Predicción de valores en mercados financieros

Celia Calvo González
Raquel Blanco Morago

Grado en Ingeniería Informática

Curso académico: 2018/2019

Director: Rafael Caballero Roldán

Resumen

En este trabajo se van a comparar diversos métodos para predecir el futuro de ciertos índices bursátiles a partir de los datos históricos de otros o de él mismo. En concreto, se seleccionan uno índices bursátiles a los que se les aplican distintos experimentos. En primer lugar, utilizamos el método ARIMA, en el cual se utiliza el pasado de la misma serie temporal de la cual se quiere predecir el futuro y se compara con el sencillo método de Naïve, consistente en proyectar el último valor conocido. Posteriormente, se realizan experimentos con distintos métodos de regresión (Regresión Lineal, Gradient Boosting Regressor, Random Forest Regressor y Voting Regressor) utilizando incrementos en lugar de los valores reales, para lo que es necesario hacer un preajuste de los datos de entrada. Además, incorporamos varias variables independientes para mejorar la predicción. Como último método de predicción se realiza un experimento utilizando redes neuronales, en concreto se trata de percepción multicapa con backpropagation.
Finalmente, se comparan y analizan los resultados de todos estos métodos a través de las medidas de error típicas y se exponen las conclusiones finales.

Palabras clave: índices bursátiles, predicción, regresión, ARIMA, series temporales, redes neuronales, RMSE.

Abstract

In this work we will compare different methods to predict the future of certain stock indices from the historical data of others and itself. Specifically, one stock index is selected to which different experiments are applied. First, we use the ARIMA method, which uses the pasts of the same time series from which we want to predict the future and compares it with the simple Naïve method (consisting of projecting the last known value). Subsequently, experiments are performed with different regression methods (Linear Regression, Gradient Boosting Regressor, Random Forest Regressor and Voting Regressor) using increments instead of real values, for which it is necessary to make a preset of the input data. In addition, we incorporate several independent variables to improve predictions. The last method of prediction is an experiment using neural networks, specifically multi-layer perceptron with backpropagation.
Finally, the results of all these methods are compared and analyzed through the typical error measures and the final conclusions are reveal.

Keywords: stock market, forecast, regression, ARIMA, time series, neural networks, RMSE.

Tabla de contenidos

Capítulo 1

Introducción

En este capítulo presentamos el problema que queremos resolver, la literatura que existe acerca del tema, nuestros objetivos, y resumimos la estructura del trabajo.

1.1 Problema a resolver

Este trabajo trata el problema de la predicción de valores bursátiles. Como veremos en la siguiente sección, es un tema que por su gran importancia ha sido tratado en numerosos trabajos anteriores. En el nuestro buscamos una perspectiva diferente: se trata de localizar qué valores pueden predecir el futuro de otro. Es decir, buscamos valores *refugio* de otros valores, en el sentido de que los inversores se muevan a ellos anticipando cambios en otros valores que consideren, por ejemplo, van a tener más riesgo.

Hemos decidido afrontar el problema con varios métodos de aprendizaje automático para tener una visión más amplia de la capacidad de predicción de estos valores. Esto nos permite hacer un estudio comparativo y determinar qué método es el que ofrece mejores resultados con nuestro subconjunto de valores.

1.2 Objetivos

Nuestro objetivo es predecir un cierto valor en un cierto futuro f, conociendo los datos de los p días anteriores de otros k valores. Por ejemplo, podríamos querer calcular el valor del Euro a 30 días, sabiendo lo que han hecho la última semana el Yen japonés y la Libra esterlina.

Para ello, primero comenzamos por utilizar el método ARIMA, basado en la predicción de una serie a partir de pasados de ella misma, comprobando así que en la mayoría de ocasiones no da buenos resultados. Con el objetivo de mejorar los resultados, probaremos otras técnicas en las que se tienen en cuenta valores de otros índices distintos al que se quiere predecir, siendo el objetivo principal encontrar estos índices y que hagan buenas predicciones.

1.3 Tecnologías empleadas

El código de los algoritmos utilizados ha sido desarrollado en R y Python, utilizando los entornos rStudio y Jupyter Notebook respectivamente.

R y **Python** están posicionados actualmente como los lenguajes más populares en cuanto a aprendizaje automático. R está desarrollado específicamente para hacer modelos estadísticos, llevar a cabo análisis y resultados gráficos, en cambio, Python es un lenguaje de propósito general pero que cuenta con multitud de librerías, en nuestro caso podemos destacar *pandas* para operar con la estructura de los datos (*Dataframes*) y *sklearn*, librería específica para poner en práctica experimentos de aprendizaje automático. Como repositorio del código desarrollado hemos utilizado *GitHub*.

Además, para desarrollar la memoria de este proyecto hemos usado *Overleaf*, un servicio de **LaTeX** para desarrollar textos científicos online.

Debido a que algunos de los experimentos que hemos realizado tardaban mucho tiempo en ejecutarse, optamos por usar para acelerar el proceso una *DSVM* (acrónimo de Data Science Virtual Machine), un máquina virtual en la nube que ofrece *Azure*, preconfigurada y aprovisionada específicamente para hacer ciencia de datos, usar este servicio **quitaba carga de trabajo** en a nuestro ordenador local facilitándonos hacer otras cosas y **reducía el tiempo** de ejecución de los experimento debido a que las características del "hardware" de la máquina virtual son superiores.

1.4 Estructura de la memoria

Comenzamos haciendo una recopilación de algunos artículos interesantes relacionados con nuestro trabajo (métodos de regresión y de redes neuronales en valores bursátiles) en el capítulo 3. Esto es importante para conocer la situación actual en cuanto a técnicas y métodos utilizados, los resultados que se han obtenido hasta ahora y comprobar las innovaciones que incorporamos en nuestro trabajo.

Seguidamente, describimos nuestro conjunto de datos en el capítulo 4, es decir, qué datos tenemos, su significado y cómo interpretarlos. Además, explicamos los distintos errores de predicción a tener en cuenta y la preparación de los datos para los métodos basados en aprendizaje automático que utilizaremos en el capítulo 6.

A continuación, en el capítulo 5, utilizamos uno de los métodos más conocidos de series temporales, el método ARIMA; describiremos las etapas a seguir para la realización de un modelo para más adelante aplicarlo a nuestros índices seleccionados y discutir los resultados con respecto al método Naïve.

Seguidamente, tenemos el capítulo más destacable, el capítulo 6, donde introducimos los métodos de predicción con regresión que vamos a utilizar para más adelante desarrollar el experimento y discutir los resultados obtenidos.

Más adelante, podemos encontrar el capítulo 7, en el cual se introducirán los conceptos básicos sobre las redes neuronales (qué son, cómo se usan y por qué usarlas), para posteriormente realizar el experimento con perceptrón multicapa y analizar los resultados que se han obtenido con este método.

Inmediatamente después, en el capítulo 8, exponemos las contribuciones personales de cada una de las autoras de este trabajo.

Por último, en el capítulo 9, encontramos las conclusiones a las que hemos podido llegar al aplicar a nuestro conjunto de datos los métodos de predicción comentados en todos los anteriores capítulos.

Capítulo 2

Introduction

In this chapter we present: the problem we want to solve, the literature that exists on the subject, our objectives, and a summarization of the structure of this study.

2.1 Problem to solve

This paper deals with the problem of stock market prediction. As we will see in the next section, it is a topic that, due to its great importance, has been addressed in numerous previous works. Thus, we look for a different perspective: we propose to seek for values that can be used to predict the future of another. Thus, it is to say, we look for values that act as *shelter* other values, in the sense that investors move to them anticipating changes in other values that they consider. For example, they will have more risk.

We have decided to face the problem with several machine learning methods to gain a broader vision of the predictability of these values. This allows us to make a comparative study and to determine which method offers the best results with our subset of values.

2.2 Objectives

Our objective is to predict a certain value in a certain future (f), knowing the data of the previous days (p) of other values(k). For example, we might want to calculate the value of the Euro at 30 days(f), knowing what the Japanese yen and the Pound sterling(k) have done last week(p).

To do this, we first began by using the ARIMA method, based on the prediction of a series from past ones of itself, proving that in most cases it does not give good results. In order to improve the results, we will try other techniques that consider values from other indices than what we want to predict, the main objective being to find these indices and make good predictions.

2.3 Technologies used

The code of the algorithms used has been developed in R and Python, using the rStudio and Jupyter Notebook environments respectively.

R and **Python** are currently positioned as the most popular languages in terms of machine learning. R is developed specifically to make statistical models and to carry out analysis and graphical results. While, Python is a general purpose language but also has a multitude of libraries. In our case, we can emphasize *pandas* to operate with the structure of the data (*Dataframes*) and *sklearn* (a specific library) to implement machine learning experiments. As a repository of the developed code we have used GitHub.

In addition, to develop the memory of this project we have used *Overleaf*, a service of **LaTeX** to develop scientific texts online.

Due to the fact that some of the experiments took a long time to execute, we opted to accelerate the process by using **DSVM** (acronym of Data Science Virtual Machine). This is a virtual machine in the cloud that **Azure** offers, preconfigured and provisioned specifically to do data science. We used this service in order to **remove the workload** on our local computer, thus making it easier for us to do other work. In turn **reducing the execution time** of the experiments because the "hardware" features of the virtual machine are superior.

2.4 Memory structure

We started by compiling some interesting papers related to our work (regression methods and neural networks in stock values) in chapter 3. This is important to check: the current state of the art (in terms of techniques and methods used), the results that have been obtained so far, and the innovations that we incorporate in our work.

Next, we describe our data set in chapter 4. That is, what data we have, their meaning and how to interpret them. In addition, we explain the different prediction errors to be taken into account and the preparation of the data for the methods based on machine learning that we will use in chapter 6.

Then, in chapter 5, we use one of the best-known methods of time series, the ARIMA method; we will describe the steps to follow for the realization of a model to later apply it to our selected indices and discuss the results with respect to the Naïve method.

Afterwards, we have the most remarkable chapter, chapter 6: where we introduce the prediction methods with regression that we will use to later develop the experiment and discuss the corresponding results.

Later, we move to chapter 7, which introduces the basic concepts about neural networks, and shows the performance of the experiment with multilayer perceptron as well as analyze the results that have been obtained with this method.

Immediately after, in chapter 8, we expose the personal contributions of each one of the authors of this work.

Finally, in chapter 9, we find the conclusions that we have been able to apply to our set of prediction methods discussed in all the previous chapters.

Capítulo 3

Estado del arte

3.1 Artículos relacionados con el método ARIMA

La predicción en series temporales es un área importante, este se asemeja a nuestro objetivo de predecir el comportamiento futuro de índices bursátiles, en donde las observaciones del pasado se analizan para desarrollar un modelo. Uno de los modelos más extensamente usados en series temporales es el llamado ARIMA (acrónimo del inglés *Autoregressive Integrated Moving Average*), a continuación repasamos algunos artículos que emplean esta técnica de predicción de series temporales [Zhang, 2003].

1. Time Series Forecasting Using Hybrid ARIMA and ANN Models Based on DWT Decomposition [Khandelwal et al., 2015]

En primer lugar, vamos a mencionar los resultados obtenidos de la combinación del método ARIMA y redes neuronales usando una descomposición DWT (acrónimo del inglés *Discrete Wavelet Transform*).

El modelo ARIMA es muy conocido por su precisión y flexibilidad con diferentes tipos de series temporales. [Khandelwal et al., 2015]. En este artículo se pretende separar un conjunto de datos en su componente lineal y no lineal, ya que ARIMA asume una linealidad en los datos y las redes neuronales (ANN) que ajustan mejor series temporales no lineales. Vamos a centrarnos en los resultados que se han obtenido para *las tasas de cambio semanal de la libra esterlina al dólar estadounidense (1980-1993)*. Los resultados han sido los siguientes:

	RMSE	**MSE**	**MAPE**
ARIMA	0.01357203	0.0001842	6.46
ANN	0.01042593	0.0001087	4.73
Híbrido	0.009528798	0.0000907	4.32

Donde el **RMSE** es la raíz del promedio de los cuadrados de la diferencia entre el valor predicho y el valor observado, el **MSE** es el promedio de los cuadrados de las diferencias entre

el valor predicho y el valor observado y el **MAPE** es la media de los errores porcentuales en valor absoluto. Para más información, se puede consultar el apartado de medidas de error 4.2.1.

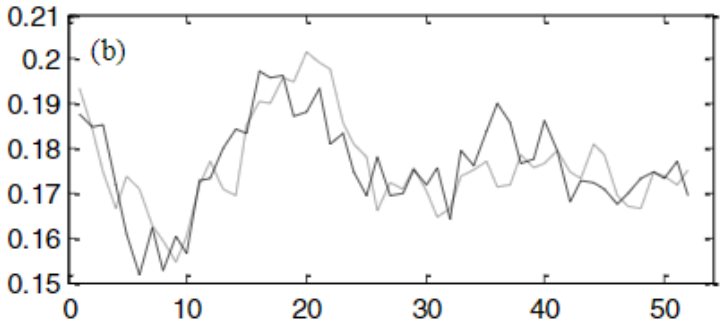


Figura 3.1: Gráfica de las predicciones y los valores reales de *las tasas de cambio semanal de la libra esterlina al dólar estadounidense (1980-1993)*
Fuente: [Khandelwal et al., 2015]

Para poder evaluar estos errores hay que tener en cuenta que el rango de los datos usados es [2.5-1.0], por lo que calculamos el error relativo, que es en ARIMA→ 0.00904802, en ANN→ 0.00695062 y en Híbrido→ 0.006352532. Teniendo en cuenta estos errores, se puede concluir: en primer lugar, que la combinación de los dos métodos ha dado un resultado satisfactorio, ya que ha mejorado a ambos por separado; y, como se había explicado anteriormente, se podía presuponer viendo la gráfica que la red neuronal ha llevado a cabo una mejor predicción ya que se trata de una función no lineal.

Finalmente, se considera que **separar los datos en una parte lineal y otra no lineal ofrece los mejores resultados.**

2. Comparison of arima and artificial neural networks models for stock price prediction. [Adebiyi et al., 2014]

En este otro artículo se compara la efectividad de ARIMA frente a redes neuronales prediciendo valores de la bolsa de Nueva York. Se centra específicamente en predecir con ARIMA el *Dell stock index*. Han llevado a cabo la predicción de 31 valores y el NRMSE que ha resultado ha sido de RMSE→ 0.268141790317 NRMSE→<= 0.1072567.

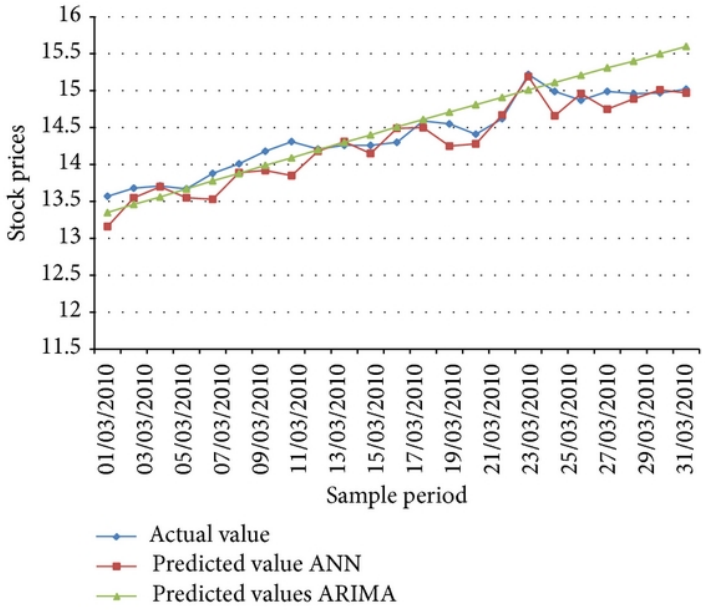


Figura 3.2: Gráfica de las predicciones y los valores reales de *Dell stock index*
Fuente: [Adebiyi et al., 2014]

Finalmente, concluye que **ambos métodos se ajustan bastante bien a los valores reales**, pero en este caso **el modelo de redes neuronales ha tenido mejores resultados que ARIMA**.

3. Gold price forecasting using arima model [Guha and Bandyopadhyay, 2016]

Este metal amarillo se ha apropiado de la atención de todas aquellas personas que tienen como propósito la inversión. En este tercer artículo se intenta predecir el **precio del oro en la India** utilizando el método ARIMA con datos de 10 años. Podemos observar en la Figura 3.3 que el modelo entrenado se ha ajustado a los datos con bastante exactitud. Además, en la gráfica también se puede observar la predicción que haría ese modelo para los 6 meses siguientes.

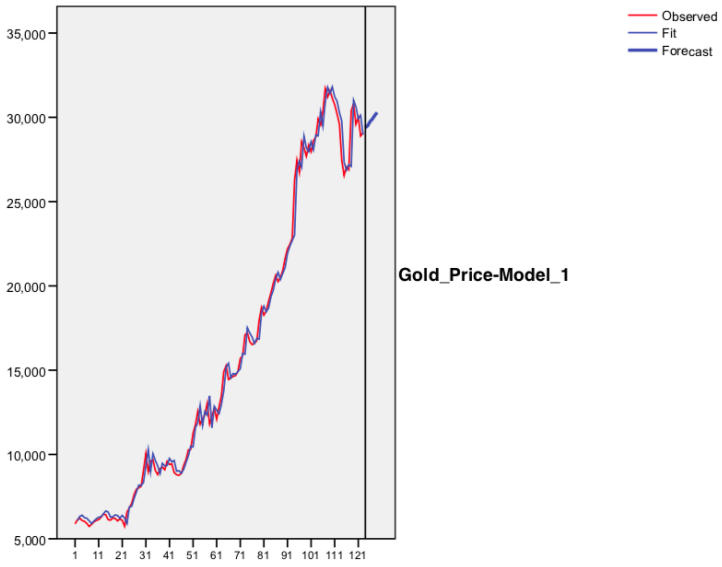


Figura 3.3: Gráfica de las predicciones y los valores reales de *oro en la India*
Fuente: [Guha and Bandyopadhyay, 2016]

Con este estudio se puede decir que **el modelo creado con ARIMA se ajusta bastante fielmente a los datos reales**, pero **no podemos asumir que en otras predicciones (para días posteriores a los que se tienen) con ese modelo ocurra lo mismo.**

3.2 Artículos relacionados con métodos de regresión múltiple

A continuación, comentaremos dos artículos relacionados con regresión múltiple, en concreto regresión lineal y regresión logística. Esto es debido a la importancia de esta técnica en los métodos que vamos a utilizar en nuestro trabajo y con el objetivo de comprender su utilidad en la predicción de valores bursátiles.

