

INSTITUTO DE EDUCACION SECUNDARIA ABASTOS



# IA de Análisis y Predicción Financiera con Inteligencia Artificial para el Mercado de Divisas

Memoria del Proyecto

Curso de Especialización en Inteligencia Artificial y Big Data

## Autores

Adrián Martínez

Edvard Khachatryan

Gabriel Mailat

Juan Luján

Vicente Real

**Tutor:** Vicent Tortosa

Curso 2023/2024

Grupo 8IA

30 de mayo de 2024



# Índice

## **1. Identificación y Objetivos del Proyecto**

### **1.1. Identificación**

### **1.2. Objetivos**

## **2. Diseño del Proyecto**

### **2.1. Tecnologías**

### **2.2. Capas de la Arquitectura**

### **2.3. Flujo de Trabajo/Procesamiento de los Datos**

## **3. Desarrollo del Proyecto**

### **3.1. Implementación del Bot**

### **3.3. Dificultades**

## **4. Evaluación y Conclusiones Finales**

### **4.1. Evaluación del Rendimiento**

### **4.2. Conclusiones**

### **4.3. Conclusiones individuales del proyecto**

## **5. Planes de futuro**

## **6. Referencias y Bibliografía**

## **7. Agradecimientos**

# 1. Identificación y Objetivos del Proyecto

## 1.1. Identificación

El proyecto se centra en la creación de un bot diseñado para predecir las tendencias del mercado financiero, específicamente en relación con las acciones de Apple Computer, Inc. (AAPL), aunque su funcionalidad es extrapolable y se puede aplicar a la hora de realizar el análisis de otros activos financieros. La presente herramienta se ha concebido con el objetivo de proporcionar a los usuarios una ayuda que les permita analizar e incluso predecir las tendencias del mercado, y que, por lo tanto, les permita mejorar la toma de decisiones a la hora de realizar inversiones financieras.

El principal reto al que se enfrenta este proyecto es la volatilidad inherente del mercado financiero y la necesidad de predecir movimientos de precios que pueden estar influenciados por una multitud de factores, incluyendo noticias y eventos externos. Para abordar esta complejidad, la herramienta se divide en dos partes, cada una con un enfoque específico en términos de horizonte temporal y fuentes de datos:

### Predicción a Largo Plazo:

Este componente del bot está entrenado para predecir las tendencias del precio de las acciones de Apple para los próximos 10 días. Para ello, se ha utilizado un extenso conjunto de datos históricos, que abarca desde el 1 de enero de 2010 hasta la actualidad. La elección de un periodo tan largo permite capturar patrones y tendencias a largo plazo, proporcionando una base sólida para las predicciones.

Dado que los datos históricos de noticias no están disponibles para todo este periodo, este modelo se basa exclusivamente en los datos cuantitativos históricos del mercado, como precios de apertura, cierre, máximos, mínimos y volúmenes de transacciones. Estos datos proporcionan información sobre la evolución del mercado a lo largo del tiempo, lo que permite identificar tendencias a largo plazo.

### Predicción a Corto Plazo:

El segundo componente de la herramienta se centra en realizar predicciones a muy corto plazo, específicamente para los próximos 30 minutos. Este modelo se entrena utilizando datos más recientes, a partir del 3 de mayo de 2024. La elección de esta fecha se debe a la disponibilidad de datos de noticias que pueden ser analizados para comprender su impacto inmediato en el precio de las acciones.

A diferencia del modelo a largo plazo, este modelo incorpora análisis de sentimientos derivados de noticias financieras recientes. Las noticias se analizan para determinar el sentimiento general y su posible influencia en el mercado. Esta información se integra con los datos de mercado para mejorar la precisión de las predicciones a corto plazo.

La combinación de estos dos enfoques permite que el bot ofrezca predicciones tanto a largo como a corto plazo, proporcionando una herramienta versátil para los inversores. Al integrar análisis cuantitativos y cualitativos, el bot puede ofrecer una visión más completa del mercado, ayudando a los usuarios a tomar decisiones informadas basadas en datos históricos y el impacto inmediato de las noticias.

## 1.2. Objetivos

### Objetivo Principal

El objetivo principal del proyecto es desarrollar una herramienta que pueda analizar datos financieros y predecir tendencias del mercado, proporcionando una herramienta avanzada y confiable para mejorar la toma de decisiones en inversiones financieras. Este software debe ser capaz de interpretar tanto datos históricos como actuales, integrando diversas fuentes de información para ofrecer predicciones precisas y útiles.

### Objetivos Específicos

Para alcanzar el objetivo principal, se han definido una serie de objetivos específicos que guiarán el desarrollo y la implementación del proyecto:

#### 1. Recopilación de Datos Financieros:

Obtener una base de datos extensa y detallada de los precios de las acciones de Apple, comenzando desde el 1 de enero de 2010 hasta la fecha actual. Esta base de datos proporcionará los cimientos necesarios para identificar patrones y tendencias a largo plazo en el comportamiento del mercado.

Recopilar datos actuales y relevantes a partir del 3 de mayo de 2024, incluyendo precios de apertura, cierre, máximos, mínimos y volúmenes de transacciones, así como noticias financieras que puedan influir en el precio de las acciones a corto plazo.

## 2. Análisis de Sentimientos:

Desarrollar un modelo de análisis de sentimientos que pueda evaluar el impacto de las noticias financieras en el precio de las acciones de Apple. Este análisis permitirá identificar el sentimiento general (positivo, negativo o neutral) y cuantificar su posible influencia en el mercado.

Integrar los resultados del análisis de sentimientos con los datos de mercado para mejorar la precisión de las predicciones a corto plazo, ofreciendo una visión más completa del impacto inmediato de las noticias en los precios de las acciones.

## 3. Modelado Predictivo:

Entrenar un modelo de predicción a largo plazo utilizando la base de datos histórica. Este modelo se centrará en identificar patrones y tendencias a lo largo del tiempo, proporcionando predicciones para los próximos 10 días. Se evaluarán diferentes enfoques y algoritmos para seleccionar el modelo que ofrezca la mayor precisión en las predicciones.

Desarrollar un modelo de predicción a corto plazo que combine datos recientes del mercado con resultados de análisis de sentimientos. Este modelo se enfocará en prever movimientos de precios para los próximos 30 minutos, aprovechando la información inmediata derivada de las noticias financieras.

## 4. Validación y Evaluación:

Realizar pruebas exhaustivas para evaluar la precisión de las predicciones de ambos modelos en condiciones de mercado reales. Estas pruebas incluirán la validación cruzada y la comparación de las predicciones con datos reales del mercado para medir el rendimiento y la confiabilidad de los modelos.

Ajustar y optimizar los modelos basándose en los resultados de las pruebas, mejorando continuamente su precisión y efectividad.

## 5. Visualización y Presentación de Resultados:

Desarrollar herramientas de visualización que permitan a los usuarios comprender fácilmente las predicciones y el análisis de datos. Estas herramientas deben presentar los resultados de manera clara y accesible, facilitando la interpretación y el uso práctico de las predicciones del bot.

Crear una interfaz de usuario intuitiva y atractiva que permita a los inversores interactuar con el bot, acceder a predicciones y análisis detallados, y tomar decisiones informadas basadas en datos precisos y actualizados.

