UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

"ESTUDIO DE MODELOS DE DEEP LEARNING PARA ANÁLISIS DE SENTIMIENTOS EN TITULARES DE NOTICIAS FINANCIERAS"

TRABAJO FIN DE GRADO

Septiembre - 2023

AUTOR: Alejandro Lozano Pastor
DIRECTOR/ES: Jesús Javier Rodríguez Sala

RESUMEN

En este proyecto se tratará de abarcar un análisis del sentimiento de mercado, enfocado en empresas de capital abierto, es decir, aquellas que cotizan en bolsas de valores, ofreciendo a los inversores oportunidad de inversión en la compra de sus acciones.

Se realizará mediante el tratamiento y análisis de titulares de noticias financieras, asociadas a las entidades mediante el símbolo de sus acciones por el cual es conocida en el mercado o su propio nombre. Implementando técnicas de Deep Learning como el Procesamiento de Lenguaje Natural (NLP en inglés) sobre estos datos obtendremos su polaridad, valor cuantitativo sobre la positividad, negatividad y neutralidad del titular. Esta característica nos será realmente útil para poder visualizar patrones o correlaciones entre las noticias y los resultados de la gráfica bursátil con el fin de extraer los resultados y conclusiones.

AGRADECIMIENTOS

Aprecio a mi familia y amigos por estar a mi lado siempre que lo necesitara. También quisiera agradecer a mi tutor del proyecto, por todas aquellas reuniones y dedicación puesta.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Capítulo 1
Introducción

1.1.- Mercados financieros

Los mercados financieros abarcan cualquier lugar o sistema que proporcione a los participantes los medios para negociar activos financieros [1], otorgando a su comprador el derecho a recibir ingresos futuros por parte del vendedor. Así, se facilita la interacción entre quienes requieren capital con aquellos que están dispuestos a invertir.

Existe una fuerte relación entre el desarrollo del mercado y los indicadores macroeconómicos [2], utilizados en el estudio a nivel global de la economía. En base a ello, encontramos fundamental el correcto desarrollo de los mercados financieros nacionales.

Al ser un concepto amplio, sus funciones son clasificadas en base a diferentes criterios, en nuestro caso indagaremos en la diferenciación según su tipo de activos, donde encontramos [3] el mercado monetario, abarcando aquellos activos financieros a corto plazo, de

reducido riesgo, con alta liquidez y el mercado de capitales tratando transacciones de activos financieros a medio y largo plazo.

1.2.- Mercado de capitales

Los mercados de capitales son mercados financieros que reúnen a compradores y vendedores para negociar acciones, bonos, divisas y otros activos financieros. De este modo encontramos diferencias entre mercados en base al activo negociado.

Si, como objeto de negociación, encontramos préstamos o créditos bancarios, estamos hablando sobre el mercado crediticio o de crédito [4]. Este se desarrolla entre inversores y entidades de crédito las cuales ofrecen financiación, reservándose el derecho de su devolución a cambio de un interés en el plazo y forma pactado.

Por otra parte, si se negocia el intercambio de activos financieros tales como acciones, fondos u obligaciones estamos haciendo referencia el mercado de valores [4]. Haremos énfasis en este tipo de mercado, siendo objeto de interés del trabajo.

1.3.- Mercado de valores

El mercado de valores es, a grandes rasgos, el espacio donde asisten compañías interesadas en captar recursos financieros, las cuales ponen en circulación acciones, e inversores que disponen de capital y quieren invertir en estas empresas [5] esperando beneficios.

Debemos distinguir entre mercados primarios y mercados secundarios, siendo ambos parte del mercado de valores. La diferencia radica en la naturaleza de las acciones, en los mercados primarios los activos financieros son de nueva creación, es decir, los emisores ponen en circulación en este mercado los títulos nuevos y los demandantes o inversores los adquieren, como sucede cuando una empresa se instaura en una bolsa de valores y ofrece por primera vez sus acciones.

Posteriormente, los valores ya adquiridos por los inversores se negocian mediante su venta por parte del comprador y la posible compra por otros inversores, dando paso al mercado secundario. Es en este mercado es donde resultan los beneficios de los inversores.

También existe diferenciación según la variación de los activos, existen mercados de valores de renta fija donde las inversiones generan intereses de forma regular, siendo determinado de antemano por el emisor del valor. Estos mercados son considerados más confiables en comparación con los mercados de valores de renta variable, donde los rendimientos de las acciones no son fijos y conocidos de antemano por el inversor, sino

que dependen de la situación económica del contexto en el que se enmarca un determinado mercado, también llamados dentro del marco como “bolsas”, y de la situación específica de la compañía a la que pertenecen estos activos. En este tipo de mercado los beneficios tienen una fuerte dependencia de las fluctuaciones de las acciones.

Por esta razón, el análisis de las bolsas de valores es un objeto difícil de estudio, donde los investigadores del campo aplican diferentes técnicas con el fin de obtener mejores rendimientos. Una de estas técnicas es el análisis del sentimiento del mercado, el cual trata de estudiar posibles variaciones en el valor de las acciones de una empresa ocasionadas por la influencia en la sociedad.

1.4.- Sentimiento del mercado

Cada día se genera una enorme cantidad de datos procedentes de las redes sociales, noticias y otras fuentes de información sobre sucesos diversos, los cuales son comunicados casi de forma inmediata. El saber de esta información nos puede generar influencias y posiblemente nuestras acciones varíen en base a ello.

De este modo, al encontrar noticias relacionadas con los mercados financieros, podemos ser animados o disuadidos de realizar ciertas operaciones, como vender acciones de una empresa que ha sido calificada negativamente o comprar aquellas de nuevas compañías descubiertas con gran potencial emergente. Si estas acciones son realizadas por un grupo notable de inversores pueden ser percibidos los cambios en el mercado.

En los tiempos donde el acceso a esta información no estaba tan generalizado, ya se habían encontrado evidencias de correlación entre estos dos campos, siendo en el año 2007 Aul C. Tetlock pionero en su estudio. Mediante su publicación en la revista The Journal Of Finance [6], analizó los artículos periodísticos de la columna “Abreast of the Market” del Wall Street Journal desde 1984 hasta 1999 donde, haciendo uso del programa General Inquirer (GI), concluyó que las noticias con índole negativa afectan de forma más prolongada en el tiempo y anomalías en los valores generan un aumento temporal del volumen del mercado.

Más adelante, junto a una mayor disponibilidad de los datos, gracias a la proliferación de plataformas en línea como las redes sociales o portales de noticias digitales, serían anunciados mayores resultados, como el expuesto por el Fondo Monetario Internacional en 2018 [7]. Analizando más de 4,5 millones de artículos de la agencia de noticias Reuters publicados en todo el mundo entre 1991 y 2015, descubrieron que su medida obtenida del sentimiento de las noticias predecía con solidez los rendimientos diarios en los mercados avanzados y emergentes. También notificaron diferencias entre los impactos de las noticias

globales y locales, donde éstas últimas solamente alteraban los rendimientos durante unos pocos días.

1.5.- Deep Learning

Nos encontramos en un periodo emocionante dentro del mundo de la computación. En estos momentos es posible abarcar problemas que se pensaban inalcanzables hasta hace pocos años y, lo mejor de todo, es que estamos logrando grandes resultados sobre ellos. Gran parte de esto se debe al uso de técnicas de Inteligencia Artificial, y más en concreto, de Deep Learning, el cual se caracteriza por el uso de redes neuronales.

Estas técnicas se encuentran en evolución, favorecidas en gran medida por la aparición del Big Data, el gran crecimiento computacional, llegando a poder ejecutar modelos en potentes servidores alojados en la nube o el surgimiento de herramientas como TensorFlow, Keras o PyTorch [8], enfocadas al entrenamiento de estos sistemas. Estos avances han facilitado notablemente la implementación en multitud de proyectos, eliminando gran parte de las restricciones hardware y mejorando la barrera de aprendizaje. Por ende, también se encuentran en expansión resolviendo problemas en campos donde, a priori, no estaban relacionados con la informática, como pueden ser el marketing [9], medicina [10], diseño [11] o los mercados financieros, objeto de estudio del trabajo.

Algunas de las aplicaciones del Deep Learning en el campo financiero son el trading algorítmico [12], donde algunos investigadores están comenzando a implementar modelos de redes neuronales en sus estudios o algoritmos capaces de asistir e incluso operar como en el caso de los robots automatizados [13]; el análisis de valores temporales [14] tratando datos históricos de las acciones con el fin de encontrar patrones que puedan volver a suceder; o el anteriormente mencionado análisis de sentimiento del mercado, el cual trataremos de abordar en este proyecto.

1.6.- Justificación del proyecto

Desde su comienzo, la Inteligencia Artificial no ha tenido un desarrollo fácil, con muchos periodos prometedores seguidos por momentos desesperanzadores. Sin embargo, en la actualidad estamos viviendo uno de los momentos de mayor crecimiento, logrando objetivos que parecían imposibles hasta hace pocos años.

Estamos observando el gran potencial que tienen las técnicas de Inteligencia Artificial, como el Deep Learning, siendo incorporadas a problemas que abarcan multitud de campos, manifestando una gran flexibilidad.

La motivación del proyecto viene con la implementación de técnicas de este campo y los mercados de valores, donde, a pesar de ser un entorno de difícil estudio, encontramos una gran cantidad de datos que pueden ser analizados.

La recolección y tratamiento de estos datos representan a su vez una gran parte de la motivación del trabajo, implementando nuevas técnicas y experimentando su aprendizaje.

1.7.- Objetivos

En este proyecto se tratará de abarcar un análisis del sentimiento de mercado mediante el tratamiento de noticias financieras e implementación de técnicas de Deep Learning.

1.7.1.- Objetivos principales

- Exploración de los mercados financieros.
- Uso de técnicas de extracción de datos.
- Tratamiento y análisis de datos.
- Implementación de redes neuronales.

1.7.2.- Objetivos personales

- Ampliar conocimientos sobre Inteligencia Artificial.
- Aprendizaje en uso de redes neuronales.
- Ampliar conocimientos en lenguaje Python.
- Mejorar habilidades de tratamiento de datos.

1.8.- Límites del proyecto

Para la obtención de resultados no se tomarán en cuenta datos a tiempo real, sino archivos con datos históricos, con el fin de encontrar relaciones que puedan ser extrapoladas a casos futuros.

El proyecto se centrará en un único campo del mercado financiero, los mercados de valores, dejando así fuera el resto de ellos, aunque no se descartan adaptaciones en el futuro.

Capítulo 2 Antecedentes y estado de la cuestión

2.1.- Análisis del sentimiento financiero

El análisis de sentimientos se encuentra dentro de las áreas de investigación de más rápido crecimiento relacionadas con la computación según Mäntylä et al. [15] donde, en 2018, tras realizar un extenso estudio bibliográfico descubrieron que, el mercado financiero es gran tema de interés dentro del análisis de sentimientos, observando una gran cantidad de artículos relacionados con este ámbito. Antes de ser despertado el interés por los mercados financieros, el análisis de sentimientos se centraba en las valoraciones de productos disponibles en la web, junto a la irrupción del análisis de sentimientos moderno a mediados de la década de 2000.

A medida que aumentaba la disponibilidad de noticias y comentarios financieros en línea, los inversores y operadores empezaron a utilizar el análisis del sentimiento para fundamentar sus decisiones de negociación e inversión. Así surgió un nuevo nicho de mercado donde proliferaron muchas empresas poniendo en práctica diversos métodos para el estudio de este campo y facilitando así la tarea de inversión.

Actualmente encontramos una amplia oferta de algoritmos empleados en el marco del mercado financiero, aunque también existe una carencia de documentación acerca de ellos, debido a que la mayoría se utilizan comercialmente y su funcionamiento no es de dominio público.

2.1.1.- MoneyControl

Para analizar el sentimiento financiero, MoneyControl utiliza una combinación de análisis de expertos y comentarios de los usuarios. Así, son capaces de seguir el sentimiento del mercado y proporcionar información sobre los factores que impulsan el sentimiento de los inversores y las tendencias del mercado. También utilizan noticias y fuentes de redes sociales para supervisar el sentimiento y proporcionar información más fiable a sus lectores.

Ofrecen una prueba gratuita de su servicio MoneyControl PRO, gracias al cual podemos desbloquear análisis más detallados como el sentimiento del mercado relativo a indicadores técnicos o medias móviles [16].

2.1.2.- RavenPack

La empresa RavenPack utiliza algoritmos de procesamiento del lenguaje natural y aprendizaje automático para analizar artículos de noticias, redes sociales y otras fuentes textuales con el fin de medir el sentimiento de los inversores hacia determinadas empresas o mercados.

Mediante su software sofisticado, RavenPack Edge, proporcionan fuentes de datos personalizadas con un gran número de filtros y herramientas de análisis para ayudar a las empresas a tomar decisiones de inversión. Su tecnología es capaz de analizar datos en varios idiomas y abarca una amplia gama de clases de activos, como acciones, bonos, materias primas y divisas [17].

2.1.3.- John Snow Labs

John Snow Labs es una empresa que ofrece soluciones de procesamiento del lenguaje natural (PLN) para diversos sectores, entre ellos el financiero. Utilizan su propia librería de aprendizaje profundo llamada Spark NLP [18]. Esta librería es capaz de hacer uso de la mayoría de los modelos de lenguaje encontrados dentro en el estado del arte relacionado con el procesamiento del lenguaje natural, siendo más de 15000 el número de ellos y entre

los cuales encontramos un gran número de variantes del modelo BERT, los cuales analizaremos más adelante.

2.2.- Web scraping

La extracción de datos es una parte esencial del trabajo con ideas nuevas e innovadoras, o simplemente, para la creación de nuevos conjuntos de datos. La recolección manual de datos requiere demasiado tiempo y esfuerzo y como resultado obtenemos imprecisiones y falta de completitud.

Por ello surgieron técnicas automáticas denominadas "web scraping" (traducido como raspado de web en español), las cuales realizan el proceso de recolección de datos estructurados ofrecidos por las páginas webs. Se trata de una técnica muy útil para investigadores, los cuales no disponen de acceso a grandes bancos de datos necesarios para realizar los estudios o que simplemente no existen previamente y deben ser creados desde cero.

Mediante su uso podemos obtener los datos requeridos para la formación de grandes bancos de datos personalizados y enfocados en la tarea específica que deseamos realizar, consiguiendo una mayor calidad de los datos, aumentando así la calidad del estudio [].

También se debe tener en cuenta que nos encontramos en la era de la información, donde los datos tienen un valor muy alto tanto financiero como social, por ello muchas empresas no permiten la realización de este tipo de técnicas en sus páginas web. Esta situación está condicionada por muchos factores, como el tipo de datos buscados o el propósito de su extracción, además de las diferentes políticas de uso que pueden ser adoptadas en las empresas. En este trabajo trataremos de obtener los datos teniendo en cuenta esta premisa.

2.2.1.- Análisis sintáctico (Parsing)

Mediante el análisis sintáctico de textos, también conocido como "parsing", es posible la recopilación de datos incluidos dentro de una página web. El análisis más común es el del lenguaje HTML que permite extraer los datos visualizados por los navegadores. Para ello se debe realizar una serie de pasos con el fin de obtener el árbol DOM (Document Object Model), el cual describe el contenido del documento HTML.

Algunos de estos pasos son la entrada de flujo de caracteres Unicode, los cuales son decodificados y preprocesados para su tokenización seguida de una etapa de construcción de árbol. La etapa de construcción del árbol es reentrante, lo que significa que mientras la etapa de construcción del árbol está manejando un token, el tokenizador podrá reanudarse,

haciendo que se emitan y procesen más tokens antes de que se complete el procesamiento del primer token [19].

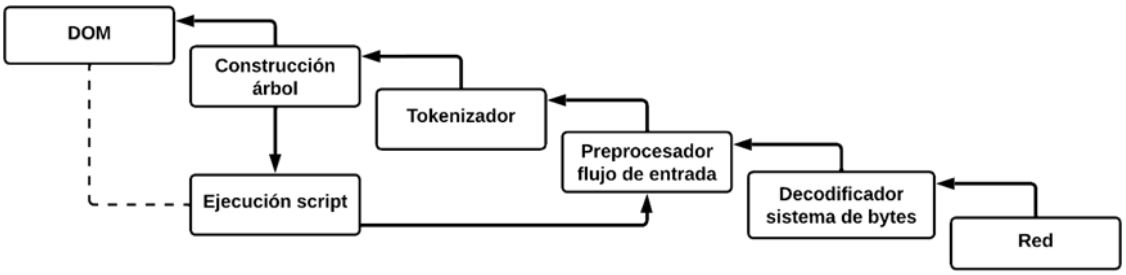

Fig 2.1: Resumen análisis sintáctico HTML.

Se trata de una técnica ampliamente usada en combinación con el web scraping, pero tiene sus limitaciones, si el contenido deseado es cargado de forma dinámica en la página mediante JavaScript no será reconocido y, por lo tanto, no aparecerá en el árbol resultado. Este es el caso de la mayor parte de portales de noticias financieras encontrados, por ello no es muy adecuado su uso en el proyecto. Además hay que tener en cuenta una parte legal, ya que no todos los portales de noticias con los que podríamos trabajar permiten el uso de técnicas de web scraping para trabajar con los datos mostrados en su página web, como hemos podido comprobar, algunos de ellos indican el no permiso en su apartado términos y condiciones [20].

También existen análisis sintácticos de otros lenguajes web como XML, cuya principal funcionalidad es el almacenamiento y transporte de datos. Esto resulta muy interesante para nuestro proyecto ya que es el lenguaje propio de los RSS (Really Simple Syndication), en español «Sindicación Realmente Simple», recurso utilizado por los medios de comunicación para distribuir contenido como noticias o artículos en la web [21]. Siendo estos los datos objetivos que buscamos recopilar en este proyecto.

Algunas librerías como BeautifulSoup o la propia librería estándar del lenguaje Python, que dispone de funciones de “parsing” incluidas.

