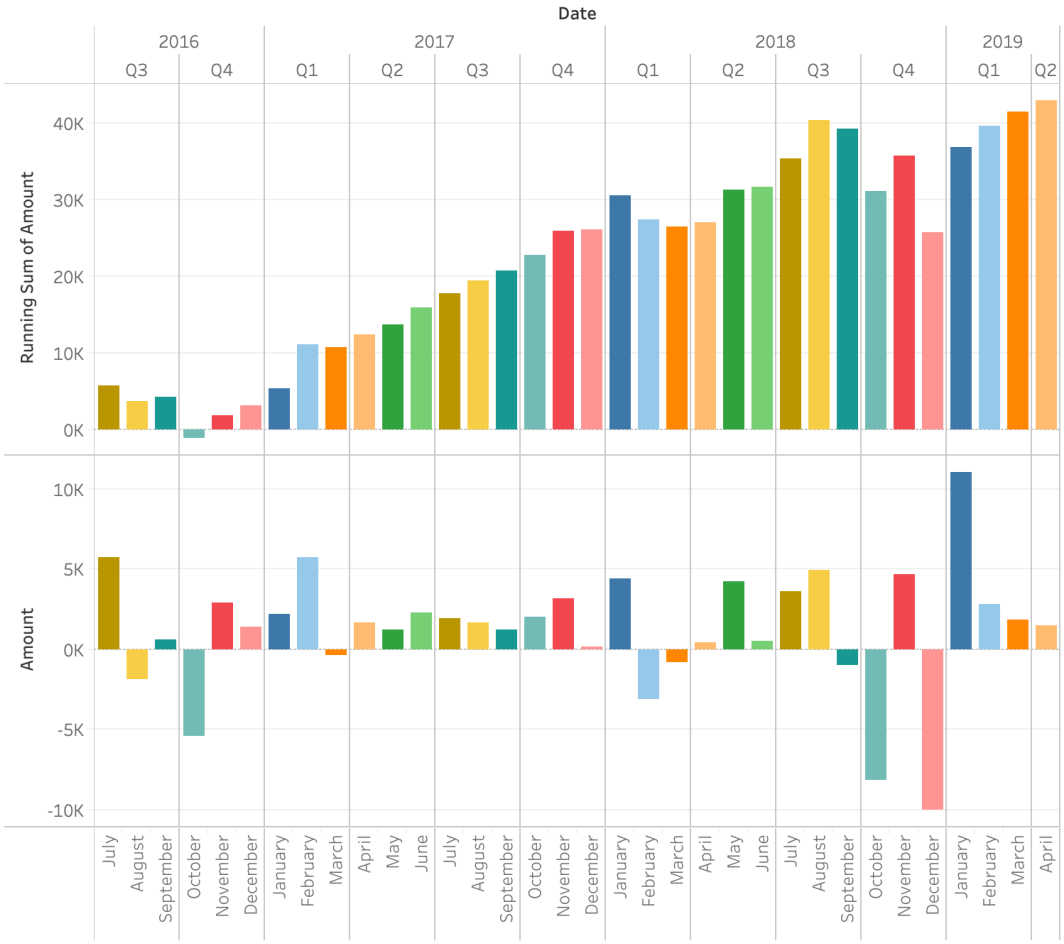


Figura 16: Rentabilidad Mensual

7.1.4. Resultados Sistema de Gestión de Cartera 2

Para la obtención de resultados, se realizó un cálculo de pesos ponderados de cada ETF, según las predicciones de la IA, asignando pesos de manera directamente proporcional en aquellos activos que se esperaban un incremento mayor del valor. La rentabilidad obtenida del periodo total fue de $49922{,}46\$$, arrojando una rentabilidad total del $49{,}92\,\%$ y anual del $18{,}83\,\%$, aumentando por tanto la rentabilidad con respecto al Sistema Gestión de Cartera 1 en un $2{,}65$ puntos anuales, lo que supone un $16{,}37\,\%$ más de rentabilidad anualmente en comparación al sistema anteriormente citado. En la Figura.17, podemos observar la rentabilidad mensual en la parte inferior y la acumulada en la superior.

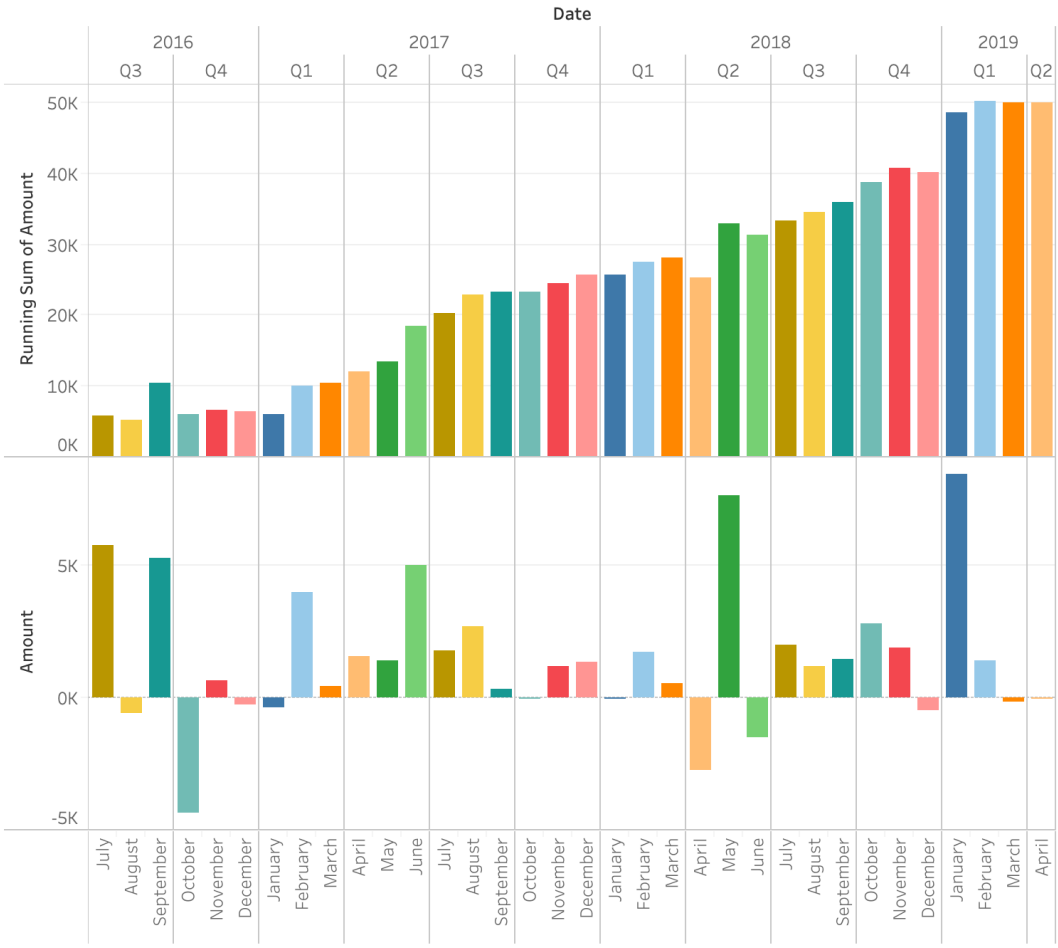


Figura 17: Rentabilidad Mensual con uso de IA

7.1.5. Resultados Sistema de Gestión de Cartera 3

Para la obtención de estos resultados se utiliza el mismo sistema que el anterior pero con el añadido de gestión de capitales mediante control del riesgo, caculado con código R. Con esta estrategia se obtuvo un beneficio total de 88875,67$ de una inversión inicial de 100000$, lo que supone una rentabilidad en el periodo del 88,87 % y una rentabilidad anual de 33,53 %.

En la siguiente Figura.18 podemos observar la rentabilidad mensual obtenida en la gráfica inferior y la mensual acumulada en la gráfica superior.