## 2. Diseño del Proyecto

### 2.1. Tecnologías

El diseño del proyecto se basa en la integración de una variedad de tecnologías que abarcan desde la recopilación de datos hasta el análisis y la visualización. Estas tecnologías se seleccionaron por su capacidad para manejar grandes volúmenes de datos, realizar análisis complejos y presentar resultados de manera efectiva. A continuación, se detallan las tecnologías utilizadas:

#### Manipulación y análisis de datos

**Pandas:** Es una librería de Python que proporciona estructuras de datos flexibles y eficientes, como Data Frames y Series, para la manipulación y análisis de datos estructurados. En el proyecto, permite la limpieza, transformación y análisis exploratorio de los datos, facilitando la preparación de los datos para su posterior análisis.

**Numpy:** Es una librería fundamental para la computación científica en Python, ofreciendo soporte para *arrays* multidimensionales y una amplia colección de funciones matemáticas y estadísticas. En el proyecto, se utiliza para realizar cálculos matemáticos y estadísticos avanzados, mejorando la eficiencia en el procesamiento de grandes volúmenes de datos.

#### Visualización de datos

**Matplotlib:** Es una librería de Python para la creación de gráficos y visualizaciones estáticas, animadas e interactivas. En el proyecto, es esencial para la representación visual de las tendencias y patrones detectados en los datos, permitiendo una mejor comprensión y comunicación de los resultados.

**Kibana:** Es una herramienta de visualización de datos para Elasticsearch, que facilita la creación de dashboards interactivos y análisis en tiempo real. En el proyecto, se utiliza para crear dashboards que visualizan y analizan los datos almacenados en Elasticsearch, proporcionando una vista integral y en tiempo real del rendimiento y las métricas clave.

## Recopilación de datos

**Selenium:** Es una suite de herramientas para la automatización de navegadores web, comúnmente utilizada para pruebas automatizadas y web scraping. En el proyecto, se emplea para automatizar la recopilación de datos de noticias de diversas páginas web, lo que permite mantener el dataset actualizado con información relevante.

**Alpha Vantage:** Es una API que proporciona datos financieros, históricos y en tiempo real, incluyendo información sobre acciones, divisas y criptomonedas. En el proyecto, proporciona los datos financieros necesarios sobre las acciones de Apple, utilizados en el análisis y modelado predictivo de los precios de las acciones.

**Yahoo Finance:** Es una herramienta que ofrece acceso a información detallada sobre el mercado de valores, incluyendo precios históricos y actuales, noticias financieras, análisis y gráficos interactivos. A través de su API y otras herramientas de acceso a datos, permite obtener datos valiosos para realizar estudios financieros y predicciones de mercado.

## Almacenamiento y búsqueda de datos

**Elasticsearch:** Es un motor de búsqueda y análisis distribuido, diseñado para manejar grandes volúmenes de datos en tiempo real. En el proyecto, almacena y permite la búsqueda eficiente de grandes volúmenes de datos recopilados y procesados, soportando las consultas necesarias para el análisis.

**Logstash:** Es una herramienta de procesamiento de datos que ingiere, transforma y almacena logs y otros datos, preparándose para su análisis en Elasticsearch. En el proyecto, procesa y transforma los datos antes de almacenarlos en Elasticsearch, asegurando que los datos estén en el formato adecuado para su análisis y visualización.

## Procesamiento de datos y consultas

**Groq:** Es una herramienta de consulta de datos que integra capacidades de procesamiento de lenguaje natural (NLP) para análisis avanzado. En el proyecto, se utiliza para realizar consultas complejas y análisis avanzados de los datos, integrando técnicas de NLP para extraer información relevante de los textos recopilados.

## Análisis y modelado predictivo

**Tensorflow:** Es un framework de código abierto para el aprendizaje automático y la inteligencia artificial, especialmente adecuado para la creación y entrenamiento de modelos de redes neuronales. En el proyecto, se emplea para desarrollar y entrenar modelos predictivos, como las redes neuronales LSTM, que se utilizan para predecir los precios futuros de las acciones basándose en datos históricos y otros factores.

**Llama3:** Es un modelo de lenguaje natural avanzado, utilizado para tareas de procesamiento de lenguaje natural (NLP) como el análisis de sentimientos. En el proyecto, se usa para evaluar cómo las noticias y otros textos afectan los precios de las acciones mediante el análisis de sentimientos, proporcionando una perspectiva adicional para los modelos predictivos.

## Desarrollo de interfaz de usuario

**Astro:** Es un framework moderno para la construcción de aplicaciones web, optimizado para la creación de interfaces de usuario rápidas y eficientes. En el proyecto, se utiliza para desarrollar la interfaz de usuario de la aplicación web, facilitando la interacción de los usuarios con los datos y las visualizaciones generadas.

**TypeScript:** Es un lenguaje de programación que extiende JavaScript con tipado estático, mejorando la mantenibilidad y la detección de errores en el código. En el proyecto, se emplea en el desarrollo del frontend de la aplicación web, proporcionando un entorno de desarrollo más robusto y eficiente.

**Tailwind CSS:** Es un framework de diseño CSS que permite crear interfaces de usuario estilizadas mediante clases utilitarias predefinidas. En el proyecto, facilita la estilización de la interfaz de usuario, asegurando una apariencia coherente y profesional de la aplicación web.

## Desarrollo y colaboración

**Google Colab:** Es una plataforma en la nube que permite el desarrollo y la ejecución de notebooks de Python, ofreciendo acceso a recursos computacionales adicionales. En el proyecto, se utiliza para la colaboración en el desarrollo y ejecución de análisis y modelos en Python, facilitando el trabajo en equipo y el uso de recursos de computación en la nube.



**Visual Studio Code (VSCode):** Es un entorno de desarrollo integrado (IDE) ligero y potente, que soporta una amplia variedad de lenguajes de programación y herramientas de desarrollo. En el proyecto, se utiliza para escribir y depurar el código del proyecto, proporcionando un entorno de desarrollo eficiente y personalizable.

**Vast.ai:** Es una plataforma de alquiler de computación en la nube diseñada específicamente para tareas intensivas en GPU, como el aprendizaje profundo (deep learning) y otras aplicaciones que requieren un gran poder de procesamiento. Su objetivo principal es ofrecer a los usuarios acceso a recursos de GPU de manera eficiente y a bajo costo.

## Gestión de experimentos

**Comet:** Es una plataforma para el seguimiento y gestión de experimentos de aprendizaje automático, que permite monitorear el rendimiento y organizar los resultados de diferentes modelos. En el proyecto, se emplea para monitorear el rendimiento de los modelos de aprendizaje automático durante su entrenamiento y gestión, asegurando que se mantenga un registro detallado de los experimentos y sus resultados.

## 2.2. Capas de la Arquitectura

El diseño de la arquitectura del proyecto se organiza en varias capas, cada una de las cuales desempeña un papel crucial en el flujo de datos y en la funcionalidad general del sistema. Estas capas son:

### Capa de Datos:

Esta capa es la que se encarga de obtener y almacenar los datos que más tarde van a ser usados para su procesamiento.

### Capa de Procesamiento:

Análisis de Datos: Pandas y Numpy se emplean para la limpieza, transformación y análisis de los datos recopilados.

Análisis de Sentimientos: Se utiliza Llama3 junto con la API de Groq para evaluar el impacto de las noticias en los precios de las acciones.

Procesamiento de Logs: Logstash se encarga de procesar y transformar los datos antes de almacenarlos en Elasticsearch.

**Capa de Visualización:**

Capa dedicada a visualizar los datos. Matplotlib y Kibana se utilizan para crear gráficos y dashboards interactivos que permiten analizar visualmente las tendencias y patrones detectados.