2.3.2.- Automatización de navegadores web

La automatización de navegadores es el proceso de realizar automáticamente operaciones en un navegador web simulando las funciones que un usuario podría realizar de forma manual, alcanzado niveles de velocidad y eficacia que no serían posibles con la intervención humana. Aplicado al campo del web scraping, esta técnica permite alcanzar mejores resultados, como por ejemplo acceder a aquellos datos cargados dinámicamente, acceder a páginas que requieren un inicio de sesión o extraer más datos navegando en su paginación [22].

Algunas empresas como Microsoft, mediante Power Automate, ofrecen servicios para diferentes casos de uso, pudiendo ser utilizados para el web scraping. También existen frameworks como Selenium [23] o Puppeteer [24], que permiten desarrollar mediante código programas centrados en la recolección de datos.

2.3.3.- Web API

Existen empresas que proporcionan APIs capaces de realizar extracción de datos en sitios web, simplificando en mayor medida el proceso. Estas API permiten a los desarrolladores extraer fácilmente datos de páginas web sin tener que escribir complejos scripts de web scraping, así ofrecen diferentes planes de suscripción junto a restricciones acordes a ellos, como por ejemplo la limitación de un número máximo de solicitudes que puede realizar un usuario al mismo tiempo, acotando así la cantidad máxima de datos a extraer [25]. Algunas herramientas encontradas en el mercado son las pertenecientes a las compañías Srapfly [26] y Scrapingbee [27], o la anteriormente mencionada RavenPack [17].

2.3.4.- Análisis web mediante visión por ordenador

Existen técnicas alternativas para la recolección de datos como el análisis mediante visión artificial, y el reconocimiento óptico de caracteres (OCR). Esta tecnología permite a los usuarios extraer datos de texto de imágenes o documentos escaneados en páginas web. Este método es también llamado “screen scraping” (“raspado de pantalla” en español) ya que es capaz de extraer cualquier información mostrada por pantalla al usuario [28].

Sin embargo, existen algunas limitaciones encontradas en el uso de software de OCR como dificultades para reconocer fuentes pequeñas o inusuales pertenecientes a la imagen de entrada, dificultades para reconocer datos de texto en columnas, tablas u otros diseños complejos o el reconocimiento sobre imágenes cuyo texto no sea muy legible, ya sea por el diseño de la paleta de colores del sitio web o una baja calidad de la imagen. Todos estos problemas pueden generar un resultado con significado alterado al original [29].

Por esta razón quizás no es una tecnología idónea para la recolección de grandes cantidades de datos, objetivo de este proyecto, pero sin duda es un método a tener en cuenta en el futuro si evoluciona positivamente.

2.3.- NLP (Natural Language Processing)

El procesamiento del lenguaje natural (NLP como acrónimo en inglés) es la disciplina de la Inteligencia Artificial que se ocupa de dotar a los ordenadores de la capacidad de

comprender textos y palabras habladas del mismo modo que lo hacen los seres humanos. Este campo combina la lingüística computacional, lenguaje humano basado en reglas, con modelos estadísticos, de aprendizaje automático y de aprendizaje profundo. Juntas, estas tecnologías permiten a los ordenadores procesar el lenguaje humano en forma de texto o datos de voz y "entender" su significado completo, con la intención y el sentimiento del hablante o escritor [30].

El PLN puede dividirse en dos subcampos que se solapan: la generación de lenguaje natural (NLG), que se centra en la generación de texto por una máquina, como ChatGPT o Google Bard, y la comprensión del lenguaje natural (NLU), que se centra en el análisis semántico o la determinación del significado del texto, el cual será puesto en práctica en este proyecto [31].

2.3.1.- BERT

BERT (por sus siglas en inglés Bidirectional Encoder Representations from Transformers) es un potente modelo de procesamiento del lenguaje natural, está basado en una arquitectura de red neuronal apoyada en transformadores que fue introducida por investigadores de Google AI en 2018.

Al término de su presentación, mostraba resultados pioneros en 11 tareas diferentes de procesamiento del lenguaje natural entre ellas el análisis de sentimiento, el etiquetado de funciones semánticas, la clasificación de frases y la desambiguación de palabras polisémicas o con múltiples significados [32].

La principal innovación técnica de BERT es la captación de información contextual, aprendiendo conjuntamente del contexto izquierdo y derecho de una palabra o token analizado, se trata de un uso bidireccional dentro de la arquitectura de Transformer. Esto contrasta con los esfuerzos anteriores, que analizaban una secuencia de texto de izquierda a derecha o combinaban el entrenamiento de izquierda a derecha y de derecha a izquierda. Esta diferencia permite a BERT generar representaciones de palabras de alta calidad que incorporan el significado y las dependencias de toda la frase [33].

Para su entrenamiento se aplicaron dos técnicas, “Masked Language Modeling“ (Modelización del lenguaje enmascarado) y “Next Sentence Prediction” (Predicción de la siguiente frase).

Masked Language Modeling (MLM)

Fue implementada por primera vez en el entrenamiento del modelo BERT por Jacob Devlin y su equipo, quienes encontraron inspiración en la famosa prueba Cloze,

pronunciada por Wilson L Taylor en 1953 [34], utilizada para el aprendizaje de idiomas. En esta prueba, se omite parte del contexto, siendo elementos, palabras o signos, y se le pide al participante reemplazar el contexto faltante.

Del mismo modo, la técnica Masked LM omite parte de la entrada de datos al modelo, siendo reemplazados por tokens llamados mask (máscara en inglés), los cuales debe ser capaz de averiguar tras el resultado de su proceso. En el modelo original, esta selección de palabras era del 15% del input [33], seleccionadas de forma aleatoria, aunque no todos los tokens dentro de la selección eran reemplazados, esto es debido a los diferentes resultados obtenidos en la variación de esta regla.

Encontraron que, si los tokens seleccionados eran siempre enmascarados el modelo no producía buenas representaciones de tokens para las palabras no enmascaradas, las cuales eran sólo usadas para el contexto, optimizando el modelo para sólamente predecir los tokens seleccionados; por otro lado, si era enmascarado el token en un 80% de las veces y reemplazados por la palabras correctas el otro 20% del tiempo, el modelo detectaba que si el token no era enmascarado era la palabra correcta y no necesitaba predecirla, obviando esta tarea de aprendizaje; y si en vez de reemplazar ese 20% por palabras correctas eran insertadas palabras aleatorias el modelo las trataba como tokens [MASK], resultando en los mismos problemas que la primera regla.

Tras este análisis se concluyó que la mejor estrategia era del orden 80-10-10 (Tabla 2.1), siendo un 80% de las veces enmascarados, un 10% del tiempo siendo la palabra correcta y el 10% restante siendo una palabra aleatoria. Los autores en el propio artículo señalaron:

"La ventaja de este procedimiento es que el codificador Transformer no sabe qué palabras se le pedirán que prediga o cuáles han sido sustituidas por palabras aleatorias, por lo que se ve obligado a mantener una representación contextual distribucional de cada token de entrada".

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

Tabla 2.1: Resultados sobre diferentes estrategias de enmascaramiento.

De esta manera, encontraron un buen método de entrenamiento de un modelo bidireccional, como es el caso de BERT, ya que el modelo es forzado a aprender el contexto de las frases, a partir de los tokens sin enmascarar, significando en una mejora

sustancial del nivel de las respuestas. Sin embargo estas mejoras del rendimiento aparecen tras un corto número de primeros pasos en el entrenamiento, esto es debido a que el modelo BERT sólamente predice aquellas pocas palabras seleccionadas con el token de enmascaramiento en cada lote o batch. Como podemos observar en la Fig. 2.2, la precisión del modelo haciendo uso de la técnica MLM es superior a la convencional al paso de los 100 mil pasos en el pre entrenamiento.

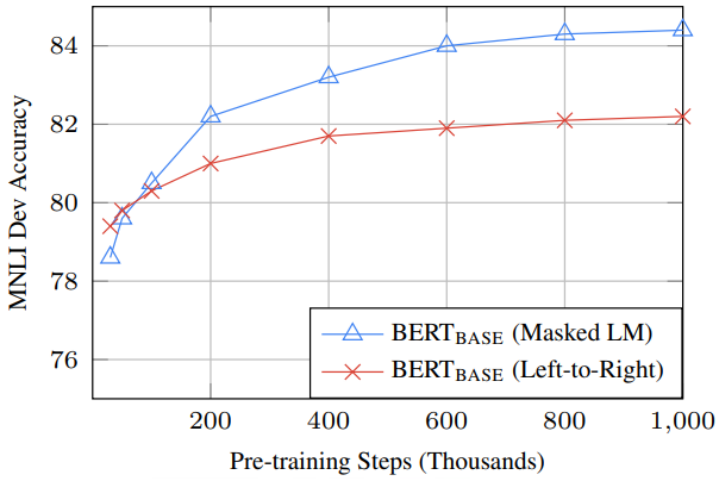

Fig 2.2: Comparación de la precisión del modelo BERT usando MLM.

Estudios posteriores han demostrado empíricamente la solidez de los modelos MLM pre entrenados, una mejor capacidad de generalización [35] y resistencia a datos fuera de distribución [36]. Sin embargo, todavía sigue sin responderse el por qué de esta robustez.

Para el uso de este entrenamiento hay que añadir ciertos elementos a la estructura del modelo, como una capa de clasificación sobre la salida del codificador, la cual juzgará el mejor resultado posible en base a porcentajes de acierto; se debe realizar una multiplicación de los resultados sobre una matriz de incrustación (embedding matrix en inglés) para transformarlos en la dimensión de vocabulario; y el cálculo de la probabilidad de cada palabra del vocabulario con la función de softmax, convirtiendo el vector de valores reales en una distribución de probabilidad [37] (Fig. 2.3).

Next Sentence Prediction (NSP)

Durante el desarrollo del modelo, encontraron que muchas de las tareas más importantes, como la respuesta a preguntas (Question Answering) y la inferencia en lenguaje natural (Natural Language Interference), se basan en la comprensión de la relación entre dos frases.

Por ello entrenaron el modelo mediante pares de frases como entrada siendo el 50% del tiempo la segunda frase una consecutiva en el dataset original, mientras que en el otro 50% se elige una frase aleatoria del dataset como segunda frase, siendo completamente

desconectada de la primera. De este modo, el modelo aprende el contexto de ambas frases, consiguiendo predecir si la segunda frase del par es la frase posterior del documento original.

Para el uso de esta técnica se requieren ciertas modificaciones en la entrada del modelo, como la incrustación (embedding) de un token CLS al principio de la primera frase, indicando que es la frase original y no la que debe averiguarse, y tokens SEP al final de cada una, que actuarán como limitadores; para la creación de los datos de entrada se tendrá en cuenta un token llamado token de segmento, el cual indica la frase a la que pertenece el dato, siendo los posibles valores frase A o B; un token de posición, el cual indica la posición a la que pertenece el dato dentro de la frase; y el propio token, ya sea el dato original o los tokens de inicio de frase o separadores. La suma de estos tres últimos tokens generarán el vector de datos que será introducido al codificador (Fig. 2.3).

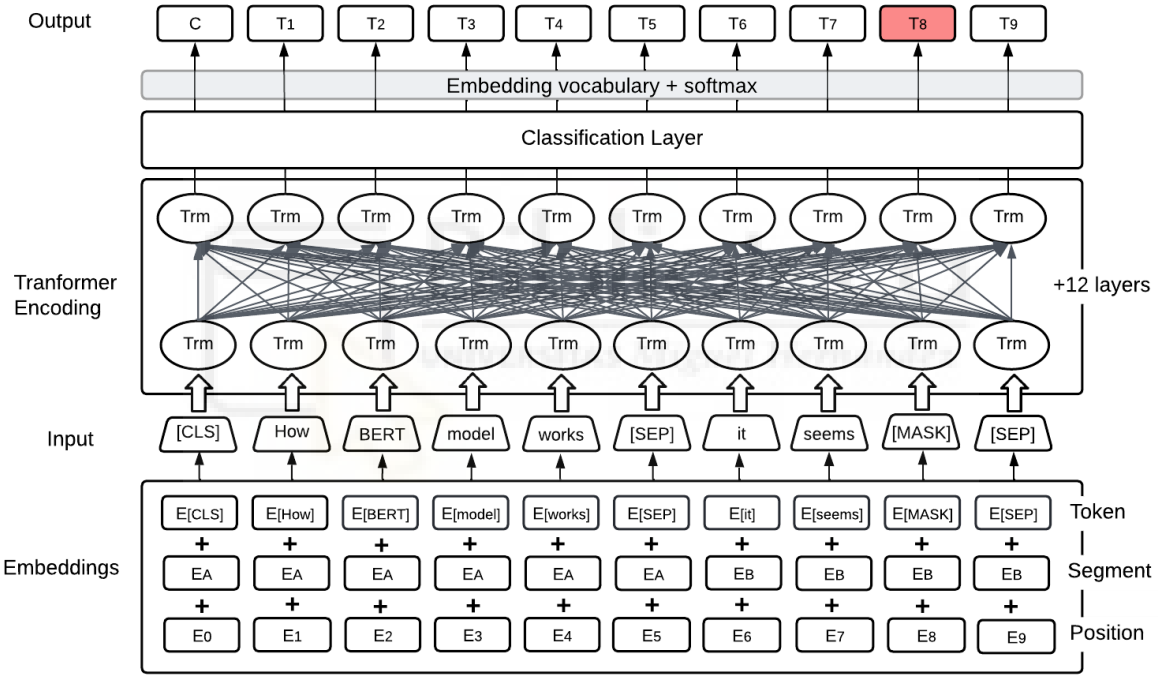

Fig 2.3: Estructura del modelo BERT en su entrenamiento.

Estas dos técnicas de entrenamiento se realizan a la vez, con el fin de obtener el mayor rendimiento de ambas. Así, el modelo fue entrenado con 2.3 mil millones de palabras obtenidas de Wikipedia y Google BooksCorpus durante un período de 4 días.

Este enorme dataset fue clave para la obtención de los resultados, pero también supone una gran carga computacional en su uso. Este inconveniente es resuelto gracias a la amplia oferta de modelos obtenidos mediante la técnica de "fine-tuning" (afinamiento en español), obteniendo modelos con rendimientos similares a los originales pero junto a un incremento en sus tamaños. Esta técnica será explicada a fondo en el Capítulo 4: Metodología y resultados donde se realizará un entrenamiento afinado de un modelo BERT.

Además, mediante esta técnica son obtenidos modelos adaptados a un campo en concreto, añadiendo capas específicas y entrenando con datos etiquetados. Con esto se obtiene un modelo más eficaz y de mayor calidad, siendo especializado en el tema propio de la investigación, proporcionando así mejores resultados.

2.3.1.1.- FinBERT

Este es el caso del modelo FinBERT, creado por la compañía Prosus en 2019 [38], se trata de un modelo lingüístico basado en BERT y perfeccionado para la clasificación de sentimientos en el ámbito financiero. Fue pre-entrenado con el dataset TRC2 [39], proporcionado por la agencia de noticias Reuters, el cual cuenta con 1,8 millones de artículos financieros publicados entre 2008 y febrero de 2009, con el fin de adaptar el modelo pre-entrenado BERT al lenguaje utilizado en el mundo financiero.

Posteriormente, fue ajustado con datos etiquetados para la clasificación del sentimiento financiero mediante el entrenamiento con el conjunto de datos "Financial PhraseBank" [40], una selección de 4500 frases de diversos artículos de noticias, que incluyen términos financieros etiquetados por 16 expertos y estudiantes del campo financiero.

Además, fue añadida una capa densa al último estado oculto de la tokenización, pertenecientes a las tareas secuenciales como la clasificación de frases o la vinculación textual. Gracias a este afinamiento, el modelo FinBERT fue capaz de conseguir un aumento de la precisión en 15% respecto al estado del arte del momento.

2.3.1.2.- distilBERT

Fue introducido en 2019 por el equipo de Hugging Face, empresa dedicada a la creación de herramientas open source de aprendizaje automático, como resultado de su búsqueda para reducir el tamaño de los grandes modelos encontrados en el actual estado del arte. Sus esfuerzos se centraron en la técnica de compresión "destilación de conocimientos" [41], donde un modelo pequeño se entrena para reproducir el comportamiento de un modelo mayor. Estos dos modelos participantes se suelen tratar como profesor y alumno, ya que, el modelo más pequeño será supervisado por el modelo de mayor tamaño.

De este modo, se creó un nuevo modelo llamado distilBERT que consiguió eliminar del modelo BERT original las incrustaciones de tipo token y el pooler (utilizado para la siguiente tarea de clasificación de frases) manteniendo el resto de la arquitectura idéntica, cambiando únicamente el número de capas en un factor de dos, resultando en 6 capas. Aún sufriendo estos cambios, el modelo es capaz de obtener más del 95% del rendimiento del modelo original siendo más ligero, con un 40% menos de parámetros [42].

2.3.1.3.- RoBERTa

El modelo RoBERTa (A Robustly Optimized BERT Pretraining Approach) [43] fue introducido en 2019 por investigadores de Facebook AI y de la Universidad de Washington como una optimización del modelo BERT, buscando una mejora del tiempo durante el pre-entrenamiento. Para lograr este objetivo, los autores realizaron algunos cambios sencillos de diseño en su arquitectura y procedimiento de entrenamiento sobre el modelo original.

En el desarrollo de este modelo experimentaron con la eliminación y adición de la pérdida Next Sentence Prediction a diferentes versiones y llegaron a la conclusión de que la eliminación de la pérdida NSP iguala o mejora ligeramente el rendimiento.

También se entrenó el modelo con un conjunto de 160 GB de datos de texto, siendo 10 veces mayor que el conjunto de datos utilizado para entrenar BERT, permitiendo así mayor facilidad en el entrenamiento en paralelo y mejorando la perplejidad del modelado del lenguaje enmascarado. Este último fue modificado, haciendo un uso dinámico de la técnica. Para lograr esto, los datos de entrenamiento fueron duplicados y enmascarados 10 veces, cada vez con una estrategia de máscara diferente a lo largo de 40 épocas o rutinas, obteniendo así sólamente 4 épocas con la misma máscara al finalizar el entrenamiento. Este cambio consigue resultados comparables o ligeramente mejores al modelo BERT original.

