GRADO EN BUSINESS ANALYTICS

TRABAJO FIN DE GRADO

ANÁLISIS DE LA DEMANDA DEL PETRÓLEO CON TECNICAS DE ML

Autor: **Álvaro Diez de Rivera De Solís**

Director: **Alejandro Rodríguez Gallego**

Madrid, 2023

INDICE

FIGURAS E ILUSTRACIONES

ABSTRACT

Español

En este estudio, se lleva a cabo un análisis de la demanda diaria de petróleo crudo en los Estados Unidos utilizando técnicas de Machine Learning (ML). A diferencia de investigaciones previas, la principal característica distintiva de este análisis es la adopción de una frecuencia diaria para el estudio de la demanda. Para ello, se recurre a la revisión de literatura y se extraen diversas variables relevantes para el análisis, obtenidas a partir de trabajos académicos previos.

Se implementan varios modelos de Machine Learning, como regresión lineal, KNN (k-vecinos más cercanos), Random Forest, redes neuronales y máquinas de vectores de soporte (SVR). Estos modelos son entrenados y evaluados en función de su capacidad para predecir la demanda diaria de petróleo crudo en el mercado estadounidense. Además habrá Nowcasting para ver la demanda en periodos diarios en vez de semanales (como se reportan).

Las conclusiones del estudio revelan que, debido a la hipótesis del mercado eficiente, los niveles de predicción alcanzados por los modelos son relativamente bajos. La hipótesis del mercado eficiente sostiene que la información disponible es rápidamente asimilada por los participantes del mercado, lo que dificulta la obtención de rendimientos superiores a través de estrategias de predicción basadas en datos históricos. En este contexto, los modelos de Machine Learning no logran predecir de manera precisa y consistente la demanda diaria de petróleo crudo en los Estados Unidos. No obstante, este trabajo contribuye al campo de la predicción de la demanda de petróleo al explorar el potencial y las limitaciones de las técnicas de ML en un marco de alta frecuencia y al proporcionar una visión detallada de la relación entre las variables relevantes y la demanda diaria de petróleo crudo en el país.

Palabras Clave: Petróleo crudo, Machine Learning, Nowcasting, USA, Predicción, Demanda

English

In this study, a daily analysis of crude oil demand in the United States is conducted using Machine Learning (ML) techniques. Unlike prior research, the main distinctive feature of this analysis is the adoption of a daily frequency for examining demand. To accomplish this, a literature review is conducted, and various relevant variables for the analysis are extracted from previous academic works.

Several Machine Learning models, such as linear regression, KNN (k-nearest neighbors), Random Forest, neural networks, and support vector machines (SVR), are implemented. These models are trained and evaluated based on their ability to predict daily crude oil demand in the US market.

The conclusions of the study reveal that, due to the efficient market hypothesis, the levels of prediction achieved by the models are relatively low. The efficient market hypothesis posits that available information is quickly assimilated by market participants, making it difficult to obtain superior returns through prediction strategies based on historical data. In this context, the Machine Learning models fail to predict the daily crude oil demand in the United States accurately and consistently. Nonetheless, this work contributes to the field of oil demand forecasting by exploring the potential and limitations of ML techniques in a high-frequency framework and by providing a detailed insight into the relationship between relevant variables and daily crude oil demand in the country.

Keywords: Crude Oil, Machine Learning, Nowcasting, USA, Forecast, Demand

INTRODUCCIÓN

El siguiente trabajo hace un análisis de la demanda del crudo, que es la fuente de energía más relevante en la actualidad a nivel global. El objetivo es hacer un modelo con técnicas de *Machine Learning* que sea capaz de predecir la cantidad demandada en Estados Unidos, país con mayor consumo.

En la actualidad se publican multitud de estudios sobre la demanda del petróleo. Existen diversas agencias que realizan análisis como la *Organización de Países Exportadores de Petróleo* (OPEC) o la Agencia Internacional de la Energía (EIA). Estos análisis son extremadamente complejos y extensos, como por ejemplo el *World Oil Outlook 2045* emitido por la OPEC. (*WOO 2022 - Home*, 2022.).

Otros estudios han usado técnicas de ML para realizar análisis de la oferta y la demanda, aunque en general su alcance ha sido nacional. Este proyecto se diferencia en que realizará un análisis de *nowcasting* en USA a frecuencia diaria. Aunque este modelo no puede acercarse a la realidad tanto como sería ideal por la cantidad de eventos imprevisibles (solo en los últimos años: revolución del shale americano, guerra comercial US-China, COVID, Guerra de Ucrania, etc.), aun así, es un buen ejercicio para tener una base sobre la que estudiar el mercado.

La metodología consiste en estudiar las variables más importantes, buscar la información de que variables pueden ser relevantes y posteriormente obtener los datos de esas variables. Posteriormente se creará una base de datos que contenga todos los datos, resolviendo problemas de frecuencias de muestreo, y ajustando para series temporales. Después, se realizará un análisis exploratorio para determinar las variables relevantes para posteriormente probar distintos modelos y analizar los resultados.

La estructura del trabajo consiste en una primera introducción al estado de la cuestión que hace un repaso del entorno energético actual, contextualizando al petróleo como fuente de energía a nivel histórico. Posteriormente, se mencionan las características que tiene (elasticidad, volatilidad, etc.) y que factores son los más relevantes para su estudio.

Después se buscan los datos, se crea el *dataset* y se hace el análisis exploratorio y los modelos.

Por último, se expondrán las conclusiones.

ESTADO DE LA CUESTIÓN

El mercado energético es un mercado apasionante y complejo, no solo por ser la base del crecimiento y del estado del bienestar del que la sociedad occidental disfruta, sino además porque en estos momentos (2022/23) estamos viviendo una crisis energética global que pone de manifiesto la importancia de entender este sector para garantizar prosperidad.

La energía existe de muchas formas distintas, y aunque generalmente se piense en electricidad también existen muchos otros productos y servicios que necesitan fuentes de energía concretas, como el combustible para los aviones o el diésel para las calderas. Este proyecto de fin de carrera se centra en el petróleo crudo, la fuente de energía más importante al representar casi un 25% de la energía consumida (Ritchie et al., 2022).

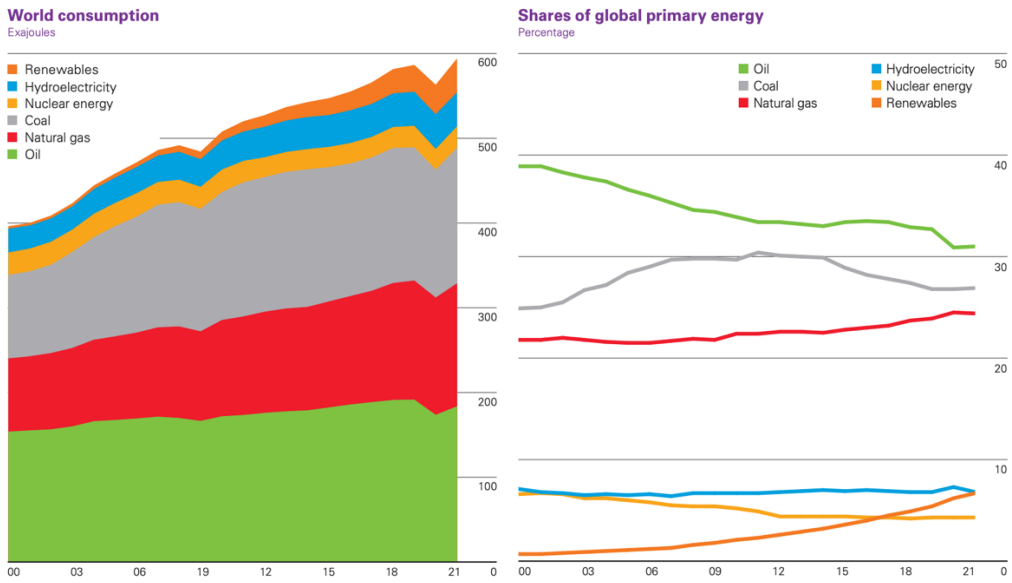

Ilustración 1 – Consumo Energético Global, EIA

Además de ser una fuente de energía imprescindible, el petróleo también tiene otras utilidades, como el asfalto, ciertos plásticos, etc. (*Use of Oil - U.S. Energy Information Administration (EIA)*, 2022). En la siguiente ilustración podemos ver el consumo de los diferentes productos derivados del petróleo, desglosados por sector.

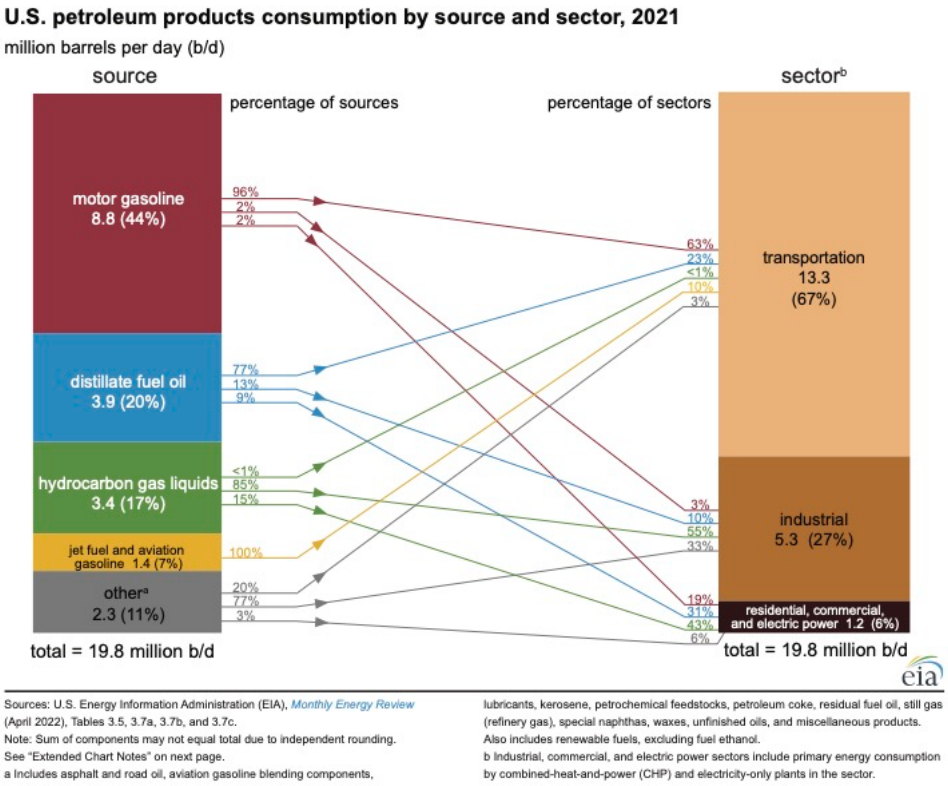

Ilustración 2 – Consumo en USA por origen de producto y tipo de consumo, EIA

Siendo tan importante este recurso los precios afectan directamente al crecimiento económico global. De hecho, en un estudio de las fluctuaciones del crudo (Abdelsalam, 2020) se detalla el impacto que tiene el precio del petróleo y su volatibilidad en el crecimiento económico. Traduciendo las conclusiones: "*El documento puede resumir los resultados de la siguiente manera: los cambios en el precio del petróleo y su volatilidad tienen un efecto opuesto para cada país exportador e importador de petróleo; para los primeros, los cambios en los precios del petróleo tienen un efecto positivo impacto, pero la volatilidad un efecto negativo. Mientras que, para los segundos, los cambios en los precios del petróleo tienen un efecto negativo, pero volatilidad un efecto positivo.*" (See abstract Abdelsalam, 2020)

La particularidad de este mercado es la extrema volatilidad. La razón fundamental para entender esta volatilidad es la inelasticidad tan grande tanto en el lado de la oferta como en el de la demanda. En el corto plazo, la elasticidad de la demanda es -0.06 mientras que en el largo plazo se sitúa cerca de -0.25 según un estudio del precio del crudo (Hamilton, 2009), estos niveles tan bajos se deben en parte a la ausencia de bienes sustitutos (no podemos usar

el coche sin combustible derivado del crudo) y en parte también a que los beneficios de usar petróleo son bienes considerados como necesarios, por lo que son los últimos en los que se reduce el gasto.

De forma similar al agua, que podría triplicarse en precio que aun así la demanda no disminuiría significativamente, la gasolina que consume un trabajador para llegar al trabajo tendría que aumentar mucho de precio para que ese individuo valorase emplear otro medio de transporte alternativo. Además, hay otros factores, por ejemplo, el desembolso inicial de un coche representa un monto muy superior al gasto anual en gasolina, por lo que los consumidores intentaran amortizar el coche usándolo lo máximo posible aprovechando que es un coste hundido y que el coste variable es relativamente bajo. Otra forma de ver este último argumento es que el gasto que suponen los productos derivados del petróleo es relativamente bajo en comparación con los ingresos de la mayoría de las sociedades (dependiendo de la parte del ciclo en la que se esté claro).

Hay muchos otros análisis publicados sobre la elasticidad de la demanda y la oferta como el del Board of Governors of the Federal Reserve System et al., 2016. Todos ellos coinciden en las causas principales de la gran inelasticidad observada en el crudo y sus derivados.

Un ejemplo de esta volatilidad se presenció en 2020, cuando se podía comprar un barril de crudo Brent por 30$, mientras que tan solo 2 años después, en 2022 el precio se multiplicó por 4 hasta tocar los 120$ bbl. Este ejemplo, aun siendo tan sorprendente, está lejos de ser el único que se ha vivido en esta industria. De hecho, la historia de la volatilidad de crudo es materia de estudio, como podemos ver en *el* libro relacionado con la volatibilidad de crudo de *Columbia University Press*, n.d. se puede dividir la historia del petróleo en varias etapas bien distinguidas en función de su volatibilidad.

En la primera etapa, tras el descubrimiento y perforación del primero pozo de petróleo en Pensilvania (1859), comenzó una etapa de volatilidad en el crudo de magnitudes inimaginables. Estamos hablando que en menos de dos años (1859-1861) el precio paso de 20$ a 10 centavos de dólar por barril.

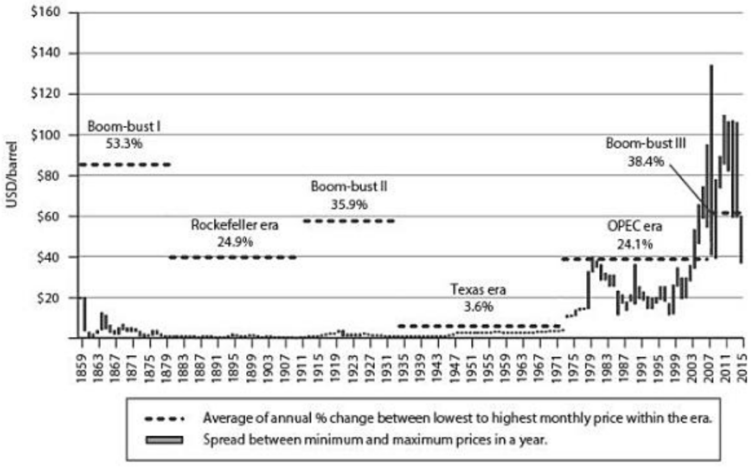

Ilustración 3 – Precio y volatilidad del crudo desde 1859, *Crude Volatility Columbia Press*

Para entender esto, es importante tener en cuenta que la utilidad inicial de crudo era la obtención de keroseno para obtener luz mediante lámparas. Es interesante, que en los primeros años la gasolina obtenida en la destilación del crudo era vista como un productor superfluo, debido a su escaso valor. Esta etapa continua con una volatilidad altísima debido a que el coste de perforación de un pozo (barrera de entrada) no era muy alto, por lo que, de forma parecida a la fiebre del oro, muchos soñadores decidieron probar suerte buscando petróleo. Muchos de los pozos salían secos, pero aun así el crecimiento de la oferta fue imparable.

Esta primera etapa, se caracterizó por el incansable intento de los productores, transportistas (ferrocarriles especialmente) y refinerías, de agruparse en carteles para controlar la producción y conseguir estabilidad de precios. El problema fue que la gran cantidad de agentes en el mercado unido la dificultad de regular y asegurar que los miembros cumplieran sus cuotas, hizo imposible que la colaboración fuera sostenible.

El fin de esta etapa ocurrió cuando *Standard Oil,* con Rockefeller al mando, consiguió obtener una mayoría del mercado, llegando a controlar el 90% de las refinerías de toto estados unidos. Este pseudo-monopolio consiguió estabilidad de precios, y al contrario de lo que se podría pensar, estos precios no fueron especialmente altos. Rockefeller era consciente de que la mejor forma de mantener y aumentar su influencia en el mercado era hacer uso de las ventajas de su escala para ofrecer mejores precios que la competencia y así conseguir

eliminarla, ya fuera comprándola o dejando que quebrase. Me parece digno de mención, que, aunque no sabemos que podría haber pasado de haber continuado el monopolio de Standard Oil, lo que sí sabemos es que tuvo ventajas para la economía y el resto de los agentes vinculados con la industria tener un precio relativamente bajo y, sobre todo, estable durante tantos años.

La tercera etapa se dio cuando los reguladores americanos decidieron en 1911 romper Standard Oil en más de 30 compañías distintas, incluyendo lo que hoy son Chevron y ExxonMobil. La razón, fue las prácticas monopolísticas de Standard Oil. Esta ruptura, sumado a que después de la Primera Guerra Mundial se empezaron a descubrir masivos nuevos yacimientos por el lado de la oferta y que la demanda creció a ritmos muy elevados, hizo que este periodo fuera muy volátil.

En esta etapa fue cuando el uso del petróleo para transporte supero por fin al uso para iluminación, las razones fueros por una parte que la luz eléctrica (gracias a la introducción comercial de la bombilla) sustituyo las lámparas de keroseno, y por otra que el aumento de los automóviles de combustibles derivados del petróleo aumento considerablemente.

No sería hasta la llamada era de Texas, caracterizada por fuerte control regulatorio, que se obtendría de nuevo estabilidad. Las herramientas utilizadas en este caso fueron un sistema de cuotas obligatorias, de las que el gobierno se hizo cargo de hacer cumplir. Los dos estados más relevantes, tanto por las medidas tomadas como por representar la mayoría de la producción americana, fueron Texas y Oklahoma. Estos estados llegaron a extremos nunca vistos para regular el precio del crudo.

Por ejemplo, Oklahoma después de pasar la ley *Conservation Act 1915* para limitar la producción a través del sistema cuotas intento restringir la oferta. En 1931 un tribunal federal rechazo la legalidad de esa ley, permitiendo a los productores seguir produciendo libremente, ante lo cual el gobernador William Henry Davis decidido ejercer el estado de emergencia y usar a la guardia nacional para que parara por la fuerza parte de los pozos. Esto se mantuvo hasta que el Tribunal Supremo de Estados Unidos, anulo la sentencia del tribunal inferior, permitiendo que se regulara la producción por ley. Este ejemplo enseña bien hasta qué punto muchos agentes estaban decididos a estabilizar los precios durante esta etapa.

La última etapa, comienza con el primer embargo de petróleo, y culmina con los eventos recientes como el absolutamente increíble crecimiento de China, la revolución del shale americano, etc. Las últimas décadas han sido tumultuosas. En los años setenta, la crisis del petróleo disparó los precios y provocó graves consecuencias económicas. Esta crisis fue causada por una serie de acontecimientos que incluyeron un embargo por parte de la Organización de Países Árabes Exportadores de Petróleo (OPEP) y una repentina disminución de la producción por parte de la Organización de Países Exportadores de Petróleo (OPEP). Esta crisis hizo que los precios del crudo se triplicaran, llegando el precio medio del barril de petróleo los 12 dólares.

En la década de los ochenta, los precios del crudo empezaron a bajar y la economía mundial comenzó a recuperarse. A mediados de los años ochenta, el precio del crudo cayó hasta situarse en torno a los 10 dólares el barril. Esto se debió en parte al auge de otras formas de energía, como el gas natural, así como al aumento de la producción de países no pertenecientes a la OPEP.

En la década de 1990, el precio del crudo empezó a subir de nuevo debido al aumento de la demanda de países en desarrollo como China e India. Esto condujo a un periodo de precios altos y sostenidos del crudo, con el precio medio del barril de petróleo por encima de los 30 dólares. A principios de la década de 2000 se produjo una rápida subida de los precios del crudo, alcanzando un máximo histórico de más de 140 dólares el barril en 2008 debido al aumento de la demanda mundial, así como a las tensiones geopolíticas en Oriente Medio.

Desde entonces, los precios del crudo han sido volátiles, con fuertes caídas debidas a la crisis financiera mundial de 2008 y la posterior recesión económica. En los últimos años, los precios se han mantenido relativamente estables, con el precio medio del barril de petróleo rondando los 50 dólares, hasta el COVID al menos. Esta etapa es la actual y por ahora parece que la volatilidad continua.

Las dinámicas de oferta y demanda en este mercado son especialmente interesantes por varios factores:

1. Mercado intensivo en capital
2. Existencia de un cartel con un 40% de la producción (OPEP+)

3. Disrupciones tecnológicas
4. Consideraciones climáticas en occidente
5. Alta sensibilidad a problemas geopolíticos

El proyecto está motivado por las disrupciones tecnológicas que se han experimentado en los últimos años en el campo del Machine Learning. Sería muy interesante comprobar cuán efectivo es un modelo entrenado para predecir la demanda futura; por ello, en este proyecto se desarrollará dicho modelo.

Analizar la demanda de este recurso requiere estudios extensos y complejos de una multitud de factores. Estos factores incluyen la actividad económica de los países, la variación del petróleo en el mix energético, el crecimiento de la población mundial y el aumento del consumo energético, entre otros.

En este proyecto, se entenderá la demanda como consumo final. En el precio influyen factores tan distintos como el arbitraje de los inventarios, los contratos de futuros y la propia percepción de precio futuro (especulación), tal como se detalla en el artículo de Hamilton (2009).

Hay diversas agencias que realizan análisis, como la Organización de Países Exportadores de Petróleo (OPEP) o la Agencia Internacional de la Energía (EIA). Estos análisis se centran en todos los factores mencionados y muchos más. Los distintos análisis son extremadamente complejos y extensos, como por ejemplo el *World Oil Outlook 2045* emitido por la OPEP (WOO 2022 - Home, 2022.).

