PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE HÁBITAT, INFRAESTRUCTURA Y CREATIVIDAD

CARRERA DE SISTEMAS DE INFORMACIÓN

TEMA:

ANÁLISIS PREDICTIVO DE LA SITUACIÓN PETROLERA PARA EXPORTACIÓN DEL ECUADOR

AUTOR

OMAR ALEJANDRO LOZADA SAÁ

TUTOR

HENRY NELSON ROA MARIN

QUITO JUNIO, 2026

DEDICATORIA

Dedico estas páginas a todo aquel que me mostro su apoyo en este camino que elegí desde mi mamá que fue la persona que hizo esfuerzos para abrirme el camino que luego me tocaba a mi trabajar, junto con mi abuelo que han sido aquellos dos los que forman parte de mi familia nuclear y de sangre que han apoyado, mi abuelo Antonio que en paz descanse. Pasando por amigos que se convirtieron en familia (Andrés y Moshe), aquellos que sin entender de términos técnicos me escuchaban y me mostraban su apoyo. Por último, a mi "yo niño" por no darse por vencido en ningún momento ni cuando el día era tan oscuro como la misma noche, por soñar en grande a pesar de que algunos dijeran "solo pasa en la compu", por no perder la sonrisa tan característica a pesar de los malos días, por no perder el brillo que ayudaba o motivaba a alguien más, por abrazar las caídas y permitir que las mismas formen parte del crecimiento.

Cuando vuelvas la vista atrás

Sonreirás al verte tan valiente y capaz.

Pequeño soñador, nunca dejes de soñar

Tu historia comienza, el mundo hay que conquistar.

AGRADECIMIENTO

Agradezco a quienes, me brindaron su labor educativa y aquellos que me abrieron las puertas para ingresar en el mundo de los sistemas de información y todo lo que ello conlleva. Todos aquellos que en algún momento me compartieron un granito de arroz de su basto conocimiento les estaré eternamente agradecido, este conocimiento no se refiere solo a lo teórico, sino todo lo técnico y también lo no técnico, aquellas situaciones que hay que saber afrontar y no se encuentran ejecutando un código, leyendo un libro o preguntándole a chatgpt, esas enseñanzas se valoran un montón.

Pasando con el lado profesional, prácticas y participación en proyectos, infinitas gracias por confiar en mí en y algunos casos mi equipo de trabajo, fueron experiencias que de igual manera me ensañaron mucho más de lo que dice un syllabus, compartir con áreas distintas a la mía y presentar resultados de calidad.

Mencionando a mi equipo de trabajo, no puede faltar a mis compañeros desde el día uno y siempre me han brindado su mano, Sebastián Jacome y Antonio Proaño. Gracias al resto de compañeros que a su vez me mostraron su confianza para trabajar con ellos y en muchos casos confiar en mí para liderar proyectos académicos.

Y por último a mi tutor de integración curricular por su activo apoyo y guía en el desarrollo de este proyecto de integración curricular.

RESUMEN

La situación geopolítica que atraviesa el mundo afecta directamente al área de hidrocarburos en Ecuador en especial el petróleo. Durante el presente año hubo conflictos que generaron un cambio en los precios del petróleo.

En el presente trabajo se analiza estas situaciones para poder generar predicciones que serán relevantes para una debida toma de decisiones por parte de la entidad encargada, que en este caso es la empresa pública Petroecuador como figura que servirá como punto central del análisis de minería de datos bajo la metodología CRISP-DM. Los algoritmos usados dentro de la metodología son *ARIMA*, *XGBoost*, *LSTM* y un modelo híbrido de *ARIMA + XGBoost*. Los datos que se van a usar son datos de un repositorio abierto que cuenta con registros de los precios del WTI desde 1986 pero de manera selectiva se va usar para el modelado solo los últimos cinco años para tomar el comportamiento postpandemia del COVID-2019. Dentro de los procesos a realizar corresponde a la ingeniería de características generando nuevas columnas que se usan frecuente en el ámbito financiero y análisis de activos en la bolsa de valores.

Al finalizar este trabajo se presentarán conclusiones justamente basados en la predicción realizada en el proceso y estas mostrarán de manera interpretativa el futuro comportamiento del precio del petróleo.

ÍNDICE

ÍNDICE DE FIGURAS, GRAFICOS Y TABLAS 8

ÍNDICE DE FIGURAS..... 8

ÍNDICE DE TABLAS 11

CAPÍTULO I: INTRODUCCIÓN.....12

1. Introducción 12

1.1 Justificación..... 12

1.2 Planteamiento del problema 12

1.3 Objetivos 13

1.3.1 Objetivo general..... 13

1.3.2 Objetivos específicos..... 13

1.4 Alcance..... 13

CAPÍTULO 2: MARCO TEÓRICO.....15

2. Marco Teórico 15

2.1 Información relevante sobre el petróleo..... 15

2.1.1 Glosario 15

2.1.2 Tipos de Crudo..... 16

2.1.3 Precios internacionales 17

2.1.4 EP Petroecuador..... 17

2.2 Metodología CRISP-DM..... 18

2.3 Modelado y algoritmos a usar 19

2.4 Herramientas y librerías para ciencia de datos con python. 21

2.4.1 Preprocesamiento y manejo del flujo de datos 21

2.4.2 Visualización de gráficos 21

2.4.3 Modelos 22

2.4.4 Métricas 23

2.5 Consideraciones para el plan preliminar de despliegue 23

CAPÍTULO 3: DESARROLLO DEL PROTOTIPO 24

3. Desarrollo del prototipo 24

3.1 Entendimiento del negocio 24

3.2 Entendimiento de los datos 25

3.3 Preparación de los datos 33

3.4 Modelado 37

3.4.1 Modelo 1: ARIMA 41

3.4.2 Modelo 2: XGBoost 43

3.4.3 Modelo 3: LSTM 45

3.4.4 Modelo 4: Híbrido, ARIMA +XGBoost 55

3.5 Evaluación 65

3.6 Despliegue 68

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES 72

4. Conclusiones y Recomendaciones 72

4.1 Conclusiones 72

4.2 Recomendaciones 74

BIBLIOGRAFIA 75

ÍNDICE DE FIGURAS, GRAFICOS Y TABLAS

ÍNDICE DE FIGURAS

Ilustración 1 *Ciclo CRISP-DM* 19

Ilustración 2 *Carga del dataset desde la API de FRED* 25

Ilustración 3 *Cabecera y cola del dataset* 26

Ilustración 4 *Tipo de datos de las columnas iniciales del dataset* 27

Ilustración 5 *Renombrar columnas del dataframe* 27

Ilustración 6 *Corrección del tipo de datos correctos de las columnas.* 28

Ilustración 7 *Dimensión y nulos de dataframe* 28

Ilustración 8 *Diagrama de caja y bigotes* 29

Ilustración 9 *Histograma de precios del WTI* 30

Ilustración 10 *Valor negativo en el dataset* 31

Ilustración 11 Valores nulos del dataset 32

Ilustración 12 *Nulos por semana del dataset* 33

Ilustración 13 *Forward Fill y Backward Fill* 34

Ilustración 14 *Creación de features* 36

Ilustración 15 *Serie de tiempo del precio del WTI, Original y con EMA12* 37

Ilustración 16 *División del dataframe en entramiento y testeo como ejemplo* 39

Ilustración 17 *Lista con todas las features* 39

Ilustración 18 *Asignar la variable predictora y objetivo* 40

Ilustración 19 *División del dataframe en entrenamiento y testeo para X y Y* 40

Ilustración 20 *Línea de tiempo con la división de entrenamiento y testeo* 41

Ilustración 21 *Creación e hiperparámetros de ARIMA* 42

Ilustración 22 *Entrenamiento, generar predicciones e indexación del modelo ARIMA*... 42

Ilustración 23 *Creación e hiperparámetros de XGBoost*..... 44

Ilustración 24 *Entrenamiento y generar predicciones de XGBoost*..... 44

Ilustración 25 *Escalado de las features y el precio para LSTM*..... 45

Ilustración 26 Función de creación de secuencias para LSTM 46

Ilustración 27 *Llamado de la función para generar arreglos de X y Y en LSTM*..... 47

Ilustración 28 *Asociación de índice de fechas en LSTM*..... 47

Ilustración 29 *Registro de fecha para división del dataframe según el índice para LSTM* 48

Ilustración 30 *Dividir el dataframe en entrenamiento y testeo para LSTM* 48

Ilustración 31 *Arquitectura y creación del modelo LSTM*..... 50

Ilustración 32 *Compilación del modelo LSTM* 51

Ilustración 33 *Creación de callbacks para LSTM* 52

Ilustración 34 *Entrenamiento del modelo LSTM* 53

Ilustración 35 *Proceso de enteramiento del modelo LSTM*..... 54

Ilustración 36 *Generar predicciones escaladas para LSTM* 54

Ilustración 37 *Desnormalización de los valores de la predicción LSTM*..... 55

Ilustración 38 *Arquitectura del modelo híbrido*..... 56

Ilustración 39 *Ponderación del modelo híbrido* 57

Ilustración 40 *Generar predicciones del modelo de ARIMA para el modelo híbrido*..... 57

Ilustración 41 *Calculo de residuales a partir de ARIMA* 58

Ilustración 42 *Recrear las features de XGBoost para el híbrido* 59

Ilustración 43 *Alienación de índice con las predicciones de los residuales*..... 60

Ilustración 44 *Creación del dataframe con los de entrenamiento* 61

Ilustración 45 *Creación del dataframe con los de prueba* 62

Ilustración 46 *Creación del modelo XGBoost para el híbrido* 63

Ilustración 47 *Generar predicciones XGBoost sobre los residuales* 64

Ilustración 48 *Generar predicciones del híbrido basado en ponderaciones* 64

Ilustración 49 *Función para calcular las métricas de cada modelo* 65

Ilustración 50 *Línea de tiempo con la comparativa de cada modelo* 66

Ilustración 51 *Línea de tiempo con la comparativa de cada modelo con zoom del último año* 67

Ilustración 52 *Plan preliminar de despliegue* 69

Ilustración 53 *Endpoint para realizar la predicción en el plan preliminar* 70

Ilustración 54 *Dashboard sobre la situación petrolera del Ecuador* 71

ÍNDICE DE TABLAS

Tabla 1 *Tipos de crudo en relación con su densidad* 16

Tabla 2 *Features de XGBoost* 34

Tabla 3 *Algoritmos seleccionados para modelado* 37

Tabla 4 *Métricas de evaluación de cada modelo* 65

TEMA:

ANÁLISIS PREDICTIVO DE LA SITUACIÓN PETROLERA PARA EXPORTACIÓN DEL ECUADOR

CAPÍTULO I: INTRODUCCIÓN

1. Introducción

1.1 Justificación

El petróleo es considerado el oro negro en nuestra economía actual no solo a nivel a país sino a nivel global. A pesar de su importancia su aporte a la economía de Ecuador ha ido en descenso tanto por factores internos o externos que han afectado directamente a la economía del país. El análisis permitirá evaluar las acciones tomadas por parte del sector petrolero, así como sus acciones futuras.

1.2 Planteamiento del problema

El petróleo era el principal producto de exportación hace unos años, pero con el paso de tiempo las exportaciones no petroleras han tomado protagonismo en especial este último año. La principal razón detrás de este descenso es la erosión del rio Coca, pero viene acompañado de otras razones como lo son la falta de mantenimiento, eventos catastróficos como la pandemia por COVID-19 y los apagones del 2024.

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar un modelo de análisis predictivo, aplicando la metodología CRISP-DM sobre datos históricos del precio del petróleo ecuatoriano, que permita proyectar el comportamiento futuro del sector y apoyar a la toma de decisiones estratégicas en las entidades del sector público energético.

1.3.2 Objetivos específicos

1. Recopilar, limpiar y analizar datos históricos del precio del petróleo ecuatoriano provenientes de fuentes oficiales, aplicando las fases de comprensión y preparación de datos de la metodología CRISP-DM.
2. Construir y evaluar modelos predictivos sobre el comportamiento futuro de los precios del crudo ecuatoriano, comparando su desempeño mediante métricas de precisión estadística y seleccionando el modelo de mayor capacidad explicativa.
3. Desarrollar un *dashboard* interactivo que visualice los indicadores históricos del sector petrolero y las proyecciones generadas por el modelo, orientado a apoyar la formulación de decisiones estratégicas en el Ecuador.

1.4 Alcance

El presente trabajo de titulación comprende el desarrollo de un prototipo de análisis predictivo aplicado al sector petrolero ecuatoriano, siguiendo las seis fases de la metodología CRISP-DM. El prototipo integrará la recopilación y preparación de datos históricos públicos de producción, precios y exportaciones de crudo, la construcción y evaluación de modelos predictivos, y la visualización de resultados mediante un *dashboard* interactivo.

El prototipo tendrá carácter estrictamente demostrativo y académico. No se contempla su despliegue en un entorno de producción ni su integración con sistemas informáticos externos. Las proyecciones generadas constituirán escenarios exploratorios con fines analíticos, no pronósticos oficiales.

CAPÍTULO 2: MARCO TEÓRICO

2. Marco Teórico

2.1 Información relevante sobre el petróleo

El petróleo como producto de exportación tiene una gran relevancia en la economía del país, se habla del producto de exportación insignia del Ecuador durante varios años.

2.1.1 Glosario

Es necesario recalcar y diferenciar conceptos que se mencionan en el presente proyecto.

- Petróleo: Hace referencia de manera general al crudo y sus derivados, aunque este último uso no es del todo apropiado.
- Crudo: Materia prima que se extrae del suelo para combustibles y otros usos con sus derivados.
- West Texas Intermediate (WTI): Es el precio marcado en base al precio por barril del petróleo de campos petrolíferos de Texas-USA.
- American Petroleum Institute (API): Indicador de la densidad del crudo de petróleo.
- Crude Oil Prices West Texas Intermediate – Cushing, Oklahoma (DCOILWTICO): Código de identificación del repositorio de FRED para los precios del WTI.
- Precio Oficial de Venta (POV): Es el precio con el que Ecuador realiza sus acuerdos económicos para exportación, este valor se basa en el WTI, pero con ciertas penalizaciones.

