Trabajo Final de Máster

Titulo a definir

David Díaz Rodríguez

Xosé Manuel Martínez Filgueira

Ramón Miñones Crespo

11/2/23

Table of contents

# Agradecimientos

# Resumen

# Introducción

# 1. Capítulo I: Consideraciones teóricas

El objetivo de este capítulo es brindar una comprensión teórica exhaustiva sobre aspectos fundamentales relacionados con las series temporales financieras y su caracterización, así como el uso de redes neuronales artificiales y la programación cuadrática en la previsión y composición de carteras. En primer lugar, se analizará en detalle la caracterización de las series temporales financieras, explorando su naturaleza y características clave para comprender su comportamiento y aplicaciones en el ámbito financiero. Además, se examinará el uso de redes neuronales artificiales en la previsión de series de tiempo, incluyendo los antecedentes de su aplicación y centrándose específicamente en las Redes Neuronales Convolucionales y las denominadas Long Short-Term Memory. A continuación, se abordará la composición de carteras mediante programación cuadrática, presentando el problema y las técnicas asociadas, y se explorará en detalle el método dual en la optimización de carteras. A través de este capítulo, se sentarán las bases teóricas necesarias para comprender y aplicar de manera efectiva estas técnicas en capítulos posteriores, permitiendo una comprensión profunda y la capacidad de utilizar estos conceptos de manera práctica en el contexto financiero.

## 1.1 Caracterización de las series temporales financieras

Este epígrafe se divide en dos sub-epígrafes. En el primero se presentan los conceptos fundamentales relacionados con las series de tiempo, incluyendo sus características, componentes y clasificaciones, y se establece su relación con el análisis de las series cronológicas y la previsión, que es el objetivo de este informe. En el segundo sub-epígrafe se profundiza en las características de los precios de las acciones, comenzando por una descripción general y avanzando hacia aspectos más específicos, así como en la importancia de la estructura en la que se encuentran los datos relacionados con estos.

### 1.1.1 Series de tiempo y sus características

En este sub-epígrafe se analizan varios textos, a partir de los cuales se puede concluir que, una serie de tiempo o serie cronológica es como el registro de los valores de una o más variables en intervalos regulares de tiempo (por ejemplo, diario, semanal, semestral, anual, etc.). El análisis de las series de tiempo puede tener distintos fines, como describir el comportamiento de las variables o predecir o pronosticar sus valores futuros, lo que es especialmente relevante para las series financieras.

El análisis de las series de tiempo es una herramienta estadística que permite estudiar el comportamiento de una variable a lo largo del tiempo. Sin embargo, no existe un consenso único sobre los componentes que se deben considerar en este tipo de análisis. Algunos autores, como Kocenda and Cerný (2017) y Anderson et al. (2017), proponen que las series de tiempo se pueden descomponer en tres componentes: tendencia, estacionalidad y ruido. Otros autores, como Dodge (2008) y Espallargas and Solís (2012), sugieren que se debe añadir un cuarto componente: el ciclo. Finalmente, hay autores que plantean que las series de tiempo pueden tener hasta cinco componentes estos son los casos de IBM (2021) y Chirinos (2018).

Tendencia: La tendencia es el patrón de cambio a largo plazo que se observa en una serie de datos. Se puede definir como la dirección general y persistente de las variaciones de la serie a lo largo del tiempo. La tendencia puede ser positiva ([Figure 1.1](#fig-tend_pos)), negativa ([Figure 1.2](#fig-tend_neg)) o nula ([Figure 1.3](#fig-tend_nul)), dependiendo de si la serie aumenta, disminuye o se mantiene constante en el largo plazo. La tendencia se puede identificar mediante el análisis gráfico o mediante métodos estadísticos. La tendencia es importante para entender el comportamiento histórico y proyectar el futuro de una serie de datos. Este componente es común en los distintos criterios mencionados.

Estacionalidad: Llamada también variación cíclica regular: Se refiere a la variación correspondiente a los movimientos de la serie que ocurren cada cierto periodo de tiempo, [Figure 1.4](#fig-est_err). Este componente es, al igual que la tendencia, común en los criterios mencionados. Diferenciándose en que aquellos autores que exponen cuatro y cinco componentes llaman estacionalidad a las variaciones periódicas correspondientes a periodos menores o iguales a un año (como son periodicidad diaria, semanal, mensual, o anual), mientras que las variaciones periódicas correspondientes a periodos mayores las contemplan en un componente llamado, variaciones cíclicas. Por lo que para determinar la estacionalidad de una serie de tiempo es necesario analizarlas en un periodo no menor de dos años.

Un componente que no se puede explicar por los otros elementos de la serie de tiempo es la variación irregular o error. Este componente también se conoce como variación aleatoria, ruido o residuo, y se muestra en la [Figure 1.4](#fig-est_err). La variación irregular es común en los tres criterios mencionados anteriormente. Algunos autores distinguen entre la variación irregular, que es ocasional y aleatoria, y la variación atípica, que es causada por eventos aislados que alteran el comportamiento de la serie. La variación atípica se puede clasificar en varios tipos: aditiva, de innovación, de cambio de nivel, transiente, de estacionalidad aditiva y de tendencia local.

Una forma de categorizar las series de tiempo es según el grado de variabilidad que presentan a lo largo del tiempo, según lo expuesto en Villagarcía (2006) se puede distinguir entre series homocedásticas y heterocedásticas. Las series homocedásticas son aquellas que mantienen un rango constante de variación, como se muestra en la [Figure 1.3](#fig-tend_nul). Por el contrario, las series heterocedásticas son aquellas que cambian el rango de variación, aumentando o disminuyendo su amplitud, como se ilustra en las [Figure 1.1](#fig-tend_pos) y [Figure 1.2](#fig-tend_neg).

Las series cronológicas son un tipo de proceso estocástico que se caracteriza por ordenar las variables aleatorias según el tiempo. Esto significa que cada momento tiene asociado un valor de la variable que depende del azar y que puede cambiar a lo largo del tiempo. Según Ruiz (2011), un proceso estocástico es “una colección o familia de variables aleatorias, ordenadas según un subíndice que en general suele ser el tiempo” (p.01).

Un proceso estocástico es una colección de variables aleatorias que representan la evolución de un fenómeno aleatorio en el tiempo. Según Ruiz (2011), los estados son los posibles valores que puede tomar la variable aleatoria, y pueden ser discretos o continuos, dependiendo de la naturaleza de la variable. Asimismo, el tiempo puede ser discreto o continuo, según los cambios de estado se produzcan en intervalos fijos o variables. Ruiz (2011) clasifica los procesos estocásticos en cuatro tipos, según el espacio de estados y el tiempo, como se muestra en la [Table 1.1](#tbl-pro_esto).

Según la definición de serie temporal ofrecida anteriormente, “una serie de tiempo o serie cronológica es como el registro de los valores de una o más variables en intervalos regulares de tiempo”, y a lo expuesto con respecto a los procesos estocásticos en Castillo and Varela (2010), Villavicencio (2010) y el propio Ruiz (2011) se coincide con lo afirmado por este último: “una serie temporal es una realización parcial de un proceso estocástico de parámetro tiempo discreto” (p.09). Debido a que, en el análisis de series temporales, aunque el tiempo es continuo se adoptan intervalos discretos (días, semanas, meses, años), los cuales se establecen por convenio por los analistas o por las fuentes que suministran los datos.

Un concepto clave en el análisis de series de tiempo es el de estacionariedad. Una serie de tiempo es estacionaria cuando sus propiedades estadísticas, como la media, la varianza y la covarianza, no cambian con el tiempo. Esto implica que la serie no presenta tendencia, ciclos ni estacionalidad. Como señalan Castillo and Varela (2010), Villavicencio (2010) y Ruiz (2011), la estacionariedad es una condición necesaria para poder predecir el comportamiento futuro de una serie de tiempo usando técnicas estadísticas. En la [Figure 1.3](#fig-tend_nul) se muestra un ejemplo de una serie de tiempo estacionaria.

