Graduado en Matemáticas e Informática

Universidad Politécnica de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

TRABAJO FIN DE GRADO

Predicción de fluctuaciones de la Bolsa basado en las noticias

Autor: Manuel Rodríguez

Director: Damiano Zanardini

MADRID, JUNIO 2019

To my parents, for your unlimited love & support.

Resumen

Este proyecto consiste en la implementación de un algoritmo de aprendizaje automático que, a partir de las noticias (de toda índole) más importantes a nivel mundial, intente predecir cómo se verán afectados los principales índices bursátiles del mundo. Para este estudio, se han elegido índices relevantes de las principales zonas económicas, como son:

- **GSPC**. *S&P500*. [1] El índice Standard & Poor's 500 (Standard & Poor's 500 Index), también conocido como S&P 500 es uno de los índices bursátiles más importantes de Estados Unidos. Al S&P 500 se lo considera el índice más representativo de la situación real del mercado. El índice se basa en la capitalización bursátil de 500 grandes empresas que cotizan en las bolsas NYSE o NASDAQ y captura aproximadamente el 80 % de toda la capitalización de mercado en Estados Unidos.
- **ILF**. *S&P Latin America 40*. [2] El S&P Latin America 40 es un índice bursátil de Standard & Poor's. Realiza un seguimiento del mercado bursátil latinoamericano. El S&P Latin America 40 es uno de los siete índices principales que componen el S&P Global 1200 e incluye valores altamente líquidos de los principales sectores económicos de los mercados de renta variable mexicanos y sudamericanos. En este índice están representadas empresas de Brasil, Chile, Colombia, México y Perú, y representa aproximadamente el 70 % de la capitalización bursátil de cada país. Proporciona cobertura de los componentes líquidos de gran capitalización de cada país clave de América Latina.
- **N225**. *Nikkei 225*. [3] Nikkei 225, comúnmente denominado índice Nikkei, es el índice bursátil más popular del mercado japonés. Lo componen los 225 valores más líquidos que cotizan en la Bolsa de Tokio. Desde 1971, lo calcula el periódico Nihon Keizai Shinbun (Diario Japonés de los Negocios), de cuyas iniciales proviene el nombre del índice.
- **SPEUP**. *S&P Europe 350*. [4] El S&P Europe 350 Index es un índice bursátil de valores europeos. Es parte del S&P Global 1200. Las acciones constituyentes se seleccionan en función de su relevancia para el mercado en general, incluyendo el equilibrio del sector industrial, la longevidad (para minimizar la rotación del índice) y la liquidez de las acciones.

- **STOXX50E**. *Euro Stoxx 50*. [5] El EURO STOXX 50 es un índice bursátil de valores de la zona euro diseñado por STOXX, un proveedor de índices propiedad de Deutsche Börse Group. Según STOXX, su objetivo es "proporcionar una representación de primer orden de los líderes de Supersector en la Eurozona". Se compone de cincuenta de las mayores y más líquidas existencias. Los futuros y opciones sobre índices del EURO STOXX 50, negociados en Eurex, se encuentran entre los productos de este tipo más líquidos de Europa y del mundo.
- **SHA: 000001**. *SSE Composite Index*. [6] El SSE Composite Index es un índice bursátil con todos los valores (acciones clase A y clase B) que se negocian en la bolsa de Shanghái - Shanghai Stock Exchange (SSE).

El desarrollo de este trabajo ha consistido, en primer lugar, en un análisis del lenguaje sobre los titulares, en dos fases: primero extrayendo el sentimiento, es decir, lo positivo o negativo que sea el titular de la noticia y en segundo lugar, extrayendo los conceptos contenidos en el mismo. Por ejemplo, si el titular de la noticia es:

"May's Brexit Deal Defeated 202-432"

Las entidades extraídas serían May, por la política inglesa *Theresa May* y *Brexit*, el conocido proceso de independencia británica que tanta relevancia ha adquirido en los últimos meses. Junto con estos conceptos, se extrae también el análisis de sentimiento, resumido en un valor entre -1 y 1, que nos indicará si el titular es positivo o negativo.

En conjunción con las fluctuaciones en el valor de los índices anteriormente mencionados se prepara el sistema para que la máquina pueda aprender de los datos históricos y ser, de esta forma, capaz de emitir predicciones en base a los eventos más notables de un período de tiempo.

A nivel técnico, la base de todo el trabajo se ha realizado en el lenguaje de programación Python, muy utilizado en el campo de la Inteligencia Artificial y el *Machine Learning*, en inglés, también conocido como Aprendizaje Automático en español. Además, se han utilizado una serie de librerías para este lenguaje, de las cuales podemos destacar *Pandas*, *NumPy* y *scikit-learn*.

Abstract

The aim of this project has been to implement a machine learning algorithm that, based on the most relevant worldwide news, tries to predict how the main stock market indexes around the world will be affected. For this study, relevant indexes of the main economic zones have been chosen, such as:

- **GSPC**. *S&P500*. [1] The S&P 500, or just the S&P, is an American stock market index based on the market capitalizations of 500 large companies having common stock listed on the NYSE, NASDAQ, or the Cboe BZX Exchange.
- **ILF**. *S&P Latin America 40*. [2] The S&P Latin America 40 is a stock market index from Standard & Poor's that tracks Latin American stocks. The S&P Latin America 40 is one of seven headline indices making up S&P Global 1200 and includes highly liquid securities from major economic sectors of Mexican and South American equity markets. Companies from Brazil, Chile, Colombia, Mexico and Peru are represented in this index, accounting for approximately 70 % of each country's market capitalization. This index provides coverage of the large cap, liquid constituents of each key country in Latin America.
- **N225**. *Nikkei 225*. [3] The Nikkei 225, more commonly called the Nikkei, the Nikkei index, or the Nikkei Stock Average, is a stock market index for the Tokyo Stock Exchange (TSE). It has been calculated daily by the Nihon Keizai Shinbun (The Nikkei) newspaper since 1950. It is a price-weighted index, operating in the Japanese Yen, and its components are reviewed once a year. The Nikkei measures the performance of 225 large, publicly owned companies in Japan from a wide array of industry sectors.
- **SPEUP**. *S&P Europe 350*. [4] The S&P Europe 350 Index is a stock index of European stocks. It is a part of the S&P Global 1200. The constituent shares are selected for relevance to the broad market, including industry sector balance, longevity (to minimize index turnover) and liquidity of the shares.
- **STOXX50E**. *Euro Stoxx 50*. [5] The EURO STOXX 50 is a stock index of Eurozone stocks designed by STOXX, an index provider owned by Deutsche Börse Group. According to STOXX, its goal is "to provide a blue-chip representation of Supersector leaders in the Eurozone". It is made up of fifty of the

largest and most liquid stocks. The index futures and options on the EURO STOXX 50, traded on Eurex, are among the most liquid such products in Europe and the world.

- **SHA: 000001**. *SSE Composite Index*. [6] The SSE Composite Index also known as SSE Index is a stock market index of all stocks (A shares and B shares) that are traded at the Shanghai Stock Exchange.

The development has consisted, in the first place, of a language analysis on the headlines, extracting the sentiment, in other words, how positive or negative the headline is, together with the main concepts that appear in it. For example, if the headline were:

"May's Brexit Deal Defeated 202-432"

The entities extracted would be May, referring the English politician *Theresa May* and *Brexit*, the well-known process of British independence that has acquired so much relevance in recent months. Along with these concepts, the sentiment analysis is also extracted, summarized in a value between -1 and 1, which will indicate whether the headline is positive or negative.

In conjunction with the fluctuations in the value of the indexes mentioned above, the system is prepared so that the machine can learn from historical data and thus be able to issue predictions based on the most remarkable events from a period of time.

From a technical point of view, the core of all the work has been done under the Python programming language, widely used in the field of Artificial Intelligence and Machine Learning, along with a series of libraries, such as *Pandas*, *NumPy* and *scikit-learn*.

Índice general

Capítulo 1

Introducción

La inteligencia artificial ha sido, sigue y seguirá siendo un tema candente en nuestra sociedad. Ha pasado por varias etapas: antiguamente se veía como algo lejano, casi místico, representado en libros y películas como una especie de 'maldición' que acabaría con la especie humana.

Afortunadamente, en la última década hemos podido comprobar como la IA (Inteligencia Artificial), en todas sus ramas, nos permite avanzar en todos los campos de manera exponencial. Coches que se conducen solos, asistentes personales y detección preventiva de enfermedades son solo algunos de los éxitos que se han logrado últimamente.

Sin duda alguna, será la rama de la ciencia que revolucionará (y lo está haciendo) nuestra manera de entender el mundo. Con ella, naturalmente, se presentarán una serie de problemas a nivel ético, social y económico, que habrá que abordar en paralelo con los retos técnicos. Esta problemática, no obstante, queda fuera del alcance de este trabajo.

Dentro de las ramas de la Inteligencia Artificial se encuentra el *Machine Learning* o Aprendizaje Automático que, como su nombre indica, consistente en la enseñanza programática de una máquina. En vez de establecer reglas concretas como, $x = 2$, aportamos a la máquina una serie de datos a partir de los cuales ella pueda hacer pruebas y establecer relaciones que permitan realizar predicciones de manera autónoma.

En este proyecto hemos utilizado esta capacidad que nos ofrece el aprendizaje automático: las predicciones. A partir de una serie de datos históricos, transformados a valores numéricos, la máquina intentará comprender por qué ocurren ciertos eventos y establecerá así una relación causa-efecto que luego podremos utilizar para intentar averiguar que sucederá en el futuro.

Como ya se ha comentado, para poder implementar este tipo de aprendizaje son necesarios tres factores:

1. Cantidad de datos
2. Accesibilidad de los datos
3. Constancia de los datos

Sin duda alguna, el mundo bursátil cumple perfectamente los requisitos. Hay una gran cantidad de precios, índices y estadísticas disponibles, que se remontan décadas. Estos datos a su vez son relativamente accesibles al público y se puede disponer de un histórico para la mayoría de valores financieros. Por último, es un mundo constante en el sentido de que, salvo catástrofe, la bolsa de Nueva York, de Madrid, de Londres... abrirán de manera puntual el lunes por la mañana y cerrarán el viernes por la tarde. Esto nos permite comprobar la eficacia de nuestro trabajo rápidamente e incluso en tiempo real.

Nuestro planteamiento es el siguiente: Si bien es cierto que la economía es impredecible por naturaleza, pues depende en su mayor parte, del comportamiento humano, es un hecho que hay eventos globales que afectan a los mercados por su gran impacto en el día a día de la sociedad.

Pongamos por caso la situación política en Venezuela. Hay muchos intereses en la región, por su riqueza en en recursos naturales, principalmente petróleo. Al momento de escribir este Trabajo Fin de Grado todavía no se conoce el desenlace de la situación. Pero habrá ganadores y perdedores. Por tanto, se trata de un evento que, irrevocablemente, va a afectar a los mercados, lo cual implica cierta volatilidad. Cuánto y cómo son preguntas difíciles de responder, pero está claro que el precio de los valores cotizados subirá o bajará para reflejar el panorama mundial. He aquí el objetivo de este Trabajo Fin de Grado: predecir, a partir de los sucesos que ocurren cada día, cómo se verán afectadas las principales economías del mundo, reflejadas en los índices bursátiles.