2.1.2 Tipos de Crudo

Ecuador exporta dos tipos de Crudo que son el Oriente y Napo. Su diferencia es el grado de densidad API que se considera en cada uno. Mientras que el crudo Oriente se encuentra en valores que van entre 22.2 y los 24.2 grados, el crudo Napo se encuentra entre los 16.5 y los 18.5 grados.

Es importante marcar la diferencia entre los distintos tipos de crudo ya que según su densidad se le otorga un mayor o menor valor en el mercado. A un mayor valor de grados de densidad se lo considera más ligero por tanto valor de mercado, para ejemplificar mejor en la siguiente tabla se puede diferenciar los tipos de Crudo que existen de manera general y como se les cataloga alrededor del mundo.

Tabla 1

Tipos de crudo en relación con su densidad

Aceite Crudo	Densidad Grados API
Extrapesado	10.0
Pesado	10.0 - 22.3
Mediano	22.3 - 31.1
Ligero	31.1-39
Superligero	>39

Nota. Adaptado de *Análisis de las propiedades de calidad de un crudo* (p. 9), por E. G. de la Cruz Amador, 2016.

Una vez retratada la diferencia se puede notar que el Crudo Oriente es el que mayor valor entrega y de hecho es el que se produce en mayor medida.

2.1.3 Precios internacionales

Como vimos Ecuador posee dos tipos de crudo de petróleo, pero su precio es menor al de otros crudos alrededor del mundo. Los precios que por lo general se manejan para la comercialización de los crudos es el WTI y el Brent, siendo el primero el que se maneja para la exportación del crudo ecuatoriano, pero con ciertos matices.

El primero es que existen varios precios a tener en cuenta partiendo del propio WTI que funciona como referencia global del precio y va fluctuando de minutos, horas, días, etc.; a partir de este se debe tomar en cuenta el precio spot del WTI que es el precio que se va a usar de base para realizar acuerdos y transacciones un día y hora especifico. De este se reduce el costo de logístico y de transporte que son costos menores y más consistentes en el tiempo, luego se pasa al precio de venta oficial o POV que es valor con el cual Petroecuador comercializa los dos tipos de crudo ecuatoriano (por separado). Aquí es donde el precio presenta diferencias con respecto al del WTI con ciertos descuentos que van del 5% al 15%. Para ejemplificar si el precio del WTI esta con valores de mercado entre $88 y $95, puede que el precio spot se concentre en $90 y aplicando un descuento del 15% incluyendo gastos logísticos, el precio oficial de venta es de $76.5 por barril de crudo.

2.1.4 **EP Petroecuador**

La empresa pública de hidrocarburos Petroecuador es la encargada de las actividades de referentes a todo el proceso que concierne al manejo de hidrocarburos del territorio nacional, desde su extracción, transporte hasta su refinación y comercialización.

Su finalidad como empresa es generar riquezas a partir de los recursos energéticos, pero de manera sostenible y responsable con el ambiente. De acuerdo con la plataforma oficial de EP-Petroecuador su misión es “Maximizar de manera sustentable el valor de los recursos energéticos

para el beneficio de la sociedad ecuatoriana" (EP Petroecuador, s.f., párr.2). Y su visión "Ser la empresa referente a nivel regional que refuerza la seguridad, eficiencia y de desempeño energético con criterios de transparencia, probidad y compromiso social." (EP Petroecuador, s.f., párr.3).

2.2 Metodología CRISP-DM

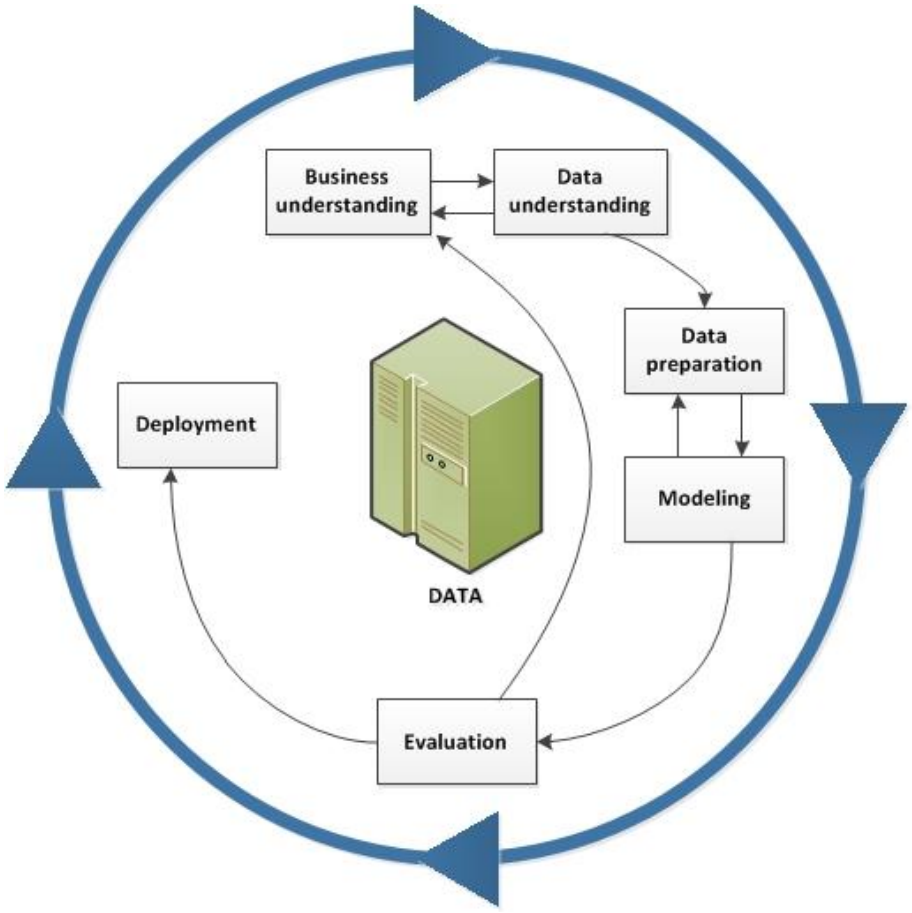
La metodología usada en este proyecto es Cross-Industry Standard Procees for Data Mining, una de las metodologías más usadas en el área de análisis de datos y machine learning. Esta consta de seis fases que son entendimiento del negocio, entendimiento de los datos, preparación de los datos, modelado, evaluación y despliegue.

Una característica relevante de CRISP-DM es la flexibilidad con la que se puede desarrollar, dando la posibilidad de regresar a ciertas fases de la metodología de manera coherente, por ejemplo para este trabajo su tuvo que realizar varios cambios en el proceso tanto, haciendo énfasis en dos fases que generaron cambios en el proceso, la fase de entendimiento de los datos ya que encontrar un *dataset* que cumpla con las cualidades necesarias para alinear al negocio fue un trabajo de prueba y error, por otro lado la fase de evaluación generaba incertidumbre en el rumbo del trabajo por lo que en dos ocasiones se reestructuro el prototipo casi desde cero.

CRISP – DM cumple con el objetivo de este prototipo al tener en cuenta tanto el negocio como la parte de un uso post modelado, que serían la fase uno y seis respectivamente.

Ilustración 1

Ciclo CRISP-DM



Nota. Tomado de "*CRISP-DM Help Overview*", por IBM SPSS Modeler Subscription, 2021 (https://www.ibm.com/docs/es/spss-modeler/saas?topic=dm-crisp-help-overview).

2.3 Modelado y algoritmos a usar

Para este proyecto se va a usar un tipo de modelado predictivo, basada en líneas de tiempo. Se va a tomar de base 3 algoritmos de *machine learning* los cuales son:

- *ARIMA*:

Este es un modelo de media móvil autorregresivo es decir que toma los propios valores de la línea de tiempo para generar predicciones, no requiere de valores adicionales. Simplemente

requiere especificar de 3 hiperparámetros. Los cuales corresponden al parámetro de *order*, y estos son *p, d y q*. Donde p es la cantidad de valores pasados que va a usar; d tiene valores predeterminados para establecer si existe estacionalidad; q es el número de errores que toma para corregir la predicción. Existen otros 12 hiperparámetros de la clase del algoritmo pero que si no se modifican permanecen en valores por default y su modificación se asocia o modelos más complejos como SARIMA o SARIMAX por lo que no es relevante mencionar para este proyecto que no contemplo esas variaciones.

- *XGBoost*:

Este es un algoritmo de *machine learning* basado en árboles de decisión optimizado con gradientes que funcionan para ser más eficientes que los árboles de decisión. Lo optimización de gradiente consiste en generar predicciones “débiles” en base a un primer árbol al inicio y a partir de los denominados residuos (los errores que tuvo el modelo), luego el modelo genera un nuevo árbol con otras predicciones, esto hasta un número concreto de veces especificado en los hiperparámetros. Para terminar con la predicción final.

- *LSTM*:

Es un algoritmo de redes neuronales recurrentes, con un tipo de aprendizaje profundo capaz de pasar de un dato de entrada, realizar el aprendizaje mediante capas ocultas donde se encuentra las “neuronas” las cuales se encargan del aprendizaje de patrones y como resultado mediante una capa de salida se obtiene la predicción. LSTM (Long Short-Term Memory) como su nombre lo indica es un algoritmo de corto y largo plazo, de tal manera que aprende los patrones recientes y los patrones antiguos de la línea de tiempo.

2.4 Herramientas y librerías para ciencia de datos con python.

Python es un lenguaje de programación de alto nivel ampliamente usado para la ciencia de datos como es el del presente caso, que consta de varios herramientas y librerías que ayuden directamente a esta rama. Este lenguaje permite una instalación rápida de mucho de este contenido.

2.4.1 Preprocesamiento y manejo del flujo de datos

Para la parte del manejo de los datos existen varias herramientas y librerías que se usaron que son:

- Pandas: Herramienta para el análisis y manejo de datos. Crear data frames; obtener, limpiar y prepara los datos.
- Numpy: Herramienta que permite realizar operaciones matemáticas, así como funciones y arreglos, estas se integran directamente al preprocesamiento de datos del proyecto.
- Json: Librería oficial para el manejo de archivos del propio nombre, en este caso del proyecto para escribir algunos metadatos en .json.
- Pickle: Librería oficial para guardar y cargar los modelos generados como objetos únicos durante el proyecto.

2.4.2 Visualización de gráficos

La visualización con gráficos de información relevante es imprescindible en algunas de las fases del CRISP-DM y la herramienta empleada es:

- Matplotlib: Librería enfocada en generar visualizaciones en base a los datos.

2.4.3 Modelos

Para el proceso de modelamiento cada modelo usa su librería específica y el predilecto para esto es la biblioteca scikit-learn, es especializada en machine learning. Al trabajar con líneas de tiempo también se trabaja con statsmodels. A continuación, todas las librerías usadas:

- statsmodels.tsa.arima.model.ARIMA: Librería para el pronóstico con Autoregressive Intgrated Moving Avarage (ARIMA)
- xgboost.XGBRegressor: Libreria para la implementación de gradient bosting.
- tensorflow.keras.models.Sequential: Librería encargada de agrupar por capas redes neoronales que usa LSTM.
- tensorflow.keras.layers.LSTM: Librería para una de las capas de Long Short-Term Memory (LSTM). Red neuronal recurrente para al aprendizaje de periodos de tiempo.
- tensorflow.keras.layers.Dense: Capa de red neuronal estándar que consta de parámetros de regularización.
- tensorflow.keras.layers.Dropout: Capa de red neuronal para evitar el overfitting donde se apaga aleatoriamente un número determinado de neuronas.
- tensorflow.keras.callbacks.EarlyStopping: Función que detiene el entrenamiento cuando detecta que las métricas no mejoran.
- tensorflow.keras.callbacks.ReduceLROnPlateau: Función que reduce la tasa de aprendizaje cuando detecta que las métricas no mejoran.
- tensorflow.keras.optimizers.Adam: Es un método de optimización que usa el algoritmo Adam para distribuir correctamente los pesos de una red neuronal.

2.4.4 Métricas

Para medir el rendimiento de los modelos se usan métricas que tienen sus propias librerías, en ambos casos para interpretar a breve rasgos entre menor sea el valor, mejor será el modelo.

- sklearn.metrics.mean_squared_error : Función para el cálculo del error cuadrático medio (MSE). El MSE es a breve rasgos el valor que mide en promedio los errores al cuadrado que tuvo cada modelo, por lo que tiende a penalizar bastante los errores más grandes y notorios.
- sklearn.metrics import mean_absolute_error: Función para el cálculo del error absoluto medio (MAE). El MAE mide el error que tiene el modelo, pero de manera más robusta antes los errores grandes, por lo que es útil para ver de manera más interpretativa y general la precisión del modelo.
- sklearn.metrics import r2: Métrica para determinar el coeficiente de determinación donde 1 es el mejor valor posible en el caso donde el modelo predice todos los valores dados. El coeficiente de determinación es el porcentaje que representa que tanto acierta el modelo basado en los errores que se comete con respecto al eje de la predicción.

2.5 Consideraciones para el plan preliminar de despliegue

Dentro del plan preliminar de despliegue se toma la parte de servicios para generar con pipelines un servicio que sea automatizado mediante FastAPI y la parte de visualización con PowerBI.

- PowerBI: Plataforma de Microsoft para el ambiente de Inteligencia de Negocios, con la visualización de datos y dashboards.
- FastAPI: Framework de desarrollo para generar APIs basadas en Python.

CAPÍTULO 3: DESARROLLO DEL PROTOTIPO

3. Desarrollo del prototipo

3.1 Entendimiento del negocio

Con lo empresa que se va a trabajar es EP Petroecuador. Estableciendo su misión y visión. Con información oficial extradida del portal web oficial.

Según el portal la misión es, “Maximizar de manera sustentable el valor de los recursos energéticos para el beneficio de la sociedad ecuatoriana” (EP Petroecuador, s-f-, pár.2).