1. Stock market forecasting: artificial neural network and linear regression comparison in an emerging market [Altay and Satman, 2005]

En este artículo el objetivo es determinar si los índices se pueden predecir utilizando el carácter flexible y no lineal de las *ANN* en la *Bolsa de Valores de Estambul* o no. Para ello, con un conjunto de datos que consta de los indicadores *close*, *high* y *low* de los índices de *ISE-ALL* e *ISE-30* desde 1997 hasta 2005 diariamente, semanalmente y mensualmente; emplea los métodos lineales *OLS* (*Ordinary Least Squares*) y *Buy and Hold* (comprar acciones y disponer de ellas por un largo tiempo) y no lineales (Redes Neuronales) para predecir en los índices *ISE-ALL* e *ISE-30*.

Para empezar, realiza los modelos lineales (diariamente, semanalmente y mensualmente) para *ISE-ALL* e *ISE-30* con el método *OLS* con el fin de comparar los resultados más tarde con la predicción del modelo *ANN*. Entrena con datos desde 1997 hasta 2003 y crea los modelos. Además, se utiliza el otro método lineal *Buy and Hold* donde: predicción = $(1 + v_1) \cdot (1 + v_2) \cdot ... \cdot (1 + v_n) - 1$ donde n es el número de pasados y v_i es el valor de cada pasado, obteniendo de manera promediada cuánto porcentaje se gana al final del periodo. Para ambos se predice en el periodo de 2003 a 2005 y se analizan los resultados.

A continuación, crea los modelos de redes neuronales con *backpropagation* para *ISE-ALL* e *ISE-30*, con la misma franja de entrenamiento. Utiliza un procedimiento para encontrar la mejor configuración de la red neuronal (capas ocultas, número de neuronas...). Los resultados son los siguientes: los datos estadísticos (*RMSE*, *MAE* y *Theil's U*) indican que es mejor la regresión lineal, ya que tiene menor *RMSE* diariamente y mensualmente que *ANN*. En cambio, el mejor parece ser *Buy and Hold*, ya que tiene menor *RMSE* para todos los periodos y mejor *MAE* semanal y mensualmente. Teniendo *ANN* solo mejor *RMSE* que la regresión lineal cuando se emplean datos semanales.

Seguidamente, utiliza como medida de comparación el porcentaje de valores bien predichos (utilizando un radio de acierto) para los modelos de *ANN* y de regresión lineal (ver Figura 3.4). El porcentaje de aciertos es mayor en *ANN* diariamente y semanalmente con bastante diferencia, mensualmente es igual. El mayor porcentaje se da con los datos mensuales y menor cuando son diarios.

		Regression Strategy	ANN Strategy
Daily	ISE-All	55,0 %	57,8 %
	ISE-30	54,5 %	56,3 %
Weekly	ISE-All	62,4 %	67,1 %
	ISE-30	58,8 %	65,9 %
Monthly	ISE-All	78,3 %	78,3 %
	ISE-30	73,9 %	73,9 %

Figura 3.4: Porcentajes de acierto (en un rango) para regresión y para ANN
Fuente: [Altay and Satman, 2005]

Como última comparación, tenemos una cartera hipotética de valor inicial de 1 *YTL*, si invertimos 1 *YLT* en *ISE-ALL* e *ISE-30* comparamos qué ocurre con las distintas estrategias. Para todas da mejores resultados ANN excepto para *ISE-ALL* diariamente.

Se concluye que, aunque las medidas de error dan como peor predictor a *ANN* con datos diarios y mensuales, los *ANN* predicen mejor la dirección del mercado, ya que los porcentajes de aciertos eran mayores que en regresión diariamente y semanalmente y en la cartera hipotética de 1 *YTL* también fue mejor. Con lo cual, *ANN* es una herramienta importante para las decisiones de inversión en la *Bolsa de Estambul*.

Como se indica en el artículo, solo se usan *close*, *high* y *low*, se podría mejorar esto introduciendo variables de entrada para aumentar la precisión. Es por esto, que en nuestro trabajo incluimos en el conjunto de datos el indicador de volumen diario, ya que como se explica en la subsección 4.1.1, el volumen en los distintos índices es un gran señalizador del estado del mercado y cómo progresará su valor.

2. Prediction of stock performance in indian stock market using logistic regression [Dutta et al., 2012]

En este artículo se desarrolla un modelo para clasificar las acciones de una empresa utilizando los ratios financieros con el fin de predecir las acciones de mayor rendimiento en el mercado de valores de la India. La clasificación es en dos categorías: "good" y "poor", si el valor de las acciones de una compañía en un año determinado se eleva por encima del rendimiento del mercado, se clasifica como una opción de inversión "good" (buena), en caso contrario se clasifica como una opción de inversión "poor" (pobre).

En primer lugar se crea el modelo utilizando la *regresión logística*. El objetivo de este método es predecir la probabilidad de obtener un buen rendimiento del valor al ajustar las variables a una curva logística, esto implica encontrar una combinación lineal de las variables independientes en la que existan grandes diferencias entre cada grupo.

Se tienen los datos de 30 compañías en el periodo de 2005 a 2008, con un tamaño de muestra de 118 por año de cada compañía. Las variables dependientes: son *POOR* y *GOOD* y las independientes se tratan de ratios financieros (coeficientes que proporcionan unidades contables de medida y comparación), en concreto: *incremento porcentual en ventas netas, ganancia en efectivo por acción, valor contable, precio/efectivo por acción, precio/ganancia, ganancia antes de intereses, depreciación e impuestos, activos netos por ventas y precio/valor contable*. Cabe destacar que la probabilidad de corte para la decisión tomada es 0,42. Por lo tanto, al utilizar este valor de corte, se predeciría que cualquier compañía con una puntuación superior a 0.42 sería una compañía clasificada como "good", y cualquier compañía con una puntación menor a 0.42 se clasificaría como "poor".

Finalmente, se obtienen los resultados siguientes: 68 clasificados como "poor" y 50 como "good". Para medir la bondad de ajuste se utiliza cálculo del porcentaje de aciertos, como se puede observar en la siguiente tabla, obteniendo muy buenos resultados. Se puede observar que estos 8 ratios financieros pueden clasificar a las empresas con un nivel de precisión de hasta el 74,6 % en dos categorías ("good" o "poor"), según su tasa de rendimiento.

Classification Table[a]

Observed			Predicted		
			Perf		Percentage Correct
			POOR	GOOD	
Step 1	Perf	POOR	51	17	75.0
		GOOD	13	37	74.0
	Overall Percentage				74.6

Figura 3.5: Porcentajes de acierto para regresión logística
Fuente: [Dutta et al., 2012]

Además, se utiliza el test de bondad de ajuste de *Hosmer-Lemeshow*, que consiste en agrupar las observaciones según las probabilidades esperadas y luego probar la hipótesis de que la diferencia entre los eventos esperados y observados es aproximadamente cero para todos los grupos. Como resultado obtenemos que el nivel de significación observado para el valor de *chi-cuadrado* se encuentra en 0.217, lo que indica la aceptación de la hipótesis nula del modelo, es decir, no hay mucha diferencia entre los valores observados y predichos. El valor de *chi-cuadrado* es 10.737, que nos indica que la regresión logística es muy significativa.

3.3 Artículo relacionado con el perceptrón multicapa

Forecasting of Stock Prices Using Multi Layer Perceptron [A. Victor Devadoss, 2013]

En este artículo se realiza como experimento la predicción de los valores de cuatro compañías en la Bolsa de Valores de Bombay por medio de redes neuronales, en concreto de MLP (del inglés *Multilayer Perceptron* con *backpropagation*. Se tienen datos de las compañías desde el 1 de enero de 2012 hasta el 7 de noviembre de 2013. La definición y explicación del percepción multicapa se encuentra en la sección 7.2.

A continuación, se construyen dos redes neuronales *NN1 NN2* cada una con una capa de entrada, una capa oculta y una capa de salida. Se usan 3 neuronas para la capa de entrada que corresponden con los tres precios pasados del índice a estudiar. La capa oculta consiste en 16 neuronas en *NN1* y 6 en *NN2*. Por último, en la capa de salida es una, ya que lo que se desea predecir es el valor un día después. Como función de transferencia se utiliza *sigmoid*. Se aplican estos modelos para cada una de las compañías de las que se quiere predecir el valor futuro: en primer lugar se escalan los datos en el rango [0, 1], se entrena con el 60 % de los datos y se ensaya con el 40 % restante. Los errores de ensayo para ambas redes neuronales (ver 4.2.1) se pueden observar en las siguientes figuras.

Company Name	LR	MR	Total Net Error	Total Mean Squared Error
Tata Consultancy Services Ltd	0.4	0.5	0.00035	0.00101
Infosys Technologies Ltd	0.4	0.4	0.00117	0.00382
Dr. Reddy's Laboratories Ltd	0.2	0.5	0.00036	0.00069
Sun Pharmaceutical Ltd	0.2	0.3	0.00030	0.00114

Figura 3.6: Errores de ensayo para NN1
Fuente: [A. Victor Devadoss, 2013]

Company Name	LR	MR	Total Net Error	Total Mean Squared Error
Tata Consultancy Services Ltd	0.5	0.5	0.00032	0.00045
Infosys Technologies Ltd	0.4	0.3	0.00125	0.00330
Dr. Reddy's Laboratories Ltd	0.7	0.2	0.00062	0.01033
Sun Pharmaceutical Ltd	0.7	0.5	0.00018	0.00187

Figura 3.7: Errores de ensayo para NN2
Fuente: [A. Victor Devadoss, 2013]

Los resultados predichos muestran que la red neuronal artificial ha sido capaz de predecir los valores de las acciones. En este artículo destacan que, para mejorar la precisión del modelo, los indicadores de análisis técnico (ver subsección 4.1.1) podrían utilizarse en las variables de entrada y para mejorar el rendimiento de la red.

Capítulo 4

Conjunto de datos

En este capítulo describiremos los datos de los que partimos y las transformaciones a realizar para que sean útiles como fuentes de predicción. Además, comentaremos las distintas formas de evaluar los errores de los métodos de predicción utilizados.

4.1 Descripción de los datos

A continuación, describiremos nuestros datos para una mejor comprensión de los métodos de predicción que vamos a utilizar. Comenzaremos describiendo los indicadores de análisis técnicos, llamados *OHLCV* (siglas del inglés open-high-low-close-volume), que tienen nuestros índices de la bolsa; para más tarde describir los propios índices de los que tenemos esa información.

4.1.1 Indicadores bursátiles

La elección de técnicas por parte de los *traders* (quienes compran y venden bienes, divisas o acciones) es crucial en la evaluación del estado y la predicción del comportamiento futuro de los índices bursátiles. Frecuentemente, los indicadores utilizados para la realización de estas técnicas son *open* (precio con el que se inician las transacciones en una sesión bursátil) y *close* (precio con el que finalizan). Frecuentemente, para un día concreto, el valor de *open* es similar al valor de *close* del día anterior, de existir una gran diferencia puede ser debido a algún acontecimiento puntual que justifique este cambio.

Los indicadores *high* (valor más alto que ha alcanzado en la sesión) y *low* (valor más bajo que ha alcanzado), no son tan representativos a la hora de entender el comportamiento de los índices, por lo que en la mayoría de las ocasiones no son utilizados. Con lo cual, no nos serán de gran utilidad en los métodos de aprendizaje automático que utilizaremos.

El indicador *volume* (cantidad de instrumentos financieros o acciones intercambiadas en la sesión bursátil), si bien no es tan utilizado en la predicción del comportamiento futuro de los índices, sí estará incluido en nuestros métodos de predicción, ya que esconde propiedades relacionadas con el precio real de los índices que podemos aprovechar. Por ejemplo, un volumen alto con un precio que sigue una tendencia alcista nos indica la fiabilidad del movimiento, nos confirma la tendencia. Por el contrario, un volumen alto y un precio bajo,

nos advierte de movimientos no fiables y de una posible reversión del valor del índice. Es decir, el volumen puede ser un precursor de los cambios en los precios, ya que permite medir la fuerza de una tendencia.

Cabe destacar que no existen en nuestro conjunto de datos indicadores sobre volúmenes para todos los índices, pero sí para la mayoría.

4.1.2 Tabla descriptiva

Nuestro rango de datos abarca 10 años, desde el 12 de julio de 2008 hasta el 23 de septiembre de 2018. Sin embargo, tenemos un total de 2631 días, ya que carecemos de los datos de los lunes y martes de cada semana.

A continuación, podemos observar las tablas con la descripción de cada valor bursátil de nuestros datos. Se encuentran diferenciados en tres tipos:

- *Curncy* (abreviatura del inglés *currency*), que nos indica que se trata de una moneda (ver Cuadro 4.2).

- *Index*, que nos indica que es un índice (ver Cuadro 4.3).

- *Comdty* (abreviatura del inglés *commodity*), que nos indica que es una mercancía (ver Cuadro 4.1).

Índice	**Descripción**
LMAHDS03 Comdty	Aluminio alta pureza a 3 meses
LMCADS03 Comdty	Cobre 3 meses
CO1 Comdty	Primer futuro brent (petróleo)
CL1 Comdty	Primer futuro West Texas
XAu Comdty	Primer futuro oro
XAG Comdty	Primer futuro plata
C 1 Comdty	Primer futuro maíz
w 1 Comdty	Primer futuro trigo
S 1 Comdty	Primer futuro soja
MO1 Comdty	Primer futuro CO2
TY1 Comdty	10 Year T-Note Futures (primer futuro sobre bono americano a 10 años)
FV1 Comdty	5 Year T-Note Futures (primer futuro sobre bono americano a 5 años)
TU1 Comdty	2 Year U.S. T-Note (primer futuro sobre bono americano a 2 años)
RX1 Comdty	(primer futuro sobre bono alemán a 2 años)
OE1 Comdty	(primer futuro sobre bono alemán a 5 años)
DU1 Comdty	(primer futuro sobre bono alemán a 2 años)

Cuadro 4.1: Tabla descriptiva mercancías

Índice	**Descripción**
EUR Curncy	Euro
JPY Curncy	Yen japonés
GBP Curncy	Libra esterlina
CHF Curncy	Franco suizo
CAD Curncy	Dolar canadiense
NOK Curncy	Corona noruega
HG1 Comdty	Primer futuro cobre COMEX

Cuadro 4.2: Tabla descriptiva monedas

Índice	Descripción
INDU Index	Dow Jones Industrial Average (30 mayores sociedades anónimas que cotizan en el mercado bursátil de EEUU)
SPX Index	Standard & Poor's 500 (500 grandes empresas que cotizan en las bolsas NYSE o NASDAQ)
CCMP Index	NASDAQ Composite (más de 5000 empresas que cotizan en las bolsas NASDAQ)
SPTSX Index	Renta variable de Canadá
MEXBOL Index	Bolsa Mexicana de Valores
UKX Index	Bolsa de Valores de Londres (100 compañías de mayor capitalización bursátil del Reino Unido)
CAC Index	CAC 40 (40 valores más significativos de las empresas que cotizan en la Bolsa de París)
DAX Index	DAX PERFORMANCE-INDEX (30 compañías más grandes de Alemania que cotizan en la Bolsa de Fráncfort)
IBEX Index	Ibex 35 (35 empresas con más liquidez que cotizan en el SIBE)
FTSEMIB Index	FTSE MIB (350 compañías que cotizan en la Bolsa Italiana)
OMX Index	OMX Stockholm 30 (30 valores que cotizan en la Bolsa de Estocolmo)
NKY Index	Nikkei 225 (225 valores más líquidos que cotizan en la Bolsa de Tokio)
HSI Index	Hang Seng (225 valores más líquidos que cotizan en la Bolsa de Hong Kong)
SHSZ300 Index	Shanghai Shenzhen CSI 300 (300 que cotizan en las bolsas de Shanghai y Shenzhen)
AS51 Index	ASX 200 (200 compañías que cotizan en la Bolsa de Australia)
GSPG2YR Index	Spanish Govt Generic Bonds 2Yr Note (primer futuro sobre bono español a 2 años)
GSPG5YR Index	Spanish Govt Generic Bonds 5Yr Note (primer futuro sobre bono español a 5 años)
GSPG10YR Index	Spanish Govt Generic Bonds 10Yr Note (primer futuro sobre bono a español 10 años)
EUR003M Index	Euribor 3 meses
EUR006M Index	Euribor 6 meses
EUR012M Index	Euribor 12 meses
CB3 Govt	Libor 3 meses
CB6 Govt Index	Libor 6 meses
CB12 Govt Index	Libor 12 meses
BDIY Index	Baltic dry (media del precio del transporte por mar de las principales materias primas sólidas y a granel)
ECCPEMUY Index	IPC armonizado europa (índice de precios al consumo armonizado europa)
CRY Index	Thomson Reuters/CoreCommodity CRB Commodity Index

Cuadro 4.3: Tabla descriptiva índices

4.2 Errores de predicción

El objetivo de este trabajo es comparar diferentes métodos de predicción para establecer cuál de ellos es el mejor y qué parámetros los más adecuados. A continuación, se comentan las características de cada error y cuáles son a las que se les va a dar mayor prioridad. Buscamos una medida de error con tres características:

1. Permitir comparar distintas alternativas (elección de columnas, etc). Para esto es importante que o bien el error no dependa del rango de la columna elegida o, si esto no es posible, reescalar los datos para que todas las columnas tengan el mismo rango y permitan así la comparación.