Las series de tiempo financieras presentan heterocedasticidad, es decir, varianzas que cambian en el tiempo. Esto implica que no son estacionarias y que su comportamiento depende de factores externos. Para verificar la estacionariedad de una serie de tiempo, se pueden utilizar diferentes métodos, como el correlograma, que muestra las funciones de autocorrelación y autocorrelación parcial de la serie, o las pruebas de raíz unitaria, como la de Dickey Fuller o la de Phillips Perron, que contrastan la hipótesis nula de que la serie tiene una raíz unitaria. Estos métodos se explican con más detalle en Castillo and Varela (2010), Villavicencio (2010) y Ruiz (2011). La [Figure 1.5](#fig-acf_pacf) ilustra un ejemplo de correlograma para una serie de tiempo financiera.

### 1.1.2 Características de los precios

Invertir en acciones o cualquier otro bien que cotice en el mercado de valores es una tarea compleja y desafiante, que requiere una comprensión profunda de las tendencias y fluctuaciones del mercado. En el centro de esta comprensión se encuentra la capacidad de analizar e interpretar los datos de precios del mercado de valores, lo que brinda información clave sobre el comportamiento de los participantes del mercado y los factores que impulsan los movimientos del mercado. El propósito de este sub-epígrafe es proporcionar una descripción general completa del entorno de los precios de las acciones y como son representados comunmente los mismos, señalando los aspectos más importante para la aplicación de las técnicas que se exploraran en los siguientes epígrafes.

Como se explica en la CNMV (n.d.b) las bolsas de valores son mercados organizados donde se negocian acciones y otros valores, como renta fija, warrants, certificados y fondos cotizados. En BME (n.d.) se expone que, en España, existen cuatro bolsas tradicionales (Madrid, Barcelona, Bilbao y Valencia) que forman parte del holding BME (Bolsas y Mercados Españoles), que también integra otros segmentos y sistemas de negociación, compensación y liquidación de valores. Siendo, como se explica en CNMV (n.d.c), el Sistema de Interconexión Bursátil Español (SIBE) la plataforma que permite la contratación continua y electrónica de todos los valores admitidos a cotización en las cuatro bolsas españolas.

Como expone la CNMV (n.d.a) las acciones son valores mobiliarios que representan una parte proporcional del capital social de una sociedad anónima, y sus tenedores son socios propietarios de la misma. Las acciones pueden negociarse en bolsas de valores o en otros mercados secundarios autorizados.

A partir de lo expuesto en Mitchell (2020), Pinset (2021) y C. Team (2023) se puede concluir que para explicar el precio de las acciones de una empresa, se pueden considerar los siguientes factores:

* La oferta y la demanda de las acciones en el mercado: si hay más compradores que vendedores, el precio subirá y viceversa. Esto depende de las expectativas y la confianza de los inversores en el futuro de la empresa.
* Los cambios en la gestión o la producción de la empresa: si la empresa mejora su eficiencia, su rentabilidad o su innovación, el precio de sus acciones puede aumentar. Por el contrario, si la empresa tiene problemas internos, pierde competitividad o se ve afectada por crisis externas, el precio puede bajar.
* La reputación de la empresa: si la empresa tiene una buena imagen pública, se asocia con éxitos o logros, o recibe buenas valoraciones de los analistas, el precio de sus acciones puede subir. Por el contrario, si la empresa se ve involucrada en escándalos, demandas o controversias, o recibe malas valoraciones de los analistas, el precio puede bajar.

En los textos Pinset (2021), T. I. Team (2022) y C. Team (2023) también señalan la importancia de diferenciar el precio de una empresa o acción de la misma del valor intrínseco de esta. Pudiéndose resumir teniendo en cuenta lo señalado en esos textos y lo expuesto con anterioridad que el precio de una empresa o acción es lo que los compradores y vendedores están dispuestos a pagar por ella en un momento determinado, mientras que el valor intrínseco de una empresa o acción depende en gran medida de la metodología utilizada para valorar la empresas y los objetivos del evaluador.

Una vez contextualizado de manera general el entorno en el que se encuentran los precios de las acciones y explicado algunos de los factores que pueden afectar a los mismos se explica a continuación como se encuentran estos generalmente estructurados en las distintas fuentes de las que se pueden obtener. Generalmente los precios de las acciones se encuentran registrados de forma periódica (diariamente, semanalmente, mensualmente, anualmente, etc). registrándose para cada periodo el precio de apertura, el precio más alto, el más bajo, el de cierre, el volumen y el de cierre ajustado, ver **?@tbl-pre\_estruc**.

A partir de lo expuesto en Barone (2022), Chen (2022), Downey (2022), Hayes (2021) y Ganti (2020) se puede entender que:

* El precio de apertura es el primer precio al que se negocia un activo financiero en una sesión bursátil. Este precio puede ser diferente al precio de cierre de la sesión anterior, ya que puede haber cambios en la oferta y la demanda durante el periodo en que el mercado está cerrado. El precio de apertura suele indicar el tono o la tendencia del mercado para ese día.
* El precio más alto es el mayor precio al que se negocia un activo financiero en una sesión bursátil. Este precio refleja el máximo nivel de interés de los compradores por ese activo en ese día. El precio más alto puede ser un indicador de la fortaleza o la debilidad de un activo, así como de su volatilidad.
* El precio más bajo es el menor precio al que se negocia un activo financiero en una sesión bursátil. Este precio refleja el mínimo nivel de interés de los vendedores por ese activo en ese día. El precio más bajo puede ser un indicador de la presión o la resistencia de un activo, así como de su volatilidad.
* El precio de cierre es el último precio al que se negocia un activo financiero en una sesión bursátil. Este precio es el que se utiliza para calcular el valor de mercado de ese activo al final del día. El precio de cierre suele ser el más importante para los inversores, ya que resume el resultado de las operaciones del día y muestra la dirección del mercado.
* El volumen es la cantidad de unidades de un activo financiero que se negocian en una sesión bursátil. El volumen muestra el nivel de actividad o de liquidez de un mercado o de un activo. El volumen suele acompañar a los movimientos de los precios, ya que indica el grado de consenso o de divergencia entre los participantes del mercado.
* El precio de cierre ajustado es el precio de cierre de un activo financiero que se modifica para tener en cuenta eventos como dividendos, splits, fusiones o adquisiciones que afectan al valor del activo. El precio de cierre ajustado permite comparar el rendimiento histórico de un activo con mayor precisión y consistencia.

A partir de lo expuesto en Hayes (2021) y Ganti (2020) se entiende que la diferencia entre el precio de cierre y el precio de cierre ajustado es de gran importancia, ya que el primero puede dar una imagen distorsionada del rendimiento de una acción a lo largo del tiempo, mientras que el segundo refleja el valor real de la acción después de ajustar los factores que lo alteran.

Por ejemplo, la junta directiva de una empresa puede decidir dividir las acciones de la empresa 3 por 1. Por lo tanto, las acciones en circulación de la empresa aumentan en un múltiplo de tres, mientras que el precio de sus acciones se divide por tres. Supongamos que una acción cerró a $300 el día anterior a su división de acciones. En este caso, el precio de cierre se ajusta a $100 ($300 divididos por 3) por acción para mantener un estándar de comparación consistente. De manera similar, todos los demás precios de cierre anteriores para esa empresa se dividirían por tres para obtener los precios de cierre ajustados. Ganti (2020)

Debido a ello el precio de cierre ajustado es mejor para la aplicación de técnicas de análisis de series de tiempo, ya que permite comparar el comportamiento de una acción a lo largo del tiempo sin las distorsiones causadas por los eventos corporativos. Siendo la serie de tiempo más comunmente utilizada en los estudios de los análisis de los precios de mercado la conformado por las rentabilidades calculadas a partir del precio de cierre ajustado.

## 1.2 Redes neuronales artificiales en la previsión de las series de tiempo

Este epígrafe esta dividido en tres sub-epígrafes.En el primero se abordan los antecedentes del uso de redes neuronales artificiales para el trabajo con series de tiempo, más concretamente en la previsión. En el segundo y tercer sub-epígrafes se exponen el funcionamiento de dos de las estructuras de capas de RNA usadas en el presente trabajo, siendo estas las CNN y las LSTM.