1.1. Conceptos bursátiles

Para comprender bien el objeto de investigación de este Trabajo, a continuación vamos a listar y explicar algunos conceptos de la manera más sencilla posible.

- **Acción**. [7] Una acción es una unidad de propiedad en una empresa que se puede poner a la venta a inversores. El valor total de la empresa se divide en unidades del mismo tamaño y cada una de las unidades se conoce como acción.

Para contextualizar, si una compañía vale 200 millones de dólares y emite 100 millones de acciones, cada acción tiene un valor de 2 dólares, o 200 céntimos de dólar. Cuando fluctúa el valor de la empresa, también lo hace el precio de sus acciones. Así, los inversores que compran acciones en una compañía tienen la esperanza de que aumente de valor, permitiéndoles vender las acciones a un precio mayor. Las acciones también se conocen como títulos o valores.

- **Bolsa**. La bolsa es el mercado donde se pueden comprar y vender activos financieros, como acciones o índices.
- **Índice**. Un índice bursátil es un activo financiero que es el resultado de agrupar acciones de diferentes empresas. Por ejemplo, en España está el IBEX 35, que es la agrupación del precio de una acción de las 35 empresas más valoradas.

Estos y más conceptos se pueden encontrar en la bibliografía.

1.2. Objetivos

Con este trabajo de investigación, queremos cumplir, en la medida de lo posible, los siguientes objetivos:

- Obtener datos históricos, darles formato de acorde a las necesidades del proyecto y prepararlos para ser implementados.
- Configurar un sistema que funcione y aprenda correctamente.
- Poder predecir, con un aceptable margen de error, si un determinado índice bajará o subirá, basándose en los acontecimientos de un determinado período de tiempo.

Capítulo 2

Trabajos Previos

El aprendizaje automático (*Machine Learning*, en inglés) está provocando una auténtica revolución en todos los campos del conocimiento. Si bien avances en la medicina pueden mejorar la vida de millones de personas y avances en la industria automóvil pueden hacer nuestra vida más sencilla y segura, progresos en la industria de la IA tienen un potencial infinitamente superior. En el siglo XXI, un pequeño desarrollo en el aprendizaje automático afecta enormemente al resto de disciplinas.

Por ser todos estos avances de reciente desarrollo, aún quedan muchos campos sin explorar. En el caso de este trabajo, no hemos sido capaces de encontrar ningún proyecto igual. Similares, sí. En el mismo ámbito, también. Pero nada por mejorar. Todo por crear.

La bolsa es principalmente numérica, lo cual facilita mucho la implementación de técnicas de *Machine Learning*, que tienen una gran base matemática. Además cuenta con una cualidad muy importante: genera dinero. Es decir, resultados positivos en el estudio de la bolsa pueden hacer a su creador rico. Naturalmente, si fuera tan sencillo, todo los investigadores serían millonarios y al final, la investigación dejaría de tener nada de especial. Este trabajo será nuestro humilde intento para predecir las variaciones de los mercados bursátiles a partir de los eventos que ocurren a diario.

Hay incontables proyectos de una índole similar, como este aplicando las redes sociales [8], que es el más parecido al nuestro. De hecho, era la idea inicial para este Trabajo Fin de Grado: utilizar Twitter para intentar predecir la volatilidad de los índices. Finalmente, optamos por utilizar los principales titulares de un día dado, pues se les presupone cierto *rigor* que las redes sociales simplemente no tienen. Una desventaja clara es el volumen de datos. En este proyecto se recogen los diez titulares más relevantes del día, por lo que es un caso de estudio bastante corto. Por otro lado, hay miles y miles de *tweets* de inversores, tanto profesionales como novatos. Lo cual, teóricamente, mejora el análisis, al tener más datos para comparar. Al final, nos decantamos por las noticias, pues son mucho más fáciles de obtener que los *tweets*

y son más rigurosas y fidedignas que comentarios en las redes sociales.

El resto de intentos por predecir los mercados bursátiles siguen una línea más teórica, recurriendo al análisis técnico. En la misma Universidad Politécnica de Madrid tenemos un TFG de este estilo [9]. Otros portales especializados en español también tienen sus propios casos.

Sin embargo, resulta lógico pensar que las noticias más importantes puedan dictar las oscilaciones del mercado. Al final, el mercado responde a la confianza de los inversores. Y la confianza de los inversores se debería ver afectada por el *Brexit*, por la situación EE.UU-Corea del Norte y por el resto de sucesos mundiales que ocurren a diario. Averiguar si existe esta relación y si esta relación es recurrente y predecible es el objetivo de este proyecto de investigación.

Capítulo 3

Desarrollo

3.1. Obteniendo las noticias más relevantes

3.1.1. Reddit

El primer paso en el desarrollo de la aplicación fue obtener las noticias más importantes para un día dado. Al inicio pensamos utilizar Twitter como fuente de datos, pero su API es demasiado engorrosa y limitada. Además, los datos no son enteramente fiables, pues hay bastante subjetividad dependiendo de dónde se obtengan los titulares. Si por ejemplo fuéramos a analizar los titulares de un periódico más conservador, seguramente estarían inclinados hacia al eje político derecho. Por otro lado, solamente íbamos a analizar el titular y no el cuerpo de la noticia, por lo que la objetividad y la concreción eran claves.

Otro problema que se nos planteó fue el idioma. Las técnicas de NLP (*Natural Language Processing*), es decir, las técnicas para poder analizar el lenguaje, desafortunadamente no están tan desarrolladas en español como lo están en inglés. Por lo tanto, pasamos a la búsqueda de una fuente de datos en inglés que fuera lo más rigurosa posible y que recopilase noticias no solo de su país de interés, sino también a nivel mundial.

Finalmente, optamos por utilizar Reddit, una plataforma creada en 2005 y con gran presencia en países anglosajones, que funciona como un *foro de foros*, lo que hace que se puedan encontrar en la misma casi todos los sub-foros imaginables aquí. La dirección raíz de Reddit es reddit.com. Cada sub-foro se conoce como *sub-reddit*, y se puede acceder a ellos añadiendo un r/NOMBRE al final de la dirección raíz. Por ejemplo, está r/Python, donde se agrupan las principales guías y noticias sobre este lenguaje. También está r/Formula1, donde los usuarios comentan la actualidad de la Fórmula 1. Nosotros optamos por r/WorldNews [10], que recopila diariamente

las noticias más relevantes del mundo, en inglés. Reddit cuenta también con un sistema de votos, en el cual los usuarios pueden votar (positiva o negativamente) las publicaciones de cada foro y los mejor valorados aparecen en la portada del sub-foro durante el tiempo que sea relevante.

Reddit por este lado ya tenía una gran ventaja con respecto al resto de redes sociales, pues de una manera sencilla nos agrupaba los titulares de mayor relevancia de todo el mundo, y además nos los clasificaba por importancia. Pero el factor clave para decantarnos por Reddit fue su comunidad.

Hoy en día, Internet es una maraña de titulares engañosos, que tienen como único fin conseguir *clicks* y repercusión, cueste lo que cueste. Es un concepto que se conoce como *clickbait* y que consiste en exagerar sobremanera un evento o presentarlo de manera diferente con el objetivo de llamar la atención del usuario. Pues bien, los usuarios de Reddit están muy concienciados con estas artimañanas y votan los titulares *clickbait* negativamente, y por tanto les restan importancia, de manera que no aparecen en el listado de noticias más relevantes.

3.1.2. API

Una vez teníamos una fuente de datos que nos iba a reportar resultados fiables y a partir de los cuales la máquina iba a poder aprender, llegaba la hora de exportarlos a un formato que pudiésemos utilizar. Desafortunadamente, la API de Reddit, en el momento del desarrollo había sufrido numerosos cambios y uno de ellos era la imposibilidad de limitar los resultados de búsqueda a una fecha concreta. Pero hay una API no oficial de Reddit [11], mantenida por un solo usuario, que hace de copia de seguridad para la gran mayoría de publicaciones. Esta API sí nos permitía recuperar los temas más votados para cierto día. De esta manera, una llamada a:

https://api.pushshift.io/reddit/search/submission

Con los argumentos siguientes:

- *subreddit* : worldnews
- *after* : 2019-01-15
- *before* : 2019-01-15
- *size* : 10
- *sort* : desc

Nos devuelve el siguiente resultado:

1. May's Brexit Deal Defeated 202-432
2. Trump Repeatedly Discussed Withdrawing U.S. From NATO: NYT
3. Vladimir Putin Directly Supported a Russian Plot to Infiltrate the NRA and Sow Discord in U.S., Report Claims
4. The Oceans Are Warming Fast, and Our Lives Are About to Change | A paper published in the journal Science shows that the Earth's oceans are warming at a rate that's about 40 percent faster than indicated in the 2013 U.N. Intergovernmental Panel on Climate Change report
5. A senator from Italy's far-right League has been given an 18-month prison sentence for likening the country's first black minister to an orangutan.
6. Ivanka Trump to help select candidate to lead World Bank
7. Canadians Spent $1.6 Billion on Legal Weed in 2018
8. Jared Kushner Told Donald Trump That Firing Comey and Flynn Would Help End Russia Probe, Chris Christie Says
9. McDonald's loses Big Mac trademark in the EU after legal battle with Irish chain
10. B.C. judge warned Canadian sentenced to death in China; He was Convicted of Drug Trafficking Twice in Canada before Going to China

Este proceso lo repetimos para todo el rango de fechas que nos interesaba y que venía limitado por los datos diarios de los índices bursátiles, como veremos en los próximos capítulos. De esta manera, exportamos el TOP 10 de noticias más relevantes a un fichero CSV entre el 24 de febrero de 2014 (24-02-2014) y el 31 de diciembre de 2018 (31-12-2018) para un total de 1771 días. A 10 titulares por día, teníamos 17.710 noticias para analizar.

3.2. Obteniendo los datos bursátiles

Hace unos años el popular servicio Yahoo! Finance ofrecía una API gratuita para acceder al precio de los valores bursátiles, así como a su histórico. No obstante, en el momento de realizar este trabajo, el acceso a dicha API se encontraba cerrado, por lo que no había manera gratuita de obtener el historial que necesitábamos de los índices mencionados en el resumen.

Afortunadamente, el sitio web de Yahoo! Finance [12] sigue abierto al público y permite descargar un CSV con los precios de apertura y cierra de los últimos 5 años. De ahí viene también el límite de días para recopilar las noticias de Reddit. De esta manera, fuimos descargando uno a uno los archivos, que tenían el siguiente formato:

Cuadro 3.1: Histórico de precios para el valor ^GSPC

Date	**Open**	**High**	**Low**	**Close**
2014-02-24	1836.780029	1858.709961	1836.780029	1847.609985
2014-02-25	1847.660034	1852.910034	1840.189941	1845.119995
2014-02-26	1845.790039	1852.650024	1840.660034	1845.160034

Para facilitar la comprensión, a continuación se incluye la explicación de cada uno de los valores:

- *Date* : El día de la sesión
- *Open* : El precio con el que empezó el valor la sesión
- *High* : El precio máximo que alcanzó el valor durante la sesión
- *Low* : El precio mínimo que alcanzó el valor durante la sesión
- *Close* : El precio con el que cerró el valor la sesión

Naturalmente, había que formatear estos datos para adaptarlos a lo que pasaríamos a la máquina. Comentar que estos valores son solo los básicos, en el mundo de la bolsa se analizan muchos más factores técnicos para intentar determinar las diversas oscilaciones y tendencias. Pero nuestro objetivo es predecir oscilaciones mucho más grandes, que surgen a partir de noticias y que tienen tanto impacto, que pueden mover solas el índice en cuestión.