2. Permitir comparar varios métodos entre sí. Por ejemplo, el coeficiente de correlación R^2, es útil para comparar distintas regresiones, pero no para comparar con otros métodos de predicción.

3. El error nos dé una idea de la ganancia/pérdida en una compra de valores real, a ser posible con intervalos de confianza.

4.2.1 Medidas de error típicas

Seguidamente, se encuentra una breve descripción de las medidas de error más utilizadas en modelos de aprendizaje y cuáles son las propiedades de cada una, junto con la fórmula que indica cómo se calcula.

- **RMSE** (Error cuadrático medio, acrónimo del inglés *Root Mean Squared Error*), es una medida estadística frecuentemente usada. Proporciona información acerca de lo bien que se ajusta un modelo a partir de las diferencias entre valores predichos y los valores observados.
  Como podemos observar en el siguiente cuadro, se trata del promedio de los errores al cuadrado, ya que lo interesante de esta medida es hacer notables los errores grandes (permite descartar modelos que con otras métricas pasarían por desapercibidos) y, además, evitar que se cancelen las desviaciones positivas y negativas entre sí. Este valor es útil en la comparación entre distintos modelos, siendo mejor el que tenga un valor de RMSE menor.

$$RMSE = \frac{1}{n}\sqrt{\sum_{i=0}^{n}(Y_i - \hat{Y}_i)^2}$$

Y_i = valor observado
$\hat{Y}_i$ = valor pronosticado
n = número de valores

  Una variable del RMSE es **NRMSE** (acrónimo del inglés de *Normalized Root Mean Squared Error*) como su propio nombre indica es la normalización del RMSE, útil cuando se necesita comparar el RMSE de dos conjuntos de datos que están comprendidos en distinto rango de valores, y se expresa como un porcentaje, la siguiente tabla puede servir como orientación para interpretarlo.

<10 %	Excelente
10 %-20 %	Buena
20 %-30 %	Adecuada
>30 %	Pobre

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}} \text{ o } \frac{RMSE}{\overline{y}}$$

y_{min} = valor mínimo del conjunto de datos
y_{max} = valor máximo del conjunto de datos
$\overline{y}$ = media del conjunto de datos

- **MAE** (Error medio absoluto, acrónimo del inglés *Mean Absolute Error*), se refiere a la media de las desviaciones de cada error de predicción. Se utiliza a menudo en series temporales, donde los datos deben estar equiespaciados. Se entiende que el modelo es mejor cuanto más se acerque a 0 el valor del MAE.

$$MAE = \frac{1}{n}\sum_{i=0}^{n}|Y_i - \hat{Y}_i|$$

Y_i = valor observado
$\hat{Y}_i$ = valor pronosticado
n = número de valores

- $\boldsymbol{R^2}$ (R-cuadrado o R-squared), indica cómo de cerca están los datos de la función de regresión, es decir, qué porcentaje del movimiento se explica a través de esta función. Si se acerca a 0 indica que el modelo no explica la variabilidad de los datos en torno a su media. Por el contrario, si el valor se acerca al 1 el modelo sí que explicaría la variabilidad de los datos en torno a su media.

$$R^2 = 1 - \frac{SS_{regresin}}{SS_{total}} = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y})^2}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}$$

$SS_{regresin}$ = suma cuadrada del error de la regresión
SS_{total} = suma cuadrada del error total
$\hat{Y}_i$ = valor pronosticado
$\overline{Y}$ = media de Y
Y = valor observado

- $\boldsymbol{R^2}$ **ajustado** (R-cuadrado ajustado o *R-squared adjusted*), identifica el porcentaje de varianza que se explica con la función de regresión considerando las variables de entrada, es decir, tiene en cuenta el número de variables incluidas en el modelo.

Mientras que R^2 sube en porcentaje con la inclusión de variables, R^2 ajustado no tiene por qué hacerlo, depende de si esas variables nuevas mejoran el modelo. Puede ser útil para descartar o admitir variables nuevas en la predicción. Al igual que en R^2, la predicción es mejor cuanto más se acerca a 1.

$$R^2{}_{ajustado} = 1 - \frac{n-1}{n-p} \cdot (1 - R^2)$$

R^2 = R-cuadrado
n = tamaño total de la muestra
p = número de predicciones

4.2.2 Comparativa entre las medidas

RMSE tiene el beneficio de penalizar los grandes errores, por lo que puede ser más apropiado en algunos casos. Por ejemplo, un error de 10 se interpretaría como más de dos veces peor que si fuera de 5. En cambio, en el caso del **MAE** un error de 10 significaría el doble que uno de 5.

Por lo tanto, el **RMSE** sería el más apropiado para comparar modelos si nos preocupan mucho los grandes errores, en cambio el **MAE** sería el más apropiado si lo que queremos es acercar a la realidad una pérdida o una ganancia real (e.g. cuando el error es de 20 € podemos interpretarlo como exactamente dos veces peor que el error de 10 € en la realidad).

En nuestro caso, como queremos minimizar los grandes errores, ya que hablamos de inversiones que supondrían grandes pérdidas, es más conveniente usar el RMSE.

Como ya hemos nombrado, entre R^2 y R^2 ajustado existe una gran diferencia, ya que a ambos les afecta la inclusión de nuevas variables independientes de distinta manera. Cuando se da una condición de sobreajuste, se obtiene un valor incorrectamente alto de R^2, que conduce a una capacidad disminuida de predecir. Es por esto que el R^2 ajustado es la mejor estimación del grado de relación entre la función de regresión y los valores observados.

4.3 Preparación de los datos para los métodos de predicción basados en aprendizaje automático

Con el fin de predecir los valores en un horizonte futuro f, de un valor *label*, a partir del pasado de otros valores $v1$, $v2$, ..., vamos a crear una tabla con la siguiente estructura:

label-inc	v1-inc(p-1)	...	v1-inc(1)	v2-inc(p-1)	...	v2-inc(1)	...

donde:

- La primera columna, *label-inc* representa el incremento del valor que queremos predecir desde el día actual hasta un futuro de f días.

- Para cada valor vi que se va a utilizar para predecir este incremento a futuro, usaremos sus incrementos diarios desde el pasado $p-1$ hasta el día anterior.

A continuación, se muestran una serie de figuras con las que se explica (como ejemplo) paso a paso las etapas que llevan a construir el fichero con este formato.

	label	dimension1	dimension2	dimension3
0	51	7	8	28
1	14	60	32	77
2	97	46	44	88
3	54	64	19	67
4	78	15	41	91
5	83	40	46	87
6	30	61	11	38
7	55	88	24	84
8	53	7	49	40
9	63	4	62	68
10	4	33	73	44
11	40	49	81	24
12	25	73	35	2
13	10	4	12	96
14	65	71	61	21
15	52	21	56	11
16	53	40	26	12
17	66	93	51	27
18	8	6	100	28
19	18	55	14	15

Figura 4.1: Ejemplo de datos originales; *label* es la columna a predecir, *dimension1*, *dimension2*, *dimension3* las columnas desde las que se quiere predecir
Fuente: Elaboración propia

En la Figura 4.1 se puede ver un ejemplo de fichero original, en el que señalamos con nombre *label* la columna a partir de la cual se obtienen los incrementos que queremos predecir. El resto de las columnas se tratan de los valores $v1$, $v2$, ... con los que se hará la predicción.

Para este ejemplo fijaremos el uso de 3 valores del pasado ($p = 3$) para las columnas v, para predecir lo que pasará en *label* dentro de 5 días ($f = 5$), todo esto considerando que tenemos un fichero conteniendo los datos históricos de 20 días ($n = 20$).

	Presente(p:n-1-f)	Futuro(p+f:n-1)
0	54	53
1	78	63
2	83	4
3	30	40
4	55	25
5	53	10
6	63	65
7	4	52
8	40	53
9	25	66
10	10	8
11	65	18

Figura 4.2: Columnas intermedias para calcular el incremento en *label*
Fuente: Elaboración propia

En esta segunda figura (ver Figura 4.2), se muestran las dos columnas con las que se calculará el incremento en tanto por uno para la columna *label* que queremos predecir. Como podemos observar, los valores de la columna naranja corresponden con los de la columna *label* en la Figura 4.1 desde la posición 3 hasta la 14. La razón por la cual la primera posición es la 3 es debido a que vamos a usar los valores para predecir de $p = 3$ valores pasados, por lo tanto, tenemos que tener un margen de p pasados; más adelante se explicará de qué modo se toma esta información del pasado. La última fila tomada es la de la posición 14 porque, análogamente, el último incremento de f días que se puede calcular tiene que dejar fuera f días.

De forma más general, este rango irá desde la fila p hasta la fila $n - 1 - f$ (es decir, un total de $n - f - p$ valores).

La columna azul corresponde con el valor de la columna *label* de la Figura 4.1 que se encuentra f días después que su correspondiente en la columna *presente* (columna naranja). Así, teniendo un valor en presente que corresponde con el día 3 (día p), el valor de futuro corresponde con el día $3+5$ ($p+f$). Por tanto, esta columna abarcará los valores de los días desde $p + f$ hasta $n - 1$ (por el motivo antes mencionado).

label	dimension1-pasado2	dimension1-pasado1
-0.01851852	7.57142857	-0.23333333
-0.19230769	-0.23333333	0.39130435
-0.95180723	0.39130435	-0.765625
0.33333333	-0.765625	1.66666667
-0.54545455	1.66666667	0.525
-0.81132075	0.525	0.44262295
0.03174603	0.44262295	-0.92045455
12	-0.92045455	-0.42857143
0.325	-0.42857143	7.25
1.64	7.25	0.48484848
-0.2	0.48484848	0.48979592
-0.72307692	0.48979592	-0.94520548

Figura 4.3: Tabla ya preparada para predecir mediante aprendizaje automático
Fuente: Elaboración propia

A partir de la Figura 4.3 ya se pueden aplicar los diferentes métodos de aprendizaje automático. En particular, el incremento de la columna *label* se obtiene mediante la siguiente fórmula:

$$label_i = (Futuro_i - Presente_i)/Presente_i$$

donde los valores *Presente* y *Futuro* son los de la Figura 4.2. Recordamos que este es el incremento que intentaremos predecir.

Las otras dos columnas que aparecen en la figura se han obtenido de la siguiente manera: volvamos por un momento a la Figura 4.1, donde explicamos que, para predecir el incremento que tiene lugar entre el día 3 y 5, debíamos dejar un margen de 3 (p) para usar la información del pasado. Pues bien, esa información del pasado para el primer incremento se encuentra en esos tres días (valores 0, 1, 3). En la Figura 4.3 se recoge la información de los incrementos entre días consecutivos. Por lo tanto, en una columna tendríamos el incremento entre los valores de los días 0 y 1 (la columna encabezada por el nombre *dimension1-pasado2*), y en la otra el incremento entre los días 1 y 2 (columna *dimension2-pasado1*), y responde a la fórmula:

$$dimension_j\_pasado_i = (dimension1_{i+j} - dimension1_{i+(j-1)})/dimension1_{i+(j-1)}$$

En el capítulo sobre cómo se han aplicado técnicas de regresión a nuestros datos (ver Capítulo 6), se continuará explicando el proceso iterativo con el que se eligen la columna o columnas más adecuadas. Hay que señalar que esta idea de utilizar los incrementos de

f días para *label* y de días consecutivos para el resto de las columnas se ha obtenido tras varias pruebas de ensayo y error, eligiendo esta propuesta por ser la que tiene menor tasa de error al predecir.

Capítulo 5

Predicción con ARIMA

En este capítulo analizamos un primer método de predicción para los valores de los mercados que no está basado en aprendizaje automático. Se trata del método ARIMA (modelo autorregresivo integrado de promedio móvil, acrónimo del inglés *Autoregressive Integrated Moving Average*) [Mills and Mills, 1991]. Comenzamos describiendo el método en general para pasar después a la aplicación a nuestro caso. Finalmente, discutiremos los resultados obtenidos y hablaremos de las desventajas de este método.

5.1 ARIMA como método de predicción

Desarrollado a finales de los sesenta y sistematizado por Box y Jenkins en 1976, ARIMA se trata de un modelo dinámico de series temporales [Das, 1994], donde las estimaciones de una variable se pronostican directamente **a partir de los datos del pasado de esa misma variable y no de otras variables independientes** (como ocurre en otros métodos de predicción como la regresión que utilizaremos más tarde) con el fin de encontrar patrones para la predicción de un futuro.Existen modelos más complejos que extienden ARIMA [Peter and Silvia, 2012] para la inclusión de varias variables de entrada (llamados ARIMAX), sin embargo, no se consideran en este trabajo.

Este método consiste en una combinación de: valores que corresponden a periodos anteriores (que denotaremos Y_{t-1}, ..., Y_{t-p}), errores ponderados adecuadamente que corresponden a periodos anteriores (que denotaremos ε_{t-1}, ..., ε_{t-q}) más ruido blanco (ε_t) y una constante (c).
Para establecer cuántos periodos anteriores se van a usar, se tiene el parámetro **p**, y para establecer cuántos errores de periodos anteriores se usa el parámetro **q**. Finalmente, solo queda mencionar el parámetro **d**, que corresponde con el número de diferencias que se deben de aplicar a los datos para que sea una serie estacionaria, lo que es un requisito indispensable.

Se establecen los parámetros de la manera ARIMA(p, d, q), a continuación se encuentran las fórmulas que se aplican para cada parámetro.

- **AR(p)**, componente de auto-regresión. Para ARIMA(p,0,0) tendríamos lo siguiente:

$$Y_t = c + \sum_{i=1}^{p} \varphi_i \cdot Y_{t-i} + \varepsilon_t$$

Y_t = variable a predecir en el tiempo t
c = constante
φ = coeficiente de cada parámetro p
ε_t = ruido blanco en tiempo t

- **I(d)**, componente integrado. Tenemos que, para el componente AR(p), en lugar de y_t, se utilizaría la diferencia Δy_t, siendo $\Delta y_t = y_t - y_{t-1}$. Para ARIMA(p,d,0) con $d \neq 0$, tendríamos lo siguiente:

$$\Delta Y_t = c + \sum_{i=1}^{p} \varphi_i \cdot \Delta Y_{t-i} + \varepsilon_t$$

Y_t = variable a predecir en el tiempo t
c = constante
φ = coeficiente de cada parámetro p
ε_t = ruido blanco en tiempo t

- **MA(q)**, componente de promedios móviles. Para ARIMA(0,0,q) tendríamos lo siguiente:

$$Y_t = c + \varepsilon_t - \sum_{i=1}^{q} \phi_i \cdot \varepsilon_{t-i}$$

Y_t = variable a predecir en el tiempo t
c = constante
ϕ = coeficiente de cada parámetro q
ε_t = ruido blanco en tiempo t

5.1.1 Etapas a realizar

1. **Estudio de la estacionariedad**
   ARIMA requiere una serie estacionaria, es decir, que su media, varianza y covarianza no dependan del instante observado.

   Las series estacionarias son más fáciles de predecir en el tiempo: si sabemos cómo se comportaban los datos en el pasado podemos suponer que se comportarán de una manera similar en el futuro.

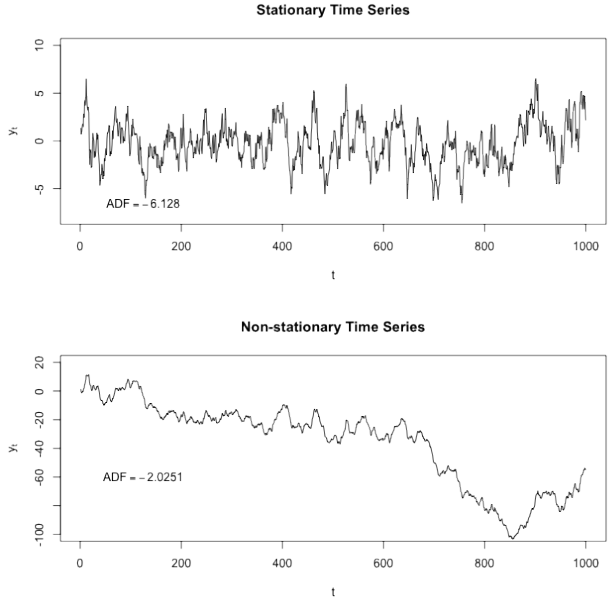
Creemos oportuno indicar las siguientes definiciones que distinguen entre estacionariedad y estacionalidad, ya que suelen confundirse debido a la ambigüedad en la traducción al castellano de las palabras *stationary* y *seasonality* del inglés.

Definiciones

Estacionariedad
Una serie es estacionaria cuando el cambio que sufre la media, varianza y covarianza dependiendo del instante de tiempo es insignificante.

Estacionalidad
Una serie es estacional cuando experimenta fluctuaciones o cambios regulares a lo largo del tiempo.

Una serie que presenta estacionalidad puede también ser una serie estacionaria, es decir no son características excluyentes.



Fuente: Protonk at English Wikipedia, CC BY-SA 3.0

Figura 5.1: Serie estacionaria vs. serie no estacionaria

Como podemos observar en la Figura 5.1, la primera gráfica se trata de una serie estacionaria, ya que tendría una media prácticamente constante sin importar el instante de tiempo en el que se mida, en cambio la segunda tiene varias tendencias durante largos periodos de tiempo y la media y varianza tomada en cualquier instante no es constante.

Aunque haciendo la gráfica de los datos podemos intuir si la serie es estacionaria o no,

es conveniente utilizar otros métodos como la **prueba aumentada de Dickey-Fuller (ADF)**, que se lleva a cabo como se describe a continuación.

