Trabajo Fin de Grado
Grado en Ingeniería de Tecnologías Industriales

# Implementación de un algoritmo de selección de características para predicción del precio de acciones bursátiles.

Autor: Fernando Losada Gata

Tutor: Daniel Rodríguez Ramírez

**Dpto. Ingeniería de Sistemas y Automática**
**Escuela Técnica Superior de Ingeniería**
**Universidad de Sevilla**

Sevilla, 2020

Proyecto Fin de Carrera
Ingeniería de Tecnologías Industriales

# Implementación de un algoritmo de selección de características para predicción del precio de acciones bursátiles.

Autor:
Fernando Losada Gata

Tutor:
Daniel Rodríguez Ramírez
Profesor titular

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2020

Proyecto Fin de Carrera: Implementación de un algoritmo de selección de características para predicción del precio de acciones bursátiles.

Autor: Fernando Losada Gata

Tutor: Daniel Rodríguez Ramírez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2020

El Secretario del Tribunal

*A mi familia*

# Agradecimientos

*Quiero agradecer en primer lugar a mis abuelos Isabel y Manolo por el cariño que me han dado siempre. A mis padres por además haberme brindado todo lo necesario para mi formación y crecimiento personal. Al tutor del proyecto Daniel Rodríguez Ramirez por el tiempo y la paciencia empleados, y la oportunidad de desempeñar un trabajo muy interesante y llevadero.*

# Resumen

El objetivo de este proyecto es el desarrollo de un algoritmo combinacional que hace uso de ciertos indicadores técnicos para predecir el precio de cierre de un índice bursátil.

Los indicadores técnicos en muchas ocasiones realizan predicciones contradictorias entre sí, al analizar la evolución del percio de un determinado índice. En este trabajo se pretende encontrar la combinación ideal de estos indicadores disponibles, con el afán de minimizar el error producido en esta predicción.

El algoritmo se implementará con la herramienta MATLAB, y se aplicará a dos casos de estudio diferentes, con datos del índice bursátil Dow Jones y de la acción Ping An Bank, respectivamente. Se realizarán pruebas de su funcionamiento y se analizarán los resultados obtenidos.

# Abstract

The objective of this project is the development of a combinational algorithm that predicts the closing price of a stock index by using certain technical indicators.

Technical indicators often make contradictory predictions when analyzing the evolution in the value of an specific index. The aim of this work is to find the ideal combination of these available indicators, in order to minimize the error produced in this prediction.

The algorithm will be implemented with MATLAB tool, and will be applied to two different study cases, with data of the Dow Jones stock index and the stock Ping An Bank, respectively. Tests of its operation will be carried out and the results obtained will be analyzed.

# Índice

# Índice de Tablas

# Índice de Figuras

# 1 INTRODUCCIÓN

Desde el nacimiento de la primera bolsa oficial de valores en Amsterdam , allá por el año 1602, hasta nuestros días, el mercado de valores y la búsqueda del perfeccionamiento en sus pronósticos ha sido un asunto que atañe a un gran sector de la población. Tanto empresas e inversionistas individuales como personal académico, acogen el análisis bursátil como medio imprescindible para realizar inversiones provechosas y con motivación didáctica, respectivamente.

De un modo u otro, el objetivo principal del estudio de este campo es la búsqueda de la rentabilidad en las inversiones. Se pretende para este propósito analizar las fluctuaciones de los valores del mercado, y hallar ciertos patrones que permitan capacitar al individuo o sistema para regirse por un método que evite en la medida de lo posible la aleatoriedad de estas fluctuaciones.

Con esta motivación nace el análisis bursátil. El análisis bursátil es el estudio del comportamiento de los mercados financieros y de los valores que lo constituyen. Este análisis supone una ayuda a la hora de tomar decisiones de inversión en momentos de incertidumbre. Aunque existen numerosos métodos de análisis, se puede clasificar principalmente en dos herramientas diferenciadas: el análisis fundamental o análisis financiero, y el análisis técnico. Este trabajo se centra en conceptos integrados en el análisis técnico.

## 1.1 Análisis técnico

El análisis técnico tuvo sus orígenes en EEUU a finales del siglo XIX con la Teoría de Dow, creada por Charles Henry Dow. Dicha teoría adquirió gran relevancia con Ralph Nelson Elliot dentro de los mercados accionarios con su Teoría de las Ondas de Elliot, y posteriormente se extendió al mercado de futuros.

Esta categoría de análisis bursátil considera que todos los factores que influyen en el comportamiento del mercado, como pueden ser las expectativas de una determinada compañía, su entorno socio-político o la componente irracional de los inversores (miedos, ansia, esperanza), están intrínsecamente recogidas en los “movimientos del mercado”. Dicha expresión corresponde a la fluctuacón de tres variables: el precio, el volumen y el interés abierto. Estas variables suponen la fuente básica de información para los analistas técnicos, a partir de las cuales desarrollarán sus predicciones.

### 1.1.1 Principios del análisis técnico

Para comprender la filosofía del análisis técnico, es preciso enumerar los tres principios fundamentales en los que se basa:

1. *El precio lo descuenta todo*. O mejor dicho, "los movimientos de mercado lo descuentan todo". Este primer principio, que puede ser considerado su piedra angular, reincide en lo recién explicado. Todos los factores directos o indirectos están reflejados en el precio, por tanto, basta con analizar la acción del mismo para tener todos los factores en consideración. La reflexión se basa en las leyes económicas fundamentales de oferta y demanda. Si la demanda de un producto crece respecto a la oferta, el precio aumenta. En cambio, si la oferta crece respecto a la demanda, el precio debería disminuir. Al considerarlo desde la perspectiva opuesta, si el precio ha aumentado, se puede intuir que la demanda es superior a la oferta, y por tanto los fundamentos son alcistas. De lo contrario, los sentimientos deberían ser bajistas.
2. *El precio se mueve en tendencias*. Según este principio, una tendencia vigente es más probable que continúe a que se revierta. Tiene cierta similitud con la primera ley de la mecánica clásica de Newton. El movimiento del los valores de mercado sigue su inercia. El objetivo principal del análisis técnico es identificar una tendencia en su etapa temprana, de este modo la probabilidad de que el valor de mercado siga la misma dirección durante cierto periodo de tiempo es alta. Esta tendencia se revertirá en algún momento, y es preciso atender a las señales que lo indiquen.
3. *La historia se repite*. Se basa en que los seres humanos solemos reaccionar de manera parecida ante circunstancias que tienden a ser iguales. Las formaciones técnicas que aparecen en las gráficas, por ejemplo, son consecuencia del sentimiento alcista o bajista del mercado, que tiende a comportarse de la misma manera en circunstancias similares. Según este principio, la clave para entender el futuro es el estudio del pasado.

Las conclusiones obtenidas provenientes del análisis técnico tienen una mayor fiabilidad a un corto o medio plazo. Esto es debido a que las cotizaciones de las acciones a largo plazo pueden tener una mayor influencia de factores externos sociales, político o internos de empresas concretas, que en el momento de la predicción pueden ser totalmente imprevisibles. Además, el análisis puramente técnico obtendrá mejores predicciones cuando se aplica a un mercado con cierto movimiento de compra y venta, es decir, un mercado suficientemente líquido para que un mayor número de inversores haya participado con su opinión acerca de cómo evolucionará la cotización de una determinada acción. De esta manera el pronóstico de la masa de inversores, y por tanto el precio de la acción a tratar, será más representativo y fiable.

El análisis técnico estudia la acción de mercado, que incluye las tres principales fuentes de información para los analistas, relacionadas con el precio:

1. Precio o cotización. Es la variable más importante de la acción de mercado.
2. Volúmen. Es la cantidad de unidades o contratos operados durante un periodo de tiempo concreto.
3. Interés abierto. Representa el número de contratos que permanecen abiertos al finalizar un periodo definido.

### 1.1.2 Tipos de análisis técnico

Cabe mencionar además que el análisis técnico puede dividirse en dos subcategorías:

- Análisis gráfico o chartista, el cuál analiza exclusivamente la información revelada en los gráficos, sin la ayuda de herramientas adicionales.
- Análisis técnico en sentido estricto, el cual emplea indicadores calculados en función de las variables características procedentes de la acción de mercado del valor analizado. También es llamado, a falta de un término formalizado, análisis técnico estadístico. Su objetivo es establecer un sistema de predicción mecánico en base a estos indicadores calculados, que reciben el nombre de indicadores técnicos.

En muchas ocasiones se realizaun acercamiento entre estas dos subcategorías. Especialmente, los chartistas suelen obtener información extra de ciertos indicadores técnicos para apoyar lo que se visualiza únicamente a partir de los gráficos de tendencia. En ese caso, es posible considerar que se lleva a cabo un análisis mixto de estos dos modos de operación.

### 1.1.3 Diferencias con análisis fundamental

Existen grandes diferencias entre ambos análisis, y cabría decir que en algunos aspectos son hasta contrapuestos.

Por un lado el análisis técnico solo tiene en cuenta el movimiento del mercado, mientras que el análisis fundamental intenta concocer su valor intrínseco, su valor real. Es decir, si el valor real de un mercado está por encima del precio actual, será preciso comprar, mientras que del modo contrario es momento de vender. En base a estos elementos desarrollarán sus predicciones. Se puede decir que el análisis fundamental estudia la causa de los movimientos de mercado, mientras que el análisis técnico estudia el efecto. Sin embargo, en muchas ocasiones los resultados obtenidos por el análisis técnico y el análisis fundamental son opuestos. Al realizar análisis mixtos, y obtener resultados opuestos, es posible que se den momentos de incertidumbre, que normalmente ocurren cuando se inician grandes movimientos de mercado. En esos momentos, el análisis fundamental puede no respaldar lo que predice el técnico, y aunque a veces vuelven a ir sincronizados pasado un tiempo, puede que sea algo tarde para algunos inversores.

Estas discrepancias pueden tener su explicación en que el precio refleja su valor según los fundamentos conocidos. Es decir, se descuenta según el conocimientos convencional, y es lo que se refleja en el mercadodel momento. Sin embargo, hay fundamentos ocultos, que van siendo reflejados en el precio, y antes de lo pervisto pueden ser el origen de una nueva tendencia inesperada.

En definitiva, se puede decir que el enfoque técnico incluye en cierto modo al fundamental, sin embargo, al contrario no. Por ello, si se debiera apostar por un único enfoque entre ambos, es probable que el más adecuado fuera un análisis técnico, ya que el análisis fundamental necesita más del enfoque técnico que vicerversa.

### 1.1.4 Aplicación al proyecto

En este trabajo se llevará a cabo un análisis técnico estadístico. Es decir, se hará uso de un conjunto de datos de indicadores técnicos en un periodo temporal concreto, y junto con los datos del precio del valor a analizar, se hallará la mejor predicción posible del precio futuro. En el siguiente capítulo se expondrá la tipología de los indicadores técnicos, y se introducirán de manera breve aquellos utilizados en el presente proyecto.

El proyecto está basado en el artículo *"A Nonlinear Technical Indicator Selection Approach for Stock Markets. Application to the Chinese Stock Market"*, publicado por Daniel Rodríguez Ramírez, tutor de este proyecto, y Gerardo Alfonso. El artículo en cuestión presenta un algoritmo, el cual se aplicaba a la bolsa de valores China, y cuya propuesta estaba enfocada en predecir la tendencia del precio de cierre de las acciones, mediante el empleo de redes de neuronas. En este proyecto el objetivo es algo distinto, ya que no se pretende únicamente predecir la tendencia de precio (si es ascendente o descendente), sino también su valor específico con la mayor precisión posible. Para ello no se usarán redes de neuronas en un principio (aunque se prepara para una posible ampliación en un proyecto futuro) y el objeto de estudio serán un índice y una acción determinados.

# 2 INDICADORES TÉCNICOS

Los indicadores técnicos son una herramienta de análisis bursátil que sirve para estudiar la evolución futura de los precios de un activo financiero. Se obtienen mediante formulación matemática y estadística de complejidad variable, y el dato fundamental utilizado es el precio de cierre del índice a analizar. Estos indicadores ayudan a interpretar si los precios de un activo se encuentran en situación de sobrecompra o sobreventa, con la finalidad de realizar una inversión de manera acertada. En definitiva, son las componentes o herramientas necesarias para llevar a cabo un análisis de tipo técnico.

## 2.1 Tipos de indicadores técnicos

Los indicadores técnicos pueden dividirse en cuatro grandes categorías:

- Tendencia: Estos indicadores ayudan a identificar cuando un valor de mercado está siguiendo una determinada tendencia, ya sea ascendente, descendente o de acumulación lateral. Cuando se encuentra en un momento de acumulación lateral los indicadores de este tipo son menos efectivos que cuando siguen una tendecia ascendente o descendente. Algunos ejemplos de esta categoría son las medias móviles, las bandas de Bollinger, el parabólico Sar, la desviación estándar o el Ichimoku.
- Osciladores: Los indicadores técnicos osciladores se mueven en un rango entre 0 y 100, y nos permiten detectar zonas de sobrecompra o sobreventa, divergencias y convergencias que nos ayudarán a anticipar cambios de tendencia. Algunos de estos osciladores también nos permiten detectar tendencias y la fortaleza de las mismas. Se construyen a través de la variable precio combinada con otros factores como la volatilidad, la velocidad de movimiento o medias móviles. Los más importantes son el MACD, RSI, estocástico, momentum, ATR y Williams.
- Volúmenes: Los indicadores de esta categoría se construyen a partir del volumen de negociación en el mercado. A esta categoría pertenecen el indicador “money flow” y la acumulación/distribución.
- Bill Williams: Combinan diferentes variables basadas en modelos matemáticos simples, aunque algo más complejos que los de categorías anteriores. Destacan el oscilador de aceleración, los fractales y el Alligator.

Los retrocesos de Fibonacci se consideran un indicador técnico, aunque no entra en ninguna de estas categorías, ya que es de tipo puramente chartista. Se trata de la búsqueda de puntos de entrada en el mercado a través de niveles de retroceso específicos que cumplen una lógica matemática basada en la sucesión numérica

de la conocida serie que le da nombre.

## 2.2 Indicadores técnicos presentes en los datos

Los indicadores técnicos son la base del algoritmo que se presenta en este proyecto. Es importante saber que a la hora de decidir acerca de una inversión, no se debe tener en cuenta únicamente un indicador técnico. Es necesario contrastar la información que proporciona con otros indicadores. Sin embargo, esto no quiere decir que cuanta más información de distintos indicadores se valore, mejor será la predicción. Por tanto, dependiendo del momento en el que se analiza el mercado y el índice que se estudie, los indicadores técnicos que aporten una información adecuada variará. En este trabajo se cuenta con 40 indicadores diferentes entre los que elegir para hallar la mejor combinación que estime el precio de cierre. La siguiente tabla recoge este conjunto de indicadores presentes en el conjunto de datos que se usará para la aplicación del algoritmo:

| Nº | Indicador | Descripción |
|---|---|---|
| 1 | SMAVG(50) | 50 days simple MA. $\frac{\sum_{i=0}^{49} C_{t-i}}{50}$ |
| 2 | SMVAG(100) | 100 days simple MA. $\frac{\sum_{i=0}^{99} C_{t-i}}{100}$ |
| 3 | SMVAG(200) | 200 days simple MA. $\frac{\sum_{i=0}^{199} C_{t-i}}{200}$ |
| 4 | Volume | Total traded volume |
| 5 | Stochastic %K (14-day) | $\frac{C_t - \min(L_{t-n,\forall n\in\{1,\dots,14\}})}{\max(H_{t-n,\forall n\in\{1,\dots,14\}}) - \min(L_{t-n,\forall n\in\{1,\dots,14\}})}$ |
| 6 | %D (3-day) | 3 days MA of stochastic %K. $\frac{\sum_{i=0}^{2} K_{t-i}}{3}$ |
| 7 | %D (5-day) | 5 days MA of stochastic %K. $\frac{\sum_{i=0}^{4} K_{t-i}}{5}$ |
| 8 | Slow %D (3-day) | MA of the 3 days %D. $\frac{\sum_{i=0}^{2} D_{t-i}}{3}$ |
| 9 | Slow %D (5-day) | MA of the 5 days %D. $\frac{\sum_{i=0}^{4} D_{t-i}}{5}$ |

| | | |
|---|---|---|
| 10 | Momentum (10-day) | Change in Price over a 10 days period. $C_t - C_{t-10}$ |
| 11 | Moving average (5D) of Momentum (10-day) | 5 days MA of 10-day Momentum. $\frac{\sum_{i=0}^{4} M_{t-i}}{5}$ |
| 12 | ROC (daily) | Change in price in % over 1 day period. $\frac{C_t}{C_{t-1}}100$ |
| 13 | Moving average (5D) of ROC | 5 days MA of ROC. $\frac{\sum_{i=0}^{4} ROC_{t-i}}{5}$ |
| 14 | Williams (14) | 14 days' Williams indicator $\frac{\max(H_n, \forall n \in \{1,...,14\}) - C_t}{\max(H_n, \forall n \in \{1,...,14\}) - \min(L_n, \forall n \in \{1,...,14\})}$ |
| 15 | A/D oscillator | Accumulation distribution indicator. $\frac{H_t - C_{t-i}}{H_t - L_t}$ |
| 16 | MA (5) | 5 days moving average. $\frac{\sum_{i=0}^{4} C_{t-i}}{5}$ |
| 17 | MA (10) | 10 days moving average. |
| 18 | Disparity 5 | $\frac{C_t}{MA_t}$ |
| 19 | Disparity 10 | $\frac{C_t}{MA_t}$ |
| 20 | OSCP | $\frac{MA_t - MA_n}{MA_t}$ |
| 21 | CMCI (13) | $\frac{(H_t + L_t + C_t)/3 - \frac{\sum (H_{t-i+1} + L_{t-i+1} + C_{t-i+1})}{3n}}{0.015(\frac{\sum \lvert H_{t-i+1} + L_{t-i+1} + C_{t-i+1} - \sum ((H_t + L_t + C_t)/3 \rvert}{n})}$ |
| 22 | RSI (14) on Close | Relative strength index. $100 - \frac{100}{1 + \frac{Average\_up}{Average\_down}}$ |
| 23 | MA (250) | 250 days simple MA. $\frac{\sum_{i=0}^{249} C_{t-i}}{250}$ |
| 24 | Z-score | Distance in standard deviations from MA |

| | | |
|---|---|---|
| 25 | AMAVG (14,2,30) | Exponential MA |
| 26 | EMAVG (5) on Close | 5 days exponential moving average with exponential term $\frac{2}{t+1}$ |
| 27 | TMAVG (5) on Close | 5 days triangular moving average with factor $\frac{t+1}{2}$ |
| 28 | VMAVG (5) on Close | Variable MA with smoothing based in volatility |
| 29 | MACD (12,26) | T1 exponential MA (fast) − T2 exponential MA (slow) |
| 30 | Hurst (25) | Exponential MA with mean reversion assumption. |
| 31 | FG (5) | Ratio of buying and selling strength (5 days) |
| 32 | Kairi (simple, 25) | Deviation of prices from its MA |
| 33 | Elder Force Index | Relationship between price and trading volume movements |
| 34 | JKHL Index | Relationship between new high and lows smoothed by two MA. |
| 35 | RMI (Close,14,10) | Relationship between overbought and oversold levels |
| 36 | PEMA15 | being $p^d_{EMA}(t) = \frac{2}{d+1} p(t) + (1 - \frac{2}{d+1}) p^d_{EMA}(t-1)$ |
| 37 | RDP_5 | $100 \frac{p(t) - p(t-5)}{p(t)}$ |
| 38 | RDP_10 | $100 \frac{p(t) - p(t-10)}{p(t)}$ |
| 39 | RDP_15 | $100 \frac{p(t) - p(t-15)}{p(t)}$ |
| 40 | RDP_20 | $100 \frac{p(t) - p(t-20)}{p(t)}$ |

Tabla 2-1 Indicadores presentes en los conjuntos de datos

Para la aplicación del algoritmo de este proyecto, se ha usado dos conjuntos de datos diferentes. En primer lugar, los valores de los anteriores 40 indicadores recopilados diariamente del índice americano Dow Jones. En segundo lugar, los valores de 24 de esos 40 indicadores recopilados minuto a minuto, para la acción Ping An Bank. Ambos conjuntos de datos contienen adicionalmente el precio de cierre del índice en el día posterior a los valores de los indicadores recogidos.