Y su visión, “Ser la empresa referente a nivel regional que refuerza la seguridad, eficiencia y desempeño energético con criterios de transparencia, probidad y compromiso social” (EP Petroecuador, s-f-, pár.3).

Y con esta información generada se establece lo siguiente:

Objetivo estratégico: Optimizar costos de las actividades petroleras para maximizar los ingresos del Estado ecuatoriano.

KPI: Costos de producción anual / Ingreso total anual por comercialización de hidrocarburos.

Pasando a la segunda parte de la fase se requiere consultar la fuente a usar que en este caso es un repositorio de FRED (Federal Reserve Bank of ST. Louis) el cual consta de datos históricos diarios del precio Spot. El cual consta de 2 campos y aproximadamente 10500 registros. La primera columna es la fecha con día mes y año del precio. La segunda columna es el precio WTI representado en dólares estadounidenses.

3.2 Entendimiento de los datos

Para iniciar la segunda fase se debe considerar la API oficial de FRED para generar un *dataframe* a partir de los datos de este.

Ilustración 2
Carga del dataset desde la API de FRED

```
import pandas as pd

fred_url = "https://fred.stlouisfed.org/graph/fredgraph.csv?id=DCOILWTICO"
df = pd.read_csv(fred_url)
```

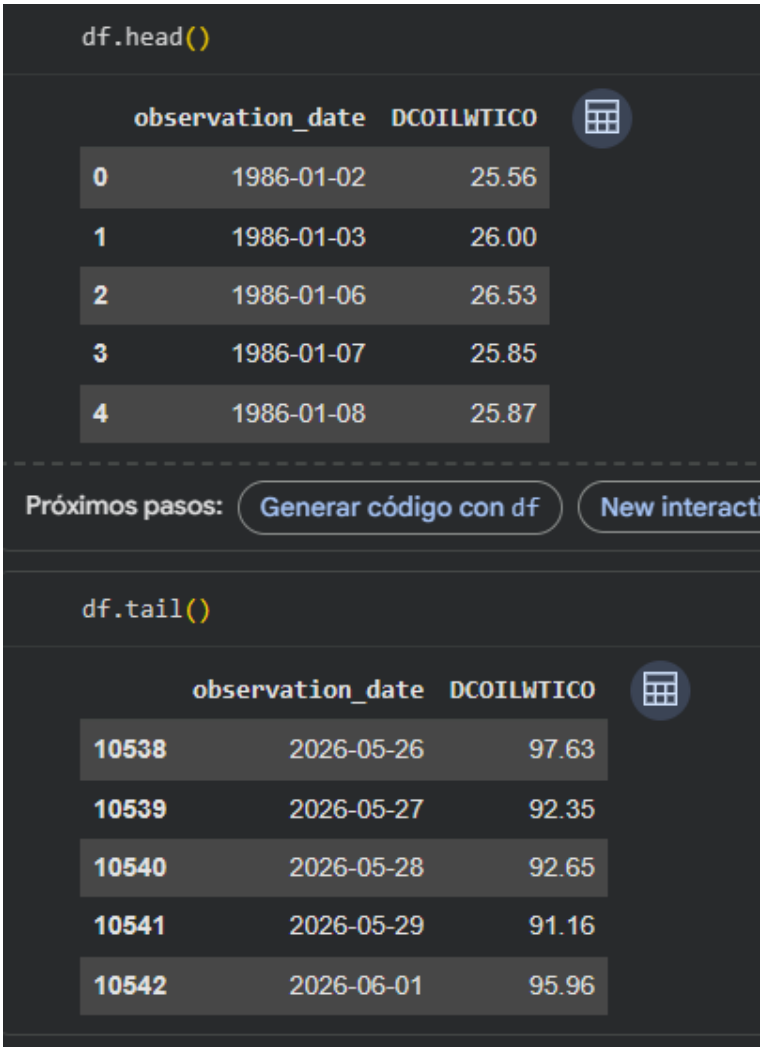
Nota. Autoría propia.

Para una consistencia en fechas se procede a usar un archivo .csv ya descargado como respaldo el cual consiste en los datos correspondientes desde el dos de enero de 1986 hasta el primero de junio de 2026.

Se comprueba que la cabecera este correcta y no exista inconsistencias iniciales tanto al inicio como al final usando *head* y *tail*.

Ilustración 3

Cabecera y cola del dataset

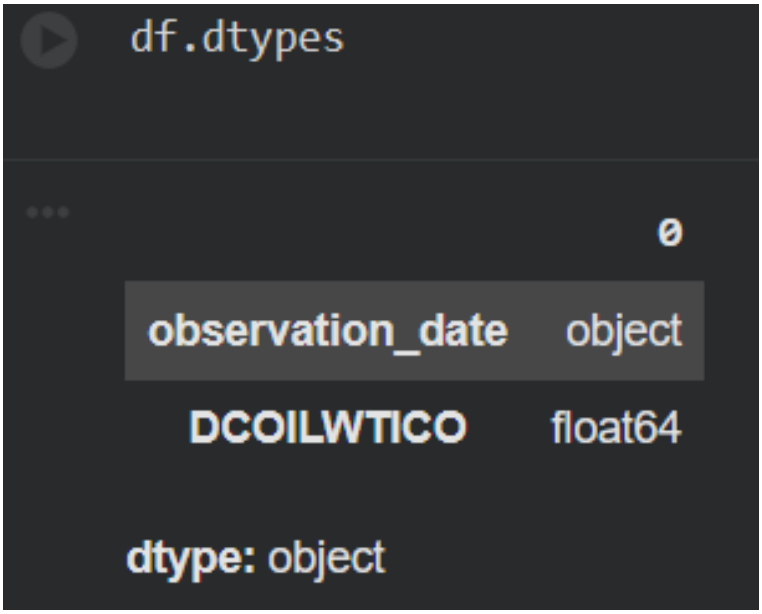


Nota. Autoría propia.

Se requiere comprobar el tipo de datos iniciales del *dataframe* para realizar correcciones según amerite, y de hecho la fecha no está como tipo de dato de fecha sino como objeto.

Ilustración 4

Tipo de datos de las columnas iniciales del dataset



Nota. Autoría propia.

Para tener una mejor guía y entendimiento de los datos se requiere renombrar las columnas a nombres más simplificados, *Date* y *Price* respectivamente.

Ilustración 5

Renombrar columnas del dataframe

```
df = df.rename(
    columns={"observation_date": "Date",
             "DCOILWTICO": "Price"})
```

Nota. Autoría propia.

A Date se le otorga el tipo de dato de fecha y se asegura que Price sea flotante ya que esto servirá para una mejor comprensión de los datos a posterior.

Ilustración 6

Corrección del tipo de datos correctos de las columnas.

```
df["Date"] = pd.to_datetime(df["Date"], errors="coerce")
df["Price"] = pd.to_numeric(df["Price"], errors="coerce")
df.head()
```

Nota. Autoría propia.

Para el análisis exploratorio de los datos se debe considerar los atributos mencionados *Date*(observation_date) y *Price* (DCOILWTICO). Por la naturaleza de los datos no existen columnas no relevantes ya que una actúa como predictora y la otra como target.

Hasta el **01/06/2026** que se ejecuta el presente notebook, el *dataframe* cuenta con 10543 registros, dos columnas y 371 nulos.

Ilustración 7

Dimensión y nulos de dataframe

```
print(df.shape)
print("Nulos:", df.isna().sum().sum())
```

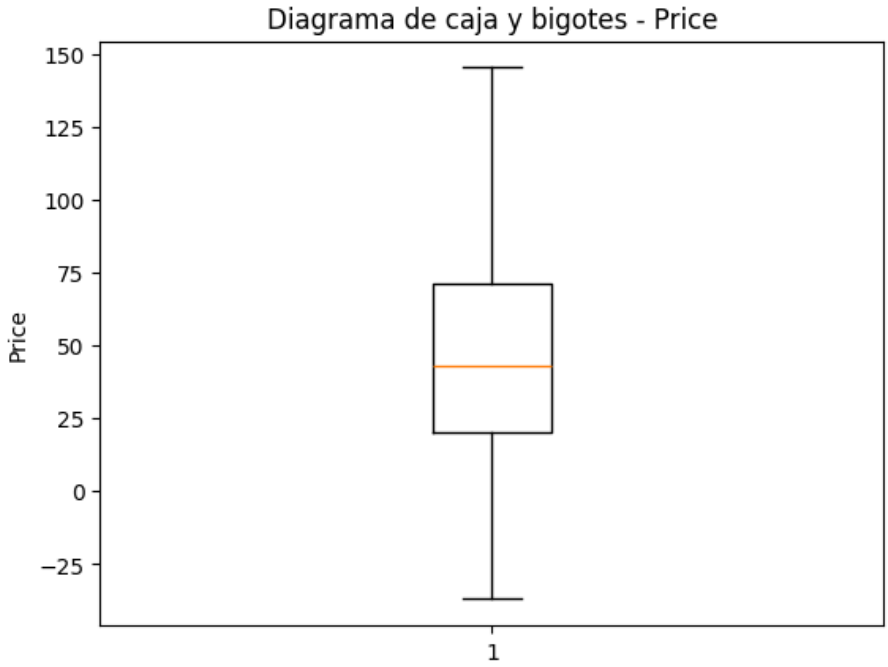
```
(10543, 2)
Nulos: 371
```

Nota. Autoría propia.

Pasando con la exploración de datos se da inicio con el diagrama de caja de bigotes donde se puede observar que no existen *outliers*, el 50% de valores son valores entre $75 y $20 aproximadamente.

Ilustración 8

Diagrama de caja y bigotes



Nota. Autoría propia a partir de datos de FRED. Generado con matplotlib.

Para más precisión se obtienen las medidas de tendencia central. Y se puede concluir lo siguiente sobre el diagrama y la estadística descriptiva:

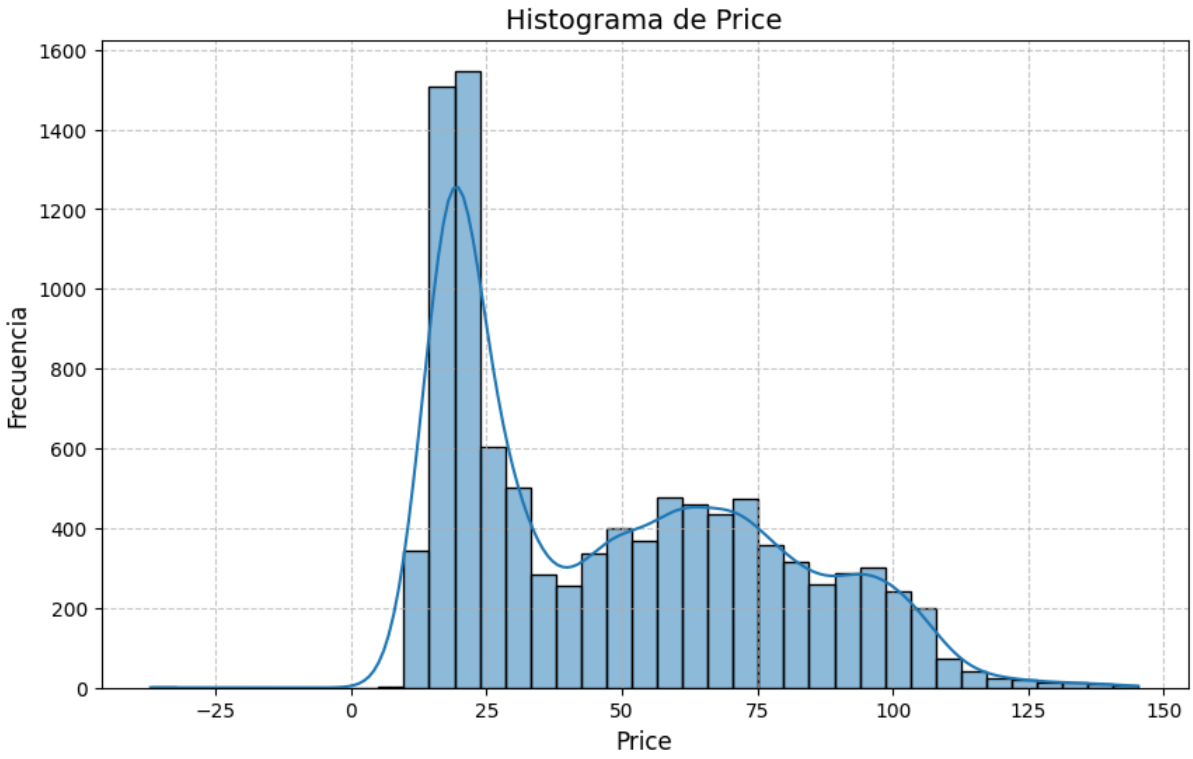
- El 50% de los datos están entre $20 y $72.
- Media: $43.41
- Mediana: $43.18

- Moda: $18.63

Pasando con el histograma de precios, se puede confirmar que la mayoría de precios a lo largo del tiempo se ubican por debajo de los $75. Sus precios máximos y mínimos que son muy raros se analizaran dentro de poco.

Ilustración 9

Histograma de precios del WTI



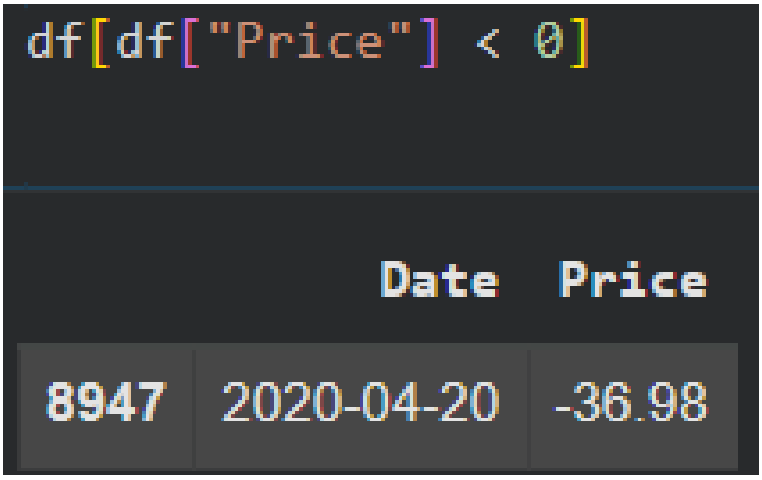
Nota. Autoría propia a partir de datos de FRED. Generado con matplotlib

Existe un único valor negativo, que a simple vista puede parecer un error, pero es un valor totalmente valido. Este valor es del 20 de abril del 2020, periodo de tiempo donde la OMS ya había declarado al COVID-19 como pandemia. Muchas personas encerradas en sus hogares y

activades económicas paralizadas, el petróleo no fue excepción. Muchas empresas petroleras a nivel mundial tenían miles de barriles estancados sin poder vender por lo que el 20 abril el precio se desplomo por un exceso de oferta de petróleo, las empresas preferían vender a perdida antes de seguir almacenando más crudo que podría paralizar por completo la producción.