La prueba ADF tiene la hipótesis nula de no estacionariedad, es decir, se debe rechazar la hipótesis para poder decir que nuestra **serie es estacionaria (p-valor < 0.05)**

Ejemplo en R. Serie no estacionaria

```
> adf.test(serie)

        Augmented Dickey-Fuller Test

data: ts(euro)
Dickey-Fuller = -3.1325, Lag
order = 13, p-value = 0.09954
alternative hypothesis: stationary
```

Si nuestra serie no es estacionaria hay tres formas habituales de hacer que sí lo sea; la primera es hacer **diferencias regulares**, si el problema de la serie es la tendencia (rara vez se tiene que hacer más de una), la segunda son las **diferencias estacionales**, si la serie tiene estacionalidad (que no es lo mismo que estacionariedad) y la tercera es **aplicar logaritmos** para estabilizar la varianza.

Otra manera de ver si una serie es estacionaria o no es haciendo las gráficas de la función de autocorrelación ACF (explicada más adelante), si esta muestra un decrecimiento lento de la función de autocorrelación indica la no estacionariedad.

2. **Determinar los parámetros del modelo**
En ocasiones, un valor que toma la serie depende de los valores anteriores, el número de estos que son significativos se establece mediante los parámetros AR(p) y MA(q). Vamos a ver dos formas de darles valor:

   - Utilizar la función `auto.arima()`, la cual prueba combinaciones de parámetros y selecciona el conjunto óptimo (cuyos parámetros están dentro de un rango de prueba).
   - Utilizar las gráficas de la función de autocorrelacción (ACF) y la función de autocorrelacción parcial (PACF).

La función **ACF** mide la correlación de dos variables separadas por k periodos y la función **PACF** indica lo mismo con la diferencia de que no considera la correlación entre los retardos intermedios. Sus gráficas pueden ayudarnos a determinar el valor para p y q. El retraso donde un retardo corta en la gráfica ACF es el valor indicado para q en MA(q) y lo mismo ocurre en la gráfica PACF para determinar el valor de p para AR(p).

Mostrar ACF y PACF en R.

```
> Acf(serie)
> Pacf(serie)
```

3. **Evaluar el modelo**
   Para evaluar cómo de bueno es nuestro modelo se pueden examinar las gráficas ACF y PACF con los valores residuales, en las cuales no debe de haber correlación, de haberla implicaría que ese patrón no ha sido capturado por el modelo. Con lo cual, se debe intentar que las correlaciones en los valores residuales sean lo más insignificantes posibles.

   Ejemplo en R. Mostrar gráficas ACF y PACF de valores residuales.

   ```
   > tsdisplay(residuals(modelo-ARIMA))
   ```

   Otra manera para evaluar cómo de bueno es un modelo frente a otro es comparar los índices AIC (Criterio de Información de Akaike [Akaike, 1998]) y BIC (Criterio de Información Bayesiano [Chen and Chen, 2008]), que indican la cantidad de información que se perdería en ese modelo, por lo que se tienen que intentar minimizar.
   La manera en que AIC y BIC lo hacen es midiendo la capacidad explicativa de un modelo y penalizando por su grado de complejidad. La fórmula general de estos índices es $xIC = (Complejidad - Bondad\ de\ ajuste)$, donde la complejidad es el número de parámetros y la bondad de ajuste corresponde al valor de máxima verosimilitud.
   Estos índices se diferencian en lo siguiente, el AIC selecciona modelos más complejos (poco generales) y BIC penaliza más la complejidad (modelos más sencillos, predicciones con menor detalle).
   Las etapas 2 y 3 se deben iterar tantas veces como se necesite hasta dar con el mejor modelo.
4. **Generalización del modelo**
   Se van a usar diferentes cantidades de datos para el entrenamiento (2631, 1950, 1300 y 650 primeros datos) para elegir el modelo más apropiado a través de la comparación de los índices BIC y AIC de cada subconjunto que nos devuelve ARIMA. La elección para el modelo final será en función de la moda.

   Obtener AIC y BIC en R.

   ```
   > summary(modelo-ARIMA)
   ```

5.2 Elección de columnas más adecuadas

A continuación, se puede ver una tabla ordenada. En la primera columna se encuentra el nombre de todos los valores de apertura que contiene nuestra base de datos, en la segunda el **error de entrenamiento** sin escalar resultado de aplicar `auto.arima()` a esa columna de datos, y en la tercera este mismo error pero escalando los datos.

Vamos a utilizar esta tabla como punto de partida para encontrar los valores son los más asequibles a la hora de predecir con el método ARIMA, aunque mejoraremos el modelo configurando los distintos parámetros a partir del procedimiento anteriormente descrito.

Debemos tener en cuenta que este no es el error calculado con las predicciones, solo corresponde a el error de entrenamiento de un modelo, por lo que no se pueden comparar estos resultados con los que provengan de hacer predicciones reales.

Índice	RMSE escalado	RMSE sin escalar
BDIY_Index_Open	0.02375	29.94270
CCMP_Index_Open	0.02653	43.03786
SPX_Index_Open	0.02894	15.66290
INDU_Index_Open	0.03111	141.20914
RX1_Comdty_Open	0.03514	0.55632
NKY_Index_Open	0.03906	180.46540
DAX_Index_Open	0.04048	104.16873
DU1_Comdty_Open	0.04140	0.08006
OE1_Comdty_Open	0.04372	0.28923
MEXBOL_Index_Open	0.04629	366.84633
GSPG10YR_Index_Open	0.04716	0.07834
JPY_Curncy_Open	0.04716	0.65033
CRY_Index_Open	0.04782	2.80881
NOK_Curncy_Open	0.04853	0.05469
OMX_Index_Open	0.04973	14.07009
GSPG5YR_Index_Open	0.05114	0.08936
CAD_Curncy_Open	0.05468	0.00725
CO1_Comdty_Open	0.05492	1.48139
XAu_Comdty_Open	0.05997	13.99842
GSPG2YR_Index_Open	0.06056	0.09345
CL1_Comdty_Open	0.06552	1.52719
AS51_Index_Open	0.06671	44.18490
EUR_Curncy_Open	0.06779	0.00835
XAG_Comdty_Open	0.06782	0.49821
UKX_Index_Open	0.06855	62.79958
LMCADS03_Comdty_Open	0.06917	100.11340
SPTSX_Index_Open	0.06941	128.62723
C_1_Open	0.06992	9.90138
MO1_Comdty_Open	0.07058	0.33497
GBP_Curncy_Open	0.07080	0.00986
HG1_Comdty_Open	0.07226	4.77475
CAC_Index_Open	0.07620	54.10814
HSI_Index_Open	0.08040	292.28105
LMAHDS03_Comdty_Open	0.08112	26.33943
SHSZ300_Index_Open	0.08235	53.19239
S_1_Open	0.08326	18.38303
TU1_Comdty_Open	0.08406	0.09453
FV1_Comdty_Open	0.09103	0.28206
TY1_Comdty_Open	0.09442	0.47651
CHF_Curncy_Open	0.09469	0.00708
w_1_Open	0.09549	12.32257
FTSEMIB_Index_Open	0.10402	323.24212
IBEX_Index_Open	0.11720	145.81189

Cuadro 5.1

Vamos a considerar los índices bursátiles BDIY, CCMP, GSPG10YR, RX1, NKY, DU1 y NOK debido a su asequible predicción con ARIMA y su interés (ver tablas 4.1, 4.2 y 4.3).

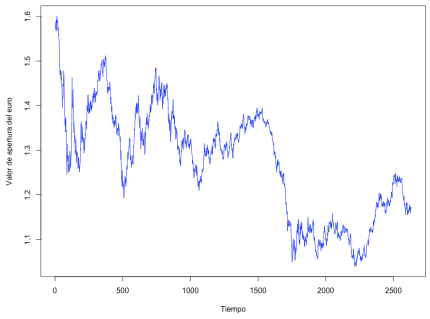
5.3 Resultados de ARIMA para nuestro conjunto de datos

Comenzaremos realizando en detalle las etapas de ARIMA con el valor de apertura del euro, que servirá como ejemplo de metodología del proceso de obtención de un modelo adecuado de ARIMA para comprender los modelos encontrados para los distintos índices que realmente se van a considerar (ver apartado 5.2). A continuación, se mostrarán los resultados para los índices elegidos y se analizarán en profundidad.

5.3.1 Ejemplo de metodología

A continuación, se van a realizar las etapas para crear un modelo con ARIMA para el valor de apertura del euro utilizando todos los datos de los que disponemos (2631 días), como ejemplo de metodología a seguir. Cabe destacar que los datos han sido escalados previamente.

1. **Estudio de la estacionariedad**
   Vamos a ver si la serie del valor de apertura del euro (ver figura 5.2) es estacionaria o no. Lo primero que debemos hacer es observar la gráfica del valor con respecto al tiempo. Aparentemente no es estacionaria, pero para asegurarnos utilizaremos la prueba ADF descrita en el apartado 5.1.1.



Fuente: Elaboración propia

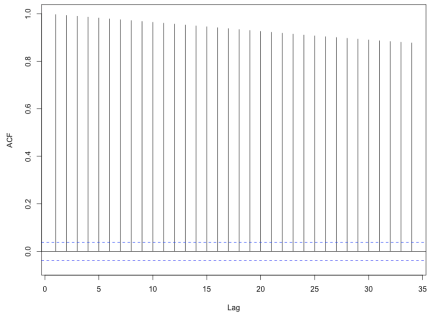
Figura 5.2: Valores de apertura del euro

Para ello utilizamos la función `adf.test()` de R. El resultado es el siguiente:

```
        Augmented Dickey-Fuller Test

data: ts(euro)
Dickey-Fuller = -3.1325,
Lag order = 13, p-value = 0.09954
alternative hypothesis: stationary
```

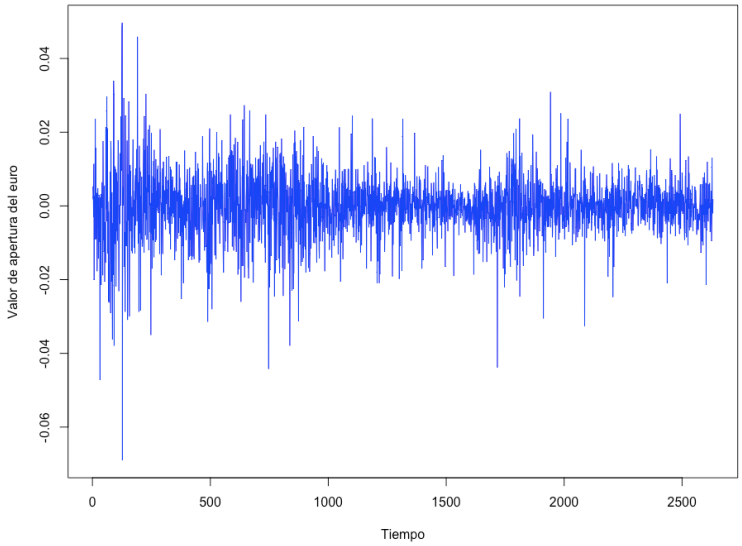
Como podemos observar p-value = 0,09954. Como 0,09954 > 0,05 la serie no es estacionaria. Luego, debemos aplicar diferencias para que sí lo sea. Como comentamos, otra forma de ver si es estacionaria es observando el comportamiento de la gráfica ACF. Podemos ver en la Figura 5.3 que el decrecimiento de la gráfica ACF no es en absoluto rápido, con lo que podemos concluir que la serie no es estacionaria y necesitaremos aplicar diferencias.



Fuente: Elaboración propia

Figura 5.3: Gráfica ACF de la apertura del euro sin diferencias

Para el cálculo de diferencias utilizaremos el valor d de la componente I(d) del método de ARIMA, pero debemos asegurarnos de cuántas diferencias aplicamos. De nuevo, realizaremos las pruebas anteriores para el valor de la apertura del euro con una sola diferencia, que como indicamos anteriormente suele ser más que suficiente.



Fuente: Elaboración propia

Figura 5.4: Valores de apertura del euro con diferencias

Podemos intuir (ver Figura 5.4) que la nueva serie sí va a ser estacionaria. Nos aseguraremos realizando la prueba ADF sobre nuestra nueva serie.

```
 adf.test(ts(euro),alternative="stationary")

       Augmented Dickey-Fuller Test

data: diff(ts(euro))
Dickey-Fuller = -14.014,
Lag order = 13, p-value = 0.01
alternative hypothesis: stationary
```

Ahora sí, nuestro p-value = 0,01, como 0,01 < 0,05, podemos asegurar que nuestra serie es estacionaria. Además, si observamos la gráfica de la ADF en la figura 5.5a, lo podemos reafirmar, ya que la función decrece rápidamente.

Podemos concluir con este experimento que para el valor de apertura del euro necesitaremos realizar una diferencia, o lo que es lo mismo, igualar el parámetro de I(d) a 1.

2. **Determinar el tipo de modelo**
   Para determinar el valor más indicado para p y q en las componentes AR y MA respectivamente, en primer lugar, utilizamos la función `auto.arima()` en R.

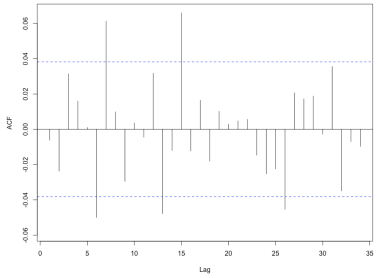
```
 Series: euro
ARIMA(0,1,0)

sigma^2 estimated as 6.976e-05: log likelihood= 8853.28
AIC= -17704.57 AICc= -17704.57 BIC= -17698.69
```

Este método ha determinado que los valores para q y para p más indicados son 0. En el caso de la diferencia, como ya sabíamos en el apartado anterior, la mejor opción es $d = 1$. Sin embargo, debemos tener en cuenta que el método `auto.arima()` únicamente prueba los valores $0 \leq p, q \leq 5$, con lo que es probable que haya otros valores para p y q que sean más apropiados.

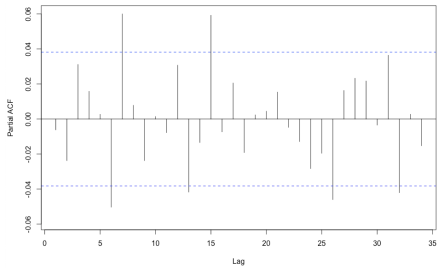
Para ello, en primer lugar observamos las ACF y PACF de nuestros valores con una diferencia (ver Figura 5.5).

Figura 5.5: Gráficas ACF y PACF del euro con diferencias



Fuente: Elaboración propia

(a) ACF de la apertura del euro con diferencias



Fuente: Elaboración propia

(b) PACF de la apertura del euro con diferencias

En ambas gráficas el eje de las abscisas corresponde al número de periodos anteriores que se toman y el de ordenadas indica la correlación que se ha encontrado. Podemos apreciar en ambas unas líneas discontinuas horizontales, las cuales indican el límite a partir del cual la correlación se puede considerar significativa. Las gráficas ACF y PACF se diferencian en que para un valor $x = 5$ en la gráfica **ACF** se mediría **la correlación de un valor con los 5 anteriores** y en la gráfica **PACF** se mediría la correlación de **un valor con otro que se encuentra en el tiempo cinco periodos atrás.**

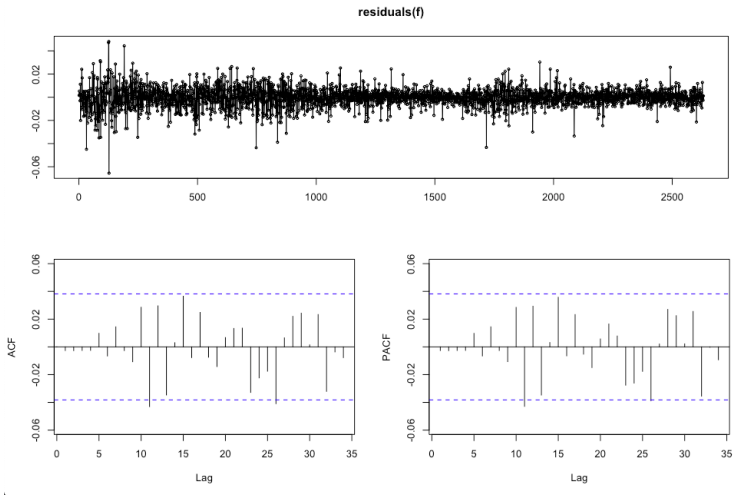
Sabemos que el valor donde corta en la gráfica ACF y en la PACF es el indicado para q y p respectivamente, con lo cual observando la figura 5.5a y la figura 5.5b obtendríamos $p = 6$ y $q = 6$, que saldría del rango con el que prueba `auto.arima()`. Por lo tanto, tendríamos un posible modelo ARIMA(6,1,6) que debemos evaluar.

3. **Evaluar el modelo**
   Para evaluar nuestro modelo ARIMA(6,1,6) la primera opción es la de observar las

gráficas ACF y PACF de los valores residuales tras aplicar el método (ver figura 5.6). En R, esto se traduciría de la siguiente manera.

```
m = #cardinal del conjunto de datos
h = #día del futuro a predecir
f <- fit <- Arima(ts(euro[1:m-h]), order=c(6,1,6))
residuals(f) #valores residuales obtenidos
```

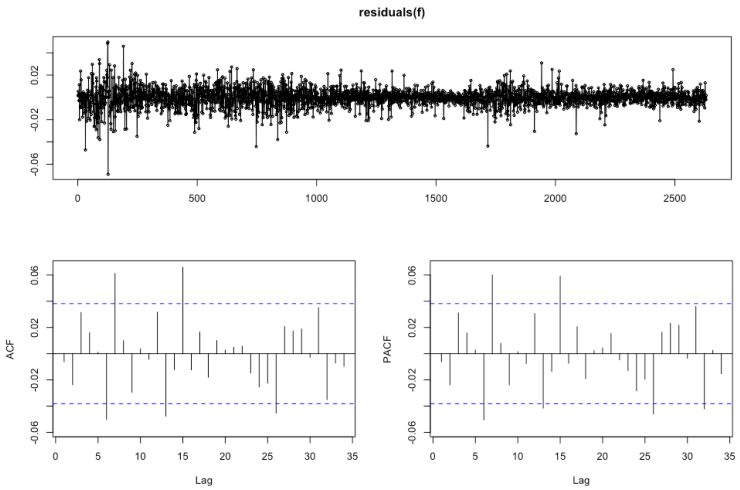
Podemos observar que apenas hay correlación en los valores residuales. Esto significa que nuestro modelo es bueno, ya que no existen patrones significativos en los valores residuales que no hayamos tenido cuenta con ARIMA(6,1,6).