2.3.2.- VADER

VADER (Valence Aware Dictionary and sEntiment Reasoner) [44] es un léxico y una herramienta de análisis de sentimientos para texto basada en reglas creada por el Instituto de Tecnología de Georgia en 2014.

Se desarrolló para el análisis de sentimientos de textos, el modelo utiliza una combinación de léxico y reglas gramaticales para determinar el sentimiento (positivo, negativo o neutro) de un texto determinado. El léxico de sentimientos contiene una lista de palabras o frases junto con sus puntuaciones de sentimiento, que indican lo positivas o negativas que son y modificadores de intensidad que pueden amplificar o atenuar el sentimiento de una palabra.

Además del léxico, VADER incorpora un conjunto de heurísticas y reglas para tratar casos como las mayúsculas, la puntuación, los modificadores o adverbios de grado y el sentimiento basado en elementos gramaticales que cambian el contexto negativamente,

como la conjunción contrastiva “pero”. Estas reglas ayudan a VADER a comprender mejor el sentimiento expresado en el texto y a realizar predicciones de sentimiento más precisas.

VADER ha ganado popularidad debido a su sencillez y eficacia, especialmente para el análisis del sentimiento en fuentes de texto informales como las redes sociales. Está implementado en varios lenguajes de programación, incluido Python, como en la librería NLTK (Natural Language Toolkit), y está disponible como herramienta de código abierto.

Capítulo 3
Hipótesis de trabajo

3.1. Python

La totalidad del trabajo será programado en el lenguaje de programación Python, actualmente es considerado el lenguaje por excelencia en el desarrollo de proyectos de data science, siendo este campo donde más se usa [45].

Esto se debe a las enormes ventajas ofrecidas tanto en el tratamiento y análisis de datos, como en las posibilidades de visualización de resultados. Una característica destacable es la gran cantidad de recursos que se pueden encontrar gracias a la comunidad de desarrolladores, y la existencia de librerías como Pandas, statsmodels, NumPy, SciPy, o Scikit-Learn, las cuales ofrecen una amplia gama de herramientas y facilitan en gran medida las tareas necesarias en proyectos de ciencia de datos.

La limpieza y el tratamiento de los datos es una de las principales tareas de los científicos de datos, y, junto a la incorporación al trabajo de estas librerías es ahorrado mucho tiempo y esfuerzo en la labor.

3.2. Spyder

La mayor parte del trabajo se realizará sobre el Entorno de Desarrollo Integrado (por las siglas IDE en inglés) Spyder, entorno científico gratuito y de código abierto escrito en Python, para Python, y diseñado por y para científicos, ingenieros y analistas de datos [46].

Ofrece un interfaz de usuario muy práctico, enfocado en la exploración de los datos, permitiendo observar variables de forma fácil y rápida, además de poder visualizar gráficas por medio de ventanas, siendo estas funcionalidades muy útiles para el desarrollo de este proyecto. En este trabajo se va a usar la versión 5.2.2 que contiene la versión 3.9 de Python.

3.3. Google Colaboratory

También conocido como su abreviación “Colab” o “Google Colab” es un servicio alojado de Jupyter Notebook ofrecido por Google que no requiere configuración y proporciona acceso gratuito a recursos informáticos, incluidas GPU y TPU [47].

Colab es especialmente adecuado para el aprendizaje automático ya que permite utilizar sus recursos para tareas con alta carga computacional en lenguajes como Python. En el presente trabajo, esta herramienta se utilizará para el entrenamiento de un modelo ajustado como alternativa al anterior entorno de programación comentado.

3.4.- Herramientas para obtención de datos

Se emplearán diferentes herramientas para la obtención de los diferentes conjuntos de datos utilizados en este proyecto. Estas serán especializadas según las funciones que haya que realizar, facilitando en gran medida sus procesos.

3.4.1.- Obtención de noticias financieras con PyGoogleNews

Creada por Artem Bugara en 2020 mediante el uso de BeautifulSoup4 y FeedParser, la herramienta PyGoogleNews simula las funciones de la ya extinta Google News API, la cual dejó de funcionar en 2011 [48].

Se trata de un envoltorio en Python del canal RSS de Google News que permite filtrar por noticias destacadas, fuentes de noticias relacionadas con temas, fuentes de noticias geolocalizadas y una amplia fuente de búsqueda de texto completo, en la propia web de la

herramienta, esta se define como "*una recopilación de todo lo que han podido averiguar sobre el funcionamiento de Google News*" [49].

Gracias a esta herramienta se pueden obtener algunos datos requeridos para el proyecto, en este caso, noticias relacionadas con una empresa o ticker procedentes de diferentes fuentes de medios de comunicación y en grandes cantidades, ya que existe un gran histórico de datos dentro del portal de Google News.

3.4.2.- Obtención de histórico de acciones con yfinance

Para poder comparar los resultados obtenidos de las noticias financieras con el contexto financiero de la empresa en cada momento, es necesario obtener una gráfica del valor de mercado de la compañía objetivo. Existen numerosas páginas web que muestran este gráfico en tiempo real, cuyos datos podrían ser obtenidos mediante web scraping, pero preferiblemente se usará para este caso una librería capaz de proporcionarnos estos datos.

Por lo tanto, en la obtención de datos pertenecientes al mercado de valores se va a utilizar la librería de Python yfinance. Fue creada por Ran Aroussi en 2019 con el objetivo de simular la extinta API de Yahoo! Finanzas de manera temporal, pero, a día de hoy, sigue recibiendo soporte y es utilizada ampliamente por desarrolladores del campo financiero.

Proporciona una gran facilidad para la obtención de los datos, pudiendo ser recopilados simplemente con una instrucción. Además, permite obtener datos pertenecientes dentro de un periodo de tiempo especificado y sobre un ticker bursátil en concreto [50].

3.5.- Datos de Entrenamiento

En la tarea de entrenamiento del modelo BERT se requiere un banco de datos con el que entrenar el modelo. Estos datos deben relacionarse con el campo noticias financieras y estar etiquetados según su sentimiento. Los datos utilizados para entrenar el modelo BERT se descargaron del dataset Financial PhraseBank [40] creado y utilizado por Malo et al. (2014) [51].

Este dataset contiene frases de noticias financieras de empresas que cotizan en OMX Helsinki, consta de 4.840 sentencias de noticias financieras en inglés, las cuales se encuentran divididas según su índice de polaridad, anotado manualmente por 16 anotadores humanos (Fig. 3.1). Estos anotadores son investigadores en el campo y estudiantes del máster de la Escuela de Negocios de la Universidad de Aalto (Finlandia) con especialidades principalmente en finanzas, contabilidad y economía.

Las noticias fueron obtenidas por el equipo de la Universidad de Aalto de la base de datos de la compañía LexisNexis mediante un raspador web automático. Gracias a esta técnica se obtuvo un subconjunto aleatorio de 10.000 artículos, de los cuales fueron excluidos los que no contenían ninguna de las entidades del léxico especificado como perteneciente al ámbito financiero. La muestra final se redujo a 53.400 frases procedentes de los mencionados artículos, y fueron clasificadas según su polaridad (*@negative, @positive, @neutral*). Por último, se eligió una muestra aleatoria de aproximadamente 5.000 frases para representar la base de datos global de noticias [52].

```
1  "According to Gran , the company has no plans to move all production to
   Russia , although that is where the company is growing .@neutral"
2  "At the request of Finnish media company Alma Media 's newspapers ,
   research manager Jari Kaivo-oja at the Finland Futures Research Centre
   at the Turku School of Economics has drawn up a future scenario for
   Finland 's national economy by using a model developed by the
   University of Denver .@neutral"
3  "For the last quarter of 2010 , Componenta 's net sales doubled to
   EUR131m from EUR76m for the same period a year earlier , while it moved
   to a zero pre-tax profit from a pre-tax loss of EUR7m .@positive"
4  "In the third quarter of 2010 , net sales increased by 5.2 % to EUR
   205.5 mn , and operating profit by 34.9 % to EUR 23.5 mn .@positive"
5  "Jan. 6 -- Ford is struggling in the face of slowing truck and SUV
   sales and a surfeit of up-to-date , gotta-have cars .@negative"
6  "Pharmaceuticals group Orion Corp reported a fall in its third-quarter
   earnings that were hit by larger expenditures on R&D and marketing
   .@negative"
```

Fig. 3.1: Vista dataset de entrenamiento para el modelo BERT.

Los investigadores consideraron relevante dividir el dataset resultante según el grado de coincidencia de los anotadores que trabajaron en su clasificación. Un primer subconjunto contenía todas las frases en las que al menos un 50% de los anotadores coincidieron al indicar la polaridad de la misma, dando como resultado un dataset de 4.840 frases. Se repitió la operación para otros niveles de conformidad obteniéndose los datasets que se indican en la Tabla 3.1.

Dataset	Conformidad (%)	Nº frases
Sentences_50Agree	50%	4.840
Sentences_66Agree	66%	4.220
Sentences_75Agree	75%	3.450
Sentences_AllAgree	100%	2.260

Tabla 3.1: Subconjuntos del dataset Financial PhraseBank.

Mediante esta diferenciación del dataset, se crearon subconjuntos con diferentes características, así encontramos el dataset 'Sentences_50Agree' el cual contiene un mayor número de datos pero de menor fiabilidad comparados con los incluidos en 'Sentences_AllAgree'. Permitiendo así, a criterio del investigador emplear el subconjunto más adecuado para su estudio.

3.6.- Herramientas para tratamiento de datos

Los datos obtenidos se deben preparar acorde a las necesidades del proyecto, de este modo se podrán obtener resultados más adecuados según los objetivos se que persigan en cada caso. Para todo ello es necesario hacer un preprocesamiento que es posible realizar haciendo uso de varias librerías especializadas en esta tarea.

3.6.1.- Pandas

Es un paquete desarrollado por Wes McKinney en 2008 mientras trabajaba en AQR Capital Management, es una de las librerías de código abierto más usadas por los investigadores [53]. Pandas presenta muchas ventajas en el trabajo con datos, como el manejo de ellos, mediante sus dos estructuras de datos principales, las Series, que soportan datos en una dimensión, y los DataFrame's, heredados del lenguaje de programación R, que soportan datos en dos dimensiones.

Sobre estas estructuras encontramos una gran flexibilidad en las funcionalidades, como la alineación automática y explícita de datos junto a etiquetas (labels), agrupación de datos o el manejo sencillo de los datos perdidos, representados como NaN. Estas funcionalidades facilitan y reducen el tiempo empleado en las principales tareas necesarias para el desarrollo de proyectos de ciencia de datos.

3.6.2.- Datetime

Para el tratamiento de datos de fechas y tiempos, se usará la librería Datetime, también del lenguaje Python [54]. Se empleará para las tareas relacionadas con la obtención de datos, tanto de noticias como del histórico de acciones, para el tratamiento de las ventanas de tiempo donde se buscan los diferentes datos.

3.7.- Visualización de datos con Plotly

La presentación de los datos es una tarea crucial dentro de los proyectos relacionados con datos, ya que nos indica, de una forma clara y concisa, los resultados obtenidos en el trabajo realizado.

Existen muchas librerías de Python enfocadas a este tipo de tareas, cada una con diferentes características, y se ha realizado una extensa investigación para encontrar aquella que se adapta mejor a los requerimientos del proyecto.

El paquete Plotly [55] ha resultado ser el más adecuado para el propósito de este proyecto. Se trata de una librería de Python para visualización de datos basada en web, por lo que los gráficos resultantes se muestran mediante una pestaña en un navegador, y no dentro del propio IDE, como hacen otras librerías. Además, Plotly permite cambiar gráficos dentro de una misma representación, lo cual proporciona una mayor libertad a la hora de crear visualizaciones de datos.

3.8.- Modelos NLP

Con el objeto de obtener un análisis de sentimientos, se hará uso de varios modelos NLP con los que se tratarán los datos. Gracias a este proceso, se obtienen las polaridades de las noticias financieras de nuestro dataset y se generarán resultados que es posible interpretar.

3.8.1.- FinBERT

Como representación del modelo BERT, se usará del modelo afinado FinBERT (Financial Sentiment Analysis with BERT) que se mencionó en el capítulo anterior. Este modelo es una mejora del modelo BERT, entrenado con datos pertenecientes al campo financiero y gracias a ello es capaz de interpretar mejor nuestro dataset. Es por eso que creemos que es el mejor modelo de la familia BERT para usar en este proyecto.

Para aplicar este modelo se usará la librería 'transformers' de Hugging Face, que permite de manera sencilla el uso de modelos pre-entrenados dentro de su catálogo web.

3.8.2.- VADER

Alternativamente al modelo BERT también se usará el modelo VADER. Como ya se ha comentado, este modelo contiene una estructura totalmente diferente y sus resultados servirán como objeto de comparación.

Para el uso de este modelo se usará la propia librería del modelo 'vaderSentiment', a pesar de encontrarse también en otras librerías como la NLTK (Natural Language ToolKit), vemos conveniente usarla de esta forma.

3.8.3.- Fine-tuned distilBERT

También se realizará un afinamiento personalizado del modelo distilBERT (visto en el capítulo anterior). Este modelo será entrenado con datos relacionados con el campo financiero pertenecientes al dataset Financial PhraseBank [40], el cual contiene noticias financieras junto a sus etiquetas del sentimiento. Para esta tarea se empleará la librería

Pandas [53], comentada anteriormente, junto a la librería NumPy [56] para el tratamiento de los datos.

En el tratamiento del modelo se hará uso de la librería transformers de Hugging Face [57], la cual ofrece funcionalidades específicas de entrenamiento de modelos, como un trainer, función encargada del entrenamiento, o AutoConfig, que facilita la configuración el modelo pre-entrenado que se desea afinar. Dentro de esta tarea también se usará la librería PyTorch [58], cuyas estructuras como tensores, matrices multidimensional que contienen elementos de un único tipo de datos, u operaciones como la función softmax, utilizada como capa final en nuestro modelo, hacen posible la realización de esta tarea. Por otra parte, la librería de scikit-learn [59] facilitará la realización de reportes, entre otras funciones.

Para la división del dataset en entrenamiento, validación y test usamos la librería Fast-ML [60], Esta permite realizar esta tarea con una simple línea de código, facilitando así en gran medida el trabajo.

Capítulo 4
Metodología y resultados

4.1.- Tareas y planificación

Para el correcto desarrollo del proyecto se llevarán a cabo una serie de tareas bien definidas, cuya planificación se muestra en la Fig. 4.1. Primero de todo, se realizarán las tareas de recopilación de datos, como la obtención de noticias financieras o el histórico de acciones bursátiles de una empresa. Estos datos se tratarán en la siguiente etapa, la cual consiste en el preprocesamiento de los datos.

En los siguientes pasos, se implementará el mejoramiento del modelo BERT utilizando el dataset de frases 'Sentences_50Agree' (ver apartado 3.5). Esta tarea es la que empleará más tiempo, ya que requiere de una investigación más profunda sobre el tema, buscando documentación o diferentes vías y métodos para su realización.

Al finalizar el entrenamiento, se dispondrá de los tres modelos objetos de estudio, siendo estos el modelo preajustado FinBERT, el léxico VADER, y nuestro modelo BERT

mejorado. Haciendo uso de estos modelos obtenemos los resultados relacionados con el sentimiento financiero de las noticias en forma de polaridades.

El siguiente paso consiste en desarrollar un sistema de visualización de datos, se estudiarán las diferentes opciones para, finalmente, elaborar gráficos que faciliten la comprensión de los resultados.

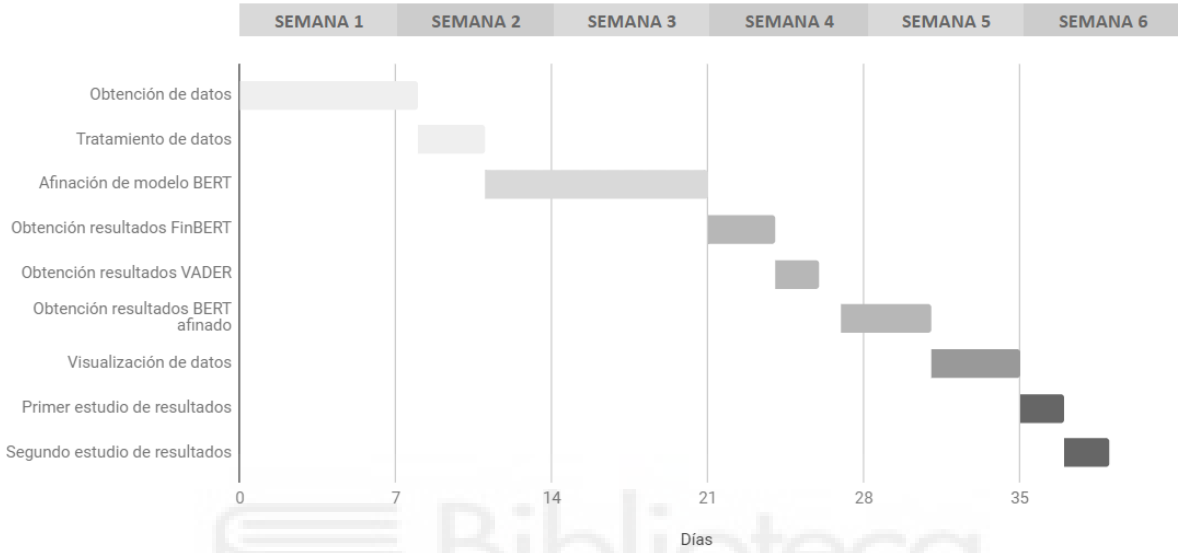

Fig. 4.1: Planificación de la metodología del proyecto.