Ilustración 10

Valor negativo en el dataset



Nota. Autoría propia.

El *dataframe* contiene nulos que corresponden a feriados, esto debido a que el mercado del crudo, así como otros mercados internacionales que cotizan en la bolsa de valores actúan bajo un horario especifico, el del WTI abre en la bolsa de valores de lunes a jueves todo el día con una hora de "descanso", los viernes hasta las 4pm hora de New York y se mantiene cerrado hasta el lunes. Y cuando existe algún feriado no abre la bolsa de valores por lo tanto si un viernes es feriado, la bolsa simplemente no tiene precio, pero si fecha. Para muestra la siguiente imagen.

Ilustración 11

Valores nulos del dataset

```
nulls = df[df["Price"].isna()]
nulls.head()
```

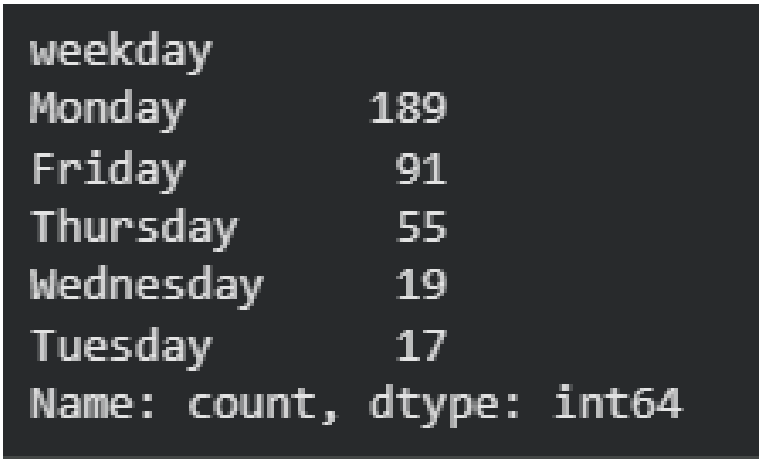
	Date	Price
32	1986-02-17	NaN
61	1986-03-28	NaN
102	1986-05-26	NaN
131	1986-07-04	NaN
172	1986-09-01	NaN

Nota. Autoría propia.

A continuación, el valor de nulos por día de la semana para ejemplificar mejor, los cuales justamente son lunes y viernes.

Ilustración 12

Nulos por semana del dataset



Nota. Autoría propia.

Los hallazgos para esta fase son los siguientes:

- La mayoría de los valores se encuentran por debajo de los $100.
- Existe un precio negativo ya explicado.
- Los valores nulos corresponden a feriados.
- De manera general el *dataset* es de buena calidad.

3.3 Preparación de los datos

Para iniciar esta fase se genera una copia en *df_clean* donde se trabajara el manejo de nulos como buena práctica, el proceso es con *df.copy()*

Para proceder con el manejo de nulos se aplica *Forward Fill* y luego *Backward Fill*. Para cada nulo se toma el último valor válido antes del nulo para rellenar los siguientes y luego aquellos que aún son nulos se rellena con el valor que le sigue al nulo. Y se comprueba que el proceso haya sido exitoso.

Ilustración 13

Forward Fill y Backward Fill

```
df_clean["Price"] = df_clean["Price"].ffill().bfill()
df_clean["Price"].isna().sum()
```

```
np.int64(0)
```

Nota. Autoría propia.

El siguiente paso es limitar el *dataset* a un periodo de tiempo actual postpandemia donde los precios toman un comportamiento que se adapta a una reactivación económica global. A partir de *df_clean* se genera una copia del *dataset* pero a partir del 01/01/2021 y esta copia se guarda en *dt_last5y.*

Para obtener mejores resultados se generan nuevas columnas que servirán como características para capturar patrones que los modelos por si solos no interpretan, estos *features* son:

Tabla 2

Features de XGBoost

Feature	Explicación	Funcionalidad
returns	Calcula el logaritmo natural del periodo *t* sobre *t-1* y con *shift* desplaza el índice en 1.	Variación en porcentaje con el precio anterior.
vol7	Calcula la deviación estándar con una ventana de 7 periodos de tiempo.	Determinar la volatilidad en periodos cortos.
SMA_5	Calcula el promedio en una ventana de 5 periodos de tiempo.	Tendencia a corto plazo

SMA_20	Calcula el promedio en una ventana de 20 periodos de tiempo.	Tendencia a mediano plazo
EMA_12	Con ewn se atribuye un valor exponencialmente ponderado en una ventana de 12 periodos de tiempo y con *span* realiza el cálculo de suavizado que determina la diferencia entre el nuevo valor y el nuevo. Y con mean realiza el mismo proceso del periodo con los nuevos valores.	Tendencia a corto plazo dando peso a las ultima fechas.
mom10	Resta el precio t a el valor de (t-10), *shift* el valor pasado.	Indica la fuerza con la que un precio sube o baja.
lag_1	Toma el valor de 1 periodo de tiempo atrás.	Uso como *feature* auto regresiva
lag_7	Toma el valor de 7 periodos de tiempo atrás.	Uso como *feature* auto regresiva

Nota: Autoría propia

Aquí el código de implementación:

Ilustración 14

Creación de features

```
dt_last5y["returns"] = np.log(dt_last5y["Price"] / dt_last5y["Price"].shift(1))
dt_last5y["vol7"]    = dt_last5y["returns"].rolling(7).std()
dt_last5y["SMA_5"]   = dt_last5y["Price"].rolling(5).mean()
dt_last5y["SMA_20"]  = dt_last5y["Price"].rolling(20).mean()
dt_last5y["EMA_12"]  = dt_last5y["Price"].ewm(span=12, adjust=False).mean()
dt_last5y["mom10"]   = dt_last5y["Price"] - dt_last5y["Price"].shift(10)
dt_last5y["lag_1"]   = dt_last5y["Price"].shift(1)
dt_last5y["lag_7"]   = dt_last5y["Price"].shift(7)
dt_last5y = dt_last5y.dropna()
print(dt_last5y.shape)
dt_last5y.head()
```

```
(1383, 10)
```

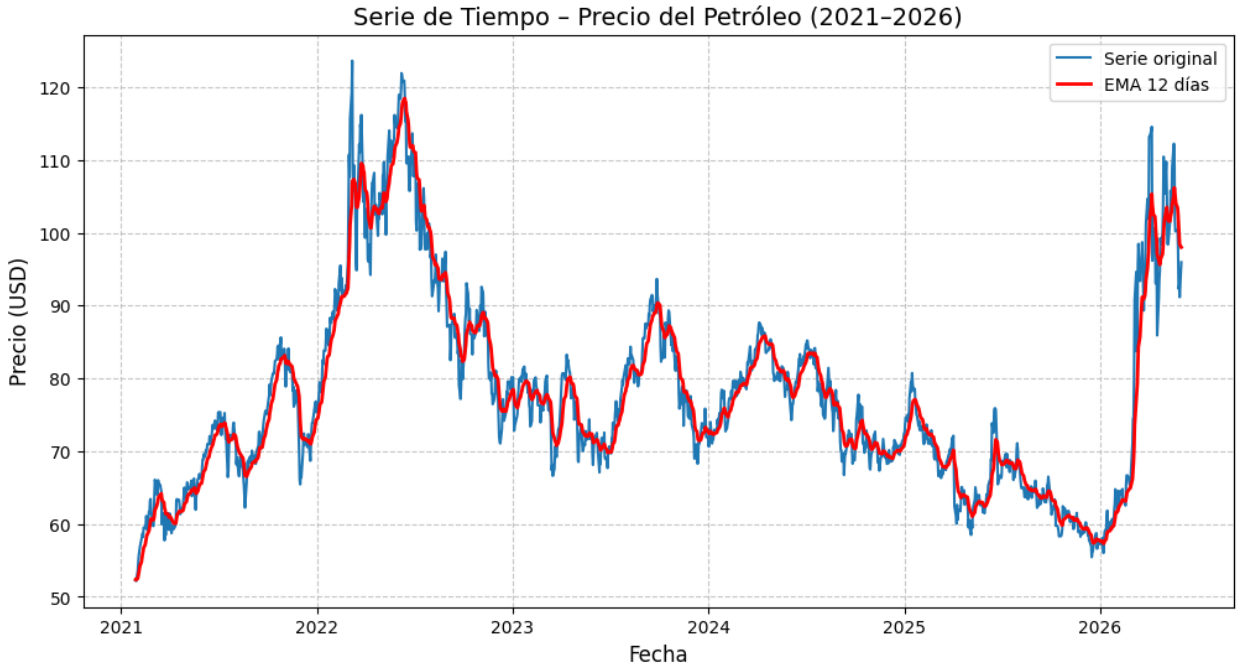
	Date	Price	returns	vol7	SMA_5	SMA_20	EMA_12	mom10	lag_1	lag_7
9150	2021-01-28	52.26	-0.010469	0.008484	52.548	51.8300	52.348576	-1.21	52.81	52.87
9151	2021-01-29	52.16	-0.001915	0.007888	52.524	52.0205	52.319564	-0.09	52.26	53.16
9152	2021-02-01	53.55	0.026300	0.013492	52.678	52.3245	52.508862	1.30	52.16	53.00
9153	2021-02-02	54.77	0.022527	0.013654	53.110	52.5740	52.856729	1.90	53.55	52.28
9154	2021-02-03	55.67	0.016299	0.014124	53.682	52.8350	53.289540	2.51	54.77	52.78

Nota. Autoría propia.

Se procede con la visualización de la serie de tiempo con EMA desde el 2021 hasta la última fecha procesada del 2026 para una muestra visual de los datos con los que se trabajara durante los siguientes pasos.

Ilustración 15

Serie de tiempo del precio del WTI, Original y con EMA12



Nota. Autoría propia a partir de datos de FRED. Generado con matplotlib

3.4 Modelado

Pasando con la cuarta fase se debe comprender que el tipo de modelado que se va a usar es predictivo, ya que el objetivo es determinar el comportamiento futuro del WTI. Se selecciono cuatro algoritmos para el modelado que se presentan en la siguiente tabla junto con su razón de uso para el modelado.

Tabla 3

Algoritmos seleccionados para modelado

Algoritmo	Razón

ARIMA (Autoregressive Integrated Moving Average)	Algoritmo predilecto para determinar tendencias en especial aquellas que se generan de partir de eventos globales, aunque pierde un poco de utilidad al tener una serie temporal con alta volatilidad.
XGBoost(Extreme Gradient Boosting)	Algoritmo ideal para capturar series de tiempo como la actual que tienen ruido y ciclos.
LSTM (Long Short-Term Memory)	A pesar de reducir el *dataframe* a los últimos 5 años es suficiente para capturar los ciclos de la serie.
Híbrido (ARIMA - Residuos de XGBoost)	Algoritmo hibrido que toma la tendencia con ARIMA y con los residuales de XGBoost le otorga el factor ruido de la propia serie.

Nota. Autoría propia.

Una vez seleccionado los modelos se requiere de dividir el *dataframe* en dos partes. La primera es de entrenamiento que corresponde al 80% de los datos que va desde el primer día del 2021 hasta el mediados de año del 2025 aproximadamente (este valor dependerá de la fecha de implementación del modelo), la segunda es de prueba y corresponde al 20% que van desde que termina los datos de entrenamiento hasta la última fecha registrada en el *dataframe*. El siguiente código es una muestra de la división del *dataframe*, para ejemplificar el rango de fechas y el manejo del *split* para la división del *dataframe*.

Ilustración 16

División del dataframe en entramiento y testeo como ejemplo

```
df_clean_indexed5y = dt_last5y.set_index('Date')

# Split 80/20
split = int(len(df_clean_indexed5y) * 0.8)
train = df_clean_indexed5y.iloc[:split]
test  = df_clean_indexed5y.iloc[split:]

print(f"Train: {train.index.min().date()} → {train.index.max().date()} ({len(train)} días)")
print(f"Test:  {test.index.min().date()} → {test.index.max().date()} ({len(test)} días)")
```

```
Train: 2021-01-28 → 2025-05-06 (1114 días)
Test:  2025-05-07 → 2026-06-01 (279 días)
```

Nota. Autoría propia.

Antes de dividir el *dataframe*, primero se requiere de definir las variables predictores y objetivo. En este caso nuestras ocho columnas predictoras junto con *Price*, se asignan a la lista *feature_cols*.

Ilustración 17

Lista con todas las features

```
feature_cols = [
    "Price","returns","vol7",
    "SMA_5","SMA_20","EMA_12",
    "mom10","lag_1","lag_7",
]
```

Nota. Autoría propia.

Lo siguiente es darle a “x” todos los valores como variable predictora y a “y” la variable objetivo de *Price*. En el proceso para “y” se requiere recorrer 7 filas que corresponde al *lag* de

una de los *features* y eliminar aquellos registros que se quedan como nulos. Y para “x” no se toma en cuenta los últimos 7 días

Ilustración 18

Asignar la variable predictora y objetivo

```
X = dt_last5y[feature_cols]
y = dt_last5y["Price"].shift(-7).dropna()
X = X.iloc[:-7]
```

Nota. Autoría propia.