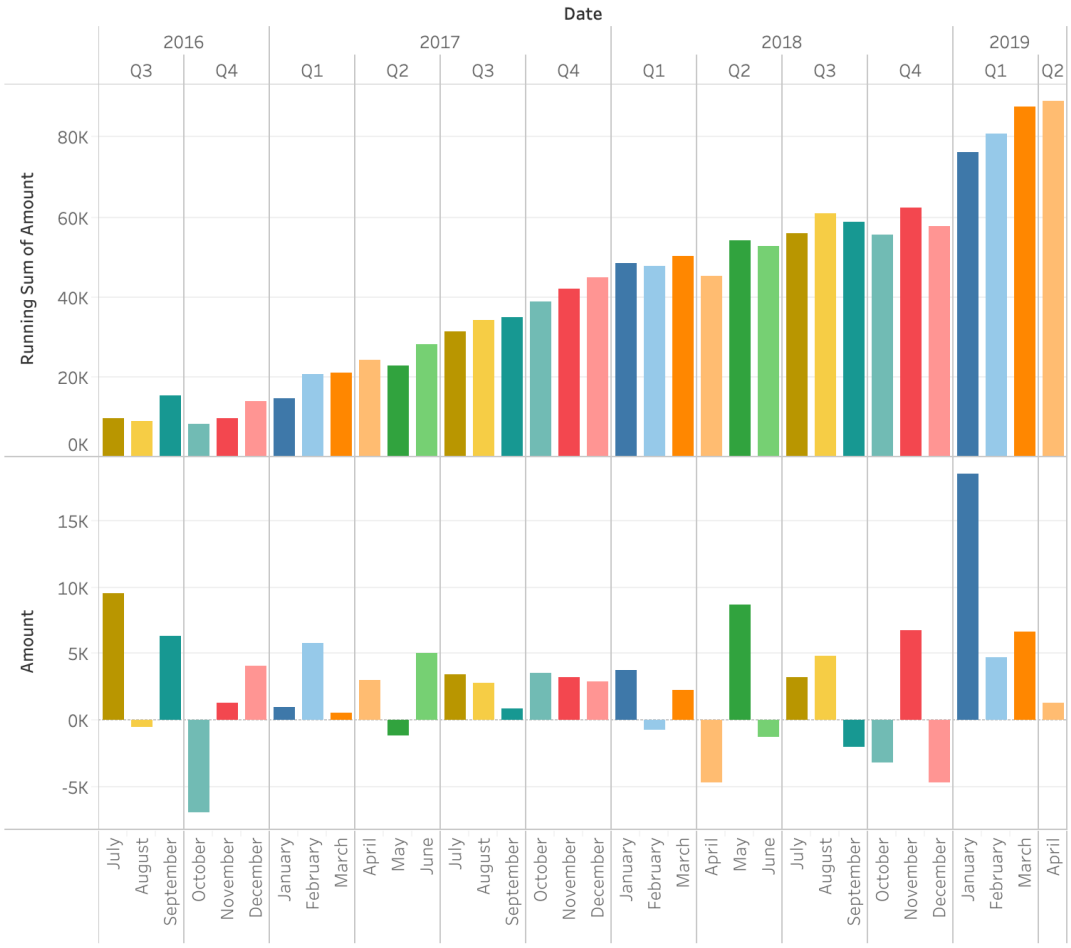


Figura 18: Rentabilidad mensual con Gestión de capitales

7.1.6. Resultados Sistema de Gestión de Cartera 4

A diferencia del Sistema anterior este usa los modelos de IA, para predicción de precios. Se obtuvo un beneficio total de 98904,13$, lo que da una rentabilidad del periodo de 98,90 %, y una rentabilidad anual de 37,32 %, lo que supone una mejora considerable con respecto a las rentabilides sin la aplicación de gestión de capitales, es decir los Sistemas Gestión de Carteras 1 y 2. Se aumentó la rentabilidad anual con respecto al Sistema de Gestión 3 en 3,79 puntos anualmente, lo que supone un incremento del 11,31 % con respecto al sistema anteriormente citado.

En la siguiente Figura.19 podemos observar la rentabilidad mensual en la gráfica inferior y la mensual acumulada en la gráfica superior.

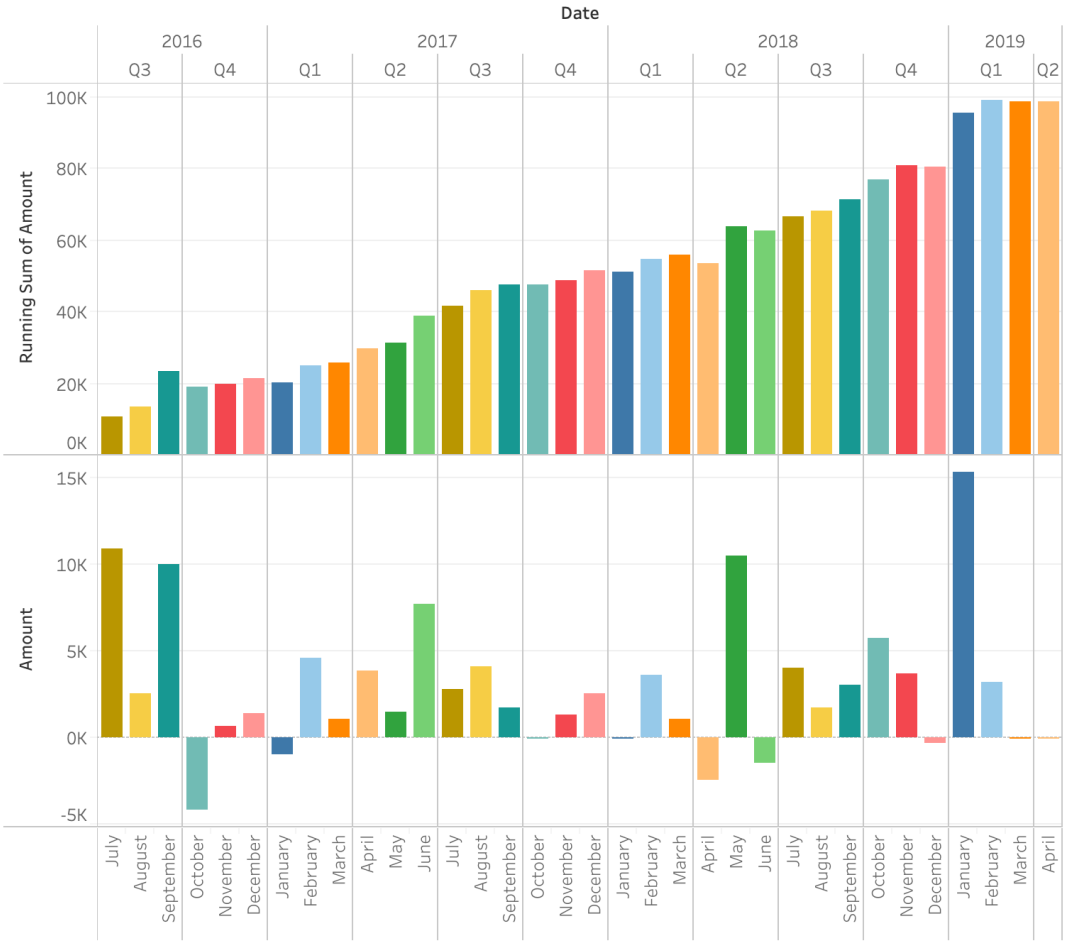


Figura 19: Rentabilidad mensual con aplicación de IA y de gestión de capitales

7.1.7. Análisis comparativo 1

En la Figura.20, vemos la comparativa de usar o no la IA, se usaron los resultados donde no interviene la gestión de capitales para poder analizar de manera aislada el uso de la IA. En la gráfica inferior podemos ver la rentabilidad mensual sin uso de IA y en la gráfica superior con uso de la IA. Podemos observar como la IA fue capaz de predecir de una manera muy aceptable los giros bajistas en los meses de Octubre y Diciembre de 2018, lográndo una rentabiliad acumulada superior al sistema que no usa la IA, en concreto 7031$, lo que supone un 2,65 % más de rentabilidad anual.