### 1.2.1 Antecedentes del uso de redes neuronales artificiales en la previsión de series de tiempo

En este sub-epígrafe se abordan las redes neuronales artificiales (en lo adelante RNA) desde lo general a lo particular explicándose desde el entorno a las características básicas de las mismas. Además se exponen los antecedentes del uso de las mismas en la solución de problemas de previsión de series de tiempo.

En Chollet and Allaire (2018) se plantea que el entorno de las RNA, está conformado por la inteligencia artificial (en lo adelante IA), machine learning o aprendizaje automatizado (en lo adelante ML) y deep learning o aprendizaje profundo (en lo adelante DL), [Figure 1.6](#fig-DLenv). Por lo que es de vital importancia conocer los aspectos de estos campos que se encuentran íntimamente relacionados con las RNA y que se exponen brevemente a continuación.

“Hacer que una máquina se comporte de tal manera que si un humano lo hiciera se le llamaría inteligente” (McCarthy et al. (2006), p.11) es la primera definición que se le dio al problema de IA. Con el objetivo de dar solución a este problema surgieron las primeras IA, las llamadas IA simbólicas.

Como se explica en Haykin (1998), Banda (2014) y Chollet and Allaire (2018), estas primeras IA, involucraban reglas codificadas creadas por los programadores. Con el objetivo de lograr que estas reglas fueran aprendidas automáticamente por las máquinas al observar los datos surgió una nueva etapa dentro del desarrollo de las IA, la denominada ML. En esta nueva etapa se da pie al surgimiento de una nueva forma de programación, diferenciándose de la clásica, en que en esta, los programadores introducen los datos y las respuestas esperadas a los mismos, y las computadoras son capaces de generar las reglas, [Figure 1.7](#fig-MLprog).

Por lo que se entiende que los modelos de ML tratan de encontrar representaciones apropiadas para sus datos de entrada: transformaciones de los datos que hacen que sea más susceptible a la tarea en cuestión. En DL, que es un sub-campo específico de ML, estas representaciones de datos son modeladas a través de arquitecturas compuestas de capas sucesivas, las que son llamadas RNA Chollet and Allaire (2018).

Tras el estudio de lo expuesto en Haykin (1998), Larrañaga (2007), Banda (2014) y Chollet and Allaire (2018) sobre las RNA se puede afirmar que están inspiradas en el funcionamiento del cerebro humano, dichos textos confirman y concuerdan en que en una RNA se pueden diferenciar tres tipos de capas: de entrada, de salida y ocultas. Una capa de entrada está compuesta por neuronas que reciben los vectores de entradas. Una capa de salida se compone de neuronas que, durante el entrenamiento reciben los vectores de salidas y que luego generan la respuesta. Una capa oculta se encuentra conectada al entorno a través de las capas de entrada y salida, este tipo de capa oculta procesa la entrada recibida para obtener la salida correspondiente, [Figure 1.8](#fig-RNAstruct).

Una de las aplicaciones de las RNA es la previsión de series temporales. cuyo objetivo es predecir los valores futuros de las variables en función de sus observaciones pasadas. Como se expuso con anterioridad las series de tiempo financieras a menudo son no lineales, ruidosas, caóticas y no estacionarias, lo que las hace difíciles de modelar y pronosticar. Las RNA tienen la ventaja de poder capturar relaciones no lineales complejas y adaptarse a condiciones cambiantes sin requerir suposiciones previas sobre la distribución o estructura de datos.

La historia de las RNA en la previsión de series temporales financieras se remonta a finales de la década de 1980 y principios de la de 1990, cuando los investigadores comenzaron a explorar el potencial de las RNA como una alternativa a los métodos estadísticos tradicionales, como el modelo autorregresivo integrado de media móviles, más conocido como ARIMA (por sus siglas en inglés Autoregressive Integrated Moving Average) y los modelos autorregresivos generalizados con heterocedasticidad condicional, más conocido como GARCH (por sus siglas en inglés Generalized Autoregressive Conditional Heteroskedasticity). Se demostró que las RNA tienen varias ventajas sobre estos métodos, como la capacidad de capturar relaciones no lineales y dinámicas, manejar datos ruidosos e incompletos y adaptarse a las condiciones cambiantes del mercado (B. Eddy Patuwo & Michael Y. Hu (1998)).

Sin embargo, las RNA también enfrentan algunas limitaciones y desafíos en el pronóstico de series temporales financieras, como la dificultad de elegir una arquitectura de red adecuada, un algoritmo de entrenamiento, una función de activación y variables de entrada; el riesgo de sobreajuste y problemas de generalización; la falta de interpretabilidad y transparencia; y el alto costo computacional y tiempo (Tealab (2018)).

Para superar estas limitaciones y desafíos, los investigadores han propuesto varias mejoras y extensiones de RNA para el pronóstico de series temporales financieras en las últimas décadas. Algunos de los principales desarrollos incluyen:

* El uso de modelos híbridos que combinan RNA con otras técnicas, como lógica difusa, algoritmos genéticos, análisis de ondículas, máquinas de vectores de soporte y aprendizaje profundo para mejorar el rendimiento y la solidez de las RNA (Wong and Guo (2010)).
* El uso de redes neuronales recurrentes (en lo adelante RNR) o bidireccional, que son un tipo especial de RNA que pueden procesar datos secuenciales y capturar dependencias temporales. Se ha demostrado que las RNR superan a las redes neuronales unidireccionales en series temporales complejas y no lineales (Guresen, Kayakutlu, and Daim (2011)).
* El uso de modelos de RNA más complejas mediante la combinación de distintas capas, como son las redes neuronales convolucionales (en lo adelante, CNN), las long short-term memory (en lo adelante, LSTM), las gated recurrent units (en lo adelante, GRU) se han aplicado a la previsión de series temporales financieras con resultados prometedores (Sezer, Gudelek, and Ozbayoglu (2020)).

La historia de las RNA en el pronóstico de series temporales financieras muestra que las mismas han ido evolucionando y mejorando con el tiempo para hacer frente a la complejidad y la incertidumbre de los mercados financieros. Sin embargo, todavía persisten algunos de los desafíos y limitaciones señalados con anterioridad como como el sobreajuste, la generalización, la interpretabilidad, la robustez y el costo computacional.

### 1.2.2 Redes neuronales convolucionales

El modelo de RNA que se usó en este trabajo está compuesto por varias capas siendo las más importantes la capa Conv1D, un tipo especifico de CNN, y la capa LSTM, ambas mencionadas en el sub-epígrafe anterior cuando se listaron las estructuras de ANN que más se utilicen en la actualidad. Este sub-epígrafe se centra en la Capa Conv1D, por lo que a se exploran los conceptos fundamentales para comprender el funcionamiento de la misma, explicándose la convolución, las redes neuronales convolucionales y Conv1D y su uso para el análisis de series temporales. Se brinda una descripción general de la convolución y cómo se puede aplicar a los datos de series temporales. Luego, se analizan las CNN y su arquitectura, que les permite aprender características automáticamente a partir de datos de series temporales. Finalmente, se explica Conv1D, un tipo específico de capa de red neuronal convolucional que es particularmente eficaz para procesar datos de series temporales.

Como se expone en Siddiqui (2023) la convolución es una operación matemática que se usa comúnmente en el procesamiento de señales y el análisis de imágenes. Implica tomar dos funciones y producir una tercera función que representa cómo una de las funciones originales modifica a la otra. En el contexto de los datos de series temporales, la convolución se puede utilizar para extraer características de los datos aplicando un filtro a la serie temporal.

Además de extraer características de los datos de series temporales, la convolución también se puede utilizar para otras tareas, como la reducción de ruido, la detección de anomalías y la predicción. Por ejemplo, se puede entrenar una CNN para predecir valores futuros de una serie temporal aprendiendo los patrones subyacentes en los datos. En general, la convolución es una herramienta poderosa para analizar datos de series temporales y sus aplicaciones son numerosas Siddiqui (2023).