En la siguiente gráfica, obtenida de EOA (Alhajji, 2023), se puede observar la predicción de demanda para 2023 realizada por las tres agencias más importantes: OPEP, *Energy Information Agency* e *International Energy Agency*.

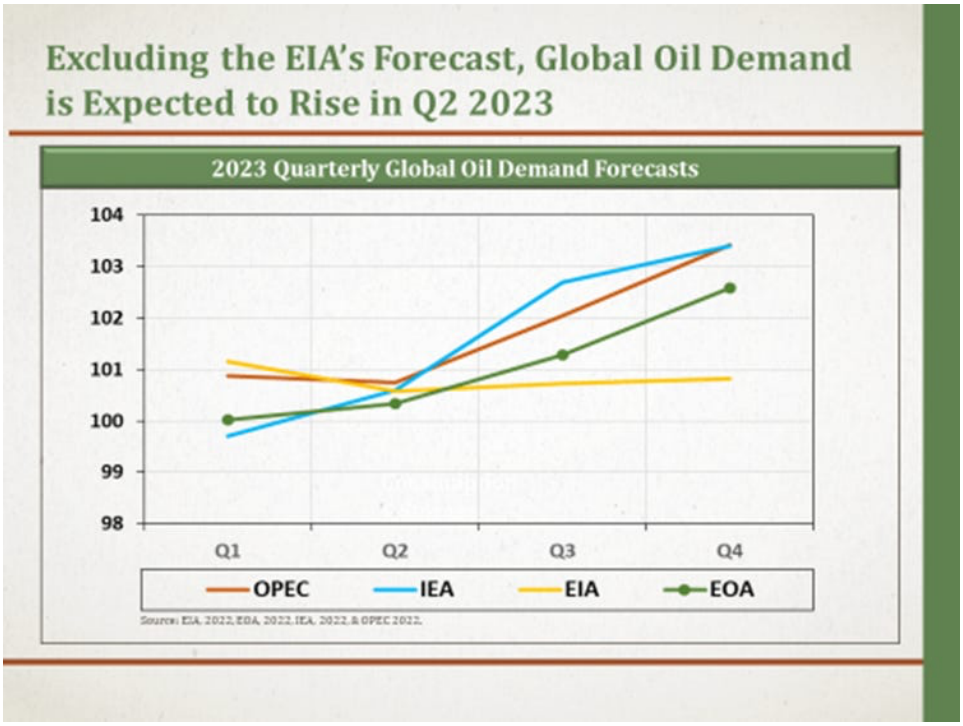

Ilustración 4 – Predicciones de la demanda de crudo en 2023, *Energy Outlook Advisors*

Podría parecer de primeras que las tres predicciones están en un rango muy reducido al ser esta cerca del 2%. Pero la realidad, es que en este mercado tan inelástico donde los costes hundidos son tan importantes, un pequeño cambio de 1-2M de barriles al día en la demanda pueden subir el precio decenas de dólares por barril. Tanto es así, que la perspectiva de perder 2-4 millones de barriles diarios de Rusia tras el comienzo de la guerra con Ucrania, hizo que los precios subieran de ~80$ a 130$.

También hay muchos otros que han comparado los análisis tradicionales con los que hacen uso de técnicas de machine Learning, por ejemplo, en el artículo (Obite et al., 2021) precisamente comparan distintos modelos usando información obtenida de Reuters y la NNPC (*Nigerian National Petroleum Corporation*) para realizar un estudio de la oferta de petróleo crudo en Nigeria anualmente.

Este artículo usa distintas técnicas como ARIMA, redes neuronales o arboles arbitrarios. Aunque los métodos utilizados son similares a los que usaremos en este proyecto, la diferencia es el enfoque. Este trabajo no se enfoca en la oferta en Nigeria, sino en la demanda en un único país (Estados Unidos) y con una frecuencia diaria.

Adicionalmente, cabe decir que la eficacia que se espera en tema de predicción es muy baja, esto se debe al concepto de eficiencia del mercado fue propuesto por Eugene Fama en su hipótesis del mercado eficiente (Fama, 1970) , EMH por sus siglas en inglés. Esta teoría sugiere que los precios de los activos financieros reflejan toda la información disponible en el mercado y, por lo tanto, es imposible superar al mercado de forma consistente a través de la selección de activos o el análisis técnico.

Existen tres formas de la hipótesis del mercado eficiente:

1. Eficiencia débil: En esta forma, se supone que los precios de los activos reflejan toda la información histórica, incluidos los precios pasados y los volúmenes de negociación. Por lo tanto, no es posible obtener beneficios mediante estrategias de análisis técnico basadas en datos históricos.

2. Eficiencia semi-fuerte: Además de la información histórica, se supone que los precios de los activos también reflejan toda la información pública, como las noticias financieras y los informes de las empresas. En este caso, tampoco es posible obtener beneficios utilizando análisis técnico o fundamental basado en información pública.

3. Eficiencia fuerte: En esta forma, se supone que los precios de los activos reflejan toda la información, tanto pública como privada. En otras palabras, incluso si un inversor tuviera acceso a información interna o privilegiada, no podría obtener beneficios de manera consistente.

Considerando que el mercado es eficiente, no se debería ser capaz de predecir la demanda, con gran exactitud ya que esto daría pie a oportunidades de arbitraje entre otras cosas. En la conclusión se extenderá en esta reflexión adaptándolo a los resultados del proyecto

Estudio de variables independientes

La primera parte consiste en estudiar que variables son más importantes para determinar la demanda. Entre estas está el tamaño de la economía, el crecimiento de la población, el número de vehículos poseídos por persona, las políticas del gobierno, los precios del crudo

(usaremos el precio americano del *West Texas Intermidiate* como proxy, al ser el índice más importante en USA), la eficiencia energética, el tiempo, los avances tecnológicos, las preferencias de los consumidores, los tipos de interés, los inventarios, los combustibles sustitutos, etc.

Para hacer la selección final de variables, se realiza un repaso de la literatura, para ver que variables son generalmente consideradas las más importantes. Esto consigue un primer filtro que asegurará que las variables tengan relevancia. De los distintos paper que se han revisado, hay que tener en cuenta que solo se va a coger una pequeña selección de variables. El criterio consiste en tener variables de la mayoría de las categorías que he encontrado en los distintos *papers*, como, por ejemplo: oferta, demanda, mercados, materias primas, economía, geopolítica y especulación.

Se comienza obteniendo la variable dependiente, la cual será la demanda obtenida de la EIA con sus análisis semanales y ajustadas a frecuencia diaria. Esta es la variable que finalmente se desea predecir.

Para realizar la selección de las variables independientes, se lleva a cabo un estudio de la literatura con el fin de determinar qué variables tiene sentido utilizar. Se inicia con un artículo sobre técnicas para la obtención de variables y sus datos relevantes para el estudio de la demanda del crudo (Calomiris et al., 2020). En este artículo, se proporcionan distintas ideas de variables relevantes para la demanda del crudo. Este estudio examina la eficacia de variables tradicionales y nuevas basadas en textos para predecir, dentro y fuera de la muestra, los rendimientos del precio del petróleo en spot, de los futuros y de las acciones de las empresas energéticas, así como los cambios en la volatilidad, la producción y los inventarios de petróleo.

De este Paper nos quedamos con **SP500**, tanto el **índice** como el **cambio** experimentado a nivel diario. También añadiremos el índice **VIX** (CBOE Market Volatility Index) como proxy del miedo en los mercados. Por último, también se hará uso de una variable para medir los niveles de los inventarios, en este caso será los **niveles de inventarios** de crudo y productos en Estados Unidos, que son publicados semanalmente por la *Energy Information Agency*.

Tabla 1: Ejemplos de variables de *"Mining for Oil Forecasts",* (Baumeister et al., 2022)

Table I
Data Definitions Summary

Variable	Definition
Dependent Variables	
FutRet^{8}	WTI front-month futures cumulative weekly returns (in %) from the end of week *t* to the end of week *t+*8
DSpot^{8}	Percent change in the WTI spot price from the end of week *t* to the end of week *t+*8
DOilVol^{8}	Level difference in the rolling 30-day realized volatility of WTI physical futures 1-month nearby contract from the end of week *t* to the end of week *t+8*
xomRet^{8}	Exxon Mobil stock returns (in %) from the end of week *t* to the end of week *t+*8 (trades on NYSE)
bpRet^{8}	British Petrol stock returns from the end of week *t* to the end of week *t+8* (ADR trading on NYSE)
rdsaRet^{8}	Royal Dutch Shell class A stock returns from Monday of week *t+1* to Monday of week *t+9* (trades on Euronext)
DInv^{8}	Percent change in U.S. crude inventories including SPR (EOP, mil. bbl) from the end of week *t* to the end of week *t+8*
DProd^{8}	Average weekly percent change in U.S. crude oil field production (mil. bbl/day) from the end of week *t* to the end of week *t+8*

Del paper de mercados energéticos de Baumeister et al (2022), obtenemos también información muy valiosa para la elección de las variables dependientes. El paper evalúa la utilidad de indicadores alternativos de actividad económica global y datos fundamentales para la correcta predicción de precios del crudo y consumo global. Para ello combina medidas de distintas fuentes, obteniendo así nuevas medidas para analizar el mercado.

La utilidad de este paper es evidente, si este trabajo quiere predecir la demanda del petróleo usando indicadores, las conclusiones de un estudio dedicado precisamente a encontrar los indicadores más relevantes, son especialmente interesantes.

Como podemos observar en la siguiente tabla obtenida del paper en cuestión hay una variedad de datos de distintas temáticas que afectan. En este sentido cogeremos de aquí un indicador de la actividad económica real (**Producto Interior Bruto** con frecuencia trimestral). Para la categoría de transporte además de usar un proxy para millas conducidas (**número de vehículos** con frecuencia trimestral), usaremos también **número de vuelos**, esto es porque son datos con mayor (frecuencia mensual, en vez de trimestral) por lo que nos servirá para mejorar la calidad de los datos de esta categoría para mejorar la predicción del consumo.

Otra categoría clave de indicador que hay que añadir es un buen indicador financiero, para esto usaremos el ***U.S. Dollar Index*** (DXY), este índice mide la fuerza de la moneda estadounidense en comparación con el resto de las monedas. Como podemos esperar, al

estar el precio del petróleo en dólares, cuando el dólar sube, el resto del mundo tiene más dificultad para comprar crudo por lo que la demanda mundial bajara y el precio también, permitiendo a Estados Unidos beneficiarse. Otro índice importante será el índice de volatilidad (**VIX**) ya que nos dará información sobre el movimiento y sentimiento del mercado en general.

Tabla 2: Indicadores Mercado Energético de "Energy Markets and Global Economic Condition", (Miao et al., 2017)

Data category	Variable	Geographic coverage	Start date	Tcode	Data source	Data delay	Nowcast rule	Data revisions
Real economic activity	World Industrial Production Index	OECD + 6 non-member countries	1973.1	5	Baumeister-Hamilton (2019)	2	AG	Y
	Conference Board Leading Economic Index	US	1973.1	5	Datastream	1	AG	Y
	Consumer Confidence Index	OECD	1974.1	5	OECD MEI	1	RW	Y
Commodity prices	Copper Price	World	1973.1	5	World Bank	0		N
Financial indicators	Real Trade-Weighted U.S. Dollar Index: Broad	World	1973.1	5	FRED	0		Y
	MSCI World Stock Price Index	World	1972.1	7	Global Financial Data	0		N
	Excess Returns on Fama-French Portfolio: Transportation	World	1973.1	7	Ken French's website	2	RW	N
Transportation	Passenger Car Registrations	OECD	1973.1	5	OECD MEI	8	AG	Y
	Total Vehicle Miles Travelled	US	1973.1	5	FRED	2	AG	Y
Uncertainty measures	Caldara-Iacoviello Geopolitical Risk Index	World	1973.1	5	Caldara-Iacoviello (2018)	0		N
	Long-Run Oil Price Uncertainty	World	1989.4	1	Bloomberg	0		N
Expectations measures	University of Michigan Index of Consumer Expectations	US	1978.1	5	Michigan Survey	0		N
	Spread between Long-Run and Short-Run Oil Price Expectations	World	1988.11	1	Bloomberg	0		N
Weather indicators	Oceanic Niño Index	World	1973.1	1	NOAA	2	RC	Y
	Residential Energy Demand Temperature Index	US	1973.1	1	NOAA	1	RW	Y
Energy-related indicators	Energy Production and Electricity Distribution	EU28	1991.1	5	FRED	3	AG	Y

NOTES: Tcode indicates the transformation of the variable where 1 indicates the variable is included in its original units, 5 refers to taking first log differences, 7 stands for annual growth rates. The delay in data release is measured in months. Nowcasts are based on the average growth rate (AG), the most recent change (RC), and no change (RW).

Por último, del paper de Miao et al. (2017) obtenemos información muy relevante, ya que esté paper identifica distintos factores para predecir el precio del crudo. Aunque no es exactamente lo que buscamos, la relación intrínseca entre precio y demanda hace que ciertos factores sean comunes. Este paper identifica seis categorías de factores influyentes: oferta, demanda, mercados, materias primas, geopolítica y especulación. De aquí, usaremos el **precio del cobre** para mejorar nuestra categoría de materias primas, además el precio del cobre también lo usa en el paper de Calomiris et al (2020) por ser muy buen proxy. Por último, añadiremos la **cotización de empresas petroleras** como dato de la industria. Para esto usaremos el XLE, índice de empresas energéticas del SP500. A partir de 2006 aparece un índice

más específico para empresas de petróleo y gas, pero tendremos que usar el primero para tener datos antes del 2006.

Este resumen aborda un proyecto de investigación que analiza las fuentes clave, incluidos estudios y análisis publicados por la Organización de Países Exportadores de Petróleo (OPEP), la Agencia Internacional de la Energía (EIA) y otros académicos que han investigado la demanda de petróleo y la elasticidad del precio. Los conceptos clave tratados en este trabajo son la elasticidad, la volatilidad y la inelasticidad de la oferta y la demanda de petróleo.

El objetivo del proyecto es responder a preguntas relacionadas con la predicción de la demanda de petróleo en Estados Unidos utilizando técnicas de Machine Learning (ML). Los principales desafíos a abordar incluyen la identificación y selección de variables relevantes, la creación de un conjunto de datos y el ajuste de series temporales.

Para abordar estos desafíos, se investigan las variables más importantes, se obtienen datos relevantes y se crea una base de datos que contiene la información necesaria. Se lleva a cabo un análisis exploratorio para determinar las variables relevantes y se prueban distintos modelos de ML para analizar los resultados.

El análisis de la demanda de petróleo surge de la importancia del petróleo como fuente de energía a nivel mundial y su impacto en el crecimiento económico global. La volatilidad en el mercado del petróleo proviene de la inelasticidad tanto en la oferta como en la demanda, la existencia de un cartel con un 40% de la producción (OPEP+), disrupciones tecnológicas, consideraciones climáticas en occidente y alta sensibilidad a problemas geopolíticos.

Los debates y preguntas clave en torno al tema incluyen la efectividad de los modelos de ML para predecir la demanda futura de petróleo, cómo abordar la volatilidad en el mercado energético y cómo las disrupciones tecnológicas y las consideraciones climáticas afectan la oferta y la demanda de petróleo. Este proyecto se centra en la aplicación de técnicas de ML para predecir la demanda de petróleo a nivel nacional en Estados Unidos, con una frecuencia diaria y por tanto utilizando un enfoque de *Nowcasting* (los datos oficiales son semanales).

OBJETIVOS DEL PROYECTO

1. El objetivo principal del proyecto es la creación de un modelo de predicción de la demanda del petróleo en Estados Unidos a nivel diario y el posterior análisis de la eficacia de este. Este objetivo incluye encontrar varios modelos distintos y compara su eficacia.
2. Para poder medir la eficacia tendremos que usar los datos semanas de inventarios obtenido por la Energy Information Agency. El objetivo secundario es ser capaces de encontrar que variables de las disponibles aportan más al modelo. Aquí la idea consiste, en ver si hay un par de atributos que concentran la mayor parte de la carga explicativa.

Si la creación de un modelo que mida con precisión las variaciones en la demanda de petróleo crudo en Estados Unidos a nivel diario se completará con éxito, este tendría numerosas utilidades en distintos sectores y contextos. A continuación, se presentan algunas aplicaciones posibles:

1. **Planificación en la industria petrolera**: Las empresas petroleras podrían utilizar las predicciones para mejorar la eficiencia en la producción, distribución y almacenamiento de petróleo crudo. Una mejor comprensión de las fluctuaciones en la demanda les permitiría ajustar sus operaciones para satisfacer la demanda sin incurrir en excesos de capacidad o escasez de suministro.

2. **Estrategias de comercialización y ventas**: Las empresas que venden productos derivados del petróleo, como gasolina, diésel y productos petroquímicos, podrían beneficiarse al adaptar sus estrategias de comercialización y ventas en función de las predicciones de demanda. Esto les permitiría anticiparse a las tendencias del mercado y optimizar sus operaciones para satisfacer la demanda y maximizar las ganancias.

3. **Política energética y seguridad energética**: Los responsables políticos y reguladores podrían utilizar las predicciones de la demanda de petróleo crudo para diseñar políticas energéticas eficientes y promover la seguridad energética. Al comprender las fluctuaciones en la demanda, podrían tomar decisiones informadas

sobre la inversión en infraestructura energética, la diversificación de fuentes de energía y la promoción de energías alternativas y renovables.

4. **Gestión de riesgos financieros y comerciales**: Los inversores y las empresas que operan en los mercados de energía podrían utilizar las predicciones de la demanda de petróleo crudo para gestionar sus riesgos financieros y comerciales. Con una comprensión más clara de las fluctuaciones en la demanda, podrían tomar decisiones más informadas sobre la compra y venta de contratos de futuros de petróleo, opciones y otros instrumentos financieros relacionados con el petróleo.

5. **Investigación económica y modelado macroeconómico**: Los economistas y analistas podrían utilizar las predicciones de la demanda de petróleo crudo como un insumo clave en sus modelos macroeconómicos. Dado que el petróleo es un recurso esencial y su demanda está estrechamente relacionada con el crecimiento económico, contar con predicciones precisas de la demanda podría mejorar la calidad y precisión de los pronósticos económicos y las estimaciones de política.

6. **Operaciones en el sector del transporte**: Las compañías de transporte, incluidas las aerolíneas, las empresas navieras y las empresas de transporte terrestre, podrían beneficiarse al anticipar las fluctuaciones en la demanda de petróleo crudo y sus derivados. Esto les permitiría ajustar sus operaciones y planificar sus compras de combustible para minimizar los costes y garantizar la continuidad del servicio.

7. **Análisis de riesgo geopolítico**: La demanda de petróleo crudo puede verse afectada por eventos geopolíticos y tensiones internacionales. Las predicciones precisas de la demanda podrían ayudar a los analistas y gobiernos a evaluar el impacto potencial de estos eventos en los mercados de energía y en la economía en general

Se profundizará en dichas conclusiones si se logra obtener un modelo con capacidades de predicción adecuadas y satisfactorias.

METODOLOGÍA DE TRABAJO

1. OBTENCION DE LOS DATOS DE LAS VARIABLES INDEPENDIENTES

Se obtienen los datos de las variables mencionadas en el estado del arte. En la tabla se puede observar las fuentes de información para cada variable y la frecuencia muestral de los datos obtenidos.

Tabla 3: Fuentes de Variables Independientes, Elaboración Propia

WTI price	SP500 Price	SP500 Change	Petroleum inventories	Vuelos	GDP	Motor Vehicle Prod	U.S. Dollar Index	Precio del cobre	Cotizacion petroleras XLM	Demanda (boed)
EIA	Factset	Factset	EIA	Bureau of Transportation Statistics	U.S. Bureau of Economic Analysis	U.S. Bureau of Economic Analysis	Factset	Factset	Factse	EIA
Diario	Diario	Diario	Semanal	Mensual	Trimestral	Trimestral	Diario	Diario	Diario	Semanal

Una vez localizados los datos, la información es procesada para posteriormente utilizarla en el modelo. Esta información viene representada en una base de datos suficientemente grande para que el modelo sea relevante. Al estar utilizando frecuencia diaria (días de bolsa abierta) la base de datos contará con 6000 puntos aproximadamente. Esto permitirá emplear técnicas altamente avanzadas de Machine Learning.

La base de datos será generada uniendo los datos obtenidos de distintas fuentes, usando una combinación de Python con la librería de Pandas y Excel. Este procesado incluye limpieza de NAs, regulación de frecuencias, eliminación de variables innecesarias, etc. Por último, se realizará un estudio descriptivo de la fuente de datos. Aquí se describirán las características más importantes de la fuente de datos y cualquier nota importante.

2. CREACIÓN DE LOS MODELOS

En esta segunda fase se procederá a elaborar el modelo más eficiente para los datos disponibles. Primero se emplearán modelos explicativos para posteriormente usar

los predictivos, dividiendo los datos en un segmento para entrenar y otro para comprobar resultados (test).

Entre los modelos predictivos a utilizar se encuentran regresiones lineales, arboles de decisión, *Random Forest* e incluso redes neuronales. Se emplearán únicamente los que arrojen los resultados más relevantes. A continuación, se detallan los modelos seleccionados:

1. **Regresión Lineal:** La Regresión Lineal es un algoritmo de aprendizaje supervisado simple y ampliamente utilizado que modela la relación entre una variable dependiente (en este caso, demanda) y una o más variables independientes (las otras características del conjunto de datos). Asume una relación lineal entre las características de entrada y la salida, lo que facilita su interpretación e implementación. En el paper realizado por Kumari & Yadav (2018) se cuentan las limitaciones que tiene está técnica, aunque también se exponen algunas ventajas.

   Las principales razones por las que la Regresión Lineal ha sido considerada como un buen modelo para predecir la demanda son:

   - **Simplicidad**: La Regresión Lineal es fácil de entender, implementar e interpretar, lo que la convierte en un buen punto de partida para el pronóstico de series temporales.
   - **Velocidad:** Debido a su simplicidad, es computacionalmente eficiente y puede manejar grandes conjuntos de datos.
   - **Interpretable:** Los coeficientes del modelo se pueden interpretar como la importancia relativa de cada característica para predecir la demanda.

   Sin embargo, la Regresión Lineal puede tener dificultades con las relaciones complejas entre las características de entrada y la salida, y es posible que no capte la estacionalidad o las tendencias inherentes en los datos de series temporales.