Ahora si se procede con la división correspondiente como se mencionó en el paso anterior pero ahora con las variable predictora y objetivo establecida. Tanto de *x* como *y*, con su respectivo *train* y *test* para cada uno.

Ilustración 19

División del dataframe en entrenamiento y testeo para X y Y

```
split = int(len(X) * 0.8)

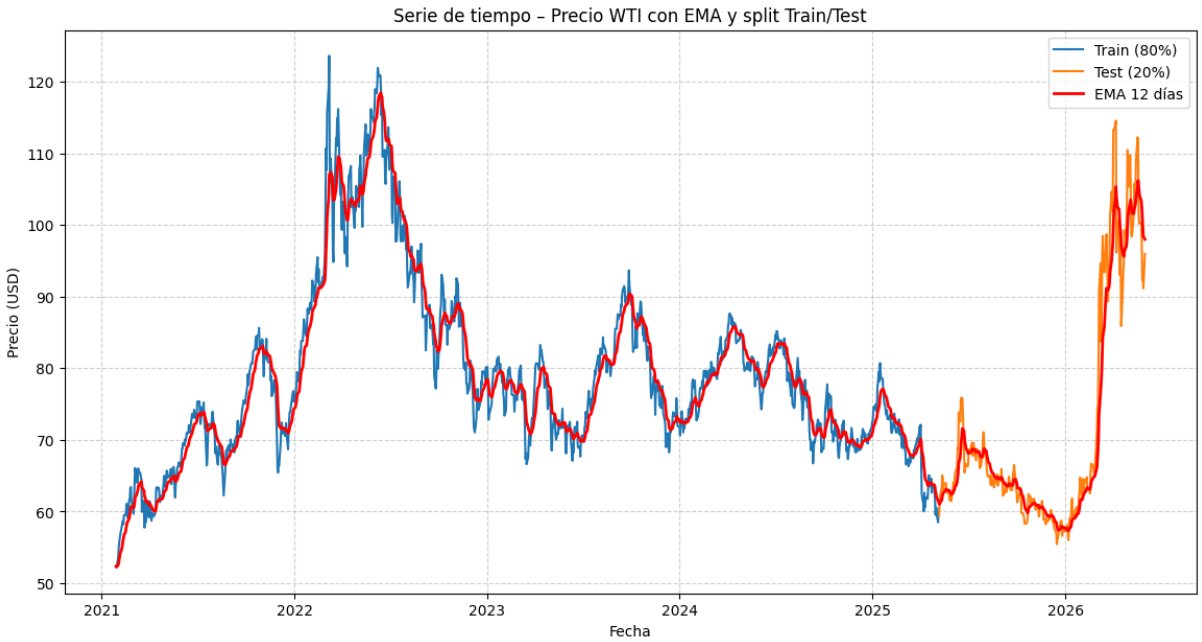
X_train = X.iloc[:split]
X_test = X.iloc[split:]

y_train = y.iloc[:split]
y_test = y.iloc[split:]
```

Nota. Autoría propia.

Ilustración 20

Línea de tiempo con la división de entrenamiento y testeo



Nota. Autoría propia a partir de datos de FRED. Generado con matplotlib

3.4.1 Modelo 1: ARIMA

Para ARIMA primero se crea el modelo y se especifica la columna a entrenar que en este caso es *Price* con los 3 hiperparámetros de ARIMA.

- Para el “p” que defina cuantos precios pasados usa para la predicción. Se usa un valor de cinco es decir una semana normal en la que abre el mercado.
- Para “d”, la diferencia que determina la estacionalidad transformando en este caso la serie se transforma una sola vez usando para cada fecha el precio anterior para determinar la diferencia del precio entre fechas.
- Para “q” es el Moving Average, determinar el número de errores que comete antes de volverse a entrenar.

Ilustración 21

Creación e hiperparámetros de ARIMA

```
arima_model  = ARIMA(
    train["Price"],
    order=(5, 1, 3))
```

Nota. Autoría propia.

Una vez generado el modelo con "fit" se entrena el modelo. Lo siguiente es generar predicciones a partir del modelo entrenado con *forecast*, la longitud o número de predicciones a realizar esta de limitado por la longitud del fragmento del *test*. Y para finalizar se necesita ajustar el índice para atribuir conectar el índice del modelo con las predicciones.

Ilustración 22

Entrenamiento, generar predicciones e indexación del modelo ARIMA

```
arima_fitted = arima_model.fit()


arima_pred = arima_fitted.forecast(
    steps=len(test)
)


arima_pred.index = test.index
```

Nota. Autoría propia.

3.4.2 Modelo 2: XGBoost

Para *XGBoost* se realiza el mismo proceso de crear el modelo, pero con los hiperparámetros propios usando scikitlearn, se va a tomar los más relevantes este, que son:

- n_estimators, el número de árboles que usa, en este caso 200 es un valor razonable para generar lo suficiente para tener alta precisión, pero no sufrir de sobreajuste.
- learning_rate, controla la tasa de aprendizaje y cuanto aporta cada árbol.
- max_depth, profundidad máxima del árbol
- subsample, fragmento de la data que se usa por árbol
- colsample_bytree, fragmento de características que se usa por árbol
- gamma, nivel mínimo que un nodo se divide, con valores bajos el árbol es permisivo y con valores alto es estricto.
- reg_alpha, regularización de los coeficientes similar a Lasso, reduciendo la complejidad innecesaria o penalizando aquellas ramas que no aportan.
- reg_lambda, regularización de los coeficientes similar a Ridge, reduciendo los pesos muy grandes evitando el sobreajuste.

Ilustración 23

Creación e hiperparámetros de XGBoost

```
xgb = XGBRegressor(
    n_estimators=200,
    learning_rate=0.04,
    max_depth=3,
    subsample=0.8,
    colsample_bytree=0.8,
    gamma=0.1,
    reg_alpha=0.5,
    reg_lambda=0.3
)
```

Nota. Autoría propia.

Una vez realizado el modelado de *XGBoost* el siguiente paso es entrenar el mismo con la data de entrenamiento esta con *fit* sobre la data. Para posterior generar la predicción del precio en de las fechas establecida en *X_test.*

Ilustración 24

Entrenamiento y generar predicciones de XGBoost

```
xgb.fit(X_train, y_train)

xgb_pred = xgb.predict(X_test)
```

Nota. Autoría propia.

3.4.3 Modelo 3: LSTM

LSTM es uno de los algoritmos más complejo ya que es una red neuronal que trabaja bajo secuencias por lo que procesa los datos de la misma forma. Por lo que el proceso es un poco más extenso.

El primer paso es normalizar los datos para que los valores estén escalados y las redes neuronales pueden entender mucho mejor, este escaldado se establece en un rango de 0 1. Primero se crea con *MinMaxScaler* los escaladores, para recordar X son los *features* y Y el precio, luego con *fit_transform*, transforma los datos en su valor escalado. Para *Y* hay un ligero ajuste antes de escalarlas y es ajustar la longitud de los datos entre *X* y *y_price.*

Ilustración 25

Escalado de las features y el precio para LSTM

```
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()

X_scaled = scaler_X.fit_transform(X)

y_price = dt_last5y["Price"].values[-len(X):]

y_scaled = scaler_y.fit_transform(
    y_price.reshape(-1, 1)
).flatten()
```

Nota. Autoría propia.

Creación de secuencias: Como se mencionó LSTM está basado en secuencias por lo que está es la pieza clava para el modelado funcione correctamente. Para esto se define la función “create_sequences” que recibe las variables de entrada, objetivo y la longitud de cada secuencia,

para este caso la secuencia está determinada en días por tanto si el valor “seq_length” es 60 cada secuencia va a tener 60 días. Dentro de la función se crea dos listas vacías una para cada parte de los datos.

En el bucle recorre toda la serie temporal con la secuencia establecida hasta llegar a la longitud de la seria menos 60 y una vez finalizado transforma las listas en arreglos usando *Numpy.*

Ilustración 26

Función de creación de secuencias para LSTM

```
def create_sequences(X, y, seq_length=60):
    Xs, ys = [], []

    for i in range(len(X) - seq_length):
        Xs.append(X[i:i + seq_length])
        ys.append(y[i + seq_length])

    return np.array(Xs), np.array(ys)
```

Nota. Autoría propia.

Una vez creado se realiza el llamado a la función pasando como parámetros los datos escalados y como resultado se espera *Xs* y *Ys* como arreglos.

Ilustración 27

Llamado de la función para generar arreglos de X y Y en LSTM

```
Xs, ys = create_sequences(
    X_scaled,
    y_scaled,
)
```

Nota. Autoría propia.

Como resultado se obtiene “Xs: (1326, 60, 9)”, que es las dimensiones del arreglo los datos, que se interpreta como 1326 secuencias a usar, 60 días por cada secuencia y las 9 *features* generadas.

Asociar fechas a secuencias: Al generar las secuencias los registros asociados dejaron de estar conectados a una fecha como índice por lo que se requiere volver a generar un índice por secuencia consultando con el *dataframe* original. La fecha establecida de la secuencia es aquella que se va a predecir es decir que si la secuencia es de 60 se establece la fecha de secuencia del día 61.

Ilustración 28

Asociación de índice de fechas en LSTM

```
full_datetime_index = pd.to_datetime(
    df_clean_indexed5y.index[
        SEQ_LENGTH : SEQ_LENGTH + len(Xs)
    ].strftime('%Y-%m-%d')
)
```

Nota. Autoría propia.

Para definir correctamente la división entre el entrenamiento y el testeo se guarda la fecha partir del penúltimo registro indexado del fragmento de enteramiento inicial y este se usa para determinar en *full_datetime_index* cuál será el punto donde se va dividir los datos.

Ilustración 29

Registro de fecha para división del dataframe según el índice para LSTM

```
split_date = train.index[-1]

split_idx = np.searchsorted(
    full_datetime_index,
    split_date
)
```

Nota. Autoría propia.

Una vez teniendo el valor del cual se va a dividir los datos para entrenamiento y testeo, se aplica para la división tanto de Xs como ys. Y se hace el mismo proceso para las fechas indexadas

Ilustración 30

Dividir el dataframe en entrenamiento y testeo para LSTM

```
X_train_lstm = Xs[:split_idx]
X_test_lstm  = Xs[split_idx:]

y_train_lstm = ys[:split_idx]
y_test_lstm  = ys[split_idx:]

train_datetime_lstm = full_datetime_index[:split_idx]
test_datetime_lstm  = full_datetime_index[split_idx:]
```

Nota. Autoría propia.

Dando como resultado esta cantidad de datos tanto para entrenamiento como testeo:

X_train_lstm: (1053, 60, 9)

X_test_lstm: (273, 60, 9)

El siguiente paso es la construcción de la red secuencial del modelo, el modelo contara con 3 capas que son:

1. La primera es de 128 neuronas que mantiene la secuencia completa y toma la forma de la secuencia de 60 días y de los *features*.
2. La segunda es de 64 neuronas manteniendo la secuencia completa.
3. La última es de 32 neuronas manteniendo la secuencia completa.

Entre cada capa se realiza un apagado aleatorio del 20% de las neuronas usando "Dropout" con el objetivo de evitar que el modelo aprenda de los datos. Después de construir las 3 capas LSTM se crea 2 capas más adicionales que es "Dense" la primera con un valor de 16 que represente los valores de la última capa LSTM en datos más compactos y conectados para que luego se puede interpretar la salida; la segunda capa can un valor de 1 es la predicción de valores futuros.

Ilustración 31

Arquitectura y creación del modelo LSTM

```
model = Sequential([

    LSTM(
        128,
        return_sequences=True,
        input_shape=(SEQ_LENGTH, X.shape[1]),
    ),

    Dropout(0.20),

    LSTM(
        64,
        return_sequences=True,
    ),

    Dropout(0.20),

    LSTM(
        32,
    ),

    Dropout(0.15),

    Dense(16,),

    Dense(1)

])
```

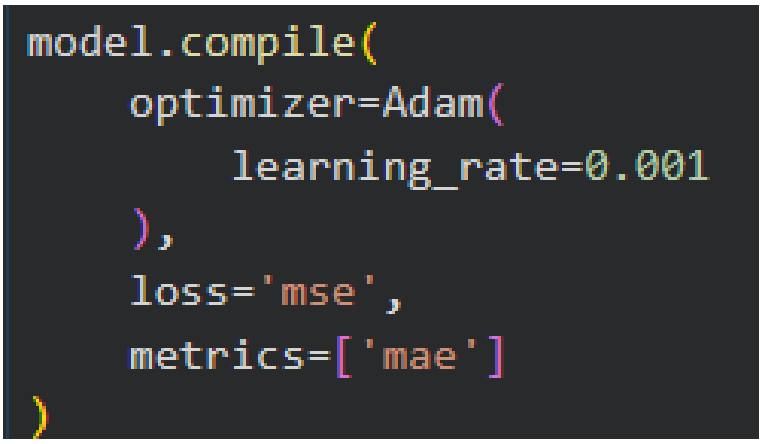
Nota. Autoría propia.

Continuando con el proceso se procede a compilar el modelo, dentro de los hiperparámetros de la compilación se establece el valor de Adam que optimiza y control a los pesos de cada salida. Con *learning_rate* se controla la tasa de aprendizaje del modelo con valores

más bajos es un modelo más estable pero lento. Y el modelo realiza correcciones basado en el MSE.

Ilustración 32

Compilación del modelo LSTM



Nota. Autoría propia.

Una parte importante del proceso con el modelo compilado es le creación de “callbacks” o el control del entrenamiento de manera automatizado. Esta parte consta de 2 “callbacks” que son:

- EarlyStopping, este se encarga de detener el entrenamiento si es que en 10 épocas de la red no mejora el modelo basado en el MSE y retorna a la mejor época del entrenamiento.
- ReduceLROnPlateau, se encarga de reducir la tasa de aprendizaje a la mitad cada 5 épocas de la red cuando el modelo no mejora de igual manera basado en el MSE.