Por último, se realizará un estudio analizando las relaciones entre la polaridad obtenida de las noticias sobre una empresa concreta y sus movimientos en los valores de mercado de sus acciones. Se harán dos tipos de estudios: el primero se centrará en los valores de polaridad más extremos proporcionados por el modelo FinBERT; el segundo estudio se aplicará a los tres modelos teniendo en cuenta todos los valores de polaridad.

4.2.- Obtención de datos

Para la obtención de noticias financieras se barajaron diversas posibilidades. Acorde a las técnicas de web scraping mencionadas en el capítulo 2, la primera de ellas consiste en el uso de librerías con funciones parsing como BeautifulSoup.

Es muy difícil extraer datos de noticias financieras utilizando este método porque la mayoría de los portales web cargan dinámicamente su contenido mediante JavaScript, lo que es un inconveniente para las técnicas de parsing, que requieren disponer desde el inicio todo el documento completo que necesita ser analizado.

Ante la ineficacia del método anterior, se intentó utilizar el framework de automatización de navegador web Selenium, pero se descubrió una barrera de seguridad en el acceso al

sitio web objetivo SeekingAlpha. Al emplear un navegador automatizado sobre esta página, aparece una ventana emergente con indicaciones de error, por lo que no es posible usar este método para acceder a la información que ofrecen estos portales, como, por otra parte, también se indica en los términos y condiciones de uso de estas páginas [61].

Se descubrió ADVFN, una página web alternativa con grandes bancos de datos en forma de noticias, pero los datos que proporciona son de un nivel técnico muy alto, lo que dificulta en gran medida el análisis de los modelos. Algunas herramientas como las web API, quedaron fuera del estudio porque requieren suscripciones con un coste económico para su uso.

Tras estudiar del caso y realizar varias pruebas, finalmente se optó por utilizar la herramienta Pygooglenews, la cual obtiene titulares de noticias mediante el raspado de RSS (siglas en inglés de "Sindicación Realmente Simple") de Google News [62]. Esta herramienta se presentó anteriormente en el capítulo 3 (apartado 3.4.1).

Los datos de noticias financieras requeridos para el proyecto incluirían el propio título de la noticia, cuerpo y fecha de publicación. Mediante el uso de la herramienta Pygooglenews no es posible obtener el cuerpo de la noticia, pero sí el título, el medio comunicativo donde se publicó, y la fecha. Esto limitará en parte los resultados, ya que no se podrá analizar el texto completo de las noticias, sólamente la información ofrecida en sus títulos, donde los lectores prestan su primera atención.

La librería utilizada permite insertar una sentencia como entrada de datos, que puede contener operadores lógicos como OR o AND para producir resultados más completos. Así, nuestra entrada de datos será del tipo "[Nombre Compañia] OR [Ticker]", generando una búsqueda de notícias relacionadas con el nombre de la empresa o su ticker bursátil, también llamado etiqueta de cotización, con el que opera en bolsa. En este proyecto, utilizaremos la entrada de datos "Microsoft OR MSFT", de modo que Microsoft es la empresa objetivo sobre la cual queremos obtener noticias relacionadas y 'MSFT' es su símbolo de cotización, identificador con el que opera en la bolsa de valores estadounidense NASDAQ [63]. Las noticias obtenidas proceden de diversos medios de comunicación y se encuentran disponibles en Google News, al ser recopiladas de este portal.

Otra funcionalidad interesante de la herramienta es su capacidad para filtrar las noticias en función de una ventana temporal especificada. Para realizar una comparación entre las noticias y las acciones financieras de la empresa se obtendrán noticias publicadas durante todo un año natural. En este trabajo nos centraremos en los periodos de 2020 y 2021, los cuales serán estudiados de forma separada. Se eligieron teniendo en cuenta el importante número de acontecimientos que se produjeron en ellos, caracterizados por la aparición de la pandemia de COVID-19. Estos acontecimientos podrían haber dado lugar a cambios

significativos en las acciones y, por tanto, a más oportunidades de descubrir resultados interesantes en sus estudios.

Al realizar una búsqueda se obtienen sólamente 100 titulares de noticias, siendo el límite máximo que permite la herramienta en una única petición. Esta cantidad de datos no es suficiente para el desarrollo del proyecto, recordemos que estamos tratando de recolectar noticias pertenecientes a todos los días del año. Además, estas noticias están repartidas desigualmente a lo largo del periodo establecido, habiendo una gran cantidad de días donde no hay ninguna noticia relativa a la empresa Microsoft.

Para abordar estas cuestiones se implementa la función "get_news" (ver figura 4.2) que consiste en un bucle que realiza una petición diaria de noticias, desde el 1 de enero de 2020 hasta el 1 de enero de 2021, que cumplan con la condición establecida en la cadena "search".

```
1  import pandas as pd
2  from pygooglenews import GoogleNews
3  import datetime
4
5  gn = GoogleNews()
6
7  def get_news(search):
8      stories = []
9
10     start_date = datetime.date(2020,1,1)
11     end_date = datetime.date(2021,1,1)
12     delta = datetime.timedelta(days=30)
13     date_list = pd.date_range(start_date, end_date).tolist()
14
15     for date in date_list[:-1:30]:
16         result = gn.search(search, from_=date.strftime('%Y-%m-%d'),
17                            to_=(date+delta).strftime('%Y-%m-%d'))
18         newsitem = result['entries']
19
20         for item in newsitem:
21             story = {
22                 'title':item.title,
23                 'published':datetime.datetime.strptime(item.published, '%a,
24                                                        %d %b %Y %X GMT')
25             }
26             stories.append(story)
27     return stories
28
29 df = pd.DataFrame(get_news('Microsoft OR MSFT'))
```

Fig. 4.2: Uso de "Pygooglenews" para la obtención de noticias en el año 2020.

Obsérvese la línea 29 del algoritmo de la figura 4.2., finalmente se obtiene un dataframe (df) devuelto por la función "get_news" a la que se llama con la cadena de búsqueda "Microsoft OR MSFT" (serach). Con la ejecución de este código se obtienen

aproximadamente unos 1200 registros de noticias publicadas a lo largo del año 2020, en formato título de la notícia y fecha de publicación (Fig. 4.3). En el estudio del proyecto también trabajaremos con noticias del año 2021, las cuales se obtienen de idéntica forma, modificando convenientemente las líneas 10 y 11 del algoritmo de la figura 4.2.

Index	title	published
0	Microsoft makes 'carbon negative' pledge - bbc.com	2020-01-16 08:00:00
1	Microsoft pledges to be 'carbon negative' by 2030 - The Guardian	2020-01-16 08:00:00
2	Microsoft's new Edge Chromium browser launches on Windows and ... - The Verge	2020-01-15 08:00:00
3	NSA found a dangerous Microsoft software…	2020-01-14 08:00:00
4	Why Microsoft embraced Chromium, and what the new Edge says ... - GeekWire	2020-01-15 08:00:00
5	Microsoft Azure Flaws Could Have Let Hac…	2020-01-30 08:00:00

Fig. 4.3: DataFrame de los datos de noticias financieras obtenidos por Pygooglenews.

Para la obtención de datos financieros sobre la cotización en bolsa se utiliza la librería yfinance vista en el capítulo 3 (apartado 3.4.2). Mediante su empleo se pueden obtener datos diarios pertenecientes a cualquier empresa en un periodo de tiempo determinado, en nuestro caso, se necesitan los correspondientes a la empresa Microsoft en los dos años naturales 2020 y 2021 (Fig. 4.4).

```
1 import yfinance as yf
2
3 df_price = yf.download('MSFT', start = '2020-01-01', end = '2021-01-05')
```

Fig. 4.4: Uso de "yfinance" para obtener datos de cotización de Microsoft en 2020.

Index	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-02 00:00:00	158.78	160.73	158.33	160.62	155.422	22622100
1	2020-01-03 00:00:00	158.32	159.95	158.06	158.62	153.487	21116200
2	2020-01-06 00:00:00	157.08	159.1	156.51	159.03	153.884	20813700
3	2020-01-07 00:00:00	159.32	159.67	157.32	157.58	152.48	21634100
4	2020-01-08 00:00:00	158.93	160.8	157.95	160.09	154.909	27746500
5	2020-01-09 00:00:00	161.84	162.22	161.03	162.09	156.844	21385000

Fig. 4.5: DataFrame de los datos financieros obtenidos por yfinance.

Cada registro de información sobre cotización diaria incluye cuatro valores: el valor de apertura (cotización al iniciar la jornada), los valores máximos y mínimos (cotizaciones máxima y mínima a lo largo de la jornada), y el valor de cierre (cotización al finalizar la jornada), adicionalmente, cada registro también incluye el volúmen (cantidad de acciones que han cambiado de intermediarios durante la jornada) y el valor de cierre ajustado (ajustamiento posterior del valor de cierre que considera acciones corporativas).

La Fig. 4.5 muestra un extracto de esta información. Si bien para el presente estudio solo se emplearán los valores de cierre, el resto de valores pueden resultar de interés para estudios futuros.

4.3.- Tratamiento de datos

El preprocesamiento de datos es una tarea esencial en la ciencia de datos, y más aún para entrenar modelos de Deep Learning, ya que estos requieren de una adecuada preparación para su correcta entrada. Por ello, se realizarán ciertas modificaciones sobre los datos obtenidos con el objetivo de adecuar su calidad y formato a los métodos que posteriormente utilizaremos.

Teniendo en cuenta los titulares de las noticias obtenidas, resulta útil separar los nombres de las agencias de publicación que figuran en los títulos de cada artículo. De este modo, se amplía el conjunto de datos añadiendo una nueva columna para las fuentes de las noticias en el DataFrame. Como resultado, es creada nueva variable de interés aunque no será tratada en este proyecto. Además, esta modificación mejorará nuestros resultados, ya que las fuentes de las noticias, en principio, carecen de connotaciones que puedan ser útiles en la búsqueda del sentimiento por parte de los modelos. En consecuencia, las fuentes deben separarse del input de titulares que será analizado, con el fin evitar añadir trabajo adicional a los modelos.

Muchos de los titulares obtenidos contienen al final los caracteres "..." añadidos a consecuencia del límite de caracteres impuesto por la herramienta Pygooglenews. La herramienta ignora la información restante en los títulos largos de más de 60 caracteres y la sustituye por puntos suspensivos. El único propósito de tener estos caracteres era indicar que la información obtenida estaba incompleta y no incluía toda la información original. Estos caracteres se eliminan de los datos porque son irrelevantes al no poder obtener las palabras originales que faltan (Fig. 4.6).

Index	title	published	source
0	Microsoft makes 'carbon negative' pledge	2020-01-16 08:00:00	bbc.com
1	Microsoft pledges to be 'carbon negative' by 2030	2020-01-16 08:00:00	The Guardian
2	Microsoft's new Edge Chromium browser launches on Windows and	2020-01-15 08:00:00	The Verge
3	NSA found a dangerous Microsoft software flaw and alerted the firm	2020-01-14 08:00:00	The Washington Post
4	Why Microsoft embraced Chromium, and what the new Edge says	2020-01-15 08:00:00	GeekWire
5	Microsoft Azure Flaws Could Have Let Hackers Take Over Cloud	2020-01-30 08:00:00	The Hacker News

Fig. 4.6: Datos tratados tras la limpieza y creación de la columna 'Source'.

Debido a la capacidad del entorno de programación Spyder donde se ejecuta el código, se descubrieron problemas con la introducción de datos en los modelos. Si se introducían a los modelos todos los datos disponibles (unas 1200 noticias aproximadamente por periodo), el entorno producía errores de rendimiento resultando imposible continuar con el proceso. Debido a esto, dividiremos la entrada de titulares en cuatro partes iguales y se realizarán los análisis de sentimiento en varias ejecuciones, una para cada trozo del dataset. Al finalizar todas las ejecuciones se agruparán nuevamente en un DataFrame de resultados. Este proceso funciona exactamente igual que si la entrada no estuviera dividida.

Otra tarea de preprocesamiento necesaria ha consistido en modificar los formatos de fecha de las noticias para que coincidieran con los valores de cotización y así poder compararlos. Los primeros vienen en formato fecha-hora (aaaa-mm-dd hh:mm:ss), sin embargo las cotizaciones, dado que se han descargado datos de temporalidad diaria, no incluyen la hora (aaaa-mm-dd). En consecuencia, las horas, los minutos y los segundos se eliminan del dataset de noticias. Esto facilita la comparación de valores entre los dos conjuntos de datos.

Al obtener los datos, a veces se han obtenido varias noticias pertenecientes al mismo día, lo que imposibilitaba una clara visualización gráfica de los datos. Como consecuencia, las noticias se filtran de tal forma que permanece aquella que obtuvo una mayor puntuación de polaridad en cada uno de esos días. Estas noticias son las que, según los modelos, producirían un mayor impacto en los lectores.

4.4.- Afinación de modelo BERT

Para conseguir mayor variabilidad en los resultados, se realizará una afinación del modelo pre entrenado BERT. Debido al peso y la consecuente dificultad de trabajo del modelo, el proceso se realizará sobre el modelo variante DistilBERT, que tiene capacidades similares con tamaño reducido el cual se introdujo en el capítulo 2.

4.4.1.- Fine-tuning de modelo

El concepto de “fine-tuning” (o “ajuste fino”), acuñado por Brandon Carter en 1974 [64], ha sido objeto de un gran número de estudios a lo largo del tiempo. Este se ha relacionado con el campo de la física, la psicología y recientemente, la computación. En física teórica, el ajuste fino es el proceso donde los parámetros de un modelo deben ajustarse con gran precisión para que encajen con determinadas observaciones. Algunos físicos utilizan este concepto para describir circunstancias donde, en algunos casos, parece que nuestro universo fuera expresamente creado para ello, como en la aparición de la vida [65]. Este universo se conoce como “fine-tuned universe” (“universo bien ajustado”) y se puede encontrar también en discusiones filosóficas [66]. Siendo uno de los temas donde la ciencia y la filosofía llegan a coexistir.

Siguiendo este concepto, surgió el fine-tuning en modelos de machine learning [67], donde arquitecturas predefinidas se reutilizan y modifican para obtener mejores resultados en campos más especializados. En nuestro caso, la arquitectura pertenece al modelo BERT a través del modelo más ligero DistilBERT, ambos comentados en el apartado 2.3.

Son muchas las ventajas de utilizar este método de entrenamiento, siendo una de las más significativas el ahorro tanto de tiempo y recursos computacionales. Esto se debe a que se parte de un modelo previamente pre-entrenado, y no de cero para obtener los resultados deseados.

Además, empleando esta técnica podemos obtener mejores resultados en un campo específico, sobre todo si el modelo se entrena con datos de ese campo. Este proyecto se centra especialmente en las notícias financieras, ámbito donde el modelo DistilBERT no fue adaptado. De este modo, el modelo se enfrentará por primera vez a los conceptos característicos del mundo financiero.

Para la implementación de esta técnica a alto nivel, es decir, sin profundizar demasiado en la modificación de los componentes internos del modelo, es necesaria una preparación y cargado de los datos, para posteriormente realizar correctamente el entrenamiento, y finalmente la puesta a punto para su utilización [68].

4.4.2.- Preparación de los datos

Como se comentó al inicio del capítulo, para el entrenamiento del modelo se ha seleccionado el subconjunto "*Sentences_50Agree*" del dataset Financial Phrasebank, definido en el capítulo 3 (apartado 3.5). Esta selección contiene el mayor número de datos entre las disponibles, siendo esta la razón de su selección frente al resto. La finalidad del entrenamiento es la adaptación del modelo al léxico perteneciente al campo financiero, por ello se prioriza la cantidad de datos frente a la precisión en la polarización de ellos.

	label	text
0	neutral	Technopolis plans to develop in stages an area...
1	negative	The international electronic industry company ...
2	positive	With the new production plant the company woul...
3	positive	According to the company 's updated strategy f...
4	positive	FINANCING OF ASPOCOMP 'S GROWTH Aspocomp is ag...

	label	text
0	1	Technopolis plans to develop in stages an area...
1	0	The international electronic industry company ...
2	2	With the new production plant the company woul...
3	2	According to the company 's updated strategy f...
4	2	FINANCING OF ASPOCOMP 'S GROWTH Aspocomp is ag...

Fig. 4.7: Transformación valores de campo "label" ("label encoding").

Para trabajar con estos datos de entrenamiento, previamente hay que calcular su polaridad o sentimiento. La figura 4.7 muestra un extracto de cómo queda etiquetado cada titular como "negativo", "neutral" o "positivo", asignando los valores enteros 0, 1 y 2,

respectivamente a cada una de dichas etiquetas, procedimiento denominado “label encoding” [69] (o “codificación de etiqueta”) dentro del campo del Data Science.

Durante el entrenamiento de un modelo se utiliza una gran cantidad de datos, pero no todos tienen las mismas funciones dentro del proceso y son suministrados en diferentes etapas. Por ello, el conjunto de datos se divide en tres bloques para cada una de las diferentes etapas del aprendizaje: entrenamiento, prueba y validación, con el fin de llevar a cabo un correcto entrenamiento del modelo. Para hacer esta división se va a seguir la regla del 80/10/10: 80% del total el tamaño del conjunto de datos para la fase de entrenamiento, que consiste en crear y refinar las reglas del modelo ajustando sus parámetros; el 10% de los datos se destinará para el conjunto de prueba, etapa en al que se evalúa la eficacia del entrenamiento o la precisión del modelo tras el entrenamiento; y el último 10% serán los datos de validación, empleados para validar el modelo mientras se entrena, proporcionando una primera toma de contacto a datos desconocidos [70].

Todos estos datos deben convertirse a un formato de entrada entendible por el modelo. Para poder entender las sentencias y darle significado a sus palabras, estas deberán seguir un formato más comprensible. Por ello hay que “tokenizarlos”, es decir, dividir las palabras dentro de una misma frase y almacenarlas una a una en nuevos registros compuestos de “tokens”. De este modo, los modelos son capaces de procesar mejor las oraciones e interpretar sus contextos. Para realizar esta tokenización se utiliza un código como el que se muestra en la figura 4.8.

```
1 import pandas as pd
2 from transformers import AutoTokenizer, AutoModelForSequenceClassification
3 import torch
4
5 # dado un dataframe 'df' como el de la figura 4.6
6 headlines = df["title"].tolist()
7 tokenizer = AutoTokenizer.from_pretrained("ProsusAI/finbert")
8 inputs = tokenizer(headlines, padding=True, truncation=True,)
```

Fig. 4.8: Código para tokenizar titulares de noticias.