2. **Regresión Ridge**: La Regresión Ridge es una extensión de la Regresión Lineal que introduce un término de regularización para evitar el sobreajuste. Es una forma de regularización L2, que agrega un término de penalización proporcional al cuadrado de la magnitud de los coeficientes.

   Ha sido considerada una buena opción para pronosticar la demanda gracias a su:

   - **Regularización:** La Regresión Ridge ayuda a prevenir el sobreajuste, haciéndola más robusta al ruido y las fluctuaciones en los datos.
   - **Selección de características**: Puede manejar multicolinealidad entre las características de entrada, lo que puede ser útil cuando hay variables altamente correlacionadas en el conjunto de datos.
   - **Estabilidad**: La Regresión Ridge puede producir modelos estables y más generalizables.

   Sin embargo, la Regresión Ridge todavía asume una relación lineal entre las características de entrada y la salida, lo que podría no ser suficiente para datos complejos de series temporales.

3. **Regresión de Máquinas de Vectores de Soporte (SVR)**: La SVR es un tipo de Máquina de Vectores de Soporte (SVM) que se utiliza para tareas de regresión. Funciona encontrando un límite de decisión (o hiperplano) que se ajusta mejor a los datos mientras mantiene el margen máximo entre los valores predichos y los valores reales. En el paper de Drucker et al. (1996), se comparan las distintas características con Ridge y Regresión Lineal obteniendo resultados muy positivos.

   Las principales razones por las que se ha elegido este modelo para pronosticar la demanda son:

   1. **Flexibilidad**: La SVR puede modelar relaciones no lineales entre las características de entrada y la salida utilizando funciones kernel, lo que la hace más versátil que los modelos lineales.

2. **Robustez**: La SVR es menos sensible a los valores atípicos, ya que se centra en los puntos de datos más cercanos al límite de decisión.

3. **Ajustable**: La SVR tiene varios hiperparámetros (como la función kernel y los parámetros de regularización) que se pueden ajustar para mejorar el rendimiento del modelo.

Sin embargo, la SVR puede ser computacionalmente costosa para grandes conjuntos de datos y podría requerir un ajuste sustancial de parámetros para obtener los mejores resultados

4. **Red Neuronal *Feedforward* (FFNN)**: Una Red Neuronal *Feedforward* es un tipo de red neuronal artificial que consta de múltiples capas de neuronas interconectadas. Puede modelar relaciones complejas y no lineales entre las características de entrada y la salida, por lo que es considerada una herramienta poderosa para una amplia gama de tareas, incluido el pronóstico de series temporales. Las redes neuronales ya han sido capaces de resolver problemas que ninguna otra técnica de *Machine Learning* ha podido resolver, como se explica en el paper de Hochreiter & Schmidhuber (1997).

   Ha sido seleccionado para predecir la demanda por:

   - **Modelado no lineal**: Las FFNN pueden modelar relaciones complejas y no lineales entre las características de entrada y la salida, lo que las hace adecuadas para datos de series temporales que exhiben estacionalidad, tendencias u otros patrones.

   - **Flexibilidad**: La arquitectura de las FFNN se puede adaptar para adaptarse al problema en cuestión, con diferentes números de capas y neuronas, funciones de activación y técnicas de optimización.

   - **Aprendizaje de características**: Las FFNN pueden aprender automáticamente características relevantes de los datos, lo que reduce la necesidad de ingeniería manual de características.

   Sin embargo, las FFNN pueden ser intensivas en cómputo, propensas al sobreajuste y difíciles de interpretar en comparación con modelos más simples.

Además, pueden requerir un ajuste sustancial de hiperparámetros y mayores cantidades de datos para lograr un buen rendimiento.

5. **Bosques Aleatorios (*Random Forest*):** El Bosque Aleatorio es un método de aprendizaje conjunto que se puede utilizar para tareas de clasificación y regresión. Funciona construyendo múltiples árboles de decisión durante el entrenamiento y genera la predicción media de los árboles individuales para tareas de regresión, como pronosticar la "demanda". Su utilidad queda bien probada en el paper donde se demuestra su eficacia para problemas tan complejos como el de detectar la caligrafía de distintas personas (Ho, 1995).

   Las características principales por las que resulta útil a la hora de pronosticar la demanda son:

   - **Relaciones no lineales:** Los Bosques Aleatorios pueden capturar relaciones no lineales entre las características y la variable objetivo, lo cual puede ser común en conjuntos de datos del mundo real. En comparación, los modelos lineales como la regresión lineal podrían tener dificultades con patrones no lineales.

   - **Interacciones entre características**: Los Bosques Aleatorios pueden manejar interacciones complejas entre características, lo cual es útil cuando las relaciones entre las características y la variable objetivo no son fácilmente separables.

   - **Robustez ante valores atípicos:** Los Bosques Aleatorios son menos sensibles a los valores atípicos en el conjunto de datos porque utilizan un mecanismo de voto mayoritario o promedio de los árboles individuales. Esto los hace más robustos que los modelos que se ven muy afectados por los valores atípicos, como la regresión lineal o las máquinas de vectores de soporte.

- **Manejo de valores faltantes**: Los Bosques Aleatorios pueden manejar eficazmente los valores faltantes. Pueden dividir los datos según la presencia o ausencia de un valor o utilizar divisiones sustitutas para encontrar la mejor división posible para los valores faltantes. En comparación, otros modelos requieren imputar valores faltantes antes del entrenamiento.

- **Importancia de las características**: Los Bosques Aleatorios pueden proporcionar puntuaciones de importancia de las características, lo que puede ser útil para comprender qué características contribuyen más a las predicciones del modelo. Esto puede ser valioso para la selección de características, interpretación y comprensión de la estructura subyacente de los datos.

- **Reducción del sobreajuste**: Los Bosques Aleatorios utilizan el bagging (agregación de bootstrap) y la selección aleatoria de características para crear árboles individuales diversos, lo que puede ayudar a reducir el sobreajuste. Esto los hace más generalizables a nuevos datos no vistos en comparación con los modelos propensos al sobreajuste, como árboles de decisión individuales.

- **Paralelizable y escalable**: Los Bosques Aleatorios son inherentemente paralelizables, ya que cada árbol de decisión se puede entrenar de forma independiente. Esto los hace adecuados para conjuntos de datos grandes y entornos de computación de alto rendimiento.

Sin embargo, es esencial considerar que los Bosques Aleatorios podrían no ser siempre la mejor opción para tareas de pronóstico de series temporales, especialmente cuando hay un componente temporal fuerte en los datos. Modelos como ARIMA, SARIMA o redes neuronales LSTM están diseñados específicamente para capturar dependencias temporales y podrían superar a los Bosques Aleatorios en ciertas tareas de pronóstico de series temporales.

En resumen, los Bosques Aleatorios pueden ser una buena opción para predecir la "demanda" debido a su capacidad para manejar relaciones no lineales, interacciones entre características, robustez ante valores atípicos y reducción del sobreajuste. Sin embargo, si los datos de la serie temporal tienen patrones temporales fuertes, puede valer la pena considerar otros modelos específicos de series temporales también.

6. K-Vecinos más cercanos (KNN): El algoritmo de K-Vecinos más cercanos es un método de aprendizaje supervisado que puede utilizarse para tareas de clasificación y regresión. Funciona encontrando los k vecinos más cercanos en el conjunto de entrenamiento para una instancia de prueba y promediando sus valores objetivo para tareas de regresión, como pronosticar la "demanda". Su utilidad es bien conocida en problemas como el reconocimiento de patrones y la clasificación (Altman, 1992).

   Sería una buena opción para pronosticar la demanda ya que:

   - **Relaciones no lineales**: El algoritmo KNN puede capturar relaciones no lineales entre las características y la variable objetivo, lo cual puede ser común en conjuntos de datos del mundo real. En comparación, los modelos lineales como la regresión lineal podrían tener dificultades con patrones no lineales.
   - **Interacciones entre características**: El algoritmo KNN puede manejar interacciones complejas entre características, lo cual es útil cuando las relaciones entre las características y la variable objetivo no son fácilmente separables.
   - **Robustez ante valores atípicos:** KNN es menos sensible a los valores atípicos en el conjunto de datos, ya que utiliza un mecanismo de voto mayoritario o promedio de los k vecinos más cercanos. Esto lo hace más robusto que los modelos que se ven muy afectados por los valores atípicos, como la regresión lineal o las máquinas de vectores de soporte.
   - **Escalabilidad**: KNN se adapta fácilmente a conjuntos de datos más grandes y es fácil de actualizar con nuevos datos. Sin embargo, puede

volverse más lento a medida que el conjunto de datos crece, especialmente en términos de tiempo de consulta.

- **Simpleza:** KNN es un algoritmo simple y fácil de entender, lo que puede ser útil en casos en los que se requiere una explicación intuitiva del modelo.

Sin embargo, es esencial considerar que el algoritmo KNN podría no ser siempre la mejor opción para tareas de pronóstico de series temporales, especialmente cuando hay un componente temporal fuerte en los datos. Modelos como ARIMA, SARIMA o redes neuronales LSTM están diseñados específicamente para capturar dependencias temporales y podrían superar a KNN en ciertas tareas de pronóstico de series temporales.

Además, KNN puede verse afectado por la "maldición de la dimensionalidad" cuando se enfrenta a un gran número de características, lo que puede requerir técnicas de selección y reducción de características.

En resumen, el algoritmo KNN puede ser una buena opción para predecir la "demanda" debido a su capacidad para manejar relaciones no lineales e interacciones entre características, y su robustez ante valores atípicos. Sin embargo, si los datos de la serie temporal tienen patrones temporales fuertes o un gran número de características, puede valer la pena considerar otros modelos específicos de series temporales o utilizar técnicas de reducción de dimensionalidad.

Conclusión de los modelos

Al considerar estos modelos para el pronóstico de series temporales, es fundamental tener en cuenta las dependencias temporales en los datos. Técnicas como la diferenciación, el rezago y las ventanas deslizantes pueden ser útiles para capturar la información temporal en los datos y mejorar el rendimiento del modelo. Además, es importante evaluar y comparar el rendimiento de cada modelo utilizando métricas adecuadas y validación cruzada en el tiempo para seleccionar el modelo más apropiado para predecir la demanda en función de su conjunto de datos específico.

Finalmente se añadirá un modelo naive que siempre de como previsión que el cambio será 0. Esto servirá para entender como de buenos son los modelos comparándolos con un modelo aleatorio.

3. ANÁLISIS DEL MODELO

Por último, se procederá a analizar la efectividad del modelo y su precisión. El uso de métricas como el Error Cuadrático Medio (RMSE, por sus siglas en inglés, *Root Mean Squared Error*) y el Error Absoluto Medio (MAE, por sus siglas en inglés, *Mean Absolute Error*) para comparar diferentes modelos de regresión es una buena forma de evaluar su rendimiento y seleccionar el más adecuado. En la siguiente explicación, se describen los conceptos subyacentes a cada métrica y por qué son útiles para comparar modelos de regresión en nuestro estudio de demanda.

1. **RMSE (Root Mean Squared Error):** El RMSE es una métrica que mide la diferencia entre los valores reales y los valores predichos por un modelo de regresión. Se calcula tomando la raíz cuadrada de la media de las diferencias al cuadrado entre los valores reales y predichos. En otras palabras:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2}$$

   donde "yi" son los valores reales, "ŷi" son los valores predichos, y "n" es el número de observaciones.

   El RMSE tiene las siguientes características:

   - Se encuentra en la misma escala que los datos originales, lo que facilita su interpretación.
   - Es más sensible a errores grandes que otras métricas, como el MAE, ya que los errores se elevan al cuadrado antes de ser promediados.
   - Penaliza más los errores grandes que los errores pequeños, lo que puede ser útil si es importante minimizar errores extremos en las predicciones.

2. **MAE (Mean Absolute Error**): El MAE es otra métrica que mide la diferencia entre los valores reales y los valores predichos por un modelo de regresión. Se calcula tomando

el promedio de las diferencias absolutas entre los valores reales y predichos. En otras palabras:

$$MAE = \frac{\Sigma|yi - \hat{y}i|}{n}$$

donde "yi" son los valores reales, "ŷi" son los valores predichos, y "n" es el número de observaciones.

El MAE tiene las siguientes características:

- Al igual que el RMSE, se encuentra en la misma escala que los datos originales, facilitando su interpretación.
- Es menos sensible a errores grandes que el RMSE, ya que los errores no se elevan al cuadrado antes de ser promediados.
- Proporciona una medida de error más "robusta" en presencia de valores atípicos, ya que no penaliza errores extremos tanto como el RMSE.

La comparación de modelos de regresión usando RMSE y MAE es útil porque:

1. Ambas métricas son fáciles de entender e interpretar, ya que se encuentran en la misma escala que los datos originales.
2. El RMSE y el MAE ofrecen diferentes perspectivas sobre el error de predicción. Mientras que el RMSE es más sensible a errores grandes y puede ser útil para minimizar errores extremos, el MAE es más robusto en presencia de valores atípicos y proporciona una medida de error que no penaliza tanto los errores grandes.
3. El uso conjunto de RMSE y MAE puede ofrecer una visión más completa del rendimiento de un modelo de regresión, permitiendo a los analistas y científicos de datos seleccionar el modelo que mejor se adapte a sus necesidades específicas.

En resumen, el uso de RMSE y MAE para comparar diferentes modelos de regresión es una buena práctica, ya que ambas métricas proporcionan información valiosa sobre el rendimiento de los modelos y ayudan a identificar sus fortalezas y debilidades. Al considerar tanto el RMSE como el MAE, los analistas y científicos de datos pueden tomar decisiones informadas sobre qué modelo es el más adecuado para su aplicación específica, teniendo en cuenta la importancia de minimizar errores extremos y la robustez frente a valores atípicos.

Además, al comparar modelos usando estas métricas, se pueden identificar áreas de mejora en los modelos y ajustar sus parámetros o enfoques para optimizar su rendimiento. También es posible utilizar estas métricas como criterio de selección en procesos de ajuste de hiperparámetros, como la validación cruzada, para encontrar el mejor conjunto de hiperparámetros que minimice el error de predicción.

Por lo tanto, el uso de RMSE y MAE en la comparación de modelos de regresión es una forma efectiva de evaluar y seleccionar el modelo más adecuado para un problema específico, garantizando que las predicciones sean lo más precisas y confiables posible.

Finalmente, después de analizar los modelos haremos una reflexión sobre el sentido de la calidad de los distintos modelos respecto a la teoría de mercados eficientes.

RESULTADOS

PREPROCESADO, ANALISIS DE LOS DATOS Y ANALISIS EXPLORATORIO

Para la obtención de los datos se realiza una búsqueda en múltiples fuentes de información. Los datos son anuales empezando por el año 2000 (por la disponibilidad de los datos). Los datos se han obtenido con una frecuencia diaria (días hábiles). Estos datos son procesados, eliminando NAs, formateando las distintas frecuencias, y otros pequeños ajustes.

La siguiente imagen muestra los datos del dataframe procesado.