# 3 ALGORITMO

A continuación se introducirá el plantamiento teórico del problema a solucionar por el algortimo, junto con los pasos necesarios para su implementación práctica.

Siendo $X_T^i(t)$ una tupla de $T$ valores del i-ésimo indicador de un conjunto de hasta N indicadores técnicos no lineales, computados en un periodo de tiempo $t$ (normalmente el periodo de tiempo será medido en días, pero podrían ser semanas o meses), como por ejemplo la media móvil:

$$X_T^i(t) = \left\{ X^i(t-(T-1)), X^i(t-(T-2)), \ldots, X^i(t) \right\} \tag{3.1}$$

donde $X^i(t)$ es el valor del indicador técnico i-ésimo evaluado en el tiempo $t$.

Llamaremos $Y_T(t)$ a una tupla de $T$ valores de los precios de cierre de mercado, computados en un periodo de tiempo $t$.

$$Y_T(t) = \left\{ Y(t-(T-1)), Y(t-(T-2)), \ldots, Y(t) \right\} \tag{3.2}$$

Como se menciona anteriormente, los indicadores técnicos en el mercado de valores pueden producir señales contradictorias, y algunos métodos no lineales pueden tener problemas en los mínimos locales cuando se usan variables de entrada de gran dimensión, como por ejemplo en $N$ o $T$. De este modo, el algoritmo que se presenta en este capítulo pretende solucionar estos problemas, siguiendo los pasos que se enumeran a continuación.

## 3.1 Pasos a seguir

1. Dividir la base de datos disponible de $X_T^i(t)$ y $Y_T(t)$ en dos subconjuntos, un subconjunto de estimación, $S_e \triangleq \left\{X_{T,e}^i(t), Y_{T,e}(t)\right\}$ y uno de validación o test $S_v \triangleq \left\{X_{T,v}^i(t), Y_{T,v}(t)\right\}$.
2. Generar $C_S$, un conjunto de $M$ combinaciones de $\left\lfloor \frac{N}{2} \right\rfloor$ números aleatorios en el rango $\{1, 2, ..., N\}$. Comprobar las posibles repeticiones en cada combinación y entre combinaciones. Cada combinación deberá estar formada por números sin repetición, y una combinación no puede ser exactamente igual a otra. Repetir el proceso hasta que se den estas condiciones.
3. Para cada una de las combinaciones de las $M$ escogidas en $C_S$, denominadas $I \in C_S$, seguir los siguientes pasos:
   a. Computar $\hat{Y}_{T,v}(t)$, a partir de $S_e$ y $X_{T,v}^i(t)$ mediante el método conveniente. Esta será la predicción del valor del precio de cierre de mercado en cada momento $t$.
   b. Calcular el error $\xi_T(t)$ de tal forma que

$$\xi_T(t) = \left|\hat{Y}_{T,v}(t) - Y_{T,v}(t)\right|$$

con el resultado del error total siendo:

$$\xi^{Total} = \frac{\sum_{\forall t \in S_v} \xi_T(t)}{card(Sv)} \tag{3.3}$$

donde $card(Sv)$ es la dimensión del conjunto de validación. De esta forma este es el error total de la combinación inicial elegida en el paso 2.

   c. Generar aleatoriamente un único valor $i_a \in \{1, 2, ..., N\}$ para cada combinación. Este valor hará referencia a un indicador candidato a ser añadido a cada una de las combinaciones. Debe garantizarse que no estuviera previamente incluido en los $\frac{N}{2}$ valores originales. En caso de que estuviera incluido, se volverá a generar un valor $i_a$ diferente hasta que no haya repetición. Se hará lo propio para todas las combinaciones.
   d. Generar aleatoriamente un único valor $i_r \in \{1, 2, ..., N\}$ para cada combinación, Estos valores corresponderán a un indicador técnico que será eliminado de cada combinación del set original.
   e. Formar un nuevo set de combinaciones $I$ donde $I^{up} \triangleq I \cup \{i_a\}$. Del mismo modo, elaborar otro set $I^{down}$ donde $I^{down} \triangleq I - \{i_r\}$. Si alguna de estas nuevas combinaciones ya están en $C_s$, volver al paso 3c o 3d hasta que no haya repetición. De este modo, tendremos $3M$ combinaciones distintas.
   f. Como se hizo previamente en los pasos 3a, computar $\hat{Y}_{T,v}(t)$, esta vez con los 2 nuevos sets

de combinaciones.

g. De la misma forma que en el paso 3b, calcular los errores totales de los nuevos sets, a los que llamaremos $\xi_{up}^{Total}$ y $\xi_{down}^{Total}$, respectivamente.

h. Dados los 3 sets de combinaciones anteriores $I$, $I^{up}$, $I^{down}$, y sus respectivos errores $\xi^{Total}$, $\xi_{up}^{Total}$ y $\xi_{down}^{Total}$, tomar de cada combinación, las dos que tengan menor error. No se hará la selección entre todas las combinaciones, sino entre cada trío de combinaciones formados por la original y las dos que provienen de ella. Esto dará lugar a las $2M$ combinaciones con menor error.

i. Seleccionar las $M$ combinaciones de menor error, esta vez entre todas las combinaciones restantes.

j. Evaluar el error $\xi^{Total}$ mínimo de todas estas $M$ combinaciones $I \in C_s$.

4. Repetir el paso 3 comenzando por 3d con el nuevo set $I$, hasta que se llegue a una condición de fin determinada. Esta condición se dará al obtener un error mínimo, o bien al llegar a un número máximo de iteraciones, de forma que no se de un bucle infinito.

Para una mejor explicación de cada paso, se ilustrará con un pequeño ejemplo, con solo dos combinaciones $(M = 2)$ y una iteración. Se asumirá que el número total de indicadores técnicos disponibles es 6, esto es, $\{X^1, X^2, X^3, X^4, X^5, X^6\}$. En primer lugar, el algoritmo selecciona aleatoriamente 2 combinaciones de tres indicadores técnicos, la mitad de los disponibles. Por ejemplo:

$$C_S = \begin{Bmatrix} I_1 \\ I_2 \end{Bmatrix} = \begin{Bmatrix} \{2,4,6\} \\ \{1,2,5\} \end{Bmatrix} \tag{3.5}$$

Se computará el valor del error obtenido en ambas combinaciones:

$$\xi^{Total} = \begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} = \begin{Bmatrix} 0.6 \\ 0.7 \end{Bmatrix} \tag{3.6}$$

A continuación, el algoritmo obtendrá los valores $i_a$ e $i_r$, tras los intentos necesarios para que no haya repetición en las combinaciones. El valor $i_a$ corresponderá directamente al número de indicador que se añadirá, mientras que $i_r$ se refiere al número de columna de cada combinación cuyo indicador será eliminado.

$$i_a = \begin{Bmatrix} 1 \\ 3 \end{Bmatrix}$$

$$i_r = \begin{Bmatrix} 3 \\ 2 \end{Bmatrix} \tag{3.8}$$

De este modo se formarán los nuevos set de combinaciones $I_{up}$ e $I_{down}$:

$$I^{up} = \begin{Bmatrix} I_1^{up} \\ I_2^{up} \end{Bmatrix} = \begin{Bmatrix} \{2,4,6,1\} \\ \{1,2,5,3\} \end{Bmatrix} \tag{3.9}$$

$$I^{down} = \begin{Bmatrix} I_1^{down} \\ I_2^{down} \end{Bmatrix} = \begin{Bmatrix} \{2,4\} \\ \{1,5\} \end{Bmatrix} \tag{3.10}$$

Llegado este punto, hemos pasado de tener dos combinaciones a tener dos familias de combinaciones, cada cual formada por 3 de ellas $\left(I^{up}, I^{down}, I\right)$ , y por tanto tenemos un total de 6 (3M). En cada una de ellas no hay indicadores repetidos, y ninguna es igual a otra. Esto es importante para no usar el mismo indicador dos veces en una misma combinación, y para no computar el error de dos combinaciones exactamente iguales.

Una vez que se han generado los nuevos conjuntos, se evalúan sus errores totales, lo que da lugar a los dos set de errores nuevos $\xi_{up}^{Total}$ y $\xi_{down}^{Total}$ , y al original $\xi^{Total}$ :

$$\xi_{up}^{Total} = \begin{Bmatrix} \xi_1^{up} \\ \xi_2^{up} \end{Bmatrix} = \begin{Bmatrix} 0.4 \\ 0.8 \end{Bmatrix}$$

$$\xi_{down}^{Total} = \begin{Bmatrix} \xi_1^{down} \\ \xi_2^{down} \end{Bmatrix} = \begin{Bmatrix} 0.9 \\ 0.3 \end{Bmatrix}$$

$$\xi^{Total} = \begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} = \begin{Bmatrix} 0.6 \\ 0.7 \end{Bmatrix}$$

A continuación, el objetivo es pasar a tener únicamente 2M combinaciones. Esta se llevará a cabo seleccionando las dos combinaciones con menor error de cada familia. La posibilidad de aplicar este algoritmo en redes de neuronas (tema que no es materia de este proyecto, pero que podría ser una ampliación interesante del mismo) hace útil añadir este paso intermedio, en el que no se guardan directamente las M mejores combinaciones, sino que se seleccionan en un primer momento entre las de la cada familia de forma independiente. Por tanto, de la primera familia nos quedaremos con $I_1^{up}$ e $I_1$, mientras que de la segunda nos quedaremos con $I_2^{down}$ e $I_2$ . De este modo, las combinaciones que permanecerán en el proceso serán las siguientes:

$$C_S = \begin{Bmatrix} I_1^{up} \\ I_1 \\ I_2^{down} \\ I_2 \end{Bmatrix} = \begin{Bmatrix} \{2,4,6,1\} \\ \{2,4,6\} \\ \{1,5\} \\ \{1,2,5\} \end{Bmatrix}$$

con sus errores:

$$\xi = \begin{Bmatrix} 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \end{Bmatrix}$$

Ahora, se reducirá el número de combinaciones al original ( $M = 2$ ), eligiendo las que tengan un error menor, que serían las siguientes:

$$C_S = \begin{Bmatrix} I_2^{down} \\ I_1^{up} \end{Bmatrix}$$

con sus errores:

$$\xi = \begin{Bmatrix} 0.3 \\ 0.4 \end{Bmatrix}$$

Aquí se llegaría al final de la primera iteración. Si siguiéramos con más de una iteración, este último conjunto obtenido, $C_S$, sería con el que se comenzaría la siguiente, y se repetiría de nuevo el mismo proceso con estas $M$ combinaciones. En este pequeño ejemplo solo contábamos con una única iteración, por tanto nos quedaríamos con $I_2^{down}$ como la mejor combinación de indicadores técnicos, que serían $\{X^1, X^5\}$, con su error (0.3).

# 4 IMPLEMENTACIÓN

La implementación del algoritmo descrito se ha realizado con la herramienta MATLAB y de forma modular, para facilitar la corrección de errores y ladesarrollo de posibles mejoras futuras. Se pueden distinguir fácilmente seis módulos independientes que se exponen a continuación, junto con sus tareas a realizar y la comunicación entre ellos. Se mencionan en cada apartado los nombres de las funciones de MATLAB correspondientes a cada módulo, y los códigos pueden consultarse en el apéndice del proyecto.

## 4.1 Procesamiento de datos

En primer lugar, para pasar la base de datos de una hoja Excel a MATLAB, se hace uso del script "Procesar_Datos.m". Estos datos son volcados en la matriz "Data1". Las filas de "Data1" están ordenadas por fechas de los datos recogidos, y cada columna contiene los valores de un indicador técnico distinto, salvo la última, que tiene los precios de cierre del índice por cada fecha. Este script se ha realizado de forma independiente a las demás funciones, para que no sea necesario que se ejecute siempre al hacer varias pruebas.

## 4.2 Generador de combinaciones

El generador de combinaciones se encargará de confeccionar la combinación inicial $C_S$. Se llevará a cabo mediante la función "c_generator.m", que tendrá como argumentos de entrada:

- El número de combinaciones deseado
- El número de indicadores que se desea que tenga cada combinación.
- El número total de indicadores entre los que se elige cada combinación.

La salida será una variable tipo cell que contenga este conjunto de combinaciones deseado. Se han usado celdas en lugar de matrices, ya que más adelante se deberán guardar combinaciones de distintas dimensión, y para esto es mucho más práctico usar este tipo. Cabe recordar que esta función asegura que ninguna de las combinaciones sea igual a otra del conjunto, y que ningún indicador estará repetido en una misma combinación.

## 4.3 Extracción de los datos de entrada

Este módulo se encarga de extraer de la matriz "Data1" los conjuntos de entrenamiento y de test, dependientes de la combinación que se quiera evaluar en el momento. Para ello, se usa la función "computation.m", la cual tiene como argumentos de entrada:

- La matriz de datos "Data1"
- El conjunto de combinaciones
- La longitud del set de entrenamiento
- La longitud del set de test

Como argumentos de salida tendrá:

- Las entradas del conjunto de entrenamiento, "X_DAT"
- Las entradas del conjunto de test, "X_TEST"

## 4.4 Optimización Direct Weight

Para realizar la predicción del comportamiento del precio, se ha usado el método Direct Weight. Este método consiste en encontrar una estimación de la función óptima mediante optimización convexa, y es aplicable para la identificación de sistemas no lineales. Para llevarlo a cabo en código, se ha implementado la función "DirectWeight.m", que tiene como argumentos de entrada:

- Las entradas del conjunto de entrenamiento, "X_DAT", creado anteriormente
- Las salidas del conjunto de entrenamiento "Y_DAT"
- Las entradas del conjunto de test, "X_TEST", creado anteriormente
- Número de puntos vecinos en los que se va a apoyar la predicción

La función obtendrá una matriz de predicciones de cada elemento del conjunto de test, es decir, "Y_PRED", y las pasará como salida.

## 4.5 Generador de combinaciones "up" y "down"

Con el objetivo de generar las nuevas combinaciones "up" y "down", se diferencia otro modulo, cuya función en MATLAB se llamará "I_updown.m". Como se explica en el capítulo dos, en el paso 3e es necesario crear dos conjuntos de combinaciones nuevos, uno de ellos con un indicador técnico menos cada una, y otro con un indicador técnico más. Es necesario asegurar que los conjuntos nuevos tengan combinaciones únicas y con indicares sin repetición. La función mencionada tendrá como entradas:

- El conjunto de combinaciones original

Y como salidas:

- El nuevo conjunto de combinaciones up
- El nuevo conjunto de combinaciones down

Tal y como ocurre con el conjunto original, estos nuevos conjuntos estarán recogidos en variables tipo cell.

## 4.6 Programa principal

El programa principal se encarga de relacionar los módulos anteriores y realizar algunas tareas propias para seguir los pasos del algoritmo. Se ha implementado en el script "main.m", y concretamente lleva a cabo las siguientes acciones de manera ordenada:

1. Inicialización de valores de los parámetros (número de combinaciones, número de iteraciones, límite del error, número de vecinos)
2. Asignación de las longitudes deseadas de los conjuntos de datos (entrenamiento y test), y recopilación de las salidas de los mismos ("Y_DAT", "Y_TEST")
3. Generación del conjunto de combinaciones original, llamando al módulo 2 referido anteriormente
4. Computación del conjunto de combinaciones original, mediante el siguiente procedimiento:
    a. Extracción de los datos de entrada llamando al módulo 3
    b. Optimización Direct Weight llamando al módulo 4
    c. Evaluación del error de cada combinación
5. Comienzo del bucle de iteraciones
    a. Generación del conjunto de combinaciones "up" y "down" llamando al módulo 5
    b. Computación del conjunto "up", siguiendo los mismos pasos que en el paso 4 para el original
    c. Computación del conjunto "down" del mismo modo
    d. Selección de las dos combinaciones con menos error de cada familia
    e. Selección de las $M$ mejores combinaciones. Se guarda la mejor combinación y su error. Vuelta al inicio del paso 5 con estas $M$ combinaciones y sus errores hasta que llegue al máximo de iteraciones
6. Representación gráfica:
    a. Resultado de la mejor combinación obtenida. Comparativa con los datos de salida reales
    b. Evolución del error a lo largo de las iteraciones

Cabe señalar que se ha optado por una implementación orientada a la ejecución paralela de ciertos tramos de código, concretamente en las computaciones de los conjuntos de combinaciones; es decir, los pasos 4, 5b y 5c. Para ello se han usado los commandos de MATLAB "parpool" y "parfor", que crean distintos trabajadores según el número de núcleos que posea el dispositivo, y ejecutan las iteraciones de forma paralela. Las pruebas y comparativa de su eficiencia se desarrollarán en los siguientes capítulos de pruebas y resultados.

# 5 PRUEBAS Y RESULTADOS: DOW JONES

En este capítulo se exponen los resultados obtenidos al probar el código con los datos extraídos de ciertos indicadores para el índice Dow Jones. Concretamente, se usan datos diarios desde el día 20 de octubre de 1993 hasta el día 8 de julio de 2020. Estas pruebas son de suma importancia, no solo para verificar el correcto funcionamiento del algoritmo para distintos valores de entrada, sino también para comprobar qué elementos y valores conviene usar para alcanzar los mejores resultados en el menor tiempo posible. Esta documentación debería facilitar al usuario del algoritmo el conocimiento de las condiciones más favorables, para que funcione en cada caso de la mejor manera posible.

## 5.1 Elementos a considerar

A la hora de ejecutar el algoritmo, hay ciertos parametros y elementos a tener en cuenta. Algunos de estos parámetros son el número de combinaciones, iteraciones y de puntos vecinos. Todo esto, junto con otros elementos como la paralelización en ciertos tramos, condicionan la eficiencia del mismo.

A continuación se presentan las variables que han sido objeto de estudio en este capítulo, para confeccionar una documentación completa de la puesta en marcha del algoritmo.