Por este motivo, eliminamos los valores de máximo y mínimo y calculamos la diferencia entre el precio de cierre y el precio de apertura. No obstante, este enfoque presenta un problema: no se está teniendo en cuenta el horario de apertura y cierre de los mercados. Dependiendo de en qué bolsa se incluya cada índice, éstos tienen un horario de compra-venta distinto. Por ejemplo, la bolsa de Madrid está abierta de lunes a viernes de 8:30 a 17:30 (horario CET+1). Por otro lado, la bolsa de Nueva York arranca la sesión a las 9:30 de la mañana hasta las 16:00 de la tarde, pero en horario CET-5.

Luego dependiendo de a qué hora ha ocurrido el evento, puede que haya sido al comienzo, en plena sesión o incluso con el mercado ya cerrado. Desafortunadamente, el nivel de recursos y complejidad requerido para obtener y analizar estos datos correctamente se sale del alcance de este proyecto, por lo que se asume una pérdida en la precisión.

Para intentar paliar esta problemática, vamos a probar dos maneras diferentes de estudiar los datos. La primera, comparar las noticias de un día con las oscilaciones bursátiles de este mismo día. La segunda, comparar las noticias de un día con las oscilaciones del día siguiente. Según el formato que le vamos a dar a los históricos, el resultado es el siguiente:

Cuadro 3.2: Histórico de precios para el valor ^GSPC

Date	**Variación Absoluta**	**Variación Relativa**
2014-02-24	10.829956	0.5896163845976 %
2014-02-25	-2.540039	-0.1374732880107 %
2014-02-26	-0.630005	-0.03413199696 %

Donde variación absoluta es la diferencia entre el precio de cierre y el precio de apertura y variación relativa es la división entre la variación absoluta y el precio de apertura, para poder analizar cuán fuerte ha sido la variación.

3.3. Preparando la base de datos

Como ya se ha podido comprobar, vamos a trabajar con una gran cantidad de datos. La mayoría los estamos exportando a formato CSV, pero nos hace falta una opción más centralizada y que nos permita realizar consultas de manera rápida y eficiente. Por ello, optamos por crear una base de datos SQL mediante SQLite3, que nos facilita mucho el almacenamiento y acceso a los datos. Creamos varias tablas, siguiendo la estructura que aparece en el esquema de la base de datos a continuación.

Como se puede ver, hay 5 tablas:

Tabla	**Contenido**
news	Diez noticias de cada día
raw_bow	Conceptos detectados, sin filtrar
indexes	Precios de apertura, cierre, variación absoluta y relativa para cada índice
sent_analysis	Análisis de sentimiento para cada una de las diez noticias de cada día
results	Resultados de las pruebas, con la configuración utilizada

3.4. Analizando los titulares

Una vez teníamos los diez titulares más importantes de cada día, había que analizarlos. El análisis tenía dos fases: extracción de conceptos y análisis de sentimiento. La extracción de conceptos consiste en identificar los elementos claves de cada oración. El análisis de sentimiento consiste en emitir una valoración, comprendida entre -1 (negativa) y 1 (positiva) sobre el sentimiento de la oración. Veremos ejemplos de ambos en las secciones siguientes.

3.4.1. Extrayendo conceptos

El proceso a partir del cual se identifica cada palabra con una etiqueta se conoce como *Part-of-Speech tagging*, (POS) y lo hemos realizado utilizando la librería Spacy [13] para Python [14]. Una característica de este proceso es que se pueden obtener de una oración las entidades contenidas que pertenecen al mundo real. Las distintas entidades que puede reconocer esta librería son:

- *PERSON* : Una persona. Ejemplo: Trump
- *NORP* : Nacionalidades o grupos políticos. Ejemplo: Labor Party
- *FAC* : Edificios, aeropuertos. Ejemplo: LAX
- *ORG* : Empresas, instituciones. Ejemplo: Apple
- *GPE* : Países, ciudades. Ejemplo: Londres
- *LOC* : Localizaciones que no pertenecen al grupo GPE. Ejemplo: Mar Mediterráneo
- *PRODUCT* : Productos, pero no servicios. Ejemplo: iPhone

Esta librería identifica algunas entidades más, como podrían ser fechas, cantidades... pero las principales son las mencionadas anteriormente. De estas, nos decantamos por las que consideramos más relevantes: NORP, PERSON, GPE y ORG. Es muy posible que conceptos como Trump, Apple o USA tengan fuerza propia en los mercados, que sean capaces de provocar una reacción en los inversores y que además aparezcan de manera relativamente frecuente en las noticias. Esto último es fundamental para el aprendizaje automático, pues la máquina podrá estudiar qué sucede cuando Trump ha aparecido en una noticia, de manera positiva o negativa (gracias al valor aportado por el análisis de sentimiento). Un ejemplo:

$$\underbrace{Apple}_{ORG} \text{ is looking at buying } \underbrace{U.K.}_{GPE} \text{ startup for } \underbrace{\$1billion}_{MONEY}$$

De estas entidades, nuestro programa descartaría *$1 billion*, pues no aporta demasiado al significado del titular. Es decir, a partir de esta extracción de entidades, sabemos que Apple y U.K. han estado relacionados en una misma noticia y que han estado presentes en el listado de noticias más importantes de un día determinado. Por lo cual, a partir de la *fuerza* propia que tengan estos conceptos en los mercados, la posición que ocupe en el *ranking* de noticias y el análisis de sentimiento, intentaremos emitir una predicción sobre si los índices subirán de manera destacable, apenas se verán afectadas o bajarán de manera relevante.

Como hemos comentado, la cantidad de 1 billón de dólares no tiene ese impacto propio que tienen otras entidades. Sí se tendrá en cuenta, por otro lado, en el análisis de sentimiento, pues aporta cierta magnitud al suceso (no es lo mismo 100 dólares que 1 billón).

3.4.2. Bolsa de palabras

Una vez hemos filtrado los conceptos, el siguiente paso es crear lo que se conoce como *Bag-of-words* en inglés, o bolsa de palabras en español. Básicamente, para cada día, vamos a contar cuántas veces ha aparecido un concepto en las noticias y además vamos a ponderar su aparición según el puesto en el que hayan aparecido. Con la ponderación haremos bastantes pruebas para ver qué sistema arroja los mejores resultados. Simplemente para ejemplificar, vamos a utilizar uno que otorga más *fuerza* a la primera posición. Es el caso de $10/posición$. De esta manera, utilizando el listado de noticias generado anteriormente como ejemplo:

1. May's $\underbrace{Brexit}_{10}$ Deal Defeated 202-432
2. $\underbrace{Trump}_{5}$ Repeatedly Discussed Withdrawing $\underbrace{U.S.}_{5}$ From $\underbrace{NATO}_{5}$: NYT
3. $\underbrace{VladimirPutin}_{3,33}$ Directly Supported a $\underbrace{Russian}_{3,33}$ Plot to Infiltrate the $\underbrace{NRA}_{3,33}$ and Sow Discord in $\underbrace{U.S.}_{3,33}$, Report Claims
4. The Oceans Are Warming Fast, and Our Lives Are About to Change | A paper published in the journal Science shows that the $\underbrace{Earth's}_{2,5}$ oceans are warming at a rate that's about 40 percent faster than indicated in the 2013 $\underbrace{U.N.}_{2,5}$ Intergovernmental Panel on Climate Change report
5. A senator from $\underbrace{Italy's}_{2}$ far-right League has been given an 18-month prison sentence for likening the country's first black minister to an orangutan.
6. $\underbrace{IvankaTrump}_{1,67}$ to help select candidate to lead $\underbrace{WorldBank}_{1,67}$
7. $\underbrace{Canadians}_{1,43}$ Spent $1.6 Billion on Legal Weed in 2018
8. $\underbrace{JaredKushner}_{1,25}$ Told $\underbrace{DonaldTrump}_{1,25}$ That Firing $\underbrace{Comey}_{1,25}$ and $\underbrace{Flynn}_{1,25}$ Would Help End $\underbrace{Russia}_{1,25}$ Probe, Chris Christie Says

9. $\underbrace{McDonald's}_{1,11}$ loses Big Mac trademark in the $\underbrace{EU}_{1,11}$ after legal battle with $\underbrace{Irish}_{1,11}$ chain

10. B.C. judge warned $\underbrace{Canadian}_{1}$ sentenced to death in $\underbrace{China}_{1}$; He was Convicted of Drug Trafficking Twice in $\underbrace{Canada}_{1}$ before Going to $\underbrace{China}_{1}$

De todos los conceptos mencionados para este día, se suman las puntuaciones que ha ido recibido cada uno. Por ejemplo, el concepto *U.S.* ha aparecido en la noticia número 2 y 3, recibiendo así una puntuación total de 8.33 para esta fecha. La puntuación para el resto de conceptos se calcula de igual manera:

- Brexit = 10
- Trump = 5
- U.S. = 5 + 3.33 = 8.33
- NATO = 5
- Vladimir Putin = 3.33
- Russian = 3.33
- NRA = 3.33
- Earth's = 2.5
- U.N. = 2.5
- Italy's = 2
- Ivanka Trump = 1.67
- World Bank = 1.67
- Canadians = 1.43
- Jared Kushner = 1.25
- Donald Trump = 1.25
- Comey = 1.25
- Flynn = 1.25
- Russia = 1.25
- McDonald's = 1.11

- EU = 1.11
- Irish = 1.11
- Canadian = 1
- China = 1 + 1 = 2
- Canada = 1

Se puede ver claramente que hay un problema, pues Trump y Donald Trump son la misma entidad. Igual que *Italy's* y *Italy*. Para resolverlo, utilizaremos la librería NLP de Google. Esta librería permite relacionar conceptos. Así, si la frase es:

"U.S. does not join plastic waste agreement signed by all 187 countries except USA"

La API de NLP de Google detectaría correctamente U.S., USA como concepto, como otras muchas APIs disponibles. Pero además nos diría que USA es un concepto que hace referencia a U.S. Siguiendo esta lógica, podemos pasarle la siguiente *frase* a la API:

"Donald Trump, Trump, Canada, Canadian, China, Russia, Russian, Canadians"

Ahora relacionaría Donald Trump con Trump, Canada con Canadian y Canadians y Russia con Russian correctamente.

3.4.3. Análisis de sentimiento

Hasta el momento nos encontramos con un listado de conceptos que sabemos que han aparecido en el TOP 10 de noticias más importantes de un día, pero no tenemos manera de saber lo positivas o negativas que serán. Para ello recurriremos al análisis de sentimiento, utilizando la librería de Google: Cloud Natural Language API [15], la cual nos dará un valor entre -1 (negativo) y 1 (positivo) para cada titular.