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 5786 entries, 2000-01-03 to 2022-12-29
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   wti_price           5786 non-null   float64
 1   sp500_index         5786 non-null   float64
 2   sp500_change        5786 non-null   float64
 3   dollar_index        5786 non-null   float64
 4   energy_index        5786 non-null   float64
 5   copper_price        5786 non-null   float64
 6   vix_index           5786 non-null   float64
 7   inventories         5786 non-null   float64
 8   demanda             5786 non-null   float64
 9   tasa_paro           5786 non-null   float64
 10  gdp                 5786 non-null   float64
 11  vehicle_production  5786 non-null   float64
dtypes: float64(12)
memory usage: 587.6 KB
```

Ilustración 5: Información Dataset , Elaboración Propia

En la imagen anterior se puede comprobar que disponemos de más de 5000, lo que permite hacer uso de algunos de los modelos más avanzados. Por otro lado, al ser todos los datos numéricos se evitará tener que implementar variables dummies u otras técnicas para dato de tipo cualitativo.

Correlaciones y Distribución de las variables

A continuación, se realiza el análisis exploratorio obteniendo las distintas correlaciones entre las variables. Se puede observar que la mayor parte de las variables no está muy correlacionada.

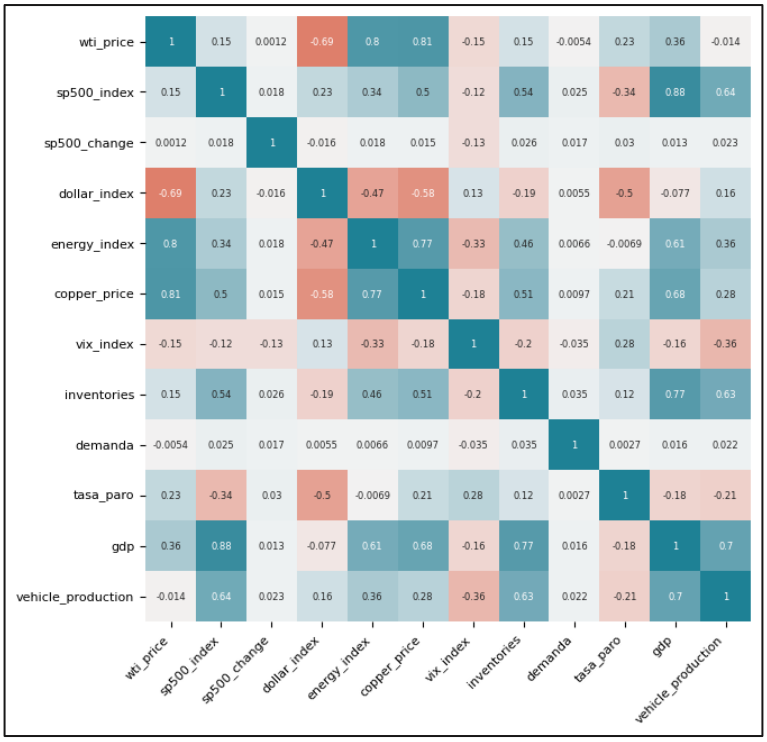

Ilustración 6: Tabla de Correlaciones, Elaboración Propia

La siguiente imagen muestra los histogramas de las distintas variables, lo cual permite observar si las distribuciones tienen alguna particularidad. Lo más destacable parece ser que la demanda, al tratarse de una variable de cambio, tiene una distribución cercana a la normal. Esto permite generar un modelo naive contra el cual será sencillo comparar los demás

modelos. Este modelo naive consiste en un modelo que siempre devuelve como salida un 0% de cambio en la demanda, y es contra el que se comparará la eficacia. de los otros modelos.

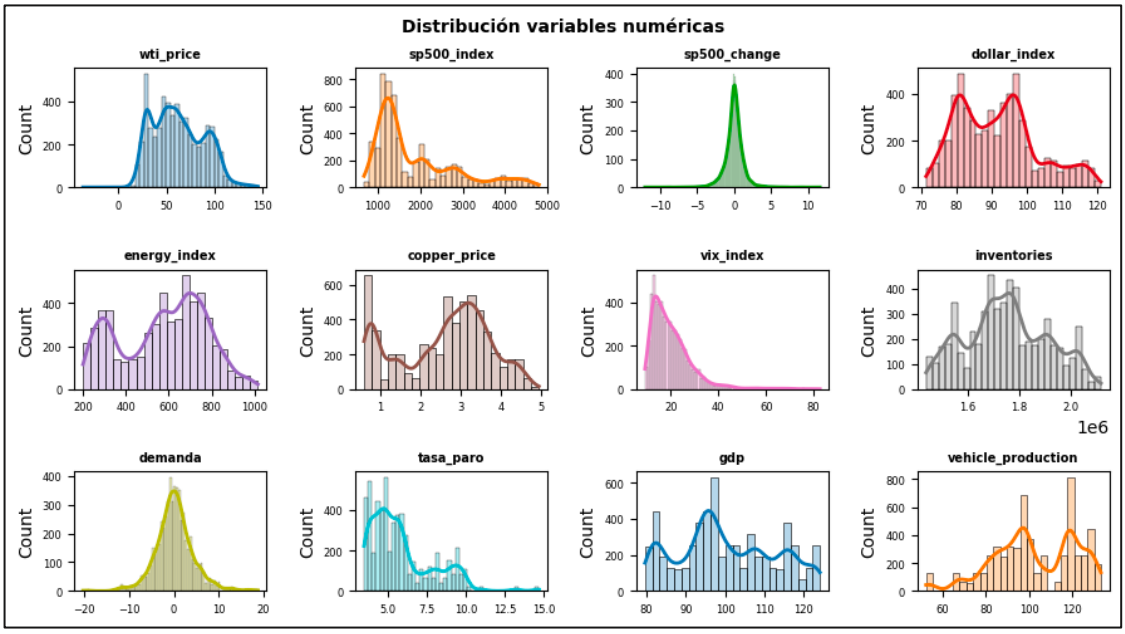

Ilustración 7: Distribución de las Variables, Elaboración Propia

A continuación, se muestra la correlación de las distintas variables con la variable dependiente (demanda). Como se puede observar, no hay ninguna correlación especialmente fuerte.

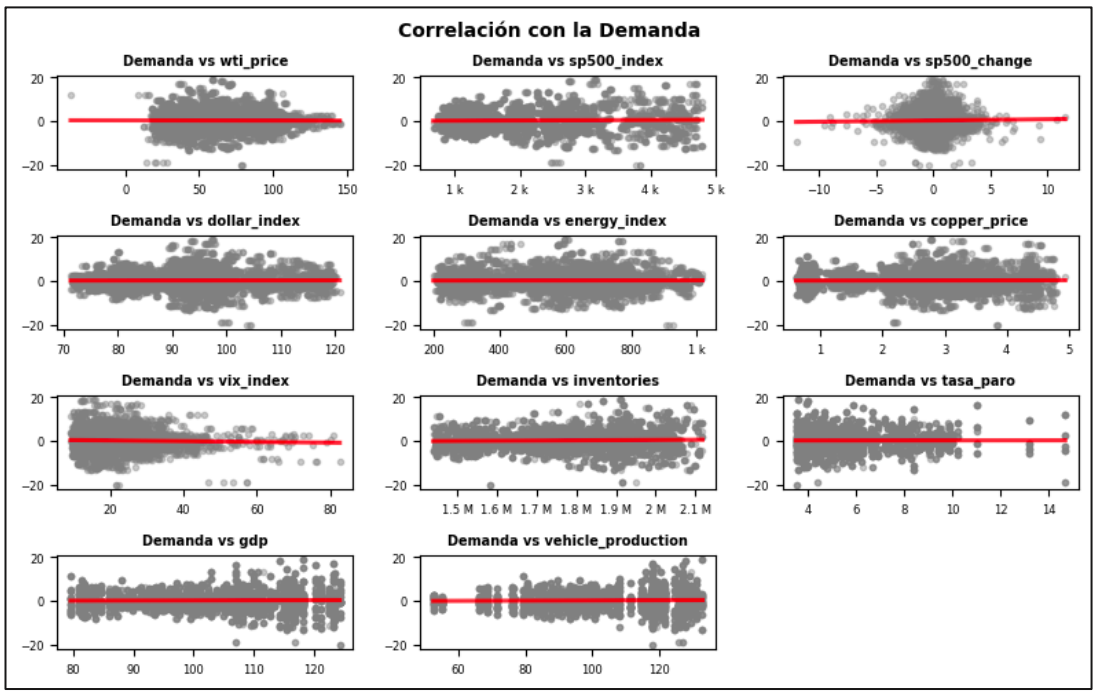

Ilustración 8: Correlación de las variables con la Demanda, Elaboración Propia

ARIMA

ARIMA (Modelo Autorregresivo Integrado de Media Móvil) es un modelo estadístico que se utiliza para analizar y pronosticar series temporales. ARIMA es un enfoque lineal que combina tres componentes principales: componentes autorregresivos (AR), componentes de media móvil (MA) e integración (I).

- Componente AR: Este componente captura la dependencia entre una observación y un número específico de observaciones anteriores (retardos).
- Componente MA: Este componente captura la dependencia entre una observación y un error de pronóstico en un número específico de observaciones anteriores.
- Componente I: La integración se refiere a la diferenciación de la serie temporal para hacerla estacionaria, es decir, para eliminar tendencias y estacionalidades.

Usar ARIMA es una buena opción para analizar y pronosticar la demanda en una serie temporal debido a las siguientes razones:

- Estacionariedad: ARIMA es adecuado para series temporales que pueden transformarse en estacionarias mediante diferenciación. La estacionariedad es importante porque permite que los modelos capten mejor las relaciones temporales en los datos.

- Estructura de dependencia temporal: ARIMA captura la estructura de dependencia temporal en la serie utilizando componentes autorregresivos y de media móvil. Esto permite que el modelo haga pronósticos basados en las correlaciones temporales en la serie.

- Interpretación de los parámetros: Los parámetros del modelo ARIMA tienen interpretaciones claras y pueden ayudar a comprender los factores subyacentes que influyen en la serie temporal de la demanda. Por ejemplo, los coeficientes de autorregresión pueden indicar cómo las observaciones pasadas afectan las observaciones futuras, mientras que los coeficientes de media móvil pueden revelar cómo los errores de pronóstico anteriores influyen en las observaciones futuras.

- Modelado de tendencias y estacionalidades: ARIMA puede extenderse a SARIMA (Modelo Autorregresivo Integrado de Media Móvil Estacional), que incluye componentes estacionales adicionales para modelar patrones estacionales en los datos. Esto puede ser útil para predecir la demanda en series temporales que exhiben patrones estacionales.

En resumen, ARIMA es una buena opción para analizar y predecir la demanda en series temporales, ya que puede capturar la estructura de dependencia temporal, modelar tendencias y estacionalidades, y proporcionar interpretaciones claras de los parámetros del modelo. Sin embargo, es importante tener en cuenta que ARIMA es un modelo lineal y puede no ser adecuado para series temporales con relaciones no lineales o de alta complejidad. En tales casos, se podrían considerar modelos más avanzados, como las redes neuronales recurrentes (RNN) o los modelos de aprendizaje profundo, como las redes LSTM.

Los resultados obtenidos usando ARIMA son los siguientes:

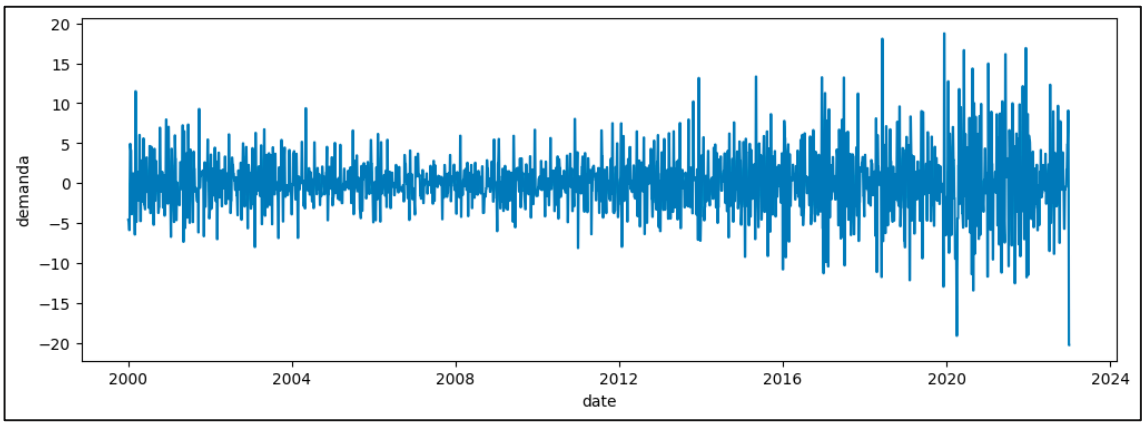

Ilustración 9: Demanda (cambio) de crudo en USA, Elaboración Propia

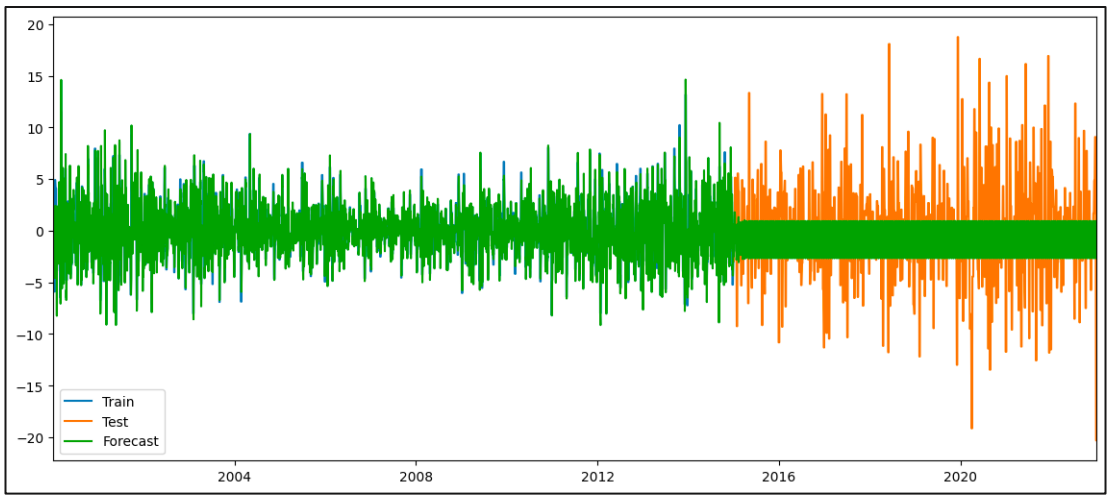

Ilustración 10: ARIMA predicción para el test y el train, Elaboración Propia

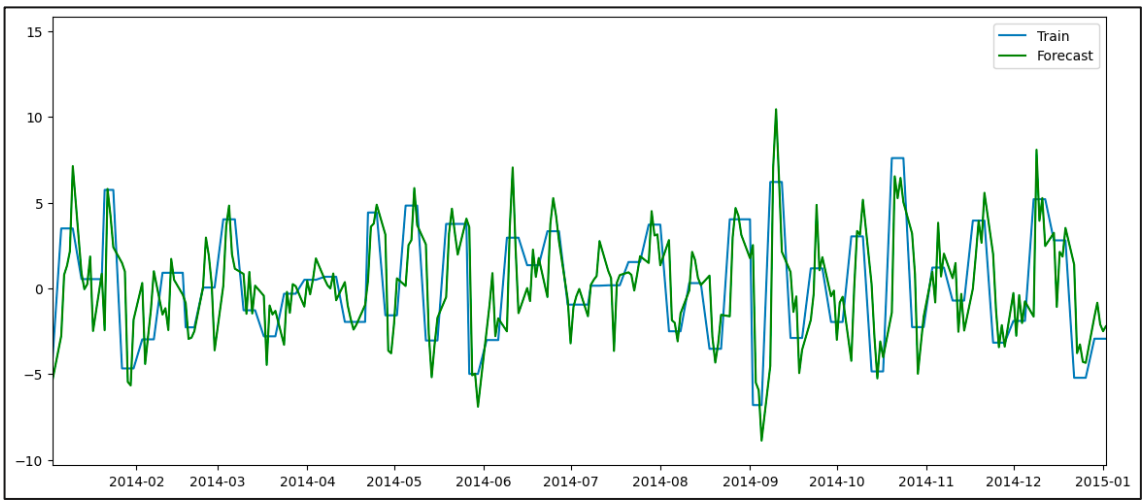

Ilustración 11: ARIMA predicción para el train, Elaboración Propia

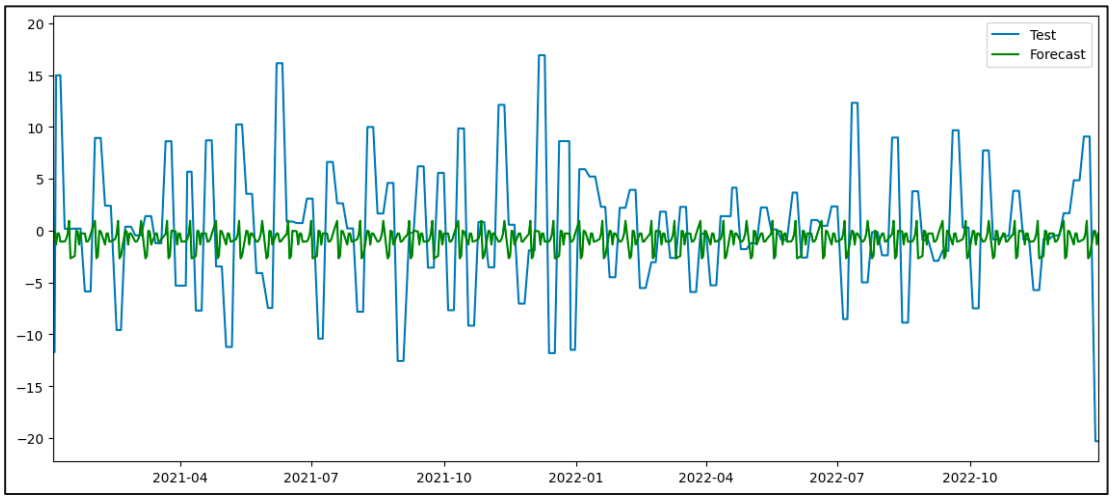

Ilustración 12: ARIMA Predicción en el test, Elaboración Propia

La capacidad explicativa del ARIMA es muy alta, pero por otro lado es muy pobre para la parte de predicción con el test. Existen varias razones por las que un pronóstico ARIMA puede no ser muy preciso en el conjunto de datos de prueba. Algunas de las razones más comunes incluyen:

1. **Estacionariedad:** El modelo ARIMA requiere que la serie temporal sea estacionaria. Si no se ha logrado la estacionariedad después de la diferenciación, el modelo no podrá capturar adecuadamente las relaciones temporales en los datos. Hay que asegurarse de realizar pruebas de estacionariedad, como la prueba Dickey-Fuller aumentada, antes de ajustar el modelo.

2. **Selección de parámetros incorrecta:** Si los valores de los parámetros p, d y q no se seleccionan correctamente, el modelo ARIMA no podrá capturar las dependencias temporales en la serie de tiempo. Puedes utilizar la función de autocorrelación (ACF) y la función de autocorrelación parcial (PACF), así como criterios de información como el Criterio de Información de Akaike (AIC) y el Criterio de Información Bayesiano (BIC), para seleccionar los valores óptimos de los parámetros.

3. **Estacionalidad no modelada:** Si la serie de tiempo tiene patrones estacionales y no se ha utilizado un modelo SARIMA para capturar la estacionalidad, el modelo ARIMA podría no ser preciso en el pronóstico. Considera ajustar un modelo SARIMA en lugar de un ARIMA si hay estacionalidad en los datos.

4. **Cambios estructurales no capturados:** Si hay cambios estructurales en la serie temporal, como cambios en la política económica o en la demanda del mercado, el modelo ARIMA podría no ser capaz de capturar estos cambios y, en consecuencia, generar pronósticos imprecisos.

5. **Relaciones no lineales:** ARIMA es un modelo lineal, por lo que no es adecuado para series temporales con relaciones no lineales o de alta complejidad. En tales casos, se podrían considerar modelos más avanzados, como las redes neuronales recurrentes (RNN) o los modelos de aprendizaje profundo, como las redes LSTM.

6. **Errores en los datos**: Si hay errores en los datos, como valores atípicos o datos faltantes, estos pueden afectar la precisión del pronóstico. Asegúrate de tratar estos problemas antes de ajustar el modelo.

Para mejorar la precisión del pronóstico, es fundamental investigar y abordar estos problemas. Además, también se puede considerar probar otros modelos de series temporales (modelos de estado-espacio, modelos estructurales o enfoques de aprendizaje automático como las redes neuronales o los bosques aleatorios). La combinación de múltiples modelos también puede mejorar la precisión del pronóstico en algunos casos.