### 5.1.1 Combinaciones

El número de combinaciones $M$ puede modificarse. $M$ es el valor de combinaciones con el que empieza y termina cada iteración. Al incrementar su valor, se necesitarán menos iteraciones para encontrar la combinación que aporte un menor error. Por otro lado, esto supondrá un aumento en el tiempo de ejecución, por lo que es interesante comprobar qué número es adecuado para maximizar la eficiencia. En las pruebas realizadas se ha hecho uso de distintas $M$, que van desde 5 hasta 10000. Si hubiera sido necesario, se habrían tomado valores mayores, pero en el ejemplo que nos atañe no ha hecho falta. En la tabla 5-1 aparece un análisis de la variación del error y del tiempo de ejecución variando el número de combinaciones, y con una única iteración.

| Combinaciones | Iteraciones | Error total | Tiempo de ejecución (s) |
|---|---|---|---|
| 5 | 1 | 342.99 | 2.167 |
| 10 | 1 | 322.57 | 4.015 |
| 20 | 1 | 320.28 | 6.600 |
| 50 | 1 | 324.19 | 15.016 |
| 100 | 1 | 319.04 | 28.880 |
| 200 | 1 | 316.95 | 56.360 |
| 500 | 1 | 313.84 | 145.704 |
| 1000 | 1 | 314.83 | 290.019 |

Tabla 5-1 Pruebas con variación en las combinaciones

La siguiente figura muestra una gráfica que representa la evolución del error según el número de combinaciones, para una sola iteración:

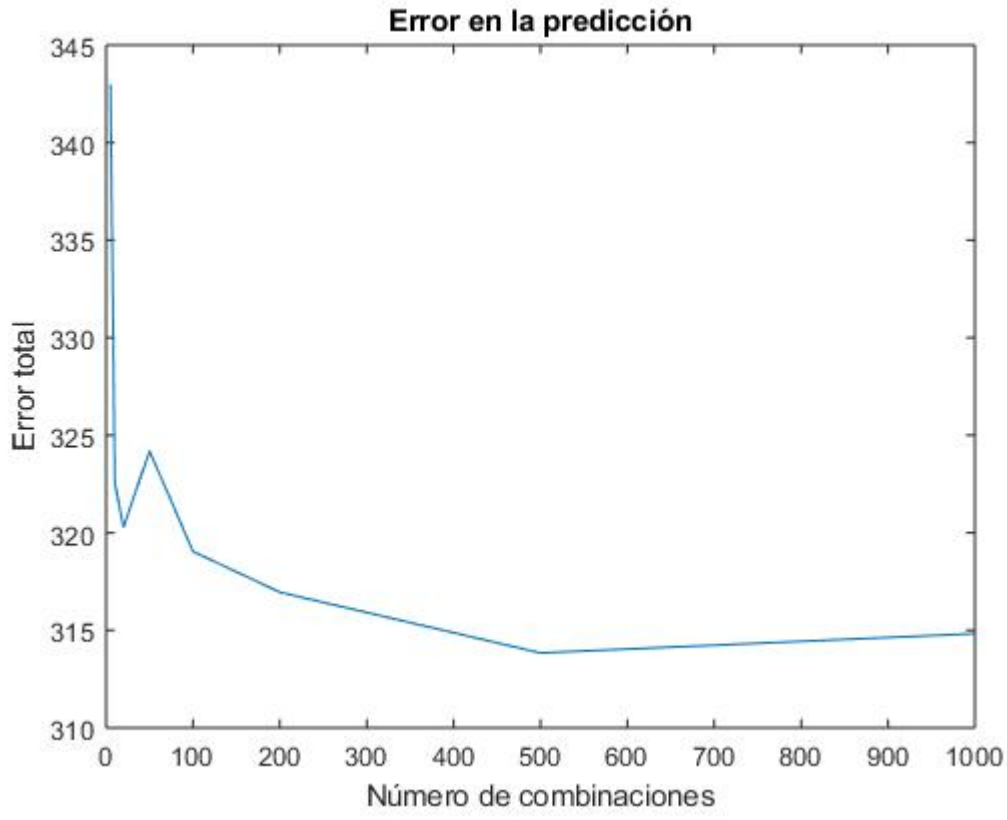


Figura 5-1 Evolución del error en la predicción dependiendo de las combinaciones.

Se puede observar cómo al aumentar las combinaciones por lo general disminuye el error, pero es una cuestión de pura probabilidad, ya que en el último tramo el error asciende, usando nada menos que el doble de combinaciones que en la anterior prueba.

### 5.1.2 Iteraciones

Con el número de iteraciones sucede lo mismo. Cuanto mayor sea, más probabilidad hay de que la búsqueda de la mejor combinación sea exitosa. Las pruebas se han realizado con un intervalo de entre una y 1000 iteraciones. La tabla 5-2 expone los resultados obtenidos con la variación de iteraciones. Esta vez se mostrarán el número de indicadores en la mejor selección de cada prueba, ya que no tienen por qué ser 20.

| Combinaciones | Iteraciones | Error total | Tiempo de ejecución | Nº de indicadores |
|---|---|---|---|---|
| 5 | 5 | 334.71 | 6.318 | 20 |
| 5 | 10 | 321.24 | 13.111 | 21 |
| 5 | 20 | 310.1 | 25.235 | 12 |
| 5 | 50 | 303.41 | 56.150 | 13 |
| 5 | 100 | 306 | 119.447 | 16 |
| 5 | 200 | 297.73 | 242.833 | 16 |
| 5 | 500 | 304.32 | 556.082 | 14 |
| 5 | 1000 | 302.92 | 1109.623 | 14 |

Tabla 5-2 Pruebas con variación en las iteraciones

De igual modo, las gráficas 5-2 a 5-9, muestran este error a medida que avanzan las iteraciones.

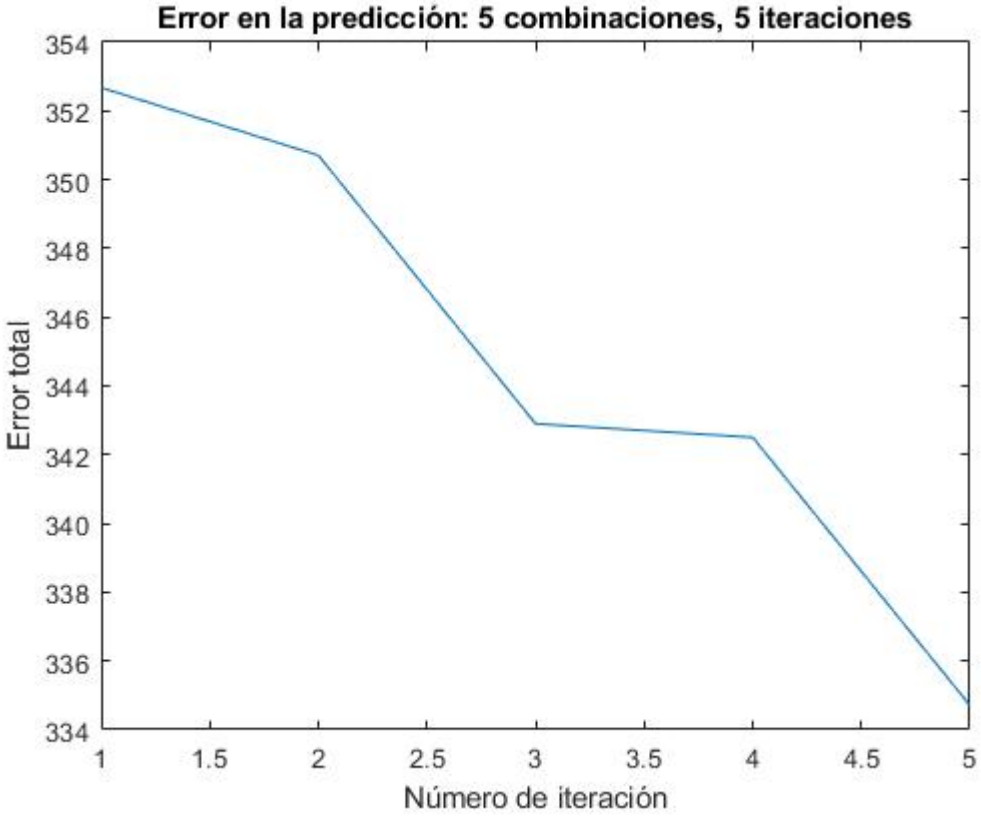


Figura 5-2 Evolución del error depediendo de las iteraciones, hasta cinco.

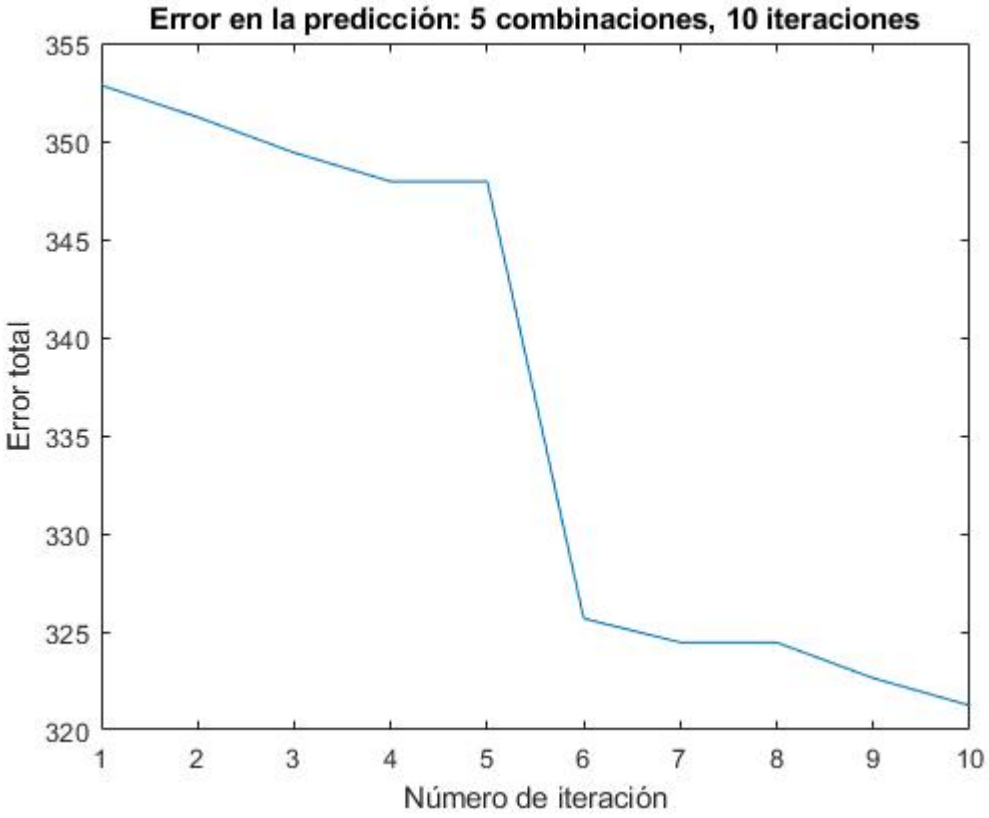


Figura 5-3 Evolución del error dependiente de las combinaciones, hasta diez.

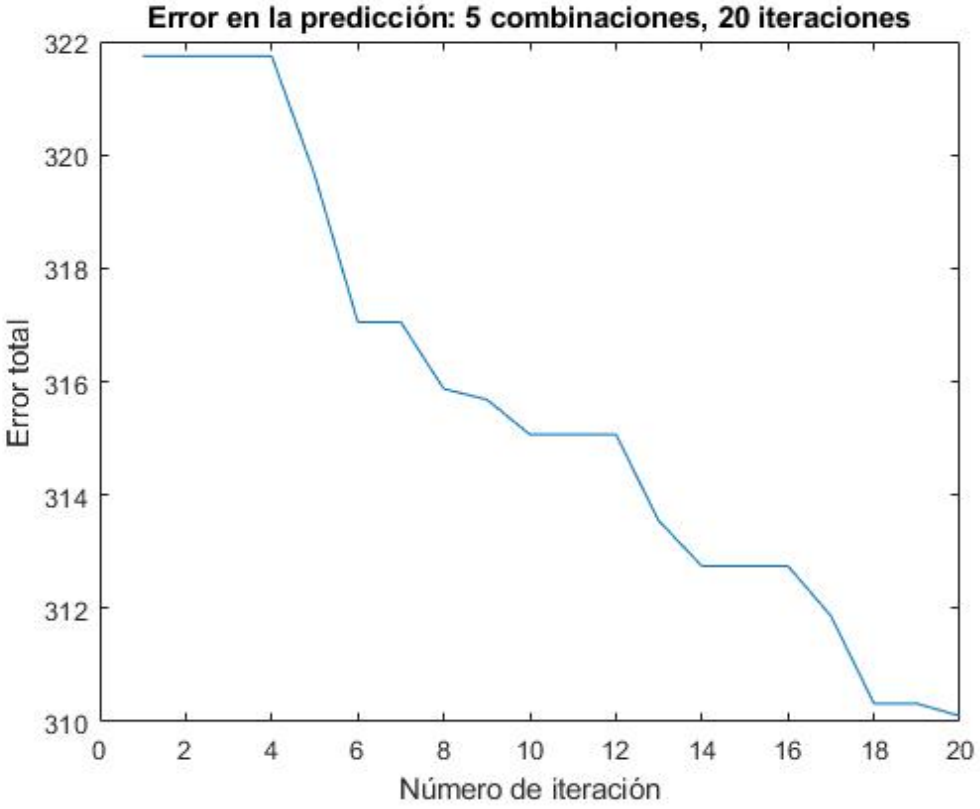


Figura 5-4 Evolución del error dependiente de las iteraciones, hasta veinte.

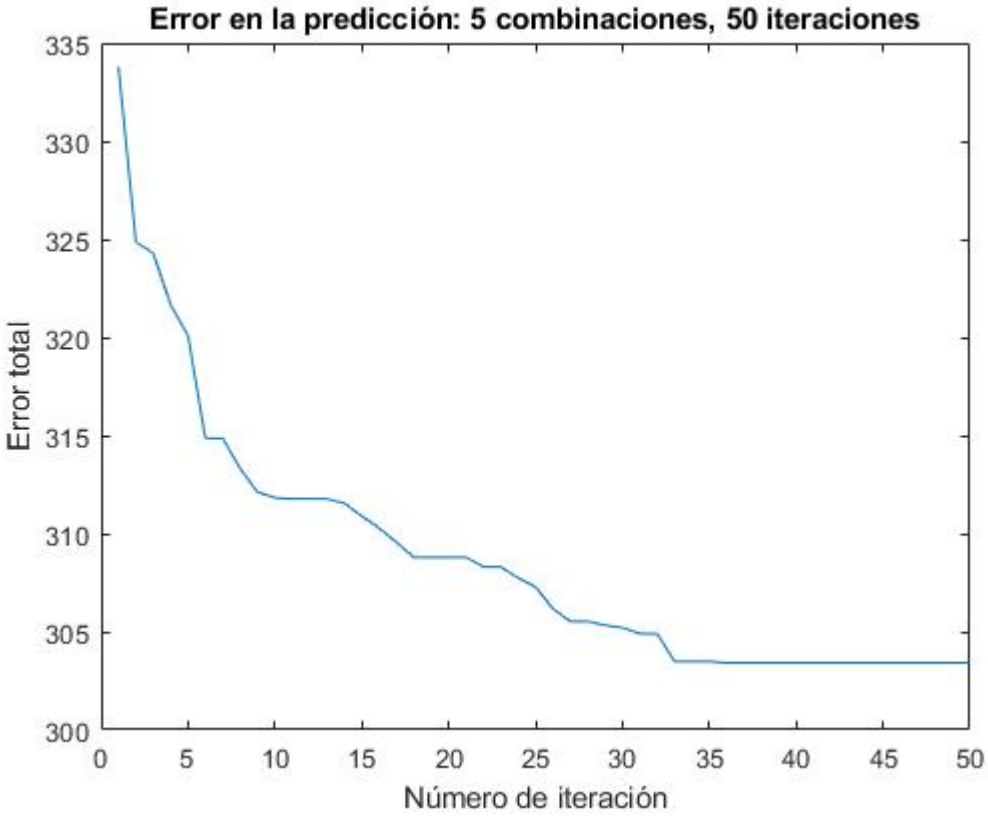


Figura 5-5 Evolución del error dependiente de las iteraciones, hasta cincuenta.

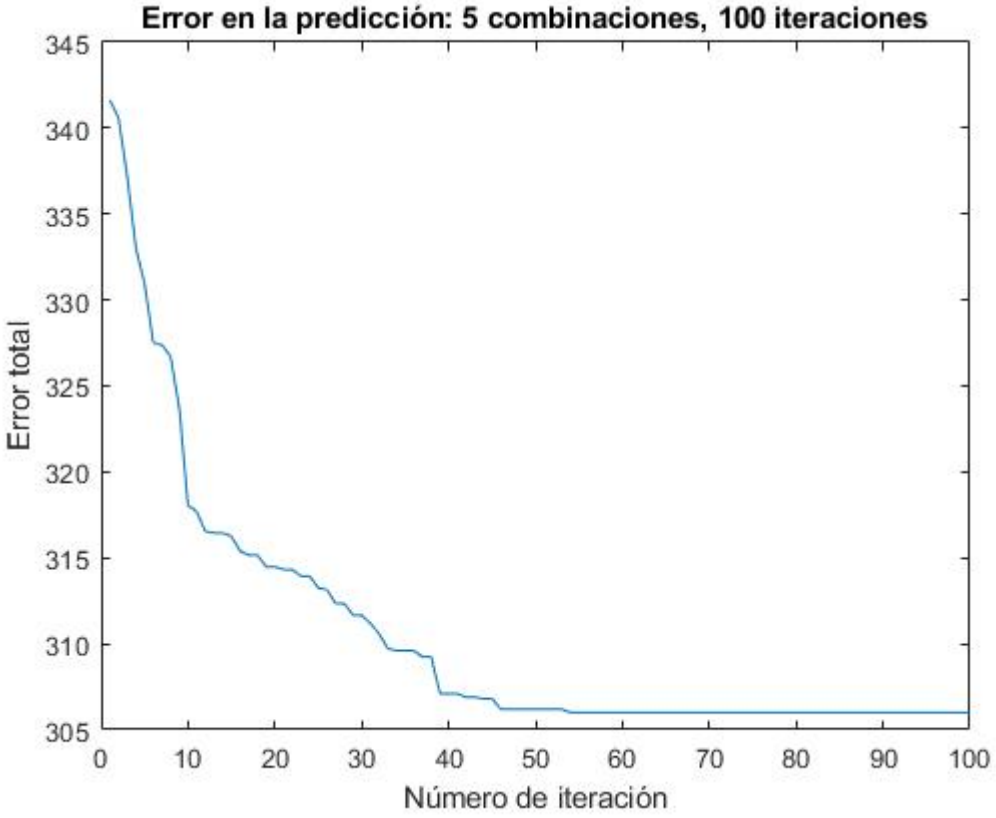


Figura 5-6 Evolución del error dependiente de las iteraciones, hasta cien.

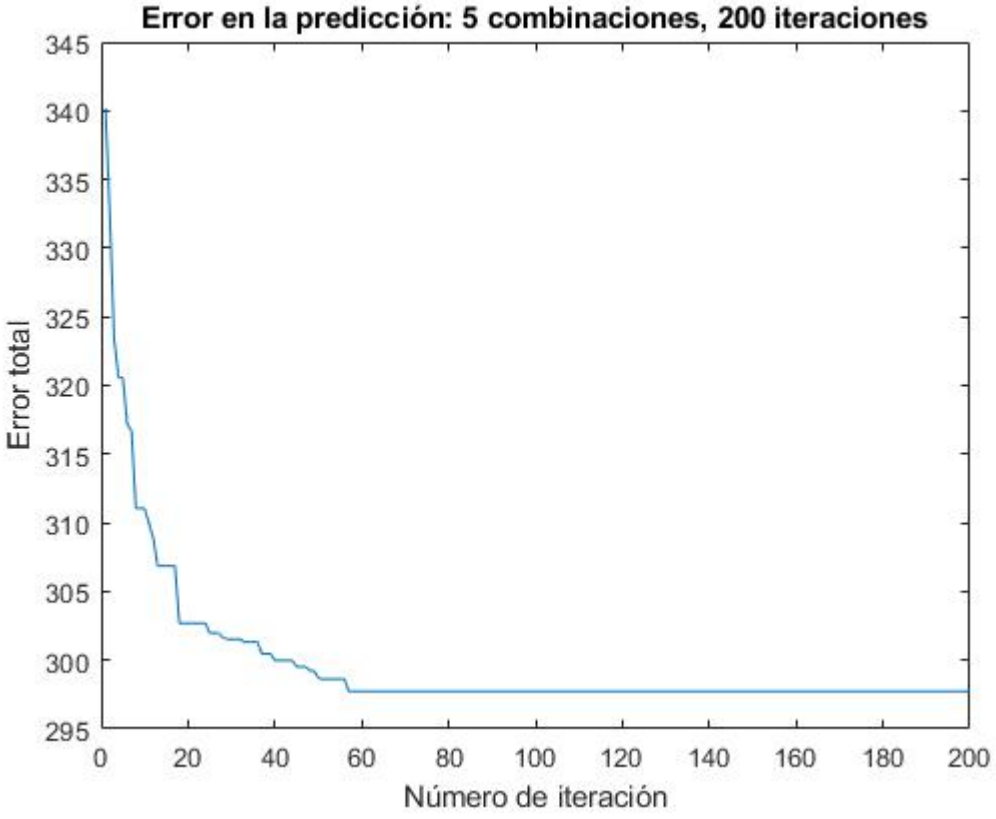


Figura 5-7 Evolución del error dependiente de las iteraciones, hasta doscientas.