La variable ‘inputs’ (línea 8) contiene los titulares tokenizados con los que alimentar el proceso de entrenamiento del modelo BERT., incluyendo los tokens de relleno característicos de este modelo CLS y SEP (ver apartado 2.3.1, figura 2.3) [71].

4.4.3.- Entrenamiento del modelo

Se va a entrenar un modelo denominado DistilBERT (BERT Destilado), basado en BERT, que es un modelo reducido de este último. Antes de lanzar este entrenamiento se requiere especificar una serie de características de configuración que se representan en un diccionario de Python (Fig. 4.9).

```
DistilBertConfig {
  "_name_or_path": "distilbert-base-uncased",
  "activation": "gelu",
  "architectures": ["DistilBertForMaskedLM"],
  "attention_dropout": 0.1,
  "dim": 768,
  "dropout": 0.1,
  "hidden_dim": 3072,

  "id2label": {
    "0": "negative",
    "1": "neutral",
    "2": "positive"
  },

  "initializer_range": 0.02,

  "label2id": {
    "negative": 0,
    "neutral": 1,
    "positive": 2
  },

  "max_position_embeddings": 512,
  "model_type": "distilbert",
  "n_heads": 12,
  "n_layers": 6,
  "pad_token_id": 0,
  "qa_dropout": 0.1,
  "seq_classif_dropout": 0.2,
  "sinusoidal_pos_embds": false,
  "tie_weights_": true,
  "transformers_version": "4.34.0.dev0",
  "vocab_size": 30522
}
```

Fig. 4.9: Configuración del modelo DistilBERT.

La mayoría de los atributos de la configuración del modelo vienen por defecto en el mismo, para la realización de este trabajo se han incluido los atributos “id2label” (líneas 10 a 14) y “label2id” (líneas 18 a 22). Tras la configuración se entrena el modelo como indica la figura 4.10.

```
model = AutoModelForSequenceClassification.from_config(config)

train_dataset = DataLoader(train_texts, train_labels)

val_dataset = DataLoader(val_texts, val_labels)

training_args = TrainingArguments(
    output_dir='/results',
    num_train_epochs=10,
    per_device_train_batch_size=64,
    per_device_eval_batch_size=64,
    warmup_steps=500,
    weight_decay=0.05,
    report_to='none',
    evaluation_strategy='steps',
    logging_dir='/logs',
    logging_steps=50)
```

```
18
19 trainer = Trainer(
20     model=model,
21     args=training_args,
22     train_dataset=train_dataset,
23     eval_dataset=val_dataset,
24     compute_metrics=compute_metrics
25 )
26
27 trainer.train()
```

Fig. 4.10: Entrenamiento del modelo DistilBERT.

En el código de la figura 4.10, la variable "model" (línea 1) es el modelo preentrenado que se quiere afinar con nuevos datos de noticias financieras. Los conjuntos de entrenamiento y validación (líneas 3 y 5) se construyen a partir de conjuntos tokenizados de titulares junto con sus etiquetas, esta información se ha obtenido del conjunto de datos "Sentences_50Agree" descrito anteriormente (Fig. 4.7). A continuación se indican los parámetros de entrenamiento (líneas 7 a 17), los más relevantes son:

- La dirección donde serán guardados los resultados.

- El número de "epochs" (épocas), ciclos completos que el modelo recorrerá el conjunto de datos de entrenamiento.

- Los pasos o steps de calentamiento o "warm up", necesarios para paliar un problema al inicio del entrenamiento. Debido al rápido cambio de los parámetros de la red, no se cumple la condición necesaria para que funcione el escalado lineal, por ello durante el tiempo de "warm up" se utiliza una tasa de aprendizaje más baja, permitiendo un correcto funcionamiento. Una vez completados estos pasos, la tasa de aprendizaje vuelve a su valor original [72].

- La evaluación, indicando que será por "steps" o (español).

- Los pasos de "logging" o registro, que son necesarios para el empleo de una buena práctica en el entrenamiento del modelo; sin embargo, no condicionan el funcionamiento del entrenamiento sino que son meramente informativos. Son los responsables de informar al desarrollador sobre el estado del proceso, facilitando la detección de posibles errores [72].

Tras especificar los argumentos se crea el objeto "trainer" (líneas 19 a 25) que contiene el modelo a refinar, los argumentos y los conjuntos de datos de entrenamiento y validación (el parámetro "compute_metrics" se explica a continuación). Finalmente se lanza el modelo para su entrenamiento (línea 27), al iniciarse, se muestra una barra de progreso dinámica junto con los datos de control del proceso la cual será actualizada durante todo su transcurso (Fig. 4.11).

[610/610 07:32, Epoch 10/10]

Step	Training Loss	Validation Loss	F1	Accuracy	Precision	Recall
50	0.991400	0.946254	0.242663	0.572314	0.190771	0.333333
100	0.913400	0.913807	0.242663	0.572314	0.190771	0.333333
150	0.840200	0.836580	0.391777	0.621901	0.381239	0.420009
200	0.793900	0.823302	0.505069	0.657025	0.574026	0.504288
250	0.757200	0.784852	0.455307	0.671488	0.753315	0.472626
300	0.629300	0.788279	0.539625	0.681818	0.611116	0.536907
350	0.512800	0.858791	0.527262	0.659091	0.568760	0.551357
400	0.457700	0.884067	0.655766	0.719008	0.668158	0.646273
450	0.348500	0.885644	0.669615	0.727273	0.694227	0.652593
500	0.242200	0.942223	0.647411	0.725207	0.699552	0.621306
550	0.192500	1.028605	0.652194	0.723140	0.680940	0.665499
600	0.115300	1.001153	0.679816	0.735537	0.715440	0.658123

Fig. 4.11: Resultado del entrenamiento del modelo.

Para medir los resultados del entrenamiento se utilizan ciertas métricas de rendimiento que se aplican automáticamente al modelo cuando el parámetro "computer_metrics" se establece tal y como se mostró en el código de la figura 4.10 (línea 24). Estos indicadores ayudarán a determinar si el proceso se ha llevado a cabo correctamente o qué posibles problemas pueden haber surgido. Las métricas empleadas son:

- **"Accuracy"** o **exactitud**. Mide el porcentaje de casos donde el modelo ha acertado.
- **Precisión**. Estudia la calidad del modelo en tareas de clasificación.
- **"Recall"** o **sensibilidad**. Estudia la cantidad del modelo en tareas de clasificación.
- **"F1-score"** o **valor-F**. Combina las medidas de precisión y recall en un sólo valor.
- **Pérdida de entrenamiento**. Evalúa el error del modelo en el conjunto de entrenamiento.
- **Pérdida de validación**. Evalúa el error del modelo en el conjunto de validación.

El resultado de cada una de estas métricas también se puede ver en la figura 4.11.

Analizando la estructura interna del modelo (Fig. 4.12) se pueden observar los componentes que lo forman. El primero de ellos es la capa inicial de incrustación o embeddings encargada de mapear la información de entrada, disminuyendo así su nivel de complejidad y permitiendo a la red aprender más sobre la relación entre las entradas, lo que se traduce en un procesamiento más eficaz de los datos [73].

También es posible observar la estructura del transformer en la sección siguiente, está formado por seis capas, cada una de las cuales se encarga de aplicar transformaciones lineales a cada palabra de la secuencia de entrada, aplicando diferentes pesos en cada capa [74]. A continuación está la capa previa al clasificador, requerida por la ausencia de pooler (elemento del modelo BERT, eliminado de DistilBERT, ver apartado 2.3.1.2).

```
DistilBertForSequenceClassification(
 (distilbert): DistilBertModel(
  (embeddings): Embeddings(
   (word_embeddings): Embedding(30522, 768, padding_idx=0)
   (position_embeddings): Embedding(512, 768)
   (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
   (dropout): Dropout(p=0.1, inplace=False)
  )
  (transformer): Transformer(
   (layer): ModuleList(
     (0-5): 6 x TransformerBlock(
      (attention): MultiHeadSelfAttention(
       (dropout): Dropout(p=0.1, inplace=False)
       (q_lin): Linear(in_features=768, out_features=768, bias=True)
       (k_lin): Linear(in_features=768, out_features=768, bias=True)
       (v_lin): Linear(in_features=768, out_features=768, bias=True)
       (out_lin): Linear(in_features=768, out_features=768, bias=True)
      )
      (sa_layer_norm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (ffn): FFN(
       (dropout): Dropout(p=0.1, inplace=False)
       (lin1): Linear(in_features=768, out_features=3072, bias=True)
       (lin2): Linear(in_features=3072, out_features=768, bias=True)
       (activation): GELUActivation()
      )
      (output_layer_norm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
     )
    )
   )
 )
 (pre_classifier): Linear(in_features=768, out_features=768, bias=True)
 (classifier): Linear(in_features=768, out_features=3, bias=True)
 (dropout): Dropout(p=0.2, inplace=False)
)
```

Fig. 4.12: Estructura interna del modelo DistilBERT.

En la penúltima línea se observa cómo procede el clasificador, reduciendo los 768 atributos obtenidos a sólo tres, que corresponden a las categorías positiva, negativa y neutra del sentimiento. Y por último aparece la capa de “dropout” (o abandono), la cual ayuda a prevenir el “overfitting” o “sobreajuste” del modelo durante su entrenamiento [75]. Este fenómeno se produce al entrenar modelos excesivamente, hasta el punto de que aprenden incluso los errores del conjunto de datos, así como reconocen y evitan problemas en lugar de aprender de ellos, conduciendo a resultados ideales para la muestra analizada, pero poco fiables para predicciones futuras [76].

Tras examinar los resultados de las métricas de pérdidas (Fig. 4.13), se observa como la pérdida de entrenamiento presenta una correcta evolución que disminuye con el transcurso

de los pasos, llegando a un valor final de 0.11. . En cambio, la pérdida de validación muestra malos resultados, al principio disminuye su valor hasta casi la mitad del entrenamiento, donde comienza a aumentar dicho valor. Este comportamiento no es el ideal ya que, esta pérdida se calcula sumando los errores de cada ejemplo del conjunto de validación, en un desarrollo adecuado, la pérdida debería ser menor en las etapas finales. Esto indica que en el entrenamiento se ha producido un mal rendimiento durante en el periodo de validación.

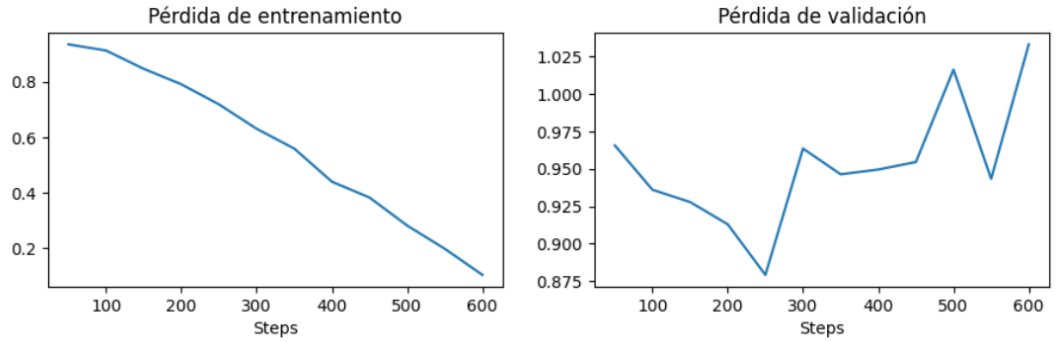

Fig. 4.13: Métricas de pérdidas obtenidas durante el entrenamiento.

Analizando el resto de métricas (Fig. 4.14) se observan valores normales en la métrica de "accuracy" o exactitud, siendo el valor más alto en el último "step" situándose en 0.77 puntos, es decir, el 77% de los datos del banco de pruebas fueron predichos correctamente por el modelo.

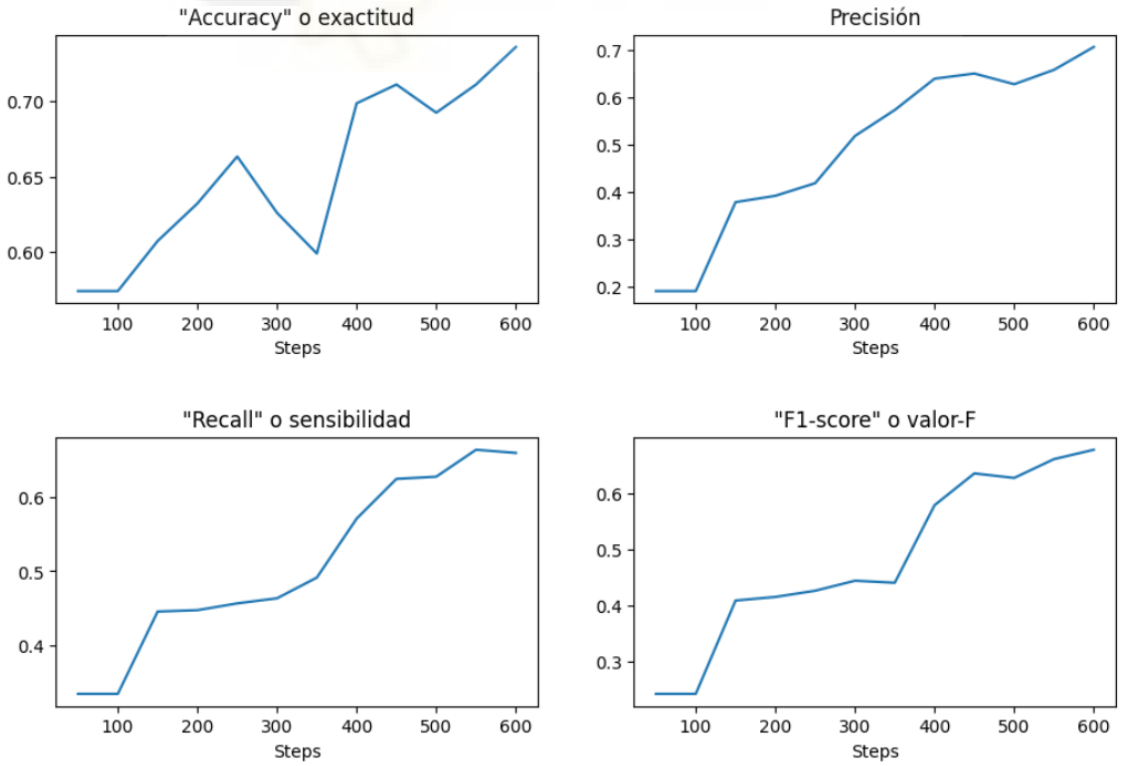

Fig. 4.14: Resto de métricas obtenidas durante el entrenamiento.

Se observaron resultados similares en las demás métricas; el “recall” o sensibilidad obtuvo un valor de 0.65 tras una evolución positiva durante todo el proceso, al igual que la función F1-score o valor-F, que fue de 0.67. En relación a la precisión, su valor máximo se alcanzó a mitad del entrenamiento, representando así una evolución no positiva de la métrica de precisión durante todo el proceso. En este punto máximo, alcanza un valor de 0.75, superior al obtenido al final del entrenamiento, siendo de 0.73.

4.5.- Uso de modelos

En el empleo de los modelos se realizan rutas de trabajo muy similares, aunque con algunas diferencias debido a sus distintas naturalezas y la variedad de métodos para el tratamiento de datos.

4.5.1.- FinBERT

Todos los atributos de las noticias deben mantenerse en listas para que el modelo FinBERT se ejecute correctamente. El proceso se lleva a cabo mediante un bucle iterativo que recorre todos las filas del conjunto de datos y almacena los atributos de cada registro en su lista correspondiente (Fig. 4.15, líneas 7 a 10).

```
1  def perform_sentiment_analysis_FinBERT(df):
2    headlines = []
3    dates = []
4    sources = []
5
6    for index, row in df.iterrows():
7      headlines.append(row["title"])
8      dates.append(row["published"])
9      sources.append(row["source"])
10
11   tokenizer = AutoTokenizer.from_pretrained("ProsusAI/finbert")
12   model = AutoModelForSequenceClassification.from_pretrained("ProsusAI/finbert")
13   inputs = tokenizer(headlines, padding=True, truncation=True, return_tensors='pt')
14
15   outputs = model(** inputs)
16   predictions = torch.nn.functional.softmax(outputs.logits, dim=-1)
17   positive = predictions[:,0].tolist()
18   negative = predictions[:,1].tolist()
19   neutral = predictions[:,2].tolist()
20   table = {"headline": headlines,
21            "positive": positive,
22            "negative": negative,
23            "neutral": neutral,
24            "source" : sources,
25            "date": dates}
26   results_df = pd.DataFrame(table, columns=["headline", "positive", "negative",
27                              "neutral", "source", "date"])
28   return results_df
```

Fig. 4.15: Obtención de resultados con modelo FinBERT.

Tras cargar tanto el tokenizador como el modelo (librería Transformers de Hugging Face), se creará el conjunto de inputs, ejecutando el tokenizador sobre los títulos del dataset (líneas 12, 13 y 14). A continuación, se crean los outputs ejecutando el modelo; la biblioteca Hugging Face permite llevar a cabo este proceso de forma coordinada en todo el conjunto de datos, obteniendo todos los resultados en una única ejecución (línea 16).

Finalmente se les aplica la función de salida "softmax", que genera las predicciones del modelo basándose en probabilidades. Estos datos se agruparán en el DataFrame de resultados junto con los títulos, las fechas de publicación y las fuentes de las noticias pertinentes (líneas 17 y siguientes).