Si bien no es una correlación exacta entre si una noticia es positiva o negativa para los mercados, nos puede dar una idea general. Esta API analiza la intención y opinión del autor al escribir, es decir, si el autor de un texto dado se está expresando de manera positiva o negativa. Esto claramente no tiene por qué indicar que esa noticia será positiva o negativa a nivel bursátil, pero sigue siendo una opción interesante que no debemos descartar. Pongamos por ejemplo:

"U.S. reaches deal with Canada, Mexico to lift steel, aluminum tariffs, clearing key obstacle to passage of Trump trade deal"

Esto nos devuelve un valor de 0.25, lo significa que el ordenador lo entiende como positivo. Efectivamente, que Estados Unidos llegue a un acuerdo comercial con otros países es positivo para los mercados en la gran mayoría de casos, pues se está generando riqueza. Hay que recordar que la bolsa es un reflejo del crecimiento económico. Si se está generando dinero, los mercados tenderán a subir. Análogamente, si se está perdiendo dinero, los mercados tenderán a bajar.

Por otro lado, para cada algoritmo de ponderación que implementaremos, crearemos una variante en la cual utilizaremos el valor que nos proporcione la API de Google de la siguiente manera: si el valor es menor a -0.25, la ponderación que obtenga un concepto en una noticia dada será multiplicado por -1. El valor de 0.25 no es arbitrario, es el umbral marcado por Google a partir del cual nos indica que la oración analizada es negativa. Entre -0.25 y 0.25 se puede entender como neutral, por lo cual no aplicaremos cambios si obtenemos un resultado en este rango. Si es mayor de 0.25, se entiende como positivo y por lo tanto, tampoco modificaremos el resultado.

3.5. Generando el conjunto de datos

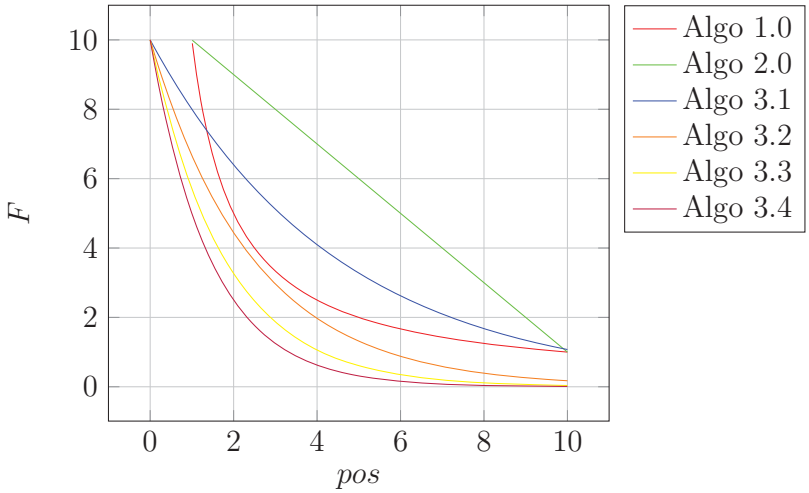
Una vez hemos recopilado todos los datos necesarios, es hora de generar un conjunto de datos que después pasaremos a los algoritmos de aprendizaje automático. Como hemos ido apuntando en anteriores capítulos, la clave de este proyecto es probar todas las variables posibles y observar cuál nos arroja mejores resultados. De esta manera, vamos a utilizar 3 algoritmos que nos darán 12 conjuntos de datos. Los algoritmos siguen la siguiente estructura, siendo F la puntuación final:

1. **Algoritmo 1**. $F = \frac{10}{pos}$. Posición 1 $\Rightarrow$ 10 puntos, Posición 2 $\Rightarrow$ 5 puntos, Posición 3 $\Rightarrow$ 3 puntos...
2. **Algoritmo 2**. $F = 11 - pos$. Posición 1 $\Rightarrow$ 10 puntos, Posición 2 $\Rightarrow$ 9 puntos, Posición 3 $\Rightarrow$ 8 puntos...
3. **Algoritmo 3**. $F = \frac{k}{exp^{pos}}$ para $k = 10$ y exp $\in (1, 2]$ con saltos de 0.25. Para $k = 10$ y $exp = 1{,}5$ tenemos $F = \frac{10}{1{,}5^{pos}}$, con lo cual Posición 1 $\Rightarrow$ 6.66 puntos, Posición 2 $\Rightarrow$ 4.44 puntos...

A partir de los algoritmos 'base', generamos otros algoritmos *espejo* a los cuales les añadimos análisis de sentimiento mediante la regla si $AS < -0{,}25 \Rightarrow F = -1 \cdot F$. De esta manera, el algoritmo 1 generará 2 conjuntos de datos, uno sin análisis de

sentimiento y otro con. El algoritmo 2, de forma análoga, generará otros 2 conjuntos de datos. Por último, el algoritmo 3 generará 2 conjuntos de datos para cada valor de exp $\in$ (1,2] con saltos de 0.25. Es decir, ocho conjuntos de datos, dando como resultado doce ficheros que son los que estudiará la máquina.

A continuación podemos ver una comparación gráfica de todos los algoritmos que vamos a emplear. Algunos le dan más importancia a estar presente en la noticia más relevante de un día y menos al resto, otros le dan una importancia igual a todos, otros un reparto de puntos mucho más gradual:



Una vez hemos explicado ya los algoritmos de clasificación, a continuación detallaremos cómo se constituye el conjunto de datos. Para cada uno de los conceptos en la bolsa de palabras, se analiza las posiciones en las que ha aparecido para cada día del período 24/02/2014 a 24/02/2019. El resultado es un conjunto de datos *madre* que guardamos en formato PKL, para poder importarlo en Python. Su formato es el siguiente:

Cuadro 3.3: Conjunto de datos madre

Date	**government**	**Egyptian**	**Pope**	**finances**	**...**
2014-02-24	1,9,	1,	2,	2,	...
2014-02-25	2,	0	0	0	...
2014-02-26	6,6,	0	0	0	...
2014-02-27	0	0	0	0	...
2014-02-28	0	0	0	0	...
2014-03-01	0	0	0	0	...
2014-03-02	2,	0	0	0	...
2014-03-03	0	0	0	0	...
2014-03-04	0	0	0	0	...
2014-03-05	1,	0	0	0	...

Como vemos, el día 24 de febrero del 2014, el concepto 'gobierno', apareció en la posición 1 y 9. A partir de esto, podemos computar su *fuerza* para ese día para cada uno de los 3 algoritmos. En el caso de aquellos que utilicen análisis de sentimiento, se hace también una consulta a la base de datos de noticias para ver si la posición 1 fue una noticia positiva o negativa y actuar en consecuencia. No es factible listar cómo son los 12 conjuntos de datos resultantes, pero mostraremos un resumen reducido para el Algoritmo 1, Algoritmo 2, Algoritmo 3.2 con $exp = 1{,}5$ y sus equivalentes con análisis de sentimiento.

Cuadro 3.4: Algoritmo 1

Date	government	Egyptian	Pope	finances
2014-02-24	11.11	10	5	5
2014-02-25	5	0	0	0
2014-02-26	3.33	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0	0	0
2014-03-01	0	0	0	0
2014-03-02	5	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	10	0	0	0

Cuadro 3.5: Algoritmo 2

Date	government	Egyptian	Pope	finances
2014-02-24	12	10	9	9
2014-02-25	9	0	0	0
2014-02-26	10	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0	0	0
2014-03-01	0	0	0	0
2014-03-02	9	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	10	0	0	0

Cuadro 3.6: Algoritmo 3

Date	**government**	**Egyptian**	**Pope**	**finances**
2014-02-24	6.93	6.67	4.44	4.44
2014-02-25	4.44	0	0	0
2014-02-26	1.76	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0		0
2014-03-01	0	0	0	0
2014-03-02	4.44	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	6.67	0	0	0

Cuadro 3.7: Algoritmo 1 con Análisis de Sentimiento

Date	**government**	**Egyptian**	**Pope**	**finances**
2014-02-24	8.88	10	5	5
2014-02-25	-5	0	0	0
2014-02-26	-3.33	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0	0	0
2014-03-01	0	0	0	0
2014-03-02	-5	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	10	0	0	0

Cuadro 3.8: Algoritmo 2 con Análisis de Sentimiento

Date	**government**	**Egyptian**	**Pope**	**finances**
2014-02-24	8	10	9	9
2014-02-25	-9	0	0	0
2014-02-26	-10	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0	0	0
2014-03-01	0	0	0	0
2014-03-02	-9	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	10	0	0	0

Cuadro 3.9: Algoritmo 3 con Análisis de Sentimiento

Date	**government**	**Egyptian**	**Pope**	**finances**
2014-02-24	6.41	6.67	4.44	4.44
2014-02-25	-4.44	0	0	0
2014-02-26	-1.76	0	0	0
2014-02-27	0	0	0	0
2014-02-28	0	0	0	0
2014-03-01	0	0	0	0
2014-03-02	-4.44	0	0	0
2014-03-03	0	0	0	0
2014-03-04	0	0	0	0
2014-03-05	6.67	0	0	0

Ya tenemos los distintos conjuntos de datos que suministraremos a los algoritmos de aprendizaje automático. Los subiremos a la base de datos SQL que hemos preparado para luego poder reutilizar los datos.

3.6. Aprendizaje automático

El campo del *Machine Learning* tiene muchas ramas pero, en general, los algoritmos se clasifican en dos: algoritmos supervisados y no supervisados. Los algoritmos supervisados son aquellos en los cuales le presentamos a la máquina unas variables, o *features*, y también introducimos unos 'resultados', llamados *labels*. A partir de ahí, la máquina va aprendiendo sobre los datos hasta ser capaz de poder emitir predicciones propias. Claramente, nuestro proyecto será del tipo supervisado. Dentro de los algoritmos supervisados, nos encontramos con algoritmos de clasificación y algoritmos de regresión.

- **Algoritmos de clasificación**. Sirven para *clasificar*. Por ejemplo, poder distinguir tipos de animales según sus características físicas: altura, peso, color de la piel...
- **Algoritmos de regresión**. Sirven para predecir los siguientes valores numéricos de una progresión. Por ejemplo, si ayer hizo 30 grados y antes de ayer 27, ¿cuánto hará hoy?

En las primeras etapas de este Trabajo Fin de Grado, pensamos en emplear algoritmos de regresión. Ya que los índices fluctúan en valores numéricos, parecía la opción más lógica. No obstante, tras las primeras pruebas nos dimos cuenta que es demasiado optimista para el tipo de datos que estábamos analizando. El mercado bursátil no es una función predecible, los resultados pasados no determinan los resultados futuros. Por ello, utilizar regresión no es la opción más óptima y nos decantamos por utilizar algoritmos de clasificación, para clasificar los valores de un índice en tres categorías:

- **Categoría 0**. El precio de cierre de un índice se encuentra entre la media de las subidas y la media de las bajadas. Es decir, se considera una fluctuación normal. Ni ha subido más de lo normal, ni ha bajado más de lo normal. A nivel de rentabilidad, es la opción menos interesante, pues no nos asegura beneficios.
- **Categoría 1**. El precio de cierre de un índice es superior a la media de subidas. Naturalmente, el objetivo al invertir en el mercado de valores es conseguir una rentabilidad, por lo tanto estamos buscando subidas que nos generen dicha rentabilidad.
- **Categoría 2**. El precio de cierre de un índice es inferior a la media de las bajadas. Lo cierto es que las inversiones no solo tienen que hacerse con vistas a que el precio suba. También hay opciones de venta, mediante la cual la inversión resulta rentable aún cuando el precio baja, por lo que es interesante crear una categoría para este tipo de variaciones.