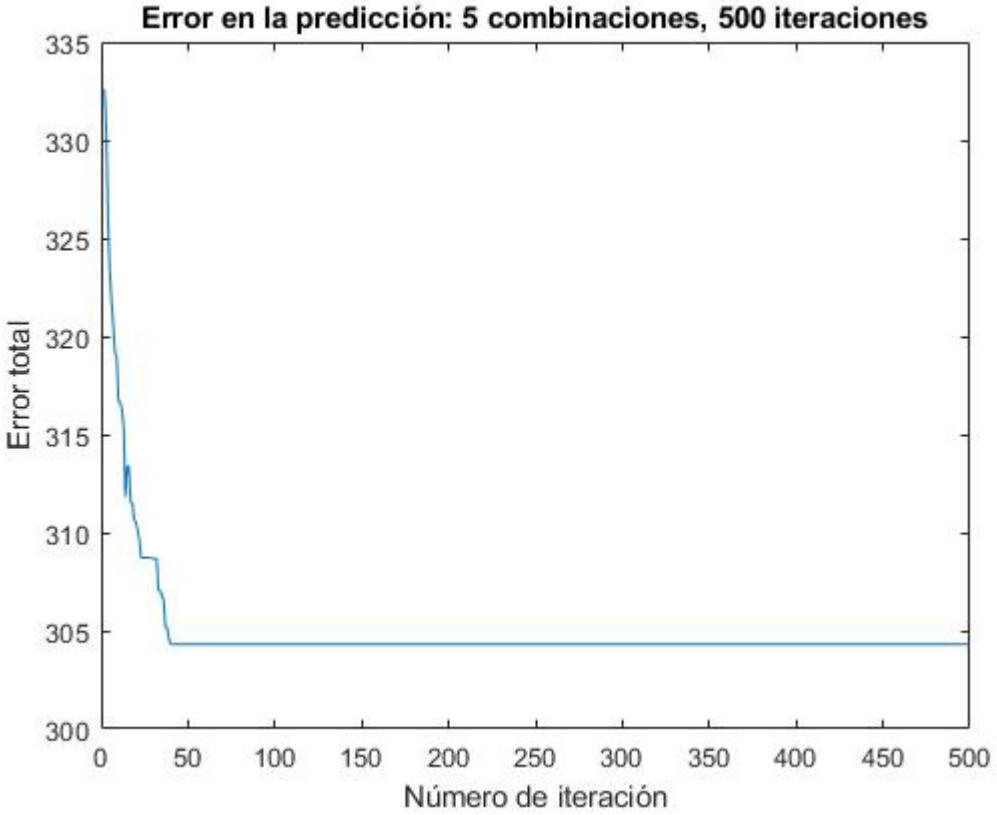


Figura 5-8 Evolución del error dependiente de las iteraciones, hasta quinientas.

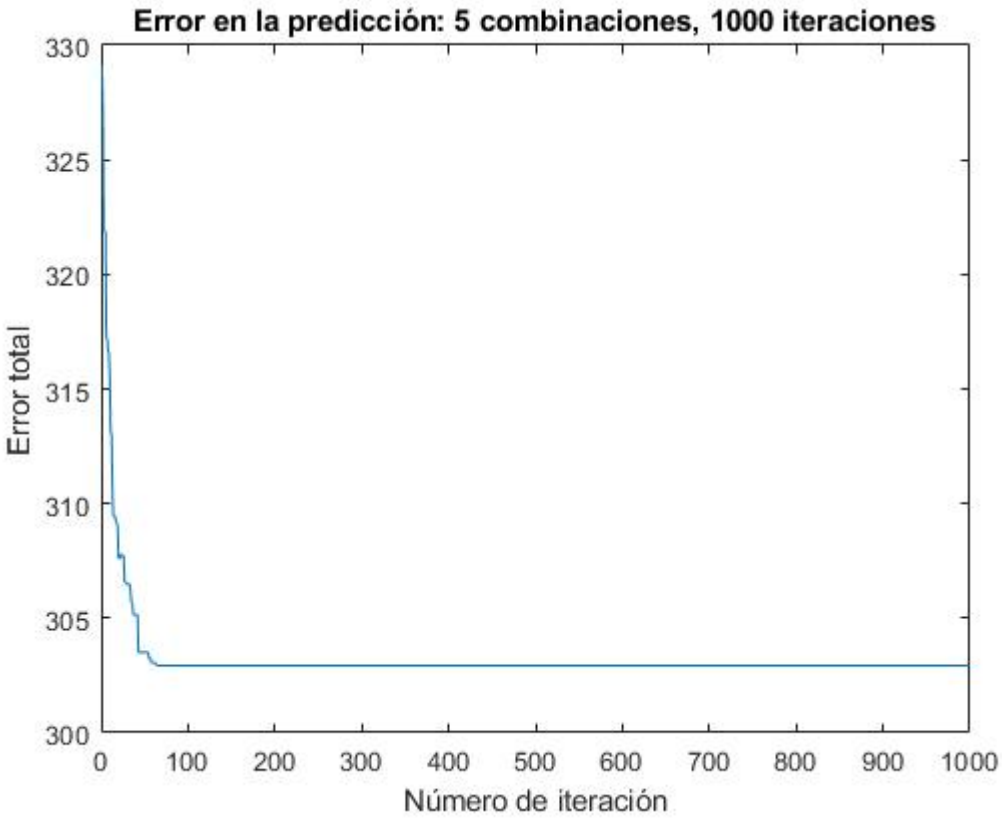


Figura 5-9 Evolución del error dependiente de las iteraciones, hasta mil.

Gracias a estas representaciones gráficas, es fácil comprobar cómo en todos los casos, a partir de entre la iteración número 40 y la número 60, el error se asienta. Esto no quiere decir que se haya alcanzado el error mínimo, sino que la mejora del error desde ese punto es pequeña o nula. Con esto podemos intuir que la búsqueda de la máxima eficiencia requerirá del aumento del número de combinaciones, en mayor medida que el número de iteraciones, llegados a este punto.

### 5.1.3 Paralelización

Como se mencionaba en el capítulo 3, los bucles de computación de los conjuntos de combinaciones se han paralelizado. Aquí podemos ver la comparación en tiempo de ejecución de algunos casos entre el proceso en serie y el proceso paralelizado.

| Combinaciones | Iteraciones | Tiempos de ejecución | | |
|---|---|---|---|---|
| | | En serie | En paralelo | % Variación |
| 5 | 1 | 7.749 | 2.167 | -72.04 |
| 10 | 1 | 14.987 | 4.015 | -73.21 |
| 20 | 1 | 31.714 | 6.600 | -79.19 |
| 50 | 1 | 73.749 | 15.016 | -79.64 |
| 5 | 5 | 27.061 | 6.318 | -76.65 |
| 5 | 10 | 58.814 | 13.111 | -77.70 |
| 5 | 20 | 106.113 | 25.235 | -76.22 |
| 5 | 50 | 273.697 | 56.150 | -79.54 |

Tabla 5-3 Tiempos de ejecución en serie y en paralelo.

El éxito de la paralelización del código es evidente, acortando entre un 70 y un 80 por ciento el tiempo de ejecución en todos los casos.

### 5.1.4 Optimización Direct Weight

En el módulo destinado a la optimización Direct Weight, hay dos elementos concretos que pueden retocarse para comprobar de qué forma se minimiza el tiempo de ejecución y el error obtenido. Estos son los que siguen a continuación:

#### 5.1.4.1 Función quadprog

En un primer momento, en el módulo al que nos referimos se hacía uso del comando “quadprog”, un solver para funciones objetivo cuadráticas con restricciones lineales. Este es el problema típico que se encarga de resolver:

$$\min_x \frac{1}{2} x^T Hx + f^T x \text{ , tal que } \begin{cases} A \cdot x \le b, \\ Aeq \cdot x = beq, \\ lb \le x \le ub. \end{cases}$$

La función “quadprog” se llama de la siguiente forma:

```
x = quadprog(H,f,A,b,Aeq,beq,lb,ub)
```

Donde la llamada para nuestro problema especifico será:

```
x = quadprog(2*eye(num_pun),[],[],[],Aeq,beq)
```

Es un comando de gran utilidad para resolver problemas de programación cuadrática. Sin embargo, en nuestro caso particular contamos con que $H$ es definida positiva y únicamente existe una restricción, que es de igualdad. Este es un caso especial del campo más general de la optimización convexa, y es posible simplificar el procedimiento de resolución. En dicho caso, la funcionalidad completa de”quadprog” no es necesaria, y ciertas entradas de la función no son útiles. Para acelerar este proceso de búsqueda del mínimo, se propone cambiar el comando haciendo uso de multiplicadores de Lagrange y buscando el límite del Lagrangiano. De este modo se puede asegurar que el problema descrito:

Minimizar

$$\frac{1}{2} x^T Hx + f^T x$$

Sujeto únicamente a

$$Aeq \cdot x = beq$$

Tiene su solución dada por el sistema lineal:

$$\begin{bmatrix} H & Aeq^T \\ Aeq & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} -f \\ beq \end{bmatrix}$$

Donde $\lambda$ es un set de multiplicadores de Lagrange que aparecen en la solución junto a $x$.

Para comprobar el correcto funcionamiento de esta modificación, se realizó una prueba. Se eligió una combinación concreta, y se computó mediante los dos métodos distintos, el Direct Weight original y el modificado. El código se puede consultar en el apéndice. La diferencia máxima entre los valores obtenidos de una forma y otra, fue de $2.874 \cdot 10^{-10}$. Por tanto se puede asegurar que la modificación es correcta.

En cuanto al tiempo de ejecución, en la prueba anterior también se cronometró cada computación, dando un resultado de 2.0138 segundos usando el comando "quadprog" y 0.60611 segundos usando el sistema de ecuaciones. Esta diferencia de tiempos para la computación de una sola combinación, demuestra que la mejora es importante, ya que por ejemplo, con 50 combinaciones y 50 iteraciones se computarán 5050 combinaciones, y el tiempo ahorrado teórico podría ascender a 7040 segundos, cerca de dos horas (en caso de no estar la tarea paralelizada). A continuación se mostrarán los resultados de algunos ejemplos en la tabla 5-4, con el proceso paralelizado.

| Combinaciones | Iteraciones | Tiempo de ejecución (s) | | |
|---|---|---|---|---|
| | | Con "quadprog" | Sin "quadprog" | %Variación |
| 20 | 10 | 190.659 | 88.851 | -53.4% |
| 50 | 20 | 809.891 | 571.744 | -29.4% |
| 50 | 50 | 2140.686 | 1290.174 | -39.73% |
| 100 | 50 | 4175.319 | 2493.425 | -40.28% |

Tabla 5-4 Tiempos de ejecución usando comando "quadprog" y método alternativo

La mejora en la disminución del tiempo de ejecución es evidente, entre 30 y 50% aproximadamente. Esto permitirá computar un mayor número de combinaciones en un menor tiempo.

#### 5.1.4.2 Número de vecinos

Hasta ahora las pruebas de los apartados anteriores se han realizado con un número de puntos vecinos de 250, con la intención de ejecutar la función Direct Weight de forma rápida, pero dando un resultado aceptable. Ahora se intentará identificar como influye este parámetro en los resultados. Con ese fin, se computará un mismo conjunto de 50 combinaciones y se evaluará su error variando el número de vecinos.En la tabla 5-5 se exponen los resultados:

| Nº de vecinos | Error | Tiempo ejecución |
|---|---|---|
| 100 | 356.75 | 23.128 |
| 200 | 340.55 | 48.42 |
| 300 | 328.64 | 5.007 |
| 400 | 320.34 | 195.64 |
| 500 | 309.19 | 292.37 |
| 600 | 307.1 | 394.17 |
| 700 | 305.46 | 579.9 |
| 800 | 301.48 | 758.68 |
| 900 | 300.36 | 1048.9 |
| 1000 | 302 | 1300.1 |
| 1100 | 300.97 | 1359.3 |
| 1200 | 301.3 | 1365.9 |
| 1300 | 303.06 | 1385.3 |
| 1400 | 304.02 | 1435.5 |
| 1500 | 305.12 | 1489.6 |

Tabla 5-5 Error cometido y tiempos de ejecución segúnel número de puntos vecinos

Y la siguente figura puede ser útil para su interpretación.

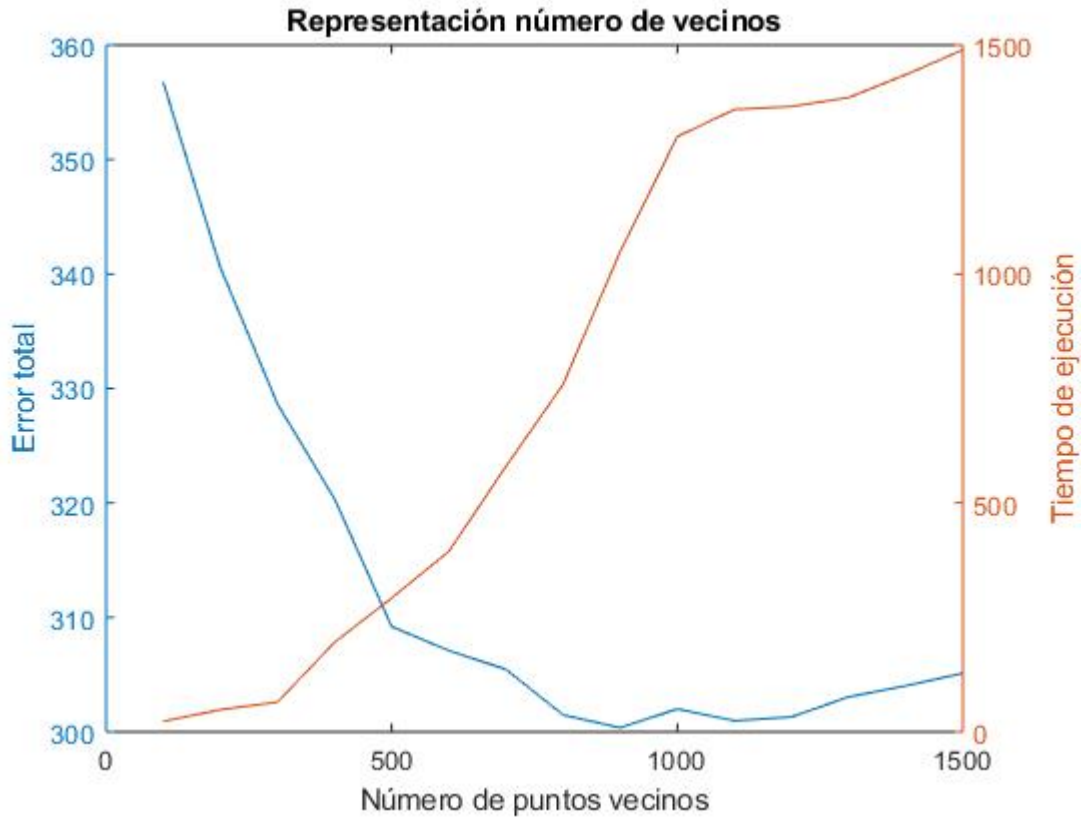


Figura 5-10 Evolución del error y el tiempo de ejecución según el número de vecinos.

Es importante señalar que al variar este parámetro no estamos mejorando la eficiencia al encontrar la mejor combinación de indicadores técnicos, se mejora únicamente el resultado de la predicción del precio. Por tanto, lo ideal es sacrificar algo de precisón en esta predicción para que el algoritmo se ejecute de manera más rápida y obtenga la mejor combinación posible en el menor tiempo posible. Una vez obtenida esta combinación, se podrá computar con un mayor número de puntos vecinos para minimizar el error en la predicción. Este número de vecinos estará entre los 500 y los 1000, ya que se puede comprobar en la figura 5-10 como para mayores números no se logra una disminución del error.

## 5.2 Resultados finales

Una vez analizados los elementos fundamentales que influyen en el desarrollo del código, se presentan los resultados finales, buscando la mejor combinación de parametros y elementos que aporten una mayor eficiencia al algoritmo. En estos resultados se incluirán las combinaciones que se seleccionan en cada caso, comprobando a partir de qué momento aproximandamente se comienza a seleccionar siempre la mejor.

| Combinaciones | Iteraciones | Error | Tiempo (s) | Combinación elegida |
|---|---|---|---|---|
| 50 | 50 | 301.59 | 517.338 | {1,2,6,7,8,9,14,16,17,18,22,24,26,33,13,15,35,40,30,5,39} |
| 50 | 100 | 300.29 | 998.466 | {4,16,18,19,20,24,26,28,31,37,15,29} |
| 100 | 50 | 297.51 | 1044.656 | {9,15,1,18,19,20,26,33,40,7,22,13,35,30,23,3,12} |
| 100 | 100 | 297.21 | 2046.706 | {12,15,16,19,20,23,26,33,35,18,22,40,6,13,3,30} |
| 200 | 100 | 297.21 | 4133.031 | {6,12,15,16,18,20,22,26,30,33,40,19,35,23,13,3} |
| 500 | 500 | 297.21 | 49839.169 | {6,12,16,18,20,22,30,33,35,40,19,26,3,13,15,23} |
| 100 | 80 | 297.96 | 1642.020 | {5,13,15,16,19,20,22,26,33,37,40,18,35,39,14,34} |
| 100 | 90 | 292.14 | 1732.242 | {2,3,5,6,7,8,16,17,21,26,27,32,34,35,14,38,30,1,40,39} |
| 1000 | 1000 | 297.21 | 199074.797 | {12,18,20,22,30,33,35,40,19,26,3,13,23,15,16,6} |
| 10000 | 60 | 291.42 | 116227.969 | {2,10,16,17,20,32,26,9,15,24,34} |
| 10000 | 70 | 296.82 | 128387.737 | {2,3,6,8,16,17,29,32,35,26,7,10,21,34,5} |

Tabla 5-6 Mejores combinaciones con errores y tiempos de ejecución.

En un principio parecía que el error mínimo global se había alcanzado con 100 combinaciones y 100 iteraciones, ya que el valor resultante era el mismo que con la prueba de duración mayor que se ha realizado, de 1000 combinaciones y 1000 iteraciones, y siempre con los mismos indicadores elegidos. Sin embargo, al realizar otras pruebas, se consiguió disminuir este error en un menor tiempo, con 100 combinaciones y 90 iteraciones. Esto es debido a la aleatoriedad del algoritmo, y aunque cuantas más combinaciones e iteraciones usemos mayor probabilidad de encontrar el mínimo global, esto no es algo seguro.

Por tanto se continuó realizando pruebas, con más combinaciones y menos iteraciones, tal y como se había concluido (a partir de 50 ó 60 iteraciones se asentaba el error en pruebas más cortas). Esto llevó a lograr un nuevo mínimo de 291.42, con 10000 combinaciones y 60 iteraciones. La combinación seleccionada era de sólo10 indicadores, mientras que los anteriores mínimos eran de 16 ó 20. Esto condujo a pensar que el motivo podría ser que ciertos mínimos locales están demasiado separados unos de otros. Es decir, si el algoritmo comienza a identificar un posible mínimo de 16 indicadores, al ser su metodología extraer y añadir indicadores de uno en uno, probablemente no haya una mejor opción con 17 ó 18 indicadores. Sin embargo, hay una combinación con 10 indicadores que es mejor, aunque al haber seguido el "camino" de otro mínimo local, el algoritmo no accederá a ella.