Analizando los resultados, se ha llegado a una hipótesis que explicaría las causas de la baja precisión del ARIMA. Por un lado, la baja calidad de los datos que son obtenidos de datos con frecuencias muy inferiores a diarias y por lo tanto nos falta información. Por otro lado,

una falta de linealidad entre los datos. Estas hipótesis probablemente explique por qué el modelo decide que la predicción más eficaz es una muy cercana al modelo naive, y es que por la teoría de mercado eficiente que comentaremos en las conclusiones es esperable que no se pueda predecir de forma ajustada los cambios.

ANALISIS EXPLICATIVO VS PREDICTIVO

El análisis explicativo se diferencia del predictivo en que el análisis explicativo se centra en comprender la relación entre las variables y en identificar los factores que influyen en la variable objetivo, mientras que el análisis predictivo se centra en hacer predicciones futuras basadas en datos históricos.

1. Análisis explicativo:

   El objetivo principal del análisis explicativo es identificar y comprender las relaciones entre las variables y determinar cómo las variables independientes afectan a la variable dependiente. En este tipo de análisis, el objetivo es probar hipótesis y en establecer relaciones de causa y efecto. Por lo general, el análisis explicativo utiliza modelos estadísticos y econométricos para cuantificar el efecto de las variables independientes en la variable dependiente.

   El análisis explicativo es útil para:

   - Identificar las variables clave que influyen en la variable objetivo.
   - Comprender la dirección y la magnitud de las relaciones entre las variables.
   - Probar hipótesis y teorías basadas en datos.
   - Establecer relaciones de causa y efecto. No es el tema de este TFG.
   - Informar la toma de decisiones y las políticas basadas en la comprensión de las relaciones entre las variables.

2. Análisis predictivo:

   El análisis predictivo utiliza algoritmos de aprendizaje automático y estadísticas para predecir resultados futuros basándose en datos históricos. En lugar de centrarse en entender las relaciones entre las variables, el análisis predictivo se enfoca en hacer predicciones lo más precisas posible. El análisis predictivo puede utilizar una variedad de técnicas, desde la regresión lineal hasta los modelos de aprendizaje profundo, dependiendo de la naturaleza de los datos y la complejidad del problema.

   El análisis predictivo es útil para:

   - Predecir eventos futuros basándose en datos históricos.
   - Identificar patrones y tendencias en los datos.
   - Ayudar en la toma de decisiones basadas en predicciones.
   - Optimizar la eficiencia operativa y reducir costos.
   - Mejorar la precisión y la eficacia de las estrategias de marketing y ventas.

En resumen, el análisis explicativo se centra en entender las relaciones entre las variables y establecer relaciones de causa y efecto, mientras que el análisis predictivo se enfoca en hacer predicciones futuras basadas en datos históricos. Ambos enfoques son valiosos en diferentes contextos y pueden complementarse entre sí en la práctica.

ANALISIS EXPLICATIVO

Se ha empezado con un modelo OLS, un modelo de Mínimos Cuadrados Ordinarios (OLS, por sus siglas en inglés) es un enfoque de regresión lineal que busca estimar la relación lineal entre una variable dependiente y una o más variables independientes. OLS minimiza la suma de las diferencias cuadradas entre los valores observados y los valores predichos por el

modelo. OLS es un método ampliamente utilizado en análisis de regresión debido a su simplicidad, facilidad de interpretación y eficiencia computacional. Sin embargo, es importante tener en cuenta que OLS tiene algunas suposiciones clave, como la linealidad, la normalidad de los errores, la homocedasticidad y la independencia de los errores. Si estas suposiciones no se cumplen, el modelo OLS puede no ser el más apropiado y podría ser necesario considerar otros enfoques de modelado.

A continuación, se pueden observar los resultados obtenidos:

Tabla 4: OLS Modelo Explicativo Resultados, Elaboración Propia

OLS Regression Results

Dep. Variable:	demanda	R-squared:	0.005
Model:	OLS	Adj. R-squared:	0.003
Method:	Least Squares	F-statistic:	2.647
Date:	Thu, 20 Apr 2023	Prob (F-statistic):	0.00222
Time:	09:38:40	Log-Likelihood:	-16532.
No. Observations:	5786	AIC:	3.309e+04
Df Residuals:	5774	BIC:	3.317e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.6641	1.962	0.848	0.396	-2.182	5.510
wti_price	0.0039	0.007	0.543	0.587	-0.010	0.018
sp500_index	0.0008	0.000	3.196	0.001	0.000	0.001
sp500_change	0.0350	0.045	0.777	0.437	-0.053	0.123
dollar_index	-0.0034	0.011	-0.321	0.748	-0.024	0.017
energy_index	0.0006	0.001	0.808	0.419	-0.001	0.002
copper_price	-0.1190	0.146	-0.815	0.415	-0.405	0.167
vix_index	-0.0110	0.009	-1.232	0.218	-0.029	0.007
inventories	2.915e-06	9.05e-07	3.222	0.001	1.14e-06	4.69e-06
tasa_paro	0.0171	0.047	0.365	0.715	-0.075	0.109
gdp	-0.0771	0.025	-3.109	0.002	-0.126	-0.028
vehicle_production	-0.0016	0.005	-0.291	0.771	-0.012	0.009

Omnibus:	299.507	Durbin-Watson:	0.596
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1003.639
Skew:	0.168	Prob(JB):	1.16e-218
Kurtosis:	5.013	Cond. No.	6.23e+07

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified
[2] The condition number is large, 6.23e+07. This might indicate that there are
strong multicollinearity or other numerical problems.

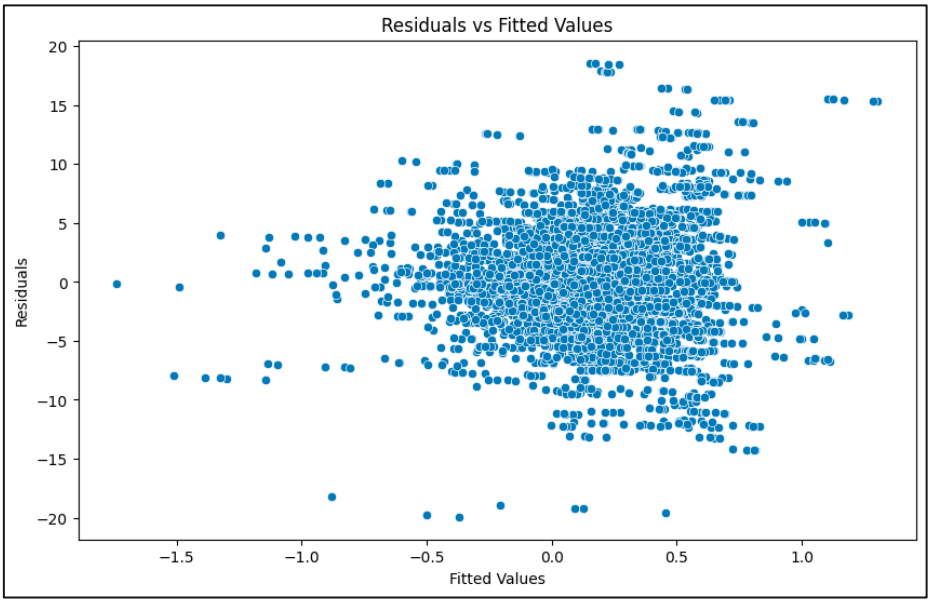

Ilustración 13: OLS Modelo Explicativo Residuales vs Valores, Elaboración Propia

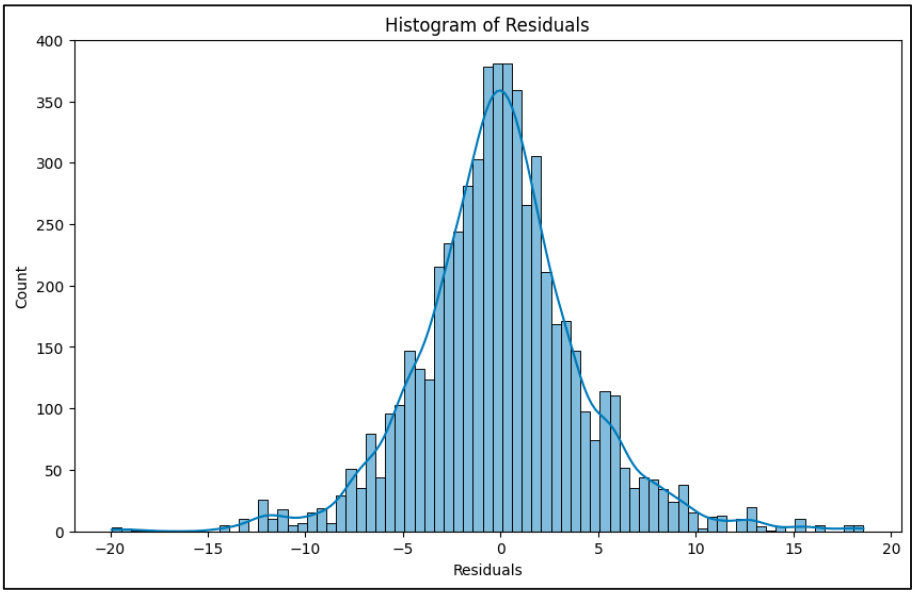

Ilustración 14: OLS Modelo Explicativo Histograma Residuales, Elaboración Propia

A la vista de los resultados cabe destacar el bajísimo nivel explicativo de las variables y los altísimos valores de residuales que tenemos.

Las conclusiones son que los únicos valores significativos por su aceptable p-valor son: sp500_index, inventarios y gdp. El aporte marginal, o valor de las betas, es prácticamente cero en el caso de los inventarios. En el caso del índice del sp500, vemos que un aumento de 1000 puntos (25% por encima del valor actual de 4000) explicaría un aumento en la demanda de 0.8%. Por último, el coeficiente del GDP destaca que es negativo es decir que parece que un aumento en el producto interior bruto tiene relación inversa con el aumento de la demanda, lo que es contra intuitivo.

ANALISIS PREDICTIVO

Para el análisis predictivo se han utilizado los siguientes modelos. A continuación, se muestran los resultados individuales de cada modelo para posteriormente compararlos y redactar las conclusiones de este estudio. Como nota, para cada modelo han sido adaptados los datos, teniendo en cuenta que son series temporales y que en algunos casos hay que normalizar. Además todos los modelos han sido optimizados encontrando los mejores hiperparametros con técnicas como RandomizedSearchCV().

RandomizedSearchCV() es una función de la biblioteca scikit-learn utilizada para optimizar hiperparametros de un modelo de aprendizaje automático. A diferencia de la búsqueda exhaustiva en la rejilla proporcionada por GridSearchCV(), RandomizedSearchCV() selecciona aleatoriamente combinaciones de hiperparametros dentro de las distribuciones especificadas, lo que permite una búsqueda más eficiente y rápida en el espacio de hiperparametros.

- **Naive**
- **Regresión Lineal**
- **Regresión con Ridge**
- **SVR**
- **Neural Network**
- **Random Forest**
- **KNN**

Regresión Lineal

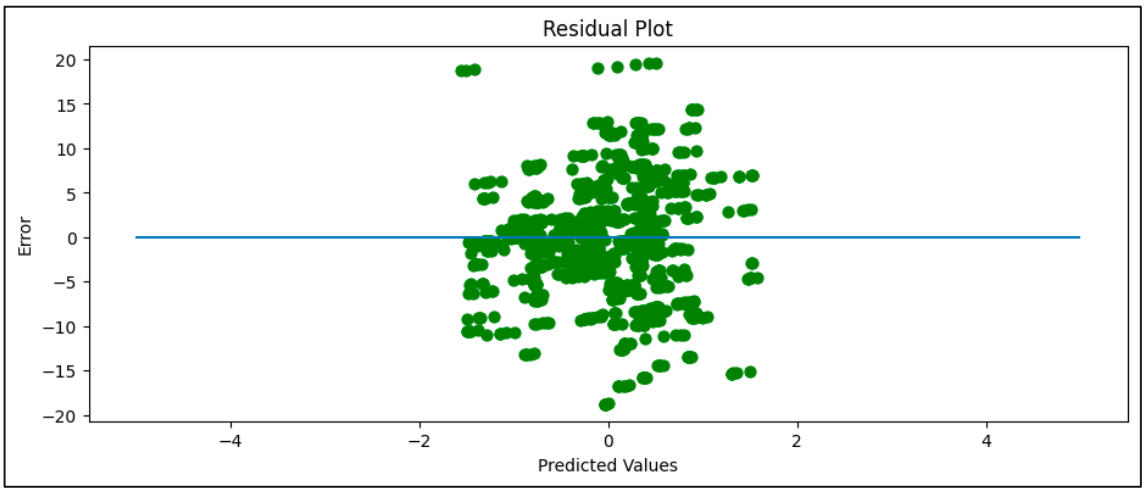

Ilustración 15: Regresión Linear Errores, Elaboración Propia

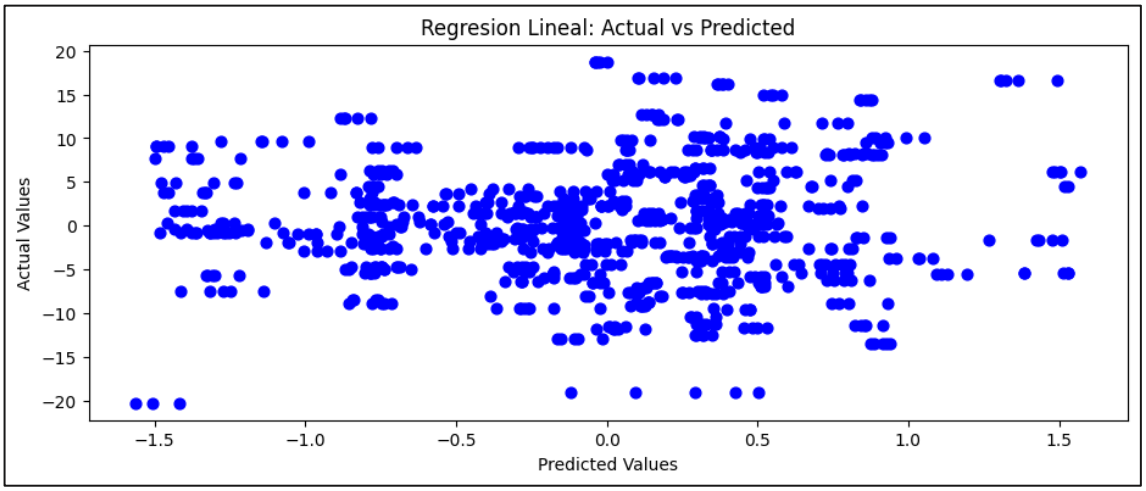

Ilustración 16: Regresión Linear Real vs Predicción, Elaboración Propia

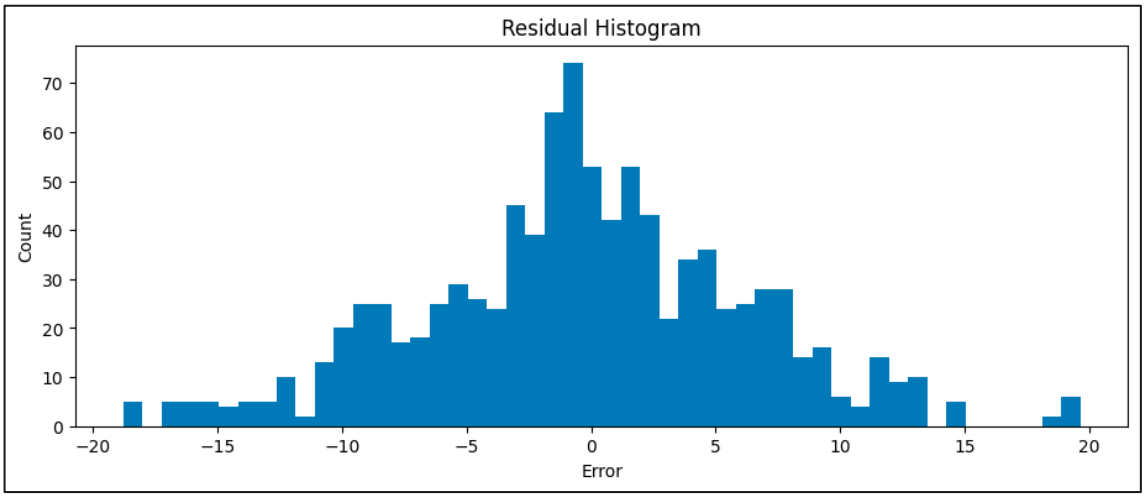

Ilustración 17: Regresión Lineal Histograma Residuales, Elaboración Propia

Regresión Lineal Con Regularización Ridge

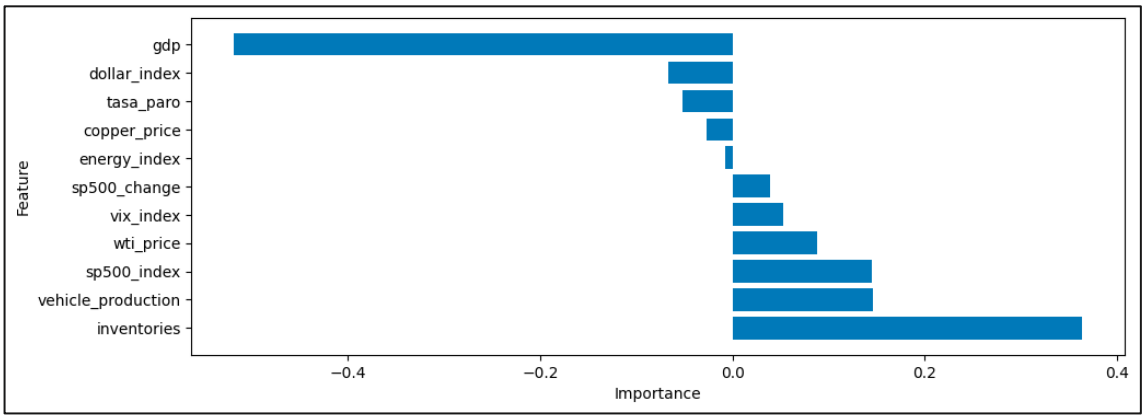

Ilustración 18: Regresión Ridge Importancia Variables, Elaboración Propia

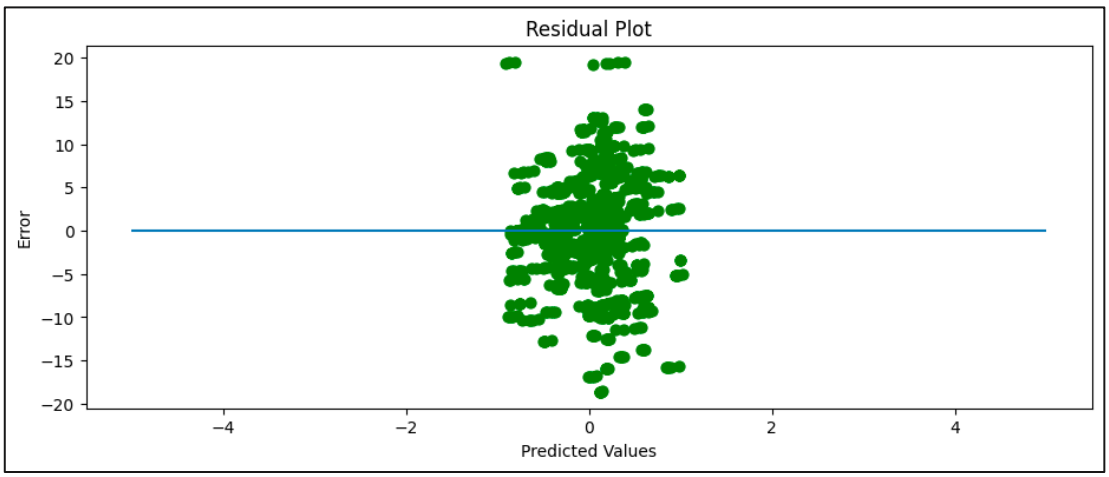

Ilustración 19: Regresión Ridge Errores, Elaboración Propia

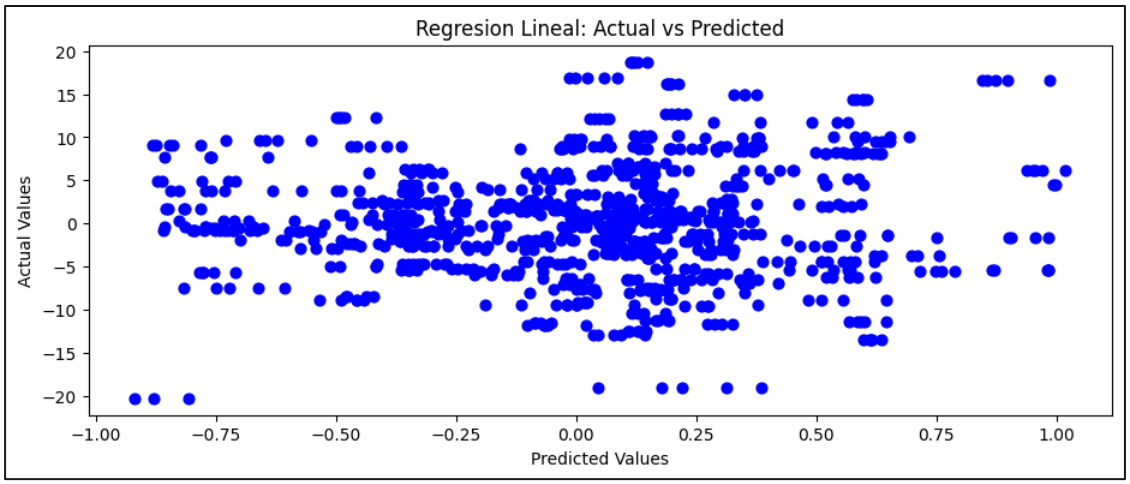

Ilustración 20: Regresión Ridge Real vs Predicciones, Elaboración Propia

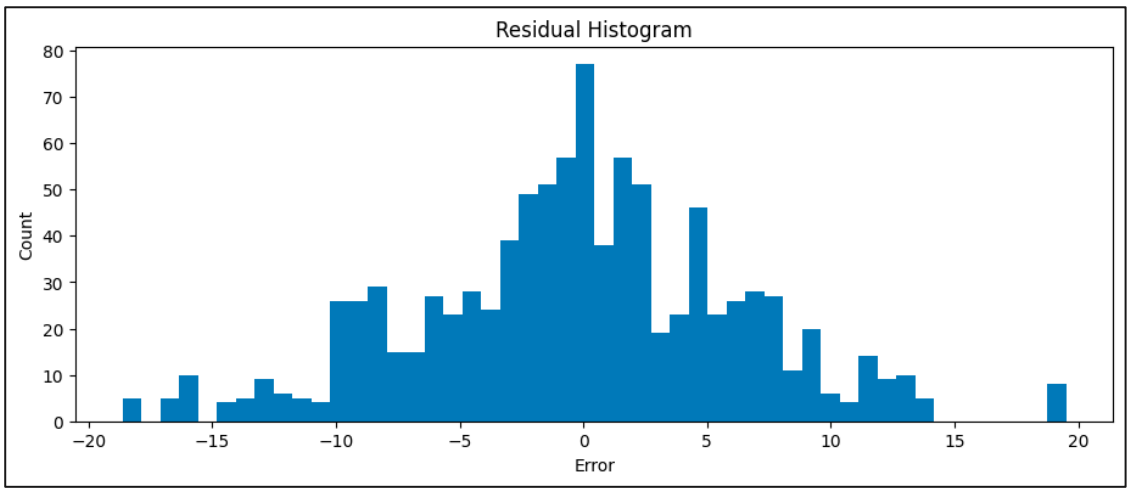

Ilustración 21: Regresión Ridge Histograma Residuales, Elaboración Propia

SVR (Support Vector Machine Regression)

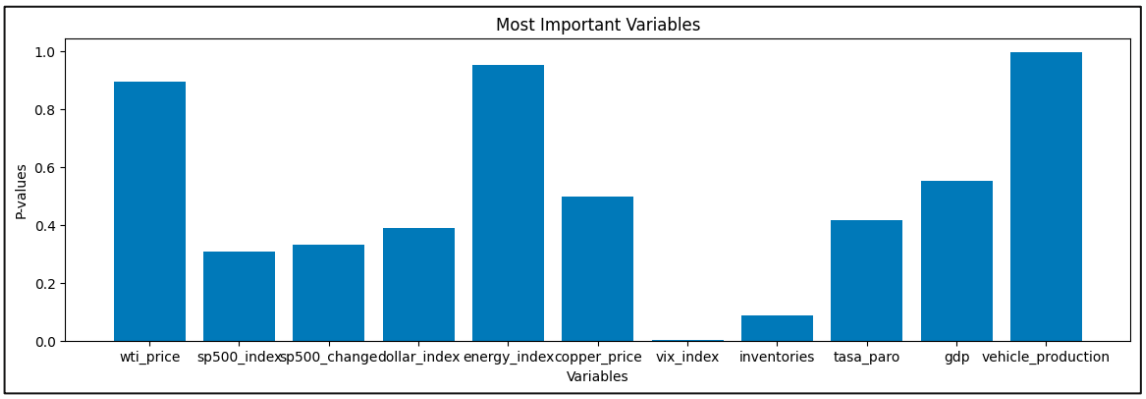

Ilustración 22: SVR Importancia Variables, Elaboración Propia

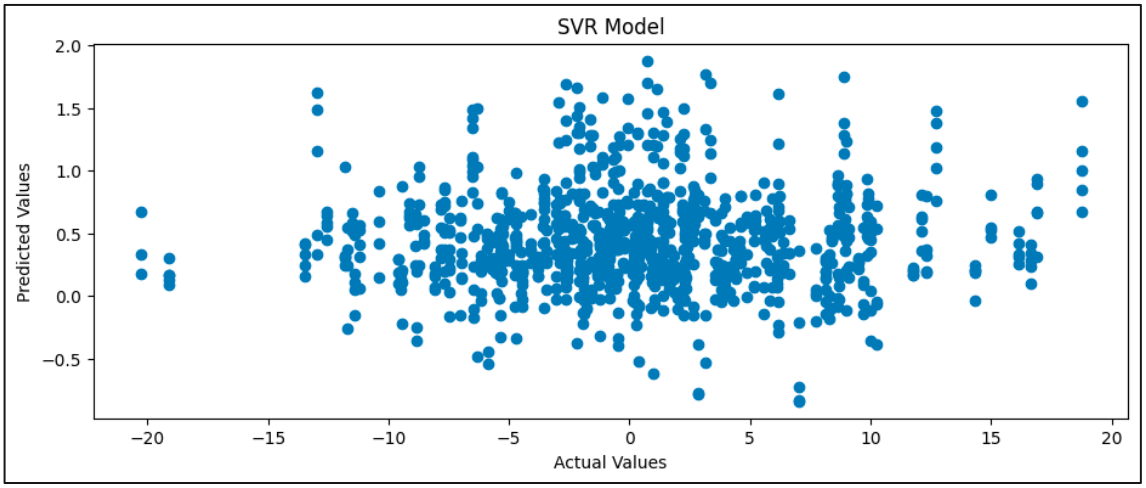

Ilustración 23: SVR Real vs Predicciones, Elaboración Propia