El algoritmo recibirá entonces el conjunto de datos correspondiente y la categoría en la que se encuentra el precio para el intervalo de tiempo correspondiente. Aprenderá sobre el conjunto de entrenamiento (80 % del conjunto total) y después emitirá una predicción sobre el 20 % restante, que se conoce como conjunto de prueba. Como hemos visto en las tablas anteriores, intentará establecer una relación entre la *fuerza* de cada concepto y cómo reacciona luego el mercado y clasificará, según su criterio, en una de las tres categorías mencionadas anteriormente.

Como se trata de un proyecto de investigación, vamos a poner a prueba los datos con tres algoritmos de clasificación distintos, muy conocidos en el campo del aprendizaje automático, para ver cuál se adapta mejor a lo que queremos conseguir. Estos son:

- **DT** [16]. Árboles de decisión. En informática, el aprendizaje mediante árboles de decisión utiliza un árbol de decisión (como modelo predictivo) para pasar de observaciones sobre un elemento (representado en las ramas) a conclusiones sobre el valor objetivo del elemento (representado en las hojas). Es uno de los enfoques de modelado predictivo utilizados en estadística, minería de datos y aprendizaje automático.
- **SVM** [17]. Máquinas de vector soporte. En el aprendizaje automático, las máquinas vectoriales de apoyo (SVM, también llamadas máquinas de vector soporte) son modelos de aprendizaje supervisados con algoritmos de aprendizaje asociados que analizan los datos utilizados para la clasificación y el análisis de regresión. Dado un conjunto de ejemplos de entrenamiento, cada uno marcado como perteneciente a una u otra de las dos categorías, un algoritmo de entrenamiento SVM construye un modelo que asigna nuevos ejemplos a una u otra categoría, convirtiéndolo en un clasificador lineal binario no probabilístico.
- **KNN** [18]. k-vecinos más próximos. En el reconocimiento de patrones, el algoritmo k-nearest neighbors (KNN) es un método no paramétrico utilizado para la clasificación y regresión. En ambos casos, la entrada consiste en los ejemplos de entrenamiento k más cercanos en el espacio de características. La salida depende de si se utiliza k-NN para la clasificación o la regresión: en la clasificación k-NN, el resultado es una pertenencia a una clase. Un objeto es clasificado por un voto de pluralidad de sus vecinos, siendo el objeto asignado a la clase más común entre sus vecinos más cercanos (k es un entero positivo, típicamente pequeño).

Una vez preparados los algoritmos y generados los conjuntos de datos, el desarrollo de la aplicación ha finalizado y solo queda comenzar el aprendizaje y extraer conclusiones. Como hemos volcado todos los datos a una base de datos SQL, ahora es más sencillo obtener máximos, mínimos, promedios y otras estadísticas de interés mediante unas simples consultas.

Capítulo 4

Análisis de resultados

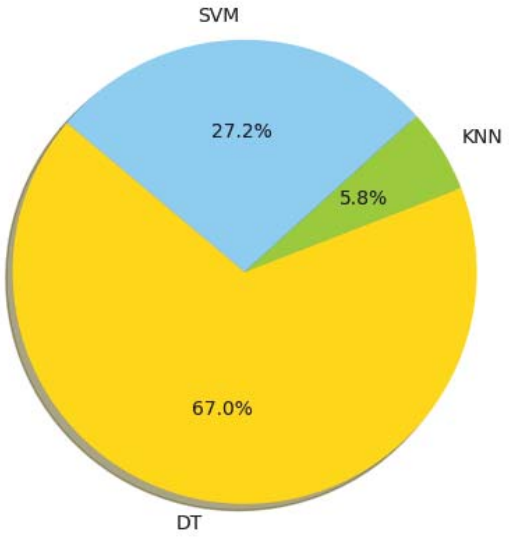
El resultado final han sido 28.331 *tests*. Hemos incluido una gran cantidad de variables para intentar averiguar qué configuración funciona mejor. El listado de variables analizadas ha sido el siguiente:

- **ml_algo**. Es el algoritmo de *Machine Learning* utilizado. Hay 3, como ya hemos mencionado: KNN, DT y SVM.
- **algo_t**. Es el algoritmo de ponderación utilizado. Hay 6 (los tres algoritmos y sus derivados con análisis de sentimiento).
- **top_x**. Es el número de conceptos a tener en cuenta para el estudio. En total, nuestro conjunto de datos había recopilado un poco más de 20.000 conceptos. Naturalmente, es una cantidad ingente de la cual no se pueden extraer conclusiones y que únicamente va a confundir a nuestro algoritmo de aprendizaje. Es lo que se conoce como *ruido* en el campo del aprendizaje automático. De esta manera, nos limitados a coger los conceptos más mencionados. Para no elegir un número de forma aleatoria, nos decantamos por probar varios, empezando por los cinco conceptos más mencionados y subiendo hasta mil.
- **index_offset**. Es la comparación del día de la noticia y cuándo verificamos los resultados en los mercados. Puede ser el mismo día (si la noticia sale el 26/4, se comprueba la variación en el precio para el 26/4) o para el día siguiente (si la noticia sale el 26/4, se comprueba la variación en el precio del 27/4). Cada uno tiene sus ventajas y desventajas. Comprobación diaria aporta más inmediatez, mientras que la comprobación al día siguiente nos da tiempo para que el efecto completo se haya notado en los mercados. Se probarán los dos y se verificará cuál funciona mejor.

Para averiguar cuál es la configuración que mejor ha funcionado, vamos a hacer un estudio caso por caso y anotar el ganador, para cada una de las variables. De

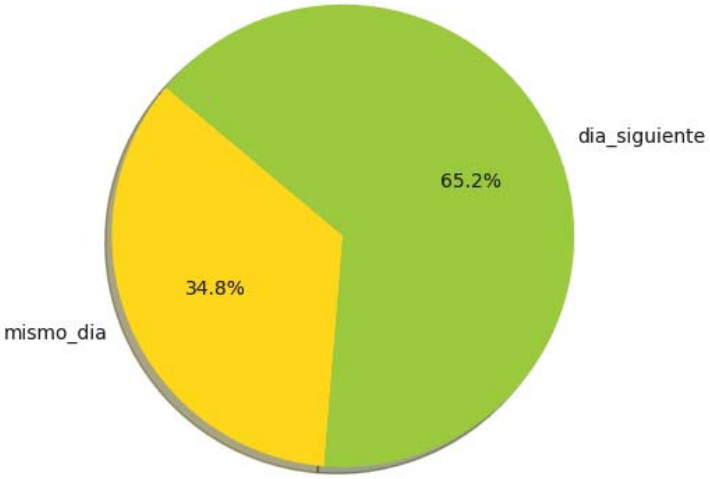
esta manera podremos ver que, por ejemplo, el algoritmo DT ha obtenido mayor precisión casi el 70 % de las comparaciones. Vamos a utilizar gráficos de sectores para visualizar mejor los resultados:

Figura 4.1: Comparativa de algoritmos de ML



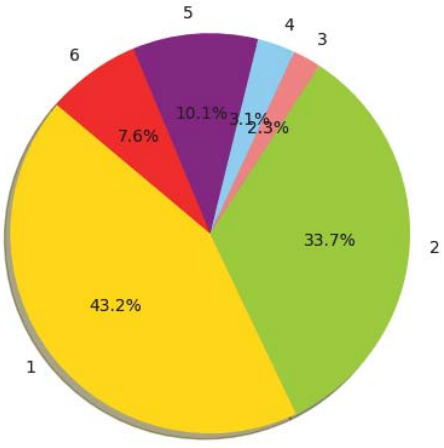
Como podemos ver, el algoritmo DT ha ganado el 67 % de las comparaciones. Un resultado muy superior a los demás, en comparación con el 27 % de SVM y apenas 5 % de KNN.

Figura 4.2: Comparativa de index_offset



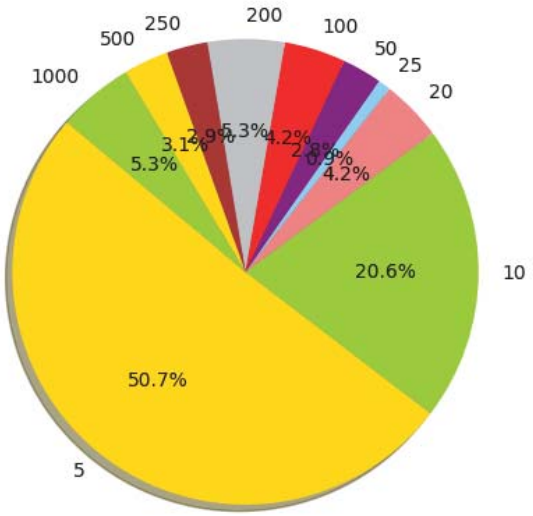
A continuación tenemos la comparativa de **index_offset**. Como podemos ver, se ha obtenido mayor precisión cuando comparamos las noticias de un día con el precio de cierre de mercado del día siguiente.

Figura 4.3: Comparativa de algoritmos de posición



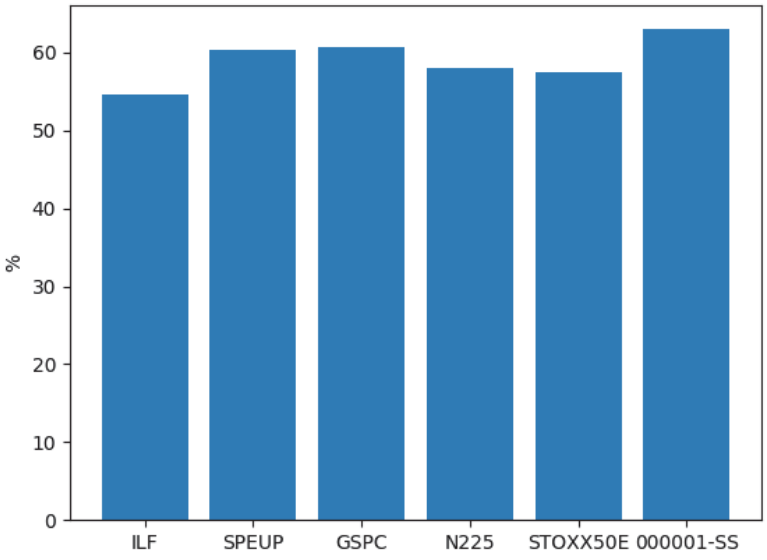
Resultado muy interesante la comparativa de algoritmos de posición. Como vemos, los algoritmos 1 y 2 son los claros ganadores. Lo verdaderamente sorprendente es el resultado de los algoritmos *espejo* en los que habíamos incluido análisis de sentimiento, pues suponíamos que incluir ese nivel de detalle resultaría en mejores resultados. Es el caso de los algoritmos 4, 5 y 6 que apenas resultan más certeros, en conjunto, un 13 % de las veces.

Figura 4.4: Comparativa de top_x



Otro gráfico muy curioso. Como podemos ver, el algoritmo *empeora* su precisión a medida que estudia más conceptos. Es decir, estudiando apenas cinco conceptos (Por ejemplo, Estados Unidos, Rusia, China, Europa y Trump) obtiene mejores resultados la mitad de las veces. Si estudia diez conceptos baja drásticamente al 20 %.