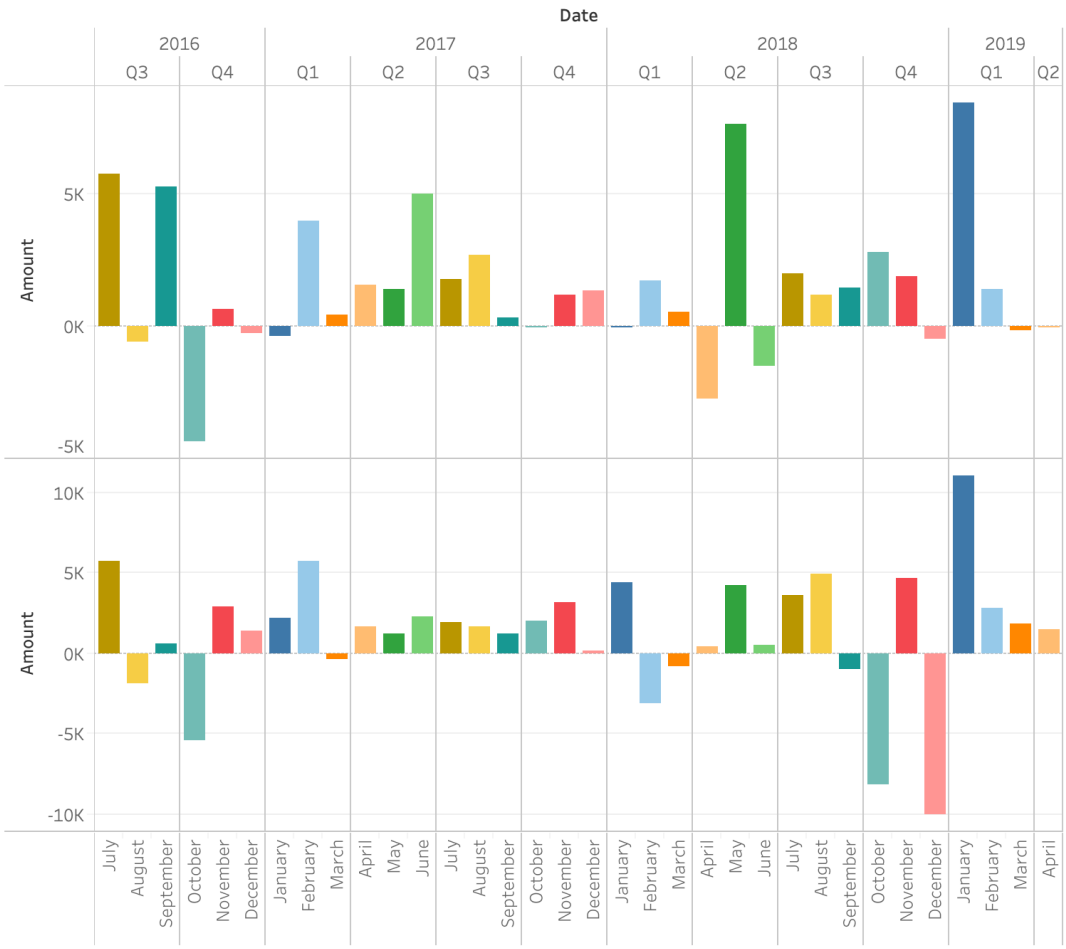


Figura 20: Comparativa uso IA

En la siguiente Tabla.7.1.7, vemos de manera esquemátizada los rendimientos obtenidos por los Sistemas de Gestión de Carteras 1 y 2.

Sistema de Gestión de Cartera	Rendimiento Total	Porcentaje anual	Sharpe Ratio	Sección
1	42891 $	16.18 %	1.09	7.1.3
2	49922 $	18.83 %	3.04	7.1.4

7.1.8. Análisis comparativo 2

En la Figura.21, se usaron los resultados donde no interviene la IA para aislar de manera más clara los resultados. En la gráfica inferior vemos una gráfica sin uso de gestión de capitales mientras en la gráfica superior vemos la gráfica con el uso de la gestión de capitales.

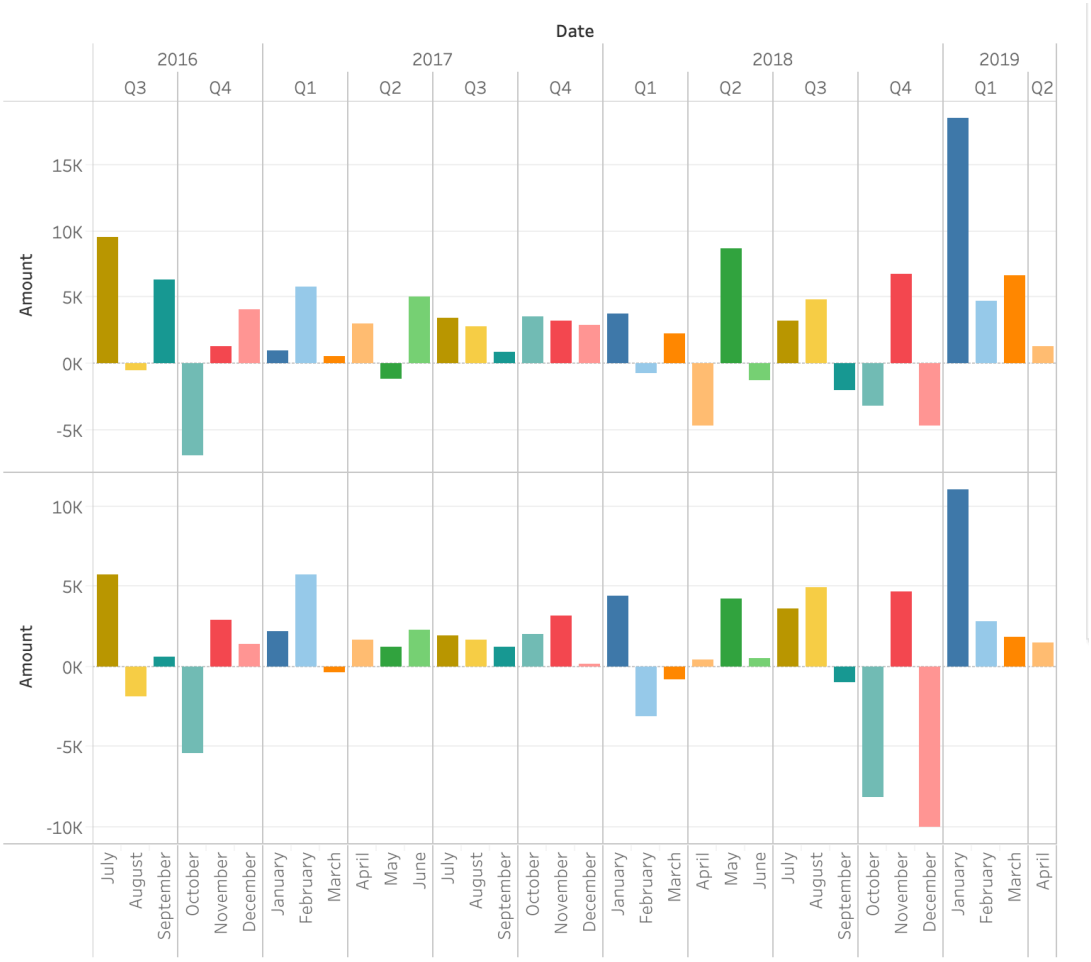


Figura 21: Comparativa uso gestión de capitales

Podemos observar con claridad la potencia de usar la gestión de capitales, y como el control del riesgo a través de la votalidad es efectivo, para esto solo hay que fijarse nuevamente en los meses de Octubre y Diciembre de 2018, donde en el sistema que no intervino la gestión de capitales obtuvo pérdidas considerables mientras el sistema que usó la gestión de capitales mediante control del riesgo, estás pérdidas fueron más limitadas, también se observa con claridad como en los meses donde se obtuvieron beneficios el sistema que usó la gestión de capitales obtuvo unos beneficios superiores, solo en el mes de Octubre de 2016 y Abril de 2018 se obtienen peores resultados, lo que supone solo dos meses de entre treinta y cuatro, por lo que fue superior en casi un 95 % de los meses. En cuanto al resultado bruto, el beneficio fue superior en un 45984$, lo que supone un 17,35 % más de rentabilidad anual con respecto a no usar la gestión de capitales.

En la siguiente Tabla.7.1.8, vemos de manera esquemátizada los rendimientos obtenidos

por los Sistemas de Gestión de Carteras 1 y 3.

Sistema de Gestión de Cartera	Rendimiento Total	Porcentaje anual	Sharpe Ratio	Sección
1	42891 $	16.18 %	1.09	7.1.3
3	88875 $	33.53 %	2.08	7.1.5

7.1.9. Tabla resumen de los resultados obtenidos

En la siguiente Figura.23, podemos observar la Desviación Estandar (DE) Mensual acumulada de los diferentes sistemas. Para el cálculo de SR, se utilizó la DE acumulada total mensual.