Fuente: Elaboración propia

Figura 5.6: Gráfica, ACF y PACF de los valores residuales para ARIMA(6,1,6)

Cabe destacar que si nos hubiésemos conformado con los resultados dados por `auto.arima()`, que nos indicaba como mejor opción ARIMA(0,1,0), habría ocurrido lo contrario (ver figura 5.7), habríamos dejado patrones detectables sin incorporar a nuestro modelo.



Fuente: Elaboración propia

Figura 5.7: Gráfica, ACF y PACF de los valores residuales para ARIMA(0,1,0)

Como indicamos en el apartado 5.1.1, podemos ver cómo de bueno es nuestro modelo con respecto a otros comparando los valores de AIC y BIC que nos retorna ARIMA. Como sabemos que `auto.arima()` probó todas las combinaciones posibles con $0 \leq p, q \leq 5$ y la mejor (con menor AIC y BIC) fue ARIMA(0,1,0), sabemos que el resto combinaciones en ese rango son peores.

Para ARIMA(6,1,6) tenemos que AIC es -17714.25 y BIC es -17637.87, ambos valores (si nos fijamos en el apartado 2) son menores que los dados por el modelo generado por `auto.arima()`. Con lo que queda comprobado que nuestro modelo es el mejor encontrado hasta el momento. Podríamos seguir repitiendo este proceso para encontrar un modelo más preciso, pero podemos intuir a través de los experimentos que ARIMA(6,1,6) es un buen modelo de predicción.

Sin embargo, observando las gráficas PACF y ACF (ver figura 5.5), vemos que se debe probar con otros valores de p y q y posteriormente comparar sus BIC y AIC. Es decir, iteramos volviendo a las etapas 2 y 3 determinando los modelos ARIMA(6,1,6), ARIMA(7,1,7), ARIMA(6,1,7) y ARIMA(7,1,6).

4. **Generalización del modelo**

	AIC	BIC
	Desde 1 hasta 2631	
(0,1,0)	-17704.57	-17698.69
(6,1,6)	-17714.25	-17637.87
(7,1,6)	-17718.59	-17636.34
(6,1,7)	-17719.08	-17636.83
(7,1,7)	-17719.35	-17631.22
	Desde 1 hasta 1950	
(0,1,0)	-12769.63	-12764.05
(6,1,6)	-12775.55	-12703.07
(7,1,6)	-12783.15	-12705.1
(6,1,7)	-12783.28	-12705.23
(7,1,7)	-12783.75	-12700.13
	Desde 1 hasta 1300	
(0,1,0)	-8237.88	-8232.71
(6,1,6)	-8243.44	-8176.24
(7,1,6)	-8240.73	-8168.36
(6,1,7)	-8244.2	-8171.82
(7,1,7)	-8255.96	-8178.42
	Desde 1 hasta 650	
(0,1,0)	-3945.08	-3940.6
(6,1,6)	-3954.65	-3896.47
(7,1,6)	-3953.97	-3891.31
(6,1,7)	-3959.23	-3896.57
(7,1,7)	-3957.07	-3889.94

Determinamos a partir de la moda que el modelo que mejor resultados da es ARIMA(6,1,7).

5. **Evaluar las predicciones del modelo en el tiempo**
   Para evaluar cuál es el mejor periodo a predecir en el tiempo, comenzamos entrenando con cinco años de datos e intentamos predecir qué ocurrirá en x periodos (columna periodos de tiempo a predecir). A continuación, se aumenta el conjunto de datos de entrenamiento en y valores y se vuelve a hacer una predicción. Repetimos esto hasta terminar con el conjunto de datos.

[y] Avance de nº datos de entrenamiento	[x] Periodo de tiempo a predecir (días)	RMSE (error de predicción)
7	1	0.006481534
7	7	0.01568641
7	15	0.02257918
7	30	0.03148379
7	60	0.04970971

Cuadro 5.2: Tabla correspondiente a predicciones para el valor de apertura del euro

5.3.2 Resultados para los índices elegidos

Para los valores de apertura de los siguientes índices se realizan las etapas de ARIMA, iterando en ellas las veces que sean necesarias y utilizando los índices BIC y AIC para comparar las distintas configuraciones posibles de los parámetros del modelo.

Índice (apertura)	Modelo ARIMA
CCMP_Index_Open	ARIMA(0,1,1)
GSPG10YR_Index_Open	ARIMA(14,1,14)
BDIY_Index_Open	ARIMA(6,2,3)
RX1_Comdty_Open	ARIMA(1,2,1)
NKY_Index_Open	ARIMA(1,2,16)
DU1_Comdty_Open	ARIMA(2,2,5)
NOK_Curncy_Open	ARIMA(1,2,1)

Cuadro 5.3: Modelos de ARIMA utilizados en los distintos índices

A continuación, se compararán para cada índice de la tabla anterior el RMSE dado por ARIMA con el dado por Naïve para analizar cuánto de bueno es el método y si merece la pena utilizarlo. Naïve es mucho más sencillo que ARIMA, únicamente consiste en tomar como valor de predicción el último valor conocido, por o que es útil para comparar con otros métodos pero no para hacer realmente predicciones, con lo que en este apartado podremos determinar en qué en ocasiones ARIMA merecerá la pena y en cuáles no. En ambos métodos el entrenamiento se ha realizado varias veces, inicialmente con la mitad de los datos y posteriormente añadiendo cada vez datos de siete días más. Además, las pruebas se han hecho con periodos distintos, es decir, para 1, 7, 15, 30 y 60 días después (futuro).

- *BDIY_Index_Open*, Baltic dry (transporte por mar de las principales materias primas)

Periodo	ARIMA RMSE	Naïve RMSE
1	0.01611143	0.029967789092013
7	0.10459385	0.194763064539995
15	0.17474749	0.33957300857362
30	0.25431621	0.48408399451707
60	0.27949432	0.404474059881938

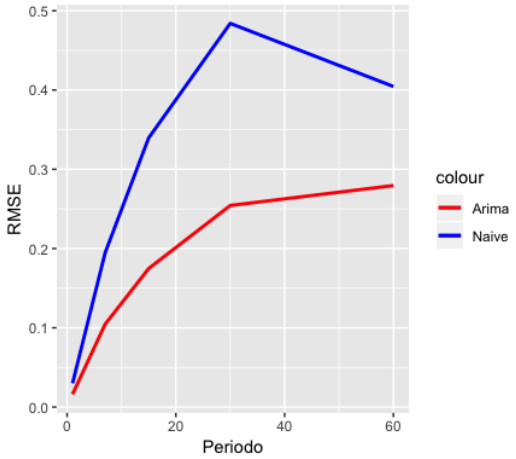


Figura 5.8: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el BDIY

Como podemos observar, el modelo de ARIMA utilizado tiene menor RMSE que el modelo Naïve. Cabe destacar que el RMSE para Naïve crece rápidamente con el aumento del periodo a predecir, cosa que no ocurre con el RMSE del modelo utilizado de ARIMA, donde el error crece, pero no tan bruscamente.

Por lo tanto, para la predicción de este índice bursátil, el método ARIMA es muy útil. Se puede interpretar a partir de esta información y la definición del método ARIMA, que un valor en un día de este índice se explica a partir de valores pasados de él mismo, en concreto, los de los parámetros p y q del modelo.

Cabe destacar que para Naïve, el RMSE es menor en el periodo de sesenta días que en el de treinta.

- *CCMP_Index_Open*, Bolsa NASDAQ

Periodo	ARIMA RMSE	Naïve RMSE
1	0.02914435	0.0248824106252457
7	0.07911989	0.0381883549088299
15	0.10871678	0.0684860721605474
30	0.14089971	0.0952111385144418
60	0.19882457	0.146896778429152

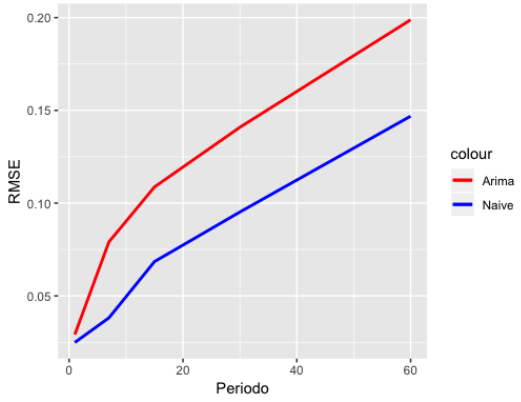


Figura 5.9: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el CCMP

Se puede observar como para este índice ARIMA no da buenos resultados, ya que Naïve, que es mucho más sencillo, tiene menor RMSE en cualquier periodo en el que quiera predecirse un valor. Con lo cual, la serie temporal de este índice no se predice bien a partir de valores de su pasado.

- *GSPG10YR_ Index_ Open*, bono español a 10 años

Periodo	ARIMA RMSE	Naïve RMSE
1	0.03817747	0.0364160894952635
7	0.07454101	0.0631309840166907
15	0.10864322	0.0971769145209907
30	0.15407236	0.170163804295398
60	0.23204214	0.329546188161634

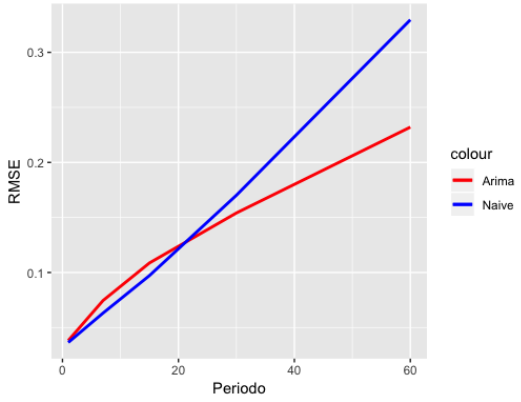


Figura 5.10: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el GSPG10YR

Los errores son muy parecidos en cualquier periodo excepto el último, con lo que para estos periodos tampoco es recomendable utilizar el método de ARIMA para la predicción, ya que con Naïve se obtiene un error solo un poco por debajo en los cuatro primeros periodos y solo un poco por encima en el penúltimo.

Sin embargo, para la predicción a sesenta días ARIMA mejora mucho; con lo que, si nos interesa predecir en este periodo de tiempo, sí es útil el método ARIMA.

- *RX1_ Comdty_ Open*, bono alemán a 2 años

Periodo	ARIMA RMSE	Naïve RMSE
1	0.03834081	0.03050880388413
7	0.11448750	0.0786414307163308
15	0.15619048	0.109262483516575
30	0.22929576	0.164145726835862
60	0.30793143	0.157785550840015

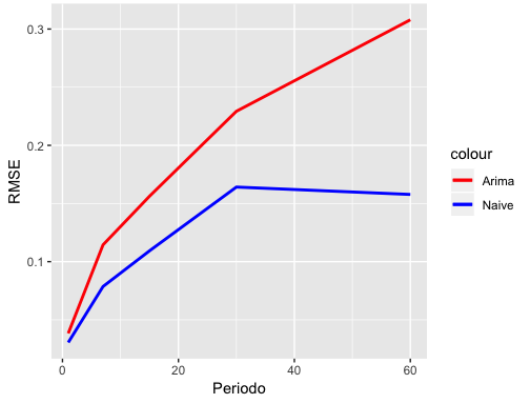


Figura 5.11: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el RX1

Se puede observar que no merece la pena utilizar ARIMA para la predicción de este índice. Sin embargo, cabe destacar que Naïve en este caso predice mejor en un periodo de sesenta días que en un periodo de treinta, cosa que no suele ocurrir frecuentemente.

- *NKY_ Index_ Open*, Bolsa de Tokio

Periodo	ARIMA RMSE	Naïve RMSE
1	0.03849304	0.0469252934335542
7	0.11451086	0.117306048466862
15	0.15573463	0.148867381279556
30	0.22890492	0.195907781747393
60	0.30759872	0.235687447181418

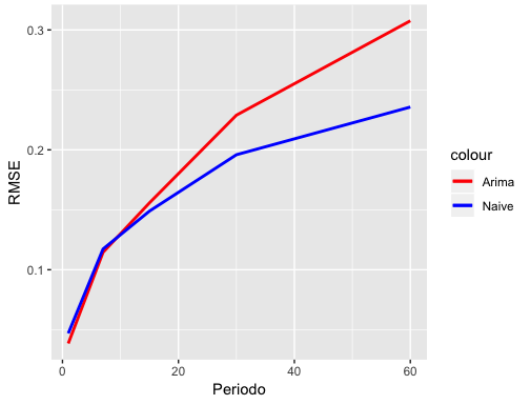


Figura 5.12: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el NKY

Podemos observar que el no merece la pena utilizar ARIMA, puesto que para los dos primeros periodos el RMSE de Naïve es muy similar y para el resto de periodos el RMSE de ARIMA se dispara por encima del de Naïve.

- *DU1_ Comdty_ Open*, bono alemán a 10 años

Periodo	ARIMA RMSE	Naïve RMSE
1	0.02327568	0.0215252454587909
7	0.04722450	0.0491436630340525
15	0.07043681	0.0775912084296811
30	0.09060923	0.10753289924862
60	0.10492578	0.0874196559748425

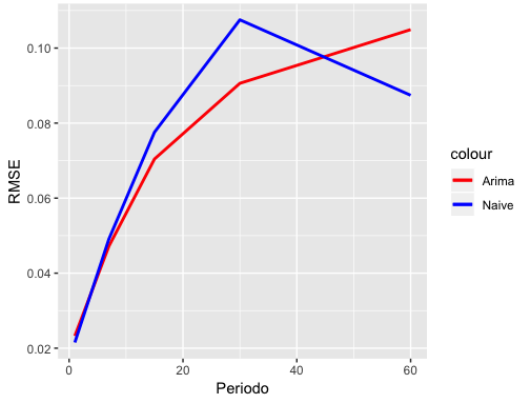


Figura 5.13: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el DU1

En este caso, Naïve tiene un menor error en el periodo de un día, pero es bastante similar al de ARIMA. Para el resto de periodos el modelo de ARIMA predice bastante mejor que Naïve. Con lo cual, si hemos de elegir un modelo para predecir sería ARIMA.

- *NOK_ Curncy_ Open*, corona noruega

Periodo	ARIMA RMSE	Naïve RMSE
1	0.04669559	0.0366327654050201
7	0.11737283	0.0832436954304209
15	0.16846745	0.101954988228676
30	0.24030717	0.109834447651985
60	0.35174907	0.135994692364902

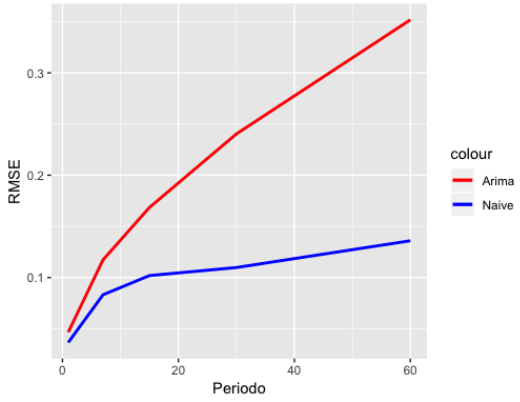


Figura 5.14: RMSE de ARIMA y Naïve en distintos periodos de tiempo para el NOK

Podemos observar que el modelo ARIMA no da buenos resultados en absoluto, ya que su RMSE es mucho mayor que el de Naïve, sobre todo en periodos más grandes como treinta y sesenta días, donde el error se dispara.

Se puede concluir que en la mayoría de las ocasiones ARIMA no da buenos resultados, excepto para aquellos índices que se explican muy bien con pasados de ellos mismos. Un índice no va a depender normalmente únicamente de sí mismo, ya que, en el mundo real, puede verse afectado por los demás. Es por esto que, ya que poseemos información sobre el resto de índices, podemos utilizar otros métodos que nos permitan relacionar nuestro índice a predecir con el resto para mejorar esos RMSEs de ARIMA y Naïve anteriormente calculados.

Capítulo 6

Predicción con Regresión

En este capítulo presentamos los métodos de predicción basados en aprendizaje automático, y en particular en los métodos basados en *regresión*.

6.1 Conceptos básicos

El término regresión, introducido en 1889, se podría entender como "ir hacia atrás". Se trata de un proceso estadístico para evaluar las relaciones de las variables, entre una **variable dependiente** y (a menudo llamada *label*) con una o varias **variables independientes** $\overline{X}$. La idea [Shalev-Shwartz and Ben-David, 2014] es buscar algún tipo de patrón (normalmente una dependencia funcional f), que permita explicar los valores ya conocidos de un cierto *conjunto de entrenamiento* T tal que:

$$T = \{(y_1, \overline{X_1}), \ldots, (y_n, \overline{X_n})\}$$

$$y_i \approx f(\overline{X_i}) \text{ para } i = 1 \ldots n$$

.

La idea entonces es utilizar esta función f para calcular (o *predecir*) un posible valor y' de un valor de la variable independiente $\overline{X'}$ no considerado hasta el momento; es decir, para obtener $y' = f(\overline{X'})$. Se trata por tanto de un procedimiento de *aprendizaje supervisado* [Mitchell et al., 1990].

Aquí el símbolo $\approx$ indica que no se obtiene exactamente el valor y_i sino una aproximación. La bondad de la aproximación dependerá de la medida de error que decidamos considerar (ver subsección 4.2.1). Hay que tener en cuenta que nuestro objetivo no es minimizar el error sobre los datos de entrenamiento T, sino sobre los valores nuevos predichos. Un exceso de *sobreajuste* de la función f a los datos de entrenamiento puede resultar en peores predicciones [Babyak, 2004].