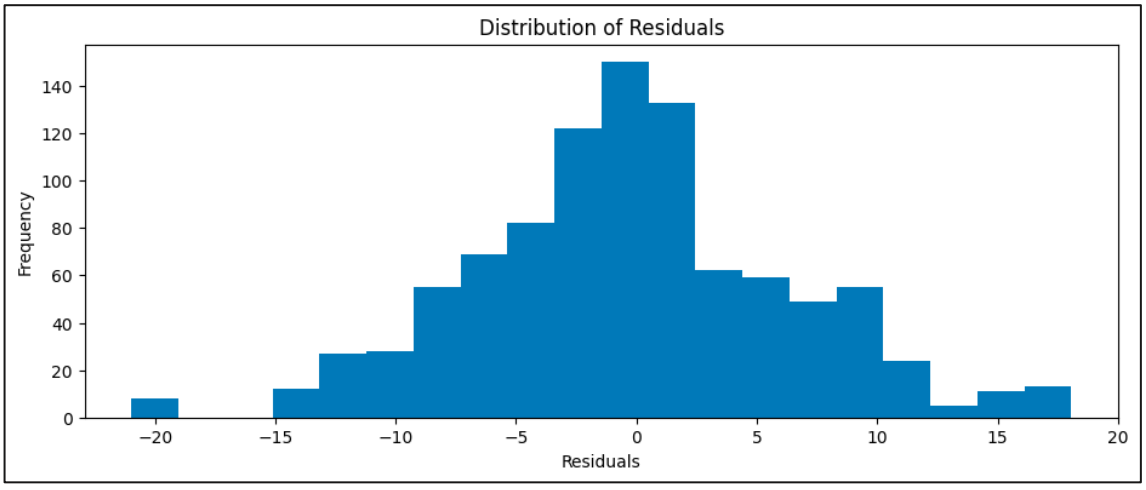

Ilustración 24: SVR Histograma Residuales, Elaboración Propia

Red Neuronal

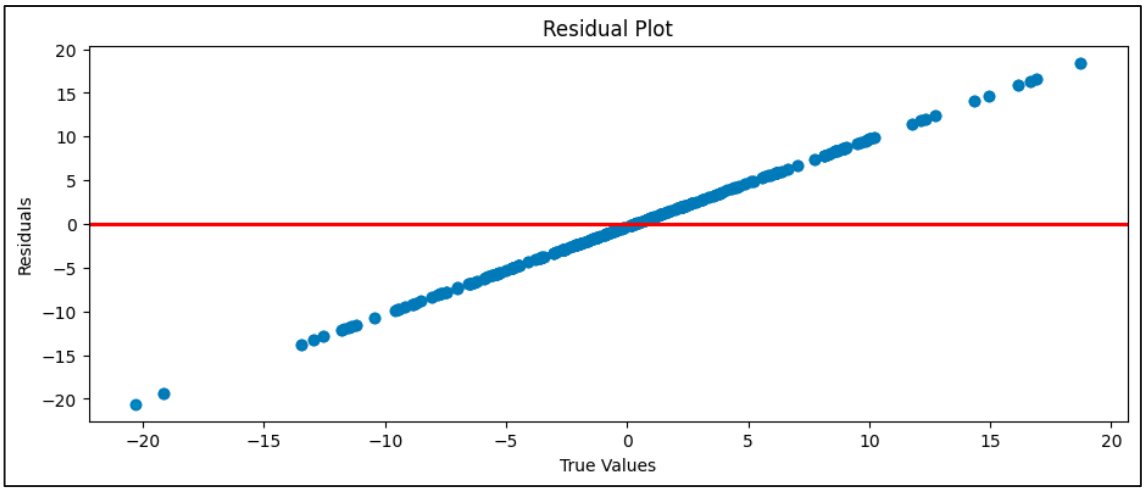

Ilustración 25: Red Neuronal Residuales, Elaboración Propia

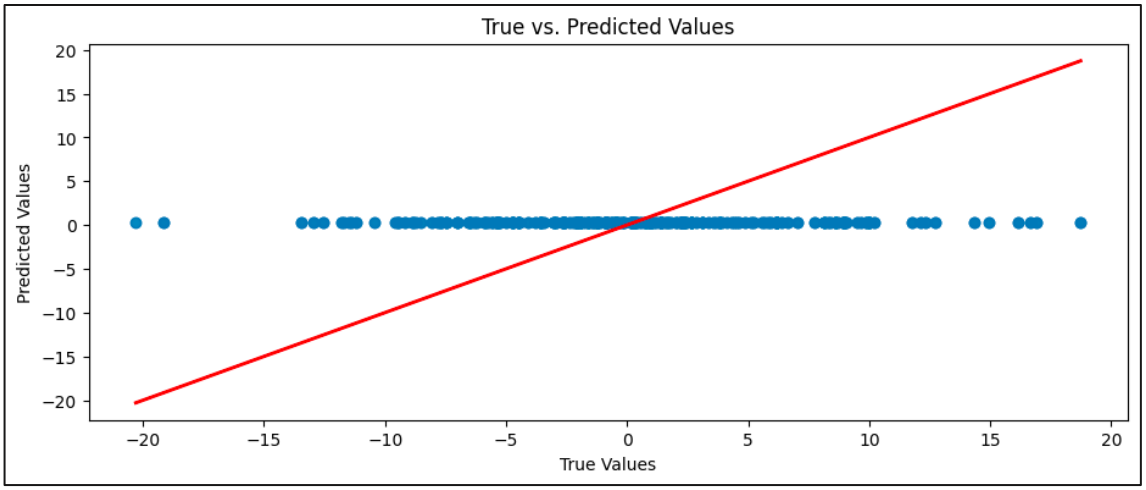

Ilustración 26: Red Neuronal Real vs Predicciones, Elaboración Propia

Random Forest

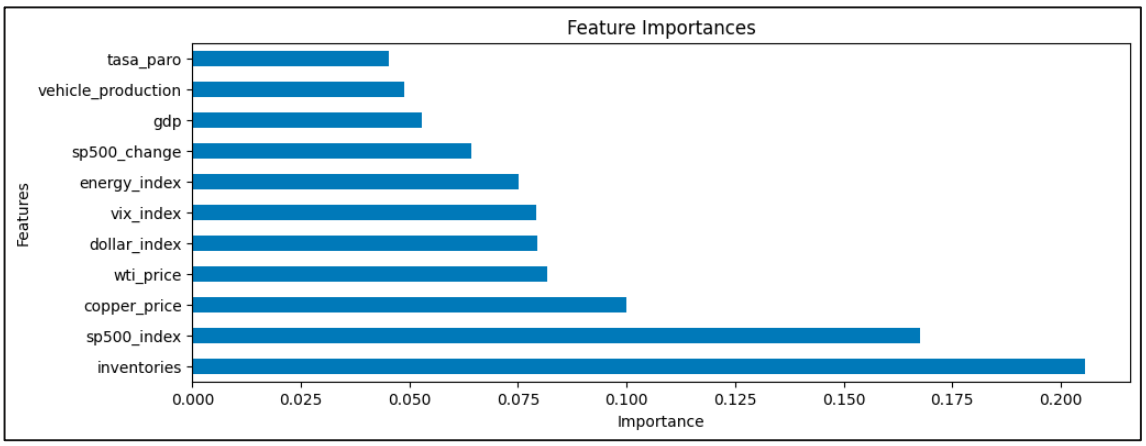

Ilustración 27: Random Forest Importancia Variables, Elaboración Propia

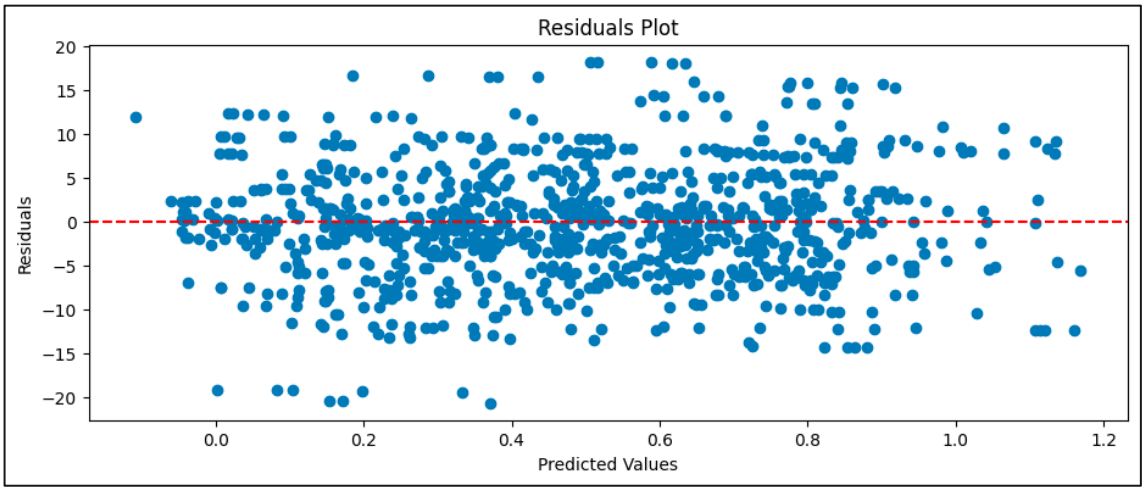

Ilustración 28: Random Forest Errores, Elaboración Propia

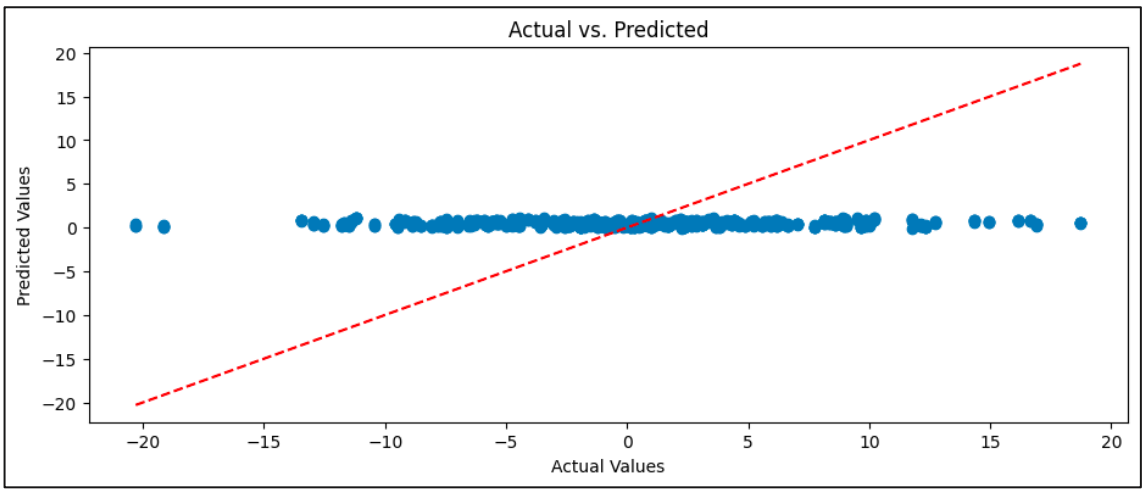

Ilustración 29: Random Forest Real vs Predicción, Elaboración Propia

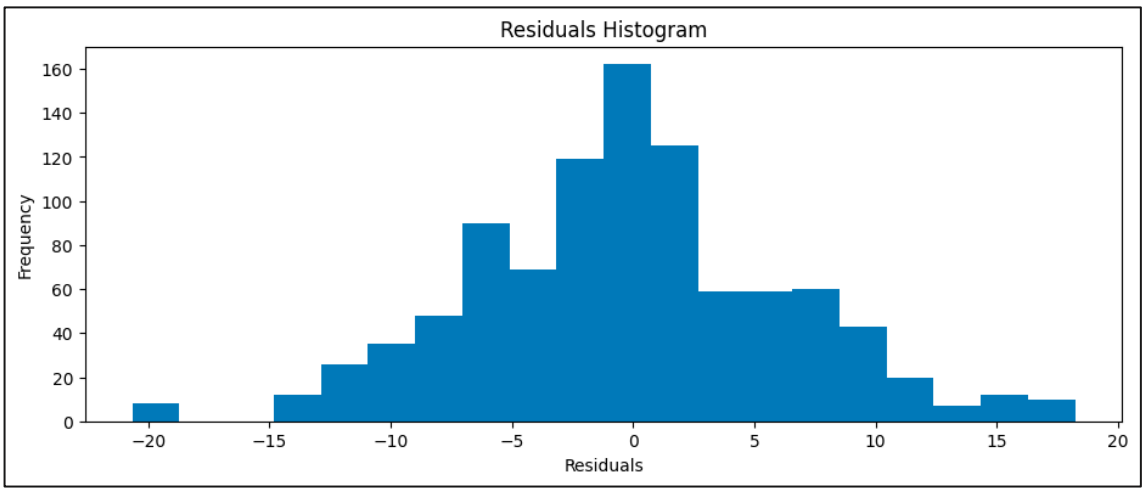

Ilustración 30: Random Forest Histograma Residuales, Elaboración Propia

Modelo Naive

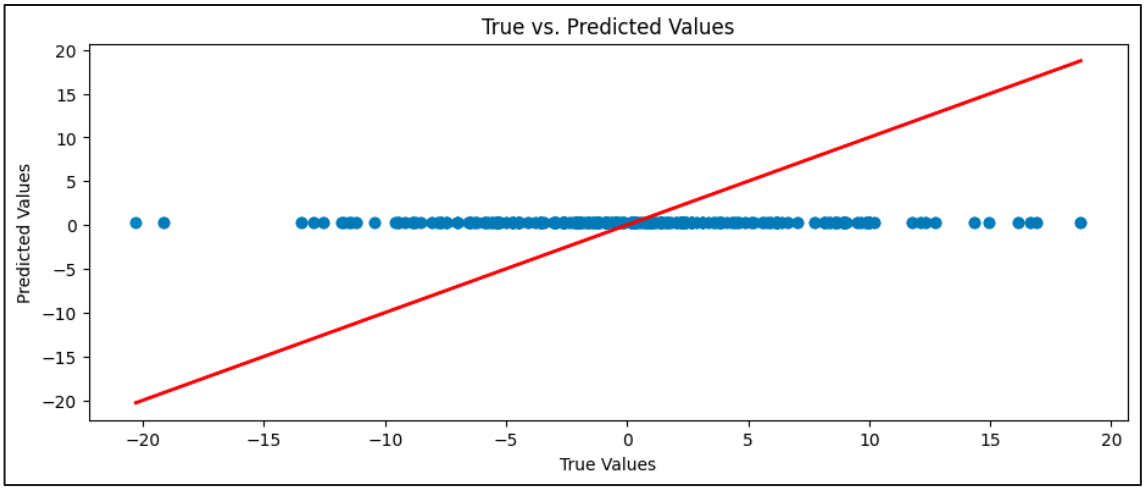

Ilustración 31: Naive Real vs Predicciones, Elaboración Propia

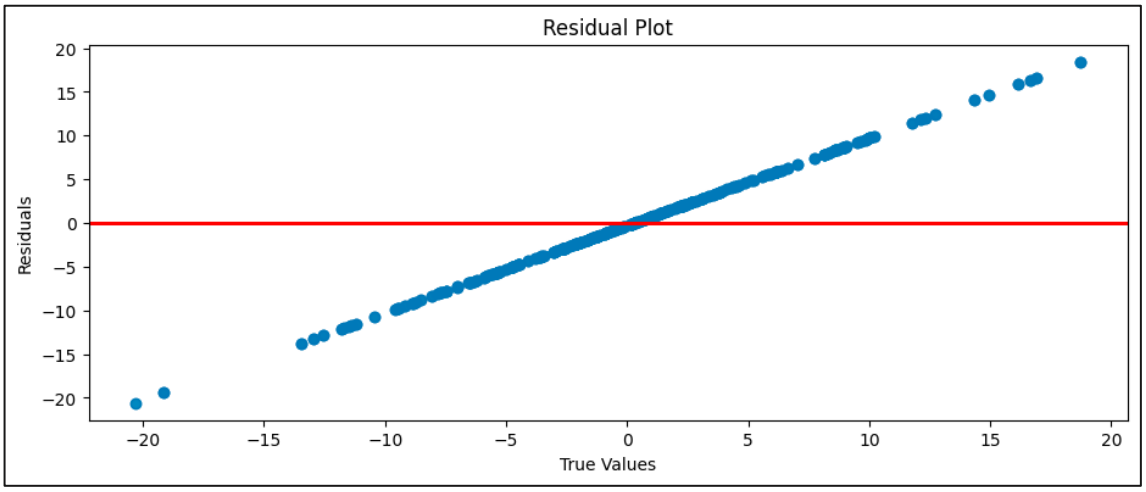

Ilustración 32: Naive Errores, Elaboración Propia

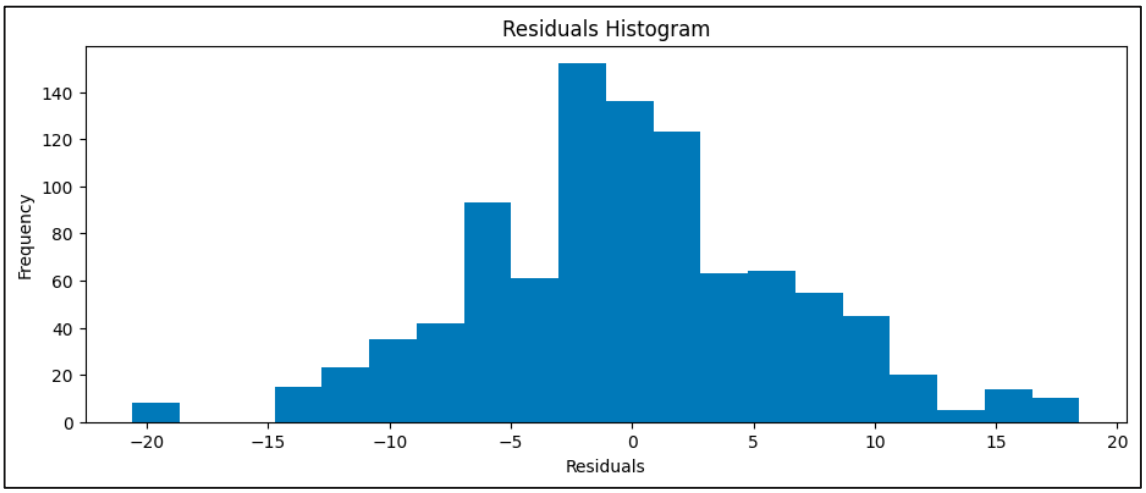

Ilustración 33: Naive Histograma Residuales, Elaboración Propia

Modelo KNN

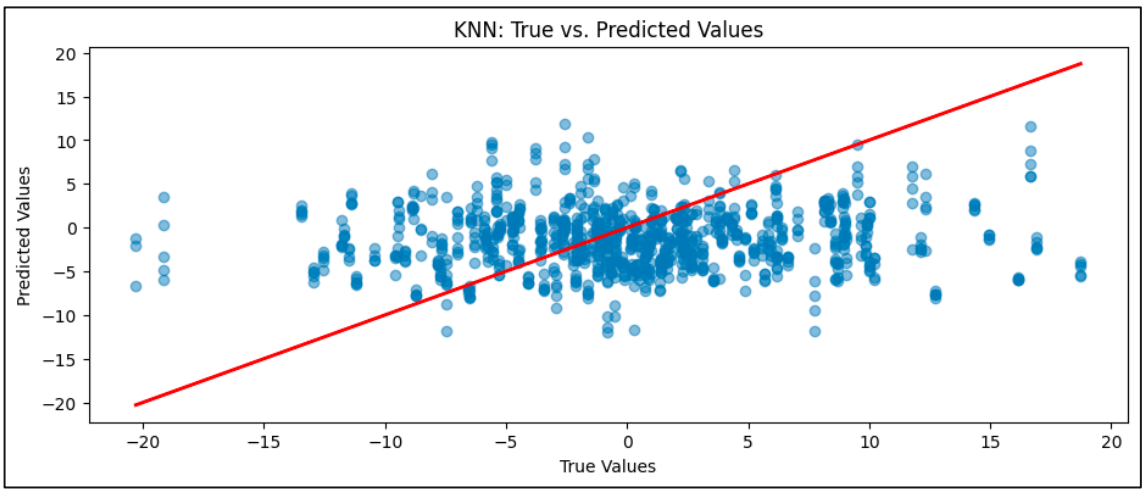

Ilustración 34: KNN Real vs Predicciones, Elaboración Propia

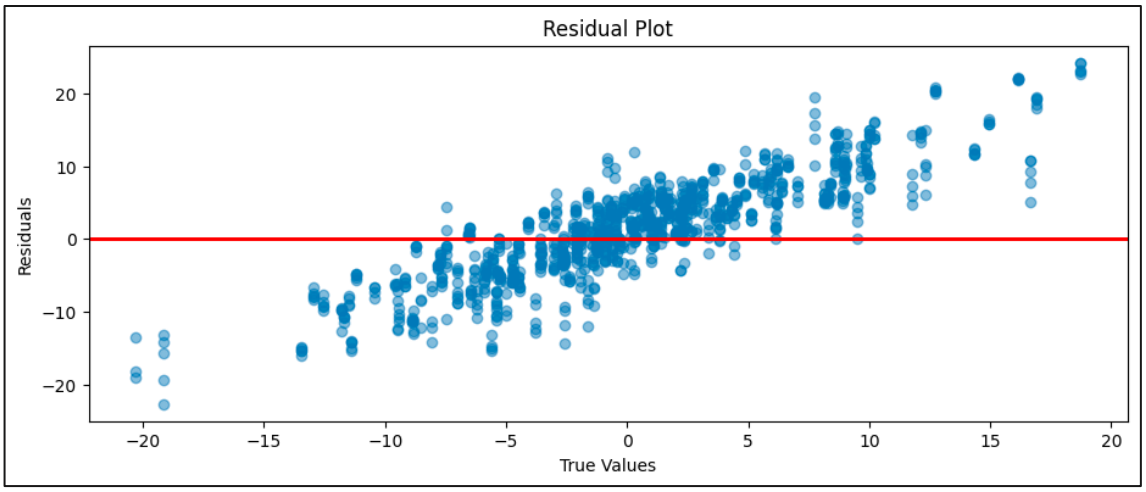

Ilustración 35: KNN Errores, Elaboración Propia

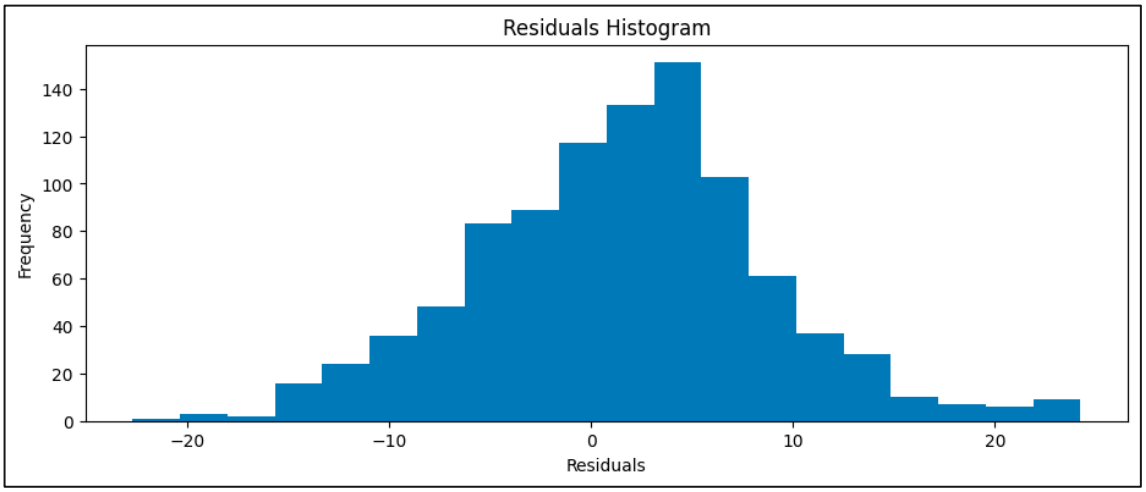

Ilustración 36: KNN Histograma Residuales, Elaboración Propia

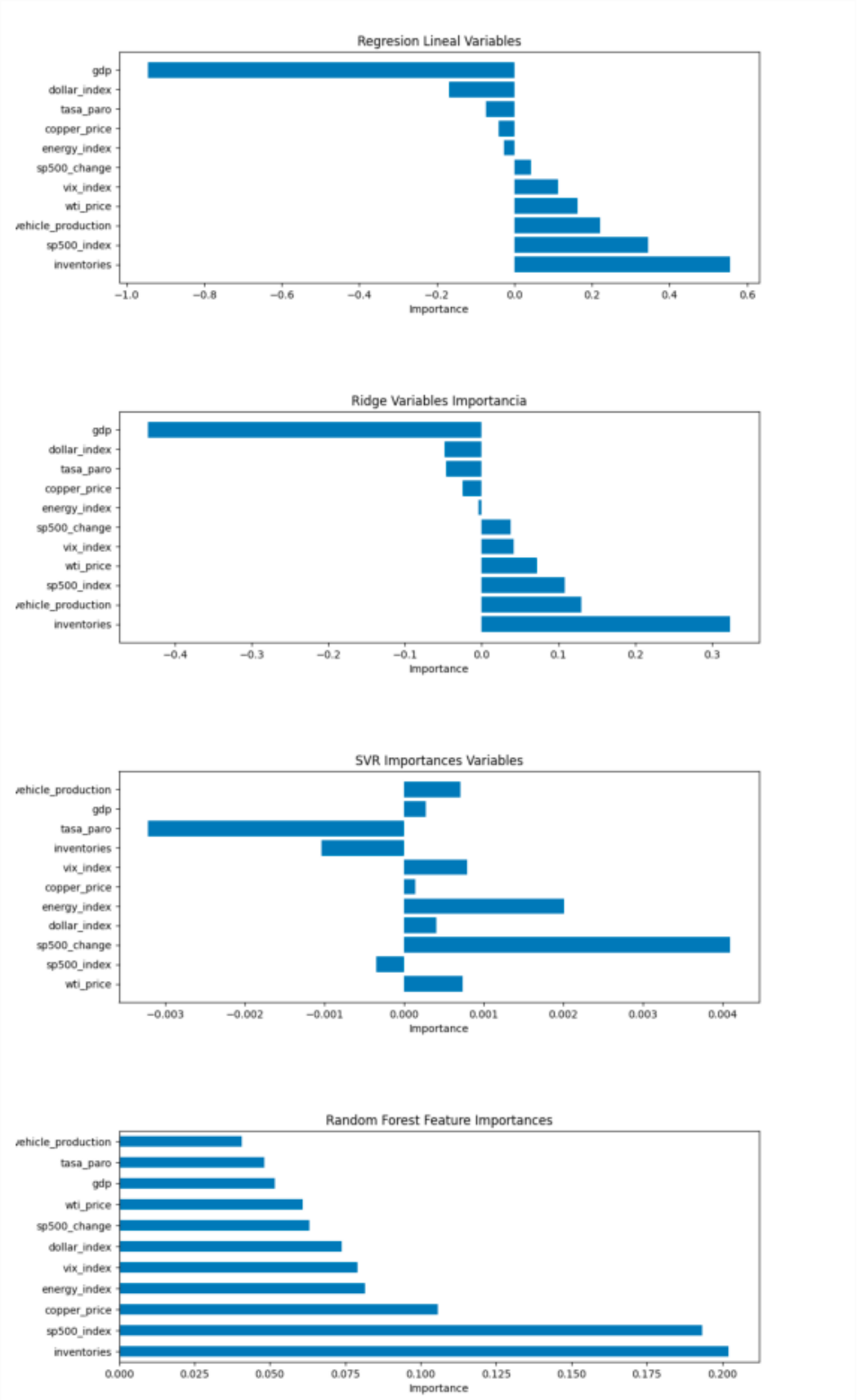

Ilustración 37: Variables Importancia

CONCLUSIONES

El objetivo principal de este proyecto es la creación de un modelo de predicción de la demanda del petróleo en Estados Unidos a nivel diario y el posterior análisis de la eficacia de este. Este objetivo incluye encontrar varios modelos distintos y compara su eficacia. Como se puede comprobar, se ha completado este objetivo con éxito aun considerando que los modelos no son buenos prediciendo.

En términos de importancia de variables, se comprueba que no hay demasiado consenso entre los distintos modelos. Aunque existen dos variables que sobresalen, estás son el GDP y los inventarios. Tiene sentido, especialmente los inventarios, ya que si se reducen será probablemente por un aumento de demanda (o alternativamente una bajada en la oferta).

A continuación, se muestran las capacidades de predicción con RMSE y MAE:

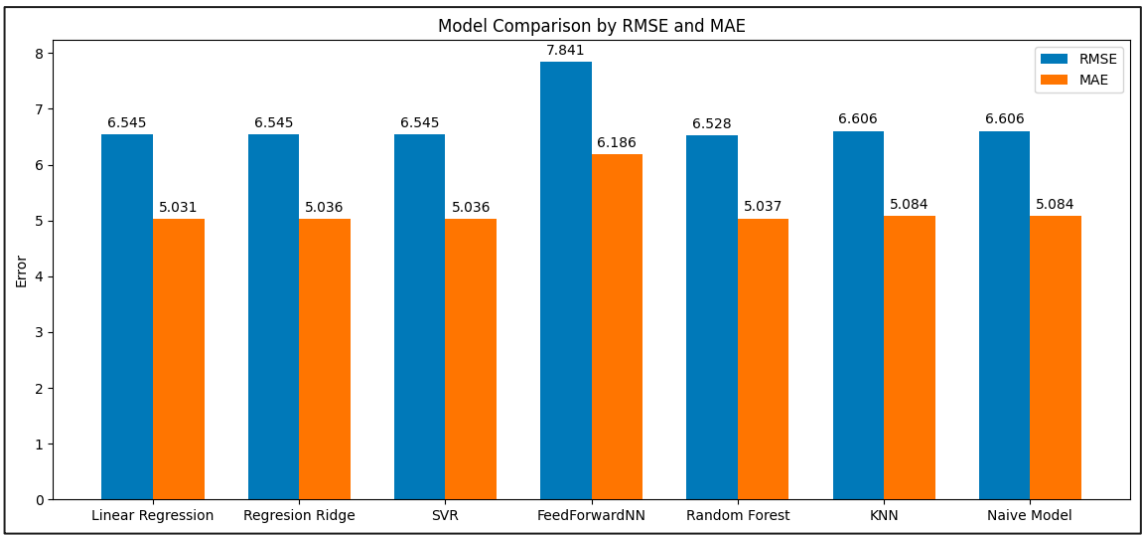

Ilustración 38: Resultados Modelos, Elaboración Propia

Como se puede observar, los resultados obtenidos con los modelos son prácticamente los mismos que los obtenidos con el modelo Naive. Este fenómeno (como hemos especulado previamente) se debe a la baja calidad de los datos, especialmente por su frecuencia. No

resulta sorprendente que a priori no sea posible predecir la demanda diaria de forma avanzada, ya que si lo fuera esto implicaría que los mercados no son eficientes y habría oportunidades de arbitraje que serían explotadas rápidamente por los inversores.

Si bien la teoría del mercado eficiente ha sido objeto de debate y crítica, es importante considerarla al abordar la predicción de variables como la demanda diaria. En un mercado eficiente, cualquier intento de predecir la demanda diaria de forma avanzada y consistente sería infructuoso, ya que la información relevante ya estaría incorporada en los precios de los activos y, por lo tanto, en las variables explicativas.

Dicho esto, es posible que se puedan hacer predicciones de la demanda diaria en un mercado semi-eficiente si se cuenta con información no pública o si se utilizan técnicas de análisis avanzadas que aún no han sido adoptadas por la mayoría de los participantes del mercado. Sin embargo, estas oportunidades podrían ser efímeras, ya que los mercados tienden a ajustarse rápidamente a medida que los inversores explotan las ineficiencias.

En resumen, aunque los modelos no han sido capaces de superar al modelo Naive en la predicción de la demanda diaria, esto podría ser consistente con la teoría del mercado eficiente. Sería interesante explorar si la incorporación de información adicional o el uso de técnicas de análisis más avanzadas podrían mejorar las predicciones, pero es importante tener en cuenta que las oportunidades de arbitraje en un mercado eficiente o semi-eficiente podrían ser limitadas y temporales.

Además, se puede observar cómo incluso las *Energy Information Agency* erra en sus predicciones semanales, y eso que tiene acceso (al ser el regulador de *Oil&Gas* en USA) a cantidades ingentes de información confidencial de las empresas. Un buen ejemplo es el reporte de inventarios semanales. En la siguiente tabla se muestran los resultados y las predicciones previas realizadas 1 semana antes de la publicación de los datos oficiales:

Tabla 5: EIA Predicción de Demanda vs Datos Reales, EIA

Release Date	Time	Actual	Forecast
Apr 05, 2023	10:30		0.092M
Mar 29, 2023	10:30	-7.489M	0.092M
Mar 22, 2023	10:30	1.117M	-1.565M
Mar 15, 2023	10:30	1.550M	1.188M
Mar 08, 2023	11:30	-1.694M	0.395M
Mar 01, 2023	11:30	1.165M	0.457M

Precisamente por esto no resulta extraño que no se hayan podido obtener mejores resultados.

El objetivo secundario consistía en determinar si había alguna variable especialmente relevante. Tras analizar la importancia de las variables en la mayoría de los modelos se puede ver que no hay mucha coincidencia entre modelos, lo cual tiene sentido ya que si el modelo naive asigna mismo peso a todas las variables y los modelos son parecidos en cuanto a capacidad predictiva, sería extraño que destacara alguna variable en todos ellos. Aun así, las que más se repiten son inventarios y nivel de producto interior bruto.

RECURSOS EMPLEADOS

Los recursos que emplear para el proyecto son principalmente:

- Python con Virtual Studio Code y las librerías correspondientes
- Internet, FactSet, Word

BIBLIOGRAFIA

Abdelsalam, M. A. M. (2020). Oil price fluctuations and economic growth: The case of MENA countries. *Review of Economics and Political Science*, *ahead-of-print*(ahead-of-print). https://doi.org/10.1108/REPS-12-2019-0162

Baumeister, C., Korobilis, D., & Lee, T. K. (2022). *Energy Markets and Global Economic Conditions*.

Board of Governors of the Federal Reserve System, Caldara, D., Cavallo, M., & Iacoviello, M. (2016). Oil Price Elasticities and Oil Price Fluctuations. *International Finance Discussion Papers*, *2016*(1173), 1–59. https://doi.org/10.17016/IFDP.2016.1173

Calomiris, C., Cakir Melek, N., & Mamaysky, H. (2020). Mining for Oil Forecasts. *The Federal Reserve Bank of Kansas City Research Working Papers*. https://doi.org/10.18651/RWP2020-20

Crude Volatility | Columbia University Press. (n.d.). Retrieved January 12, 2023, from https://cup.columbia.edu/book/crude-volatility/9780231178143

Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A., & Vapnik, V. (1996). Support Vector Regression Machines. *Advances in Neural Information Processing Systems*, *9*.

https://proceedings.neurips.cc/paper_files/paper/1996/hash/d38901788c533e8286cb6400b40b386d-Abstract.html

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, *25*(2), 383. https://doi.org/10.2307/2325486

Hamilton, J. D. (2009). Understanding Crude Oil Prices. *The Energy Journal*, *30*(2). https://doi.org/10.5547/ISSN0195-6574-EJ-Vol30-No2-9

Ho, T. K. (1995). Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, *1*, 278–282 vol.1. https://doi.org/10.1109/ICDAR.1995.598994

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kumari, K., & Yadav, S. (2018). Linear regression analysis study. *Journal of the Practice of Cardiovascular Sciences*, *4*, 33. https://doi.org/10.4103/jpcs.jpcs_8_18

Miao, H., Ramchander, S., Wang, T., & Yang, D. (2017). Influential factors in crude oil price forecasting. *Energy Economics*, *68*(C), 77–88.

Obite, C. P., Chukwu, A., Bartholomew, D. C., Nwosu, U. I., & Esiaba, G. E. (2021). Classical and machine learning modeling of crude oil production in Nigeria: Identification of an eminent model for application. *Energy Reports*, *7*, 3497–3505. https://doi.org/10.1016/j.egyr.2021.06.005

Ritchie, H., Roser, M., & Rosado, P. (2022). Energy. *Our World in Data*. https://ourworldindata.org/energy-mix

Use of oil—U.S. Energy Information Administration (EIA). (n.d.). Retrieved November 9, 2022, from https://www.eia.gov/energyexplained/oil-and-petroleum-products/use-of-oil.php

WOO 2022—Home. (n.d.). Retrieved November 9, 2022, from https://woo.opec.org/index.php

ANEXO

Código de procesamiento de datos