**Capa de Interfaz de Usuario:**

En esta capa el cliente puede ver los resultados de las tendencias predecidas. Se utilizó Astro, TypeScript y Tailwind para desarrollar una interfaz de usuario interactiva y estilizada, permitiendo a los usuarios acceder a las predicciones y análisis de manera intuitiva.

## 2.3. Flujo de Trabajo/Procesamiento de los Datos

El flujo de trabajo del proyecto sigue una secuencia bien definida de etapas, desde la recopilación de datos hasta la presentación de los resultados al usuario. Este flujo asegura que los datos se procesen de manera eficiente y que los resultados sean precisos y útiles.

### 1. Recopilación de Datos:

Se utiliza Selenium para automatizar la extracción de noticias de diversas páginas web.

La API de Alpha Vantage y Yahoo Finance se emplea para obtener datos financieros, históricos y en tiempo real de las acciones de Apple.

### 2. Almacenamiento de Datos:

Los datos recopilados se almacenan en Elasticsearch, permitiendo una gestión eficiente y búsquedas rápidas.

### 3. Procesamiento de Datos:

Con Pandas y Numpy, se realiza la limpieza y transformación de los datos, preparando el dataset para el análisis. Se hacen las oportunas correlaciones para eliminar campos, limpieza de valores nulos, o extracción de senos y cosenos para la temporalidad recurrente.

Se aplica el modelo Llama3 mediante la API de Groq para realizar un análisis de sentimientos sobre las noticias recopiladas.

#### 4. Análisis y Modelado Predictivo:

Se desarrollan y entrenan modelos de redes neuronales LSTM utilizando Tensorflow. Estos modelos se utilizan para predecir las tendencias de precios a corto y largo plazo. Además, se utilizaron modelos de machine learning como Decision Tree, Linear Regresión, Ridge Regression y Lasso Regresión.

Los resultados del análisis de sentimientos se incorporan al dataset, mejorando la precisión de las predicciones.

Comet se utiliza para monitorear el rendimiento de los modelos y gestionar los experimentos de aprendizaje automático.

#### 5. Visualización de Resultados:

Matplotlib se utiliza para generar gráficos que representen las tendencias y predicciones de precios y continuidad de los senos y cosenos de las temporalidades.

Kibana permite la creación de dashboards interactivos, facilitando el análisis visual de los datos en tiempo real.

#### 6. Presentación de Resultados:

Se desarrolla una interfaz web utilizando Astro, TypeScript y Tailwind. Esta interfaz permite a los usuarios interactuar con el bot, visualizar predicciones y acceder a análisis detallados de los datos.

## 3. Desarrollo del Proyecto

### 3.1. Implementación del Bot

La implementación del bot para la predicción de tendencias en el mercado financiero de las acciones de Apple (AAPL) se llevó a cabo en varias fases, cada una de las cuales implicó la utilización de diferentes tecnologías y enfoques metodológicos.

## Recopilación y Preparación de Datos

### Obtención de Datos Financieros:

Se empleó la API de Alpha Vantage y la API de Yahoo Finance para obtener datos financieros sobre las acciones de Apple. Los datos recolectados incluyeron las columnas date, open, high, low, close y volume.

```
import yfinance as yf
ticker = "AAPL"
FECHA_INICIO = "2022-06-01"
data = yf.download("AAPL", start=FECHA_INICIO, interval="60m")
```

```
def descargar_datos_alpha_vantage(ticker):
    ts = TimeSeries(key=API_KEY, output_format='pandas')
    data, _ = ts.get_intraday(symbol=ticker, interval='1min', outputsize='full')
    return data

ticker = "AAPL"

# Descargar datos con intervalo de 1 minuto
data = descargar_datos_alpha_vantage(ticker)

data.index = pd.to_datetime(data.index)

# Filtrar datos a partir del 3 de mayo de 2024
fecha_inicio = '2024-05-03'
data_filtrada = data[data.index >= fecha_inicio]

mapeo_columnas_nombre = {
    '1. open': 'open',
    '2. high': 'high',
    '3. low': 'low',
    '4. close': 'close',
    '5. volume': 'volume'
}

data_filtrada.rename(columns=mapeo_columnas_nombre, inplace=True)

aapl_df = data_filtrada.sort_index(ascending=True)
aapl_df.to_csv('./datasets/AAPL.csv')
```

Selenium se utilizó para automatizar la extracción de noticias financieras de diversas páginas web. Estas noticias proporcionaron el contexto necesario para el análisis de sentimientos.

La primera página a escarpar es “*investing.com*”.

Para ello se crea una lista de *URLs* dentro de la misma web.

```
# Lista de URLs a revisar
urls_noticias = [
    'https://es.investing.com/equities/apple-computer-inc-news',
    'https://es.investing.com/equities/apple-computer-inc-news/2',
    'https://es.investing.com/equities/apple-computer-inc-news/3',
    'https://es.investing.com/equities/apple-computer-inc-news/4',
    'https://es.investing.com/equities/apple-computer-inc-news/5',
    'https://es.investing.com/equities/apple-computer-inc-news/6',
    'https://es.investing.com/equities/apple-computer-inc-news/7',
    'https://es.investing.com/equities/apple-computer-inc-news/8',
    'https://es.investing.com/equities/apple-computer-inc-news/9',
    'https://es.investing.com/equities/apple-computer-inc-news/10',
]
```

Luego, se crea un bucle que recorre todas las *URL*'s. Por cada página se abre un driver de Selenium y en caso de que la página sea la deseada, se ejecuta un código para obtener los datos de las noticias.

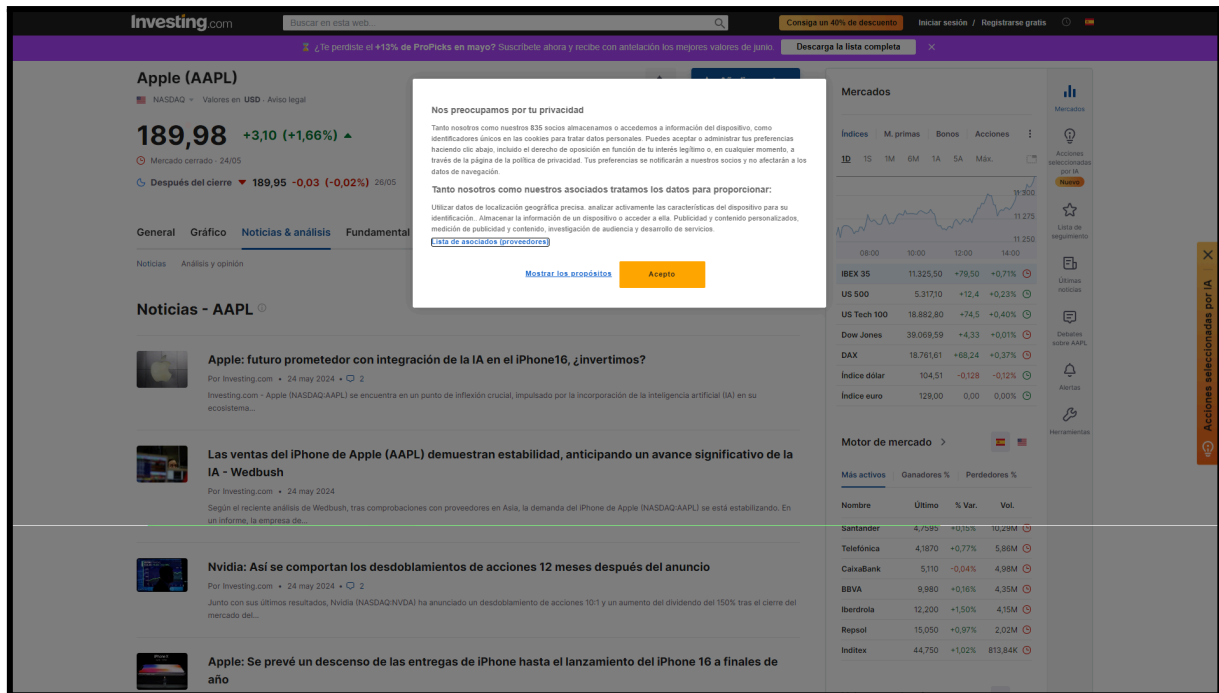
```
for pagina_principal in urls_noticias:
    driver.get(pagina_principal)

    if "investing.com" in pagina_principal:
```

La web que se va a atacar es la siguiente.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager

# Configuración del navegador
chrome_options = Options()
chrome_options.add_argument("--headless")
chrome_options.add_argument("--disable-gpu")
chrome_options.add_argument("--no-sandbox")
service = Service(ChromeDriverManager().install())
driver = webdriver.Chrome(service=service, options=chrome_options)
```



Cuando se accede a la página, lo primero es aceptar las cookies para poder acceder al resto del contenido. En caso de no ser la primera vez que se accede a la página, esta opción no aparece, por lo que no se hace nada.

```
try:
    WebDriverWait(driver, 10).until(EC.visibility_of_element_located((By.CSS_SELECTOR, "#onetrust-accept-btn-handler"))).click()
    print("Cookies aceptadas.")
except Exception as e:
    pass
```

Ahora que ya se tiene acceso completo a la página, el *scraping* se ha planteado de la siguiente manera. Primero se abren todas las noticias que hay en diferentes pestañas. Selenium se coloca en una de las pestañas y obtiene el contenido de la noticia. Una vez lo ha conseguido se cierra la pestaña y continúa con la siguiente. Hay noticias que son "premium". En ese caso se cierra la pestaña directamente, ya que no se puede leer su contenido.

El código para realizar esto es el siguiente:

```

WebDriverWait(driver, 10).until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, "article[data-test='article-item']")))
contenedores_noticias = driver.find_elements(By.CSS_SELECTOR, "article[data-test='article-item']")
links = []
textos = []

for contenedor in contenedores_noticias:
    if contenedor.find_elements(By.CSS_SELECTOR, "svg"):
        print("Noticia PRO encontrada y omitida:", contenedor.text)
        continue
    titulo = contenedor.find_element(By.CSS_SELECTOR, "a[data-test='article-title-link']")
    links.append(titulo.get_attribute('href'))
    textos.append(titulo.text)

print(f"Total de enlaces encontrados: {len(links)}")

```

```

# Abrir cada enlace en una nueva pestaña
for link in links:
    driver.execute_script("window.open(arguments[0]);", link)
print("Todas las pestañas de noticias abiertas.")

```

```

# Cambiar a cada pestaña, extraer el texto y cerrar la pestaña
for handle in driver.window_handles[1:]:
    driver.switch_to.window(handle)
    time.sleep(5)

if driver.current_url == 'https://es.investing.com/equities/null':
    print("Pestaña con URL 'null' encontrada y cerrada.")
    driver.close()
    driver.switch_to.window(driver.window_handles[0])
    continue

try:
    WebDriverWait(driver, 10).until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, "div#article p")))
    parrafos_noticia = driver.find_elements(By.CSS_SELECTOR, "div#article p")
    texto_completo = " ".join([parrafo.text for parrafo in parrafos_noticia])

    # Extraer la fecha del artículo
    fecha_elemento = driver.find_element(By.XPATH, '//div[@class="flex flex-row items-center"]/span')
    fecha_noticia = fecha_elemento.text.strip()

    if fecha_noticia:
        fecha_texto = fecha_noticia.replace("Publicado ", "")

        # Convertir la fecha al formato YYYYMMDDHHmmss
        try:
            fecha_formateada = datetime.strptime(fecha_texto, "%d.%m.%Y, %H:%M").strftime("%Y%m%d%H%M%S")
        except ValueError as ve:
            print(f"Error al formatear la fecha: {fecha_texto} - {ve}")
            driver.close()
            driver.switch_to.window(driver.window_handles[0])
            continue

        noticias_links.append(driver.current_url)
        noticias_textos.append(texto_completo)
        fechas_noticias.append(fecha_formateada)
        print("=*80")
        print("Título:", driver.title)
        print("Fecha de la noticia:", fecha_formateada)
        print("Texto completo de la noticia:", texto_completo)
    else:
        print("Fecha vacía encontrada y omitida.")
except Exception as e:
    print("Error al intentar obtener el texto completo de la noticia:", str(e))

driver.close()
driver.switch_to.window(driver.window_handles[0])

```

## Análisis de Sentimientos:

Se personalizó un modelo basado en Llama3 usando la API de Groq para realizar el análisis de sentimientos. Este modelo devuelve un valor entre 0 y 1 que refleja cómo puede afectar una noticia al precio de mercado de Apple.

Este resultado se incluyó en el dataset bajo la columna **Resultado**.

El código utilizado es el siguiente:

```
client = Groq(API_KEY)
resultados = []

for texto in noticias_textos:
    chat_completion = client.chat.completions.create(
        messages=[
            {
                "role": "system",
                "content":
                "You are a financial bot specialized in predicting the stock market value
                of Apple Inc. (AAPL). Your role involves analyzing the sentiment of input
                text to assess its impact on Apple's stock. You provide a sentiment score
                ranging from 0 to 1, where 0.5 is neutral. Texts positively influencing Ap
                ple's stock will score closer to 1, while those negatively affecting it wi
                ll score closer to 0. You will ONLY provide the sentiment score as a numbe
                r without any additional explanation or any other output."
            },
            {
                "role": "user",
                "content": texto,
            }
        ],
        model="llama3-8b-8192",
        temperature=0,
        max_tokens=4
    )
    resultados.append(chat_completion.choices[0].message.content)

df = pd.DataFrame({
    "Fecha": fechas_noticias,
    "URL": noticias_links,
    "Texto": noticias_textos,
    "Resultado": resultados
})
```



Consiste en un bucle que recorre todas las noticias ya obtenidas anteriormente. Dentro de este bucle se crea un modelo basado en el LLM ***llama3-8b-8192***. Este modelo es modificado mediante el *prompt* que se puede observar en la imagen. La temperatura se ajusta en 0 porque se busca precisión y objetividad en las respuestas del modelo. También se pone el límite en 4 tokens, *ya que* únicamente se busca un número decimal como resultado.

Los resultados se guardan en un CSV para ser

## Procesamiento y Almacenamiento de Datos

Se ejecuta el mismo proceso de procesamiento y almacenamiento de datos tanto para el análisis de sentimientos como para la predicción de nuestro modelo. Es el siguiente los procesos:

### Procesamiento de Datos:

Los datos financieros y los resultados del análisis de sentimientos se limpian y transforman utilizando Pandas y Numpy.

Después del proceso de EDA, como la eliminación de valores nulos, la normalización de datos y la creación de nuevas características derivadas, se exportan estos datos en formatos csv.

```
date,open,high,low,close,volume,Resultado
2024-05-03 04:00:00,183.013,183.013,182.113,182.363,41230.0,0.93
2024-05-03 04:01:00,182.364,182.803,182.303,182.453,29693.0,0.93
2024-05-03 04:02:00,182.493,182.603,182.423,182.553,7365.0,0.93
2024-05-03 04:03:00,182.573,182.603,182.373,182.403,13246.0,0.93
2024-05-03 04:04:00,182.413,182.503,182.363,182.413,4702.0,0.93
2024-05-03 04:05:00,182.413,182.613,182.373,182.603,8625.0,0.93
```

Al almacenarse en el mismo servidor, seguidamente otro archivo de Python ejecutará un contenedor, el cual copiará los datos en dicho contenedor, ejecutando un Logstash, que será el encargado del filtrado y la ingesta de datos en Elasticsearch.

Antes de nada, debe crearse el *mapping*, que será el encargado de decir a Elasticsearch las columnas que van a entrar y de qué tipo tienen que ser.

```

## Mapping de news_scored
PUT /news_scored
{
  "mappings": {
    "properties": {
      "@timestamp": {
        "type": "date"
      },
      "url": {
        "type": "keyword",
        "null_value": "false"
      }
    }
  }
}

```

Se crea el archivo de configuración, el encargado de recibir, filtrar y hacer la ingesta

```

input {
  file {
    path => "/home/resultado.csv"
    start_position => "beginning"
    sincedb_path => "/dev/null"
  }
}

filter {
  csv {
    separator => ";"
    autodetect_column_names => true
  }
}

output {
  stdout {
    codec => rubydebug
  }
}

```

Posteriormente, estos datos serán consultados por la aplicación web, la cual mostrará lo que se crea necesario.

Para lograr este paso es necesario la utilización de la API de Elasticsearch, el cual nos facilita unas instrucciones para ponerla en marcha. En este caso se ha creado un *template* que se encarga de hacer una consulta y traer todos los resultados de ella.

```

## Ver el contenido del indice con un script:
PUT _scripts/get_all_scored
{
  "script": {
    "lang": "mustache",
    "source": {
      "query": {
        "match_all": {}
      },
      "size": 1000
    }
  }
}

```




En la siguiente imagen presentamos el código para poder ver la existencia de dicho *template* desde el mismo elasticsearch.

```

### Probamos el anterior query:
POST /news_scored/_search/template
{
  "id": "get_all_scored"
}

```

Por último, este *template* será usado por la aplicación web y mostrado en ella de manera actualizada cada día.

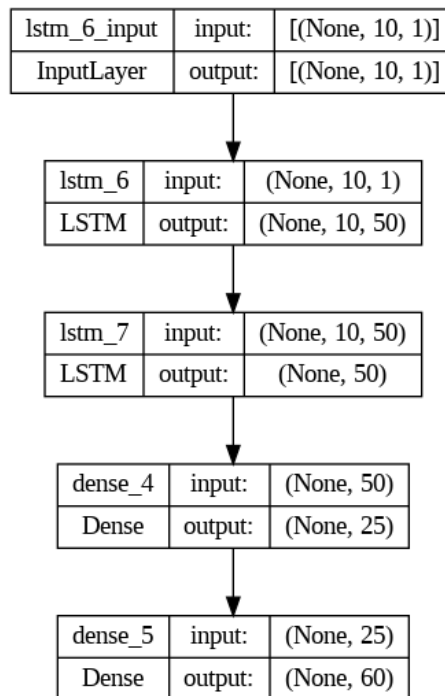
 <p><b>TalkMarkets</b></p> <p><b>Nvidia Puts Up Great Q1 Numbers, +100% YTD</b></p> <p>Looking for stock market analysis and research with proves results? Zacks.com offers in-depth financial research with over 30years of proven results.</p> <p>23-05-2024</p> <p><a href="#">IR A LA NOTICIA</a></p>	 <p><b>Yahoo</b></p> <p><b>3 Blue Chip Stocks That You Can Buy and Hold for Years</b></p> <p>You won't be taking on much risk by investing in these stalwarts.</p> <p>23-05-2024</p> <p><a href="#">IR A LA NOTICIA</a></p>	 <p><b>Benzinga</b></p> <p><b>Blowout Nvidia Earnings – AI Factories And Sovereigns Are Hidden Gems</b></p> <p>Looking for stock market analysis and research with proves results? Zacks.com offers in-depth financial research with over 30years of proven results.</p> <p>23-05-2024</p> <p><a href="#">IR A LA NOTICIA</a></p>
--	--	--

## Modelado Predictivo

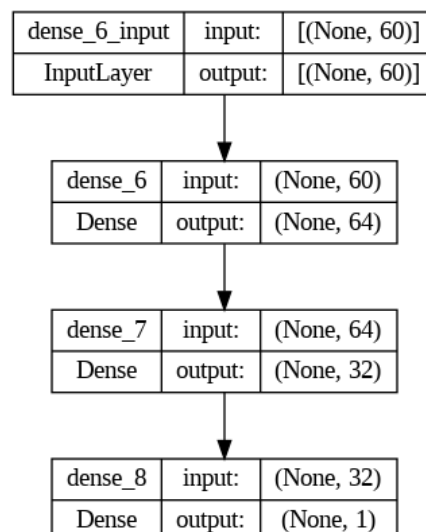
### Desarrollo de Modelos LSTM:

Se desarrollaron dos modelos de redes neuronales LSTM utilizando TensorFlow:

Modelo a corto plazo: Predice el precio de las acciones para los próximos 30 minutos.



Modelo a largo plazo: Predice el precio de las acciones para los próximos 10 días



Los modelos se entrenaron utilizando el dataset combinado de datos financieros y resultados del análisis de sentimientos. Comet se utilizó para monitorear el rendimiento de los modelos y gestionar los experimentos de aprendizaje automático.

## Entrenamiento y Validación:

Los modelos se entrenaron utilizando técnicas de validación cruzada para asegurar su generalización. Se ajustaron los hiperparámetros para optimizar el rendimiento.

Se evaluaron los modelos utilizando métricas como el error cuadrático medio (MSE) y la precisión en las predicciones.

## Interfaz de Usuario

### Desarrollo Frontend del proyecto:

El desarrollo con Astro se planteó de manera modular, se trabajó mediante componentes reutilizables, en los que le pasas las propiedades de cada objeto e internamente manejan los datos.



Se implementó el uso de websockets para poder ver en tiempo real el precio de las acciones, en este caso de Apple y Amazon.

```
socket.addEventListener('open', () => {
  for (const containerId in subscribedSymbols) {
    if (subscribedSymbols.hasOwnProperty(containerId)) {
      const symbol = subscribedSymbols[containerId];
      socket.send(JSON.stringify({ type: 'subscribe', symbol }));
    }
  }
});
```

Para la interfaz de usuario se utilizó un framework de CSS llamado Tailwind CSS, este proporciona clases predeterminadas que pones en el HTML, de esta manera el desarrollo de los estilos ha ido mucho más rápido.

## 3.2. Integración de Tecnologías

La integración de diversas tecnologías fue esencial para el éxito del proyecto. A continuación, se describe cómo se integraron estas tecnologías en el flujo de trabajo del proyecto:

### 1. Automatización y Web Scraping:

Selenium se configuró para automatizar la recopilación de noticias financieras. Este proceso permitió obtener un flujo constante de datos de texto que luego se analizaron para evaluar su impacto en el mercado.

### 2. Análisis de Sentimientos:

Se implementó un modelo de Llama3 para el análisis de sentimientos, el cual se integró con la API de Groq. Este modelo evaluó las noticias recolectadas y generó puntuaciones de sentimiento que se añadieron al dataset.

### 3. Almacenamiento y Procesamiento de Datos:

Elasticsearch y Logstash se integraron para almacenar y procesar grandes volúmenes de datos de manera eficiente. Kibana se utilizó para la visualización y análisis interactivo de estos datos.

### 4. Modelado Predictivo:

Tensorflow se utilizó para desarrollar y entrenar los modelos LSTM. Comet se empleó para realizar un seguimiento detallado de los experimentos, permitiendo ajustes y mejoras en los modelos de manera iterativa.

### 5. Visualización de Resultados:

Matplotlib se utilizó para crear gráficos detallados de las predicciones y tendencias detectadas. Kibana permitió la creación de dashboards interactivos para un análisis más profundo.

## 6. Interfaz de Usuario:

Se desarrolló una interfaz de usuario utilizando Astro, TypeScript y Tailwind. Esta interfaz permite a los usuarios interactuar con el bot, visualizar predicciones y acceder a análisis detallados de los datos.

## 3.3. Dificultades

Durante el desarrollo del proyecto, se enfrentaron varias dificultades que se resolvieron a través de la colaboración y el uso de herramientas adecuadas:

### 1. Limitaciones en la API de Twitter:

Se intentó utilizar la API de Twitter para obtener tweets relevantes, pero se encontraron limitaciones en el uso gratuito de la API. Esto llevó a buscar alternativas de scraping en otras fuentes de noticias financieras.

### 2. Integración de Múltiples Tecnologías:

La integración de diversas tecnologías presentó desafíos técnicos. Se dedicó tiempo a asegurar que todas las herramientas y plataformas se comunicaran de manera efectiva y eficiente.

### 3. Optimización de Modelos:

El entrenamiento de los modelos LSTM requirió ajustes constantes de hiperparámetros y validación cruzada para mejorar la precisión. Comet fue fundamental para monitorear estos experimentos y realizar mejoras iterativas.

### 4. Gestión de Volúmenes de Datos:

La gestión y el procesamiento de grandes volúmenes de datos financieros y de noticias presentaron desafíos en términos de almacenamiento y velocidad de procesamiento. Elasticsearch y Logstash fueron cruciales para manejar estos problemas de manera eficiente.

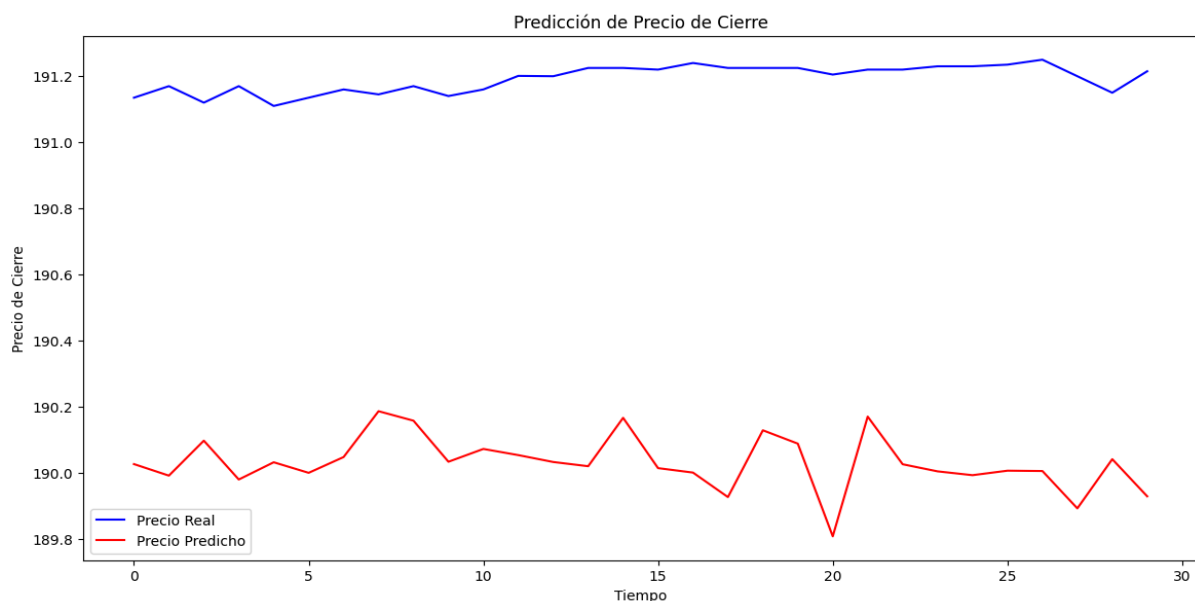
## 5. Entrenamiento de grandes Volúmenes de Datos:

El entrenamiento con un gran volumen de datos (ventana de 960), en el modelo predictivo por minutos, hacía que en entrenamiento durase mucho tiempo, haciendo que la GPU de Google *Collaboratory* nos parara dicho entrenamiento.

## 4. Evaluación y Conclusiones Finales

### 4.1. Evaluación del Rendimiento

Predicción modelo cada 30 minutos:

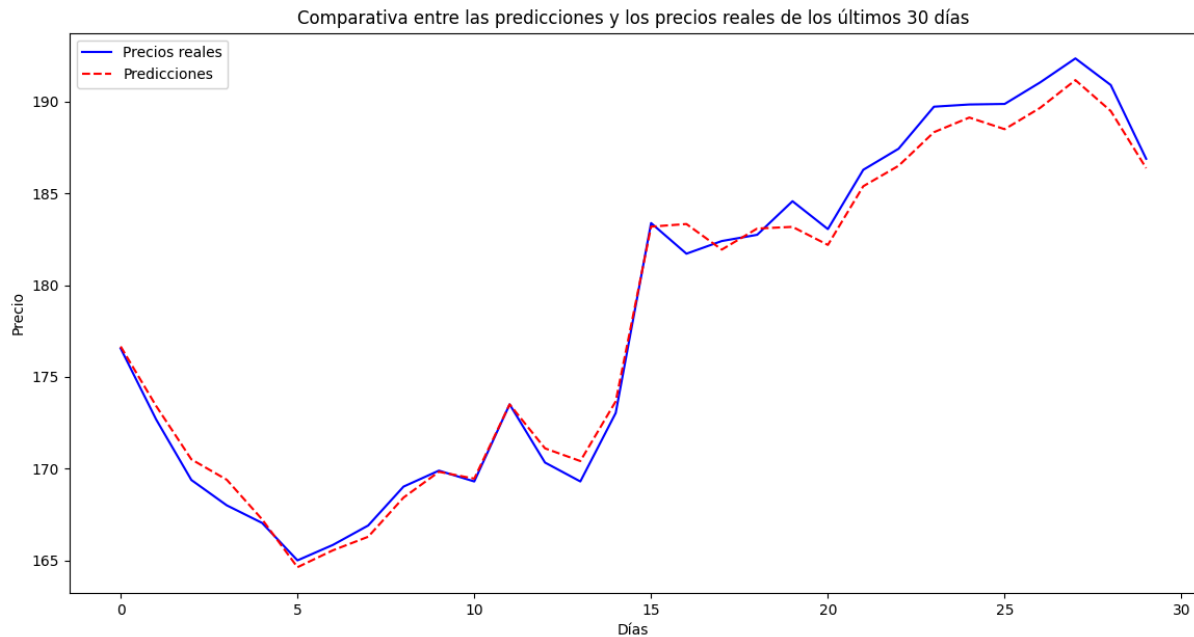


En la imagen anterior se puede observar, en periodos de minutos, el precio real como es sutilmente más alto a nuestra predicción un 0.592%.

Podemos decir que nuestro modelo se acerca bastante al resultado final, pero este tiende a ser mucho más fructífero que el valor real.

Predicción diaria del modelo LSTM:





Si bien es cierto que las predicciones de los precios no son infalibles, el valor añadido de la arquitectura LSTM para esta clase de problemas es su capacidad para detectar patrones en temporalidades cortas y largas, por lo que a pesar de que esté lejos de ser una arquitectura perfecta, es probablemente una de las más adecuadas a la hora de detectar y predecir las tendencias.

## 4.2. Conclusiones

El proyecto demuestra la viabilidad de utilizar un enfoque basado en aprendizaje automático para la predicción de tendencias del mercado financiero. La integración de tecnologías diversas permitió la creación de un sistema robusto y eficiente.

Se ha logrado obtener dos modelos con un *accuracy* aceptable. Estos son elegidos para ser guardados en un servidor y de forma automatizada ejecutar un archivo Python que carga el modelo, hace las nuevas predicciones, y estos son guardados en Elasticsearch. Todos los datos guardados en base de datos, se consumen por una aplicación web que muestra los datos y resultados.

El resultado obtenido es el esperado. Llegar a un mínimo producto viable, o en otras palabras, obtener un modelo aceptable. Superando expectativas, se ha montado un servidor donde se aloja Elastic Stack para guardado y consulta de datos, aparte de montar una aplicación que también se encuentra en producción. Se ha comenzado con hacer toda la conexión de los diferentes sistemas y crear las automatizaciones. Pero el tiempo se agota y faltan pinceladas por dar. Aún con todo esto, el objetivo principal de esta investigación se ha logrado, más aparte se han demostrado todas las habilidades obtenidas durante el curso.

## 4.3. Conclusiones individuales del proyecto

### Adrián Martínez:

Al inicio del proyecto, se había previsto realizar web scraping o utilizar alguna API para obtener noticias, con la intención de aplicar un modelo de análisis de sentimientos. Sin embargo, también se exploró el método Wyckoff y se desarrolló un código para intentar detectar sus fases, con la posibilidad de mejorar la efectividad del modelo en conjunto.

Se ha encontrado difícil realizar web scraping en ciertos sitios web, lo que ha limitado la cantidad de noticias disponibles para el análisis. Se había planeado obtener un mayor volumen de noticias de lo actualmente disponible. Además de esto se utilizó unos algoritmos para poder sacar las tendencias de Wyckoff, pero tras varios experimentos y demostración de resultados a algunos expertos, no se consiguió distinguir las diferentes fases de las tendencias de dicho método.

Para abordar esta situación, se logró encontrar una solución a través de un proceso de prueba y error.

El mínimo producto a mostrar consistiría en un dataset que contenga al menos 50 noticias, cada una con una puntuación asignada por el análisis de sentimientos.

### Gabriel Mailat

El objetivo principal ha sido lograr elaborar una herramienta que aportase utilidad de cara a los usuarios que quieran obtener rentabilidad en los mercados financieros.

Los contratiempos incluyeron la complejidad inherente del proyecto, que demandaba constantes ajustes a medida que surgían nuevas ideas, lo que dificultaba alcanzar los objetivos en el plazo de un mes. Además, la asistencia a clases también consumía parte del tiempo disponible para el proyecto.

Una solución viable sería optar por trabajar de forma remota, evitando así la necesidad de asistir físicamente a clase y ahorrando tiempo en desplazamientos, lo que permitiría dedicar más tiempo al proyecto. Además, otra opción sería priorizar los aspectos esenciales del proyecto, descartando temporalmente los elementos secundarios. Una vez completadas las tareas principales, se podría considerar abordar las cuestiones secundarias si el tiempo lo permite.

También cabría mencionar que para lograr una herramienta sofisticada y altamente fiable requiere una inversión significativa de tiempo y recursos económicos. A la

hora de realizar el entrenamiento de los modelos se ha invertido mucho tiempo en la ejecución de cada uno de los experimentos y dado que la versión gratuita de Google Colab ofrece unos recursos bastante limitados,

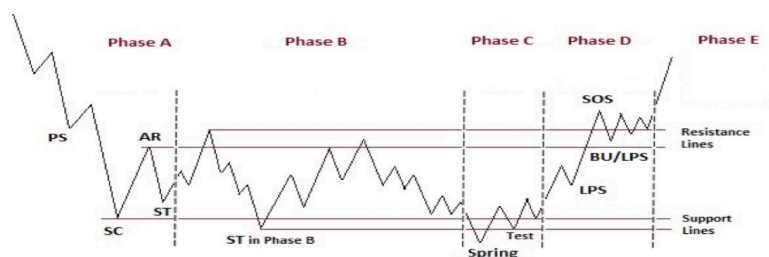
Your session crashed after using all available RAM. X

a modo de experimento se ha utilizado como alternativa el alquiler de GPUs en la nube, concretamente en Vast.io

Con una potencia computacional de aproximadamente 300 teraflops el mismo experimento previamente realizado en Google Colab donde había durado 10 minutos, en la plataforma de Vast.io se ejecutó en menos de un minuto. Los costes de las GPUs son variados aunque para el uso de los casi 300 teraflops el coste oscilaba entre los 2 y los 5 euros por hora.

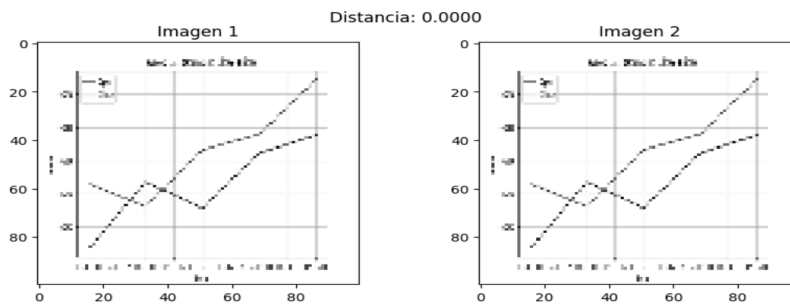
m:23898 host:135125	MZ52	↑2192 Mbps ↓3787 Mbps	1000 ports	verified	\$4.603/hr
<b>4x L40S</b>	PCIe 4.0, 16x 12.0 GB/s			Max Duration 24 days	
<b>293.1 TFLOPS</b>	AMD EPYC 7313 ...	nvme	<b>406.5 DLPerf</b>	Reliability 98.2%	<b>RENT</b>
45 GB Max CUDA: 12.4	32.0/64 cpu 258/516 GB	2575 MB/s 3202.4 GB	88.3 DLP/\$/hr		

Otra de las barreras que se tuvieron que afrontar durante el desarrollo del proyecto fue la detección de patrones clave en el análisis técnico de los mercados, como podría ser el patrón de Wyckoff.



Conocer en cuál de las fases del patrón de Wyckoff se encuentra el mercado supone una gran ventaja, ya que esto implica aumentar las probabilidades de realizar operaciones con éxito, puesto que de alguna manera se conocería de antemano cuál sería la próxima fase.

Para abordar este problema, a modo de experimento se implementó un algoritmo basado en una arquitectura convolucional que encontrase la similitud entre imágenes, por lo que a partir de una gráfica se pretendía detectar la similitud con otra gráfica en la que estuviera presente el patrón de Wyckoff.



*Similitud entre imágenes mediante el uso de la distancia euclidiana (a menor distancia, mayor similitud).*

Sin embargo, debido a la escasez de tiempo y recursos, todavía no ha resultado ser posible materializar esta idea de manera sólida, ya que no se tiene acceso a un conjunto de datos con gráficas que sirvieran como ejemplo para una arquitectura convolucional que aprendiese el patrón de Wyckoff y así poder detectar sus fases casi en tiempo real en una gráfica de cualquier activo financiero.

## Vicente Real

El propósito principal de este proyecto ha sido desarrollar una herramienta útil para los usuarios que buscan obtener beneficios en los mercados financieros. No obstante, el proyecto ha encontrado varios retos, principalmente por su complejidad y la necesidad de ajustes constantes a medida que surgían nuevas ideas. Esto ha dificultado el logro de los objetivos en el tiempo previsto de un mes.

A pesar de los retos encontrados, se ha logrado desarrollar una herramienta útil para los usuarios que buscan obtener rentabilidad en los mercados financieros. La utilización de tecnologías modernas como Astro, TypeScript y Tailwind CSS ha permitido crear una interfaz de usuario eficiente, robusta y atractiva. Se espera que esta herramienta siga evolucionando y mejorando en el futuro, ofreciendo aún más valor a sus usuarios.

## Juan Luján

Se desarrolló un proceso estructurado para encontrar un modelo con alta capacidad predictiva para el mercado de valores. Esto incluyó la investigación sobre la obtención de datos necesarios, reuniones con un experto en bolsa para comprender las necesidades del mercado, la comunicación de ideas y patrones esenciales al equipo, y la investigación específica del modelo LSTM, junto con la exploración de otras metodologías, como modelos de Machine Learning, para capturar las temporalidades relevantes y de predicción. Finalmente, se llevaron a cabo experimentos con los nuevos modelos, variando hiperparámetros y visualizando los resultados en gráficas para evaluar su eficacia.

Los contratiempos encontrados incluyeron la dificultad para obtener una visión global del funcionamiento del mercado de valores, lo que requirió una extensa lectura de documentación y la consulta de expertos para comprender los conceptos. Además, a nivel de modelos, se encontraron ejemplos que no funcionaban como se esperaba, con falta de claridad en sus objetivos, junto con la presencia de librerías obsoletas. También surgió la dificultad de determinar qué campos eran relevantes para el entrenamiento del modelo. La transformación de senos y cosenos y la manera de hacer que tuviera una continuidad la gráfica cuando nos encontrábamos en saltos de datos de los fines de semana o festivos.

La solución implicaría aprovechar la documentación disponible, buscar apoyo tanto en compañeros como en expertos, llevar a cabo experimentos para explorar diferentes enfoques y utilizar herramientas de inteligencia artificial para optimizar el proceso.

El mínimo producto viable consistiría en un modelo cuya predicción sea suficientemente válida y relevante para el objetivo del Trabajo de Fin de Máster.

## Edvard Khachatryan Sahakyan

En una primera instancia, los objetivos personales eran la organización y estructuración del proyecto. Dirección del equipo y fijar los objetivos individuales. Análisis de las diferentes aplicaciones web que ofrecen datos para el entrenamiento de la “serie temporal” que predecirá el futuro de un activo en concreto. Entrenamientos de varios modelos, e investigación de cómo poder mejorarlos.

Se han encontrado varias limitaciones de los recursos en Colab, han sido el mayor contratiempo, ya que se podría estar dedicando horas, y por este motivo, encontramos una limitación muy grande.

Se ha creado una segunda cuenta de “Google” para duplicar el tiempo de “GPU” permitido, y al acabar en una cuenta, se ha aprovechado la segunda. Por otro lado, se ha reducido el número de entrenamientos de prueba. Si se ha considerado un entrenamiento innecesario o sin sentido, no se ha ejecutado.

El mínimo producto a mostrar incluirá un modelo de regresión LSTM con la mejor puntuación posible, así como la visualización de los datos utilizando Comet para ofrecer una representación clara y efectiva de los resultados obtenidos.

## 5. Planes de futuro

Hacer predicciones de un día entero por minutos (959 min) que van desde que tenemos datos 4:00 hasta que cierra el día financiero 19:59 y así poder ejecutar el modelo una vez se tiene los datos del día anterior y poder mostrar los del día en curso.

Actualmente, tenemos el modelo entrenado con el histórico de datos y las noticias que hacen referencia a Apple, se tendría como objetivo hacer entrenamiento de otras compañías con la intención de hacer sus predicciones.

Conseguir noticias de las compañías que se quieren hacer la predicción por días desde el día y año que se quiere entrenar el modelo de predicción.

Construir una API robusta para que las peticiones vayan a través de ella con un sistema de logueo de usuarios según si son suscriptores normales o premium.

## 6. Referencias y Bibliografía

- Artículos y libros sobre aprendizaje automático y análisis de datos.
- Documentación oficial de las tecnologías utilizadas (Pandas, Tensorflow, etc.).
- Sitios web relevantes y publicaciones académicas:
- <https://github.com/iambharathvaj/Stock-Price-Prediction/blob/master/Apple%20-%20Stock%20Price%20Prediction.ipynb>
- <https://medium.com/@kunalchhablani14/predicting-stock-prices-using-machine-learning-338ab2fe4e5b>
- <https://www.kaggle.com/code/faressayah/stock-market-analysis-prediction-using-lstm>
- Chat GPT

## 7. Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a todas las personas que han contribuido a la realización de este proyecto. En primer lugar, a Vicent Tortosa, cuya orientación y apoyo han sido fundamentales, así como a Chelo Richart y Sergio Aparicio por su ayuda y experiencia. Finalmente, extendemos nuestro agradecimiento a nuestros amigos traders por las enriquecedoras charlas que han compartido con nosotros. Sin ellos, este proyecto no habría sido posible.



edu presentación:

De manera muy resumida, podemos ver un servidor, dónde se va a alojar toda la arquitectura bigdata, y el mismo proceso ETL en ella. Los modelos ya entrenados consumiran de un elasticsearch que hay en ella, y usando logstash inyectarán las predicciones de vuelta en elasticsearch.

Por medio de templates para su consumo, la aplicación web consumirá estas predicciones y las imprimirá. Al igual que las noticias.

La obtención de datos viene de diferentes puntos como yahoofinance, alphavantage, y webscrapping de varias paginas de noticias, estos datos se tratan, pasan por todo el proceso de analisis exploratorio de datos, su visualización y al final su inyección a elasticsearch.

Este proceso se produce cada día a las 04:00 de la mañana.

—

De manera más gráfica, aquí encontramos la aruitectura del servidor. En la maquina se encuentran los pesos de los modelos guardados en formato .h5. Para predecir consumiran los datos guardados del proceso de EDA. y posteriormente arrancaran contenedores efimeros de logstash, los cuales harán la inyección de datos. Posteriormente el consumo de estos datos para la app y por kibana para visualizarlos.

—

Por ultimo se ha añadido a está investigación una tecnología interesante, que es COMET. Está nos facilita mucho la visualización de datos de los entrenamientos de nuestros modelos, rendimientos y evaluaciones. Se ha aplicado para el entrenamiento de modelos por minutos, en 3 entrenamientos distintos y este es el resultado.