Figura 4.5: Porcentaje de precisión promedio para cada índice



Por último, el porcentaje de precisión promedio para cada índice. Sorprendente resultado del índice chino, por encima del resto con bastante diferencia. En el siguiente nivel se encuentran el europeo, americano y japonés, más o menos similares. En último lugar, el latinoamericano, lo cual era de esperar, porque las noticias más importantes concernientes a esta región no suelen ser muy frecuentes. La parte positiva es que todos superan el 50 % de precisión, lo cual era uno de los objetivos del proyecto.

No obstante, estos gráficos pueden mostrar resultados erróneos, debido a que se está comparando el acierto global y no el acierto por categorías, lo cual explicaremos en los siguientes párrafos para ver por qué es importante. A continuación, vamos a listar otras estadísticas relevantes:

- Mayor porcentaje de precisión: 69.7 % {**Test**: 28.286, **Índice**: 000001-SS, **Algoritmo**: DT}
- Menor porcentaje de precisión: 37.2 % {**Test**: 10.363, **Índice**: ILF, **Algoritmo**: KNN}
- Porcentaje medio de precisión: 58.3 %
- Porcentaje medio de precisión para el algoritmo KNN: 57.2 %
- Porcentaje medio de precisión para el algoritmo SVM: 57.1 %

- Porcentaje medio de precisión para el algoritmo DT: 60.5 %

Como comentamos anteriormente, uno de los grandes problemas eran las horas de cierre de mercado y la hora a la que sale la noticia. Por eso, comparamos los índice de dos maneras: resultados del día versus resultados del día siguiente. Es decir, si la noticia es del día 26/04/2018, lo estudiamos para el precio de cierre de mercado del día 26/04/2018 (en el caso del mismo día) o con el precio de cierre de mercado del día siguiente, 27/04/2018. Esta es la variable **index_offset** que hemos mencionado. Otras estadísticas interesantes:

- Mayor porcentaje de precisión para comparación al mismo día: 66.80 %
- Mayor porcentaje de precisión para comparación al día siguiente: 69.70 %
- Porcentaje medio de precisión para comparación al mismo día: 57.89 %
- Porcentaje medio de precisión para comparación al día siguiente: 59.11 %

A priori, parecen datos esperanzadores. Poder predecir correctamente casi el 70 % de las variaciones bursátiles es algo soñado por muchos *brokers* y empresas del sector. Desgranaremos aún más los datos para ver si efectivamente hemos encontrado la máquina de hacer dinero (o no). Por otro lado, de todos los índices, ILF debería ser el más difícil de predecir. Las potencias mundiales son Estados Unidos, Asia y Europa. Naturalmente, las noticias más relevantes mundialmente concernirán estas regiones, por lo que no habrá muchos casos de estudio para poder aprender bien sobre qué funciona en el mercado latinoamericano. De momento, los primeros resultados son interesantes.

Por otro lado, estos porcentajes se refieren a la precisión total, no por categorías. Veamos a continuación qué tal predice cada una de las categorías:

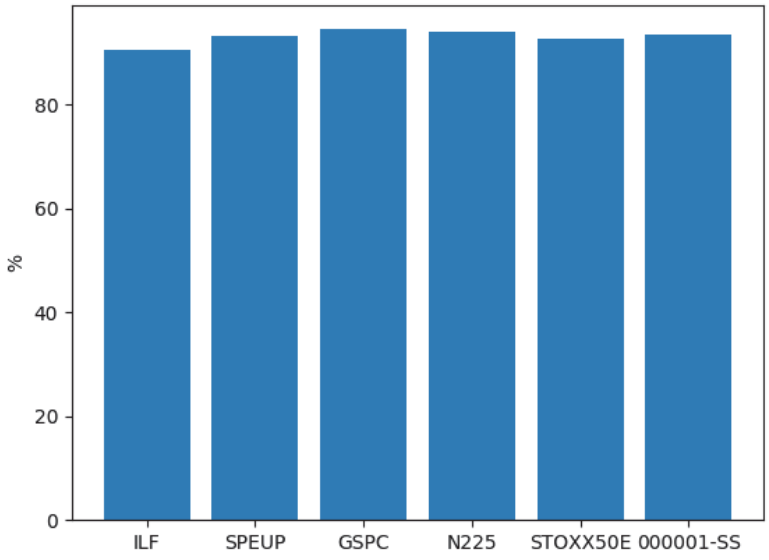
- Mayor porcentaje de precisión para la categoría 0: 100 % {**Test**: 1, **Índice**: ILF, **Algoritmo**: DT}
- Mayor porcentaje de precisión para la categoría 1: 42.85 % {**Test**: 10.243, **Índice**: ILF, **Algoritmo**: KNN}
- Mayor porcentaje de precisión para la categoría 2: 38.33 % {**Test**: 747, **Índice**: ILF, **Algoritmo**: KNN}
- Porcentaje medio de precisión para la categoría 0: 92.95 %
- Porcentaje medio de precisión para la categoría 1: 4.27 %
- Porcentaje medio de precisión para la categoría 2: 3.57 %

Los valores máximos superan el puro azar (33.33 %), pero los porcentajes medios de precisión son desoladores para los casos 'de dinero'. Las inversiones necesitan volatilidad para generar un retorno y nuestros algoritmos no parecen ser consistentes en la predicción, con porcentajes medios entre el 3 y el 5 por ciento para este tipo de casos. A continuación vamos a adentrarnos en un análisis más profundo, estructurado por categorías.

4.1. Categoría 0

Desde un punto de vista financiero, la categoría 0 es la menos interesante. No nos asegura obtener una rentabilidad a nuestra inversión, por lo que es de poca importancia. No obstante, vamos a comprobar las principales estadísticas y resultados.

Figura 4.6: Porcentaje de precisión promedio para cada índice



Si bien a primera vista esta gráfica podría ser un gran indicador de éxito, con más de 90 % de precisión, plantea una problemática: Si la precisión media, de todos los casos, era de un 50-60 %, y la precisión media de la categoría 0, es decir, de la fluctuación normal, es del noventa... significa que en el resto de categorías nos encontraremos gráficas y precisiones muy bajas.

Lo más probable es que, por defecto, el algoritmo de aprendizaje recurra a la categoría 0 y no sabe distinguir correctamente entre una categoría y el resto. Esto es

debido a que los índices tienen fluctuaciones normales con mayor frecuencia. De ahí, *normales*. Es mucho más corriente una fluctuación de este tipo que drásticas subidas o bajadas. Hay muchos más casos de categoría 0 que casos de categorías 1 o 2 en todos los índices, como podemos comprobar en esta tabla:

Cuadro 4.1: Distribución por categorías

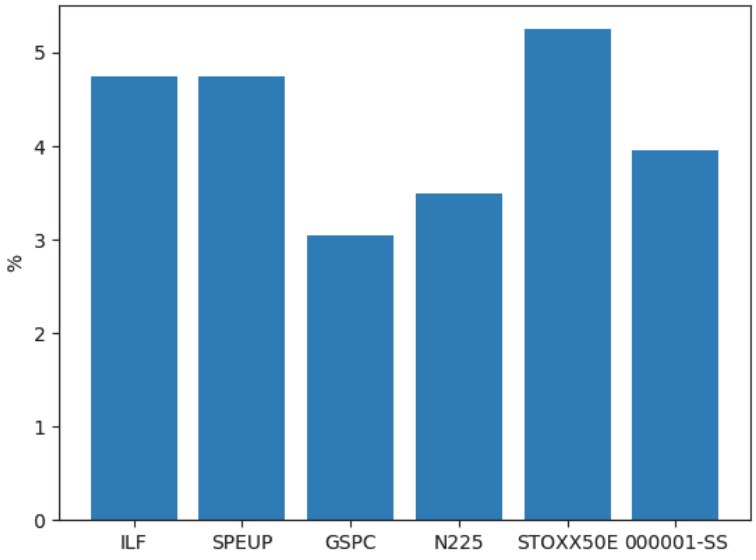
Índice	**Categoría 0**	**Categoría 1**	**Categoría 2**
SPEUP	790	247	222
000001-SS	820	248	153
STOXX50E	753	239	229
N225	754	254	193
GSPC	797	224	200
ILF	720	240	261

Efectivamente, la categoría 0 ¡llega incluso a duplicar a los casos de categorías 1 y 2 en conjunto!

4.2. Categoría 1

Seguramente la categoría 1 es la más interesante de todas. Representa todas las subidas fuera de lo normal. Si bien también se puede sacar beneficio de la categoría 2 (mediante la venta) es más engorroso y la operación más sencilla de realizar es, sin duda alguna, la de compra un valor y esperar a que su precio suba para, posteriormente, vender el valor y de la diferencia sacar nuestro beneficio.

Figura 4.7: Porcentaje de precisión promedio para cada índice



Como ya anticipábamos antes, los resultados son muy malos. Apenas 3-5 % de precisión es un valor pésimo. Por otro lado, veamos los mejores resultados obtenidos para esta categoría:

Cuadro 4.2: Máximas Precisiones - Categoría 1

Índice	**Máxima Precisión**
ILF	42.85 %
STOXX50E	42.30 %
GSPC	35.55 %
SPEUP	35.29 %
000001-SS	30.95 %
N225	28.57 %

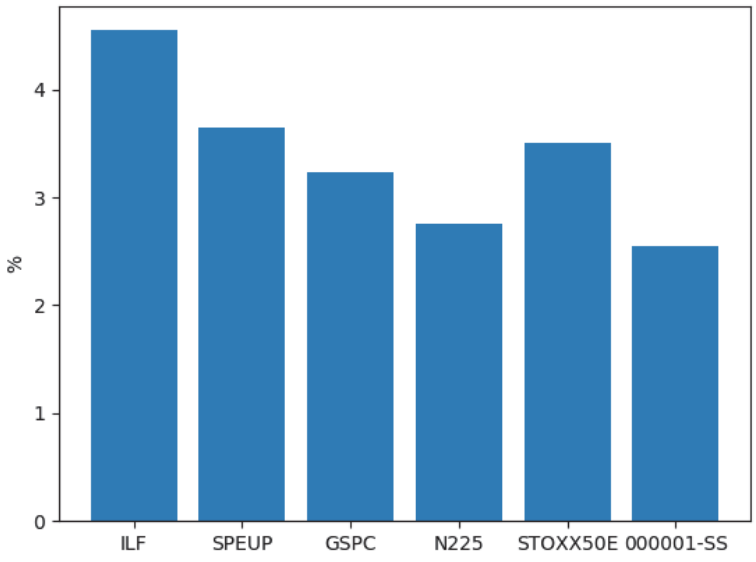
Resultados más esperanzadores. Si bien no somos capaces de predecir cuándo un valor va a subir con exactitud, porcentajes de entre 25 %-45 % son bastante buenos, teniendo en cuenta además que, según la rentabilidad, podría incluso llegar a justificarse si la inversión se predice correctamente una de cada dos o tres veces.

No acertar una predicción tampoco implica irse al otro extremo (categoría 2), por lo que se pueden tener tres recomendaciones de inversión y que dos resulten ser categoría 0 y una categoría 1, que es al final la que más rentabilidad tiene. Como la otra categoría, a priori, no resta, la inversión global sería positiva.