Ilustración 33

Creación de callbacks para LSTM



```
callbacks = [

    EarlyStopping(
        monitor='val_loss',
        patience=10,
        restore_best_weights=True,
        verbose=1
    ),

    ReduceLROnPlateau(
        monitor='val_loss',
        factor=0.5,
        patience=5,
        min_lr=1e-5,
        verbose=1
    )

]
```

Nota. Autoría propia.

El siguiente paso es entrenar el modelo generado con *fit* e indicar los parámetros del entrenamiento que son el conjunto de datos tanto de entrenamiento de X como de Y, usando un 15 por ciento de los datos como validación, también se establece un máximo de 100 épocas para el entrenamiento; divide el *dataframe* se divide en grupos llamados “batch” y cada uno cuento con 16 muestras, es decir. Toma los *callbacks* creados

Ilustración 34

Entrenamiento del modelo LSTM

```
history = model.fit(

    X_train_lstm,
    y_train_lstm,

    validation_split=0.15,

    epochs=100,
    batch_size=16,

    callbacks=callbacks,

    verbose=1

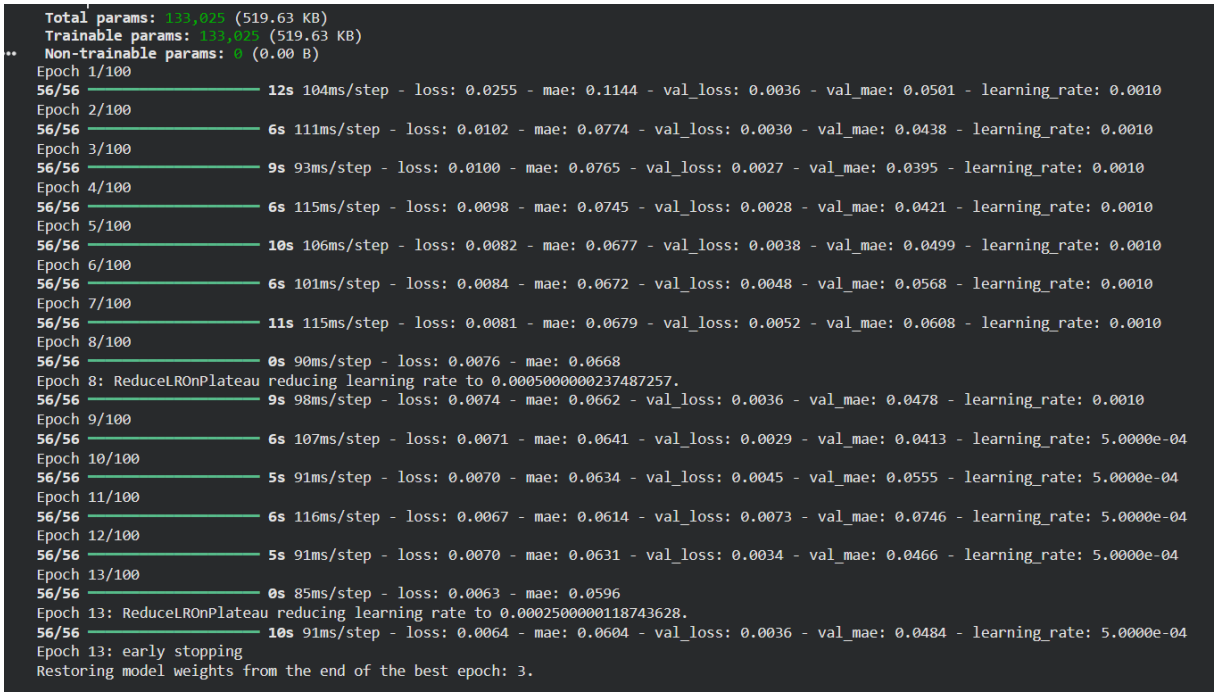
)
```

Nota. Autoría propia.

Una vez concluido el entramiento se obtiene una retroalimentación visual que se interpreta de la siguiente manera. La red neuronal realiza un conjunto de iteraciones 13 veces cada una con 56 *batches*. En la época 8 y 13 se aplica el *callback ReduceLROnPlateau* pero no mejora porque finalmente en la época 13 se aplica el otro *callback EarlyStopping.*

Ilustración 35

Proceso de enteramiento del modelo LSTM

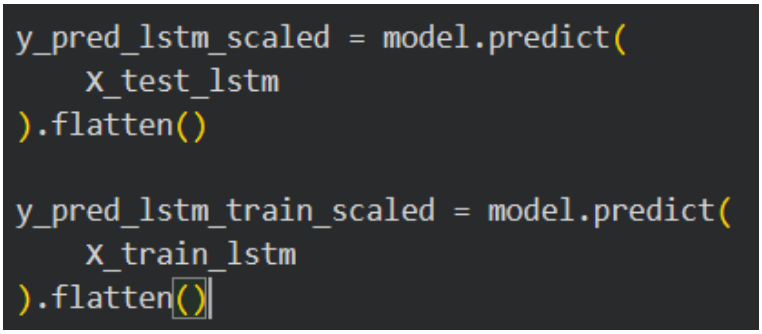


Nota. Autoría propia.

Una vez concluido con el entrenamiento se procede a hacer la predicción tanto para X como Y en cada variable se guardan los valores aun escalos.

Ilustración 36

Generar predicciones escaladas para LSTM



```
y_pred_lstm_scaled = model.predict(
    X_test_lstm
).flatten()

y_pred_lstm_train_scaled = model.predict(
    X_train_lstm
).flatten()
```

Nota. Autoría propia.

Así como se normalizo los valores para pasar los datos a un rango de 0 a 1, ahora se requiere el proceso inverso para mostrar las predicciones.

Ilustración 37

Desnormalización de los valores de la predicción LSTM

```
lstm_pred = scaler_y.inverse_transform(
    y_pred_lstm_scaled.reshape(-1, 1)
).flatten()

y_test_lstm_real = scaler_y.inverse_transform(
    y_test_lstm.reshape(-1, 1)
).flatten()

lstm_train_pred = scaler_y.inverse_transform(
    y_pred_lstm_train_scaled.reshape(-1, 1)
).flatten()
```

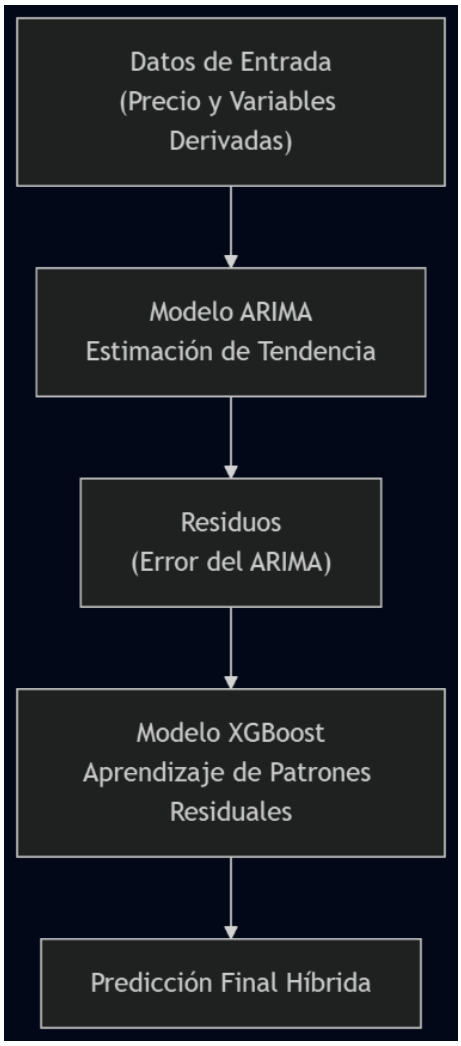
Nota. Autoría propia.

3.4.4 Modelo 4: Híbrido, ARIMA +XGBoost

La idea del modelo híbrido es tomar la tendencia general de ARIMA y a partir de este realizar correcciones para agregar el elemento volátil propio del comportamiento natural del activo.

Ilustración 38

Arquitectura del modelo híbrido



Nota. Autoría propia. Realizado en Mermaid.

El primero paso es establecer el peso de cada modelo teniendo en cuenta que la predicción base es ARIMA. Al estar en una época tensa geopolíticamente hablando lo mejor es darle más peso a XGBoost.

Ilustración 39

Ponderación del modelo híbrido

```
ARIMA_WEIGHT = 0.1
XGBOOST_WEIGHT = 0.9
```

Nota. Autoría propia.

Como ya existe el modelo ARIMA y su debido proceso su reutiliza parte de lo ya usado para esta modelo, generando directamente predicciones basado en el conjunto de datos de entrenamiento.

Ilustración 40

Generar predicciones del modelo de ARIMA para el modelo híbrido

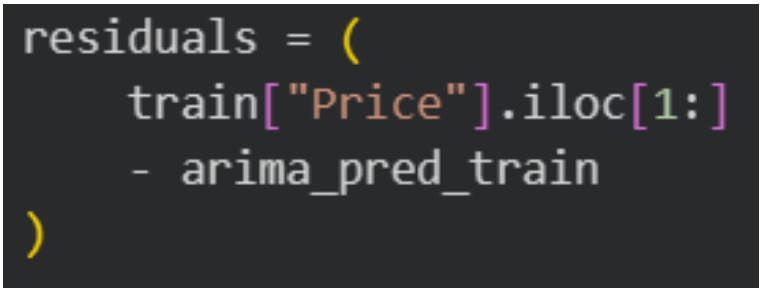
```
arima_pred_train = arima_fitted.predict(
    start=train.index[1],
    end=train.index[-1]
)
```

Nota. Autoría propia.

Con la predicción realizado el siguiente paso es calcular los residuales de ARIMA, estos residuales se refiere a la diferencia entre el valor real y la predicción.

Ilustración 41

Calculo de residuales a partir de ARIMA



```
residuals = (
    train["Price"].iloc[1:]
    - arima_pred_train
)
```

Nota. Autoría propia.

Al ser nuevos valores se requiere de volver a construir los *features* sobre la predicción de ARIMA, son exactamente las mismas 9 columnas, pero con los valores ajustados a la predicción ya realizada.

Ilustración 42

Recrear las features de XGBoost para el híbrido

```
df_for_res = df_clean_indexed5y.copy()

df_for_res["returns"] = np.log(
    df_for_res["Price"]
    / df_for_res["Price"].shift(1)
)

df_for_res["vol7"] = (
    df_for_res["returns"]
    .rolling(7)
    .std()
)

df_for_res["SMA_5"] = (
    df_for_res["Price"]
    .rolling(5)
    .mean()
)

df_for_res["SMA_20"] = (
    df_for_res["Price"]
    .rolling(20)
    .mean()
)

df_for_res["EMA_12"] = (
    df_for_res["Price"]
    .ewm(span=12, adjust=False)
    .mean()
)

df_for_res["mom10"] = (
    df_for_res["Price"]
    - df_for_res["Price"].shift(10)
)

df_for_res["lag_1"] = (
    df_for_res["Price"]
    .shift(1)
)

df_for_res["lag_7"] = (
    df_for_res["Price"]
    .shift(7)
)

hybrid_feature_cols = [
    "Price",
    "returns",
    "vol7",
    "SMA_5",
    "SMA_20",
    "EMA_12",
    "mom10",
    "lag_1",
    "lag_7"
]
```

Nota. Autoría propia.

El siguiente paso es alinear y asegurar que los datos de la predicción junto con los residuales coincidan tanto en orden como en formato de año/mes/día.

Ilustración 43

Alienación de índice con las predicciones de los residuales

```
residuals.index = pd.to_datetime(
    residuals.index.strftime("%Y-%m-%d")
)

arima_pred.index = pd.to_datetime(
    arima_pred.index.strftime("%Y-%m-%d")
)

df_for_res.index = pd.to_datetime(
    df_for_res.index.strftime("%Y-%m-%d")
)
```

Nota. Autoría propia.

Ya con los datos alineados se requiere generar un *dataframe* "X_train_res" que serán los datos de entrenamiento donde va a estar tanto las predicciones de ARIMA como las columnas adicionales de los *features*. Estas ya van a contar con las fechas alineadas entre sí.

Ilustración 44

Creación del dataframe con los de entrenamiento

```
X_train_res = (
    df_for_res.loc[
        residuals.index.intersection(
            df_for_res.index
        ),
        hybrid_feature_cols
    ]
    .dropna()
)

residuals_aligned = residuals.loc[
    X_train_res.index
]
```

Nota. Autoría propia.

Similar al proceso anterior también se requiere generar un *dataframe* “X_test_res” que serán los datos de prueba. De igual manera con fechas alineadas entre sí según el periodo correspondiente.

Ilustración 45

Creación del dataframe con los de prueba

```
X_test_res = (
    df_for_res.loc[
        arima_pred.index.intersection(
            df_for_res.index
        ),
        hybrid_feature_cols
    ]
    .dropna()
)

arima_pred_aligned = arima_pred.loc[
    X_test_res.index
]
```

Nota. Autoría propia.

El siguiente paso es realizar la predicción con *XGBoost* sobre los residuales. Siguiendo el mismo proceso aplicado sobre *XGBoost* con ligeros ajuste en los hiper parámetros, teniendo en cuenta es que ARIMA realiza predicciones sin tener en cuenta los residuales para la predicción lo ideal es aplicar sobre los residuales más árboles y un ratio de aprendizaje mayor para una capacidad de aprendizaje mayor, así valores más altos en ambas regularizaciones ayudan a evitar el sobreajuste.

Con el modelo establecido se realiza el entrenamiento respectivo.

Ilustración 46

Creación del modelo XGBoost para el híbrido

```
xgb_res = XGBRegressor(
    n_estimators=350,
    learning_rate=0.03,
    max_depth=3,
    subsample=0.8,
    colsample_bytree=1,
    gamma=0.1,
    reg_alpha=0.8,
    reg_lambda=0.5
)
xgb_res.fit(
    X_train_res,
    residuals_aligned
)
```

Nota. Autoría propia.

Una vez hecho el entrenamiento el objetivo es generar una predicción de residuos y con su respectivo índice de fechas.