4.5.2.- VADER

Los resultados con el modelo VADER se obtienen de forma similar al modelo anterior. Con un bucle (Fig. 4.16, línea 11) se obtienen las listas de los datos (líneas 12, 13, 14) con los que se ejecuta el modelo para obtener la polaridad de los títulos (línea 15).

```
1  def perform_sentiment_analysis_VADER(df):
2    results_df = pd.DataFrame([], columns=['date','source', 'title',
3                                           'positive', 'neutral', 'negative'])
4    sentiment_dict = []
5
6    sid_obj = SentimentIntensityAnalyzer()
7
8    for index, row in df.iterrows():
9      title = row["title"]
10     date = row["published"]
11     source = row["source"]
12     sentiment_dict = sid_obj.polarity_scores(title)
13
14     results_df = results_df.append({'title' : title,
15                                     'date' : date,
16                                     'source': source,
17                                     'positive' : sentiment_dict['pos'],
18                                     'neutral' : sentiment_dict['neu'],
19                                     'negative' : sentiment_dict['neg']
20     },ignore_index = True)
21   return results_df
```

Fig. 4.16: Obtención de resultados con modelo VADER.

Como resultado, después de cada iteración se obtendrá un nuevo registro, que se añadirá al DataFrame de resultados (líneas 14 a 20). De este modo, al concluir la última iteración del bucle, el dataset de resultados estará totalmente terminado.

4.5.3.- Fine-tuned DistilBERT

A diferencia de los otros dos modelos que se ejecutan en el IDE Spyder, la ejecución de nuestro propio modelo afinado BERT se realiza mediante un script de Python en el entorno Google Colaboratory (véase el apartado 3.3).

```
 1 headlines = []
 2 dates = []
 3 sources = []
 4 pos = []
 5 neg = []
 6 neu = []
 7
 8 for index, row in df_news.iterrows():
 9   title = row["title"]
10   single_sentence_probas = sentiment_model.predict_proba(title)
11
12   headlines.append(title)
13   dates.append(row["published"])
14   sources.append(row["source"])
15   pos.append(single_sentence_probas[0][2])
16   neg.append(single_sentence_probas[0][0])
17   neu.append(single_sentence_probas[0][1])
18
19 table = {"headline": headlines,
20           "positive": pos,
21           "neutral": neu,
22           "negative": neg,
23           "source" : sources,
24           "date": dates}
25 results_df = pd.DataFrame(table, columns=["headline", "positive", "neutral",
26                           "negative", "source", "date"])
```

Fig. 4.17: Obtención de resultados con modelo fine-tuned DistilBERT.

La estructura del proceso (ver Fig. 4.17) es idéntica al modelo VADER, utilizando un bucle for y llamando a la función del modelo (línea 10), que genera los valores de sentimientos de cada título en las sucesivas iteraciones.

4.6.- Visualización de datos

Nuestro objetivo principal con respecto a la visualización de datos es crear gráficos que combinen la información del análisis de sentimientos con la evolución de los datos históricos del precio de acciones. Esta combinación busca encontrar coincidencias entre los sentimientos positivos y negativos de los diferentes titulares, y las subidas o bajadas del precio de las acciones en las mismas fechas.

Para cumplir este hito se empleó el método “subplots” de la librería “plotly” que permite combinar varias gráficas en un mismo lienzo, así como incluir una capa “slider” que permite deslizar y hacer zoom en la gráfica general.

De los dos tipos de gráficas que interesa generar, el primero representa en un diagrama de barras los valores de polaridad predichos por cada modelo y se disponen ordenados cronológicamente según la fecha de cada titular. Las noticias caracterizadas por valores positivos (> 0) se representarán en color verde, las de polaridad negativa (< 0) se representarán en color rojo, y cuando la noticia es neutra (= 0) o no hay noticias sobre la empresa analizada (en nuestro caso Microsoft) no se dibujará ninguna línea (Fig. 4.18).

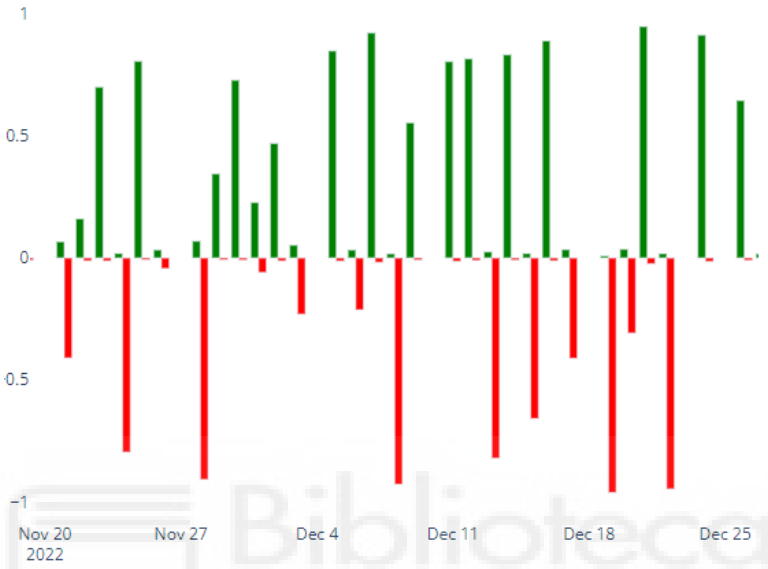

Fig. 4.18: Representación de gráfico de barras.

La segunda gráfica representa los datos históricos del precio de las acciones de la empresa mediante un gráfico de velas japonesas (o "candlestick"). Esta representación es especializada en este tipo de datos, permitiendo representar visualmente atributos financieros como el precio de apertura, valores máximo y mínimo, y el cierre (Fig. 4.19).

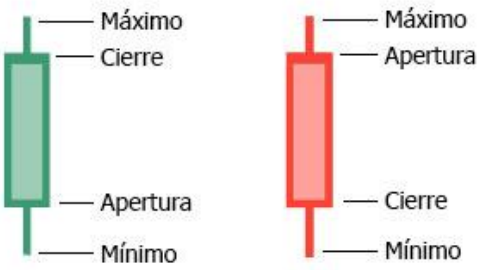

Fig. 4.19: Estructura de las velas japonesas.

En la figura 4.20 se muestra un ejemplo de un gráfico de velas japonesas generado por la librería "plotly" con datos históricos de cotización de la empresa Microsoft. Nótese que el gráfico es interactivo y al poner el ratón sobre una de las velas aparece un cuadro con la información detallada de dicha vela.

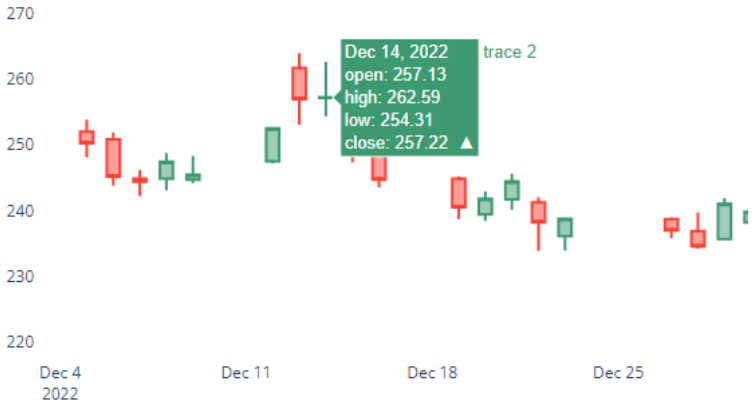

Fig. 4.20: Representación de atributos ofrecidos por el gráfico de velas japonesas.

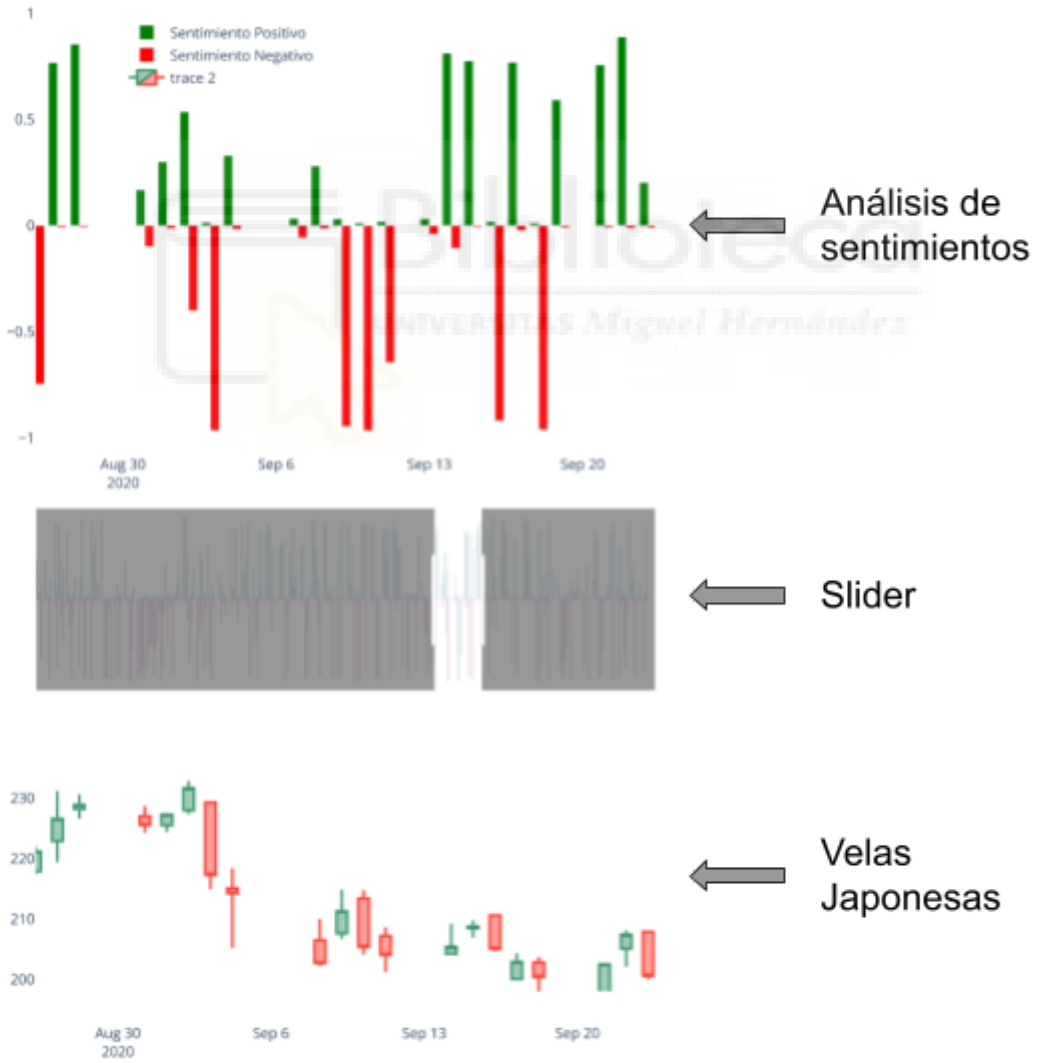

Fig. 4.21: Visualización completa de resultados en periodo indicado por ”slider”.

Finalmente, se unen ambas gráficas en un único lienzo, junto con un deslizador de rangos o "range slider". Este elemento permite realizar interacciones con los gráficos de manera conjunta y sincronizada, obteniendo visualizaciones más completas (Fig. 4.21).

4.7.- Resultados

Los resultados obtenidos por los modelos se almacenan en los DataFrame de resultados. Estos están compuestos por las fechas de publicación de los artículos, sus títulos, la fuentes comunicativas donde fueron publicados y los valores de sentimiento en forma de polaridad, siendo negativos, neutros o positivos (Fig. 4.22). Nótese que los valores de polaridad de los tres campos deben sumar uno, esto se debe a que los modelos predijeron estos resultados en base a probabilidades según el léxico empleado o el contexto de las palabras dentro de los títulos.

Index	date	headline	positive	negative	neutral	source
0	2020-01-01 00:00:00	WhatsApp no Longer Supported on Windows Phones	0.0191475	0.0685437	0.912309	Digital Information World
1	2020-01-02 00:00:00	3 Things Microsoft Does Better Than Google	0.460677	0.0104419	0.528881	The Motley Fool
2	2020-01-03 00:00:00	Datadog Adds Support for Microsoft Azure DevOps	0.0938506	0.0117044	0.894445	DevOps.com
3	2020-01-05 00:00:00	These Charts Chronicle How Microsoft Has Changed Under Satya	0.176382	0.0191402	0.804478	Business Insider
4	2020-01-06 00:00:00	Microsoft Scoops Up Ex-Bridgewater Sales Chief Sháka Rasheed	0.0382518	0.419544	0.542204	Institutional Investor
5	2020-01-07 00:00:00	Microsoft Is Winning The 'Cloud War' Against Amazon: Report	0.455115	0.0164992	0.528386	Forbes

Fig. 4.22: DataFrame de resultados, obtenido por el modelo FinBERT.

Gracias a estos datos podemos observar cuáles fueron las noticias más negativas o positivas del dataset, las cuales pudieron generar un posible impacto en los mercados. Aunque, también ha de tenerse en cuenta que el término correlación no necesariamente implica causalidad. Es posible que la correlación entre noticias económicas y movimientos bursátiles no fuera causada expresamente por la publicación de las noticias y estas fueran producto del movimiento, apareciendo tiempo después. Es por ello que se van a estudiar las correlaciones entre los resultados de las noticias y los datos de las acciones pertenecientes al mismo día o al siguiente día con datos de cotización, en aquellos periodos en los que la bolsa está cerrada, fines de semana o vacaciones, no habrá valores de precios disponibles, por lo que se buscará relacionar el impacto de cada noticia con la siguiente jornada en la que los mercados estén abiertos.

El atributo utilizado para el tratamiento de los datos bursátiles será el precio de cierre (último valor obtenido en el día). De este modo podremos comprobar si las noticias preceden (o no) al valor financiero, ya que este valor será el último del día de mercado.

Para comenzar el análisis, en la tabla 4.1, se muestra una primera observación de los resultados proporcionados por los tres clasificadores, FinBERT, VADER y Fine-tuned BERT, a la hora de determinar la polaridad de los titulares extraídos con la librería

PyGoogleNews. Esta tabla muestra, para cada clasificador, el número de titulares que obtuvieron una mayor puntuación en el atributo “positivo”, “negativo” o “neutral”.

	FinBERT	VADER	fine-tuned BERT
Positivos	88 (28%)	13 (4%)	194 (58%)
Neutrales	136 (43%)	301 (95%)	134 (40%)
Negativos	91 (29%)	4 (1%)	9 (3%)
TOTAL	**315**	**318**	**337**

Tabla 4.1: Polaridades obtenidas por los modelos, periodo 2020.

Como podemos observar, hay diferencias claras entre los tres clasificadores. El modelo FinBERT, si bien genera una mayoría de clasificaciones neutras, también proporciona un número significativo de clasificaciones positivas y negativas, se podría decir que es el modelo que devuelve unos resultados más equilibrados en cuanto a su distribución. Por otra parte, los modelos VADER y BERT afinado, prácticamente no clasifican casi ninguna noticia como negativa, en el caso concreto de VADER también proporciona muy pocas noticias positivas, siendo casi todas neutras. En cambio el modelo BERT afinado distribuye la mayor parte de las noticias como positivas o negativas, predominando las positivas. La figura 4.23 muestra una comparación de cómo quedaría la clasificación de titulares para los modelos FinBERT y BERT.

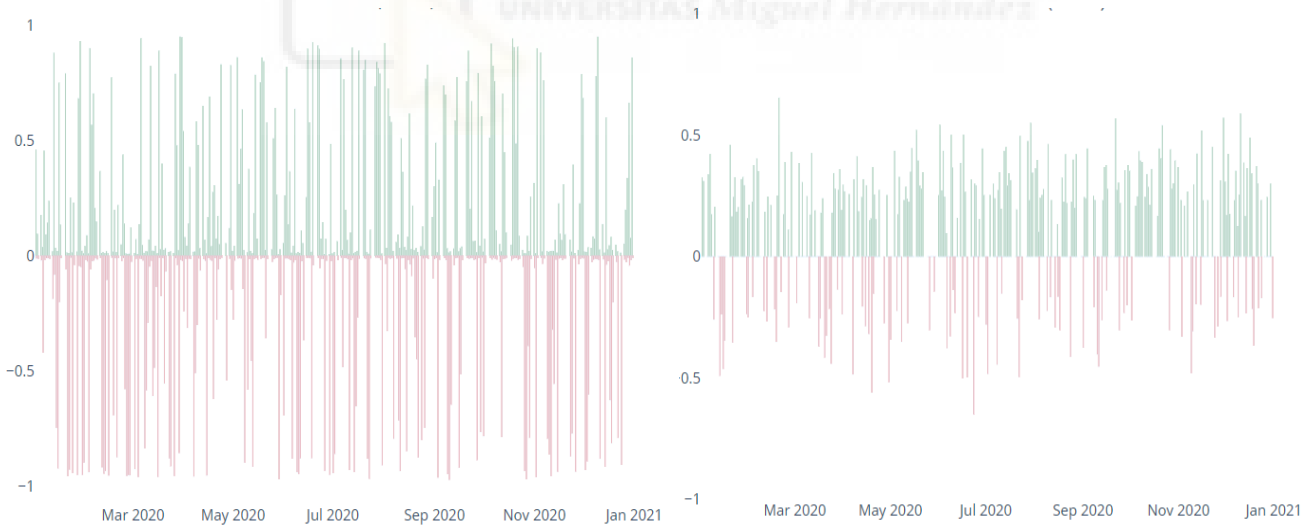

Fig. 4.23: Comparación de resultados entre modelos FinBERT (izq.) y VADER (der.).

Si analizamos de igual manera los resultados del segundo periodo, año 2021, se observan resultados similares, es decir, el modelo FinBERT genera los resultados más equilibrados, VADER clasifica casi todas las muestras como neutras y Fine-tuned BERT clasifica mayoritariamente los titulares como positivos o negativos, predominando los positivos (ver tabla 4.2).