4.3. Categoría 2

Por último, la categoría 2. Esta representa bajadas fuera de lo normal. Hay ciertas operaciones bursátiles que nos permiten sacar rentabilidad de este tipo de operaciones, es decir, comprar a un precio y ganar con la diferencia entre el precio final y el inicial, siempre que vendamos con un precio final más bajo que el inicial. De manera análoga, perdemos cuando el precio sube. Los resultados son:

Figura 4.8: Porcentaje de precisión promedio para cada índice



Efectivamente, sigue la misma línea que la categoría 1. No hemos sido capaces de establecer una relación clara y concisa entre las categorías y las noticias.

Cuadro 4.3: Máximas Precisiones - Categoría 2

Índice	**Máxima Precisión**
ILF	38.33 %
GSPC	35.13 %
STOXX50E	29.54 %
SPEUP	29.26 %
N225	26.08 %
000001-SS	25 %

Por otro lado, como en la categoría 1, los resultados máximos son reveladores. Aunque todos los índices pierden precisión para este caso, se mantiene más o menos el orden. El mercado latinoamericano se destaca como el claro ganador, mientras que los asiáticos son perdedores, en ambos. Los mercados americano y europeo se asientan entre el 30 % y el 35 % de acierto.

4.4. Caso de Estudio

Para finalizar, el siguiente paso es ver cómo se comportaría la máquina en el mundo real. Con los datos de entrenamiento de los últimos cinco años, reutilizamos lo aprendido para realizar predicciones sobre el intervalo del tiempo durante el cual se desarrolló el TFG. Es decir, entre el 25 de Febrero de 2019 y el 25 de Mayo de 2019. Para este caso de estudio, nos centramos en el índice americano, *S&P 500*, que durante la fase de aprendizaje había obtenido un 35 % de precisión máxima. Este índice funcionó mejor con el algoritmo de aprendizaje automático SVM, por lo que replicamos la configuración del mismo y lo exportamos para poder ser utilizado en futuras predicciones.

Después, preparamos el conjunto de datos para este intervalo de tiempo y por último realizamos la predicción. Nos centramos únicamente en las predicciones de categoría 1, es decir, comprar y esperar al día siguiente para vender, confiando en una subida de los mercados. Para cada inversión, compramos 10 unidades del índice al precio de cierre de ese día y vendimos el lote al precio de cierre del día siguiente. Mencionar que no se han tenido en cuenta las recomendaciones de inversión que caían en fin de semana, pues los mercados estaban cerrados. Estos fueron los resultados:

Cuadro 4.4: Máximas Precisiones - Caso 0

Fecha	Importe Compra	Importa Venta	Beneficio	Rentabilidad
08-03-2019	27.430,70 €	27.833,00 €	402,30 €	1,47 %
12-03-2019	27.915,20 €	28.109,20 €	194,00 €	0,69 %
14-03-2019	28.084,80 €	28.224,80 €	140,00 €	0,50 %
21-03-2019	28.548,80 €	28.007,10 €	-541,70 €	-1,90 %
25-03-2019	27.983,60 €	28.184,60 €	201,00 €	0,72 %
27-03-2019	28.053,70 €	28.154,40 €	100,70 €	0,36 %
05-04-2019	28.927,40 €	28.957,70 €	30,30 €	0,10 %
08-04-2019	28.957,70 €	28.782,00 €	-175,70 €	-0,61 %
09-04-2019	28.782,00 €	28.882,10 €	100,10 €	0,35 %
16-04-2019	29.070,60 €	29.004,50 €	-66,10 €	-0,23 %
17-04-2019	29.004,50 €	29.050,30 €	45,80 €	0,16 %
25-04-2019	29.261,70 €	29.398,80 €	137,10 €	0,47 %
01-05-2019	29.237,30 €	29.175,20 €	-62,10 €	-0,21 %
Total	371.258,00 €	371.763,70 €	505,70 €	0,14 %

Un total de 13 inversiones, 9 cerradas con beneficios y 4 con pérdidas. Si bien hay más operaciones exitosas, la rentabilidad se queda en un 0.14 %. La parte positiva es que si el algoritmo no detecta correctamente una subida de categoría 1, no implica una bajada de categoría 2, sino que en la mayoría de casos, acaba en la categoría de fluctuaciones normales y se reduce bastante el riesgo. En este sentido, hubo trece días de categoría 1, de los cuales el algoritmo predijo correctamente 3. Es decir, un 23 % de acierto. Aún así, en estas 13 inversiones el resultado final sigue siendo positivo, por lo mencionado anteriormente.

Sin duda alguna, es una aplicación real de nuestro algoritmo que, a falta de más casos de estudio, ha podido pasar sin pérdidas, cumpliendo así el objetivo del Trabajo.

Capítulo 5

Conclusión

En conclusión, el proyecto es capaz de predecir correctamente en torno a un 30 % de las variaciones notables (subidas o bajadas), dependiendo del índice. Hemos cumplido todos los los objetivos planteados en el capítulo 1 y aunque no tengamos una fiabilidad del 100 %, son resultados aceptables. Como hemos visto en el caso de estudio anterior, no perdemos dinero. Veamos ahora los mejores resultados para cada uno de los índices:

Cuadro 5.1: Precisión Máxima por Índice

Índice	**C0**	**C1**	**C2**
ILF	100 %	42.85 %	38.33 %
GSPC	100 %	35.55 %	35.13 %
SPEUP	100 %	35.29 %	29.26 %
N225	100 %	28.57 %	26 %
STOXX50E	100 %	42.30 %	29.54 %
SHA: 000001	100 %	30.95 %	25 %

Como vemos, resultados mejorables pero que se pueden utilizar ya en el mundo real para conseguir una rentabilidad en los mercados. Por último, vamos a ver los resultados finales de precisión, pero en vez de por índice, por algoritmo de aprendizaje automático (KNN, DT y SVM):

Cuadro 5.2: Precisión Máxima por Índice

Algoritmo	**C0**	**C1**	**C2**
SVM	100 %	37.5 %	35.13 %
DT	100 %	9.52 %	7.5 %
KNN	100 %	42.85 %	38.33 %

Como vemos, el algoritmo de árboles de decisión es el que peor se adapta a nuestro planteamiento, siendo los algoritmos SVM y KNN los que mejores resultados obtienen. Para maximizar aún más la precisión y la rentabilidad, vamos a detallar en el siguiente apartado posibles mejoras que se pueden implementar.

5.1. Posibles mejoras

Por último, vamos a compartir una serie de mejoras que, a nuestro modo de ver, mejorarán notablemente los resultados obtenidos:

- **Más antigüedad**. En mi opinión, el problema principal ha sido la falta de datos históricos. Lamentablemente, de manera gratuita únicamente están disponibles los últimos 5 años. Entre fines de semana y días festivos, esto se reduce a uno 1000-1300 casos de estudio (cada día es un caso de estudio). Números muy pequeños para extraer conclusiones. Precisar más (precios intradiarios, por ejemplo) y tener más profundidad es fundamental para que la máquina pueda aprender correctamente.
- **Resultados al minuto**. Actualmente, estamos estudiando la correlación entre una serie de noticias y el impacto que tienen en los mercados a nivel diario (24 horas) e incluso más, al día siguiente (48 horas). Estamos perdiendo mucha precisión. Por ejemplo, si una noticia sale por la mañana del día 23, nosotros estamos estableciendo la relación entre el impacto de esa noticia y el precio de cierre de ese día 23 (o el 24, dependiendo de la configuración). Entre que ha salido la noticia y se ha cerrado el mercado, pueden haber salido otras noticias de menor importancia, pero que han suavizado/amplificado el efecto original. Con los recursos disponibles, plantear un análisis tan minucioso resulta una ardua tarea. Pero sin duda estamos convencidos de que se mejorará la precisión.
- **Mayor cantidad de indicadores bursátiles**. Estamos suponiendo que los índices son un buen indicador de la situación bursátil de las distintas regiones del mundo. Y mientras esto es relativamente cierto, un índice no deja de ser

un conjunto de valores, de empresas, que varían su cotización por muchos motivos. Por ejemplo, si el Brexit sale adelante, las empresas europeas que tengan grandes acuerdos comerciales con Reino Unido se verán mucho más afectadas que aquellas cuyo negocio no guarda relaciones con esta región. Por lo tanto, utilizar más indicadores bursátiles es una mejora que puede ayudar a la comprensión del problema planteado y, por tanto, a la precisión.

- **Mejorar el análisis de sentimiento**. Lo ideal sería preparar un algoritmo de análisis de sentimiento propio, que fuese entrenado con noticias reales, para ver cuándo una noticia va a tener un impacto positivo o negativo. No es lo mismo que una oración sea subjetivamente positiva a que luego los mercados se vayan a comportar de forma positiva. Como ya hemos explicado, los mercados evalúan si se está generando dinero. Teniendo este enfoque en mente, se puede desarrollar una librería que comprenda mejor los titulares.

- **Utilizar el factor salience**. En la librería NLP de Google, a parte de toda la información de entidades, relación de conceptos y análisis de sentimiento que nos aporta, también nos ofrece un factor, llamado *salience*. Este factor mide cuán relevante es un concepto en una frase. Naturalmente, no es igual de importante decir, "Trump amenaza con romper lazos con Rusia", que decir "La ONU prepara una. estrategia para abarcar la situación en Venezuela de manera pacífica junto a los líderes Macron, Merkel y Trump". En el primer caso, Trump se sitúa en el epicentro de la noticia. En el segundo, queda relegado a un segundo plano.

Creo fehacientemente que esta investigación aún tiene mucho recorrido. Ha sido un orgullo poder llevarla acabo y animo a cualquier interesado a continuar el trabajo aquí presentado. Se puede contactar conmigo a través del correo electrónico: manuel.rodriguez.rodriguez@alumnos.upm.es