A menudo, y este es nuestro caso, la regresión se utiliza para predecir el **comportamiento de una variable en el futuro**. De modo que cuando decimos que existe regresión, implica que una variable se explica respecto a otra con la llamada línea de regresión.

6.1.1 Tipos de regresión

Comenzamos diferenciando los tipos de regresión existentes:

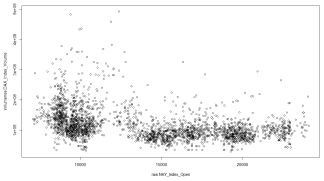
- Dependiendo de la cantidad de variables independientes, la regresión puede ser **simple** o **múltiple**.

  La regresión simple es aquella que consta de dos variables, la dependiente y la independiente, mientras que la múltiple cuenta con más de una variable independiente.

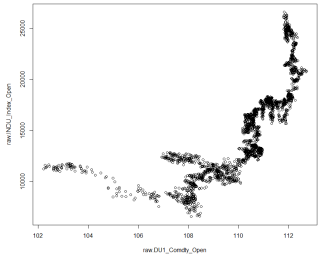
- Dependiendo del tipo de función puede ser **lineal** o **no lineal**.

  Llamamos regresión lineal a aquella cuya función que explica la relación entre las variables resulta ser una línea recta, en el caso contrario hablaremos de regresión no lineal que a su vez puede ser parabólica, exponencial, potencial...

Figura 6.1: Tipos de regresión según f(x)



(a) Reg.Lineal



(b) Reg.Exponecial

Fuente: Elaboración propia

En la Figura **6.1a** se puede deducir que se trata de una relación lineal, el eje X corresponde a *NKY_ Index_ Open* y el eje Y a *DAX_ Index_ Volume*. Por el contrario, el la Figura **6.1b** donde están representados en el eje X *DU1_ Comdty_ Open* y en el eje Y *INDU_ Index_ Open*, nos encontramos con una relación no lineal, más específicamente representa una función exponencial.

Nuestro objetivo es identificar el comportamiento de una variable dependiente usando un conjunto de ellas, por lo que usaremos regresión múltiple. A su vez, analizaremos la relación de cada variable de este conjunto con la variable dependiente.

6.2 Métodos de regresión

Veamos los distintos tipos de regresión que vamos a emplear en este trabajo.

6.2.1 Regresión Lineal

Para la aplicación de este método se supone que la relación entre la variable dependiente y las independientes es lineal, siguiendo la siguiente función:

$$Y = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p + \varepsilon$$

$\beta_0 =$ término independiente
$\beta_1, \beta_2 ... \beta_p =$ coeficientes parciales de la regresión
$X_1, X_2, ..., X_p =$ valores de las variables independientes
$\varepsilon =$ error de observación debido a variables no controladas

Este modelo es sencillo, pero suele tener buenos resultados en pronósticos a largo plazo. Por desgracia, las variables no suelen tener relaciones tan sencillas.

Podemos destacar su uso en series temporales, donde se dan sus mejores resultados, tanto cuando una variable cambia en función del tiempo como cuando hay una relación causal (cambio por causa de una variable). También es muy usado en el análisis de la demanda.

6.2.2 Boosted Regression Trees

El **BRT** (por sus siglas en inglés) es un método supervisado que combina dos algoritmos: los árboles de regresión y el *boosting* (método para combinar muchos modelos simples y dar una mejor predicción), consta de varias iteraciones en las cuales se van ajustando los árboles.

La manera de construir los árboles es la siguiente: se selecciona un subconjunto aleatorio de entre todos los datos para construir un árbol, después de haber creado el árbol se vuelve a introducir en el conjunto de datos completo el subconjunto tomado, abriendo así la posibilidad de que se vuelvan a usar para construir otro árbol posteriormente. La mejora que se añade al usar **boosting** consiste en **aplicar pesos a los datos**, de modo que aquellos que no han sido bien predichos, en iteraciones siguientes tendrán mayor probabilidad de ser seleccionados para construir otro árbol y ser ajustados en este. Este ajuste iterativo es único en *boosting* [Elith et al., 2008].

Este modelo no necesita trasformación previa de datos ni la eliminación de valores atípicos, y se ajusta a relaciones no lineales y complejas.

6.2.3 Random Forest Regressor

Es uno de los modelos más efectivos para análisis predictivo. Combina la idea de *bagging* y los árboles de regresión. La idea principal es construir una serie de árboles de decisión y luego promediar sus resultados [Breiman, 2001].

La manera de construir los árboles es la misma que en el método *Boosted regression trees* antes mencionado, con la diferencia de que los datos no tienen pesos y el componente aleatorio, que se introduce al crear los árboles de la siguiente manera, en lugar de dividir un nodo por la característica más importante, la selecciona de entre un subconjunto aleatorio, lo que genera gran diversidad en los árboles.

$$g(x) = f_0(x) + f_1(x) + f_2(x) + ...$$

$g(x) =$ Es el modelo final
$f_i(x) =$ Los modelos simples

Algo a tener en cuenta es que evitan el sobreajuste que se podría dar con un árbol profundo y complejo, usando varios árboles e incluyendo un factor aleatorio. Otra de sus ventajas

es que puede manejar grandes conjuntos de datos. En cambio, su mayor inconveniente es que los mejores resultados se obtienen para una gran cantidad de árboles, lo que supone a su vez un gran incremento en el consumo de recursos computacionales (tiempo y memoria).

6.2.4 Voting Regressor

Aunque no se trata propiamente de un método diferente, sino de una combinación de métodos y promedios, mencionamos la técnica *voting regressor* [An and Meng, 2010] aquí porque ha sido una de las que ha dado mejores resultados en nuestros experimentos. En particular, nuestro método emplea la combinación de tres regresores mencionados anteriormente: un *gradient boosting regressor*, un regresor lineal, y un *random forest regressor* con 10 árboles.

La combinación de estos tres regresores promediándolos evita el sobreajuste que se podría dar si se ajustase uno de los regresores demasiado. Además, evita explorar todas las posibilidades, lo que es de interés en un contexto con gran número de variables (y tiempo considerable requerido por cada experimento), aunque a cambio tiene el problema, como veremos, de que puede caer en *mínimos locales*.

6.3 Nuestros datos

A continuación, se van a explicar los pasos que se han seguido con una variable dependiente hasta que se obtienen las variables independientes que hacen las predicciones de esta variable dependiente lo más ajustadas posibles.

El esquema que se ha usado para la selección de variables ha sido el llamado *método forward*, que consiste en cada iteración añadir el término que sea más significativo para el modelo. Este método evita explorar todas las posibilidades, lo que es de interés en un contexto con gran número de variables(y tiempo requerido por cada experimento), aunque a cambio tiene el problema, como veremos de que puede caer en *mínimos locales*.

6.3.1 Conjunto de datos

Los datos de entrada consisten en una tabla en la cual se encuentra el valor que queremos predecir y los demás valores, entre los que vamos a realizar la selección de los más a apropiados para hacer la predicción. Se van a realizar los experimentos con dos tablas principalmente, la primera *rawVol_ open* que contiene la información del volumen de los índices bursátiles y sus correspondientes valores de apertura y la segunda *raw_ open* que contienen solo los valores de apertura. Esta segunda tabla tiene la información de más índices bursátiles que la primera ya que no tenemos el volumen de todos los índices.

6.3.2 Desarrollo del experimento

A continuación, vamos a aplicar el método de aprendizaje automático, para lo cual debemos seguir los siguientes pasos: en primer lugar debemos crear la columna *label* con los incrementos como se explica en la sección 4.3, en segundo lugar, por cada posible valor "**candidato**" a ser una variable independiente se prepara la tabla correspondiente de la forma que se ve en la Figura 4.3 de la sección 4.3. Una vez que se tiene esta tabla, tenemos que evaluar cómo de bueno es ese valor "candidato" para predecir nuestra variable dependiente, para ello

calcularemos el *RMSE* entre las predicciones que hace nuestro método y los incrementos reales. Para hacer las predicciones primero se tiene que **entrenar el modelo** que hayamos optado por usar, en nuestro caso el entrenamiento se ha hecho como mínimo con datos de 1500 días, después de entrenar se pone en práctica el modelo haciendo la predicción de un valor.

Los valores de test que se introducen en el modelo ya entrenado no deben estar en el conjunto de datos que se han usado para entrenar, ya que en este caso la predicción no estaría siendo real, porque esos datos ya habrían sido utilizados para que el modelo aprendiera sus relaciones.

Volviendo a la evaluación del modelo, se trata de repetir la fase de entrenamiento y predicción con diferentes conjuntos β veces (en nuestro experimento $\beta = 100$). Finalmente obtendremos una serie de predicciones que, junto con los incrementos reales, nos permitirán calcular el *RMSE*, que será nuestra medida de referencia.

Hasta el momento, se ha explicado el proceso desde que se elige un valor independiente hasta se obtiene el *RMSE* para evaluar su idoneidad. A continuación, se describe el proceso *forward* que hemos mencionado con anterioridad, mediante el que se obtendrá el **mejor conjunto de valores independientes** para predecir nuestra variable dependiente.

Se trata de un proceso iterativo. En cada iteración se intenta añadir al mejor valor o valores de las iteraciones anteriores un valor nuevo que aporte una mejora en el *RMSE* de la predicción. Las iteraciones terminan en el caso de que se llegue a un número fijado o en el que la nueva columna no mejorase en absoluto la predicción.

A continuación, se puede ver un esbozo de la implementación para dicho proceso iterativo de selección de valores independientes. En primer lugar, se deben se establecer α = máximo número de variables independientes que queramos para el experimento (tener en cuenta el tiempo) y β = número de predicciones que se van a hacer usando la variable que corresponda, a partir de las que se calculará el RMSE.

En la linea 10 del código se puede ver cómo se crea la tabla de datos con la que se entrena el modelo y se hacen las predicciones. Esta tabla consta de: la columna *label*, que contiene la información sobre los pasados del conjunto de variables independientes elegidas en otras iteraciones (*columnas*) y la información sobre los pasados del nuevo valor "candidato" (*valor*). Al terminar el *for* de la línea 18 ya se han completado los β entrenamientos y sus correspondientes predicciones, por lo que en la línea 19 se calcula el *RMSE* entre las predicciones hechas y los valores reales. Si el error resultante mejora el mínimo hasta el momento (el mínimo *RMSE*) se establece un nuevo mínimo y se guarda ese valor (línea 23). En la línea 26, una vez terminada la iteración, si ha habido alguna mejora el error ($cambios > 0$), se añade el valor correspondiente con esta mejora al conjunto *columnas*.

Algorithm 1 Bucle para elección de mejores valores predictores

Input: d: tabla con los valores, *label*: lista de valores a predecir
Output: *columnas*: lista de las mejores columnas predictoras para el valor dependiente

```
Iteraciones = α
columnas = []
terminado = False
p = 20
f = 30
rsme = 1
for n = 0; n < Iteraciones ∨ !terminado; n++ do
    cambios = 0
    for valor ∈ d ∧ valor ∉ columnas do
        datos = label + (pasados(valor, p) + columnas)
        predice, reales = []
        for i ∈ range(0, β) do
            modelo.train(1500 + i))
            dia = 1500 + i + f
            pred = modelo.predice(dia)
            predice.append(pred)
            reales.append(label[dia])
        end for
        new_rmse = rmse(reales, predichos)
        if new_rmse < rmse then
            cambios+ = 1
            rmse = new_rmse
            new_valores = pasados(valor, p)
        end if
    end for
    if cambios > 0 then
        columnas = columnas + new_valores
    else
        terminado = True
    end if
end for
```

6.4 Discusión de los resultados

En este apartado se discuten los resultados obtenidos en la predicción de algunos índices y los distintos métodos que se han usado para ello, mencionados en la sección 6.2. El conjunto de índices que se han elegido son los mismos para los que se hizo una primera predicción con ARIMA con el fin de poder comparar los resultados más adelante.

Para intentar conseguir llegar a tener los mejores resultados en el menor tiempo posible se han hecho pruebas con varios archivos que contienen más o menos información, en primer lugar se supuso que la mejor opción era usar un archivo con toda la información que teníamos a nuestro alcance, pero, como era de esperar, el tiempo en obtener resultados era muy

elevado, por lo que finalmente se han utilizado un archivo que contiene únicamente el valor de apertura de los índices y otro que contiene el valor de apertura y el volumen de acciones intercambiadas en una sesión bursátil. Esta decisión se tomó porque, después de hacer varias pruebas, en muchas ocasiones no había mejora al usar más datos; además, teniendo más datos había más riesgo de caer en mínimos locales.

A continuación, nos centraremos en explicar los resultados obtenidos para los valores *NKY_Index_Open* y *CCMP_Index_Open*. Las predicciones se han hecho para el valor en 30 días; nos pareció una medida adecuada ya que en la mayoría de experimentos ya hechos se trabaja con predicciones en menos tiempo. Para cada uno se ha realizado un total de 8 experimentos, con los dos conjuntos de datos anteriormente mencionados y con los cuatro métodos distintos (Regresión Lineal, Gradient Boosting Regressor, Random Forest Regressor y Voting Regressor).

Después de ver los resultados obtenidos se puede deducir que el volumen de acciones intercambiadas aporta información valiosa, pues aunque sean menos datos (ya que no tenemos la información del volumen de todos los índices los resultados) mejoran al tener acceso a esta información. El método que mejores predicciones ha hecho, en general, es el **Voting Regressor**; esto era de esperar, pues este método realmente es una combinación de métodos. Comparando los otros tres métodos restantes, en ninguno de los casos la regresión lineal ha resultado ser la mejor opción. Gradient Boosting Regressor y al Random Forest Regressor son las mejores opciones, ya que la diferencia en los resultados es mínima, y debemos de destacar que estas pequeñas diferencias pueden ser por causa del componente aleatorio de los métodos, ya que tras repetir el experimento llevado a cabo con estos dos métodos los resultados y las variables dependientes han variado, lo que también abre la posibilidad de que las variables independientes elegidas en cada caso, aunque no sean las mismas, sí que pueden pertenecer a un mismo subgrupo; por ejemplo, si en una primera ejecución las variables dependientes son(x_1, y_1, z_1) y en una segunda (x_2, y_2, z_2) podría resultar que x_1 y x_2 perteneciesen a un subconjunto e y_1 e y_2 a otro subconjunto con características en común.

En esta tabla se pueden ver reflejados los errores obtenidos con cada uno de los métodos usando el conjunto de datos con el que mejor resultado se ha obtenido.

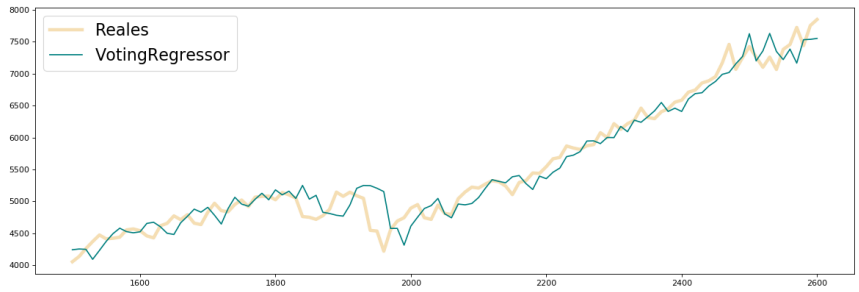
	NRMSE	
	NKY_Index_Open	**CCMP_Index_Open**
Linear Regression	0,08491333	0,064507226
Gradient Boosting Regressor	0,071885368	0,06145602
Random Forest Regressor	0,08521506	0,05531515
Voting Regressor	0,080648	0,053564

La siguientes gráficas muestran 111 valores reales y los correspondientes predichos con el modelo que ha resultado comportarse mejor para cada uno de los valores y usando los siguientes valores independientes: para **NKY_Index_Open** → **TU1_Comdty_Volume** y para **CCMP_Index_Open** → **MEXBOL_Index_Volume y TU1_Comdty_Open**, estos valores predichos se han calculado entrenando un modelo hasta cierto día y prediciendo que ocurriría en 30 días, esto se ha repetido en las predicciones de 111 días separados por un intervalo de 10 días y comenzando con un conjunto de entrenamiento de 1500 días.

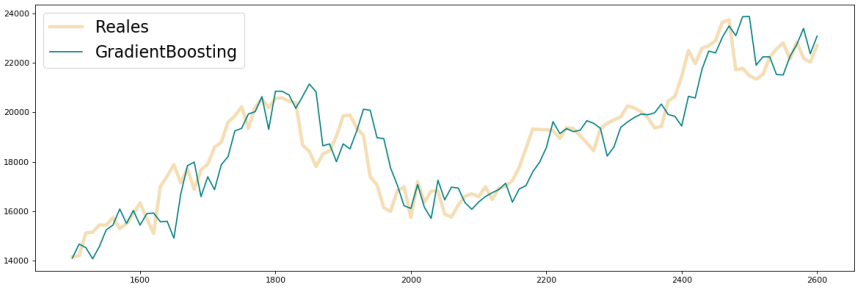
En las siguientes gráficas se aprecia que el valor predicho es el valor real desplazado, esto puede recordar a Naïve, y en efecto, todos los métodos tienden a repetir patrones conocidos,

pero existen correcciones con respecto a Naïve que hacen que el error sea menor.

Figura 6.2: Graficas de los valores reales vs los valores predichos



(a) Valores reales y predicciones del Voting Regressor para el *CCMP_Index_Open*



(b) Valores reales y predicciones del Gradient Boosting Regressor para el *NKY_Index_Open*

Capítulo 7

Predicción con redes neuronales

7.1 Qué son las redes neuronales

Llamadas RNA o ANN (acrónimo de Artificial Neural Network). Se empezó a desarrollar esta técnica de aprendizaje automático en la de cada de los 40. Inspirándose en el comportamiento del cerebro humano, se intentan crear modelos para resolver problemas complejos, tienen capacidad de aprender, reconocer patrones tal y como hacen las neuronas de nuestro cerebro.