Las CNN fueron por primera vez introducidas en Lecun et al. (1998) son un tipo de modelo de aprendizaje profundo que se usa comúnmente para el análisis de imágenes. Sin embargo, como se ha mencionado con anterioridad también se pueden utilizar para el análisis de series temporales, ya que son muy adecuados para aprender características a partir de datos que tienen una estructura espacial o temporal.

La arquitectura de una CNN consta de una o más capas convolucionales, que aplican filtros a los datos de entrada para extraer características. Cada filtro es un conjunto de pesos que se aprenden durante el proceso de entrenamiento. Al deslizar el filtro sobre los datos de entrada, la capa convolucional calcula un producto escalar en cada posición, produciendo un nuevo mapa de características Lecun et al. (1998).

En un contexto de series de tiempo, una CNN puede aprender a extraer automáticamente características de los datos en diferentes escalas e intervalos de tiempo, lo que la convierte en una herramienta poderosa para el análisis de series de tiempo. Una ventaja clave de usar una CNN para el análisis de series de tiempo es que reduce la necesidad de ingeniería de características manual. En lugar de diseñar filtros a mano, CNN aprende a extraer automáticamente características de los datos, haciéndolo más flexible y adaptable a diferentes tipos de datos de series temporales.

En general, la arquitectura de una CNN le permite aprender características automáticamente a partir de datos de series temporales, lo que la convierte en una herramienta poderosa para el análisis de series temporales, siendo las Conv1D una de las estructuras de CNN más usadas para esta tarea.

Como se explica en Jing (2020) Conv1D es un tipo específico de capa de CNN que está diseñado para procesar datos unidimensionales, como datos de series temporales. Mientras que las CNN tradicionales están diseñadas para procesar datos bidimensionales, Conv1D está optimizado específicamente para datos unidimensionales, lo que lo hace más eficiente y eficaz para el análisis de series temporales.

La arquitectura de una capa Conv1D es similar a la de una CNN tradicional, pero con algunas diferencias clave. En lugar de usar filtros bidimensionales, Conv1D usa filtros unidimensionales, que se aplican a la serie temporal de entrada para extraer características. Las características que se extraen de la serie dependerán de las distintas configuraciones usadas para la configuración del filtro y la cantidad de filtros utilizados, siendo la fórmula para calcular la cantidad de característica que extrae cada filtro la siguiente [Equation 1.1](#eq-cnn-lout) (Jing (2020)):

Donde:

*Lout*: es la longitud de la salida del proceso de filtrado o la cantidad de características.

*Lin*: la longitud del vector de entrada, correspondiendo en el análisis de series de tiempo a la cantidad de observaciones que contienen las muestras de la serie de tiempo que se pasan al filtro.

*kernel\_size*: es el tamaño del filtro, lo que define cuantas observaciones del vector de entradas se pasan al filtro cada vez. [Figure 1.9](#fig-HJks) representa como el tamaño del filtro puede afectar la longitud del vector de salida.

*stride*: representa la cantidad de pasos u observaciones en las que se mueve la selección de observaciones que se pasa al filtro. [Figure 1.10](#fig-HJstride) representa como el parámetro stride puede afectar la longitud del vector de salida.

*dilation*: es la distancia de las observaciones que pasan al filtro. [Figure 1.11](#fig-HJdilation) representa como el parámetro dilation puede afectar la longitud del vector de salida.

*padding*: representa la cantidad de zeros que se añade a cada extremo del vector. [Figure 1.12](#fig-HJpadding) representa como el parámetro padding puede afectar la longitud del vector de salida.

En general, Conv1D es una herramienta poderosa para procesar datos de series temporales y sus ventajas incluyen la eficiencia computacional y la capacidad de capturar dependencias temporales en los datos. Sus casos de uso son numerosos y abarcan diferentes campos, lo que lo convierte en una herramienta valiosa para el análisis de series temporales.

### 1.2.3 Long short-term memory

En el presente subepigrafe se explica por qué las LSTM son una de las estructuras más usadas de RNA en la previsión de series de tiempo, partiendo de una breve explicación de las RNR y porque estas son de utilidad en la solución de problemas de previsión de series de tiempo, profundizando en el porque las LSTM se diferencian del resto de las RNN, y el funcionamiento de cada una de las capas que componen la estructura de una capa LSTM.

En Olah (2015) se explica que una RNN puede considerarse como copias múltiples de la misma red, [Figure 1.13](#fig-CORNRstruct), expone que este aspecto revela que las RNR están íntimamente relacionadas con secuencias y listas, lo que hace que este tipo de RNA sea el que se use naturalmente para el trabajo con series de tiempo.

Las RNR convencionales presentan un problema en lo relacionado con la capacidad de retener la información, como se explica en Olah (2015), las RNN estándar se desempeñan con gran capacidad solo si, la información relevante para la situación actual es reciente, es decir donde la brecha entre la información relevante y el lugar en que se necesita es pequeña, [Figure 1.14](#fig-CORInclose); expone además que a medida que crece la brecha, las RNN estándar son incapaces de acceder a la información relevante, [Figure 1.15](#fig-CORInaway).

Como se ha mencionado con anterioridad las LSTM son un tipo de RNR que puede aprender dependencias a largo plazo en datos secuenciales. Estas fueron propuestas en Hochreiter (1997) y ha sido ampliamente utilizado para diversas tareas como el modelado del lenguaje, el reconocimiento de voz, la traducción automática, la descripción de imágenes y la previsión de series de tiempo.

La idea principal de LSTM es introducir una celda de memoria que pueda almacenar y actualizar información durante largos pasos de tiempo. La celda de memoria está controlada por tres puertas: una puerta de entrada, una puerta de olvido y una puerta de salida. Estas puertas son redes neuronales que aprenden a regular el flujo de información dentro y fuera de la célula [Figure 1.16](#fig-CODrnrlstm).

La puerta de entrada decide qué cantidad de la nueva entrada agregar al estado de la celda. La puerta de olvido decide qué parte del estado de celda anterior mantener o borrar. La puerta de salida decide qué parte del estado de celda actual se va a enviar a la siguiente capa. Olah (2015) basado en lo expuesto en Hochreiter (1997), describe la operativa de las puertas en cuatro pasos:

1. Decidir qué información se olvida del estado de la celda a través de la puerta, forget gate layer . Esta puerta ve a , estado oculto del período de tiempo anterior, y , entrada del instante de tiempo actual, y genera un número entre 0 (deshacerse) y 1 (mantener) para cada número en el estado de la celda , [Figure 1.17](#fig-COLSTMstep1), [Equation 1.2](#eq-lstm-fstep).

1. Decidir qué nueva información se almacena en el estado de la celda. Para esto primero la puerta llamada input gate layer decide qué valores actualizar y luego, una capa tanh (tangente hiperbólica) crea un vector de nuevos valores candidatos () que podrían agregarse al estado, [Figure 1.18](#fig-COLSTMstep2), [Equation 1.3](#eq-lstm-sstepf) y [Equation 1.4](#eq-lstm-ssteps).

1. Se actualiza el estado de la celda anterior, en el nuevo estado de la celda . Se multiplica el estado anterior por , olvidando lo necesario, luego se agrega $i\_t\* \_t $. Estos son los nuevos valores candidatos, escalados según cuánto se necesita actualizar cada valor de estado, [Figure 1.19](#fig-COLSTMstep3), ecuación 1.09.

1. Se genera una salida basándose en el estado de celda. Ejecutándose primero una capa sigmoidea que decide qué partes del estado de la celda es la salida; luego el estado de la celda pasa a través de una función tanh (escalando los valores entre −1 y 1) y se multiplican por la salida de la puerta, output gate, [Figure 1.20](#fig-COLSTMstep4), [Equation 1.6](#eq-lstm-fstepf) y [Equation 1.7](#eq-lstm-fsteps).

Las LSTM pueden aprender a capturar dependencias a largo plazo ajustando los valores de la puerta a través de la propagación inversa. Por ejemplo, si una determinada entrada es relevante para una salida posterior, la puerta de entrada aprenderá a dejarla entrar, y la puerta olvidada aprenderá a conservarla en el estado de celda hasta que sea necesaria. Por el contrario, si una entrada es irrelevante u obsoleta, la puerta de entrada aprenderá a ignorarla, y la puerta olvidada aprenderá a borrarla del estado de la celda.