Anexos: Código fuente

main.py

Fichero Python que importa los conjuntos de datos generados, prepara los datos bursátiles, los une y comienza el proceso de aprendizaje.

```
import six
import sqlite3
import pandas as pd
import numpy as np

from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

from google.cloud import language
from google.cloud.language import enums
from google.cloud.language import types

from datetime import datetime, timedelta

conn = sqlite3.connect('tfg.db')
c = conn.cursor()

client = language.LanguageServiceClient()

'''
Indexes:
- ILF:
- SPEUP:
- GSPC:
- N225:
- STOXX50E:
```

```
- 000001-SS

We've got 6 indexes, from all over the world. We're going to test
    ↪ our classifier against all of them
to find out which algorithm performs best against each index.
'''

indexes = [
'ILF',
'SPEUP',
'GSPC',
'N225',
'STOXX50E',
'000001-SS']

'''
Algorithm representation:
algo_ALGO_K_EXP.pkl -> (algo_t, k, exp)
'''

algorithms = [
(1, 0, 0),
(2, 0, 0),
(5, 0, 0),
(6, 0, 0)
]

for y in range(1, 6): #Algo 3 WITHOUT sentiment analysis
    for z in range(100, 200, 25):
        algorithms.append((3, y, z/100))
for y in range(1, 6): #Algo 4 WITH sentiment analysis
    for z in range(100, 200, 25):
        algorithms.append((4, y, z/100))

'''
We will use 3 multiclass classification algorithms using sci-kit
    ↪ learn. They are:
- DT: Decision Tree
- SVM: Support Vector Machines
- KNN: K-Nearest Neighbours
'''

ml_algos = [
'DT',
```

```
'SVM',
'KNN']

index_offset = [
'next_day',
'same_day']

def DT(X, y):
        X_train, X_test, y_train, y_test = train_test_split(X, y,
            ↪ random_state = 0, test_size = 0.2)
        dtree = DecisionTreeClassifier(max_depth = 2).fit(X_train,
            ↪ y_train)
        dt_pred = dtree.predict(X_test)
        accuracy = dtree.score(X_test, y_test)
        cm = confusion_matrix(y_test, dt_pred)
        return accuracy, [cm[i][i]/cm[i].sum() for i in range(0, 3)]

def SVM(X, y):
        X_train, X_test, y_train, y_test = train_test_split(X, y,
            ↪ random_state = 0, test_size = 0.2)
        svm_model = SVC(kernel = 'linear', C = 1).fit(X_train,
            ↪ y_train)
        svm_pred = svm_model.predict(X_test)
        accuracy = svm_model.score(X_test, y_test)
        cm = confusion_matrix(y_test, svm_pred)
        return accuracy, [cm[i][i]/cm[i].sum() for i in range(0, 3)]

def KNN(X, y):
        X_train, X_test, y_train, y_test = train_test_split(X, y,
            ↪ random_state = 0, test_size = 0.2)
        knn_model = KNeighborsClassifier(n_neighbors = 7).fit(X_train
            ↪ , y_train)
        accuracy = knn_model.score(X_test, y_test)
        knn_pred = knn_model.predict(X_test)
        cm = confusion_matrix(y_test, knn_pred)
        return accuracy, [cm[i][i]/cm[i].sum() for i in range(0, 3)]

def prepare_index(index, offset):
        query = "SELECT * FROM indexes WHERE `index` = '" + index + "
            ↪ ';"
        df = pd.read_sql_query(query, conn)
        df.set_index('date', inplace = True)
        for to_delete in ['id', 'index', 'open', 'close', 'abs_var']:
                df.drop(to_delete, axis = 1, inplace = True)
```

```
        df['rel_var'] = df['rel_var'].replace(',', '.', regex = True)
        df = df[pd.to_numeric(df['rel_var'], errors = 'coerce').
            ↪ notnull()]
        df = df.apply(pd.to_numeric)
        positive_threshold = df[df['rel_var'] > 0].mean(axis = 0)
        negative_threshold = df[df['rel_var'] < 0].mean(axis = 0)
        if offset == "same_day":
                df['rel_var'] = df.apply(lambda x : (1 if x['rel_var']
                    ↪  > positive_threshold['rel_var'] else (2 if x['
                    ↪ rel_var'] < negative_threshold['rel_var'] else
                    ↪ 0)), axis = 1)
        elif offset == "next_day":
                for i, row in df.iterrows():
                        shift_df = df.shift(-1)
                        next_day = shift_df.at[i, 'rel_var']
                        if next_day > positive_threshold['rel_var']:
                                row['rel_var'] = 1
                        elif next_day < negative_threshold['rel_var']:
                                row['rel_var'] = 2
                        else:
                                row['rel_var'] = 0
        df.columns = ['variation_category']
        return df

def verbose(index, ml_algo, algo_t, k, exp, top_xx, accuracy,
    ↪ c_accuracy, offset):
        print('=' * 20)
        print(u'{:<16}: {}'.format('index', index))
        print(u'{:<16}: {}'.format('ml_algo', ml_algo))
        print(u'{:<16}: {}'.format('algo_t', algo_t))
        print(u'{:<16}: {}'.format('k', k))
        print(u'{:<16}: {}'.format('exp', exp))
        print(u'{:<16}: {}'.format('top_xx', top_xx))
        print(u'{:<16}: {}'.format('accuracy', accuracy))
        print(u'{:<16}: {}'.format('c0_accuracy', c_accuracy[0]))
        print(u'{:<16}: {}'.format('c1_accuracy', c_accuracy[1]))
        print(u'{:<16}: {}'.format('c2_accuracy', c_accuracy[2]))
        print(u'{:<16}: {}'.format('index_offset', offset))
        to_sql((index, ml_algo, algo_t, k, exp, top_xx, accuracy,
            ↪ c_accuracy[0], c_accuracy[1], c_accuracy[2], offset))

def to_sql(param):
        global c
        global conn
```

```
        c.execute("INSERT INTO results (‘index‘, ml_algo, algo_t, k,
            ↪ exp, top_x, accuracy, c0_accuracy, c1_accuracy,
            ↪ c2_accuracy, index_offset) VALUES (?, ?, ?, ?, ?, ?, ?,
            ↪  ?, ?, ?, ?)", param)
        conn.commit()
        print("SQL Success!")


def top_x(df_copy):
        top_xs = [5, 10, 20, 25, 50, 100, 200, 250, 500, 1000]
        resulting_dfs = []
        for top_xx in top_xs:
                df = df_copy.copy()
                df = df.groupby(df.columns, axis = 1).sum()
                df_top = df.sum(axis = 0)
                df_top.sort_values(inplace = True, ascending = False)
                top_df = df_top.head(top_xx)
                left = [item for item in df.columns.tolist() if item
                    ↪ not in top_df.index.tolist()]
                df.drop(columns = left, inplace = True)
                resulting_dfs.append(df)
        return resulting_dfs

def cluster(df_copy, relationships):
        df = df_copy.copy()
        for relationship in relationships:
                anchor = str(relationship[0])
                for related in relationship:
                        if related == anchor or anchor not in df.
                            ↪ columns.tolist() or related not in df.
                            ↪ columns.tolist():
                                continue
                        else:
                                df[anchor] = df[anchor] + df[related]
                                df.drop(related, axis = 1, inplace =
                                    ↪ True)
        return df

def relationships(df_copy):
        df = df_copy.copy()
        df = df.reindex(df.mean().sort_values().index, axis = 1)
        relationships = []
        columns = df[df.columns[-1000:]].columns.tolist()
        text = ", ".join(columns)
```

```
        if isinstance(text, six.binary_type):
                text = text.decode('utf-8')
        document = types.Document(content=text, type=enums.Document.
            ↪ Type.PLAIN_TEXT)
        entities = client.analyze_entities(document).entities
        for entity in entities:
                relation = list(dict.fromkeys([mention.text.content
                    ↪ for mention in entity.mentions]))
                relation.sort(key = lambda x : len(x))
                relationships.append(relation)
        return relationships

for index in indexes:
        for algorithm in algorithms:
                algo_t = algorithm[0]
                k = algorithm[1]
                exp = algorithm [2]
                df = pd.read_pickle("algo_" + str(algo_t) + "_" + str(
                    ↪ k) + "_" + str(exp) + ".pkl")
                df = df.astype(float)
                top_xs = top_x(cluster(df, relationships(df)))
                topxs = [5, 10, 20, 25, 50, 100, 200, 250, 500, 1000]
                for offset in index_offset:
                        index_df = prepare_index(index, offset)
                        for count, top_xx in enumerate(top_xs):
                                df = top_xx.join(index_df)
                                df.dropna(inplace = True)
                                df['variation_category'] = df['
                                    ↪ variation_category'].apply(np.
                                    ↪ int64)
                                X = np.array(df.drop(['
                                    ↪ variation_category'], axis = 1))
                                y = np.array(df['variation_category'])
                                for ml_algo in ml_algos:
                                        if ml_algo == "DT":
                                                accuracy, c_accuracy = DT
                                                    ↪ (X, y)
                                        elif ml_algo == "SVM":
                                                accuracy, c_accuracy =
                                                    ↪ SVM(X, y)
                                        elif ml_algo == "KNN":
                                                accuracy, c_accuracy =
                                                    ↪ KNN(X, y)
                                        verbose(index, ml_algo, algo_t, k
```

```
                                        ↪ , exp, topxs[count],
                                        ↪ accuracy, c_accuracy,
                                        ↪ offset)
```

news2db.py

Fichero Python que recibe el listado de noticias para cada día del intervalo y lo importa a la base de datos.

```
import requests
import sqlite3
import datetime
import time
from datetime import timedelta, date

def range_date(start_date, end_date):
        for n in range(int((end_date - start_date).days)):
                yield start_date + timedelta(n)

n_news = 10
subreddit = 'worldnews'
uri = 'https://api.pushshift.io/reddit/search/submission'

start_date = date(2014, 2, 22)
end_date = date(2019, 2, 24)

conn = sqlite3.connect('tfg.db')
c = conn.cursor()
for news_date in range_date(start_date, end_date):
        print(str(news_date))
        date_beg = int(datetime.datetime(news_date.year, news_date.
            ↪ month, news_date.day, 0, 1).timestamp())
        date_end = int(datetime.datetime(news_date.year, news_date.
            ↪ month, news_date.day, 23, 59).timestamp())
        params = {'subreddit':subreddit, 'after':date_beg, 'before':
            ↪ date_end, 'size':n_news, 'sort_type':'num_comments', '
            ↪ sort':'desc'}
        petition = requests.get(uri, params = params)
        news = petition.json()
        news_list = []
        for new in news['data']:
                news_list.append(new['title'])
        c.execute("INSERT INTO news ('date', n1, n2, n3, n4, n5, n6,
            ↪ n7, n8, n9, n10) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
            ↪ ?)", (news_date.strftime("%d/%m/%Y"), news_list[0],
            ↪ news_list[1], news_list[2], news_list[3], news_list[4],
            ↪  news_list[5], news_list[6], news_list[7], news_list
            ↪ [8], news_list[9]))
        time.sleep(0.5)
```

```
        conn.commit()
```

Bibliografía

[1] S&P500,
https://es.wikipedia.org/wiki/S&P_500

[2] S&P Latin America 40,
https://en.wikipedia.org/wiki/S&P_Latin_America_40

[3] Nikkei 225,
https://es.wikipedia.org/wiki/Nikkei_225

[4] S&P Europe 350,
https://en.wikipedia.org/wiki/S&P_Europe_350

[5] Euro Stoxx 50,
https://en.wikipedia.org/wiki/Euro_Stoxx_50

[6] SSE Composite Index,
https://es.wikipedia.org/wiki/SSE_Composite_Index

[7] ¿Qué es una acción?,
https://www.ig.com/es/acciones/explicacion-acciones

[8] Realtime Data Mining aplicado a la predicción de índices de bolsa incluyendo Social Media Analytics,
https://upcommons.upc.edu/bitstream/handle/2117/112197/126952.pdf

[9] Predicción de valores de bolsa mediante minería de datos para mercado de alta frecuencia,
http://oa.upm.es/43108/1/TFG_ISABEL_VEGAS_VILLALMANZO.pdf

[10] r/WorldNews,
https://www.reddit.com/r/worldnews

[11] Pushshift API,
https://pushshift.io

[12] Yahoo! Finance,
https://es.finance.yahoo.com

[13] spaCy,
https://spacy.io

[14] Python,
https://www.python.org

[15] Google Cloud Natural Language API,
https://cloud.google.com/natural-language

[16] Decision Tree Learning,
https://en.wikipedia.org/wiki/Decision_tree_learning

[17] Support-Vector Machine,
https://en.wikipedia.org/wiki/Support-vector_machine

[18] K-Nearest Neighbors,
https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Sat Jun 08 19:22:00 CEST 2019
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)