Los desarrollos actuales de los científicos se dirigen al estudio de las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas. Así, la inteligencia artificial es un intento por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas. [Matich, 2001]

En este estudio vamos a centrarnos en comparar los resultados aplicando el perceptrón multicapa y aplicando Backpropagation, los cuales explicaremos más adelante.

7.2 Cómo funcionan las redes neuronales

En primer lugar, se va a explicar el funcionamiento de la unidad básica de procesamiento: una neurona perceptrón. Cada una de las neuronas tiene un conjunto de entradas, n componentes $(w_1, \ldots, w_n)$, más una entrada adicional llamada sesgo (w_0), y una salida. El cálculo de esa salida consiste en un suma ponderada de las entradas $(\sum)$, donde se decide con qué intensidad cada entrada afecta a la neurona y finalmente, antes de la obtención final de la salida, se pasa por una función de activación, que consiste en lo siguiente: devolver una salida en función a la entrada, los valores de salida suelen estar comprendidos en un rango, se van a diferenciar tres de los tipos de funciones de activación [Calvo, 2008]:

1. **Rectified Lineal Unit**
   Anula los valores negativos, no esta acotada, discreta por la función:
   $$f(x) = max(x, 0) = \begin{cases} 0 \ for \ x < 0 \\ x \ for \ x \geq 0 \end{cases}$$

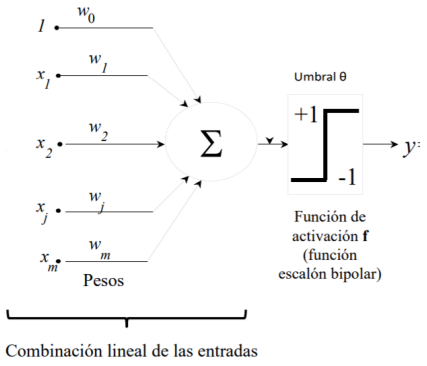
2. **Sigmoide**
   Transforma los valores de entrada a un rango $[0, 1]$, tiene una lenta convergencia, esta

descrita por la función:

$$f(x) = \frac{1}{1 - e^{-x}}$$

3. **Tangente hiperbólica**
   Transforma los valores a una escala $[1, -1]$, similar a la sigmoide, también tiene una lenta convergencia, descrita por la función:

$$f(x) = \frac{2}{1 - e^{-2x}} - 1 = tanh(x)$$



Fuente: Apuntes UCM I.A.

Figura 7.1: Funcionamiento interno de una neurona perceptrón

La red se inicializa con pesos aleatorios, que van cambiando a medida que la neurona va aprendiendo. Usar una sola neurona perceptrón limita la introducción de problemas más complejos, es por esto que surge el **perceptrón multicapa** (*multi-layer perceptron* o *MLP*), una red de flujo hacia delante (*feedforward*). El *MLP* consta de 3 capas, la **capa de entrada** consta de tantas neuronas como variables de entrada, las **capas ocultas**, con varias neuronas cada una, y la **capa de salida**, que en este caso como tratamos un problema de regresión es solo una neurona que no usa la función de activación.

Aplicar **backpropagation** supone una mejora, se basa en operar de forma recursiva capa tras capa moviendo el error hacia atrás (error que resulta de comparar el valor predicho y el real) hasta llegar a las neuronas iniciales, es decir, propagando una sola vez el error obtendremos el error correspondiente para cada neurona y los valores correspondientes a los pesos y el sesgo.

7.3 Por qué usar redes neuronales

A continuación, se presentan las ventajas que ofrece usar redes neuronales.

- El perceptrón multicapa en regresión es apreciado porque es capaz de reflejar relaciones no lineales.

- Cada vez se acercan más a esa idea original de reproducir el funcionamiento del cerebro humano.
- Las redes neuronales permiten buscar la combinación de parámetros que mejor se ajusta a un determinado problema.
- Una RNA puede manejar cambios no importantes en la información de entrada, como señales con ruido u otros cambios en la entrada.
- La auto-organización.
- Entre sus usos destacan las predicciones financieras
- Una red con la estructura adecuada puede aprender a realizar cualquier operación, sin necesidad de que le facilitemos la fórmula entre las variables de entrada y la salida.
- Tolerancia a fallos, si la red sufre un fallo, se ve afectada, pero no sufre una caída.
- Inserción fácil con la tecnología existente.

7.4 Discusión de resultados

Para aplicar redes neuronales se ha puesto en práctica el mismo procedimiento que se ha usado con anterioridad para usar los métodos de regresión en el apartado 6.3. Hemos decidido usar el perceptrón multicapa fijando una capa oculta y 100 neuronas en esta capa. La función de activación que se ha usado en este primer experimento es la función *Rectified Lineal Unit*.

Al igual que en el análisis de resultados de regresión lineal, se van a comentar los resultados obtenidos con los índices **NKY_Index_Open** y **CCMP_Index_Open**. Estos resultados se pueden ver en las tablas siguientes:

En esta primera tabla se pueden ver los **NRMSE** de cada uno de los experimentos.

	NKY_Index_Open	**CCMP_Index_Open**
Datos con volúmenes	0,069722115189414	0,0539758854585616
Datos sin volúmenes	0,0771046217405224	0,0528421260351978

Cuadro 7.1

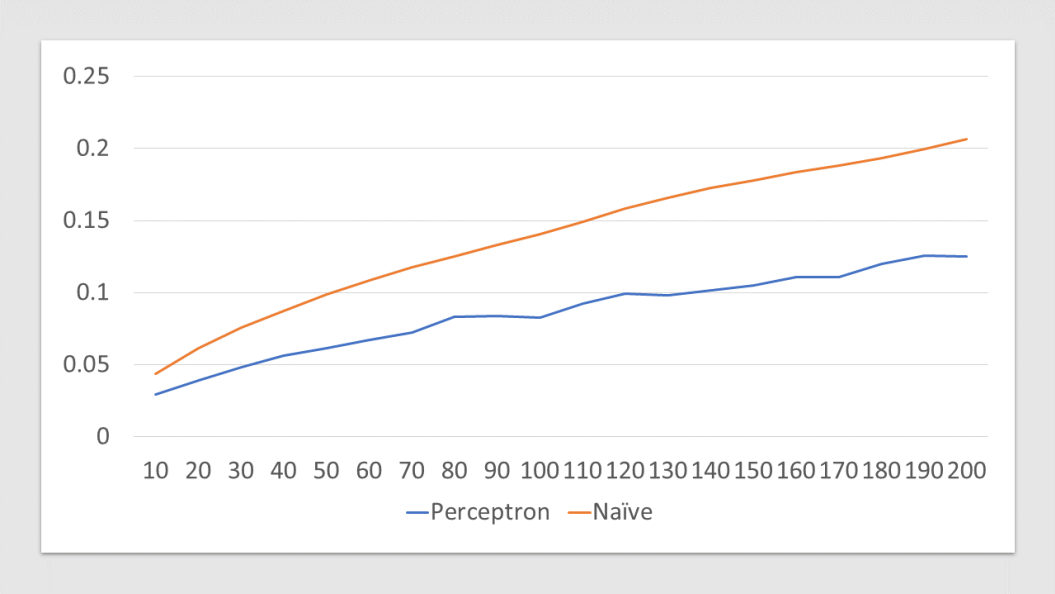
En esta segunda tabla vemos las variables independientes elegidas en cada caso.

	NKY_Index_Open	**CCMP_Index_Open**
Datos con volúmenes	LMAHDS03_Comdty_Open DAX_Index_Open	HG1_Comdty_Open DAX_Index_Open
Datos sin volúmenes	CAC_Index_Open	C_1_Open

Cuadro 7.2

En la primera columna del Cuadro 7.1, *NKY_Index_Open*, se aprecia una diferencia notable entre los resultados obtenidos con la información de los volúmenes y sin ellos. En cambio, con el valor *CCMP_Index_Open*, la diferencia es un poco mayor (apenas una milésima), lo que puede deberse al componente aleatorio del método. En cambio, si nos fijamos en el Cuadro 7.2, vemos que aunque se ha utilizado información de los volúmenes para la predicción, las variables predictoras corresponden todas a índices de los valores de apertura. Anteriormente, en el apartado 6.4, se ha mencionado que el conjunto de datos que contiene la información de volúmenes es menor, ya que no se tiene la información sobre el volumen de todos los índices bursátiles; esta puede que sea la clave, es decir, teniendo en cuenta la existencia de mínimos locales (aunque el uso de backpropagation los reduce) con un conjunto de datos mayor (*Datos sin volúmenes*) la probabilidad de caer en un mínimo local sería mayor que con un conjunto de datos más pequeño (*Datos con volúmenes*). Aún con estas diferencias de resultados, todos se pueden considerar un modelo aceptable para hacer predicciones.

A continuación, se puede observar una gráfica en la que se muestra la evolución del error del perceptrón a lo largo del tiempo. Se ha fijado el pasado en 30 días y se ha ido cambiando el valor de futuro que usa el algoritmo. También se puede ver cómo evoluciona la predicción de Naïve. Cabe destacar que al establecer el valor de pasado en 30 no significa que solo use la información de los 30 días anteriores al día que se establece como último valor conocido, sino que por cada fila de la tabla (en nuestro caso por cada incremento) usa sus 30 días anteriores, y esto con cada día, para entrenar el modelo. Por lo que, finalmente, el modelo habrá aprendido con unos valores del pasado muy superiores a 30 días.



Fuente: Elaboración propia

Figura 7.2: Evolución del error en la predicción a lo largo del tiempo

En la gráfica observamos que el error en las predicciones del perceptrón se van estabili-

zando poco a poco, esto sugiere que existe una tendencia global que apenas cambia con el tiempo.

Llevando a cabo una fase de *Tune Hyperparameters* para intentar mejorar el resultado que hemos obtenido para el valor *CCMP_ Index_ Open*, se ha aumentando el número de capas ocultas a dos: la primera con 100 neuronas (como en el primer experimento) y la segunda con 50 neuronas. Además, se ha cambiado la función de activación a una de tipo *Sigmoide*; se han realizado varias ejecuciones para asegurar que el resultado no haya sido un caso aislado debido a el componente aleatorio del método y, en efecto, en todas las ejecuciones hechas se han mejorado los resultados, siendo ahora el menor **NRMSE** $\rightarrow$ **0,05049577255548418** con las variables independientes ***CRY_ Index_ Open*** y ***MO1_ Comdty_ Open***.

Capítulo 8

Contribuciones personales

8.1 Raquel

- **Preprocesamiento de los datos**

  Contribución en la creación de scripts en Python para el pretratamiento de los datos brutos que obtuvimos al comienzo de este trabajo. Consistieron en la transposición de la tabla de datos (puesto que en un principio los valores de los indicadores no se trataban de las columnas, si no de las filas) y en el tratamiento de *missing values* en algunos valores bursátiles, que decidimos eliminar por completo de nuestro conjunto de datos, ya que eran demasiados.

  Además, mi compañera y yo realizamos otros algoritmos en Python que no fueron finalmente parte de nuestro trabajo. Consistían en scripts que escalaban los datos, realizaban diferencias, incorporaban una columna de volatilidad...

- **Resumen**

  Inclusión del resumen del trabajo que se realiza a continuación y de las palabras clave.

- **Introducción**

  Breve explicación de los objetivos de nuestro trabajo y de la estructura de la memoria que vamos a seguir.

- **Estado del arte**

  Búsqueda de artículos relacionados con regresión múltiple, dada la importancia que tiene en nuestro trabajo los resultados obtenidos con otros experimentos usando estos métodos. Además, búsqueda del artículo relacionado con la predicción con perceptrón multicapa y su posterior descripción.

  Los artículos han sido encontrados a través de Google Scholar, escogiendo aquellos que más importancia tenían para nuestro trabajo. En cada uno de ellos he descrito el problema inicial del que parte ese artículo, siempre relacionado con la predicción valores bursátiles (Bolsa de valores Estambul, de La India y de Bombay). A continuación, he descrito los datos de los que se disponían en cada uno de los artículos para posteriormente exponer los métodos de predicción y las técnicas de análisis de resultados que

han sido utilizados. Estas dos últimas tarea siendo algo tediosas ya que tanto algunos métodos utilizados (*Buy and Hold*) como algunas técnicas de análisis de resultados (*test de bondad de ajuste de Hosmer-Lemeshow, chi-cuadrado*) no los conocía hasta el momento.

Además, he visto necesario incluir algunas figuras del artículo para una mayor comprensión de los resultados. Por último, he expuesto una breve conclusión de lo que repercute cada artículo en nuestro trabajo.

- **Conjunto de datos**

  Introducción de un párrafo en el que se explica lo que vendrá a continuación en el capítulo.

  Búsqueda y exposición de los distintos indicadores bursátiles de los que disponemos en nuestro conjunto de datos (los llamados *OHLCV*) y explicación de cada uno de ellos a partir de la información recopilada de artículos y periódicos de Internet, haciendo bastante hincapié en el indicador *volume* y describiendo su relación en cuanto a los movimientos de los valores y su importancia en nuestro trabajo.

  Incorporación de la tabla descriptiva, en la que se ha hecho una búsqueda acerca de qué significa cada uno de valores bursátiles que componen nuestro conjunto de datos. Además, se han separado en distintas tablas los diferentes tipos de valores bursátiles (índices, monedas y mercancías).

  Contribución en la búsqueda de información sobre las medidas de error típicas (RMSE, MAE, R-cuadrado y R-cuadrado ajustado) y su posterior inclusión en la memoria del trabajo. He contribuido tanto en la definición como en la creación de los cuadros de fórmulas.

- **Predicción con ARIMA**

  Contribución en la búsqueda y explicación detallada del método ARIMA. En concreto, redacción del primer párrafo que introduce lo que se realizará a continuación, contribución tanto en la búsqueda y definición breve de ARIMA en los primeros párrafos de la sección 5.1 como de la inclusión de las fórmulas que la definen.

  A continuación, he realizado la explicación de la elección de las columnas (índices bursátiles) que utilizamos en la predicción dada la tabla realizada por mi compañera.

  A partir de la descripción de las etapas de ARIMA, he realizado el ejemplo de metodología para el valor de apertura del euro con el desarrollo del código necesario en R y la incorporación de las gráficas que se muestran. Este proceso se ha hecho en R a través de la consola de comandos de rStudio. Los resultados de cada etapa han sido incluidos en la memoria con sus correspondientes gráficas. Algo que no se incluye en la memoria y que realicé fue otras tres iteraciones en las etapas. Como se indica en el último párrafo de la etapa 3, esto se hizo para encontrar otros posibles modelos que mejorasen al que se tenía hasta el momento. Decidí no incluir esta información puesto que el procedimiento es igual al realizado en las etapas anteriores, cambiando únicamente el resultado obtenido, que muestro a continuación en la siguiente etapa.

  Estas etapas (con sus respectivas iteraciones) fueron realizadas para cada uno de los siete valores bursátiles escogidos, seleccionando los que menor AIC y BIC resultaron

tener. Para ello, realicé en R una serie de scripts para prueba de las distintas configuraciones en cada uno de los valores bursátiles en los que se comparaban AICs y BICs de todos ellos para obtener el mejor.

A continuación, realicé un nuevo script en R para el cálculo de Naïve para cada uno de los valores bursátiles. Se trata de un algoritmo simple en el que el entrenamiento va aumentando de siete en siete y se hace una predicción a 1, 7, 15, 30, 60; para cada periodo se almacena el valor real junto al predicho para poder así calcular el *RMSE* de Naïve en cada periodo.

Finalmente, incluí una tabla comparativa entre el *RMSE* dado por Naïve y el dado por ARIMA para cada uno de los valores bursátiles junto a sus respectivas gráficas. Para cada una de estas tablas y gráficas he comparado ambos errores y he podido concluir para qué casos ARIMA es un buen método y para cuáles no merece la pena utilizarlo.

- **Predicción con regresión**

  Realización de distintas ejecuciones en mis ordenadores locales del algoritmo en Python desarrollado por mi compañera (en su mayoría de los métodos de regresión lineal) para su posterior incorporación en los resultados de la memoria. En mi caso, al contar únicamente con mis ordenadores personales el proceso ha sido bastante lento.

- **Conclusión**

  Contribución en la exposición de las conclusiones a las que hemos llegado con la realización de este trabajo. En concreto, conclusión de la importancia de usar volúmenes en la predicción de valores bursátiles y conclusión de la envergadura de añadir varias variables independientes (distintas del propio valor bursátil que se quiere predecir) en los métodos de predicción utilizados.

8.2 Celia

- **Preprocesado de los datos**
  Elaboración de diversos algoritmos en Python para transformarlos datos, como puede ser escalarlos, rotarlos (cambiar filas por columnas), hacer ciertas rotaciones a algunas columnas con el fin de tener en una fila de la tabla la información necesaria para hacer una predicción, dividirlos por tipo de columna (seleccionar solo la información de los valores de apertura), unirlos (unir los volúmenes con los demás datos ya que se encontraban en una tabla aparte) y limpiarlos, quitar valores que podían ser causa de error (NaN, 0s que dificultaban las operaciones y celdas vacías). No todos estos archivos finalmente han sido usados.

- **Introducción**
  Explicación de las tecnologías que se han utilizado y por qué se han utilizado, investigación de cómo se podía usar algún servicio en la nube para ejecutar nuestros experimentos, para acelerarlos, sin que fuese necesario ningún coste, una vez elegida la máquina virtual de ciencia de datos de Azure, desplegarla y aprender a conectarse a ella. Para que se haya llevado a cabo sin coste use un código de promoción que me facilitó Microsoft en un curso anterior, esto ha permitido, aparte de poder usar los servicios gratis, tener acceso a una máquina virtual de mayor nivel que si se hubiese usado la prueba gratuita por ser estudiante.

- **Estado del arte**
  Búsqueda de estudios en los que se ha aplicado ARIMA para llevar a cabo la predicción de índices bursátiles. Una vez encontrados, se han descrito brevemente los artículos y se han expuesto las conclusiones a las que se ha llegado en cada uno de ellos y se han incorporado tablas recopilatorias de los resultados obtenidos en cada artículo.