Ilustración 47

Generar predicciones XGBoost sobre los residuales

```
res_correction = pd.Series(
    xgb_res.predict(X_test_res),
    index=X_test_res.index
)
```

Nota. Autoría propia.

El último paso es generar la predicción basada en la ponderación marcada en un proceso ya ejecutado. Y se aplica un formula con la ponderación donde la predicción de ARIMA se multiplica por 0.1 y se suma con la predicción de ARIMA con la corrección de residuales de *XGBoost* multiplicado por 0.9.

Ilustración 48

Generar predicciones del híbrido basado en ponderaciones

```
hybrid_pred = (
    ARIMA_WEIGHT * arima_pred_aligned
    + XGBOOST_WEIGHT * (
        arima_pred_aligned
        + res_correction
    )
)
```

Nota. Autoría propia.

3.5 Evaluación

Una vez generado todos los modelos, el siguiente paso es evaluar estos según 3 métricas que son: el error absoluto medio (MAE), error cuadrático medio (MSE) y el coeficiente determinación (R^2). El rango aceptable es con un R^2 superior a 0.70 pero sin llegar al máximo de 1.0 que representaría un claro sobreajuste.

Para calcular estas métricas se va a usar las funciones que se dispone de la biblioteca de *scikit-learn* que corresponde a las 3 métricas mencionadas, estas se acoplan a una función creada que va a calcular los valores de las métricas para cada modelo.

Ilustración 49

Función para calcular las métricas de cada modelo

```
def metrics(y_true, y_pred, name):
    mae  = mean_absolute_error(y_true, y_pred)
    rmse = np.sqrt(mean_squared_error(y_true, y_pred))
    r2   = r2_score(y_true, y_pred)
```

Nota. Autoría propia.

Tabla 4

Métricas de evaluación de cada modelo

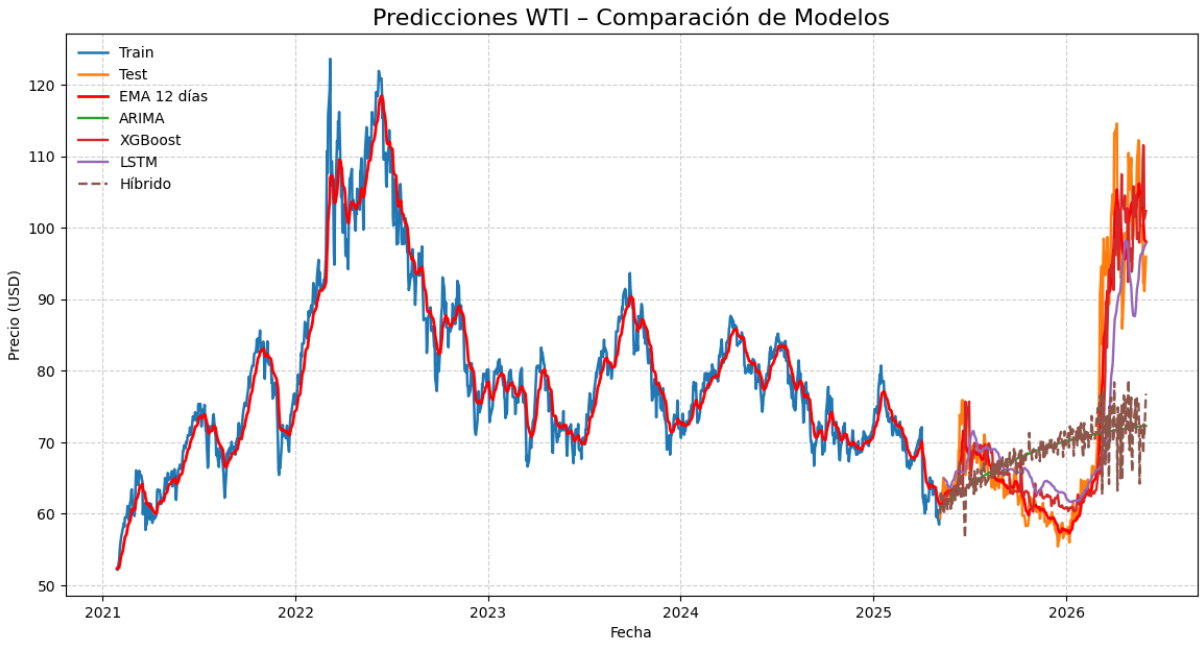
Modelo	MAE	RMSE	R^2
ARIMA	10.9345	14.9141	0.1137
XGBoost	4.1986	6.3139	0.8414
LSTM	5.7404	8.2718	0.7313
Híbrido ARIMA + XGBoost	10.8453	14.7513	0.1329

Nota. Autoría propia.

Las métricas revelan una clara superioridad del modelo XGBoost sobre los demás en espacial sobre ARIMA y el modelo híbrido. Para plasmar de mejor manera se procede con la visualización de las predicciones.

Ilustración 50

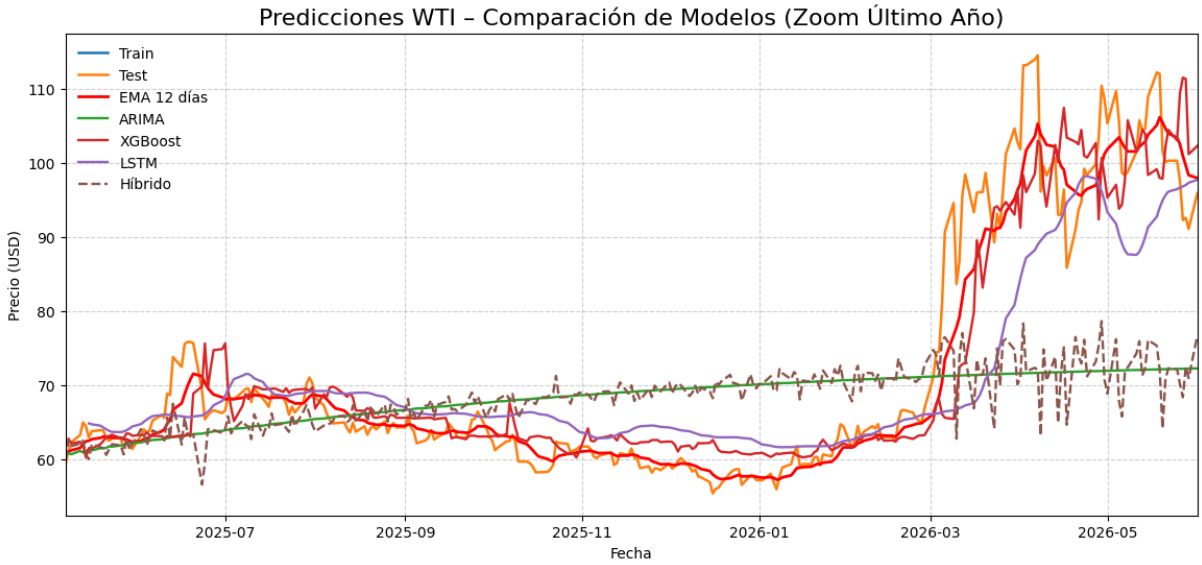
Línea de tiempo con la comparativa de cada modelo



Nota. Autoría propia. Generado con matplotlib

Ilustración 51

Línea de tiempo con la comparativa de cada modelo con zoom del último año



Nota. Autoría propia. Generado con matplotlib

Tanto con las métricas como la visualización de las predicciones sobre el conjunto de fechas de testeo se puede generar ciertas conclusiones de manera preliminar para cada modelo.

a) XGBoost, demostró ser el mejor modelo en con un R^2 de 0.84 que es un rango bastante aceptable y sin llegar a un valor tan altos que pueden significar un sobreajuste.

b) LSTM, logró predecir de manera efectiva el cambio de tendencia, pero no el ruido incluido dentro de ese cambio.

c) ARIMA, determina de manera muy lejana el cambio de tendencia y por lo mismo se tomó en cuenta más que nada como base para la implementación del hibrido.

d) Hibrido, durante el primero periodo de tiempo donde no existe tanto ruido ni cambios en tendencia agresivos, parece funcionar correctamente, pero en cuento

el ruido o el cambio de tendencia es drástico el modelo deja de funcionar como se esperaba.

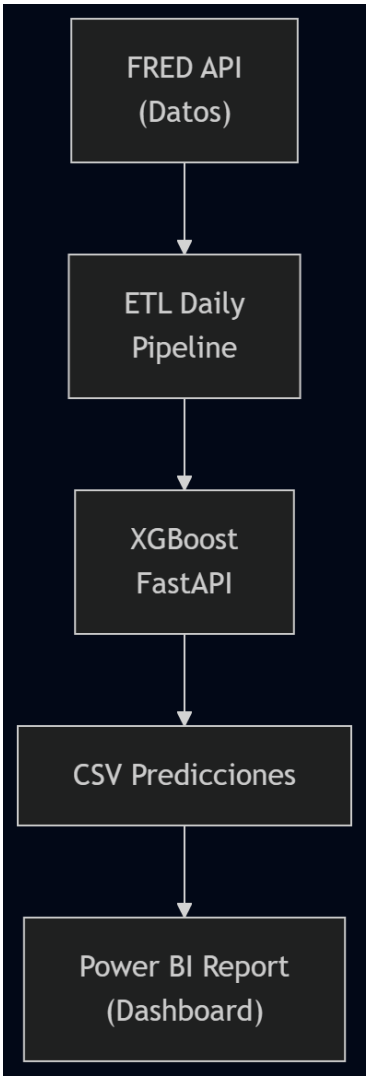
3.6 Despliegue

En la fase 6 de CRISP-DM tiene como objetivo plantear la implementación en un entorno real. Como se menciono en el alcance este proyecto solo contempla una fase preliminar y no el despliegue real. Para el despliegue se va a considerar el modelo seleccionado, *XGBoost* con su debida implementación y visualización en un *dashboard* de *PowerBI.*

La arquitectura planteada consiste en la descarga desde la API de FRED de los datos actuales, luego la data pasa por un pipeline de ETL, una vez con los datos en orden mediante FastAPI se carga la data y el modelo para de manera seguida generar las predicciones en ".csv" las mismas que se cargan dentro de la herramienta de *PowerBI* para la visualización de la predicción junto con otros datos de soporte para la toma de decisiones referente a las exportaciones.

Ilustración 52

Plan preliminar de despliegue



Nota. Autoría propia.

En un entorno de pruebas se generó el respectivo código en Python que se muestra a continuación para cumplir con el objetivo de generar predicciones. La idea del flujo como usuario es acceder a un *endpoint* que permita realizar los 4 primeros pasos propuestos en la arquitectura sin hacer todo manualmente, este planteamiento es escalable para que el llamado a

la API ni si quiera necesite interacción del usuario ya que se puede implementar llamadas automatizadas a la API en días programados por ejemplo el 1 de cada mes. Y una vez generado él ".csv" el usuario únicamente tendrá que acceder a *PowerBI* cerrando el flujo del proceso de usuario.

Ilustración 53

Endpoint para realizar la predicción en el plan preliminar

```python
@app.get("/prediccion_xgboost")
def generar_prediccion_xgboost():
    """Endpoint para predicción con XGBoost"""
    if xgb_model is None:
        return {
            "status": "error",
            "mensaje": "Modelo XGBoost no disponible"
        }

    df = cargar_y_preprocesar_data()
    df_pred = predict_xgboost_future(df, dias_futuro=180)

    if df_pred is None:
        return {
            "status": "error",
            "mensaje": "Error en predicción XGBoost"
        }

    df_real = df.copy().reset_index(drop=True)
    df_real["Type"] = "REAL"

    df_final = pd.concat([df_real, df_pred], ignore_index=True)
    output_path = "predicsvoil4/prediccion_xgboost.csv"
    df_final.to_csv(output_path, index=False, float_format="%.4f")

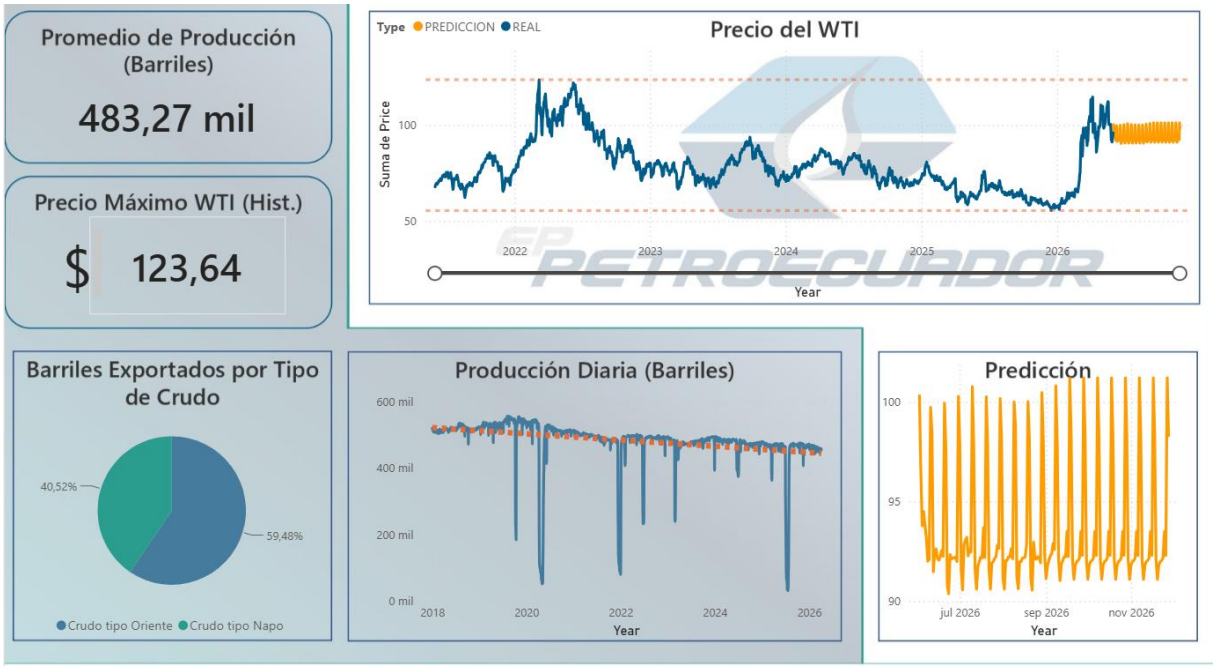
    return {
        "status": "OK",
        "mensaje": "Predicción XGBoost generada correctamente",
        "archivo": output_path,
        "filas": len(df_final)
    }
```

Nota. Autoría propia.