```
PROCESADO DE LOS DATOS
# Importación de bibliotecas

# Manipulación de datos
import pandas as pd
import numpy as np

# Visualización de datos
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import style
import matplotlib.ticker as ticker

# Modelos estadísticos
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.arima.model import ARIMA
from pmdarima import auto_arima

# Modelos de aprendizaje automático
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPClassifier

# Métricas y evaluación de modelos
from sklearn.metrics import (
    r2_score, confusion_matrix, roc_curve, precision_recall_curve,
    accuracy_score, precision_score, recall_score,
    f1_score, roc_auc_score, mean_squared_error, auc,
classification_report
)
from sklearn.feature_selection import f_regression

# Preprocesamiento y selección de modelos
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import (
    TimeSeriesSplit, GridSearchCV, RandomizedSearchCV, KFold,
    cross_val_score, learning_curve
)
from sklearn.pipeline import Pipeline
from sklearn.inspection import permutation_importance

# Redes neuronales con PyTorch
import torch
import torch.nn as nn
import torch.optim as optim

# Utilidades adicionales
```

```
from PIL import Image

# Supresión de advertencias
import warnings
from sklearn.exceptions import ConvergenceWarning

# Configuración de advertencias
warnings.filterwarnings("ignore", category=ConvergenceWarning)
#Leemos los distintos excels con la informacion en funcion de su
frecuencia temporal
df = pd.read_csv("./data.csv") #Infer dataformat, parsedates,
index_col="date"
df['date'] = pd.to_datetime(df['date'])
df.set_index('date', inplace=True)
df

df['demanda'] = df["demanda"].shift(1, fill_value = -4.582861)
#Last value available
df
#ELIMINAR fila
Procesado de los datos
#Vemos los tipos de datos
df.dtypes
df.info()
df.isna().sum()
df.corr()
# Heatmap matriz de correlaciones
#
==========================================================================
============
plt.figure(figsize=(8,8))
sns.heatmap(df.corr(),annot=True)

# Heatmap matriz de correlaciones (mejorado)
#
==========================================================================
============
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(7, 7))
corr_matrix = df.select_dtypes(include=['float64',
'int']).corr(method='pearson')
sns.heatmap(
    corr_matrix,
    annot     = True,
    cbar      = False,
    annot_kws = {"size": 6},
    vmin      = -1,
    vmax      = 1,
    center    = 0,
    cmap      = sns.diverging_palette(20, 220, n=200),
    square    = True,
    ax        = ax
)
ax.set_xticklabels(
    ax.get_xticklabels(),
    rotation = 45,
    horizontalalignment = 'right',
)
ax.tick_params(labelsize = 8)
# Gráfico de distribución para cada variable numérica
```

```
#
================================================================================
============

fig, axes = plt.subplots(nrows=3, ncols=4, figsize=(9, 5))
axes = axes.flat
columnas_numeric = df.select_dtypes(include=['float64',
'int']).columns

for i, colum in enumerate(columnas_numeric):
    sns.histplot(
        data    = df,
        x       = colum,
        stat    = "count",
        kde     = True,
        color   =
(list(plt.rcParams['axes.prop_cycle'])*2)[i]["color"],
        line_kws= {'linewidth': 2},
        alpha   = 0.3,
        ax      = axes[i]
    )
    axes[i].set_title(colum, fontsize = 7, fontweight = "bold")
    axes[i].tick_params(labelsize = 6)
    axes[i].set_xlabel("")


fig.tight_layout()
plt.subplots_adjust(top = 0.9)
fig.suptitle('Distribución variables numéricas', fontsize = 10,
fontweight = "bold");
# Gráfico de distribución para cada variable numérica
#
================================================================================
============
# Ajustar número de subplots en función del número de columnas
fig, axes = plt.subplots(nrows=4, ncols=3, figsize=(8, 5))
axes = axes.flat
columnas_numeric = df.select_dtypes(include=['float64',
'int']).columns
columnas_numeric = columnas_numeric.drop('demanda')
for i, colum in enumerate(columnas_numeric):
    sns.regplot(
        x           = df[colum],
        y           = df['demanda'],
        color       = "gray",
        marker      = '.',
        scatter_kws = {"alpha":0.4},
        line_kws    = {"color":"r","alpha":0.7},
        ax          = axes[i]
    )
    axes[i].set_title(f"Demanda vs {colum}", fontsize = 7,
fontweight = "bold")
    #axes[i].ticklabel_format(style='sci', scilimits=(-4,4),
axis='both')
    axes[i].yaxis.set_major_formatter(ticker.EngFormatter())
    axes[i].xaxis.set_major_formatter(ticker.EngFormatter())
    axes[i].tick_params(labelsize = 6)
    axes[i].set_xlabel("")
    axes[i].set_ylabel("")
```

```
# Se eliminan los axes vacíos
for i in [11]:
    fig.delaxes(axes[i])

fig.tight_layout()
plt.subplots_adjust(top=0.9)
fig.suptitle('Correlación con la Demanda', fontsize = 10,
fontweight = "bold")
ARIMA
# create a figure containing a single axes
fig, ax = plt.subplots(1, 1, figsize=[12, 4])
sns.lineplot(data=df['demanda'], ax=ax);
warnings.filterwarnings('ignore')
# Fit auto_arima function to AirPassengers dataset
stepwise_fit = auto_arima(df['demanda'], start_p = 1, start_q = 1,
                          max_p = 3, max_q = 3, m = 12,
                          start_P = 0, seasonal = True,
                          d = None, D = 1, trace = True,
                          error_action ='ignore',   # we don't
want to know if an order does not work
                          suppress_warnings = True,  # we don't
want convergence warnings
                          stepwise = True,
                          n_jobs=-1)           # set to stepwise

# To print the summary
stepwise_fit.summary()
warnings.filterwarnings('default')
d1 = pd.to_datetime('2000-01-03')
d2 = pd.to_datetime('2015-01-02')
d3 = pd.to_datetime('2022-12-29')
train =  df.loc[:d2]['demanda']
test = df.loc[d2:]['demanda']

train_demanda = train.resample('B').ffill()
train_demanda.index.freq = 'B'

model = sm.tsa.statespace.SARIMAX(train_demanda, order=(2,0,3),
seasonal_order = (2,1,0,12))
result = model.fit()

# forecast the test set
forecast = result.predict(start=d1, end=d3)

# Plot the results - First Figure
plt.figure(figsize=(14, 6))  # Adjust the size of the figure
(width, height)
plt.plot(train_demanda, label='Train')
plt.plot(test, label='Test')
plt.plot(forecast, label='Forecast')
plt.xlim(pd.to_datetime('2000-01-03'), test.index[-1])
plt.legend(loc='best')
plt.show()

# Plot the results - Second Figure
plt.figure(figsize=(14, 6))  # Adjust the size of the figure
(width, height)
plt.plot(train_demanda, label='Train')
```

```
plt.plot(forecast, label='Forecast', color='green')
plt.xlim(pd.to_datetime('2014-01-03'), train_demanda.index[-1])
plt.legend(loc='best')
plt.show()

# Plot the results - Third Figure
plt.figure(figsize=(14, 6))  # Adjust the size of the figure
(width, height)
plt.plot(test, label='Test')
plt.plot(forecast, label='Forecast', color='green')
plt.xlim(pd.to_datetime('2021-01-03'), test.index[-1])
plt.legend(loc='best')
plt.show()

ANALISIS EXPLICATIVO
X = df.drop('demanda', axis = 'columns')
y = df['demanda']
#LINEAL REGRESION
#MODELO
model = smf.ols('demanda ~ wti_price + sp500_index + sp500_change
+ dollar_index + energy_index + copper_price + vix_index +
inventories + tasa_paro + gdp + vehicle_production', data=df)
results = model.fit()
results.summary()
# create the scaler
scaler = StandardScaler()
scaler.fit(X)
# transform the data
X_standardized = scaler.transform(X)
X = pd.DataFrame(X_standardized, columns=X.columns).values
y = y.values
#MODELO
model = smf.ols('demanda ~ wti_price + sp500_index + sp500_change
+ dollar_index + energy_index + copper_price + vix_index +
inventories + tasa_paro + gdp + vehicle_production', data=df)
results = model.fit()
results.summary()

plt.figure(figsize=(10, 6))
sns.scatterplot(x=results.fittedvalues, y=results.resid)
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.title('Residuals vs Fitted Values')
plt.show()

plt.figure(figsize=(10, 6))
sns.histplot(results.resid, kde=True)
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')
plt.show()

#DECISION TREE
#Procesado de datos
X = df.drop('demanda', axis = 'columns')
y = df['demanda']

# Create a decision tree regressor object
regressor = DecisionTreeRegressor()
```

```
param_grid = {
    'criterion': ['squared_error', 'friedman_mse',
'absolute_error'],
    'splitter': ['best', 'random'],
    'max_depth': [3, 5, 7],
    'min_samples_split': [2, 3, 5, 6],
    'min_samples_leaf': [1, 2, 4, 5, 6]
}
grid_search = GridSearchCV(estimator=regressor,
param_grid=param_grid, cv=5, n_jobs=-1)
grid_search.fit(X, y)
best_params = grid_search.best_params_
print(best_params)

# Make predictions using the testing data
regressor =
DecisionTreeRegressor(criterion=best_params['criterion'],
                                splitter=best_params['splitter'],

max_depth=best_params['max_depth'],

min_samples_split=best_params['min_samples_split'],

min_samples_leaf=best_params['min_samples_leaf'])

regressor.fit(X, y)
y_pred = regressor.predict(X)

# Plot the predicted values
plt.figure(figsize=(11, 4))
plt.scatter(x = y, y = y_pred, color = 'blue', label =
'Predicted')
# Set the title
plt.title('Decision Tree Regression')
# Set the x-axis label
plt.xlabel('Demanda real')
# Set the y-axis label
plt.ylabel('Prediccion')
# Show legend
plt.legend()
# Show the plot
plt.show()

# Calculate the residuals
residuals = y - y_pred
# Plot the residuals
plt.figure(figsize=(11, 4))
plt.scatter(y, residuals, color = 'red', label = 'Residuals')
# Set the title
plt.title('Residual Plot')
# Set the x-axis label
plt.xlabel('Independent Variables')
# Set the y-axis label
plt.ylabel('Residuals')
# Show legend
plt.legend()
# Show the plot
plt.show()
```

```
#Plot a histogram of the residuals
plt.figure(figsize=(11, 4))
plt.hist(y_pred - y, bins=50)
plt.title("Residual Histogram")
plt.xlabel("Error")
plt.ylabel("Count")
plt.show()

# Get feature importance scores
importances = regressor.feature_importances_
# Get the names of the features
features = X.columns
# Set figure size
plt.figure(figsize=(11, 4))
# Plot bar chart
plt.bar(features, importances)
# Add title and labels
plt.title('Importance of Each Variable')
plt.xlabel('Features')
plt.ylabel('Importance Score')
# Show plot
plt.show()

# print the importance of each variable
for i, importances in enumerate(importances):
  print('The importance of {} is {}'.format(X.columns[i],
importances))
######################################################################
######################################################################
######################################################################
######################################################################
######################################################################
######################################################################
######################################################################
######

ANALISIS PREDICTIVO
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
MODELO 1: REGRESION LINEAL
##REGRESION LINEAL MULTIPLE
params = {'fit_intercept': [True, False], 'copy_X': [True, False]}
model = LinearRegression()
tsvc = TimeSeriesSplit(n_splits= 2)

random_search = RandomizedSearchCV(model,
param_distributions=params, cv=tsvc, n_iter=4, n_jobs=-
1).fit(X_train, y_train)
best_params = random_search.best_params_

model =
LinearRegression(fit_intercept=best_params['fit_intercept'],
copy_X=best_params['copy_X']).fit(X_train, y_train)
y_pred = model.predict(X_test)
#Calculating R2
r2 = r2_score(y_test, y_pred)
print("\nR-Squared:", r2)
```

```
print("R2 is a measure of how well a model fits the data. It is
calculated by taking the squared difference between the predicted
values and the actual values, dividing by the sum of squared
errors of the predicted values, and subtracting that from 1. A
higher R2 value indicates that the model is a better fit for the
data.")