Esto conlleva la necesidad de ejecutar el algoritmo varias veces para comprobar cuál es la mejor combinación de cada "camino", que aporten los mínimos locales diferentes, y decidir entre ellas la que aporta el mínimo global. Esto es posible automatizarlo con una mejora, donde en un principio no se parte con un set de combinaciones homogéneas en cuanto a número de indicadores siguiendo estrictamente el algoritmo la mitad de los disponibles, sino que cada combinación contenga un número distinto de indicadores.

Se ha realizado esta pequeña modificación con buenos resultados. En concreto, si hay M combinaciones, se ha comenzado con M/3 combinaciones de 10 indicadores, M/3 combinaciones de 20 indicadores y M/3 combinaciones de 30 indicadores, aproximadamente. El mínimo error conseguido es igual que el anterior, pero han sido necesarias bastantes menos combinaciones e iteraciones, y por tanto mucho menos tiempo de ejecución. Estos nuevos resultados quedan recogidos en la tabla 5-7, al igual que las gráficas de las pruebas

más cortas y más largas realizadas que obtengan este último resultado (figuras 5-11 y 5-12, respectivamente).

| Combinaciones | Iteraciones | Error | Tiempo(s) | Combinación elegida |
|---|---|---|---|---|
| 500 | 60 | 291.42 | 453.269 | {2,9,10,16,17,26,32,15,20,24,34} |
| 500 | 60 | 297.21 | 5515.089 | {3,13,15,16,19,22,23,26,33,35,18,20,12,40,6,30} |
| 10000 | 70 | 291.42 | 105650.595 | {2,16,20,26,17,32,9,10,15,34,24} |
| 5000 | 60 | 291.42 | 47342.159 | {2,15,16,17,20,26,32,10,9,34,24} |
| 5000 | 60 | 291.42 | 46631.909 | {2,16,17,20,26,9,32,10,15,34,24} |
| 4000 | 50 | 291.42 | 34858.810 | {9,16,17,20,26,2,32,10,24,15,34} |
| 3000 | 50 | 291.42 | 24762.196 | {2,16,17,26,20,15,9,10,32,24,34} |
| 2500 | 50 | 291.42 | 21615.774 | {16,17,15,20,9,26,10,34,24,32,2} |

Tabla 5-7 Mejores combinaciones con errorres y tiempos de ejecución (con modificación)

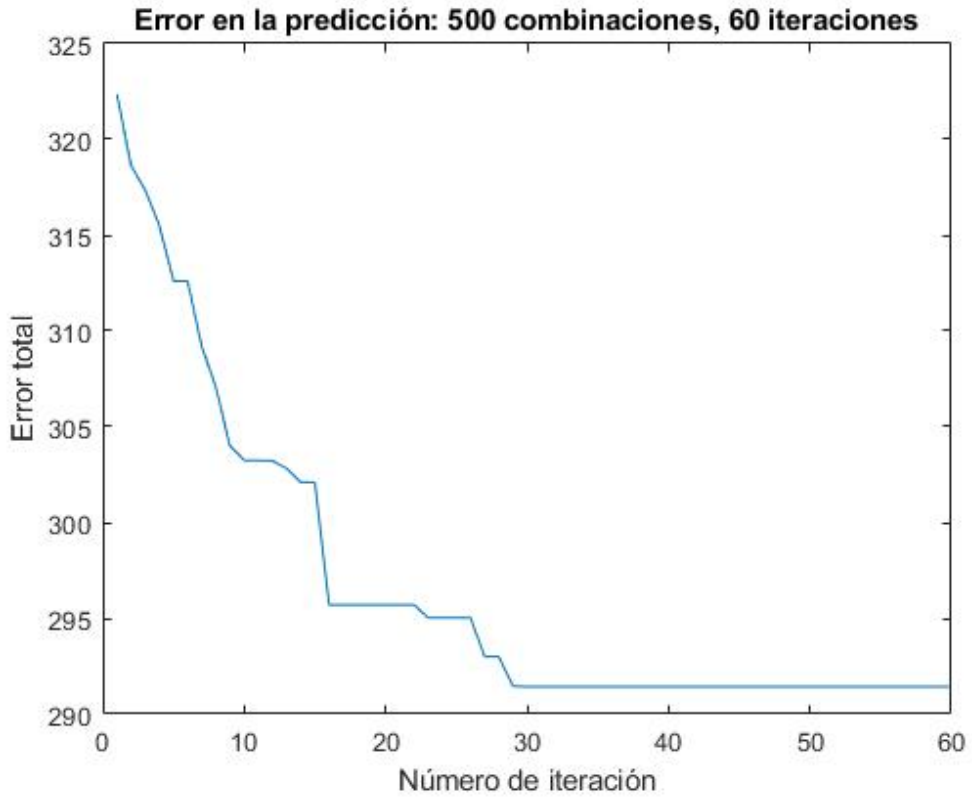


Figura 5-11 Evolución del error con 500 combinaciones y 60 iteraciones

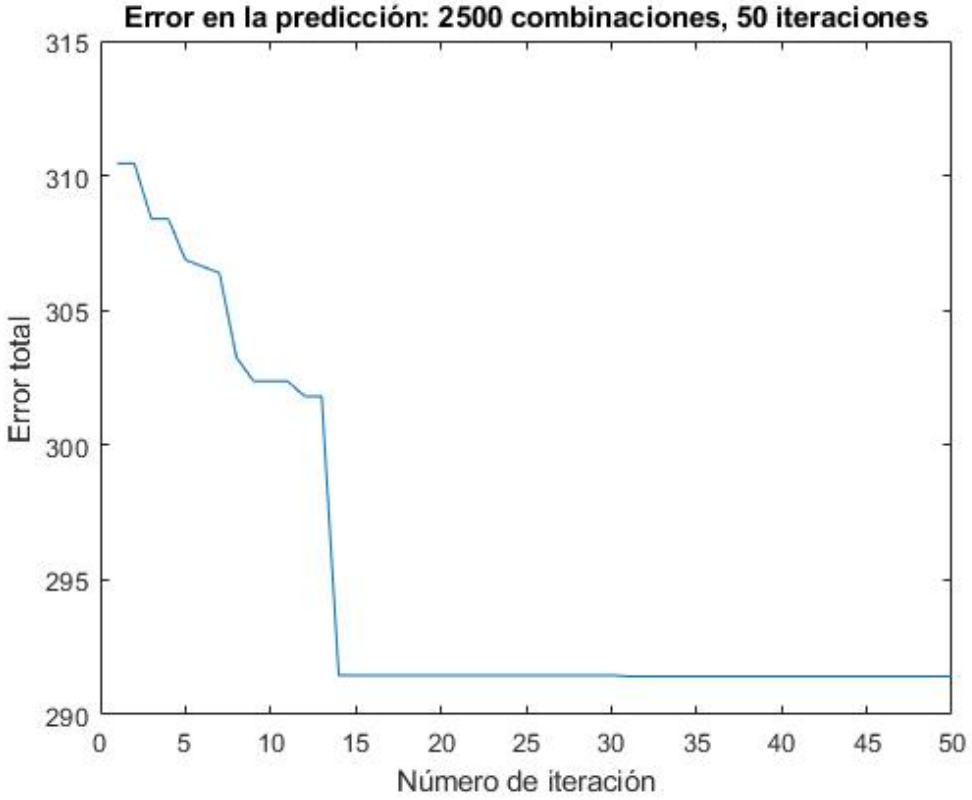


Figura 5-12 Evolución del error con 2500 combinaciones y 50 iteraciones

## 5.3 Conclusiones

La combinación seleccionada en este caso cuenta con once indicadores diferentes. Como se puede comprobar, la tipología de este grupo de indicadores es muy variada. A continuación se indican cuales son concretamente.

2: SMVAG(100)

9: Slow %D (5-day)

10: Momentum (10-day)

15: A/D oscillator

16: MA (5)

17: MA (10)

20: OSCP

24: Z-score

26: EMAVG (5) on Close

32: Kairi (Simple,25)

34: JKHL Index

Además, se pueden consultar sus fórmulas y una breve descripción de cada uno de estos indicadores elegidos en la tabla 2-1.

Para demostrar la utilidad y el correcto funcionamiento del algoritmo, obtendremos el error que se produciría si seleccionáramos todos los 40 indicadores que disponemos y realizamos la predicción con ellos. En este caso el error es de 354.089. Por tanto, al ser el error obtenido de nuestra mejor combinación de 291.42, podemos decir que el porcentaje de mejora es del 17.7%.

Se puede también analizar un error relativo con este error total y la media de valores que tiene el precio del índice en este conjunto de test. Esta media será de 26309, por lo que el error relativo será de un 1.1077%.

Se concluyó en el apartado 6.1.3 que el número de puntos vecinos elegidos para la computación final de la combinación seleccionada sería de 500 a 1000 aproximadamente. Sin embargo, se comprueba que con esta combinación concreta no se mejora este error, por lo que se realizará la predicción final con el mismo número de puntos vecinos que con las pruebas, es decir, 250. La representación gráfica de la predicción, así como la del precio real, es mostrada en la figura 5-13.

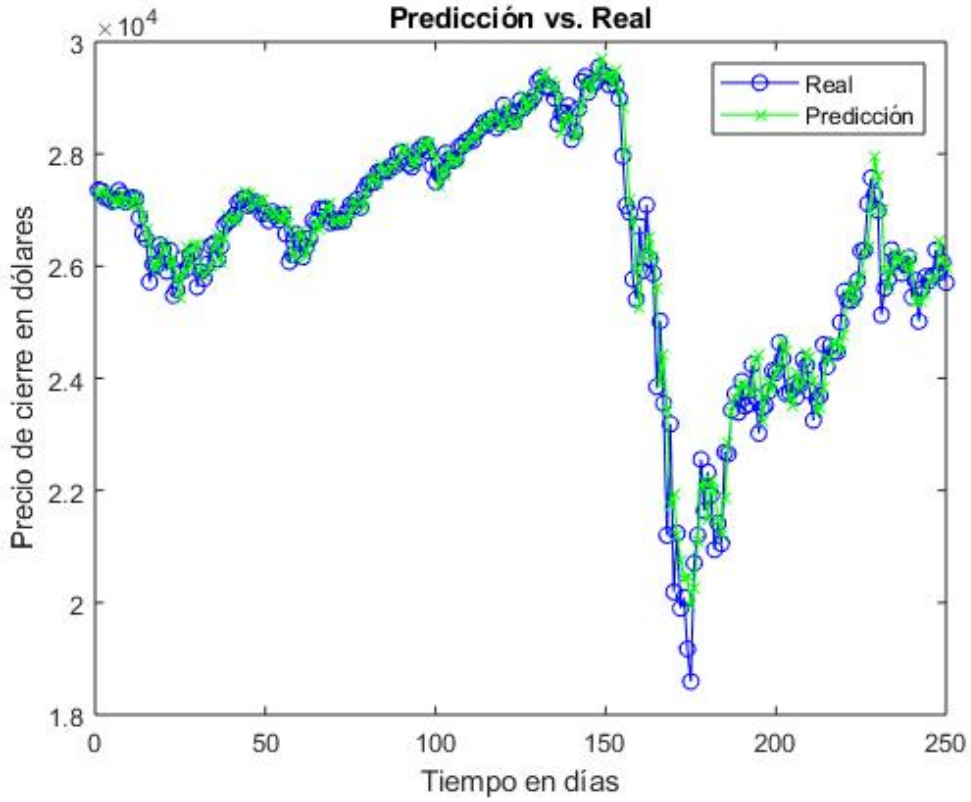


Figura 5-13 Comparativa predicción y precio real

Se amplía la zona estudiada de mayor volatilidad en la figura 5-14 , con la idea de mostrar de forma más clara el comportamiento de la predicción.

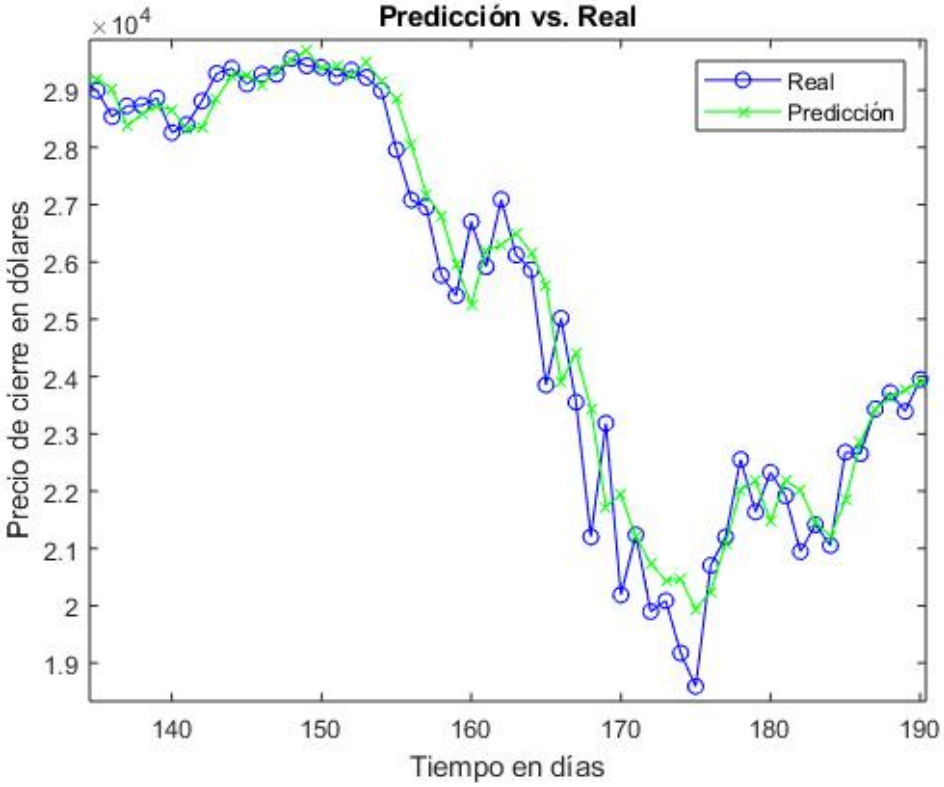


Figura 5-14 Comparativa predicción y precio real (ampliada)

# 6 PRUEBAS Y RESULTADOS: PING AN BANK

Una vez expuestos detalladamente las pruebas y resultados del algoritmo aplicado al primer conjunto de características, se procederá a un análisis del segundo conjunto. En este caso, se tratará de la acción Ping An Bank, con datos minutarios de 24 indicadores técnicos diferentes y el precio de la acción. Los datos abarcarán desde las 09:25 del 29 de diciembre de 2017 hasta las 15:00 del 15 de enero de 2019.

## 6.1 Elementos a considerar

Para llevar a cabo este análisis, se variarán los elementos de modo simliar que en el primer caso de estudio y se recogerán las figuras y tablas pertinentes. Se incluyen únicamente los elementos que proporcionarán un contraste analizable entre los resultados de las dos aplicaciones. Concretamente, se han considerado el número de combinaciones, el número de iteraciones y el número de vecinos. Algunos otros, como la paralelización o la función quadprog se obviarán, ya que se puede intuir fácilmente que proporcionalmente el ahorro de tiempo computacional será similar.

### 6.1.1 Combinaciones

La variación de combinaciones se desarrollará del mismo modo que en capítulo anterior. En la tabla 6-1 quedan expuestos los resultados obtenidos. En la figura 6-1 aparece información gráfica sobre la evolución del error dependiendo del número de combinaciones.

| Combinaciones | Iteraciones | Error total | Tiempo de ejecución (s) |
|---|---|---|---|
| 6 | 1 | 0.00013329 | 29.029 |
| 10 | 1 | 6.6908e-5 | 55.619 |
| 20 | 1 | 6.522e-5 | 99.302 |
| 50 | 1 | 6.5337e-5 | 241.840 |
| 100 | 1 | 1.3014e-5 | 438.068 |
| 200 | 1 | 6.1243e-5 | 933.744 |
| 500 | 1 | 1.7677e-6 | 2331.661 |
| 1000 | 1 | 1.8971e-6 | 4774.859 |

Tabla 6-1 Pruebas con variación en las combinaciones

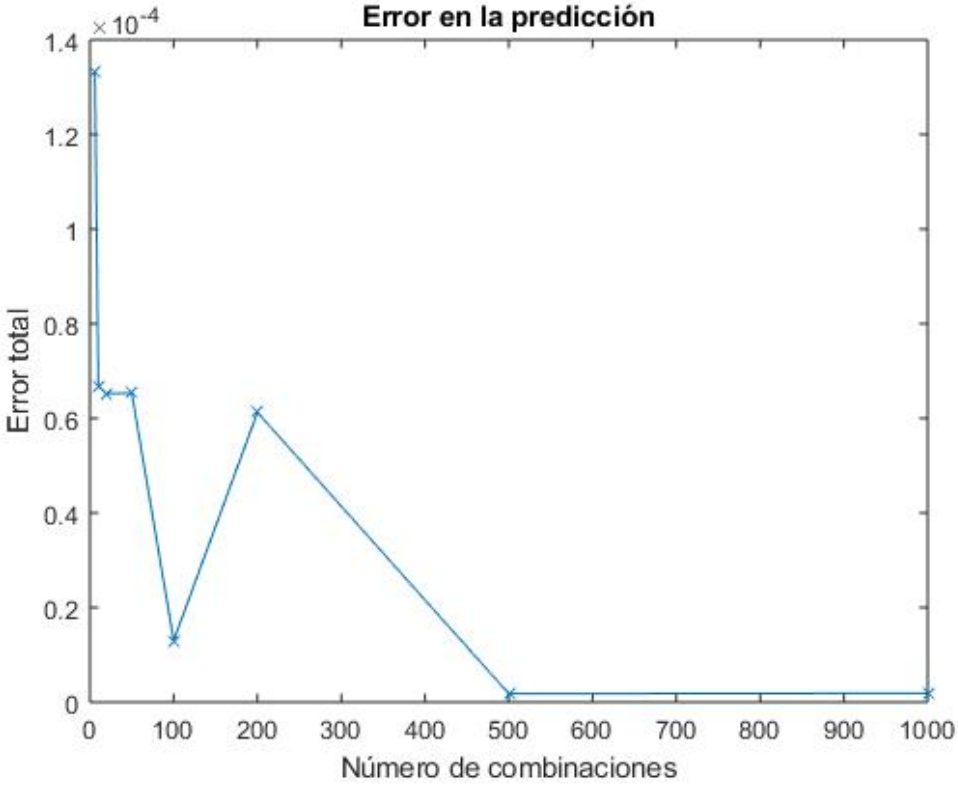


Figura 6-1 Evolución del error total respecto al número de combinaciones usado

### 6.1.2 Iteraciones

Se procede con el mismo método para analizar como repercute el cambio de número de iteraciones en el resultado del error total, los tiempos de ejecución y el número de indicadores seleccionados en la mejor combinación. Los resultados obtenidos aparecen en la tabla 6-2 y en las figuras 6-2 a 6-9.

| Combinaciones | Iteraciones | Error total | Tiempo de ejecución | Nº de indicadores |
|---|---|---|---|---|
| 6 | 5 | 7.4619e-5 | 124.081 | 16 |
| 6 | 10 | 6.1572e-5 | 379.790 | 17 |
| 6 | 20 | 5.3975e-5 | 708.635 | 17 |
| 6 | 50 | 4.9256e-5 | 1390.704 | 17 |
| 6 | 100 | 4.835-5 | 288.924 | 10 |
| 6 | 200 | 4.9389e-5 | 3300.833 | 9 |
| 6 | 500 | 4.9388e-5 | 7360.311 | 9 |
| 6 | 1000 | 4.9692e-5 | 16940.598 | 17 |

Tabla 6-2 Tabla de pruebas con variación en las iteraciones

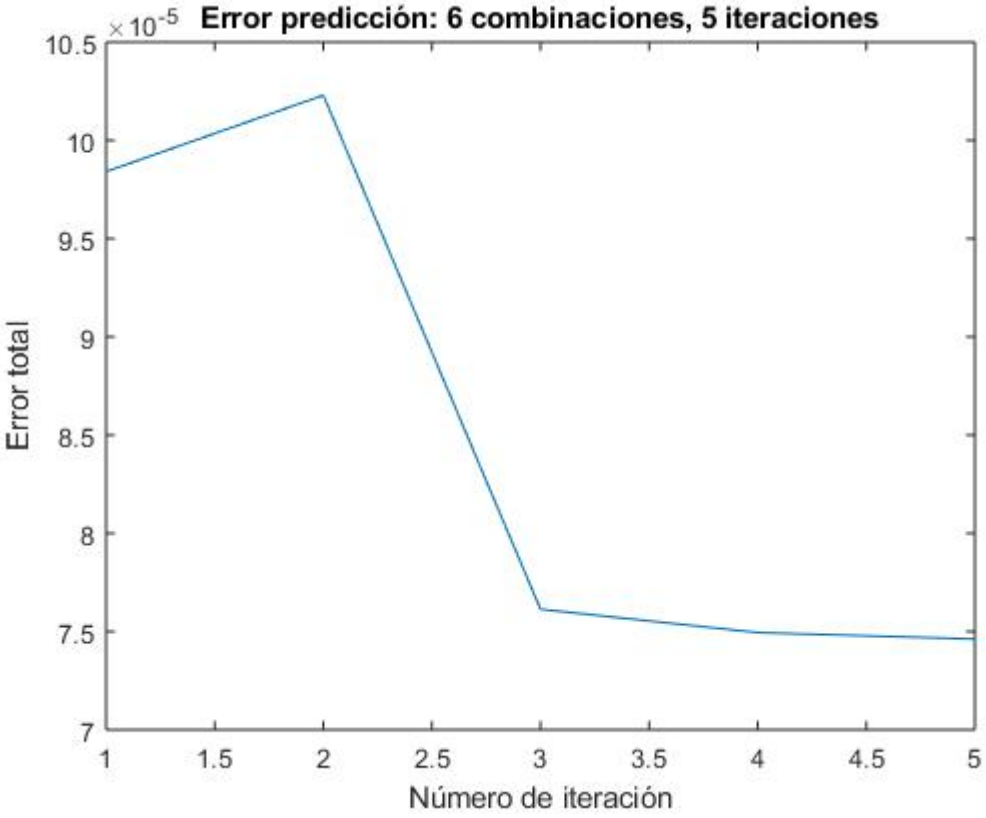


Figura 6-2 Evolución del error depediendo de las iteraciones, hasta cinco.