	FinBERT	VADER	fine-tuned BERT
Positivos	85 (27%)	11 (3%)	208 (65%)
Neutrales	113 (36%)	298 (94%)	104 (33%)
Negativos	116 (37%)	7 (2%)	6 (2%)
TOTAL	**314**	**316**	**318**

Tabla 4.2: Polaridades obtenidas por los modelos, periodo 2021.

Hecha esta clasificación, a continuación se muestran dos estudios en los que se busca detectar correlaciones entre las polaridades positivas y negativas de las noticias y posibles cambios en el precio de las acciones de la compañía Microsoft.

4.7.1.- Primer estudio

Dado que el objetivo final de un trabajo de este tipo es apoyar la toma de decisiones de un inversor para poder realizar operaciones de compra/venta lo más acertadas posible, se va a realizar un primer estudio en el que se van a considerar solo aquellos titulares que hayan obtenido valores muy altos (en valor absoluto) de polaridad positiva o negativa. Para este supuesto, se establece como umbral el valor 0,8 (80%) para determinar si una noticia debe o no ser tomada en cuenta, es decir, solo se considerarán las noticias con una polaridad positiva mayor o igual a 0,8 y las de polaridad negativa menor o igual a -0,8. Dado que solamente el modelo FinBERT proporciona suficientes noticias positivas y negativas, este primer estudio se realizará únicamente con los titulares clasificados por dicho modelo.

	2020	2021
Positivos	66 (63%)	80 (63%)
Negativos	39 (37%)	48 (38%)
TOTAL	**105**	**128**

Tabla 4.3: Resultados de valores altos obtenidos por FinBERT, periodo 2020-2021.

La tabla 4.3 muestra la cantidad de titulares positivos y negativos que superan el umbral del 80% con el modelo FinBERT. Nótese que, si bien, antes de aplicar dicho umbral este modelo generaba más noticias negativas que positivas (tablas 4.1 y 4.2), ahora se tiene que hay más titulares positivos que negativos entre los que superan el límite del 80%. Esto se cumple tanto para el año 2020 como para el 2021.

A continuación, se van a buscar relaciones entre estas noticias, o mejor dicho, entre las fechas en las que se publicaron y las fluctuaciones en el precio de las acciones en esa misma fecha o fechas posteriores, la ideas es que un titular positivo debería producir un aumento en el precio de la acción y, de igual modo, un titular negativo debería provocar una bajada de dicho precio. En la figura 4.24 se muestra el código implementado para

buscar las mencionadas relaciones entre titulares y precios para el caso de titulares con polaridad positiva (para negativa solo hay que cambiar la condición de la línea 2).

```
1  for index, row in results.iterrows():
2    if row['negative'] < row['positive']:
3      if df_price.loc[df_price['Date'] == row['date']].index !=0:
4        idant = df_price.loc[df_price['Date'] == row['date']].index
5      else:
6        for ind, row2 in df_price.iterrows():
7          if row2['Date'] > row['date']:
8            idant = df_price.loc[df_price['Date'] == row2['Date']].index
9            break;
10     precio = df_price.iloc[idant[0]]
11     if (idant[0] + 1) < len(df_price.axes[0]):
12       nextprecio = df_price.iloc[idant[0] + 1]
13       next_diff = nextprecio['close_diff']
14
15     negatives.append(row['positive'])
16     differences.append(precio['close_diff'])
17     next_differences.append(next_diff)
18     dates.append(row['date'])
19
20 table22 = { "negatives": negatives, "differences": differences,
21              "next_differences" : next_differences, "dates": dates }
22
23 correlations_df = pd.DataFrame(table22, columns=["negatives", "differences",
24                                                  "next_differences", "dates"])
```

Fig. 4.24: Detección de correlaciones positivas.

Tras buscar relaciones entre titulares y la cotización de acciones se obtienen los resultados que se muestran en las tablas 4.4 y 4.5.

Cierre	Corr. / Positivos (%)	Corr. / Negativos (%)
Día Actual	32 / 66 (48%)	25 / 39 (64%)
Siguiente	26 / 66 (39%)	19 / 39 (49%)
Actual o siguiente	50 / 66 (76%)	35 / 39 (90%)

Tabla 4.4: Correlaciones con valores altos obtenidas por FinBERT, periodo 2020.

Cierre	Corr. / Positivos (%)	Corr. / Negativos (%)
Día Actual	31 / 80 (39%)	23 / 48 (48%)
Siguiente	33 / 80 (41%)	26 / 48 (54%)
Actual o siguiente	50 / 80 (63%)	34 / 48 (71%)

Tabla 4.5: Correlaciones con valores altos obtenidas por FinBERT, periodo 2021.

Observando los resultados, se descubre que, para titulares positivos, de los 66 que había disponibles para el año 2020, en 32 ocasiones (48%) el precio de las acciones de Microsoft experimentó un aumento del precio el mismo día de publicación del titular y en 26 ocasiones (39%) se predijo un aumento del precio al día siguiente. Globalmente, hubo

aumento de precio, bien el mismo día o bien al día siguiente, en 50 ocasiones (76%). De forma análoga se observa como para los titulares negativos se ha detectado una correlación directa (bajada de precios) 35 de 39 veces (90%) de forma global (mismo día o siguiente). Igualmente, para el año 2021 (tabla 4.5) también se ha detectado una alta tasa de correlación entre los titulares y la evolución del precio, siendo esta del 63% para titulares positivos y del 71% para titulares negativos.

4.7.2.- Segundo estudio

El siguiente estudio se puede considerar una ampliación del anterior, ahora se van a considerar todos los titulares sin la restricción de una puntuación mínima de 0,8. Se consideran noticias negativas aquellas cuyo valor supere al indicado en el campo positivo, y del mismo modo, serán noticias positivas aquellas cuya puntuación en el campo positivo supere a la del negativo. Además se analizarán los resultados de los tres modelos, ya que con todos ellos se obtienen suficientes noticias tanto positivas como negativas.

	FinBERT	VADER	fine-tuned BERT
Positivos	196 (62%)	176 (70%)	313 (93%)
Negativos	119 (38%)	76 (30%)	24 (7%)
TOTAL	**315**	**252**	**337**

Tabla 4.6: Segundo estudio polaridades obtenidas por los modelos, periodo 2020.

	FinBERT	VADER	fine-tuned BERT
Positivos	174 (55%)	175 (66%)	305 (96%)
Negativos	140 (45%)	91 (34%)	13 (4%)
TOTAL	**314**	**266**	**318**

Tabla 4.7: Segundo estudio polaridades obtenidas por los modelos, periodo 2021.

Las tablas 4.6 y 4.7 indican la cantidad de titulares positivos y negativos obtenidos en los tres modelos en los años 2020 y 2021 respectivamente. En esta ocasión se van a ignorar los valores neutros, salvo cuando sean del 100%, es decir, siempre que un titular tenga un valor positivo mayor que el negativo (independientemente de su valor neutro) se calificará como positivo, igualmente en el caso contrario. Hay una excepción a esta regla en el modelo VADER y es que este modelo clasifica algunas noticias como 100% neutras, estas se han eliminado del dataset para hacer el estudio.

Las figuras 4.25 y 4.26 muestran enamarillo las veces que los titulares negativos según el modelo FinBERT y el modelo VADER, respectivamente, se correspondieron con bajadas del precio de las acciones en el año 2020.

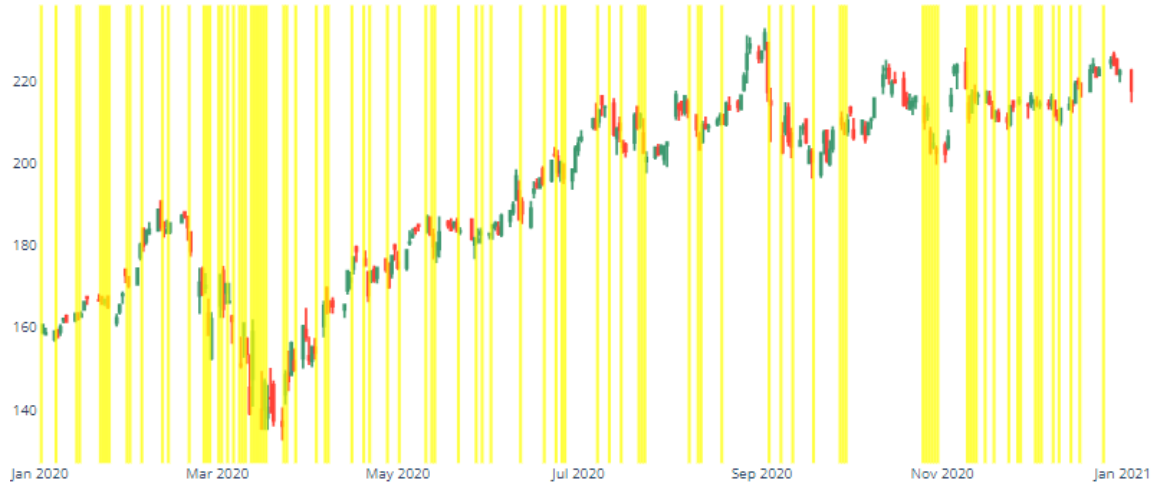

Fig. 4.25: Correlaciones de noticias negativas obtenidas por FinBERT en 2020.

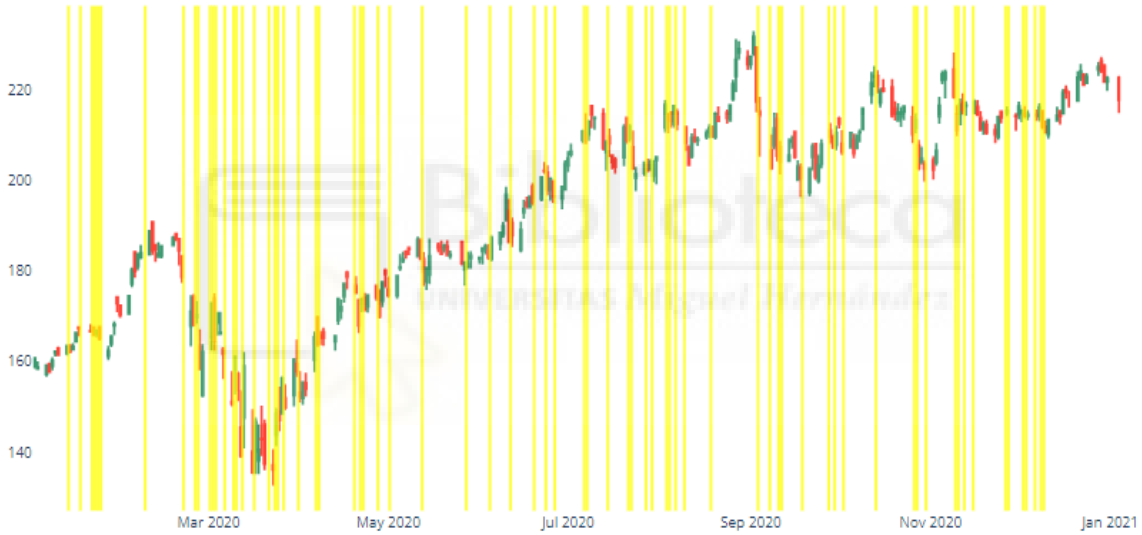

Fig. 4.26: Correlaciones de noticias negativas obtenidas por VADER en 2020.

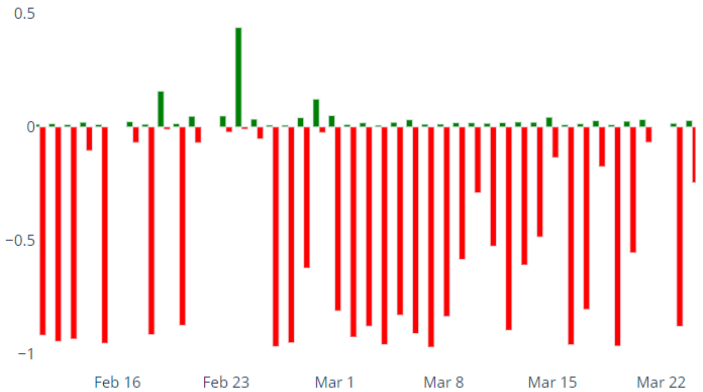

Fig. 4.27: Resultados obtenidos por FinBERT, periodo 11 febrero a 24 marzo de 2020.

Ampliando la zona de febrero a marzo de 2020 (Fig. 4.27) se observa como para el clasificador FinBERT se dan muchas más noticias negativas que positivas, coincidiendo con la bajada de precios que se puede apreciar en esas mismas fechas en las figuras previas (4.26 y 4.27). Este fenómeno también ha sido captado, aunque en menor medida, por el resto de clasificadores.

Numéricamente, las tablas 4.8 (año 2020) y 4.9 (año 2021) muestran, para cada clasificador la la cantidad de veces que se produjo una bajada del precio de la acción coincidiendo con la publicación de un titular negativo.

Cierre	**Correlaciones / Negativos (%)**		
	FinBERT	**VADER**	**fine-tuned BERT**
Día Actual	55 / 119 (46%)	39 / 76 (51%)	12 / 24 (50%)
Siguiente	47 / 119 (39%)	33 / 76 (43%)	8 / 24 (33%)
Actual o siguiente	87 / 119 (73%)	57 / 76 (75%)	17 / 24 (71%)

Tabla 4.8: Segundo estudio correlaciones de noticias negativas obtenidas, periodo 2020.

Cierre	**Correlaciones / Negativos (%)**		
	FinBERT	**VADER**	**fine-tuned BERT**
Día Actual	62 / 140 (44%)	43 / 91 (47%)	7 / 13 (54%)
Siguiente	71 / 140 (51%)	47 / 91 (52%)	8 / 13 (62%)
Actual o siguiente	102 / 140 (73%)	67 / 91 (74%)	10 / 13 (77%)

Tabla 4.9: Segundo estudio correlaciones de noticias negativas obtenidas, periodo 2021.

Observando las tablas, se descubre que, para titulares negativos, el modelo que mejores resultados arroja es VADER donde, de los 76 que había disponibles para el año 2020, en 39 ocasiones (51%) el precio de las acciones de Microsoft experimentó una bajada del precio el mismo día de publicación del titular, y en 33 ocasiones (43%) se predijo una disminución del precio al día siguiente. Globalmente, hubo aumento de precio, bien el mismo día o bien al día siguiente, en 57 ocasiones (75%).

En cuanto a los titulares negativos recogidos en 2021 el modelo que mejor rinde es nuestro modelo entrenado BERT (Fine-tuned BERT), el cual, aunque con menos cantidad de titulares disponibles, ha detectado una correlación directa (bajada de precios) en 7 ocasiones (54%) en el mismo día, y en 8 ocasiones (62%) al día siguiente . Globalmente, el modelo Fine-tuned BERT presenta una tasa de aciertos de bajadas de precios del 77% en el año 2021.

Como dato curioso, resaltar que en el año 2020 se producen más correlaciones directas entre los titulares negativos y las bajadas de precio en el mismo día que en el día siguiente,

sin embargo, en el año 2021 ocurre a la inversa, las predicciones de bajadas de precios son mayores al día siguiente al de publicación del titular .

A continuación se muestran los resultados del análisis análogo realizado con titulares con polaridad positiva.

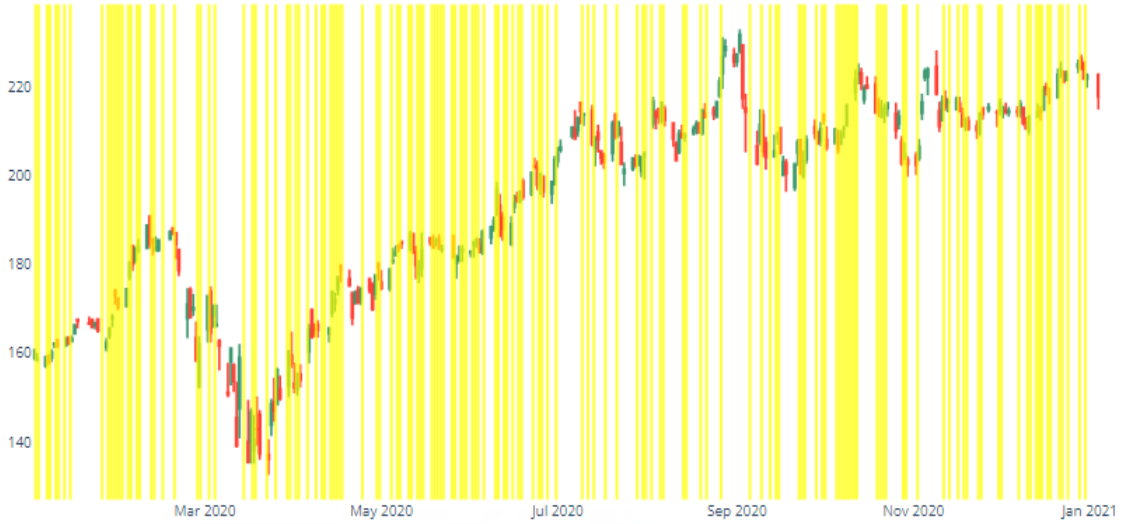

Fig. 4.28: Correlaciones de noticias positivas obtenidas por FinBERT en 2020.