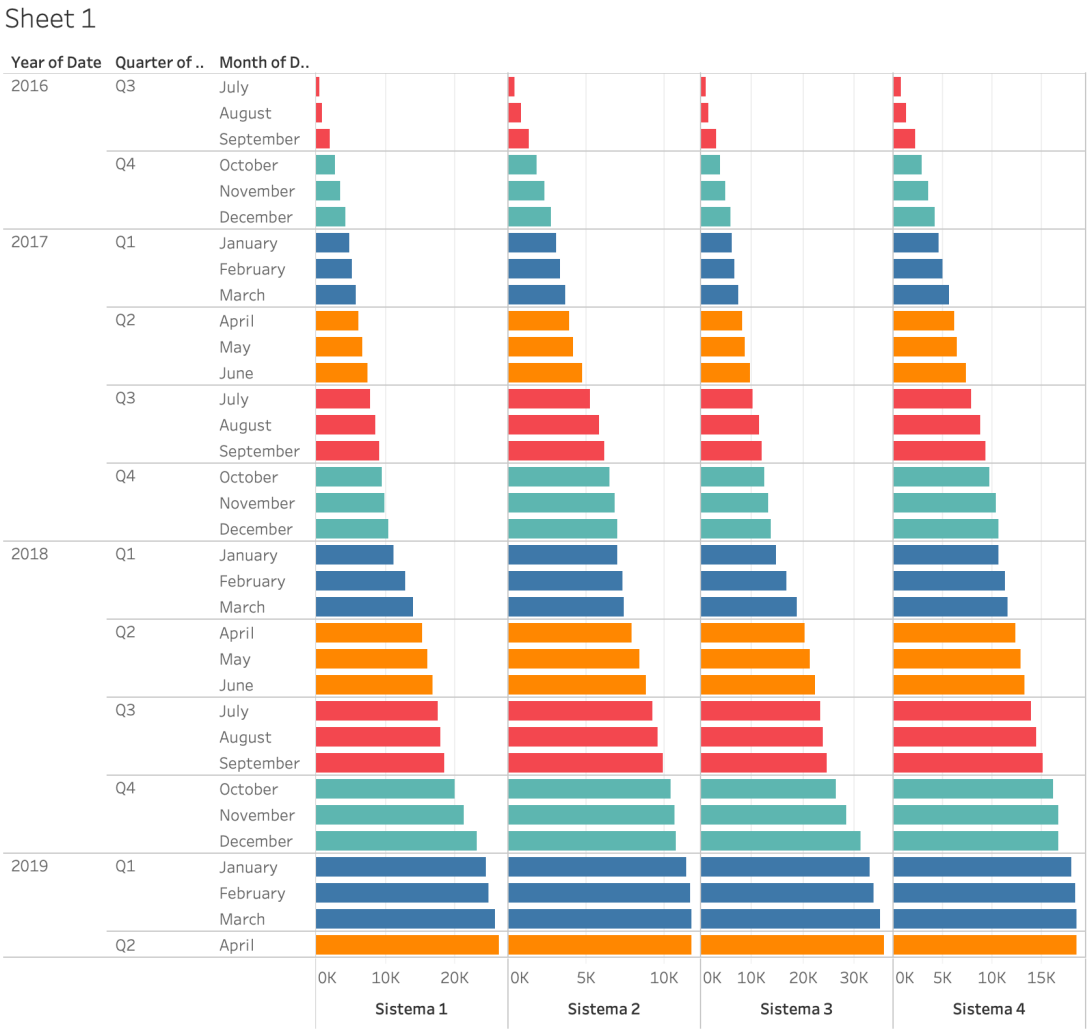


Figura 22: Desviación Standard Mensual acumulada

En la siguiente Tabla 7.1.9 podemos ver un resumen de la desviación estandar acumulada mensual y del retorno libre de riesgo acumulado compuesto.

Sistema de Gestión de Cartera	DE mensual acumulada	Retorno libre de riesgo
1	26285 $	14074 $
2	11766 $	14074 $
3	35906 $	14074 $
4	18594 $	14074 $

En la tabla 7.1.9, podemos observar los resultados obtenidos por los distintos sistemas, su rendimiento anual y su SR, recordemos que un SR por encima de 2, es considerado bueno por los inversores y por encima de 3 como excelente, en cambio por debajo de 1 es considerado ineficiente o malo. Se observa que tanto la IA, como el sistema de gestión de capitales mejoran este SR, la combinación de ambos sistemas provoca una simbiosis que hace mejorar este SR.

Sistema de Gestión de Cartera	Rendimiento Total	Porcentaje anual	Sharpe Ratio	Sección
1	42891 $	16.18 %	1.09	7.1.3
2	49922 $	18.83 %	3.04	7.1.4
3	88875 $	33.53 %	2.08	7.1.5
4	98904 $	37.32 %	4.56	7.1.6

8

Conclusiones y Líneas Futuras

8.1. Conclusiones

8.1.1. IA en la gestión de carteras

De la investigación realizada en el Capítulo 2 y de los resultados obtenidos en el Capítulo 7, y que podemos ver con mayor claridad en la Tabla.7.1.7, se puede afirmar que el uso de la IA para la gestión de cartera arroja resultados positivos y que resulta superior a la estrategia de *Buy and Hold*, es decir a comprar y mantener a lo largo del tiempo la inversión, además como se observa en la Figura.18, la IA parece especialmente útil para periodos de alta volatilidad acaecidos por algún acontecimiento inesperado que provoca una correción rápida en los mercados financieros, también podemos observar en la Tabla 7.1.9, que la IA reduce el riesgo, esto queda demostrado al aumentar el indicador SR.

8.1.2. Gestión de capitales mediante el control del riesgo a través de la volatilidad

El control del riesgo mediante la volatilidad, llevado a cabo con código R, mediante técnicas derivadas de Machine Learning como es la recta de regresión, ha permitido una gestión eficiente de capitales, como se demuestra en la Figura.19 y en la Tabla.7.1.8, aumentando las rentabilidades obtenidas en la gestión equivalente que no usa esta gestión de capitales, permitiendo un apalancamiento del capital en épocas de bonanza, además disminuye el riesgo con respecto al beneficio obtenido como se extrae de la Tabla 7.1.9, al aumentar el indicador SR.

8.1.3. Muestra de resultados

En el Capítulo.7, podemos ver los resultados obtenidos con los distintos sistemas, además de un análisis de estos resultados, la representación mediante gráficas y tablas se muestra muy adecuada para la muestra de estos resultados, se puede observar con claridad la mejora en el rendimiento obtenido tanto en la aplicación del modelo de IA, como en la gestión de capitales mediante control del riesgo a través de la predicción de la volatilidad, así como una reducción de riesgo en proporción a las ganancias obtenidas.

8.2. Líneas Futuras

8.2.1. Técnicas de IA

Este TFG se centró en el uso de Deep Learning y en concreto se usó la red neuronal LSTM, que se ha mostrado eficiente para la predicción de series temporales, queda pendiente el estudio de otras técnicas IA como puede ser la técnica Random Forest y otro tipo de RNN, realizar una comparativa con los resultados obtenidos en este TFG y poder comparar que técnica obtiene mejores rendimientos.

8.2.2. Estructura del modelo de IA

En este TFG, se usó una estructura en dos etapas, una primera etapa para predicción de valores por grupos según sectores y una segunda etapa para predicción de activos de manera individual, queda pendiente como futuras investigaciones cambiar esta estructura para estudiar la posibilidad de obtener mejores resultados mediante el uso de otro tipo de estructuras.

9

Referencias

9.1. Lista de Referencias

[1] J. Ma Weiming. *Mastering Python for Finance*, segunda edición, Packt, 2019. [En línea].Disponible en: https://learning.oreilly.com

[2] Y.Hilpish . *Artifial Intelligence in Finance*, primera edición, O'Reilly Media, Inc. 2020 [En línea]. Disponible en : https://learning.oreilly.com

[3] H. Yuna , M. Lee , Y. Seon Kang y J. Seok *Portfolio management via two-stage deep learning with a joint cost.* Octubre, 2019. Accedido en: Marzo, 03, 2021. [En línea].
Disponible en :https://www.sciencedirect.com /science/article/abs/pii/S0957417419307584

[4]M. Lopez de Prado.*Advances in Financial Machine Learning* John Wiley & Sons, Inc, 2018 [En línea]. Disponible en: https://learning.oreilly.com

[5] Harry M. Markowitz.*Portfolio Selection* Marzo, 1952.Accedido en: Marzo, 08, 2021. [En línea].
Disponible en :https://onlinelibrary.wiley.com /doi/abs/10.1111/j.1540-6261.1952.tb01525.x

[6] Harry M. Markowitz.*Foundations of Portfolio Theory* Junio, 1991. Accedido en:Febrero, 27,2021. [En línea].
Disponible en:https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1991.tb02669.x

[7] J.M Gómez y J.A Jimenez. "Selección óptima de portafolios basada en cadenas de Markov de primer y segundo orden". redalyc.org ,
https://www.redalyc.org/jatsRepo/1552/155260967005/html/index.html (Accedido en 19, Marzo, 2009).

[8]C. B. Kalayci, O. Ertenlice, M. A. Akbay.*A comprehensive review of deterministic models and applications for mean-variance portfolio optimization* Febrero, 2019. Accedido en: Marzo, 2, 2021 .[En línea].

Disponible en : https://www.sciencedirect.com/science/article/pii/S0957417419301162

[9] H.A. Tayali, S. Tolun . *Dimension reduction in mean-variance portfolio optimization*Agosto, 2017. Accedido en: Marzo, 5, 2021. [En línea].
Disponible en: https://www.sciencedirect.com/science/article/abs/pii/S0957417417306139

[10]R. Wang, F. Nie , R. Hong, X. Chang, X. Yang y W. Yu. *Fast and Orthogonal Locality Preserving Projections for Dimensionality Reduction* Octubre, 2017. Accedido en: Marzo, 7, 2021.[En línea]. Disponible en: https://ieeexplore.ieee.org/document/7976386

[11] F. Nie, D. Xu, X. Li, S. Xiang. *Semisupervised Dimensionality Reduction and Classification Through Virtual Label Regression* Junio, 2011. Accedido en: Marzo, 09, 2021.[En línea] .
Disponible en : https://ieeexplore.ieee.org/document/5645697

[12] S. Tolun, H.A. Tayali *Data Mining for Multicriteria Single Facility Location Problems* 2016. Accedido en: Marzo, 09, 2021. [En línea] .
Disponible en: https://www.igi-global.com/gateway/chapter/150291

[13] R. Gaujoux,C. Seoighe. *A flexible R package for nonnegative matrix factorization* Julio, 2010. Accedido en: Marzo, 03, 2021. [En línea].
Disponible : https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-11-367

[14]C.D. Sigg,J.M. Buhmann *Expectation-maximization for sparse and non-negative PCA* Julio, 2008. Accedido en: Marzo, 03, 2021. [En línea].
Disponible en: https://dl.acm.org/doi/10.1145/1390156.1390277

[15] E. Alpaydin. *Introduction to Machine Learning* , cuarta edición, MIT press, Marzo, 2020.[En línea] .Disponible en: https://books.google.es/books?hl=es&lr=&id=tZnSDwAAQBAJ

[16] C. lorio, G. Frassio, A. D´Ambrosio, R.Siciliano. *A P-spline based clustering approach for portfolio selection* Abril, 2018. Accedido en: Marzo, 03, 2021. [En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S095741741730787X

[17] D. León, A.Aragón, J.Sandoval, G.Hernández, A.Arévalo,J.Niño. *Clustering algorithms for Risk-Adjusted Portfolio Construction*, 2017. Accedido en: Marzo, 09, 2021. [En línea].
Disponible en:https://www.sciencedirect.com/science/article/pii/S187705091730772X

[18] ”Machine learning”, wikipedia.org. https://en.wikipedia.org/wiki/Machine_learning . (Accedido en Marzo, 17, 2021).