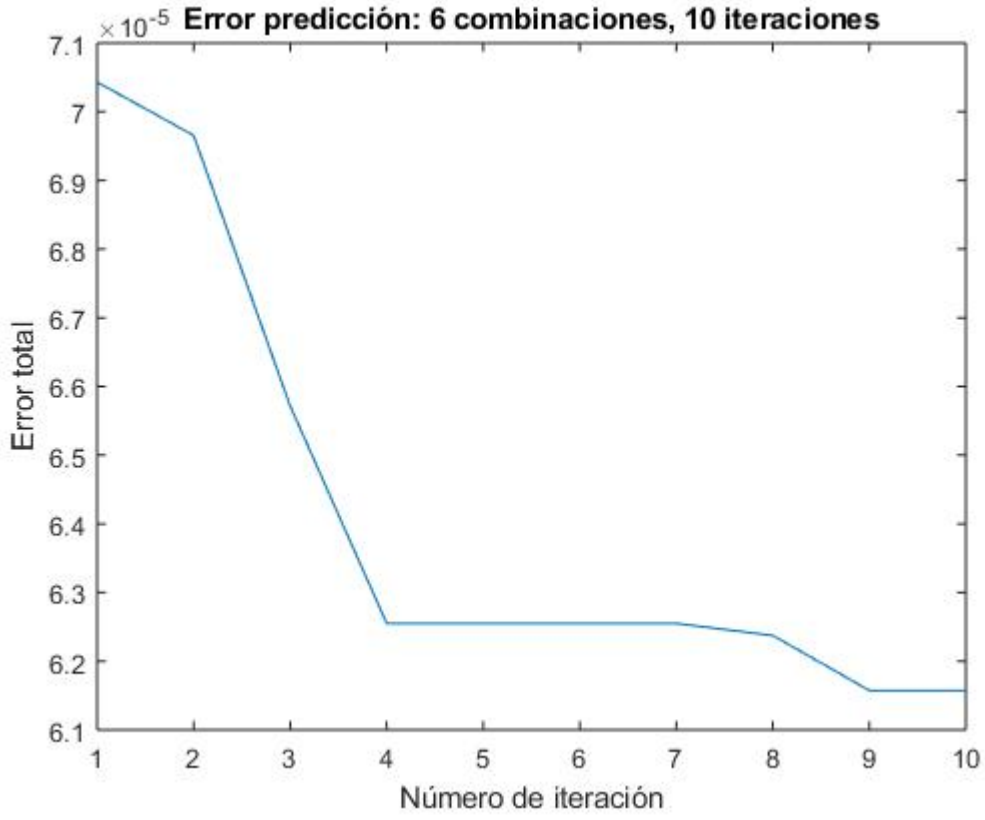


Figura 6-3 Evolución del error depediendo de las iteraciones, hasta diez.

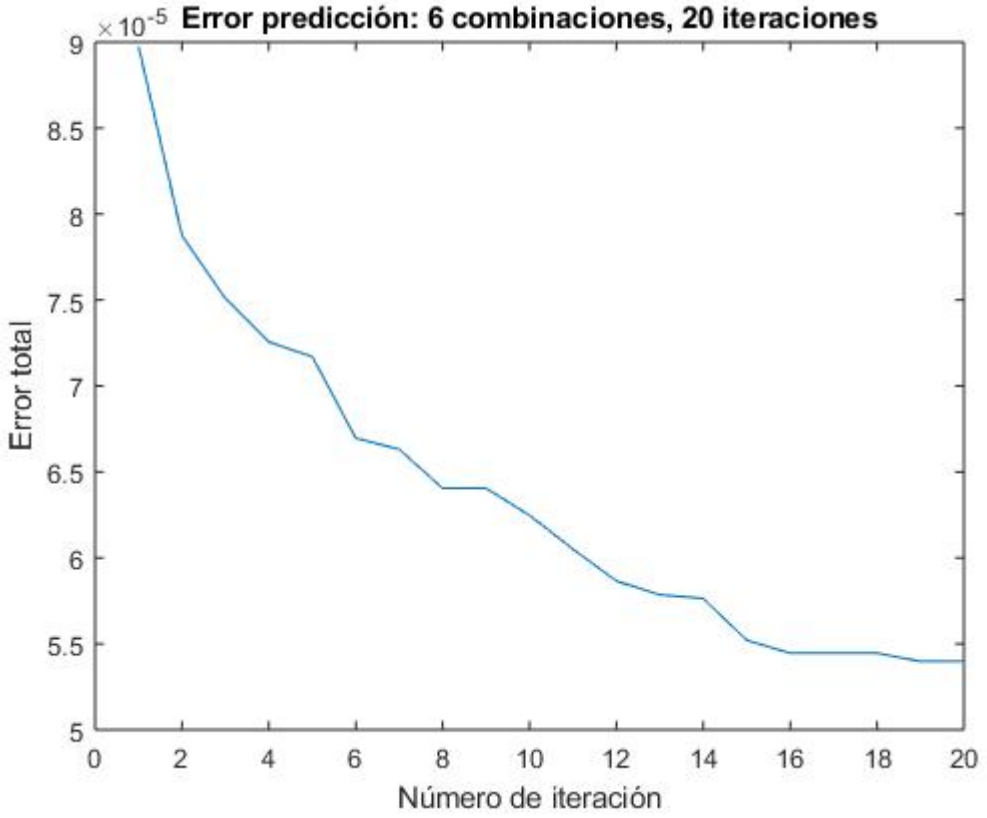


Figura 6-4 Evolución del error depediendo de las iteraciones, hasta veinte.

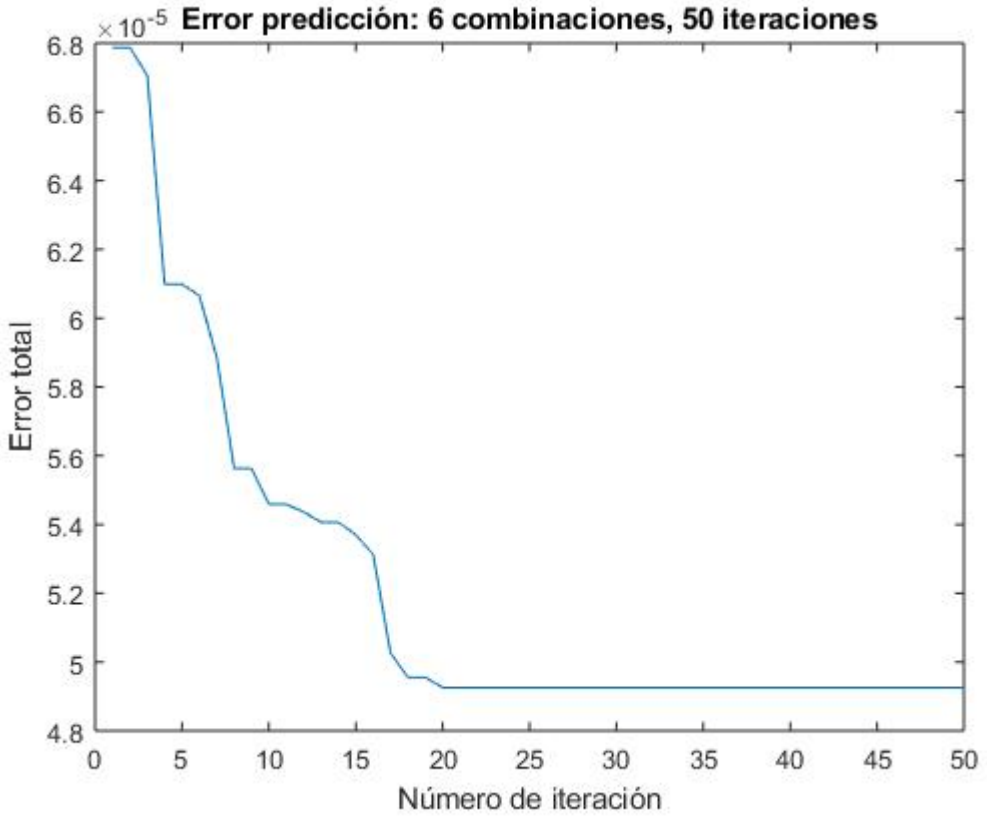


Figura 6-5 Evolución del error depediendo de las iteraciones, hasta cincuenta.

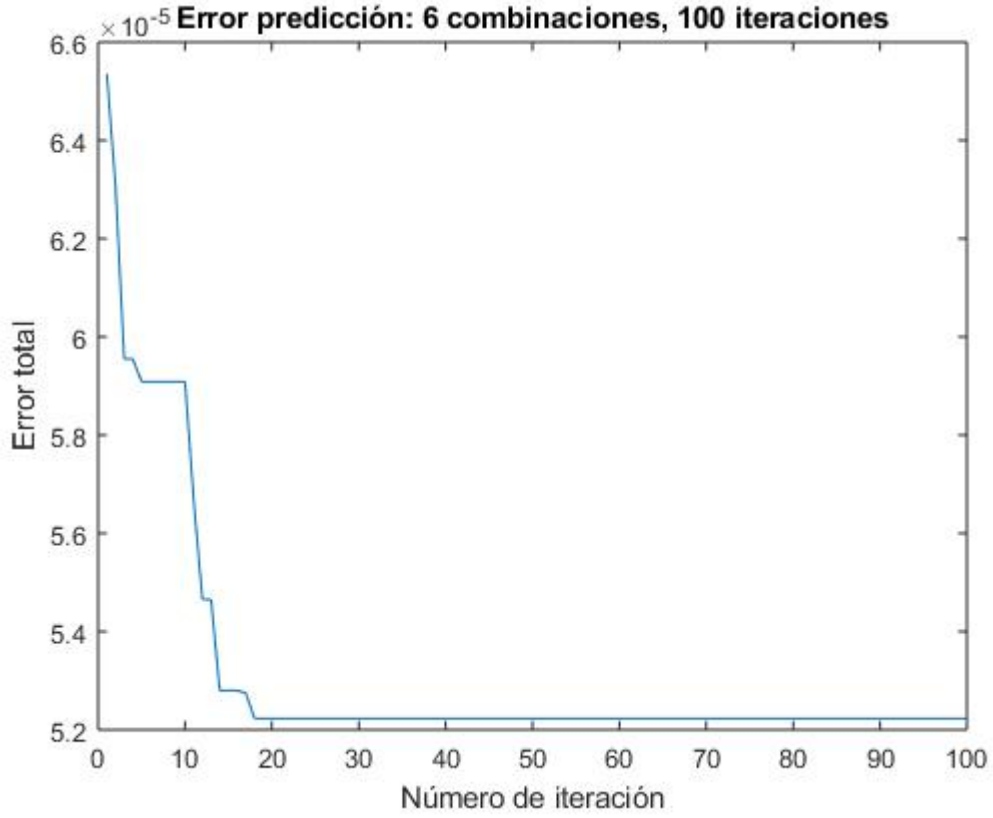


Figura 6-6 Evolución del error depediendo de las iteraciones, hasta cien.

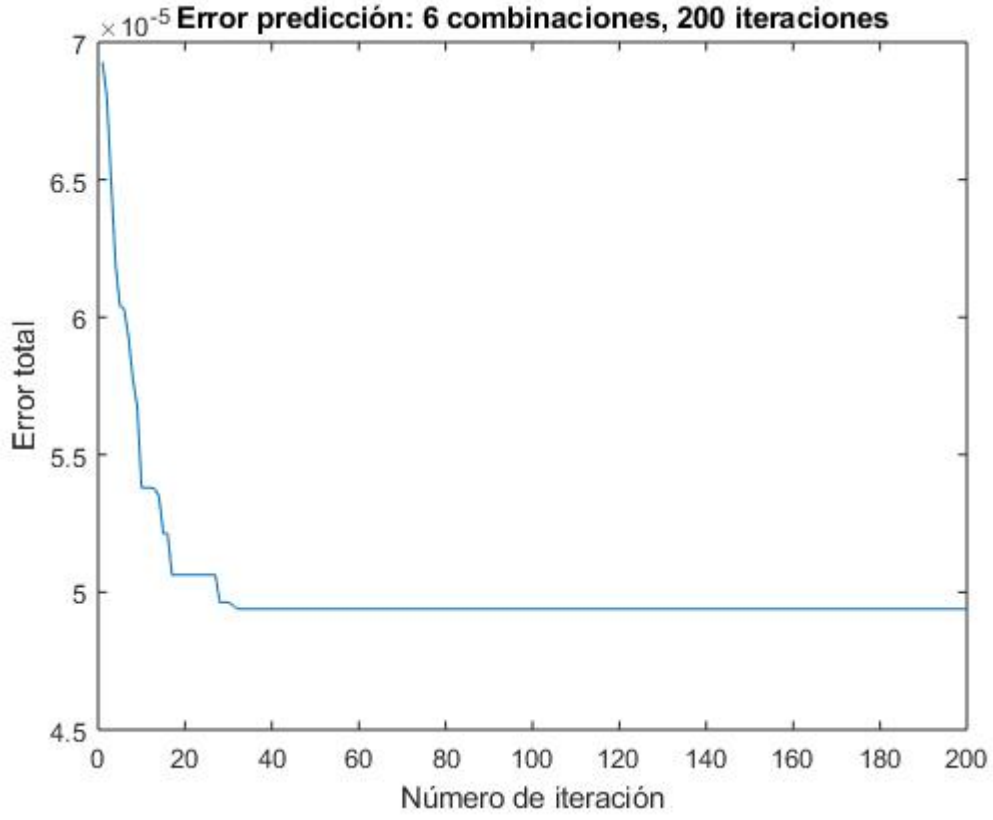


Figura 6-7 Evolución del error depediendo de las iteraciones, hasta doscientas.

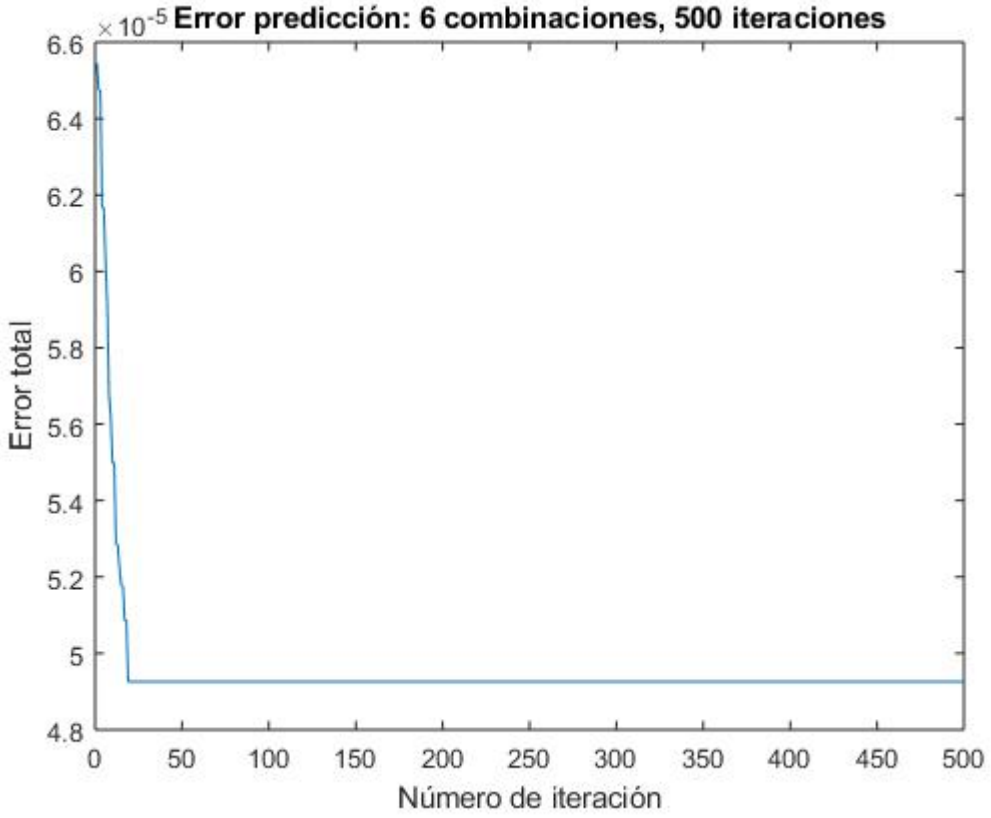


Figura 6-8 Evolución del error depediendo de las iteraciones, hasta quinientas.