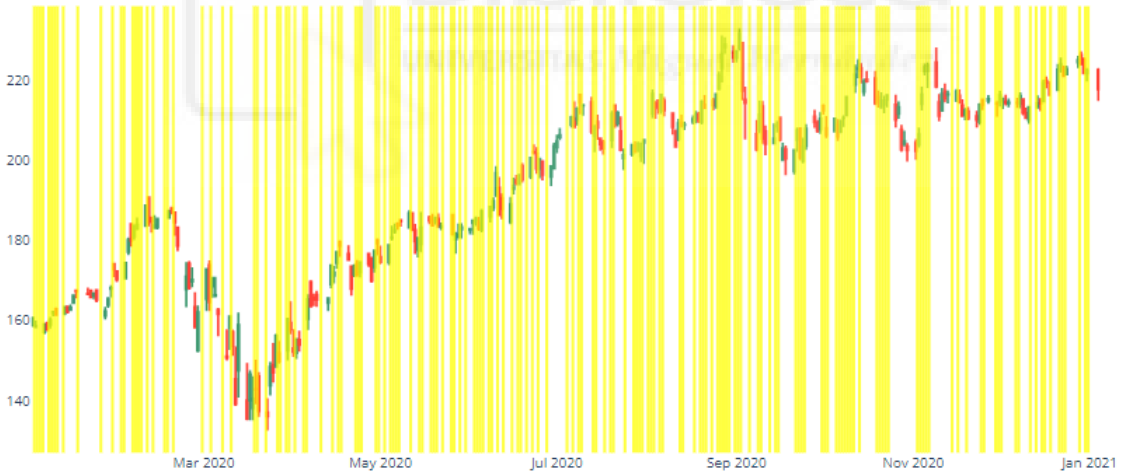

Fig. 4.29: Correlaciones de noticias positivas obtenidas por VADER en 2020.

Cierre	Correlaciones / Positivos (%)		
	FinBERT	VADER	fine-tuned BERT
Día Actual	118 / 196 (60%)	103 / 176 (59%)	180 / 313 (58%)
Siguiente	106 / 196 (54%)	100 / 176 (57%)	170 / 313 (54%)
Actual o siguiente	169 / 196 (86%)	153 / 176 (87%)	270 / 313 (86%)

Tabla 4.10: Segundo estudio correlaciones positivas obtenidas, periodo 2020.

Cierre	Correlaciones / Positivos (%)		
	FinBERT	VADER	fine-tuned BERT
Día Actual	85 / 174 (49%)	85 / 175 (49%)	152 / 305 (50%)
Siguiente	90 / 174 (52%)	87 / 175 (50%)	156 / 305 (51%)
Actual o siguiente	131 / 174 (75%)	133 / 175 (76%)	232 / 305 (76%)

Tabla 4.11: Segundo estudio correlaciones positivas obtenidas, periodo 2021.

Como se observa en las tablas 4.10 y 4.11, los resultados para los titulares positivos son muy similares entre los tres modelos. En general, destaca ligeramente el modelo VADER el cual, para titulares positivos, de los 176 que había disponibles para el año 2020, en 103 ocasiones (59%) el precio de las acciones de Microsoft experimentó una subida el mismo día de publicación del titular y en 100 ocasiones (57%) se produjo un aumento del precio al día siguiente. Globalmente, hubo aumento de precio, bien el mismo día o bien al día siguiente, en 153 ocasiones (87%).

En cuanto a los titulares positivos recogidos en 2021 el modelo que mejor rinde, aunque ligeramente, es nuestro modelo entrenado BERT, el cual sobre 305 noticias positivas, ha detectado una correlación directa (subida de precios) en 152 ocasiones (50%) el mismo día, y en 156 ocasiones (51%) al día siguiente . El valor global para este clasificador (mismo día de la publicación o día siguiente), también se ha detectado una alta tasa de correlación entre los titulares y la evolución del precio, siendo esta del 76%.

Capítulo 5
Conclusiones y trabajo futuro

5.1.- Conclusiones

Tras analizar los resultados, y considerando las predicciones globales, se tiene que las predicciones realizadas con los titulares positivos son moderadamente mejores que con los negativos. Esto difiere de la intuición general que tenemos las personas de que una noticia negativa suele tener más impacto que una positiva, aunque para medir esto correctamente habría que cuantificar si cuando el precio sube, o baja, lo hace en mayor o menor cantidad.

En cuanto a la predicciones considerando cada uno de los modelos, y también en el ámbito global, no hay especial diferencia entre las tasas de aciertos de los modelos ya entrenados FinBERT y VADER, y el modelo afinado en este trabajo Fine-tuned BERT.

A nivel personal, con la realización del proyecto, he tenido la oportunidad de adquirir habilidades y conocimientos valiosos en el campo del aprendizaje profundo y la ciencia de datos. Estas experiencias han enriquecido significativamente mi conjunto de habilidades y ampliado mi comprensión en varios aspectos clave:

- Conocimiento profundo de la estructura de modelos de deep learning avanzados, como BERT, y su adaptación para tareas de procesamiento de lenguaje natural (NLP), enriqueciendo mi comprensión sobre las técnicas de NLP de última generación.

- Aprendizaje de técnicas básicas de obtención de datos, como el web scraping, para recopilar información relevante de fuentes en línea. Esta habilidad es crucial para construir conjuntos de datos válidos para llevar a cabo investigaciones con datos actualizados.

- Mejora de conocimientos en Python, usando librerías como Plotly, Pandas, yfinance, o las librerías específicas para Deep Learning como Transformers (HuggingFace) o vaderSentiment (VADER).

5.2.- Posibles desarrollos futuros

El modelo entrenado en este proyecto, Fine-tuned BERT, presentó ciertos valores incorrectos en algunas de sus métricas de rendimiento, se podría indagar más en este ámbito para tratar de mejorar su funcionamiento.

Se ha realizado un proyecto en el que se han buscado correlaciones entre titulares de prensa sobre una empresa en concreto, Microsoft, y la evolución del precio de sus acciones en bolsa. Este estudio se podría extender a más compañías cotizadas de diferentes sectores (tecnológicas, banca, alimentación, etc.), e incluso a otros tipos de activos como futuros, materias primas, mercado FOREX (divisas) o criptomonedas.

También se puede analizar el impacto de las diferentes noticias según la agencia de comunicación que las publica, en los datos recopilados existe un campo “source” que indica cual es la agencia origen de la noticia, quizá se puedan descubrir posibles sesgos o identificar agencias mejor informadas o especializadas en un activo o sector en concreto.

Adicionalmente, para mejorar los modelos se podrían aplicar técnicas de análisis de streamings de datos para ir actualizando el modelo con los nuevos datos que se usan para hacer predicciones. Nótese que tras la predicción realizada por un modelo sobre si un titular es positivo o negativo, en el plazo de uno o dos días se podrá saber si el precio de las acciones evolucionó de forma coherente (o no) con dicha predicción. Esta información se podría utilizar para ir mejorando el modelo dinámicamente.

Otra posible mejora podría consistir, tal y como se ha apuntado anteriormente, en analizar si las fluctuaciones del precio de las acciones son suaves o abruptas, y comprobar si

noticias con alta polaridad positiva producen (o no) subidas de precios más pronunciadas (análogo para polaridades negativas).

Por último cabría realizar una simulación de toma de decisiones de compra/venta de acciones para ver si, efectivamente, la utilización de estos modelos resulta rentable haciendo inversión real, considerando parámetros como el momento exacto de compra y de venta de un activo, o las comisiones y gastos de cada operación.

Bibliografía

[1] Handbook of Development Metrics
M. Ayhan Kose, Eswar Prasad, Kenneth Rogoff, Shang-JinWei.
North-Holland (2009)

[2] Financial Markets
https://www.occ.gov/topics/supervision-and-examination/capital-markets/financial-markets/index-financial-markets.html
Office of the Comptroller of the Currency / 07-12-2022

[3] Manual de mercados financieros
Jose Luis Martin Marin y Antonio Trujillo Ponce
Thomson Paraninfo (2004)

[4] Diferencias entre Mercado de Capitales y Mercado de Valores
https://www.inesem.es/revistadigital/juridico/diferencia-entre-mercado-de-capitales-y-mercado-de-valores/
Tanoj Vashi de la Torre (2019)

[5] Diccionario Financiero. Vocabulario de inversión
https://www.eurekers.com/cursosbolsa/mercado-de-valores-definicion
Nieves Pérez (2018)

[6] Giving Content to Investor Sentiment: The Role of Media in the Stock Market
Paul C. Tetlock
The Journal Finance (2007)

[7] Media Sentiment and International Asset Prices
Samuel P. Fraiberger, Do Lee, Damien Puy, and Romain Ranciere
International Monetary Fund (2018)

[8] Artificial intelligence in manufacturing and logistics systems: algorithms, applications, and case studies.
Chen-Fu Chien, Stéphane Dauzère-Pérès, Woonghee Tim Huh, Young Jae Jang, James R. Morrison
Taylor & Francis (2020)

[9] Machine learning in marketing: A literature review, conceptual framework, and research agenda
Eric W.T. Ngai, Yuanyuan Wu
Journal of Business Research (2022)

[10] Deep Learning in Medical Image Analysis
Gobert Lee, Hiroshi Fujita
Springer Cham (2020)

[11] Deep Learning for Automated Product Design
Carmen Krahe, Antonio Bräunche, Alexander Jacob, Nicole Stricker, Gisela Lanza
Procedia CIRP (2020)

[12] Deep learning of algorithmic trading strategies & limit order book dynamics.
Bercich, Justin
Macquarie University (2019)

[13] The Secretive World Of MEV, Where Bots Front-Run Crypto Investors For Big Profits.
https://www.forbes.com/sites/jeffkauflin/2022/10/11/the-secretive-world-of-mev-where-crypto-bots-scalp-investors-for-big-profits/?sh=732b0eb52d8d
Levon Biss (2022)

[14] Real-time yield estimation based on deep learning
https://www.researchgate.net/publication/316865640_Real-time_yield_estimation_based_on_deep_learning
Maryam Rahnemoonfar (2017)

[15] The evolution of sentiment analysis—A review of research topics, venues, and top cited papers.
Computer Science Review, Volume 27, Pages 16-32.
Mika V. Mäntylä, Daniel Graziotin, Miikka Kuutila (2017)

[16] MoneyControl
https://www.moneycontrol.com/amp
MoneyControl/ 07-04-2023

[17] RavenPack Edge
https://www.ravenpack.com/products/edge
RavenPack / 30-04-2023

[18] Spark NLP
https://sparknlp.org
John Snow Labs / 08-05-2023

[19] How browsers work: Behind the scenes of modern web browsers
https://web.dev/howbrowserswork/
Tali Garsiel, Paul Irish (2011)

[20] Términos de uso SeekingAlpha
https://about.seekingalpha.com/terms
Seeking Alpha Ltd. (2021) / 13-06-2023

[21] XML and Web Technologies for Data Sciences with R
Springer New York, NY
Deborah Nolan, Duncan Temple Lang (2013)

[22] What is Browser Automation? Guide to Get Started
https://www.browserless.io/blog/2022/09/16/browser-automation/
Girish Patil (2022)

[23] Selenium Webdriver
https://www.selenium.dev/documentation/webdriver/
Selenium / 22-06-2023

[24] Puppeteer
https://pptr.dev/
Puppeteer / 22-06-2023

[25] Scrapingbee Pricing
https://www.scrapingbee.com/#pricing
Scrapingbee / 22-06-2023

[26] Scrapfly: Web Scraping API
https://scrapfly.io/
Scrapfly / 22-06-2023

[27] ScrapingBee
https://www.scrapingbee.com/
Scrapingbee / 22-06-2023

[28] What Is Screen Scraping and How Does It Work?
https://www.octoparse.com/blog/what-do-you-know-about-a-screen-scraper
Ansel Barrett (2022)

[29] 9 Discreet Disadvantages of Optical Character Recognition
https://www.octoparse.com/blog/what-do-you-know-about-a-screen-scraper
Haisam Abdel Malak (2023)

[30] A complete guide to Natural Language Processing
https://www.deeplearning.ai/resources/natural-language-processing/
Deeplearning.AI (2023)

[31] What is Natural Language Processing (NLP)?
https://www.ibm.com/topics/natural-language-processing
IMB / 09-05-2023

[32] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova
Google AI Language (2019)

[33] ¿Qué es un Modelo Transformer?
https://la.blogs.nvidia.com/2022/04/19/que-es-un-modelo-transformer/
Rick Merritt (2022)

[34] Cloze procedure: A new tool for measuring readability
Wilson L Taylor
Journalism Bulletin (1953)

[35] Unlearn dataset bias in natural language inference by fitting the residual.
He He, Sheng Zha, and Haohan Wang.
Association for Computational Linguistics (2019)

[36] Pretrained transformers improve out-of-distribution robustness
Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song
Association for Computational Linguistics (2020)

[37] BERT Explained: State of the art language model for NLP
https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
Rani Horev (2018)

[38] FinBERT: Financial Sentiment Analysis with Pre-trained Language Models
Dogu Tan Araci
University of Amsterdam (2019)

[39] Reuters Corpora (RCV1, RCV2, TRC2)
https://trec.nist.gov/data/reuters/reuters.html
Reuters (2004)

[40] Financial PhraseBank
https://www.researchgate.net/publication/251231364_FinancialPhraseBank-v10
Malo, Pekka & Sinha, Ankur & Takala, Pyry & Korhonen, Pekka & Wallenius, Jyrki. (2013)

[41] DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter
Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf
Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS (2013)

[42] Distilling the Knowledge in a Neural Network
Geoffrey Hinton, Oriol Vinyals, Jeff Dean
NIPS Deep Learning Workshop (2014)

[43] RoBERTa: A Robustly Optimized BERT Pretraining Approach
Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer and Veselin Stoyanov
Facebook AI and University of Washington (2019)

[44] VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text
Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer and Veselin Stoyanov
Facebook AI and University of Washington (2019)

[45] Python Developers Survey 2022 Results
https://lp.jetbrains.com/python-developers-survey-2022/#GeneralPythonUsage
JetBrains (2022) / 28-06-2023

[46] Spyder: The Scientific Python Development Environment
https://www.spyder-ide.org/
Spyder / 28-06-2023

[47] Google Colaboratory
https://colab.google/
Google (2023) / 05-07-2023

[48] Web Archive Google News Search API
http://web.archive.org/web/20110822040220/http://code.google.com/apis/newssearch/
Web Archive / 28-06-2023

[49] PyGoogleNews: Scrape, normalize and mine Google News with Python
https://docs.newscatcherapi.com/open-source/pygooglenews
NewsCatcher Engineering Team (2022) / 28-06-2023

[50] Reliably download historical market data from with Python
https://aroussi.com/post/python-yahoo-finance
Ran Arussi (2019) / 28-06-2023

[51] Good Debt or Bad Debt: Detecting Semantic Orientations in Economic Texts
Malo, Pekka & Sinha, Ankur & Takala, Pyry & Korhonen, Pekka & Wallenius, Jyrki.
Journal of the American Society for Information Science and Technology (2014)

[52] Dataset financial_phrasebank
https://huggingface.co/datasets/financial_phrasebank
Hugging Face / 07-07-2023

[53] pandas
https://pandas.pydata.org/about/index.html
NumFOCUS Inc. (2023) / 28-06-2023

[54] datetime: Basic date and time types
https://docs.python.org/3/library/datetime.html
Python (2023) / 29-06-2023

[55] Plotly Open Source Graphing Library for Python
https://plotly.com/python/
Plotly (2023) / 29-06-2023

[56] NumPy: The fundamental package for scientific computing with Python
https://numpy.org/
NumPy (2023) / 07-07-2023

[57] Transformers: State-of-the-art Machine Learning for PyTorch, TensorFlow, and JAX.
https://huggingface.co/docs/transformers/index
Hugging Face (2023 / 06-07-2023

[58] PyTorch: End-to-end machine learning framework
https://pytorch.org/features/
PyTorch (2023) / 06-07-2023

[59] scikit-learn: Machine Learning in Python
https://scikit-learn.org/stable/index.html
scikit-learn developers / 06-07-2023

[60] Fast-ML
https://github.com/samarth-agrawal-86/fast_ml
Samarth Agrawal (2021) / 06-07-2023

[61] Terms of Use
https://about.seekingalpha.com/terms
SeekingAlpha (2021) / 21-06-2023

[62] Google News RSS Feed
https://rss.app/rss-feed/google-news-rss-feed
RSS.app / 06-07-2023

[63] Microsoft Corporation Common Stock (MSFT)
https://www.nasdaq.com/es/market-activity/stocks/msft
Nasdaq / 06-07-2023

[64] Large Number Coincidences and the Anthropic Principle in Cosmology.
Brandon Carter
International Astronomical Union (1974)

[65] Modern Cosmology & Philosophy
John Leslie
Prometheus Books (1998)

[66] Fine-tuning
https://plato.stanford.edu/entries/fine-tuning/#FineTuniForLifeEvid
Stanford Encyclopedia of Philosophy (2017) / 07-07-2023

[67] What Is Fine-Tuning and How Does It Work in Neural Networks?
https://blog.pangeanic.com/what-is-fine-tuning
Moisés Barrios Torres (2023)

[68] Fine-tuning: Learn how to customize a model for your application.
https://platform.openai.com/docs/guides/fine-tuning
OpenAI / 07-07-2023

[69] Label encoding
https://saturncloud.io/glossary/label-encoding/
SaturnCloud / 08-07-2023

[70] What Is Training Data? How It's Used in Machine Learning
https://learn.g2.com/training-data
Amal Joby (2021)

[71] Sentiment Analysis with Deep Learning. How to train your own high performing sentiment analysis model
https://towardsdatascience.com/how-to-train-a-deep-learning-sentiment-analysis-model-4716c946c2ea
Edwin Tan (2021)

[72] Trainer
https://huggingface.co/transformers/v4.8.2/main_classes/trainer.html
Hugging Face / 10-07-2023

[73] Word Embedding
https://datascientest.com/es/word-embedding
DataScientest / 09-07-2023

[74] The Transformer Model
https://machinelearningmastery.com/the-transformer-model/
Stefania Cristina (2022)

[75] Dropout Layer
https://keras.io/api/layers/regularization_layers/dropout/
Keras / 09-07-2023

[76] Dropout: A Simple Way to Prevent Neural Networks from Overfitting
Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever & Ruslan Salakhutdinov
Department of Computer Science University of Toronto (2014)

[77] NVIDIA Turing GPU Architecture. Graphics Reinvented
https://images.nvidia.com/aem-dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf
NVIDIA Corporation (2018)