[19] A. Fenerich et al. *Use of machine learning techniques in bank credit risk analysis* Septiembre, 2020.Accedido en: Marzo, 26, 2021. [En línea].

Disponible en: https://www.scipedia.com/public/Fenerich_et_al_2020a

[20] M.J. Ariza-Garzón, J.Arroyo, A. Caparrini y M.J.S. Vargas. *Explainability of a Machine Learning Granting Scoring Model in Peer-to-Peer Lending* Marzo, 2020. Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://ieeexplore.ieee.org/abstract/document/9050779

[21] A.S. Krah, z. Nikolic y R. Korn. *Machine Learning in Least-Squares Monte Carlo Proxy Modeling of Life Insurance Companies* Febrero, 2020.Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://arxiv.org/abs/1909.02182

[22] J.B. Heaton, N.G. Polson, J.H. Witte. *Deep learning for finance: deep portfolios* Octubre, 2016. Accedido en: Marzo, 09, 2021. [En línea].
Disponible en: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2838013

[23] T. Fischer, C. Krauss. *Deep learning with long short-term memory networks for financial market predictions* Diciembre, 2017. Accedido en: Marzo, 09, 2021. [En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0377221717310652

[24] C. Olah. Ünderstanding LSTM Networks", github.io.
http://colah.github.io/posts/2015-08-Understanding-LSTMs (accedido Marzo, 10, 2021).

[25] Y.Ma, R.Han y W.Wang.*Portfolio optimization with return prediction using deep learning and machine learning* Marzo, 2021. Accedido en: Marzo, 26, 2021.[En línea]. Disponible en: https://www−sciencedirect−com.uma.debiblio.com/science/article/pii/S0957417420307521

[26] H.Jacobs *What explains the dynamics of 100 anomalies?* Agosto, 2015. Accedido en: Marzo, 09, 2021. [En línea]. Disponible en:
https://www.sciencedirect.com/science/article/pii/S0378426615000679?via %3Dihub

[27] J. Green, J.R.M Hand y X.F.Zhang . *The supraview of return predictive signals* Junio, 2013. Accedido en: Marzo, 09, 2021. [En línea].
Disponible en:https://link.springer.com/article/10.1007/s11142-013-9231-1

[28] P.M. Addo, D. Guegan y B. Hassani .*Credit Risk Analysis Using Machine and Deep Learning Models* Junio, 2018. Accedido en:Marzo, 30, 2021.[En Línea].
Disponible en : https://search.proquest.com/docview/2125015174

[29] Van-Sang Ha y Ha-Nam Nguyen. *Credit scoring with a feature selection approach based deep learning* 2016. Accedido en: Marzo, 30, 2021.[En línea]. Disponible en: https://www.matec-conferences.org/articles/matecconf/abs/2016/17/matecconf_mimt2016_05004

[30] W. Chen, H. Zhang, M.K. Mehlawat y L. Jia*Mean–variance portfolio optimization using machine learning-based stock price prediction* Noviembre, 2020.Accedido en: Marzo, 26, 2021. [En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S1568494620308814

[31] J.Nobre y R.F. Neves .*Combining Principal Component Analysis, Discrete Wavelet Transform and XGBoost to trade in the financial markets* Febrero, 2019. Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0957417419300995

[32] S.Dey, Y. Kumar, S.Saha y S. Basak *Forecasting to Classification: Predicting the direction of stock market price using Xtreme Gradient Boosting* OCtubre, 2016. Accedido en: Marzo, 26, 2021.[En línea]. Disponible en: https://www.researchgate.net/publication/309492895

[33] .ªn End-to-End Guide to Understand the Math behind XGBoost",analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost (Accedido Marzo, 26, 2021).

[34] P.Jain y S.Jain. *Can Machine Learning-Based Portfolios Outperform Traditional Risk-Based Portfolios? The Need to Account for Covariance Misspecification* Julio, 2019.Accedido en: Marzo, 26, 2021.[En línea]. Disponible en: https://www.mdpi.com/2227-9091/7/3/74

[35] M.L de Prado. *https://search.proquest.com/docview/1826404948* 2016. Accedido en: Marzo, 26, 2021.[En línea]. Disponible en: https://search.proquest.com/docview/1826404948

[36] F.D. Paiva, R.T.N. Cardoso, G.P. Hanaoka, W.M. Duarte *Decision-making for financial trading: A fusion approach of machine learning and portfolio selection* Agosto, 2018. Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0957417418305037

[37] Q.Wen, Z.Yang, Y.Song y P.Jia.*Automatic stock decision support system based on box theory and SVM algorithm* 2010.Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0957417409005107

[38] V.N. Vapnik.*An Overview of Statistical Learning Theory* Septiembre, 1999. Accedido en: Marzo, 26, 2021.[En línea].
Disponible en: https://ieeexplore–ieee–org.uma.debiblio.com/document/788640

[39] Ho, T. K. Random decision forests". In *Proceedings of the third international conference on document analysis and recognition* 1995, pp. 278–282.

[40] L.Breiman *Random Forests* 2001 .Accedido en: Marzo, 26, 2021.[En línea].Disponible en:https://link.springer.com/article/10.1023/A:1010933404324

[41] L.O. Orimoloye, M-C. Sung, T.Ma y J.E.v. johnson. *Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices* Julio, 2019. Accedido en: Marzo, 27, 2021.[En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0957417419305305

[42] R.Singh y S.Srivastava. *Stock prediction using deep learning* Diciembre, 2016. Accedido en: Marzo, 27, 2021.[En línea]. Disponible en: https://link.springer.com/article/10.1007/s11042-016-4159-7

[43] Y. LeCun y Y.Bengio. *Convolutional Networks for Images, Speech, and Time-Series* Accedido en: Marzo, 27, 2021.[En línea].
Disponible en: https://www.researchgate.net/publication/2453996

[44] E.Hoseinzade y S.Haratizadeh *CNNpred: CNN-based stock market prediction using a diverse set of variables.* Accedido en: Marzo, 27, 2021.[En línea].
Disponible en: https://www.sciencedirect.com/science/article/pii/S0957417419301915

[45] O.B. Sezer y A.M.Ozbayoglu. *Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach* Accedido en: Marzo, 27, 2021.Disponible en: https://www.sciencedirect.com/science/article/pii/S1568494618302151

[46] F.Soleymani y E. Paquet. *Financial portfolio optimization with online deep reinforcement learning and restricted stacked autoencoder—DeepBreath* Febrero,2020. Accedido en: Marzo, 27, 2021.[En línea]. Disponible en:
https://www−sciencedirect−com.uma.debiblio.com/science/article/pii/S0957417420302803

[47] T.Wong y Z.Luo. *Recurrent Auto-Encoder Model for Large-Scale Industrial Sensor Signal Analysis* Springer. Julio, 2018.[En línea].
Disponible en: https://link.springer.com/chapter/10.1007 %2F978-3-319-98204-5_17

[48] Y.LeCun, Y.Bengio y G. Hinton. *Deep learning.* Accedido en: Marzo, 27, 2021.[En línea].
Disponible en: https://www.nature.com/articles/nature14539

Apéndice A
Manual de Instalación

A.1. Introducción

Este TFG, se desarrolló con el entorno de desarrollo **PyCharm**, que es específico para **Python**, en el sistema operativo MacOs , por lo que

se recomienda lanzar la aplicación en este entorno de desarrollo y con este sistema operativo , ya que la configuración del proyecto está realizada para este IDE y para este SO, si se realiza en Windows puede requerir configuraciones adicionales. Este entorno de desarrollo tiene dos versiones la Community, que es gratuita y para desarrollo de Python puro y la versión Professional que es de pago y está diseñada para desarrollo científico y para desarrollo web, de estas dos versiones escogemos la **Professional**, ya que la UMA dispone de licencia para uso de esta herramienta y nos ofrece más ventajas que la versión pura Community. Para muestra de resultados además de la librería de Python Matplotlib se usó la herramienta Tableau, que carga los archivos csv generados en los resultados finales, por lo que se explica también la instalación de esta herramienta.