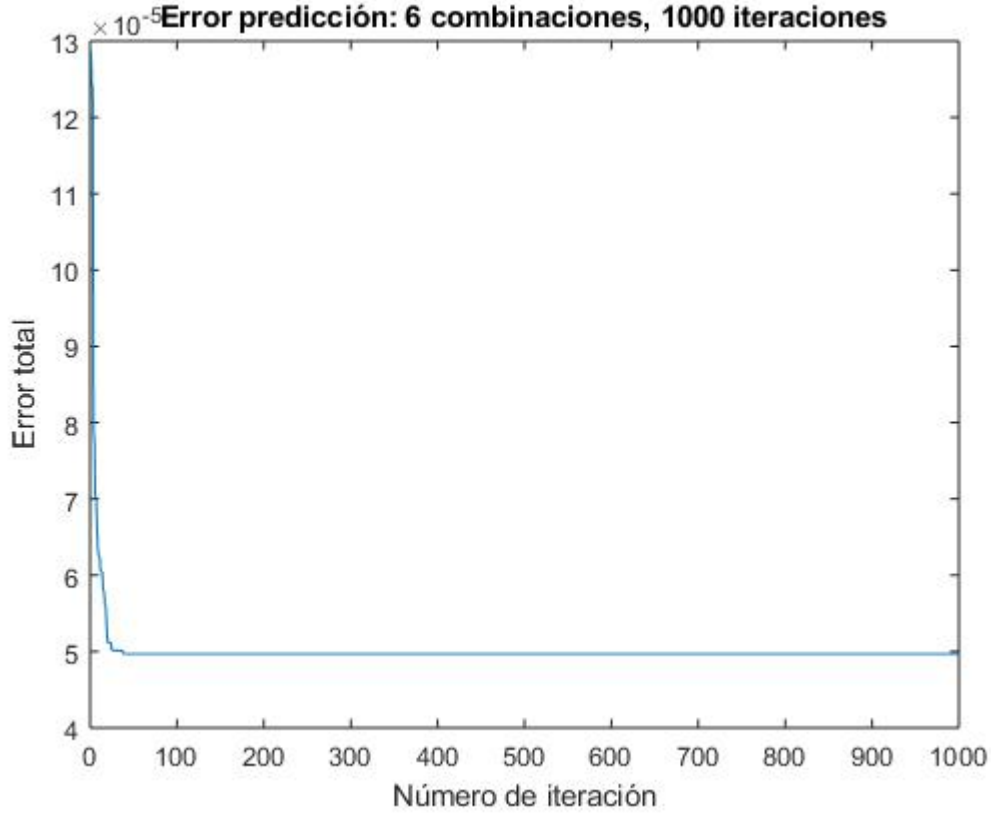


Figura 6-9 Evolución del error depediendo de las iteraciones, hasta mil.

Estas figuras son muy útiles para identificar cómo el error obtenido se asienta antes de la iteración número 30 en todas ellas. Por tanto, para obtener los resultados finales será más eficiente usar iteraciones de valores entre 20 y 30.

### 6.1.3 Número de vecinos

Es interesante comprobar como afectará la variación del número de puntos vecinos dentro de la optimización Direct Weight en este nuevo ejemplo. Para ello se computará un mismo set de combinaciones alterando únicamente este parámetro y se comparan sus errores en la tabla 6-3.

| Nº de vecinos | Error | Tiempo ejecución |
|---|---|---|
| 100 | 8.4681e-5 | 127.58 |
| 200 | 6.2479e-5 | 148.64 |
| 300 | 5.8968e-5 | 210.97 |
| 400 | 5.6568e-5 | 526.27 |
| 500 | 5.5305e-5 | 752.39 |
| 600 | 5.5125e-5 | 896.77 |
| 700 | 5.4959e-5 | 779.73 |
| 800 | 5.5356e-5 | 703.04 |
| 900 | 5.5258e-5 | 1089.22 |
| 1000 | 5.5175e-5 | 1251.01 |
| 1100 | 5.4843e-5 | 1504.73 |
| 1200 | 5.5165e-5 | 1978.42 |
| 1300 | 5.594e-5 | 2414.97 |
| 1400 | 5.5644e-05 | 2875.03 |
| 1500 | 5.5343e-05 | 3030.67 |

Tabla 6-3 Pruebas con variaciones en el número de puntos vecinos

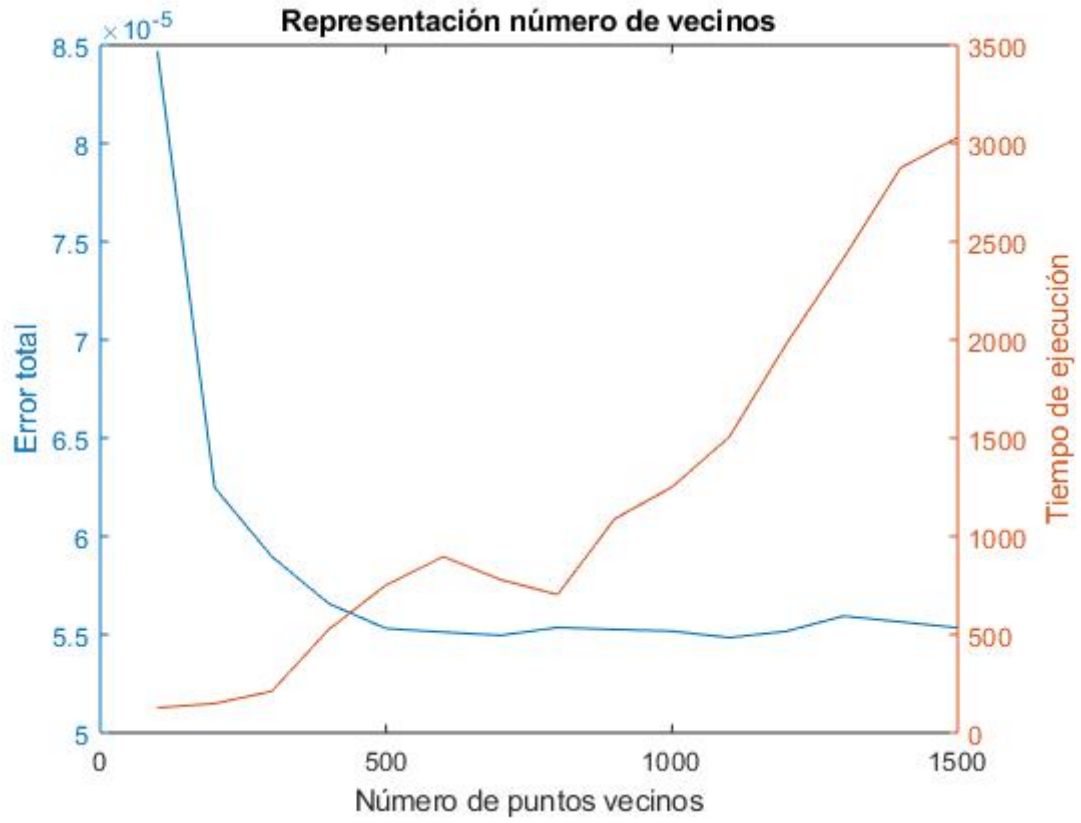


Figura 6-10 Evolución del error y el tiempo de ejecución según el número de vecinos

Como se aprecia en la figura 6-8, del mismo modo que en el caso del Dow Jones, se puede intuir que aproximadamente 200 ó 250 puntos vecinos donde apoyar la predicción en el módulo de Direct Weight es bastante apropiado. En este intervalo nos aseguraremos de obtener un error suficientemente bajo para hacernos una idea de cual puede llegar a ser la potencia de la predicción de antemano, y un tiempo de ejecución suficientemente corto a su vez para no lastrar la eficiencia de las pruebas.

El objetivo de probar aumentando el número de vecinos es ver cuántos se usarán una vez se obtenga la mejor combinación, para realizar una predicción más acertada, aunque sea más larga, ya que solo será necesario en ese momento computar una única combinación. En este caso observamos como a partir de 500 puntos vecinos, el error apenas disminuye, por tanto usar un número mayor no aportaría una mejora de resultado.

## 6.2 Resultados finales

Las pruebas finales directamente con el número de iteraciones máximo en el intervalo contemplado en el capítulo 6.1.2, es decir, 30 iteraciones. Además, se han desarrollado desde el primer momento con varios puntos iniciales en cuanto a número de indicadores del primer conjunto de combinaciones (la modificación implementada con los resultados finales del primer caso). Aparecen los resultados en la tabla 6-4, al igual que la represetación gráfica de la prueba más corta y la más larga, en las figuras 6-11 y 6-12, respectivamente.

| Combinaciones | Iteraciones | Error | Tiempo (s) | Combinación elegida |
|---|---|---|---|---|
| 100 | 30 | 2.7857e-7 | 5469.814 | {18,16} |
| 100 | 30 | 2.7857e-7 | 6291.028 | {18,16} |
| 500 | 30 | 2.7857e-7 | 29180.896 | {18,16} |
| 200 | 30 | 2.7857e-7 | 9890.887 | {18,16} |
| 1000 | 30 | 2.7857e-7 | 20254.934 | {16,18} |
| 2500 | 30 | 2.7857e-7 | 50492.380 | {18,16} |

Tabla 6-4 Mejores combinaciones con errorres y tiempos de ejecución

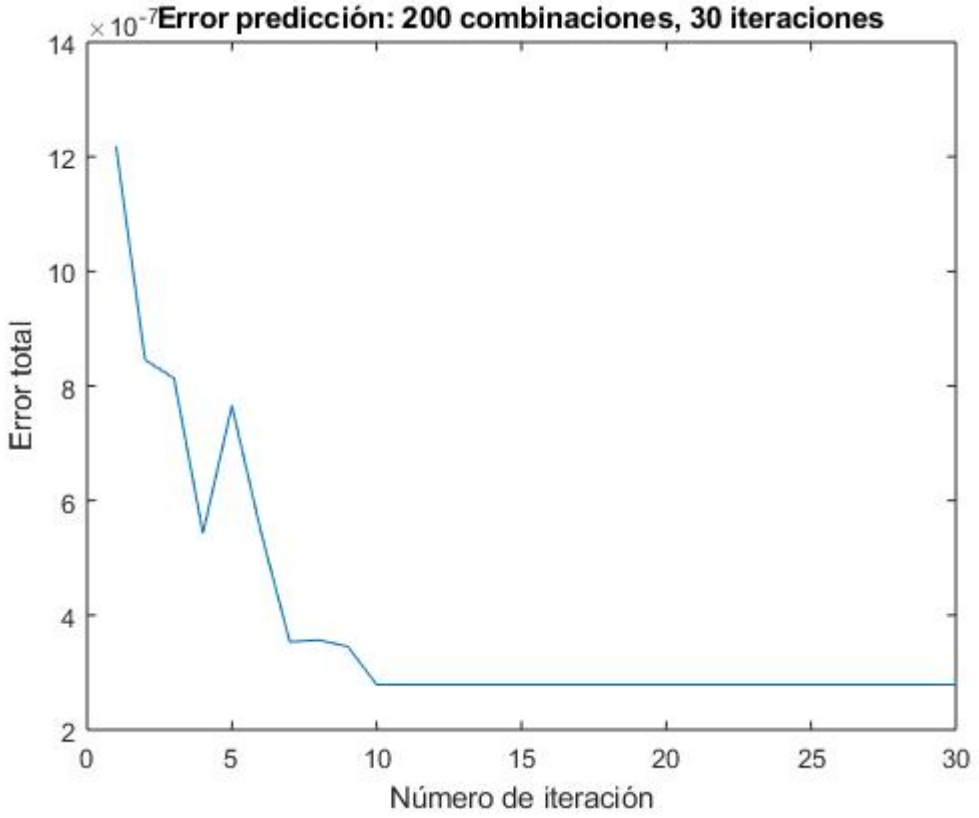


Figura 6-11 Evolución del error con 100 combinaciones y 30 iteraciones

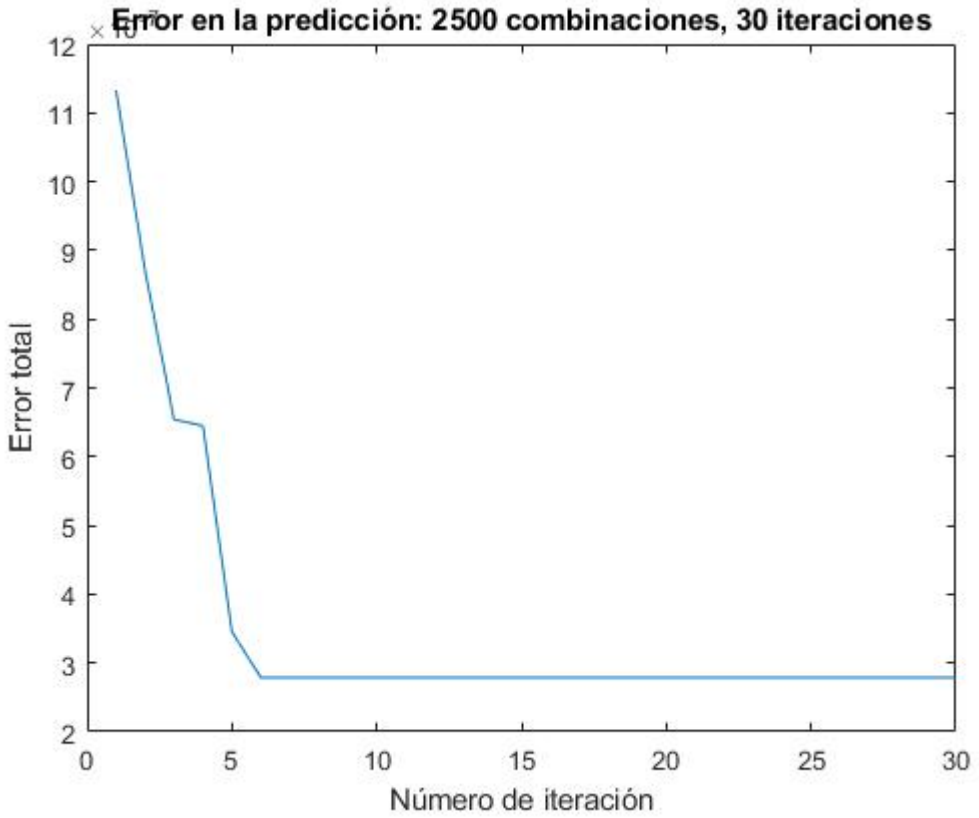


Figura 6-12 Evolución del error con 2500 combinaciones y 30 iteraciones

## 6.3 Conclusiones

Como conclusion, en este caso ha sido más rápido hallar la mejor combinación. Con solo 100 combinaciones y 30 iteraciones se ha comprobado cómo casi todas las pruebas conducen a la combinación de indicadores también elegida con 2500 combinaciones y 30 iteraciones. En concreto la elegida consta únicamente de dos de los veinticuatro posibles indicadores de los que disponíamos. El error cometido sera de 2.7857e-7, y los indicadores seleccionados son:

16: MA(5). Se trata de la media móvil, en este caso tomada en cinco minutos.

18: Disparity(5). Se trata de la disparidad, en este caso tomada en cinco minutos.

Sus respectivas fórmulas pueden consultarse en la tabla 2-1, en lugar de días se referirán a minutos.

Además, para comprobar la utilidad del algoritmo, se computa una combinación con los 24 indicadores disponibles, cuyo error es de 1.5034e-4. Por tanto el porcentaje de mejora será de un 99.81% del primer error.

Se halla de igual modo el error relativo teniendo en cuenta el valor medio de los precios reales, que es de 10.16492. Por tanto, el error relativo cometido será de 0.00000274%.

La representación gráfica de la predicción así como la del precio real se puede consultar en la figura 6-13. En la figura 6-14 aparece ampliada la zona estudida de mayor volatilidad, con la idea de mostrar de forma más clara el comportamiento de la predicción.

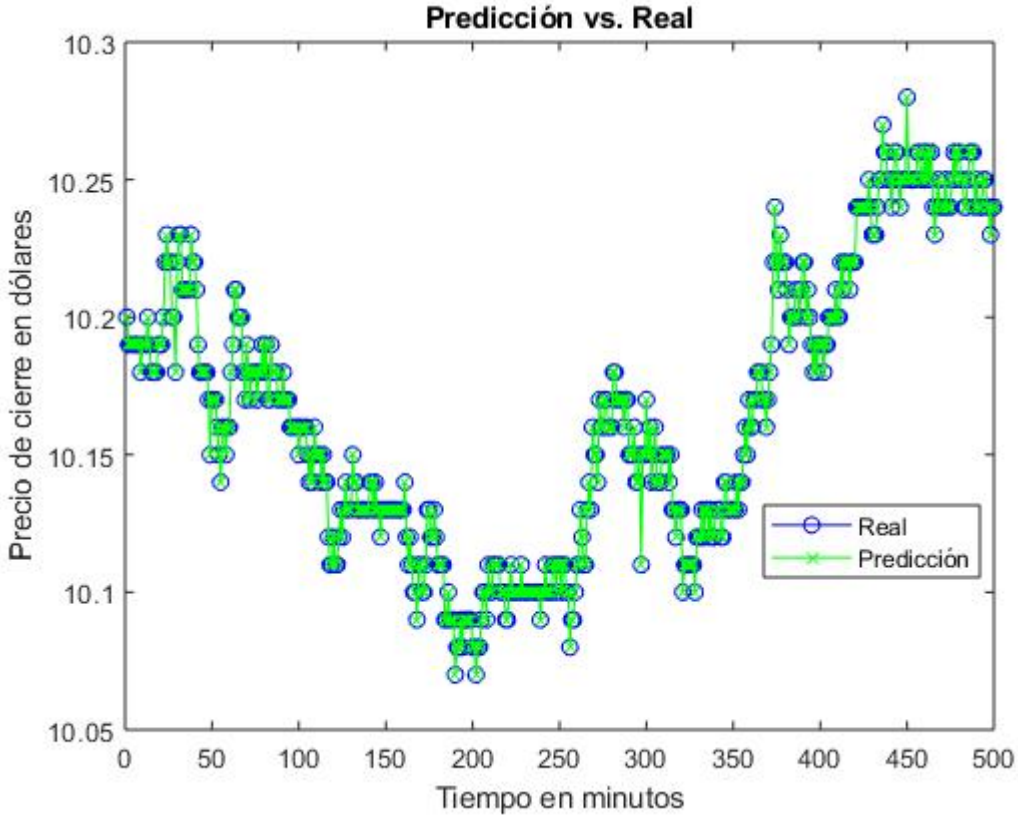


Figura 6-13 Comparativa predicción y precio real

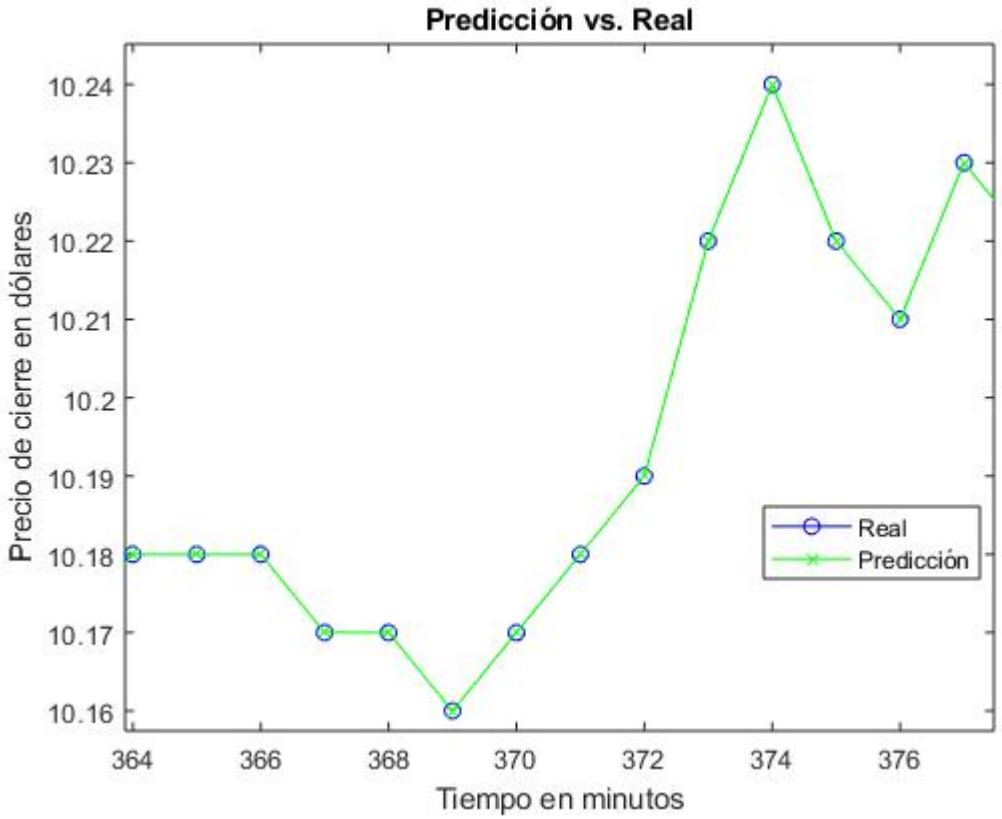


Figura 6-14 Comparativa predicción y precio real (ampliada)

# 7 CONCLUSIONES GENERALES

Es conveniente extraer una serie de conclusiones finales en cuanto a las características y al uso general del algoritmo, así como su potencia, limitaciones y posibilidad de mejora.

En primer lugar, cabe destacar la importancia de elección de los conjuntos de entrenamiento y de test. Como es lógico, los conjuntos deben variar en su tamaño (según el caso concreto que queramos analizar, el periodo de muestreo de los indicadores que están involucrados y los resultados que queramos obtener). Debido a la gran longitud de los conjuntos de datos que se manejaban, se han usado un 40% de ellos como set de entrenamiento. Lo ideal habría sido hacer uso de todos los datos disponibles, pero dadas las circunstancias del desarrollo de la pruebas y los medios con las que se han realizado (en su mayoría con un ordenador portátil de 4 núcleos), se ha debido llevar a cabo con un conjunto de menor tamaño. Para realizar los análisis en los capítulos anteriores, se han tomado una longitud de 250 unidades (un periodo similar a un año), para el primer caso, ya que los datos eran diarios.

En cambio, para el segundo caso, con datos minutarios, se ha ampliado el conjunto de test a 500 unidades (aproximadamente 8 horas). Se ha decido tomar este conjunto debido a las circunstancias mencionadas anteriormente, aunque podría ser interesante comprobar la capacidad de predicción con un conjunto de test de más de 500 minutos. Esto supondría en cierto modo un aumento del error más o menos significativo, dependiendo del caso. Se puede llegar a la conclusión que este factor es muy deseable tener en cuenta, y posiblemente lo más acertado será comprobar cuáles serían las longitudes apropiadas en cada caso concreto al que se quisiera aplicar el algoritmo.

Otra cuestión que no se ha abordado con profundidad es el establecimiento del número de indicadores que tendrán los conjuntos de combinaciones inicialmente. No se ha especificado los números de indicadores que tendrán los diferentes grupos al generar las combinaciones originales, ya que como en el caso anterior, se considera que es una cuestión dependiente de los datos que se quieran analizar. En un principio se han establecido 3 “grupos” o “puntos de partida”, y el número de indicadores que contendrá inicialmente cada grupo dependerá de los indicadores diponibles en los datos.

En el primer caso se ha comenzado con un tercio de las combinaciones de 10 indicadores, un segundo tercio de ellas con 20 indicadores y el último tercio con 30 indicadores. Los indicadores disponibles son 40, por lo que parecen puntos de partida razonables. Para el segundo caso se ha comenzado con 8, 14 y 20 indicadores respectivamente, al haber disponibles 24 de ellos. Durante las pruebas con unos datos concretos, será necesario no únicamente establecer los puntos de partida según el número de indicadors técnicos disponibles, sino también cambiar el número de grupos en los que dividamos este primer conjunto.

A su vez, puede ser intersante ir modificando estas dos variables a medida que vayamos observando donde se encuentran los mínimos locales, e ir ejecutando el algoritmo bajo estas distintas opciones iniciales.

Ejecutar el algoritmo varias veces tiene otra utilidad, que es la visualización del número de iteración a partir de la cuál el error se asienta. Como se ha comprobado, esto cambia dependiendo del caso al que se enfrente el algoritmo, y por tanto, es conveniente comprobar este aspecto del problema antes de lanzarnos en la búsqueda de la selección final de características. De este modo, el resultado se obtendrá de forma más rápida y sera más fiable.

Estas tres primeras conclusiones desembocan en otra generalidad, que es la necesidad de ejecutar el código más de una vez para cada caso concreto, y ver de qué manera proporciona un mejor resultado. Esta puede ser considerada una de las limitaciones tal y como se ha planteado el problema, aunque la práctica de estudiar las características de nuestros datos iniciales antes de apresurarse a realizar una inversion en base a los resultados obtenidos es siempre una buena idea, por muy automatizado que esté el código. Podría tratarse de una de las posibles mejoras de la codificación, encaminadas en este aspecto, las cuáles en principio no entrañarían gran dificultad, si bien como se ha mencionado, el modo manual de operar en un principio con un nuevo caso puede otorgar más fiabilidad.

En cuanto a los resultados obtenidos, podemos decir que son variables debido a la naturaleza del problema. Por ejemplo, el error relativo cometido en el segundo caso es mucho menor que en el primero. Ocurre de modo parecido con el porcentaje de mejora al seleccionar la mejor combinación respecto al resultado obtenido con todos los indicadores disponibles. Esto determina que con datos extraídos en intervalos menores de tiempo la predicción sea posiblemente más fiable. Los números de indicadores seleccionados en cada caso son muy dispares, sobre todo en cuanto a número (con Dow Jones fueron necesarios once, mientras que con Ping An Bank únicamente dos). Estas son algunas de las diferencias entre los dos casos de estudio, que conducen a pensar que las características del segundo sean más favorable para la aplicación del algoritmo.

La mejora más interesante a realizar probablemente sea el llevar a cabo predicciones a más largo plazo. Es decir, en casos en los que funcione mejor el algoritmo, como con la acción Ping An Bank, se podría tratar de continuar obteniendo indicadores técnicos de los precios ya predichos, e ir comprobando hasta que punto se obtiene un error acceptable.

# Apéndice: Códigos

## Función Procesar_Datos.m

```
%num=xlsread('DJDataTechnicals.xlsx');
num=xlsread('Indicators.xlsx');
Data1=num;
```

## Función main.m

```
M=6; %Número de combinaciones
maxiter=500; %Número máximo de iteraciones
limite_error=5e-9;%limite del error (no usado)
n_vec=250; %Numero de puntos vecinos

[n_filas,n_columnas]=size(Data1);
total_ind=n_columnas-1;
length_dat=floor(n_filas*0.4); %longitud de la muestra 40%
length_test=500; %longitud del test (1 año aproximadamente en el primer caso,
500 minutos en el segundo)
Y_DAT=Data1(end-length_test-length_dat:end-length_test,end);
Y_TEST=Data1(end-(length_test-1):end,end);

clear error_min
error_med=zeros(M,1);
error_med_up=zeros(M,1);
error_med_down=zeros(M,1);
c_selected_cell=cell(maxiter,1);

it=0;

%[Cs_cell]=c_generator_celda(M,20,40);%Genera M combinaciones de 20
indicadores cada una.
%PARA TENER COMBINACIONES CON DISTINTOS INDICADORES
c1=floor(M/3);
c2=c1;
c3=M-c1*2;
c1_cell=c_generator(c1,8,total_ind);
c2_cell=c_generator(c2,16,total_ind);
c3_cell=c_generator(c3,20,total_ind);
[Cs_cell]=[c1_cell;c2_cell;c3_cell];

%% I COMPUTATION
parfor i=1:M
    I=Cs_cell{i};
```

```
    [X_DAT,X_TEST]=computation(Data1,I,length_dat,length_test);
    Y_PRED = DirectWeight(X_DAT,Y_DAT,X_TEST,n_vec);
    error=abs(Y_TEST-Y_PRED);
    error_med(i,1)=sum(error)/length_test;
    %error_cuad(i,1)=sum(error.^2)/length_test;
end

c_selectedM_cell=Cs_cell;
%% BUCLE ITERACIONES
while it<maxiter

    [I_up_cell,I_down_cell]=I_updown(c_selectedM_cell,total_ind);
%% I_UP COMPUTATION
parfor i=1:M
    I_up=I_up_cell{i};
    [X_DAT,X_TEST]=computation(Data1,I_up,length_dat,length_test);
    Y_PRED = DirectWeight(X_DAT,Y_DAT,X_TEST,n_vec);
    error=abs(Y_TEST-Y_PRED);
    error_med_up(i,1)=sum(error)/length_test;
    %error_cuad(i,1)=sum(error.^2)/length_test;
end

%% I_DOWN COMPUTATION

parfor i=1:M
    I_down=I_down_cell{i};
    [X_DAT,X_TEST]=computation(Data1,I_down,length_dat,length_test);
    Y_PRED = DirectWeight(X_DAT,Y_DAT,X_TEST,n_vec);
    error=abs(Y_TEST-Y_PRED);
    error_med_down(i,1)=sum(error)/length_test;
    %error_cuad(i,1)=sum(error.^2)/length_test;
end

%% STEP 3i abd 3j ERROR SORTING AND 2M PICKING

mean_error=[error_med error_med_up error_med_down]; %Une los tres sets de
errores (original, up, down)

for i=1:M
      %Anulo fila con valor de error mayor
      [m,n]=find( mean_error==max(mean_error(i,:)));
      mean_error(i,n)=NaN;

end

c_selected2M_cell=cell(M,1);
k=1;
while k<=2*M
for i=1:M
    for j=1:3
       if isnan(mean_error(i,j))==0
        switch j
            case 1
                c_selected2M_cell{k}=c_selectedM_cell{i};
            case 2
                c_selected2M_cell{k}=I_up_cell{i};
            case 3
                c_selected2M_cell{k}=I_down_cell{i};
        end
         k=k+1;
       end
```

```
    end

end
end



%% STEP 4 (M PICKING)
c_selectedMaux=c_selectedM_cell;
c_selectedM_cell=cell(M,1);


for i=1:M
  [m,n]=find(mean_error==(min(min(mean_error))));

        switch n(1)
            case 1

                c_selectedM_cell{i}=c_selectedMaux{m(1)};
            case 2

                c_selectedM_cell{i}=I_up_cell{m(1)};
            case 3

                c_selectedM_cell{i}=I_down_cell{m(1)};
        end
   error_med(i,1)=mean_error(m(1),n(1)); %Guardo el error obtenido para la
próxima iteracion
   mean_error(m(1),n(1))=NaN; %Elimino el error obtenido de la tabla

end



%% CÁLCULO DE LA COMBINACION CON MENOS ERROR

 c_selectedM=c_selectedM_cell{1};

it=it+1;

c_selected_cell{it}=c_selectedM; %Guardo la mejor combinación de cada
iteración
error_min(it,1)=error_med(1);
if error_med(1) <limite_error; break;
end
end %Bucle iteraciones

%% MEJOR COMBNACION

v=[1:it]';
v=[v error_min];
error_min(end,1)

%  c_selected=c_selected_cell{v(end,1)};
% [X_DAT,X_TEST]=iteration(Data1,c_selected,length_dat,length_test);
% [Y_PRED] = DirectWeight(X_DAT,Y_DAT,X_TEST,250);
% error_final=abs(Y_TEST-Y_PRED);
```

```
% error_final_med=sum(error_final)/length_test;

%  figure (1)
%  x=[1:length_test];
%  plot(x,Y_TEST,'b-o')
%  hold on
%  plot(x,Y_PRED,'g-x')

figure (2)
x=1:it;
plot(x,error_min);
xlabel('Número de iteración');
ylabel('Error total');
title(['Error predicción: ',num2str(M),' combinaciones, ',num2str(maxiter),'
iteraciones']);
```

## Función c_generator.m

```
function [c]=c_generator_celda(M,num_ind,total_ind)
format shortG
c=cell(M,1);


for kk=1:M
    c{kk} = sort(randperm(total_ind,num_ind));
end

cond=0;
cond2=0;

while cond2==0
for i=1:M-1
    if cond==1
        cond=0;
    end
    for j=i+1:M
        if sum(eq(c{i}(1,:),c{j}(1,:)))==num_ind
           c{i} = sort(randperm(total_ind,num_ind));
           cond=1;
           break
       end
    end
    if cond==1
    break
    end
end
if i==M-1 && j==M
    cond2=1;
end
end
end
```

## Función computation.m

```
function [X_DAT,X_TEST]=computation(Data1,c,length_dat,length_test)
[M_c,N_c]=size(c);

for j=1:N_c
X_DAT(:,j)=Data1(end-length_test-length_dat:end-length_test,c(1,j));
X_TEST(:,j)=Data1(end-(length_test-1):end,c(1,j));
end
end
```

## Función DirectWeight.m

```
function [Y_TEST] = DirectWeight(X_DAT,Y_DAT,X_TEST,num_vec)

% Y_TEST = DirectWeight(X_DAT,Y_DAT,X_TEST)
%
%Función que hace el Direct Weight optimization para predicción.
%
%- X_DAT son las entradas pasadas.
%- Y_DAT son las salidas pasadas asociadas a X_DAT.
%Juntos forman el dataset equivalente al conjunto de training
%y validation en las NN.
%- X_TEST son las entradas que queremos predecir.
%- num_vec número de puntos vecinos en los que se va a apoyar la
%predicción. Mi recomendación es poner entre 250~500.
%
%Las dimensiones de las matrices tienen que ser las siguientes:
%- X_DAT = n_dataset*m     ->     Siendo n_dataset el número de puntos que
componen el
%dataset y m el tamaño del regresor.
%- Y_DAT = n_dataset*1     ->     Siendo n_dataset el número de puntos que
componen el
%dataset.
%- X_TEST = n_test*m       ->     Siendo n_test el número de puntos que
componen el
%conjunto de test y m el tamaño del regresor.
%- num_vec es un escalar.

if nargin~=4
    disp('Error: Hay que introducir 4 argumentos');
    return;
end

ind_Training = size(X_DAT,1);
ind_Test = size(X_TEST,1);

if ind_Training<num_vec
    disp('Error: El dataset debe tener al menos num_vec puntos');
    return;
end
```

```
num_pun = num_vec;

Y_TEST = zeros(ind_Test,1);

    for i=1:ind_Test

            x = X_TEST(i,:);

            X = X_DAT;
            Y = Y_DAT;

            [~,sel] = sort(sum((X_DAT-repmat(x,[ind_Training 1])).^2,2));

            X = X(sel(1:num_pun),:);
            Y = Y(sel(1:num_pun),:);

            Aeq = [X'; ones(1,num_pun)];
            beq = [x'; 1];

             %[res,~] = quadprog(2*eye(num_pun),[],[],[],Aeq,beq);
            Q=2*eye(num_pun);
            E=Aeq;
            d=beq;
            [m,n]=size(d);

            mat=[Q E';E zeros(m,m)];
            term=[zeros(num_pun,1);d];
            res=mat\term;

            res=res(1:num_pun);

            Y_TEST(i) = Y'*res;

    end

end
```

## Función I_updown

```
function [c_up,c_down]=I_updown_celda(c,total_ind)
[m,n]=size(c);


%BUCLE PARA ELIMINAR
%INDICADORES DE DISTINTAS COLUMNAS EN CADA COMBINACION
c_down=cell(m,n);
c_up=cell(m,n);
for i=1:m
      [m1,n1]=size(c{i});
      c_downrow=c{i};
      e=randi([1 n1]);
      c_downrow(e)=[];
      c_down{i}=c_downrow;
end
```

```
%SEGUNDO BULCE PARA COMPROBAR QUE NO HAY REPETICION

%c_up=[c zeros(m,1)];
for i=1:m
    c_row=c{i};
    while 1
    r=randi([1 total_ind]);
    if sum(ismember(c_row(1,:),r))==0; break;
    end
    end
    c_row=[c_row r];
    c_up{i}=c_row;
end
end

function [Y_TEST] = DirectWeight(X_DAT,Y_DAT,X_TEST,num_vec)
```

## Función DirectWeight_original.m

```
% Y_TEST = DirectWeight(X_DAT,Y_DAT,X_TEST)
%
%Función que hace el Direct Weight optimization para predicción.
%
%- X_DAT son las entradas pasadas.
%- Y_DAT son las salidas pasadas asociadas a X_DAT.
%Juntos forman el dataset equivalente al conjunto de training
%y validation en las NN.
%- X_TEST son las entradas que queremos predecir.
%- num_vec número de puntos vecinos en los que se va a apoyar la
%predicción. Mi recomendación es poner entre 250~500.
%
%Las dimensiones de las matrices tienen que ser las siguientes:
%- X_DAT = n_dataset*m     ->     Siendo n_dataset el número de puntos que
componen el
%dataset y m el tamaño del regresor.
%- Y_DAT = n_dataset*1     ->     Siendo n_dataset el número de puntos que
componen el
%dataset.
%- X_TEST = n_test*m       ->     Siendo n_test el número de puntos que
componen el
%conjunto de test y m el tamaño del regresor.
%- num_vec es un escalar.

if nargin~=4
    disp('Error: Hay que introducir 4 argumentos');
    return;
end

ind_Training = size(X_DAT,1);
ind_Test = size(X_TEST,1);

if ind_Training<num_vec
    disp('Error: El dataset debe tener al menos num_vec puntos');
```

```
    return;
end

num_pun = num_vec;

Y_TEST = zeros(ind_Test,1);

    for i=1:ind_Test

            x = X_TEST(i,:);

            X = X_DAT;
            Y = Y_DAT;

            [~,sel] = sort(sum((X_DAT-repmat(x,[ind_Training 1])).^2,2));

            X = X(sel(1:num_pun),:);
            Y = Y(sel(1:num_pun),:);

            Aeq = [X'; ones(1,num_pun)];
            beq = [x'; 1];

             [res,~] = quadprog(2*eye(num_pun),[],[],[],Aeq,beq);

            Y_TEST(i) = Y'*res;

    end

end
```

# 8 BIBLIOGRAFÍA

[1] Alfonso, G. and R. Ramírez, D., 2021. *A Nonlinear Technical Indicator Selection Approach for Stock Markets. Application to the Chinese Stock Market*. [online] Researchgate. Available at: <https://www.researchgate.net/publication/343497506_A_Nonlinear_Technical_Indicator_Selection_Approach_for_Stock_Markets_Application_to_the_Chinese_Stock_Market>

[2] Roll, J., Nazin, A. and Ljung, L., 2021. *Nonlinear system identification via direct weight optimization*.

[3] Pineda, A., 2021. *Guia indicadores tecnicos*. [online] Academia.edu. Available at: <https://www.academia.edu/37973154/Guia_indicadores_tecnicos>

[4] Murphy, J. and Gispert Ramis, A., 2018. *Análisis técnico de los mercados financieros*. Barcelona: Gestión 2000.

[5] Economipedia. 2021. *Indicador técnico - Definición, qué es y concepto | Economipedia*. [online] Available:<https://economipedia.com/definiciones/indicadortecnico.html#:~:text=El%20indicador%20t%C3%A9cnico%20es%20una,la%20decisi%C3%B3n%20de%20una%20inversi%C3%B3n.>

[6] Rankia.com. 2021. *Guía indicadores técnicos*. [online] Available at: <https://www.rankia.com/informacion/guia-indicadores-tecnicos>

[7] Aprendemachinelearning.com. 2021. *Aprende Machine Learning*. [online] Available at: <http://aprendemachinelearning.com/>

[8] Wright, S., 2021. *Continuous Optimization ( Nonlinear and Linear Programming )*. [online] Semanticscholar.org. Available at: <https://www.semanticscholar.org/paper/Continuous-Optimization-(-Nonlinear-and-Linear-)-Wright/c3312b37f9a719231b923cbd734654362dd26853>