## 1.3 Composición de carteras

En este epígrafe, se expone el problema de encontrar la mejor composición de cartera posible y explicar de manera general la teoría detrás de los objetivos para solucionarlo. Además, se enumerarán las técnicas más utilizadas, diferenciando entre enfoques clásicos y enfoques inteligentes. Posteriormente, se explicará qué es la programación cuadrática y se mencionarán algunas técnicas dentro de esta disciplina de optimización matemática. Se mostrará cómo el problema de optimización de carteras se puede describir como un problema de programación cuadrática. Además, se proporcionará una breve explicación del Dual Active Set Method, una técnica ampliamente utilizada en esta disciplina y que será utilizada en los capítulos siguientes.

### 1.3.1 Problema y técnicas

En este sub-epígrafe se presenta el problema de hallar la mejor composición de cartera posible, explicandose de manera general la teoría detras de los objetivos que se deben seguir para darle solución al mismo. Además se listan las técnicas más utilizadas para darle solución a este problema, diferenciandolas entre enfoques clásicos y enfoques inteligentes.

Como se explica en Gunjan (2023) la optimización de cartera es el proceso de seleccionar la mejor combinación de activos para mantener en una cartera en función de objetivos predefinidos. Los objetivos pueden ser la maximización del rendimiento o la minimización del riesgo, o ambos. La optimización de la cartera implica encontrar las ponderaciones óptimas para cada activo de la cartera de manera que la cartera general cumpla con los objetivos deseados. Esto puede ser un problema desafiante debido a la gran cantidad de activos para elegir y las complejas relaciones entre ellos.

La optimización de la cartera es un proceso importante para los inversores, ya que les ayuda a minimizar el riesgo y maximizar el rendimiento de sus inversiones. Al seleccionar cuidadosamente los activos que mantendrán en su cartera, los inversores pueden lograr el nivel deseado de riesgo y rendimiento mientras diversifican sus inversiones para reducir el riesgo general. La optimización de la cartera es un mecanismo crucial que se utiliza para reducir el riesgo de la inversión.

Existen diversas técnicas que se pueden utilizar para resolver el problema de optimización de cartera. En Gunjan (2023) estas técnicas se encuentran clasificadas en en dos categorías: enfoques clásicos y enfoques inteligentes. A continuación se explica de manera general algunas de las técnicas pertenecientes a cada enfoque.

Enfoques clásicos:

* Media-varianza: Esta técnica, propuesta en Markowitz and Markowitz (1967), se basa en la idea de minimizar la varianza para una determinada rentabilidad esperada o maximizar la rentabilidad esperada para una determinada varianza. Es una técnica de programación cuadrática paramétrica (en lo adelante, PQP) que se puede utilizar para resolver problemas de optimización cuadrática que surgen en la optimización de carteras (Aijun Zhang & Chun-hung Li & Agus Sudjianto (2008)). El enfoque de la varianza media supone que los inversores tienen aversión al riesgo y prefieren carteras con una varianza más baja. La técnica consiste en construir una frontera de cartera que representa el conjunto de carteras que ofrecen el rendimiento esperado más alto para un nivel de riesgo dado. A continuación, se selecciona la cartera óptima de esta frontera en función de las preferencias de riesgo del inversor.
* Varianza con asimetría: esta técnica amplía el enfoque de media-varianza teniendo en cuenta la asimetría de la distribución. Fue propuesta en Samuelson (1970) y se puede utilizar cuando la función de distribución no es de naturaleza cuadrática. La asimetría mide la asimetría de una distribución y puede proporcionar información adicional sobre los riesgos y rendimientos potenciales de una cartera. Al incorporar la asimetría en el proceso de optimización de la cartera, los inversores pueden comprender mejor los posibles riesgos a la baja y tomar decisiones más informadas.
* Valor en riesgo (VaR): este enfoque estadístico mide la pérdida potencial de valor de una cartera durante un período definido para un intervalo de confianza dado. Fue introducido en la primera edición de Jorion (2007) en 1997 y requiere la determinación de tres parámetros: período de tiempo, nivel de confianza y unidad de valor en riesgo. El VaR proporciona una medida de la pérdida potencial máxima que podría ocurrir con una probabilidad dada en un horizonte de tiempo específico. Las instituciones financieras lo utilizan comúnmente para administrar su exposición al riesgo y cumplir con los requisitos reglamentarios.
* Valor en riesgo condicional (CVaR): este enfoque amplía el VaR teniendo en cuenta la pérdida esperada que excede el VaR. Fue introducido en Rockafellar and Uryasev (2002) y puede manejar pérdidas extremas mediante el uso de pesos dinámicos derivados de datos históricos. CVaR proporciona una medida de la pérdida esperada que podría ocurrir más allá del umbral de VaR. También se conoce como Expected Shortfall (ES) o Tail Value-at-Risk (TVaR) y se considera una medida de riesgo más coherente que el VaR.
* Desviación media-absoluta (MAD): esta técnica se puede emplear para problemas de selección de carteras de gran escala y muy diversificados. Fue introducido en Konno and Yamazaki (1991) y penaliza tanto las desviaciones positivas como las negativas. MAD proporciona una medida de la desviación absoluta promedio de los rendimientos de la cartera de su valor medio. Se considera más sólida que las medidas basadas en la varianza, ya que es menos sensible a los valores atípicos.
* Minimax: Esta técnica utiliza la rentabilidad mínima como medida de riesgo. Fue introducido en Cai et al. (2004) y tiene ciertas ventajas cuando los rendimientos no se distribuyen normalmente. Minimax proporciona una medida del peor de los casos para una cartera al minimizar la pérdida potencial máxima que podría ocurrir. Puede ser útil para los inversores que están particularmente preocupados por los riesgos a la baja.

Enfoques inteligentes:

* Redes bayesianas: estos modelos gráficos probabilísticos se pueden utilizar para modelar el riesgo y la rentabilidad. Fueron presentados en Shenoy and Shenoy (2000) y se pueden utilizar para visualizar la relación entre diferentes variables en un modelo. Las redes bayesianas proporcionan una forma de representar dependencias complejas entre variables utilizando gráficos acíclicos dirigidos (DAG). Se pueden usar para modelar relaciones inciertas entre variables y para hacer predicciones probabilísticas sobre eventos futuros. En el contexto de la gestión de carteras, las redes bayesianas se pueden utilizar para modelar las relaciones entre diferentes activos y hacer predicciones sobre sus rendimientos futuros en función de datos históricos y otra información relevante.
* Regresión de vectores de soporte (SVR): esta técnica de aprendizaje automático se puede utilizar para determinar la cantidad a comprar y vender. Fue introducido por Drucker et al. (1996) y tiene ciertas ventajas sobre las técnicas basadas en estadísticas, como su capacidad para aprender de datos históricos. SVR implica construir un hiperplano que separa puntos de datos con diferentes etiquetas mientras maximiza el margen entre ellos. Puede usarse para tareas de regresión donde el objetivo es predecir valores continuos en lugar de etiquetas discretas. En el contexto de la gestión de carteras, SVR se puede utilizar para predecir precios de activos futuros en función de datos históricos y otra información relevante.
* Redes neuronales artificiales: como se explico con anterioridad estos modelos computacionales se pueden utilizar para resolver problemas computacionales y de aprendizaje complejos. En el contexto de la gestión de carteras, las redes neuronales se pueden utilizar para predecir futuros precios o rendimientos de activos en función de datos históricos y otra información relevante, que es para lo que se usan en el presente trabajo.
* Aprendizaje por refuerzo: este tipo de aprendizaje automático involucra a un agente o modelo que interactúa con su entorno para aprender de sus acciones. Fue presentado en Sutton and Barto (2018) y funciona para maximizar la recompensa al agente. El aprendizaje por refuerzo implica aprender a través de interacciones de prueba y error con un entorno. El agente realiza acciones en función de su estado actual y recibe recompensas o sanciones en función de los resultados de esas acciones. Con el tiempo, el agente aprende a realizar acciones que maximicen su recompensa acumulada. En el contexto de la gestión de carteras, el aprendizaje por refuerzo se puede utilizar para desarrollar estrategias comerciales que maximicen los rendimientos mientras se gestiona el riesgo.

### 1.3.2 Programación cuadrática

En este sub-epígrade se explicara que es la programación cuadrática. Cuáles son algunas de las técnicas que existen dentro de esta disciplina de la optimización matemática. Se expone además como el problema de optimización de carteras se puede describir como un problema de programación cuadratica y se expondra de manera breve como funciona una de las técnicas más usadas en esta disciplina, concretamente la denominada Dual Active Set Method, la cuál es usada en los capítulos posteriores.

La programación cuadrática se puede elegir entre las técnicas enumeradas en el sub-epígrafe anterior por varias razones. En primer lugar, es una técnica bien establecida que se ha utilizado ampliamente en la optimización de carteras. Puede manejar problemas de optimización complejos con múltiples restricciones y puede proporcionar una forma eficiente y efectiva de resolver el problema de optimización de cartera. Esto lo convierte en una herramienta útil para los inversores que buscan minimizar el riesgo mientras logran el nivel de rendimiento deseado. Finalmente, la programación cuadrática tiene una sólida base teórica y ha sido ampliamente estudiada en la literatura. Esto la convierte en una técnica confiable y bien entendida que se puede utilizar con confianza en la optimización de la cartera.

Existen diversas tecnicas de programación cuadrática, entre las más utilizadas se ecuentran:

* Interior Point: Este es un método de programación lineal o no lineal que logra la optimización al pasar por el centro del sólido definido por el problema en lugar de alrededor de su superficie. Un algoritmo de programación lineal de tiempo polinómico utilizando un método de punto interior fue encontrado por Karmarkar (1984).
* Active Set: Este es un algoritmo utilizado para identificar las restricciones activas en un conjunto de restricciones de desigualdad. Las restricciones activas se expresan entonces como restricciones de igualdad, transformando así un problema restringido por la desigualdad en un subproblema más simple restringido por la igualdad. El método de conjunto activo fue introducido por primera vez en un artículo de Beale (1959) y desarrollado por Fletcher (1971) y Bunch and Kaufman (1977).
* Dual Active Set: El método, como se expone en Goldfarb and Idnani (1982) y Goldfarb and Idnani (1983), es un algoritmo dual eficiente y numéricamente estable para la programación cuadrática definida positiva que aprovecha el hecho de que el mínimo sin restricciones de la función objetivo se puede usar como punto de partida.
* Augmented Lagrangian: Fue introducido independientemente en Magnus R. Hestenes (1969) y Powell (1969). Se utiliza para resolver problemas de optimización restringidos agregando un término de penalización a la función objetivo que penaliza cualquier violación de las restricciones. El término de penalización suele ser un múltiplo de una medida de infracción de restricción, como la suma de infracciones de restricción al cuadrado.
* Conjugate Gradient: Este es un método iterativo para resolver sistemas de ecuaciones lineales con una matriz definida positiva simétrica. También se puede utilizar para resolver problemas de optimización sin restricciones al encontrar el mínimo de una función cuadrática. El método genera una secuencia de direcciones de búsqueda que se conjugan con respecto a la matriz que define el sistema de ecuaciones o función cuadrática. El método de gradiente conjugado fue introducido originalmente en un artículo de Magnus R. Hestenes and Stiefel (1952).
* Gradient Projection: El método de proyección de gradiente fue introducido en J. B. Rosen (1960) y J. Rosen (1961). Este es un método iterativo para resolver problemas de optimización restringidos proyectando el gradiente en la región factible en cada iteración. El gradiente proyectado se utiliza entonces como dirección de búsqueda, y se realiza una búsqueda de línea a lo largo de esta dirección para encontrar una nueva iteración que satisfaga las restricciones y reduzca la función objetivo.

De las técnicas señaladas con anterioridad se seleccionó el algoritmo Dual Active Set Method (en lo adelante, DASM) que como se mencionó con anterioridad fue introducido en Goldfarb and Idnani (1982) y Goldfarb and Idnani (1983), es un algoritmo de optimización para resolver problemas de programación cuadrática. El algoritmo predice el conjunto activo de restricciones que se satisfacen con igualdad en la solución del problema. Calcula una secuencia de soluciones óptimas de problemas QP que involucran algunas de las restricciones del problema original, denominada secuencia de puntos factibles duales.

A continuación se presenta un ejemplo general de cómo podría funcionar el algoritmo DASM usando valores hipotéticos para un problema de optimización de cartera con 2 activos, el ejemplo fue construido a partir de lo expuesto en Goswami, Mondal, and Paruya (2012) y Walker (2014):

Bajo la suposición de que se trata de encontrar la mejor composición de una cartera en la que, por simplicidad, tenemos 2 activos, Se plantearia el problema cuadrático de la siguiente manera [Equation 1.8](#eq-qpop):

Suponiendo que los cuales tienen unos rendimientos mensuales medios y matriz de covarianza . Se pueden construir los vectores y matrices necesarios para el algoritmo DASM de la siguiente manera:

* El vector de rentabilidad media mensual sería .
* La matriz de covarianza C se usaría como la matriz D en DASM.
* La restricción se puede escribir en forma de matriz como . Esta sería la primera fila de la matriz en DASM.
* El requisito de rentabilidad mínima puede escribirse en forma de matriz como . Esta sería otra fila de la matriz en DASM.
* Las restricciones se pueden escribir en forma de matriz como y para límites inferiores y y para límites superiores.
* La matriz luciría así:

El vector correspondiente sería . Luego podemos usar el algoritmo DASM para resolver este problema de programación cuadrática y determinar la asignación óptima de activos en nuestra cartera.

Paso 0: Encuentre el mínimo sin restricciones resolviendo el problema de programación cuadrática sin restricciones. Establecer el número de elementos del conjunto activo A (conjunto vacío) a cero.

Paso 1: Elija una restricción violada, si la hay. En este caso, supongamos que se viola la restricción .

Paso 2: Calcule las direcciones del paso primario y dual y la longitud del paso . Supongamos que .

Paso 3: Da un paso y actualiza el conjunto activo A y la solución () par (x,A). Como , agregamos la p-ésima restricción (en este caso ) a y actualizamos y en [Equation 1.9](#eq-consup).

Donde es la pseudo-inversa o la inversa generalizada Moore-Penrose de , la matriz de los vectores normales de las restricciones en el conjunto activo . es el operador hessiano inverso reducido de .

Se repetin estos pasos de manera iterativa hasta que se satisfagan todas las restricciones y se haya determinado la asignación óptima de activos.

# Apéndices Capítulo 1

Table 1.1: Clasificaciones de procesos estocásticos

| Variable dependiente | Subíndice Discreto | Subíndice Continuo |
| --- | --- | --- |
| Discreta | Proceso de estado discreto y tiempo discreto | Proceso de estado discreto y tiempo continuo |
| Continua | Proceso de estado continuo y tiempo discreto | Proceso de estado continuo y tiempo continuo |

Tomada de: Ruiz (2011).

| Date | Open | High | Low | Close | Volume | Adjusted |
| --- | --- | --- | --- | --- | --- | --- |
| 2001-05-24 | 3.600 | 3.620 | 3.510 | 3.608 | 216270100 | -0.1292753 |
| 2001-05-25 | 3.600 | 3.676 | 3.580 | 3.602 | 50448300 | -0.1290603 |
| 2001-05-28 | 3.560 | 3.604 | 3.544 | 3.580 | 26118945 | -0.1282721 |
| 2001-05-29 | 3.562 | 3.626 | 3.562 | 3.614 | 26910070 | -0.1294903 |
| 2001-05-30 | 3.606 | 3.648 | 3.602 | 3.620 | 48229995 | -0.1297053 |
| 2001-05-31 | 3.620 | 3.676 | 3.610 | 3.670 | 24806710 | -0.1314968 |

Elaboración propia: Mediante el uso de RStudio con la base de datos histórico de “INDITEX”, obtenida de https://finance.yahoo.com/, en el periodo comprendido entre el 24-05-2001 al 31-05-2001.

|  |
| --- |
| Figure 1.1: Tendencia alcista y heterocedástica |

Elaboración propia: Mediante el uso de RStudio con la base de datos histórico del IBEX, obtenida de https://finance.yahoo.com/, en el periodo comprendido entre el 01-1995 al 01-1997.

|  |
| --- |
| Figure 1.2: Tendencia bajista y heterocedástica |

Elaboración propia: Mediante el uso de RStudio con la base de datos histórico del IBEX, obtenida de https://finance.yahoo.com/, en el periodo comprendido entre el 01-2000 al 01-2003.

|  |
| --- |
| Figure 1.3: Sin tendencia, homocedástica y estacionaria |

Elaboración propia: Mediante el uso de RStudio con la base de datos histórico del IBEX obtenida de https://finance.yahoo.com/, en el periodo comprendido entre el 01-2000 al 01-2003, usando las rentabilidades calculadas a partir del precio de cierre.

|  |
| --- |
| Figure 1.4: Descomposición: estacionalidad y error |

Elaboración propia: Mediante el uso de RStudio con la base de datos histórico del IBEX obtenida de https://finance.yahoo.com/, descomponiendo la serie de tiempo conformada por las observaciones que abarcan el periodo del 01-2000 al 01-2023.

|  |
| --- |
| Figure 1.5: Correlograma |

Elaboración propia: Mediante el uso de RStudio.

|  |
| --- |
| Relación entre IA-ML-DL  Figure 1.6: DLenv |

Tomada de: Deep learning with R de Chollet and Allaire (2018).

|  |
| --- |
| Programación clásica y machine learning  Figure 1.7: MLprog |

Tomada de: Deep learning with R de Chollet and Allaire (2018).

|  |
| --- |
| Estructura básica de una red neuronal artificial  Figure 1.8: RNAstruct |

Tomada de: Tema 14: redes neuronales de Larrañaga (2007).

|  |
| --- |
| Como kernel_size afecta el vector de salida  Figure 1.9: Como kernel\_size afecta el vector de salida |

Elaboración propia: Elaborada a partir de Jing (2020). Muestra como el tamaño del vector de salida cambia según el tamaño de filtro que se usa.

|  |
| --- |
| Como stride afecta el vector de salida  Figure 1.10: Como stride afecta el vector de salida |

Elaboración propia: Elaborada a partir de Jing (2020). Muestra como el parámetro stride afecta el tamaño del vector de salida.

|  |
| --- |
| Como dilation afecta el vector de salida  Figure 1.11: Como dilation afecta el vector de salida |

Elaboración propia: Elaborada a partir de Jing (2020). Muestra como el parámetro dilation afecta el tamaño del vector de salida.

|  |
| --- |
| Como padding afecta el vector de salida  Figure 1.12: Como padding afecta el vector de salida |

Elaboración propia: Elaborada a partir de Jing (2020). Muestra como el parámetro padding afecta el tamaño del vector de salida.

|  |
| --- |
| Despliegue del bucle de una red neuronal recurrente estándar  Figure 1.13: Despliegue del bucle de una red neuronal recurrente estándar |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| Información relevante cercana  Figure 1.14: Información relevante cercana |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| Información relevante lejana  Figure 1.15: Información relevante lejana |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| Diferencia entre los módulos de repetición  Figure 1.16: Diferencia entre los módulos de repetición |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| LSTM funcionalidad: Representación del paso 1  Figure 1.17: LSTM funcionalidad: Representación del paso 1 |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| LSTM funcionalidad: Representación del paso 2  Figure 1.18: LSTM funcionalidad: Representación del paso 2 |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| LSTM funcionalidad: Representación del paso 3  Figure 1.19: LSTM funcionalidad: Representación del paso 3 |

Tomadas de: Understanding LSTM networks, Olah (2015).

|  |
| --- |
| LSTM funcionalidad: Representación del paso 4  Figure 1.20: LSTM funcionalidad: Representación del paso 4 |

Tomadas de: Understanding LSTM networks, Olah (2015).

# 2. Capítulo 2:

# Apéndices Capítulo 2

# References

Aijun Zhang & Chun-hung Li & Agus Sudjianto, Zhi-li Wu &. 2008. “Trace Solution Paths for SVMs via Parametric Quadratic Programming.” Researchgate. 2008. <https://www.researchgate.net/publication/228577955_Trace_solution_paths_for_SVMs_via_parametric_quadratic_programming>.

Anderson, D. R., D. J. Sweeney, T. A. Williams, D. J. Camm, and J. J Cochran. 2017. *Statistics for Business & Economics*. Boston: Cengage Learning.

B. Eddy Patuwo & Michael Y. Hu, Guoqiang Zhang &. 1998. “Forecasting with Artificial Neural Networks:: The State of the Art.” *International Journal of Forecasting* 14 (1): 35–62. https://doi.org/<https://doi.org/10.1016/S0169-2070(97)00044-7>.

Banda, Hugo. 2014. *Inteligencia Artificial: Principios y Aplicaciones*. Quito, Ecuador: Escuela Politécnica Nacional.

Barone, A. 2022. “Opening Price: Definition, Example, Trading Strategies.” 2022. <https://www.investopedia.com/terms/o/openingprice.asp>.

Beale, EML. 1959. “On Quadratic Proramming.” *Naval Research Logistics Quarterly* 6 (3): 227–43.

BME. n.d. “¿Qué Es BME?” Accessed April 24, 2023. <https://www.bolsasymercados.es/esp/Sobre-BME/Que-es>.

Bunch, James R, and Linda Kaufman. 1977. “Some Stable Methods for Calculating Inertia and Solving Symmetric Linear Systems.” *Mathematics of Computation* 31 (137): 163–79.

Cai, Xiaoqiang, Kok Lay Teo, XQ Yang, and Xun Yu Zhou. 2004. “Minimax Portfolio Optimization: Empirical Numerical Study.” *Journal of the Operational Research Society* 55 (1): 65–72.

Castillo, R. A., and R. Varela. 2010. *ECONOMETRÍA PRÁCTICA: Fundamentos de Series de Tiempo*. México: Universidad Autónoma de Baja California.

Chen, J. 2022. “Today’s High.” 2022. <https://www.investopedia.com/terms/t/todayshigh.asp>.

Chirinos, S. 2018. “Series Cronológicas.” <https://www.slideshare.net/SuedimarChirinos/series-cronologicas-119058959>. 2018.

Chollet, F., and J. J. Allaire. 2018. *Deep Learning with r*. Manning Publications. <https://books.google.es/books?id=xnIRtAEACAAJ>.

CNMV. n.d.a. “Glosario Financiero: Acción.” Accessed April 24, 2023. <https://cnmv.es/Portal/Inversor/Glosario.aspx?id=0&letra=A&idlng=1>.

———. n.d.b. “Glosario Financiero: Bolsa de Valores.” Accessed April 24, 2023. <https://cnmv.es/Portal/Inversor/Glosario.aspx?id=0&letra=B&idlng=1>.

———. n.d.c. “Glosario Financiero: Servicio de Interconexión Bursátil Español, SIBE.” Accessed April 24, 2023. <https://cnmv.es/Portal/Inversor/Glosario.aspx?id=0&letra=S&idlng=1>.

Dodge, Y. 2008. “Time Series.” In *The Concise Encyclopedia of Statistics*, 536–39. New York, NY: Springer New York. <https://doi.org/10.1007/978-0-387-32833-1_401>.

Downey, L. 2022. “Today’s Low.” 2022. <https://www.investopedia.com/terms/t/todayslow.asp>.

Drucker, Harris, Christopher Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. 1996. “Linear Support Vector Regression Machines.” *Advances in Neural Information Processing Systems* 9 (9): 155–61.

Espallargas, S. D., and M. V. Solís. 2012. *Econometría y Series Temporales: Aplicaciones*. La Habana: Editorial Félix Varela.

Fletcher, Roger. 1971. “A General Quadratic Programming Algorithm.” *IMA Journal of Applied Mathematics* 7 (1): 76–91.

Ganti, A. 2020. “Adjusted Closing Price.” 2020. <https://www.investopedia.com/terms/a/adjusted_closing_price.asp>.

Goldfarb, Donald, and Ashok U. Idnani. 1982. “Dual and Primal-Dual Methods for Solving Strictly Convex Quadratic Programs.” In *Numerical Analysis*, edited by J. P. Hennart, 226–39. Berlin, Heidelberg: Springer Berlin Heidelberg.

———. 1983. “A Numerically Stable Dual Method for Solving Strictly Convex Quadratic Programs.” *Mathematical Programming* 27: 1–33.

Goswami, Nababithi, Supriyo K. Mondal, and Swapan Paruya. 2012. “A Comparative Study of Dual Active-Set and Primal-Dual Interior-Point Method.” *IFAC Proceedings Volumes* 45 (15): 620–25. https://doi.org/<https://doi.org/10.3182/20120710-4-SG-2026.00029>.

Gunjan, Siddhartha, Abhishek & Bhattacharyya. 2023. “A brief review of portfolio optimization techniques.” *Artificial Intelligence Review* 56 (5): 3847–86. <https://doi.org/10.1007/s10462-022-10273-7>.

Guresen, Erkam, Gulgun Kayakutlu, and Tugrul U. Daim. 2011. “Using Artificial Neural Network Models in Stock Market Index Prediction.” *Expert Systems with Applications* 38 (8): 10389–97. https://doi.org/<https://doi.org/10.1016/j.eswa.2011.02.068>.

Hayes, A. 2021. “What Is Closing Price? Definition, How It’s Used, and Example.” 2021. <https://www.investopedia.com/terms/c/closingprice.asp>.

Haykin, Simon. 1998. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR.

Hestenes, Magnus R. 1969. “Multiplier and Gradient Methods.” *Journal of Optimization Theory and Applications* 4 (5): 303–20.

Hestenes, Magnus R., and Eduard Stiefel. 1952. “Methods of Conjugate Gradients for Solving Linear Systems.” *Journal of Research of the National Bureau of Standards* 49: 409–35.

Hochreiter, Jürgen, Sepp & Schmidhuber. 1997. “Long Short-Term Memory.” *Neural Computation* 9 (8): 1735–80. <https://doi.org/10.1162/neco.1997.9.8.1735>.

IBM. 2021. “Characteristics of Time Series.” <https://www.ibm.com/docs/en/spss-modeler/saas?topic=data-characteristics-time-series>. 2021.

Jing, Hong. 2020. “How Convolutional Layers Work in Deep Learning Neural Networks?” Jingles, Github Blog. 2020. <https://jinglescode.github.io/2020/11/01/how-convolutional-layers-work-deep-learning-neural-networks/>.

Jorion, Philippe. 2007. *Value at Risk: The New Benchmark for Managing Financial Risk*. The McGraw-Hill Companies, Inc.

Karmarkar, Narendra. 1984. “A New Polynomial-Time Algorithm for Linear Programming.” In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, 302–11.

Kocenda, E., and A. Cerný. 2017. *Elements of Time Series Econometrics: An Applied Approach*. Prague: Karolinum Press.

Konno, Hiroshi, and Hiroaki Yamazaki. 1991. “Mean-Absolute Deviation Portfolio Optimization Model and Its Applications to Tokyo Stock Market.” *Management Science* 37 (5): 519–31.

Larrañaga, Iñaki & Moujahid, Pedro & Inza. 2007. “Tema 14. Redes Neuronales.” Departamento de Ciencias de la Computaci´on e Inteligencia Artificial, Universidad del Pa´ıs Vasco–Euskal Herriko Unibertsitatea. 2007. <http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t14-neuronales.pdf>.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. “Gradient-Based Learning Applied to Document Recognition.” *Proceedings of the IEEE* 86 (11): 2278–2324. <https://doi.org/10.1109/5.726791>.

Markowitz, Harry M, and Harry M Markowitz. 1967. *Portfolio Selection: Efficient Diversification of Investments*. J. Wiley.

McCarthy, John, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. 2006. “A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955.” *AI Magazine* 27 (4): 12. <https://doi.org/10.1609/aimag.v27i4.1904>.

Mitchell, C. 2020. “Market Price: Definition, Meaning, How to Determine, and Example.” 2020. <https://www.investopedia.com/terms/m/market-price.asp>.

Olah, Christopher. 2015. “Understanding LSTM Networks.” Colah’s blog. 2015. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.

Pinset, W. 2021. “Understanding Stock Prices and Values.” 2021. <https://www.investopedia.com/articles/stocks/08/stock-prices-fool.asp>.

Powell, Michael JD. 1969. “A Method for Nonlinear Constraints in Minimization Problems.” *Optimization*, 283–98.

Rockafellar, R Tyrrell, and Stanislav Uryasev. 2002. “Conditional Value-at-Risk for General Loss Distributions.” *Journal of Banking & Finance* 26 (7): 1443–71.

Rosen, JB. 1961. “The Gradient Projection Method for Nonlinear Programming. Part II. Nonlinear Constraints.” *Journal of the Society for Industrial and Applied Mathematics* 9 (4): 514–32.

Rosen, Jo Bo. 1960. “The Gradient Projection Method for Nonlinear Programming. Part i. Linear Constraints.” *Journal of the Society for Industrial and Applied Mathematics* 8 (1): 181–217.

Ruiz, M. C. 2011. “Tema 5: Procesos Estocásticos.” <http://www.dmae.upct.es/~mcruiz/Telem06/Teoria/apuntes_procesos.pdf>; Departamento de Matemática y Estadística. Universidad Politécnica de Cartagena. 2011.

Samuelson, Paul A. 1970. “The Fundamental Approximation Theorem of Portfolio Analysis in Terms of Means, Variances and Higher Moments.” *The Review of Economic Studies* 37 (4): 537–42.

Sezer, Omer Berat, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. 2020. “Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005–2019.” *Applied Soft Computing* 90: 106181. https://doi.org/<https://doi.org/10.1016/j.asoc.2020.106181>.

Shenoy, Catherine, and Prakash P Shenoy. 2000. “Bayesian Network Models of Portfolio Risk and Return.” In. The MIT Press.

Siddiqui, J. Rafid. 2023. “Why Convolve? Understanding Convolution and Feature Extraction in Deep Networks.” Medium, Towards Data Science. 2023. <https://towardsdatascience.com/why-convolve-understanding-convolution-and-feature-extraction-in-deep-networks-ee45d1fdd17c>.

Sutton, Richard S, and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction*. MIT press.

Tealab, Ahmed. 2018. “Time Series Forecasting Using Artificial Neural Networks Methodologies: A Systematic Review.” *Future Computing and Informatics Journal* 3 (2): 334–40. https://doi.org/<https://doi.org/10.1016/j.fcij.2018.10.003>.

Team, CFI. 2023. “What Is Stock Price?” 2023. <https://corporatefinanceinstitute.com/resources/capital-markets/stock-price/>.

Team, The Investopedia. 2022. “Intrinsic Value Defined and How It’s Determined in Investing and Business.” 2022. <https://www.investopedia.com/terms/i/intrinsicvalue.asp>.

Villagarcía, T. 2006. “Series Temporales.” <https://halweb.uc3m.es/fjnm/estind/doc_grupo1/archivos/Apuntes%20de%20series.pdf>. 2006.

Villavicencio, J. 2010. “Introducción a Las Series de Tiempo.” <http://www.estadisticas.gobierno.pr/iepr/LinkClick.aspx>; Instituto de estadística de Puerto Rico. 2010.

Walker, Ryan. 2014. “Solving Quadratic Progams with r’s Quadprog Package.” rwalk. 2014. <https://rwalk.xyz/solving-quadratic-progams-with-rs-quadprog-package/>.

Wong, W. K., and Z. X. Guo. 2010. “A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm.” *International Journal of Production Economics* 128 (2): 614–24. <https://ideas.repec.org/a/eee/proeco/v128y2010i2p614-624.html>.