A.2. Pasos de Instalación

A.2.1. Instalación de Python

Cómo se puede observar en la Figura.23, debemos instalar Python 2.7, Python 3.5 o más reciente, como también vamos a instalar **TensorFlow**, ya que lo requiere la aplicación, instalaremos la versión 3.5 o una más reciente hasta la 3.8, que son la que admite TensorFlow hasta la fecha.

A continuación se detallan una serie de pasos para la correcta instalación del Python. Antes de realizar esta acción hay que cerciorarse de que Python no está ya previamente instalado,

también puede haber conflicto si se instalan diferentes versiones, por lo que se recomienda tener instalada solo una versión.

1. Visitar la página https://www.python.org/downloads/
2. La web reconocerá el SO, sino lo hace seleccionar el nuestro
3. Descargar el instalador (ojo versión 3.5 en adelante hasta 3.8)
4. Seguir los pasos del instalador
5. Para cualquier duda en la página hay un asistente de pasos a seguir

A.2.2. Instalación de Código R

Es necesario la instalación de Código R en el equipo, podemos realizar la descarga e intalación del mismo a través de la página https://www.r-project.org/.

A.2.3. Instalación de PyCharm

Primero veamos los requisitos de instalación en la Figura.23, en este caso para el Sistema Operativo (SO) Mac, para consultar los requisitos de otro SO, seleccionar en la página web del punto 1 de instalación la pestaña Requisitos del sistema.

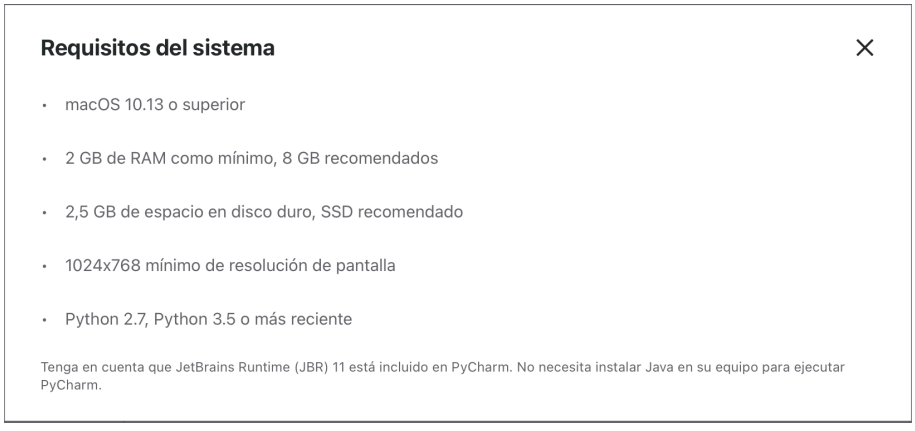


Figura 23: Requisitos de instalación

A continuación se detallan una serie de pasos para la correcta instalación del entorno de desarrollo.

1. Visitar la página https://www.jetbrains.com/es-es/pycharm/download
2. Seleccionar el Sistema Operativo de nuestra máquina
3. Descargar el instalador
4. Seguir los pasos del instalador
5. Para cualquier duda en la página hay un asistente de pasos a seguir

A.2.4. Instalación de diferentes paquetes necesarios en PyCharm

Es necesario descargar el plugin de R en PyCharm : como podemos ver en la siguiente Figura.24 donde se pueden ver los pasos necesarios para su instalación en el entorno de desarrollo de PyCharm. Enlace : https://www.jetbrains.com/help/pycharm/r-plugin-support.html

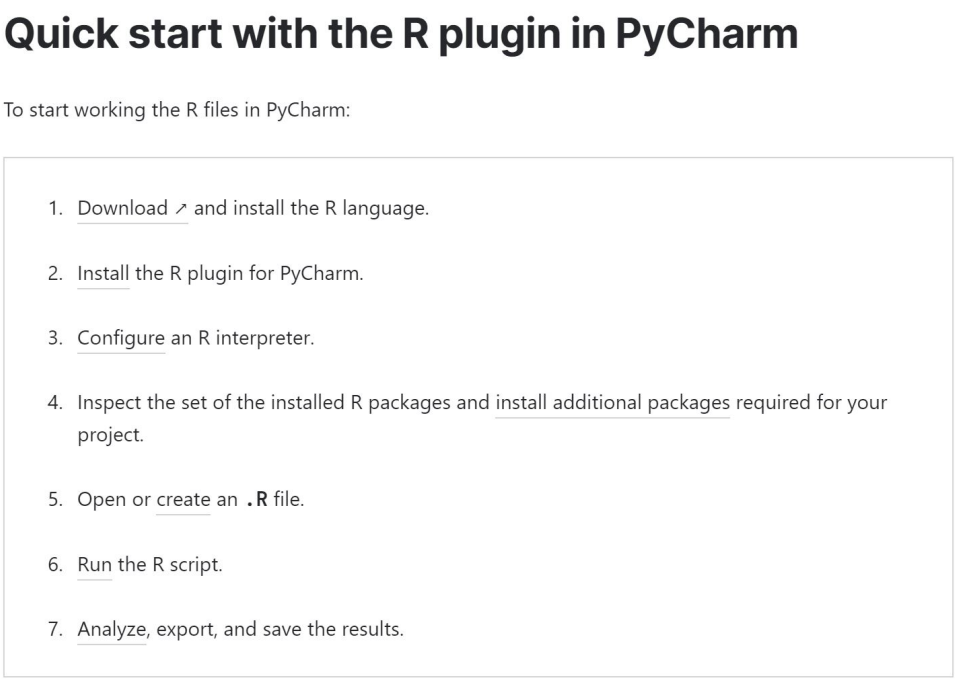


Figura 24: Pasos instalación R en PyCharm

Instalación de distintos paquetes desde el terminal: Debemos realizar la instalación de los siguientes paquetes desde el terminal.

1. pip install pandas

2. pip install pandas_datareader
3. pip isntall ta
4. pip install sklearn
5. pip install tensorflow
6. pip install rpy2

A.2.5. Instalación de Tableau

Esta herramienta es de pago, la UMA tiene licencia, aunque también dispone de una versión de prueba de 14 días. A continuación se detallan los pasos a seguir. Utilizaremos la versión Desktop.

1. Visitar la página https://www.tableau.com/products/trial
2. Seleccionar el Sistema Operativo de nuestra máquina
3. Descargar el instalador
4. Seguir los pasos del instalador

Apéndice B
Manual de Usuario

B.1. Introducción

Una vez instalado correctamente el entrono de desarrollo, Python, Código R y su plugin en PyCharm, acorde a nuestro SO (se recomienda MacOs), procedemos con la ejecución del programa.

B.2. Abrir el proyecto en PyCharm

Para ejecutar el proyecto debemos abrir PyCharm y seleccionar la pestaña Open, como vemos en la Figura.25

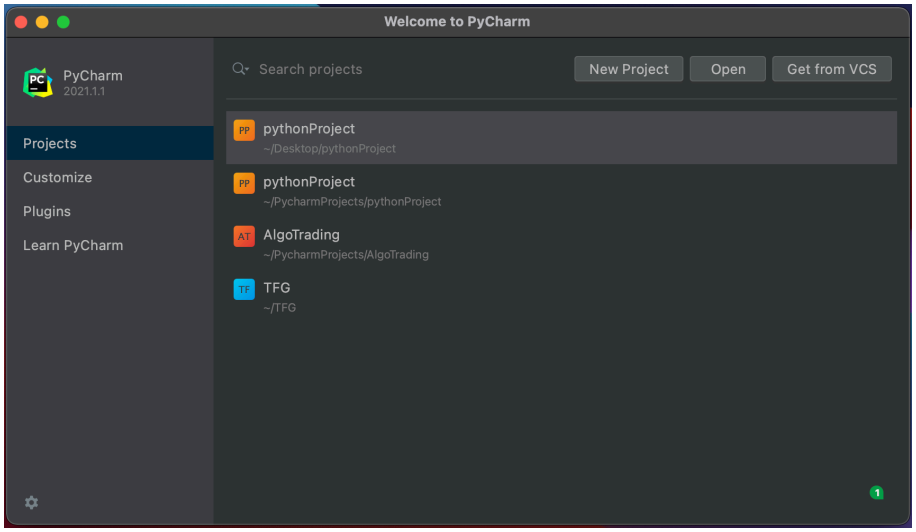


Figura 25: Menú inicio PyCharm

A continuación seguir los siguientes pasos.

1. Descomprimir el fichero que contiene el proyecto
2. Abrir desde la pestaña open del menú de PyCharm

3. El proyecto se abrirá en el entorno de desarrollo Figura.26

La vista de la Figura.26, es la vista por defecto de la versión Professional para **ciencia de datos**, seleccionar esta sino lo está en : Ventana/Restaurar vista por defecto.