#Calculate the root mean squared error
from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("\nRoot Mean Squared Error: {}".format(rmse))
print("Root Mean Squared Error (RMSE) measures the difference
between predicted values and actual values. It is a measure of how
well the model is able to predict the target values, with lower
RMSE values indicating a better fit. ")

#Calculate the Mean Absolute Error
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, y_pred)
print("\nMean Absolute Error: {}".format(mae))
print("Mean Absolute Error (MAE) is a measure of the average
magnitude of the errors in a set of predictions, without
considering their direction. It measures the average magnitude of
the errors in a set of predictions, without considering their
direction. Lower values of MAE indicate better fit.")

#Plot the residuals
plt.figure(figsize=(11, 4))
plt.scatter(y_pred, y_pred - y_test, c='g', s=40)
plt.hlines(y=0, xmin=-5, xmax=5)
plt.title("Residual Plot")
plt.ylabel("Error")
plt.xlabel("Predicted Values")
plt.show()

#Plot the actual values and the predicted values
plt.figure(figsize=(11, 4))
plt.scatter(y_pred, y_test, c='b', s=40)
plt.title("Regresion Lineal: Actual vs Predicted")
plt.xlabel("Predicted Values")
plt.ylabel("Actual Values")
plt.show()

#Plot a histogram of the residuals
plt.figure(figsize=(11, 4))
plt.hist(y_pred - y_test, bins=50)
plt.title("Residual Histogram")
plt.xlabel("Error")
plt.ylabel("Count")
plt.show()

# Get coefficients of the model
coef = model.coef_
# Get the names of the features
features = X.columns
# Create a dataframe of the features and their importance and sort
it by importance
feature_importances = pd.DataFrame({'feature': features,
'importance': coef})
```

```
feature_importances.sort_values('importance', ascending=False,
inplace=True)
print(feature_importances)
# Create a bar plot of the feature importances
plt.figure(figsize=(11, 4))
plt.barh(feature_importances['feature'],
feature_importances['importance'])
plt.xlabel('Importance')
plt.ylabel('BETAS')
plt.title('Regresion Lineal Variables')
plt.savefig('image1.png')
plt.show()

# print the importance of each variable
for i, coefficient in enumerate(coef):
  print('The importance of {} is {}'.format(X_train.columns[i],
coefficient))
performance_metrics = {
    "Linear Regression": {
        "RMSE": rmse,
        "MAE": mae,
    }
}
MODELO 2: REGRESION LINEAL CON REGULARIZACION RIDGE
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
##REGRESION LINEAL MULTIPLE

# Create a list of alphas to cross-validate against
alphas = np.arange(0.1,100,0.5)

# Create a function to return the negated mean squared error
def neg_mse(params):
    mse = -cross_val_score(Ridge(alpha=params), X_train, y_train,
scoring="neg_mean_squared_error", cv=3).mean()
    return mse

# Iterate through the alphas list and find the alpha with the
lowest mean squared error
best_alpha = 0
best_mse = np.infty
for alpha in alphas:
    current_mse = neg_mse(alpha)
    if current_mse < best_mse:
        best_alpha = alpha
        best_mse = current_mse

print('Best alpha:', best_alpha)
print('Best MSE:', best_mse)

model = Ridge(alpha=best_alpha).fit(X_train, y_train)
y_pred = model.predict(X_test)
#Calculating R2
r2 = r2_score(y_test, y_pred)
print("\nR-Squared:", r2)
print("R2 is a measure of how well a model fits the data. It is
calculated by taking the squared difference between the predicted
```

```
values and the actual values, dividing by the sum of squared
errors of the predicted values, and subtracting that from 1. A
higher R2 value indicates that the model is a better fit for the
data.")

#Calculate the root mean squared error
from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("\nRoot Mean Squared Error: {}".format(rmse))
print("Root Mean Squared Error (RMSE) measures the difference
between predicted values and actual values. It is a measure of how
well the model is able to predict the target values, with lower
RMSE values indicating a better fit. ")

#Calculate the Mean Absolute Error
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, y_pred)
print("\nMean Absolute Error: {}".format(mae))
print("Mean Absolute Error (MAE) is a measure of the average
magnitude of the errors in a set of predictions, without
considering their direction. It measures the average magnitude of
the errors in a set of predictions, without considering their
direction. Lower values of MAE indicate better fit.")

# Get the coefficients of the ridge model and the feature names
coefs = model.coef_

features = X.columns
# Create a dataframe of the features and their importance and sort
it by importance
feature_importances = pd.DataFrame({'feature': features,
'importance': coefs})
feature_importances.sort_values('importance', ascending=False,
inplace=True)
print(feature_importances)
# Create a bar plot of the feature importances
plt.figure(figsize=(11, 4))
plt.barh(feature_importances['feature'],
feature_importances['importance'])
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('Ridge Variables Importancia')
plt.savefig('image2.png')
plt.show()


#Plot the residuals
plt.figure(figsize=(11, 4))
plt.scatter(y_pred, y_pred - y_test, c='g', s=40)
plt.hlines(y=0, xmin=-5, xmax=5)
plt.title("Residual Plot")
plt.ylabel("Error")
plt.xlabel("Predicted Values")
plt.show()

#Plot the actual values and the predicted values
plt.figure(figsize=(11, 4))
plt.scatter(y_pred, y_test, c='b', s=40)
plt.title("Regresion Lineal: Actual vs Predicted")
```

```
plt.xlabel("Predicted Values")
plt.ylabel("Actual Values")
plt.show()

#Plot a histogram of the residuals
plt.figure(figsize=(11, 4))
plt.hist(y_pred - y_test, bins=50)
plt.title("Residual Histogram")
plt.xlabel("Error")
plt.ylabel("Count")
plt.show()
# Updating performance metrics for an existing model
performance_metrics["Regresion Ridge"] = {
    "RMSE": rmse,
    "MAE": mae
}
MODELO 3: SUPPORT VECTOR MACHINE
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
# create the scaler
scaler = StandardScaler()
scaler.fit(X)
# transform the data
X_standardized = scaler.transform(X_train)
X_standardized = pd.DataFrame(X_standardized,
columns=X_train.columns)
X_train = X_standardized

# transform the data
X_standardized = scaler.transform(X_test)
X_standardized = pd.DataFrame(X_standardized,
columns=X_test.columns)
X_test = X_standardized
# train the model on train set
from scipy.stats import uniform
param_distributions = {
    'kernel': ['rbf'],
    'gamma': ['scale', 'auto'],
    'C': [0.5, 4], #Posibilidad de hacer una distribucion con
lista
    'epsilon': [0.1, 0.5]
}

random_search = RandomizedSearchCV(estimator=SVR(),
param_distributions=param_distributions, n_iter=8, n_jobs=-1,
cv=5).fit(X_train, y_train)
best_params = random_search.best_params_
print(best_params)

# Create the model with the best parameters
model = SVR(kernel=best_params['kernel'],
gamma=best_params['gamma'], C=best_params['C'],
epsilon=best_params['epsilon']).fit(X_train, y_train)

# print prediction results
predictions = model.predict(X_test)
# calculate metrics
```

```
mse = mean_squared_error(y_test, predictions)
r2score = r2_score(y_test, predictions)

# print results
print("Mean Squared Error:", mse)
print("R2 Score:", r2score)

#Calculate the root mean squared error
from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("\nRoot Mean Squared Error: {}".format(rmse))

#Calculate the Mean Absolute Error
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, y_pred)
print("\nMean Absolute Error: {}".format(mae))

# visualize results
plt.figure(figsize=(11, 4))
plt.scatter(y_test, predictions)
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("SVR Model")
plt.show()

result = permutation_importance(model, X_test, y_test,
n_repeats=10, random_state=0, n_jobs=-1)
importances_mean = result.importances_mean
importances_std = result.importances_std
# Assume 'feature_names' contains the names of the features
plt.figure(figsize=(11, 4))
plt.barh(X_train.columns, importances_mean)
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('SVR Importances Variables')
plt.savefig('image3.png')
plt.show()

# calculate residuals
residuals = y_test - predictions
# create histogram
plt.figure(figsize=(11, 4))
plt.hist(residuals, bins=20)
plt.xlabel('Residuals')
plt.ylabel('Frequency')
plt.title('Distribution of Residuals')
# show plot
plt.show()
# Updating performance metrics for an existing model
performance_metrics["SVR"] = {
    "RMSE": rmse,
    "MAE": mae
}

MODELO 4: RED NEURONAL
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
```

```
# create the scaler
scaler = StandardScaler()
scaler.fit(X)
# transform the data
X_standardized = scaler.fit_transform(X_train)
X_standardized = pd.DataFrame(X_standardized,
columns=X_train.columns)
X_train = X_standardized

# transform the data
X_standardized = scaler.transform(X_test)
X_standardized = pd.DataFrame(X_standardized,
columns=X_test.columns)
X_test = X_standardized
# Convert data to PyTorch tensors
X_train_tensor = torch.tensor(X_train.values, dtype=torch.float32)
y_train_tensor = torch.tensor(y_train.values,
dtype=torch.float32).unsqueeze(1)
X_test_tensor = torch.tensor(X_test.values, dtype=torch.float32)
y_test_tensor = torch.tensor(y_test.values,
dtype=torch.float32).unsqueeze(1)

# Define the neural network
class Net(nn.Module):
    def __init__(self, input_dim, output_dim, hidden_dim1,
hidden_dim2, dropout):
        super(Net, self).__init__()
        self.fc1 = nn.Linear(input_dim, hidden_dim1)
        self.fc2 = nn.Linear(hidden_dim1, hidden_dim2)
        self.fc3 = nn.Linear(hidden_dim2, output_dim)
        self.dropout = nn.Dropout(dropout)
        self.activation = nn.LeakyReLU()

    def forward(self, x):
        x = self.activation(self.fc1(x))
        x = self.dropout(x)
        x = self.activation(self.fc2(x))
        x = self.fc3(x)
        return x

# Define the hyperparameters to search
input_dim = X_train.shape[1]
output_dim = 1
hidden_dim1_values = [64, 128]
hidden_dim2_values = [32, 64]
dropout_values = [0.2, 0.3]
lr_values = [0.001, 0.01]
num_epochs = 1000
patience = 50

# Keep track of the best model and its performance
best_model = None
best_rmse = float('inf')
best_mae = float('inf')

# Loop over all combinations of hyperparameters
for hidden_dim1 in hidden_dim1_values:
    for hidden_dim2 in hidden_dim2_values:
        for dropout in dropout_values:
```

```
            for lr in lr_values:
                # Create the model, loss function, and optimizer
                model = Net(input_dim, output_dim, hidden_dim1,
hidden_dim2, dropout)
                criterion = nn.MSELoss()
                optimizer = optim.AdamW(model.parameters(), lr=lr)

                # Implement early stopping
                best_val_loss = float('inf')
                counter = 0

                # Train the model
                for epoch in range(num_epochs):
                    model.train()
                    optimizer.zero_grad()
                    y_pred_tensor = model(X_train_tensor)
                    loss = criterion(y_pred_tensor,
y_train_tensor)
                    loss.backward()
                    optimizer.step()

                    # Check for early stopping
                    if loss.item() < best_val_loss:
                        best_val_loss = loss.item()
                        counter = 0
                    else:
                        counter += 1
                        if counter >= patience:
                            break

                # Evaluate the model on test data
                model.eval()
                with torch.no_grad():
                    y_pred_tensor = model(X_test_tensor)

                # Calculate the RMSE and MAE
                rmse = torch.sqrt(criterion(y_pred_tensor,
y_test_tensor)).item()
                mae = torch.mean(torch.abs(y_pred_tensor -
y_test_tensor)).item()

                # Check if this is the best model so far
                if rmse < best_rmse:
                    best_rmse = rmse
                    best_mae = mae
                    best_model = model

print(f'Best RMSE: {best_rmse:.4f}, Best MAE: {best_mae:.4f}')
# Convert the predicted tensor back to NumPy array
y_pred = y_pred_tensor.numpy()

# Plot true vs. predicted values
plt.figure(figsize=(11, 4))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.xlabel("True Values")
plt.ylabel("Predicted Values")
plt.title("True vs. Predicted Values")
plt.plot([y_test.min(), y_test.max()], [y_test.min(),
y_test.max()], 'r', lw=2)
```

```
plt.show()

# Plot residuals
residuals = y_test - y_pred.reshape(-1)
plt.figure(figsize=(11, 4))
plt.scatter(y_test, residuals, alpha=0.5)
plt.axhline(y=0, color='r', lw=2)
plt.xlabel("True Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()

# Updating performance metrics for an existing model
performance_metrics["FeedForwardNN"] = {
    "RMSE": rmse,
    "MAE": mae
}
MODELO 5: BOSQUE DE REGRESION
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
# Create a RandomForestRegressor
rf = RandomForestRegressor()

# Define the hyperparameters and their ranges for
RandomizedSearchCV
param_dist = {
    'n_estimators': np.arange(100, 1001, 100),
    'max_depth': [None] + list(np.arange(3, 21)),
    'min_samples_split': np.arange(2, 11),
    'min_samples_leaf': np.arange(1, 11),
    'max_features': ['sqrt', 'log2'],
    'bootstrap': [True, False]
}

# Create the RandomizedSearchCV object
random_search = RandomizedSearchCV(
    estimator=rf,
    param_distributions=param_dist,
    n_iter=100,
    cv=tscv,
    scoring='neg_mean_squared_error',
    n_jobs=-1,
    verbose=1,
    random_state=42
)

# Fit the RandomizedSearchCV object to the training data
random_search.fit(X_train, y_train)

# Get the best hyperparameters
best_params = random_search.best_params_
print(best_params)
# Train the RandomForestRegressor model with the best
hyperparameters
best_rf = RandomForestRegressor(**best_params)
best_rf.fit(X_train, y_train)
```

```
# Test the model on the test set
y_pred = best_rf.predict(X_test)

# Calculate the performance metrics as needed (e.g.,
mean_squared_error, r2_score, etc.)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print(f"RMSE: {rmse}")
# Calculate MAE
mae = mean_absolute_error(y_test, y_pred)
print(f"MAE: {mae}")

plt.figure(figsize=(11, 4))
plt.scatter(y_test, y_pred)
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs. Predicted')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)],
color='red', linestyle='--')
plt.show()

residuals = y_test - y_pred
plt.figure(figsize=(11, 4))
plt.scatter(y_pred, residuals)
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.title('Residuals Plot')
plt.axhline(y=0, color='red', linestyle='--')
plt.show()

plt.figure(figsize=(11, 4))
plt.hist(residuals, bins=20)
plt.xlabel('Residuals')
plt.ylabel('Frequency')
plt.title('Residuals Histogram')
plt.show()

plt.figure(figsize=(11, 4))
feat_importances = pd.Series(best_rf.feature_importances_,
index=X.columns)
feat_importances.nlargest(len(X.columns)).plot(kind='barh')
plt.xlabel('Importance')
plt.ylabel('Features')

plt.title('Random Forest Feature Importances')
plt.savefig('image4.png')
plt.show()
# Updating performance metrics for an existing model
performance_metrics["Random Forest"] = {
    "RMSE": rmse,
    "MAE": mae
}
MODELO 6: KNN
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
# create the scaler
scaler = StandardScaler()
scaler.fit(X)
```

```
# transform the data
X_standardized = scaler.fit_transform(X_train)
X_standardized = pd.DataFrame(X_standardized,
columns=X_train.columns)
X_train = X_standardized

# transform the data
X_standardized = scaler.transform(X_test)
X_standardized = pd.DataFrame(X_standardized,
columns=X_test.columns)
X_test = X_standardized
# Create a KNN regressor
knn = KNeighborsRegressor()

# Create a pipeline to scale the data before feeding it to the KNN
model
pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('knn', knn)
])

# Set hyperparameters to search through
param_dist = {
    'knn__n_neighbors': np.arange(1, 31),
    'knn__weights': ['uniform', 'distance'],
    'knn__metric': ['euclidean', 'manhattan', 'chebyshev',
'minkowski']
}

# Set up RandomizedSearchCV with the KNN regressor, parameter
distribution, and the time series cross-validator
random_search = RandomizedSearchCV(
    pipeline, param_distributions=param_dist, n_iter=50, cv=tscv,
scoring='neg_mean_squared_error', n_jobs=-1, verbose=1,
random_state=42
)

# Fit the model using the training set
random_search.fit(X_train, y_train)

# Print the best parameters found
print("Best parameters found: ", random_search.best_params_)

# Make predictions using the test set
y_pred = random_search.predict(X_test)

# Calculate RMSE and MAE
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)

print("RMSE:", rmse)
print("MAE:", mae)



# Plot true vs. predicted values
plt.figure(figsize=(11, 4))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.xlabel("True Values")
```

```
plt.ylabel("Predicted Values")
plt.title("KNN: True vs. Predicted Values")
plt.plot([y_test.min(), y_test.max()], [y_test.min(),
y_test.max()], 'r', lw=2)
plt.show()

# Plot residuals
residuals = y_test - y_pred.reshape(-1)
plt.figure(figsize=(11, 4))
plt.scatter(y_test, residuals, alpha=0.5)
plt.axhline(y=0, color='r', lw=2)
plt.xlabel("True Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()

plt.figure(figsize=(11, 4))
plt.hist(residuals, bins=20)
plt.xlabel('Residuals')
plt.ylabel('Frequency')
plt.title('Residuals Histogram')
plt.show()

plt.show()
# Updating performance metrics for an existing model
performance_metrics["KNN"] = {
    "RMSE": rmse,
    "MAE": mae
}
# Updating performance metrics for an existing model
performance_metrics["Naive Model"] = {
    "RMSE": rmse,
    "MAE": mae
}
MODELO 7: Modelo Naive
tscv = TimeSeriesSplit(n_splits = 5)
for train_index, test_index in tscv.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
# Fit the naive model to your data
y_pred = np.zeros(len(y_test))

# Calculate RMSE and MAE
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)

print("RMSE:", rmse)
print("MAE:", mae)

# Plot true vs. predicted values
plt.figure(figsize=(11, 4))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.xlabel("True Values")
plt.ylabel("Predicted Values")
plt.title("True vs. Predicted Values")
plt.plot([y_test.min(), y_test.max()], [y_test.min(),
y_test.max()], 'r', lw=2)
plt.show()
```

```
# Plot residuals
residuals = y_test - y_pred.reshape(-1)
plt.figure(figsize=(11, 4))
plt.scatter(y_test, residuals, alpha=0.5)
plt.axhline(y=0, color='r', lw=2)
plt.xlabel("True Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()

plt.figure(figsize=(11, 4))
plt.hist(residuals, bins=20)
plt.xlabel('Residuals')
plt.ylabel('Frequency')
plt.title('Residuals Histogram')
plt.show()

plt.show()
COMPARACION FINAL MODELOS
model_metrics = performance_metrics
model_metrics
# Extract model names, RMSE, and MAE values
model_names = list(model_metrics.keys())
rmse_values = [metrics['RMSE'] for metrics in
model_metrics.values()]
mae_values = [metrics['MAE'] for metrics in
model_metrics.values()]

# Set up the bar plot
x = np.arange(len(model_names))
width = 0.35

fig, ax = plt.subplots(figsize=(14, 6))
rects1 = ax.bar(x - width/2, rmse_values, width, label='RMSE')
rects2 = ax.bar(x + width/2, mae_values, width, label='MAE')

# Add labels, title, and custom x-axis tick labels
ax.set_ylabel('Error')
ax.set_title('Model Comparison by RMSE and MAE')
ax.set_xticks(x)
ax.set_xticklabels(model_names)
ax.legend()

# Attach a text label above each bar in rects1 and rects2,
displaying its height
def autolabel(rects):
    for rect in rects:
        height = rect.get_height()
        ax.annotate('%.3f' % height,
                    xy=(rect.get_x() + rect.get_width() / 2,
height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')

autolabel(rects1)
autolabel(rects2)

# Display the plot
```

```
plt.show()
# Open the saved images
image1 = Image.open('image1.png')
image2 = Image.open('image2.png')
image3 = Image.open('image3.png')
image4 = Image.open('image4.png')

# Create a new figure and axes for displaying the images
fig, axes = plt.subplots(nrows=4, ncols=1, figsize=(18, 12))

# Display the images in a 4x1 grid
axes[0].imshow(image1)
axes[0].axis('off')  # Remove axes for a cleaner look
axes[1].imshow(image2)
axes[1].axis('off')  # Remove axes for a cleaner look
axes[2].imshow(image3)
axes[2].axis('off')  # Remove axes for a cleaner look
axes[3].imshow(image4)
axes[3].axis('off')  # Remove axes for a cleaner look

# Show the combined plot
plt.show()
```

Codigo de agregacion de datos

```
PROCESADO DE LOS DATOS
#Importamos las librerias necesarias
import pandas as pd
import numpy as np
#Leemos los distintos excels con la informacion en funcion de su
frecuencia temporal
df_diario = pd.read_excel("./Datos_desglosados/Diario.xlsx")
df_sem = pd.read_excel('./Datos_desglosados/Semanales.xlsx')
df_men = pd.read_excel('./Datos_desglosados/Mensuales.xlsx')
df_tri = pd.read_excel('./Datos_desglosados/Trimestrales.xlsx')

#Convertimos date (fecha) al tipo datatime para poder unir los
datasets
df_diario['date'] = pd.to_datetime(df_diario['date'])
df_sem['date'] = pd.to_datetime(df_sem['date'])
df_men['date'] = pd.to_datetime(df_men['date'])
df_tri['date'] = pd.to_datetime(df_tri['date'])
#set date as the index
df_diario.set_index('date', inplace=True)
df_sem.set_index('date', inplace=True)
df_men.set_index('date', inplace=True)
df_tri.set_index('date', inplace=True)
#Cambiamos la frecuencia a diaria (business day)
df_sem2 = df_sem.resample('B').mean().fillna(method='ffill')
df_sem2
#Cambiamos la frecuencia a diaria (business day)
df_men2 = df_men.resample('B').mean().fillna(method='ffill')
df_men2
#Cambiamos la frecuencia a diaria (business day)
df_tri2 = df_tri.resample('B').mean().fillna(method='ffill')
df_tri2
#Fusionamos los datos en unico dataframe
df_merged = pd.merge(df_diario, df_sem2, on='date', how = 'left')
df_merged = pd.merge(df_merged, df_men2, on='date', how = 'left')
df_merged = pd.merge(df_merged, df_tri2, on='date', how = 'left')
df_merged
#Guardamos el data frame
df_merged.to_csv('data.csv', index=True)
```