- **Conjunto de datos**
  Contribución en la elaboración de las medidas de error típicas buscando información de la manera de obtenerlas y sus características. Elaboración de las tablas del apartado 4.3 y explicación paso a paso de cómo se lleva a cabo la creación de la tabla que contiene los datos a los que se les aplican los distintos métodos de regresión y redes neuronales.

- **Predicción con ARIMA**
  Búsqueda y redacción de parte de la información contenida en el apartado 5.1 que consiste en la explicación de cómo el método ARIMA lleva a cabo el entrenamiento de un modelo y sus predicciones. Desarrollo de los conceptos que se deben de conocer para hacer un experimento con ARIMA de las etapas que se tienen que llevar a cabo en un experimento si se quiere aplicar ARIMA a una serie temporal y cómo se lleva a cabo su evaluación.

  Algoritmo en R para la generación inicial de predicciones con `auto.arima()` para hacer la tabla contenida en el apartado 5.2 con la que elegimos que valores nos vamos a centrar en predecir.

- **Predicción con regresión**
  Búsqueda y desarrollo de la información tanto de regresión, como de los métodos

de regresión que existen y que podemos usar en Python. Para ello, he consultado información sobre regresión lineal, regresión logística, regresión con *Support Vector Machine*, *Boosted decision trees* y *Random forest*. Hice un intento de usar un modelo de clasificación dividiendo el conjunto de datos de entrenamiento en dos grupos, uno el 75 % de los datos y otro el 25 % que corresponde con los datos con el mayor incremento. Las pruebas con algoritmos de clasificación, *Suport Vector Machine* y clasificación logística, no dieron buenos resultados por lo que no se introdujeron en la memoria.

Decisión de qué métodos finalmente vamos a usar (Regresion lineal, Boosted decision tree y Random forest regressor), desarrollo de sus características en la memoria y del código necesario para probar estos métodos. Explicación de cómo hemos aplicado estos métodos a nuestros datos, discusión de los resultados obtenidos, y elaboración de las tablas y las gráficas correspondientes.

Para intentar reducir el tiempo de ejecución reduciendo las entradas de cada experimento se llevó a cabo en primer lugar la aplicación de un algoritmo de clustering, *k-means*, que agrupó los índices bursátiles en 4 grupos, en segundo lugar se aplicó *PCA* (algoritmo de reducción de la dimensionalidad) a cada grupo para reducir las dimensiones de los datos, sin perder información valiosa, y después se unieron los valores resultantes de hacer *PCA* en cada grupo. La reducción en tamaño y en tiempo fue muy significativa y las predicciones no eran nada malas, pero nos enfrentamos al problema de que una vez hecha la predicción no se podía establecer uno o varios índices bursátiles como variables independientes, y abandonamos este camino.

Algoritmo en R para la generación inicial de predicciones con `auto.arima()` para hacer la tabla 5.1 con la que elegimos que valores nos vamos a centrar en predecir.

Cálculo del NRMSE de todos los experimentos mediante un algoritmo en Python con la intención de poder comparar los resultados obtenidos para la predicción de distintos valores.

Elaboración del algoritmo en Python correspondiente con el pseudocódigo 1 para aplicar un método especifico y también el algoritmo en Python para que una vez se tienen los valores independientes para predecir una variable dependiente se lleven a cabo predicciones en el tiempo y se cree la gráfica oportuna (Figura 6.2), donde se pueden ver los valores reales y las predicciones del modelo correspondiente.

Decisión de qué experimentos se iban a llevar a cabo y cuál era la mejor manera de incorporar la información de los pasados correspondientes. Ejecuciones de algunos de estos experimentos tanto en el ordenador local como en la máquina virtual de Azure. Generación de algunas de las tablas de datos usadas como entrada en los algoritmos.

- **Predicciones con redes neuronales**
  Búsqueda y explicación de qué son las redes neuronales, qué tipo de estas vamos a usar en nuestro experimento, en qué consiste su funcionamiento y las ventajas que ofrecen. Discusión de los resultados obtenidos al aplicar el perceptrón multicapa a los datos elaboración de las gráficas y las tablas necesarias para complementar la explicación.

- **Conclusiones**
  Elaboración de parte del contenido de la memoria sobre las conclusiones finales que no ha llevado a cabo mi compañera. Obtención de la gráfica comparativa en la que se

puede ver una recopilación de las predicciones que hace cada método, para un valor, muy útil para visualmente hacerse una idea de hasta qué punto de la realidad se acercan las predicciones, y para ver fácilmente con qué método se han obtenido los mejores resultados.

Capítulo 9

Conclusiones

En este capítulo se van a exponer los resultados obtenidos después de haber hecho los correspondientes experimentos con los métodos de aprendizaje automático que nos han parecido oportunos.

En la siguiente tabla se pueden ver los errores obtenidos al predecir los incrementos con los métodos de regresión múltiple, perceptrón multicapa y Naïve. Los errores de estos métodos de aprendizaje automático **no se pueden comparar** directamente con los errores obtenidos en el capítulo 5 de ARIMA, ya que en esta predicción no se emplearon los incrementos, sino los valores reales escalados. Como se ve en la discusión de resultados del capítulo 5, para nuestros valores estudiados (*CCMP_Index_Open* y *NKY_Index_Open*), en general, Naïve se ha comportado en la predicción a 30 días mejor que ARIMA, por lo que el error (la medida que hemos usado para medir la precisión de los modelos) a mejorar es el resultado de una predicción con Naïve; este valor sí lo calculamos con los incrementos (el conjunto de datos para los que hicimos las predicciones con regresión múltiple, el perceptrón multicapa y también Naïve), por lo tanto, **toda aquella predicción de los incrementos que mejore el valor del error Naïve estará mejorando a su vez el de ARIMA.**

	CCMP_Index_Open	**NKY_Index_Open**
Linear regression	0.064507226	0.08491333
Gradient Boosting Regressor	0.06145602	0.071885368
Random Forest Regressor	0.05531515	0.08521506
Voting Regressor	0.053564	0.080648
Perceptrón multicapa	0,05049577255548418	0,069722115189414
Naïve	0.062849668149	0.0757686795150164

Cuadro 9.1: Tabla de los mejores resultados usando incrementos

Vista la tabla 9.1 y las discusiones de los resultados realizadas en cada capítulo se puede decir lo siguiente: habiendo elegido para predecir índices con una tendencia en el tiempo bastante lineal, es de esperar que tanto ARIMA como Naïve dieran resultados aceptables, nuestro reto era intentar mejorar estos resultados, y estas son las conclusiones principales a las que hemos llegado:

- Aunque en algunos casos valores bursátiles pueden predecirse considerablemente bien a partir del histórico de él mismo (como hemos visto para algunos índices en el 5), la inclusión de más variables independientes mejora la predicción. Podemos concluir entonces que sí es cierto que el comportamiento de un valor está relacionado con los pasados de otros determinados valores bursátiles.

- Las series en las que mejor resultado se puede obtener tanto en ARIMA como Naïve son aquellas en las que predomina una tendencia lineal (en las que se ha centrado este estudio). Por lo tanto, utilizando otros valores de nuestro conjunto de datos que no formen una serie con tendencia lineal, los errores de las predicciones de los métodos lineales (ARIMA y Naïve) tendrán una diferencia con el error que hemos experimentado con los métodos de aprendizaje automático (regresión y el perceptrón multicapa) mucho mayor, ya que estos últimos se adaptan mejor a regresiones no lineales. Es decir, los valores que hemos escogido en este estudio, ya que fueron elegidos por medio de `auto.arima()`, son "fáciles" de predecir con métodos lineales; si escogiéramos unos valores sin esta propiedad de tendencia lineal, sería más notable la superioridad de predicción de los métodos de aprendizaje automático.

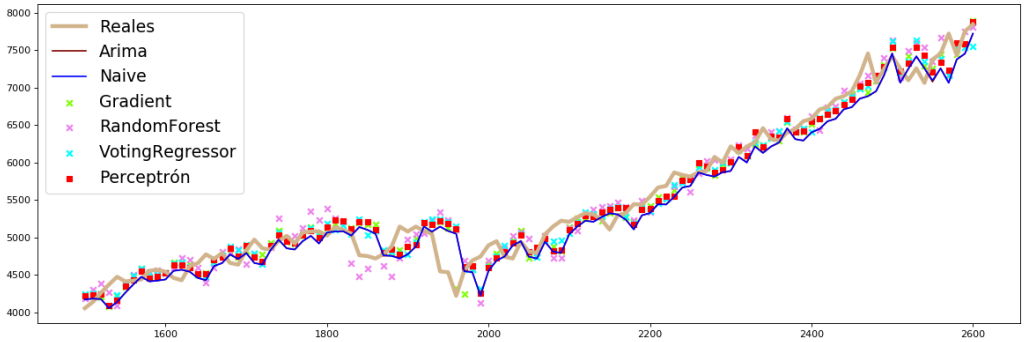
- Al incorporar datos sobre el volumen de los índices al conjunto de entrenamiento hemos podido comprobar que, en la mayoría de casos, para predecir el índice de apertura de un valor, las mejores variables independientes son las de los volúmenes de otros índices bursátiles. Como se indicaba en la subsección 4.1.1, el volumen puede ser un precursor de los cambios en los precios, y con estos experimentos hemos podido comprobar que esto es cierto.

- Aunque los resultados han sido satisfactorios, hemos encontrado evidencias de la existencia tanto de mínimos locales, como de índices que aportan información muy similar al modelo, por lo que se deberían hacer múltiples ejecuciones de cada experimento para intentar descartar mínimos locales y probar a aplicar alguna técnica de agrupamiento de valores para para confirmar la existencia de estos grupos de valores que se comportan de manera parecida.

- En ambos casos el perceptrón ha resultado ser método con el que mejores resultados se han obtenido y que menos tiempo de ejecución ha empleado.

- Haciendo experimentos hemos podido observar que tanto si se usa un pasado de 30 como si es de solo 2 los resultados apenas varían, esto ya se ha comentado en la sección 7.4, la razón es que aunque para un valor solo se usen dos pasados, para el siguiente valor se usará uno de los pasados que se usó para el anterior valor y un pasado nuevo; si esto lo generalizamos, finalmente, el modelo tiene la información de todos los pasados, la diferencia entre usar 2 pasados y 30 es la manera de pasarle la información al modelo para que aprenda.

A continuación, se puede observar una gráfica en la que visualmente se puede intuir qué método es el que ofrece los mejores resultados. Efectivamente, este método es el perceptrón, en la gráfica no se pueden apreciar los valores predichos por ARIMA, ya que al ser tan cercanos a Naïve, se solapan las gráficas. Se observa que en muchos de los casos las predicciones del perceptrón son las más cercanas al valor real y le sigue de cerca el Voting Regressor.



Fuente: Elaboración propia

Figura 9.1: Gráfica de las predicciones de todos los métodos para el *CCMP_ Index_ Open*

Finalmente, centrándonos en el *CCMP_ Index_ Open*, tenemos que como variables independientes: *LMAHDS03_ Comdty* (aluminio), ha sido el valor que en más métodos ha sido elegido como variable independiente para la predicción y el *CRY_ Index_ Open* junto al *MO1_ Comdty_ Open* (CO2), con los que se ha obtenido el mejor resultado utilizando el perceptrón multicapa.

Capítulo 10

Conclusions

In this chapter we will reveal the results obtained after performing the corresponding experiments with the methods that seemed appropriate.

In table 9.1 you can see the errors obtained when predicting the increments with the multiple regression methods, the multilayer perceptron and Naïve. These **cannot be compared** directly with the errors obtained in chapter 5 of ARIMA; since in this prediction we used the (scaled) real values instead of the increments, as seen in the discussion of results in chapter 5. For our values studied (*CCMP_Index_Open* and *NKY_Index_Open*) Naïve behaved better in the 30 day forecast than ARIMA. Thus, the error (the measurement we have used to measure the accuracy of the models) to improve is the result of a prediction with Naïve. This value is calculated with the increments (the set of data for which we made the predictions with multiple regression, the multilayer perceptron and also Na ïve). Therefore, **all the predictions of the increments that improve the value of the error Naïve will in turn improve ARIMA**.

Considering the table 9.1 and the discussions of the results included in each chapter, we observe that we have chosen values with a fairly linear trend in time. Thus, it is expectable that both ARIMA and Naïve yield acceptable results. Hence, our challenge was to try to improve these results, and these are the main conclusions to which we have reached:

- Although in some cases, the values can also be predicted throughout their own historic data (as we have seen for some stock indices in chapter 5), the inclusion of more variables independently improves the prediction. We can conclude that it is true that the behavior of a value is related to the past of other stock market values.

- The series in which the best result can be obtained in both ARIMA and Naïve are those in which a linear trend predominates (which this study has focused on). Therefore, using other values of our data set that does not form a series with a linear trend, the errors of the predictions of the linear methods (ARIMA and Naïve) will have a difference with the error we have experienced with the machine learning methods (regression and the multilayer perceptron). The difference is much higher since the latter adapt better to nonlinear regressions. That is to say, the values that we have chosen in this study, (since they were chosen by means of `auto.arima()`), are "easy" to predict with linear methods; if we chose values without this property of linear

tendency, the superiority of prediction of machine learning methods would be more noticeable.

- By incorporating data on the volume of the indices into the training set, we have been able to verify that: in most cases, to predict the value of the open index, the best independent variables are those of the volumes of other stock indices. As indicated in subsection 4.1.1, volume can be a precursor to changes in prices, and with these experiments we have been able to verify that this is true.

- Although the results have been satisfactory, we have found evidence of the existence of both local minimums and indices that provide information very similar to the model. So multiple executions of each experiment should be done to try to discard local minimums and to try to apply some technique of grouping values to confirm the existence of these groups that behave similarly.

- In both cases the perceptron has turned out to be the method with which the best results have been obtained and the least time of execution has been used.

- By doing experiments we have observed that whether a past of 30 is used or only a past of 2, the results hardly change. This has already been commented on in section 7.4. The reason is that, for a single value two pasts are used but for the following single value one of the pasts is used from the previous value, as well as a new pasts. If we generalize this, finally, the model has the information of all the past pasts, the difference between using 2 pasts and 30 is the way to introduce the information to the model so that it can learn.

Next, you can see in Figure 9.1 a graph in which you can visually guess which method offers the best results. Indeed, this method is the perceptron. In the graph the values of ARIMA are not included, since being so close to Naïve, the graphs overlap. It is observed that in many of these cases the perceptron predictions are closest to the real value and closely followed by the Voting Regressor.

Finally, focusing on the *CCMP _Index _Open*, we have *LMAHDS03 _Comdty* (aluminum) as an independent variable, which has been the value that in more methods has been chosen as an independent variable for the prediction. Our two other independent variables are *CRY _Index _Open* and *MO1 _Comdty _Open* (CO 2), with which the best result was obtained using the multilayer perceptron.

Bibliografía

[A. Victor Devadoss, 2013] A. Victor Devadoss, T. A. A. L. (2013). Forecasting of stock prices using multi layer perceptron. *International Journal of Web Technology*, 2.

[Adebiyi et al., 2014] Adebiyi, A. A., Adewumi, A. O., and Ayo, C. K. (2014). Comparison of arima and artificial neural networks models for stock price prediction. *Journal of Applied Mathematics*, 2014.

[Akaike, 1998] Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer.

[Altay and Satman, 2005] Altay, E. and Satman, M. H. (2005). Stock market forecasting: artificial neural network and linear regression comparison in an emerging market. *Journal of Financial Management & Analysis*, 18(2):18.

[An and Meng, 2010] An, K. and Meng, J. (2010). Voting-averaged combination method for regressor ensemble. In *International Conference on Intelligent Computing*, pages 540–546. Springer.

[Babyak, 2004] Babyak, M. A. (2004). What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. *Psychosomatic medicine*, 66(3):411–421.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[Calvo, 2008] Calvo, D. (2008). Función de activación – Redes neuronales. `http://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/`.

[Chen and Chen, 2008] Chen, J. and Chen, Z. (2008). Extended bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771.

[Das, 1994] Das, S. (1994). *Time series analysis*. Princeton University Press, Princeton, NJ.

[Dutta et al., 2012] Dutta, A., Bandopadhyay, G., and Sengupta, S. (2012). Prediction of stock performance in indian stock market using logistic regression. *International Journal of Business and Information*, 7(1).

[Elith et al., 2008] Elith, J., Leathwick, J. R., and Hastie, T. (2008). A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4):802–813.

[Guha and Bandyopadhyay, 2016] Guha, B. and Bandyopadhyay, G. (2016). Gold price forecasting using arima model. *Journal of Advanced Management Science Vol*, 4(2).

[Khandelwal et al., 2015] Khandelwal, I., Adhikari, R., and Verma, G. (2015). Time series forecasting using hybrid arima and ann models based on dwt decomposition. *Procedia Computer Science*, 48:173 – 179. International Conference on Computer, Communication and Convergence (ICCC 2015).

[Matich, 2001] Matich, D. J. (2001). Redes neuronales: Conceptos básicos y aplicaciones. *Universidad Tecnológica Nacional, México*.

[Mills and Mills, 1991] Mills, T. C. and Mills, T. C. (1991). *Time series techniques for economists*. Cambridge University Press.

[Mitchell et al., 1990] Mitchell, T., Buchanan, B., DeJong, G., Dietterich, T., Rosenbloom, P., and Waibel, A. (1990). Machine learning. *Annual review of computer science*, 4(1):417–433.

[Peter and Silvia, 2012] Peter, Ď. and Silvia, P. (2012). Arima vs. arimax–which approach is better to analyze and forecast macroeconomic time series. In *Proceedings of 30th International Conference Mathematical Methods in Economics. Karviná, Czech Republic*, pages 136–140.

[Shalev-Shwartz and Ben-David, 2014] Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

[Zhang, 2003] Zhang, G. P. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175.