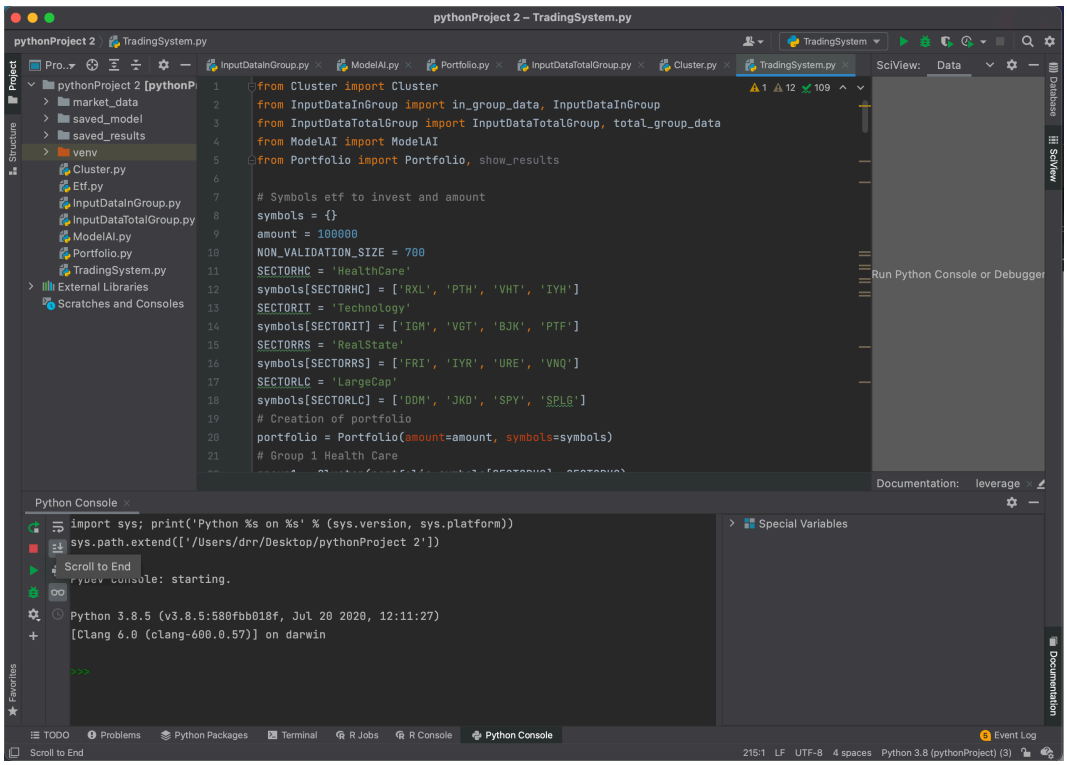


Figura 26: Vista por defecto con proyecto abierto

Como se observa, en la parte central podemos ver el código fuente, seleccionando el fichero deseado en la parte superior izquierda, sino podemos ver estos ficheros, pulsar sobre Project en la parte superior izquierda, para hacerlos visibles. En la parte inferior tenemos la consola de comandos, en la parte inferior derecha se pueden observar las diferentes variables que se van creando según el programa se está ejecutando (incluso permite la visualización de los datos que contiene), y por último en la parte superior derecha se mostrarán la visualización de resultados y si pulsamos sobre alguna variable en concreto nos mostrará su contenido si se trata de un DataFrame, u otra estructura de datos que no es posible mostrar en la parte inferior izquierda.

B.3. Configuración

Para ejecutar el programa antes debemos realizar una configuración de ejecución del proyecto, en la Figura.27, observamos el menú de configuración para ejecución del proyecto.

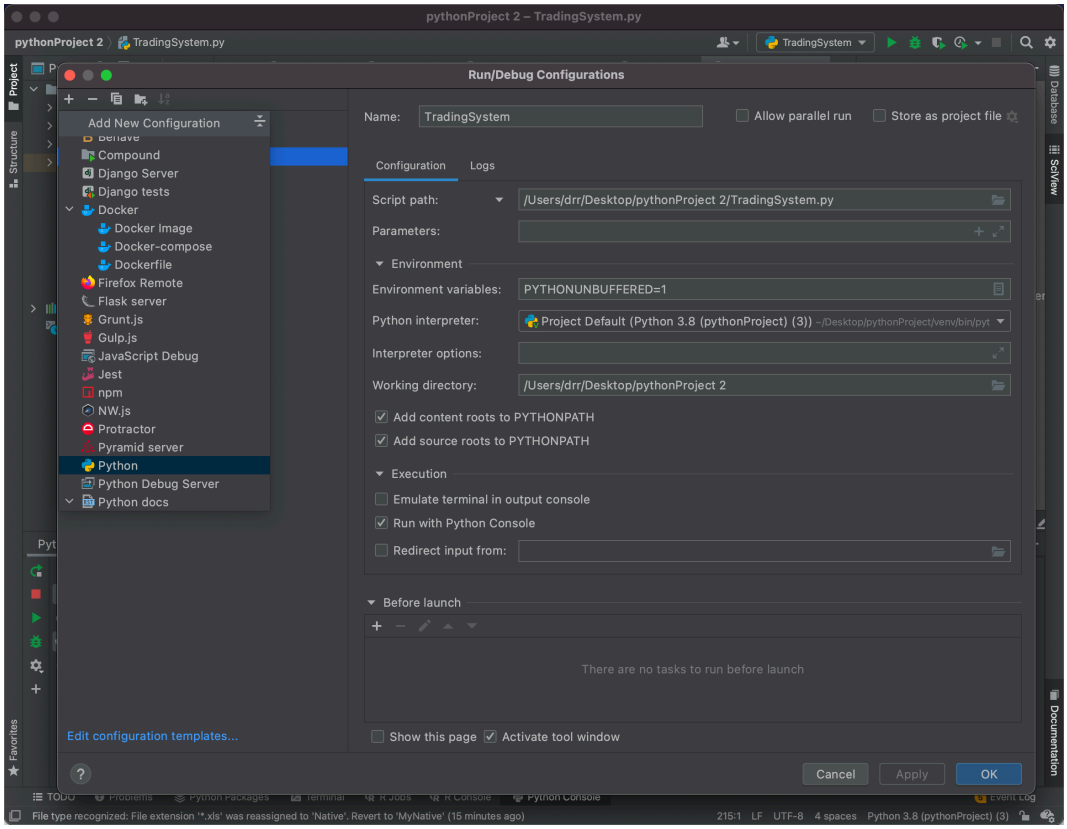


Figura 27: Configuración

Para abrir este menú basta con pulsar en la parte superior derecha el desplegable que se encuentra a la izquierda del símbolo *play*, y pulsar sobre editar configuraciones. Se nos mostrará una pantalla como la de la Figura.**??**, donde al pulsar sobre el símbolo +, de la parte superior izquierda podremos seleccionar una nueva configuración Python. Ahora procedemos a seleccionar en el apartado Script path, el archivo TradingSystem.py de la carpeta descomprimida donde se encuentra nuestro proyecto, y seleccionamos en Python interpreter el interprete Python deseado . Es posible que haya que añadir el path donde se encuentra nuestro proyecto a la variable de entorno. Seleccionar las distintas casillas tal como se muestra en la imagen.

B.4. Ejecución

Después de realizar todas estas acciones pulsamos sobre el botón OK, y ya podemos ejecutar nuestro proyecto desde el símbolo *play* de la vista principal.

El programa realizará toda la ejecución automáticamente, y mostrará los resultados en la parte superior derecha y en línea de comandos, finalizando con un .ªll okz finalizando el programa gracias a la última línea de código de TradingSystem exit(0).

El programa guarda los modelos y los distintos conjuntos de datos necesarios tales como datos de mercado y resultados en formato csv, de esta manera la ejecución es más eficiente y no es necesario perder mucho tiempo en la ejecución del proyecto, ya que solo en el entrenamiento de los modelos lleva horas, pero al estar ya entrenados no es necesario tanto tiempo y simplemente se cargan los modelos previamente guardados y que son adjuntados en el proyecto.

Nota: Es posible que haya que realizar la instalación de algún componente, y tengamos algún error en alguna línea de código, para solucionar esto solo basta con poner el ratón encima de la línea errónea e instalar el componente que nos indique PyCharm, esto se hará de maner automática pulsando sobre instalar ”nombre del componente”

B.5. Comprobación de la correcta ejecución

Es posible usar la función debug para ir viendo la ejecución del programa, pero es posible realizar otra acción para facilitar el visionado de la correcta ejecución del proyecto, podemos comentar la línea de código exit(0), simplemente usando la almohadilla #, como se muestra en la Figura.28, de esta manera el programa no finalizará aunque si mostrará todos los resultados y podremos navegar por las distintas variables en la parte inferior derecha y asi comprobar la correcta ejecución del programa y ver que contiene cada variable, lo cual puede resultar muy útil. Podemos ver en la Figura.28, como se ha comentado la línea exit(0), y como en la parte inferior se puede observar la variable portfolio y lo que contiene esta, en contreto muestra los pesos normalizados dentro de esta varibla de cada activo ETF. En la parte superior derecha se observan los resultados obtenidos, y en línea de comandos también una serie de resultados como validación de los modelos y resultados finales totales.