En *PowerBI* se visualizará información relevante para el objetivo del negocio. Esto incluye el precio del WTI, máximo, promedio de producción del crudo de petróleo y porcentaje de barriles exportados según el tipo de crudo. Estos datos son suficientes y claros para que altos directivos de EP-Petroecuador realicen una respectiva toma de decisiones. El *dashboard* también usa datos abiertos del Banco Central del Ecuador para completar los datos de la predicción, como la producción diaria por barriles y la cantidad de crudo exportado por cada tipo desde el 2023 pero que sirve de referencia de manera general.

Ilustración 54

Dashboard sobre la situación petrolera del Ecuador



Nota. Elaboración propia basado en datos abiertos del Banco Central. Desarrollado en PowerBI.

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES

4. Conclusiones y Recomendaciones

4.1 Conclusiones

Como se observó durante el proyecto el realizar un proceso de minería de datos en especial bajo CRISP-DM permite alinear la predicción del WTI al giro del negocio en este caso de EP-Petroecuador.

Para relacionar directamente la predicción generada, hay que entender justamente como afecta al giro del negocio. Un factor a considerar es que a pesar de que el precio del WTI y por tanto el PVO varie, el costo de producción sigue siendo el mismo. Por tanto, se debe tener en cuenta son los siguientes dos puntos:

- A mayor POV mayor utilidad recibe EP-Petroecuador.
- A menor POV menor utilidad recibe EP-Petroecuador

La predicción realizada permite ver que en los próximos 6 meses el precio puede cotizar dentro de un rango de 90 a 100 dólares por barril aproximadamente, este teniendo en cuenta que hasta el momento de la redacción de este proyecto aún existe cierta tensión debido a la guerra entre Irán y Estados Unidos que afecta directamente al precio del WTI.

Justamente es relevante mencionar que en caso de que la situación geopolítica varie ya sea por el conflicto actual u otros factores es necesario hacer una revisión del modelado porque los comportamientos pueden variar y así como en su momento el COVID-19 fue determinante para establecer una tendencia al alza post pandemia lo mismo puede pasar bajo otros contextos.

Es interesante mencionar el estado actual del petróleo ecuatoriano justamente basado en el *dashboard* presentado y explicaciones adicionales, la producción de petróleo a tomado una tendencia bajista debido a dos factores principales. La primera es debido a una infraestructura

pobre y en desgaste, pero el segundo factor es el cierre del bloque petrolero 44-ITT que se puso en marcha tras la consulta popular del 2023. Esto implica un riesgo fiscal si no se temo acciones.

Las acciones a tomar más optimistas por parte de Petroecuador es buscar la inversión en infraestructura, así como un plan de reducción de costos para tener mayor margen de utilidad en cada venta de barril de petróleo. Por lo mismo se menciona como la idea optimista y no la realista ya que al ser empresa publica requiere fondos del estado, y el estado actualmente no tiene el poder económico para realizar esa inversión de gran magnitud.

A breves rasgos el oro negro de Ecuador dejo atrás su época de gloria, opacado por exportaciones no petroleras en los últimos años. Y de no tomar acciones la situación para este recurso de color negro no será más que para abastecer a la propia necesidad del país.

Pasando al lado de los algoritmos usados en el proyecto, *XGBoost* demostró ser el ganador indiscutible para este tipo de predicciones a mediano plazo, debido a la naturaleza de los datos y el enfoque dado. En otros casos LSTM puede ser el modelo más valorado con un *dataset* que tenga más valores como uno que en vez de tener el cierre del precio por día sea por horas o incluso minutos, pero con un enfoque de corto plazo (de 1 a 15 días de predicción), esto debido a que las redes neuronales funcionan de mejor manera a mayor cantidad de datos ya que se generaran a su vez patrones más complejos. Pasando con el hibrido de *ARIMA+XGBoost*, sería el ideal en una época muy estable donde factores geopolíticos no estén generando tanto ruido en el WTI.

4.2 Recomendaciones

Para futuros modelados de *machine learning* se recomienda considerar e investigar la respectiva situación geopolítica que lo engloba. Así como la fuente de los datos de donde se va a iniciar dicho proceso.

Tener cuidado con los hiper parámetros de cada algoritmo ya que, dependiendo de estos, los resultados finales pueden desembocar en favorables o no. En especial con el algoritmo más complejo el cual es LSTM ya que no solo son hiper parámetros sino toda la arquitectura a tener en cuenta. Para una fuente con muchos más datos lo recomendable es usar más neuronas o una capa adicional, pero sin olvidar el *Dropout* para evitar el sobreajuste. Para XGBoost probar modificar alguno de los hiper parámetros configurados por defecto que talvez puede generar alguna aportación, aunque sea mínima.

Si se desea una implementación en un entorno real, se recomiendo previamente considerar recursos económicos, de hardware y software, así como los requisitos para una debida implementación a un sistema funcional, minimizando riesgos de seguridad y de fallos.

BIBLIOGRAFIA

About us. (s. f.). Scikit-learn. https://scikit-learn.org/stable/about.html

Admin, U., & Negocios, R. (2012, 26 junio). Breve reseña sobre la historia petrolera del Ecuador. *El Comercio*. https://www.elcomercio.com/actualidad/negocios/breve-resena-historia-petrolera-del/

Castellanos, D. S. (2024, 19 diciembre). Ranking: los países con más petróleo en el mundo y en Latinoamérica. *Bloomberg Línea*. https://www.bloomberglinea.com/economia/ranking-los-paises-con-mas-petroleo-en-el-mundo-y-en-latinoamerica/

Clark, B. (2026, 25 febrero). ¿Qué es Scikit-Learn (Sklearn)? *IBM*. https://www.ibm.com/mx-es/think/topics/scikit-learn

Crude Oil Prices: West Texas Intermediate (WTI) - Cushing, Oklahoma. (2026, 3 junio). https://fred.stlouisfed.org/series/DCOILWTICO/

de la Cruz Amador, E. G. (2016). Análisis de las propiedades de calidad de un crudo.

Ecuador recibirá ingresos por más de USD 2.723 millones tras 11 concursos spot de crudo – EP PETROECUADOR. (s. f.). https://www.eppetroecuador.ec

El petróleo en Ecuador: Historia, peso económico y desafíos futuros. (2025, 30 abril). Fiduvalor Financial Solutions. https://fiduvalor.com.ec/2025/04/30/el-petroleo-en-ecuador-historia-peso-economico-y-desafios-futuros/

EP Petroecuador adjudica exportación de más de 10 millones de barriles de crudo Oriente y Napo – EP PETROECUADOR. (s. f.). https://www.eppetroecuador.ec/?p=25382

Estadísticas de Petróleo y Minas. (s. f.). https://contenido.bce.fin.ec/documentos/informacioneconomica/SectorReal/ix_PetroleoMinas.html

Estadísticas de reservas, producción y consumo de petróleo de Ecuador - Worldometer. (s. f.). Worldometer. https://www.worldometers.info/es/petroleo/petroleo-ecuador/

Estadísticas Económicas - Banco Central del Ecuador. (2025, 31 marzo). Banco Central del Ecuador. https://www.bce.fin.ec/estadisticas-economicas/

FastAPI - FastAPI. (s. f.). https://fastapi.tiangolo.com/

GeeksforGeeks. (2025a, julio 23). *XGBoost Parameters*. GeeksforGeeks. https://www.geeksforgeeks.org/machine-learning/xgboost-parameters/

GeeksforGeeks. (2025b, julio 23). *XGBoost Parameters*. GeeksforGeeks. https://www.geeksforgeeks.org/machine-learning/xgboost-parameters/

Guerra, S. W. (2025a, julio 28). Sector petrolero de Ecuador en crisis: la mayor caída en más de 20 años. *Primicias*. https://www.primicias.ec/revistagestion/analisis/produccion-petroleo-ecuador-petroecuador-erosion-coca-codo-sote-ocp-101505/

Guerra, S. W. (2025b, diciembre 22). Menos crudo y más exportaciones no petroleras redefinen el superávit comercial de Ecuador. *Primicias*. https://www.primicias.ec/revistagestion/analisis/crudo-exportaciones-petroleras-superavit-comercial-ecuador-112190/

Hora, L. (2025, 10 diciembre). ¿Por qué Ecuador vende su petróleo más barato? La crisis detrás del castigo al crudo. *LA HORA*. https://www.lahora.com.ec/economia/por-que-ecuador-vende-su-petroleo-mas-barato-la-crisis-detras-del-castigo-al-crudo-20251210-0005.html

IBM SPSS Modeler Subscription. (2021, 17 agosto). https://www.ibm.com/docs/es/spss-modeler/saas?topic=dm-crisp-help-overview

json — JSON encoder and decoder. (s. f.). Python Documentation. https://docs.python.org/3/library/json.html

Matplotlib — Visualization with Python. (s. f.). https://matplotlib.org/

mean_absolute_error. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html

mean_squared_error. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html

MinMaxScaler. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

Misión, Visión y Valores – EP PETROECUADOR. (s. f.). https://www.eppetroecuador.ec/?p=3729

Moreira, N. (2025, 16 diciembre). Exportaciones no petroleras crecen 19,7% y baten récord. *El Diario*. https://www.eldiario.ec/negocios/exportaciones-no-petroleras-de-ecuador-crecieron-197-en-2025-y-alcanzaron-record-historico-15122025/

NumPy. (s. f.). https://numpy.org/

Ortiz, L. (2025, 16 mayo). El ocaso petrolero del Ecuador: pocos años de reservas y un nuevo modelo económico en ciernes. *Primicias*. https://www.primicias.ec/revistagestion/analisis/ocaso-petrolero-ecuador-reservas-modelo-economico-ciernes-96181/

pandas - Python Data Analysis Library. (s. f.). https://pandas.pydata.org/

pandas.DataFrame.std — pandas 3.0.3 documentation. (s. f.). https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.std.html

pandas.Series.ewm — pandas 3.0.3 documentation. (s. f.). https://pandas.pydata.org/docs/reference/api/pandas.Series.ewm.html

pandas.Series.rolling — pandas 3.0.3 documentation. (s. f.). https://pandas.pydata.org/docs/reference/api/pandas.Series.rolling.html

pandas.Series.shift — pandas 3.0.3 documentation. (s. f.). https://pandas.pydata.org/docs/reference/api/pandas.Series.shift.html

Parameters | XGBoosting. (s. f.). https://xgboosting.com/parameters/

Pazmiño, M. P. (2025, 20 agosto). Cierre del bloque 43-ITT deja pérdidas millonarias; comunidades denuncian demoras. *Diario Expreso*. https://www.expreso.ec/actualidad/economia/ecuador-indigenas-alertan-incumplimiento-en-el-cierre-de-pozos-del-bloque-43-itt-254124.html

pickle — Python object serialization. (s. f.). Python Documentation. https://docs.python.org/3/library/pickle.html

Power BI: visualización de datos | Microsoft Power Platform. (s. f.). https://www.microsoft.com/es-es/power-platform/products/power-bi

Precio Petróleo Crudo Ecuatoriano. (s. f.). https://contenido.bce.fin.ec/documentos/informacioneconomica/indicadores/real/PrecioPetroleoCrudoEcuatoriano.html

Precio petróleo (WTI). (s. f.). https://contenido.bce.fin.ec/documentos/informacioneconomica/indicadores/general/PrecioPetroleo.html

Producción Petrolera Nacional. (s. f.). https://contenido.bce.fin.ec/documentos/informacioneconomica/indicadores/real/ProduccionPetroleraNacional.html

r2_score. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

statsmodels.tsa.arima.model.ARIMA - statsmodels 0.14.6. (s. f.). https://www.statsmodels.org/stable/generated/statsmodels.tsa.arima.model.ARIMA.html

Tapia, E. (2025, 13 julio). Petroecuador apaga sus principales pozos y la producción petrolera se desploma a dos semanas de suspensión de oleoductos. *Primicias*. https://www.primicias.ec/economia/petroecuador-apagon-principales-pozos-produccion-petrolera-caida-oleoductos-100546/

Team, K. (s. f.-a). *Keras documentation: Adam*. https://keras.io/api/optimizers/adam/

Team, K. (s. f.-b). *Keras documentation: Dense layer*. https://keras.io/api/layers/core_layers/dense/

Team, K. (s. f.-c). *Keras documentation: Dropout layer*. https://keras.io/api/layers/regularization_layers/dropout/

Team, K. (s. f.-d). *Keras documentation: EarlyStopping*. https://keras.io/api/callbacks/early_stopping/

Team, K. (s. f.-e). *Keras documentation: LSTM layer*. https://keras.io/api/layers/recurrent_layers/lstm/

Team, K. (s. f.-f). *Keras documentation: ReduceLROnPlateau*. https://keras.io/api/callbacks/reduce_lr_on_plateau/

Team, K. (s. f.-g). *Keras documentation: The Sequential class*. https://keras.io/api/models/sequential/

Tune XGBoost «gamma» Parameter | XGBoosting. (s. f.). https://xgboosting.com/tune-xgboost-gamma-parameter/

Welcome to Python.org. (2026, 4 junio). Python.org. https://www.python.org/

XGBoost Documentation — xgboost 3.2.1 documentation. (s. f.). https://xgboost.readthedocs.io/en/release_3.2.0/

Zambrano, L. (2026, 4 mayo). Ecuador pierde terreno petrolero mientras la región acelera su producción. *Diario Expreso*. https://www.expreso.ec/economia-y-negocios/ecuador-pierde-terreno-petrolero-region-acelera-produccion-280888.html?hl=es-US