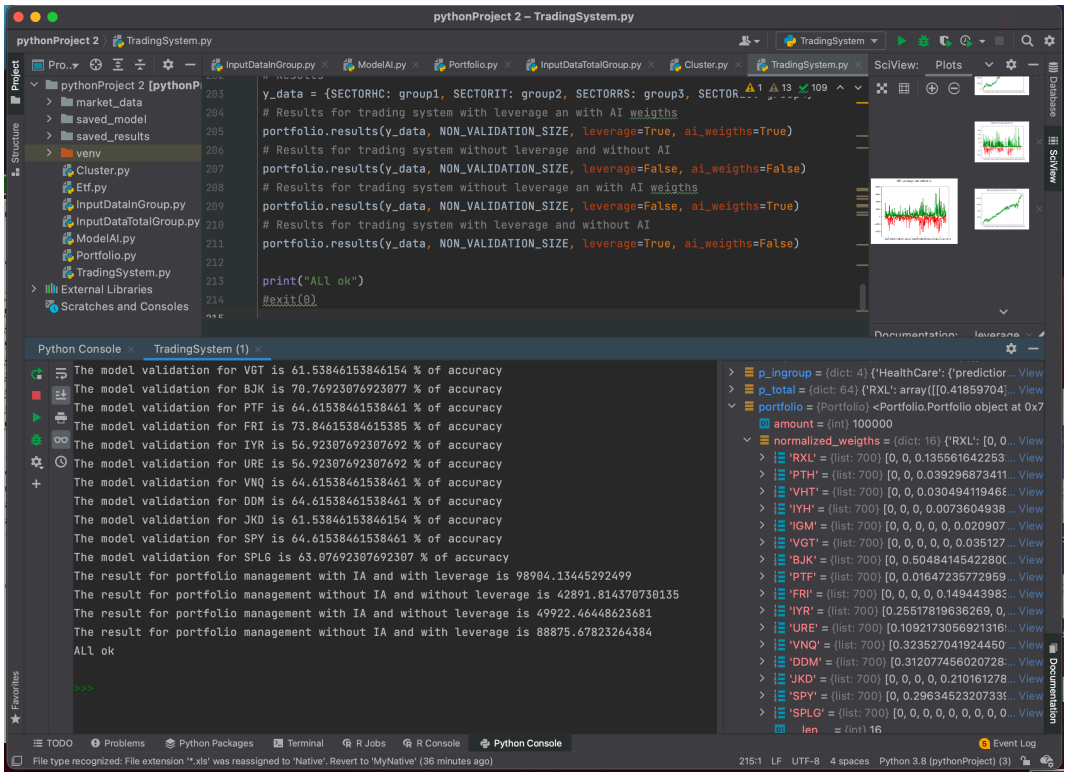


Figura 28: Visionado de variables

B.6. Muestra de resultados con Tableau

Podemos ver que en el proyecto hay una carpeta con el nombre saved_results, en esta carpeta tenemos los archivos csv, generados en la obtención de los resultados, para poder visualizarlos desde Tableau abrimos la aplicación, como podemos ver en la Figura.29, usamos la pestaña File, en la parte superior derecha y a continuación seleccionamos Open File, buscamos la carpeta saved_results dentro de nuestro proyecto y seleccionamos el resultado que queramos mostrar, posteriormente podremos abrir otro resultado y realizar muestra conjuntas de diferentes tipos de resultados.

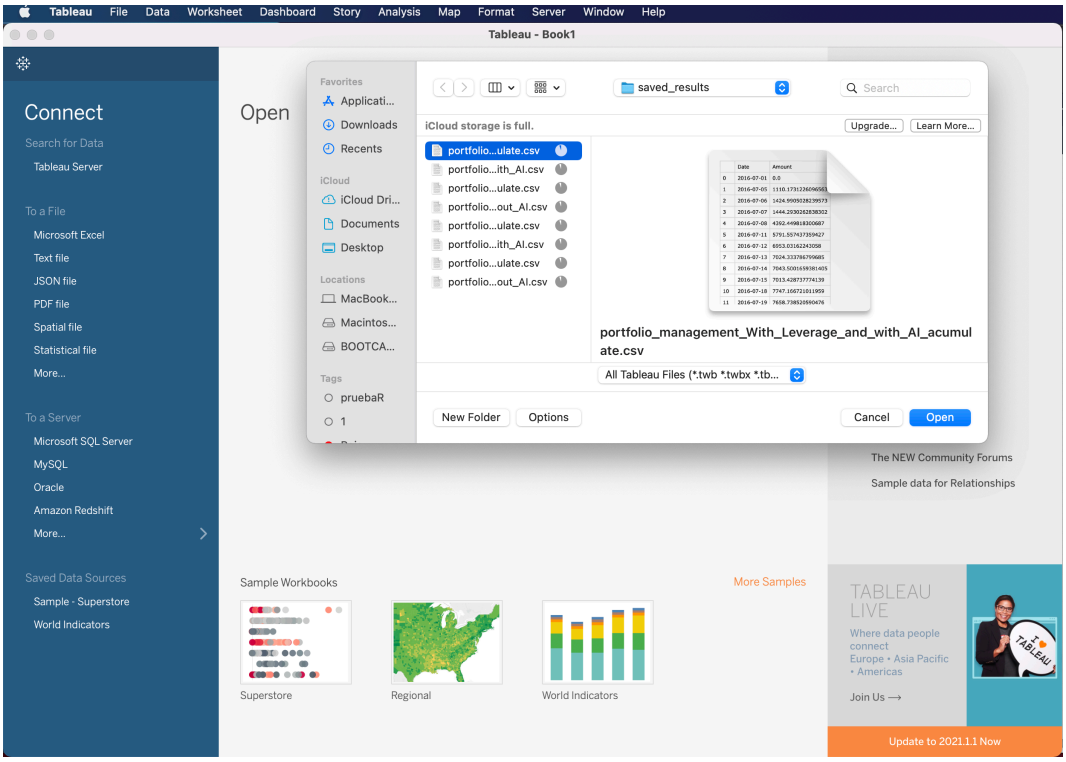


Figura 29: Muestra de resultados con Tableau

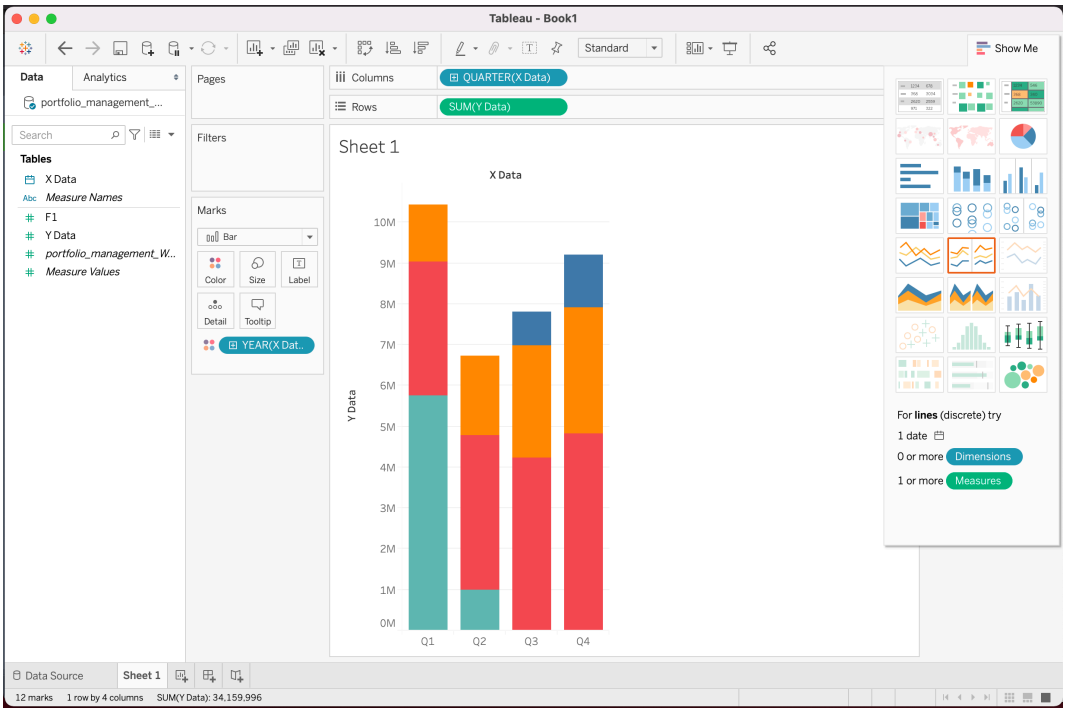


Figura 30: WorkSheet Tableau

Posteriormente a la carga del archivo deseado, pulsamos sobre ir a WorkSheet y nos mos-

trará una pantalla como la de la Figura.30, donde en ella podremos realizar distintas muestras y vistas de resultados. Tambíen podremos cargar más datos de otros resultados y seleccionarlos en la parte superior izquierda en *Data*, bastará luego con arrastrar hasta Columns o Rows el conjunto de datos deseado.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA