Facultad de Ciencias Económicas y Empresariales, ICADE

DESARROLLO DE MODELOS DE PREDICCIÓN DE PRECIOS INMOBILIARIOS UTILIZANDO TÉCNICAS DE MACHINE LEARNING, TRABAJO FIN DE GRADO

Autor: Beatriz Sicilia Gómez

Director: Rafael Castellote Azorín

Madrid | Abril 2024

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Desarrollo de Modelos de Predicción de Precios Inmobiliarios utilizando técnicas de Machine Learning

en la Universidad Pontificia Comillas en el

curso académico 2023/24 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Beatriz Sicilia Gómez Fecha: 22/04/2024

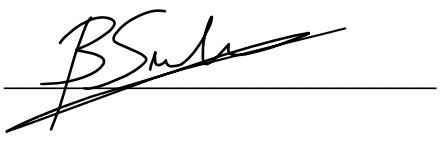
Declaración de Uso de Herramientas de Inteligencia Artificial Generativa en Trabajos Fin de Grado

Por la presente, yo, Beatriz Sicilia Gómez, estudiante del Doble Grado en Ingeniería de Telecomunicaciones y Business Analytics de la Universidad Pontificia Comillas al presentar mi Trabajo Fin de Grado titulado "Desarrollo de Modelos de Predicción de Precios Inmobiliarios utilizando técnicas de Machine Learning" declaro que he utilizado la herramienta de Inteligencia Artificial Generativa ChatGPT u otras similares de IAG de código sólo en el contexto de las actividades descritas a continuación:

1. **Metodólogo:** Para descubrir métodos aplicables a problemas específicos de investigación.
2. **Corrector de estilo literario y de lenguaje:** Para mejorar la calidad lingüística y estilística del texto.
3. **Traductor:** Para traducir textos de un lenguaje a otro.

Afirmo que toda la información y contenido presentados en este trabajo son producto de mi investigación y esfuerzo individual, excepto donde se ha indicado lo contrario y se han dado los créditos correspondientes (he incluido las referencias adecuadas en el TFG y he explicitado para que se ha usado ChatGPT u otras herramientas similares). Soy consciente de las implicaciones académicas y éticas de presentar un trabajo no original y acepto las consecuencias de cualquier violación a esta declaración.

Fecha: 22 de Abril de 2024

Firma: 

DESARROLLO DE MODELOS DE PREDICCIÓN DE PRECIOS INMOBILIARIOS MEDIANTE TÉCNICAS DE MACHINE LEARNING.

Autor: Sicilia Gómez, Beatriz.
Director: Castellote Azorín, Rafael.
Entidad Colaboradora: ICADE – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

En este trabajo de fin de grado se ha llevado a cabo una investigación exhaustiva sobre el estado actual del uso de técnicas de aprendizaje automático en el mercado inmobiliario, con un enfoque específico en la predicción de precios de venta de viviendas en Madrid. Se desarrollaron varios modelos de Machine Learning y Deep Learning utilizando datos obtenidos del portal inmobiliario Idealista.

La metodología seguida comienza con la extracción de datos a través de una API proporcionada por el portal mencionado, lo que garantizó la obtención de datos actualizados y relevantes. Estos datos se almacenaron en la nube para facilitar su acceso y gestión durante el proceso de análisis y modelado.

El análisis exploratorio de datos desempeñó un papel crucial en esta investigación, permitiendo una comprensión profunda de la naturaleza de los datos y la identificación de patrones significativos. Posteriormente, se procedió a la implementación de varios modelos de aprendizaje automático, utilizando bibliotecas de Python como scikit-learn para la regresión lineal y Random Forest, Xgboost para modelos de ensemble, y TensorFlow con Keras para modelos basados en redes neuronales.

Se llevó a cabo un ajuste de hiperparámetros para optimizar el rendimiento de los modelos y se utilizó la técnica de cross-validation para determinar su rendimiento real. La evaluación de los modelos se realizó utilizando métricas estándar en el campo, como el Mean Squared Error (MSE), Root Mean Squared Error (RMSE) y Mean Absolute Error (MAE), presentando los resultados basados en conjuntos de prueba.

Se observó que los modelos desarrollados demostraron una capacidad predictiva notable, con errores bajos, lo que sugiere su utilidad potencial en el mercado inmobiliario. Además de la evaluación de los modelos, se discuten posibles aplicaciones prácticas de los mismos en el contexto del mercado inmobiliario, como la identificación de oportunidades de inversión y la tasación automatizada de viviendas.

En conclusión, este trabajo ofrece una contribución significativa al campo al demostrar la utilidad y eficacia de los algoritmos de aprendizaje automático en la predicción de precios de viviendas en el mercado inmobiliario de Madrid.

Palabras clave: Machine Learning, Mercado inmobiliario, Modelos de regresión, Tasación automatizada, Deep Learning, Aprendizaje supervisado, Analítica de datos, Métodos de ensemble.

DEVELOPMENT OF REAL ESTATE PRICE PREDICTION MODELS USING MACHINE LEARNING TECHNIQUES.

Author: Sicilia Gómez, Beatriz.
Supervisor: Castellote Azorín, Rafael.
Collaborating Entity: ICADE– Universidad Pontificia Comillas

ABSTRACT

In this final thesis, an exhaustive investigation was conducted on the current state of using machine learning techniques in the real estate market, with a specific focus on predicting housing sale prices in Madrid. Several machine learning and deep learning models were developed using data obtained from the Idealista real estate portal.

The methodology followed begins with data extraction through an API provided by the mentioned portal, ensuring the acquisition of updated and relevant data. These data were stored in the cloud to facilitate access and management during the analysis and modeling process.

Exploratory data analysis played a crucial role in this research, allowing for a deep understanding of the data nature and the identification of significant patterns. Subsequently, multiple machine learning models were implemented using Python libraries such as scikit-learn for linear regression and Random Forest, Xgboost for ensemble models, and TensorFlow with Keras for neural network-based models.

Hyperparameter tuning was performed to optimize the models' performance, and cross-validation technique was utilized to determine their real-world performance. Model evaluation was conducted using standard metrics in the field, such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE), presenting the results based on test sets.

It was observed that the developed models demonstrated remarkable predictive capability, with low errors, suggesting their potential utility in the real estate market. In addition to model evaluation, possible practical applications of the models in the real estate market, such as identifying investment opportunities and automated property valuation, were discussed.

In conclusion, this work provides a significant contribution to the field by demonstrating the usefulness and effectiveness of machine learning algorithms in predicting housing prices in the Madrid real estate market.

Keywords: Machine Learning, Real Estate Market, Regression Models, Automated Valuation, Deep Learning, Supervised Learning, Data Analytics, Ensemble Methods.

Índice de la memoria

Índice de figuras

Índice de tablas

Índice de ecuaciones

Glosario de tecnicismos

API	Interfaz de Programación de Aplicaciones, permite la interacción entre distintos sistemas de software.
Deep Learning	Subconjunto de ML que utiliza modelos de redes neuronales profundas.
Embedding	Representación de datos de alta dimensionalidad en un espacio de menor dimensión aprendida durante el entrenamiento del modelo.
Ensemble	Técnica de ML que consiste en la combinación de múltiples modelos básicos para después combinar sus predicciones.
Forecasting	Proceso de predecir valores futuros basados en tendencias y datos históricos
Gradiente	Vector que indica la dirección y la magnitud del cambio más
Grid search	Método para buscar los mejores hiperparámetros, consiste en evaluar sistemáticamente combinaciones de valores en una matriz predefinida.
Hiperparámetros	Parámetros externos a los modelos de ML que afectan a su rendimiento y deben ajustarse antes del entrenamiento.
Learning rate	Tasa de ajuste de los parámetros durante el entrenamiento de un modelo de ML.
Machine Learning	Campo de la inteligencia artificial que permite a las máquinas utilizar los datos para aprender patrones y tomar decisiones.

Overfitting	Fenómeno en el que un modelo de ML se ajusta demasiado a los datos de entrenamiento y no es capaz de generalizar a la hora de predecir utilizando nuevas observaciones.
Red Neuronal	Modelo de ML inspirado en el funcionamiento del cerebro humano, una red con capas de neuronas interconectadas.
Regresión Lineal	Modelo de ML que modela la relación entre la variable dependiente y las explicativas mediante una línea recta.
Regularización	Técnica en la que se penalizan los modelos según su complejidad para evitar el sobreajuste.
Residuos	Diferencia entre los valores observados y los valores predichos por un modelo de regresión.

Glosario de acrónimos

API	Application Programming Interface
BCE	Banco Central Europeo
CNN	Convolutional Neural Network
DL	Deep Learning
ML	Machine Learning
MLP	Multi Layer Perceptron
RF	Random Forest

Capítulo 1. INTRODUCCIÓN

El mercado inmobiliario, con su naturaleza dinámica y competitiva, requiere una determinación precisa del precio de venta de las propiedades. Sin embargo, esto se ve desafiado por la variabilidad de los criterios de evaluación, los cuales se ven altamente influenciados por factores externos, como la oferta y la demanda, las condiciones económicas o las tendencias del momento en el mercado. Ante este escenario, surge la necesidad de contar con herramientas tecnológicas que puedan objetivar y mejorar el proceso de fijación de precios de viviendas.

El objetivo principal de este trabajo radica en desarrollar un sistema capaz de predecir, a partir de una serie de variables específicas, el precio de venta de una vivienda en el mercado inmobiliario de Madrid. Dada la naturaleza cambiante de los criterios mencionados, dependientes del contexto del mercado, se ha ideado un sistema adaptable que pueda ser alimentado con datos actualizados de manera recurrente.

Para lograr este propósito, se han empleado metodologías propias de la ciencia de datos y el aprendizaje automático, así como técnicas especializadas en la extracción de datos de entornos web. Se ha llevado a cabo una exhaustiva exploración de distintos algoritmos de Machine Learning, evaluando su rendimiento y precisión en la predicción del precio de venta de las propiedades. Además, se han utilizado técnicas de Deep Learning para su respectiva comparación con los algoritmos tradicionales.

Este enfoque integral busca proporcionar a los usuarios una herramienta eficiente y fiable para identificar oportunidades de inversión inmobiliaria o para utilizarla como un sistema de tasación confiable en el proceso de compra y venta de propiedades.

El código fuente desarrollado en este proyecto puede encontrarse en el siguiente Repositorio Github. En el Anexo I se proporciona más información sobre la estructura del código.

Capítulo 2. ESTADO DEL ARTE

2.1 MERCADO INMOBILIARIO ESPAÑOL

En esta sección, se analizarán los tres ciclos distintivos que han caracterizado la trayectoria del mercado inmobiliario en España. A través de este análisis se explorarán las tendencias, los factores impulsores y los impactos de estos ciclos en la economía y la sociedad española.

La primera etapa se extiende desde los años noventa hasta la llegada de la crisis económica mundial en 2007. Durante esta década, el mercado inmobiliario experimentó un notorio auge, alcanzando máximos históricos en el crecimiento de los precios de la vivienda y en el número de operaciones de compraventa anuales. Este rápido ascenso fue impulsado por una combinación de factores, como la fuerte inversión en el sector inmobiliario y el incremento del consumo en las familias españolas, motivado por el acceso fácil a la financiación.

La creación de la burbuja inmobiliaria se vio impulsada por una serie de factores interrelacionados. Por un lado, la disponibilidad de financiación fue facilitada por unas políticas de admisión de riesgos demasiado laxas en la Banca española. lo que incentivaba a las personas a adquirir viviendas sin considerar adecuadamente su capacidad de pago real, resultando en un endeudamiento excesivo de la población.

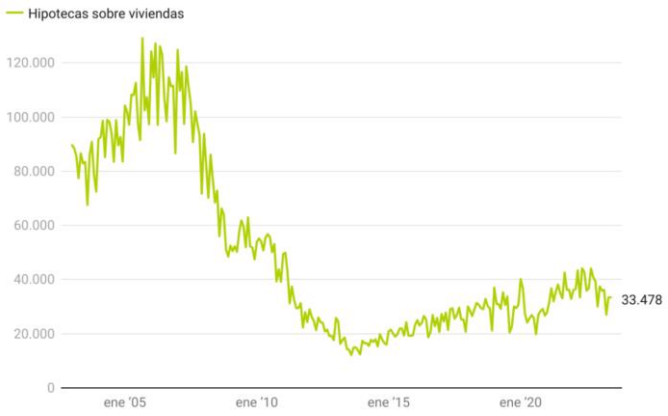


Ilustración 2-1 Evolución concesión de hipotecas en España. Fuente: idealista

Por otro lado, la especulación desempeñó un papel significativo en el aumento de los precios de la vivienda. Muchos inversores adquirieron propiedades con la intención de venderlas a un precio más elevado en el futuro, lo que alimentó aún más la demanda y elevó los precios.

Ilustración 2-2 Evolución precio del m2 de la vivienda en España. Fuente: Diario La Moncloa

Además, la construcción excesiva de viviendas, impulsada por una demanda creciente y la disponibilidad de crédito, condujo a la sobreoferta en el mercado. Se edificaron más viviendas de las necesarias, exacerbando aún más la situación y generando un desequilibrio entre la oferta y la demanda.

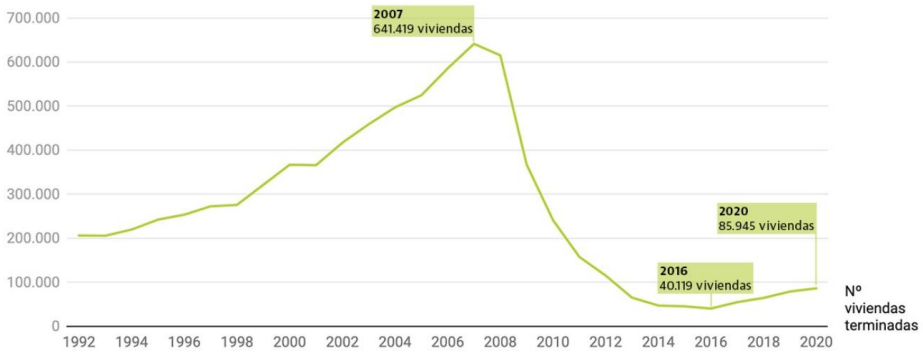


Ilustración 2-3 Evolución de la vivienda nueva terminada. Fuente: Idealista

Paralelamente, la economía global estaba en decadencia debido a la crisis financiera mundial desencadenada por la quiebra del banco estadounidense Lehman Brothers, Esta quiebra fue el resultado de su elevada exposición al mercado de alto riesgo, su dependencia de derivados financieros complejos y su falta de liquidez. Este suceso, ocurrido en septiembre de 2008, marcó el inicio de una reacción en cadena en los mercados financieros globales que a su vez coincidió con el estallido de la burbuja inmobiliaria española.

La segunda etapa, transcurre desde 2007 hasta 2014, caracterizándose por la crisis consecuente al clímax inmobiliario, focalizada en el sector de la construcción y también en el sector bancario que financió estos proyectos. El sector experimentó un ajuste severo, marcado por la contracción de la actividad y las dificultades financieras. Un ejemplo claro de ello fue la caída del precio de la vivienda, que vivió una caída acumulada del 53.3% durante este periodo (Fortuño, 2017).

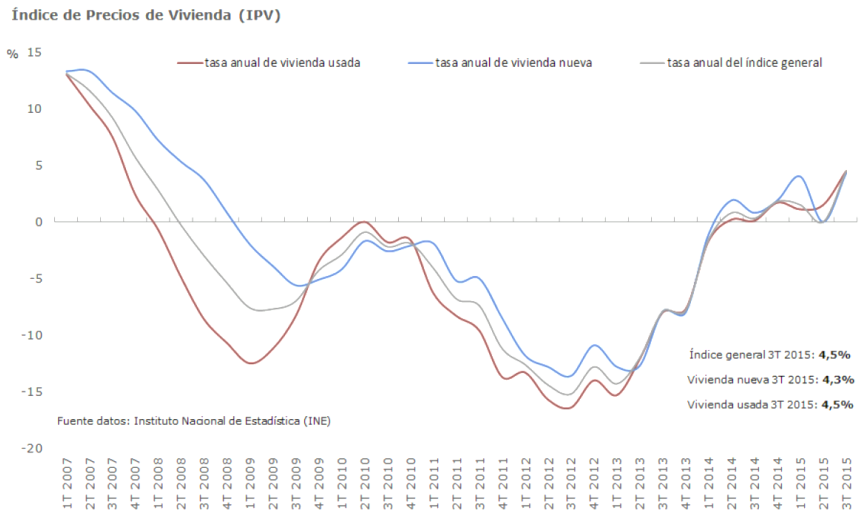


Ilustración 2-4 Evolución del IPV en España. Fuente: INE

La tercera etapa se inicia 2014, el mercado inmobiliario español comenzó a mostrar signos de recuperación después de años de crisis. Se registró un aumento en las transacciones de compraventa de viviendas, indicando una estabilización gradual en el sector.

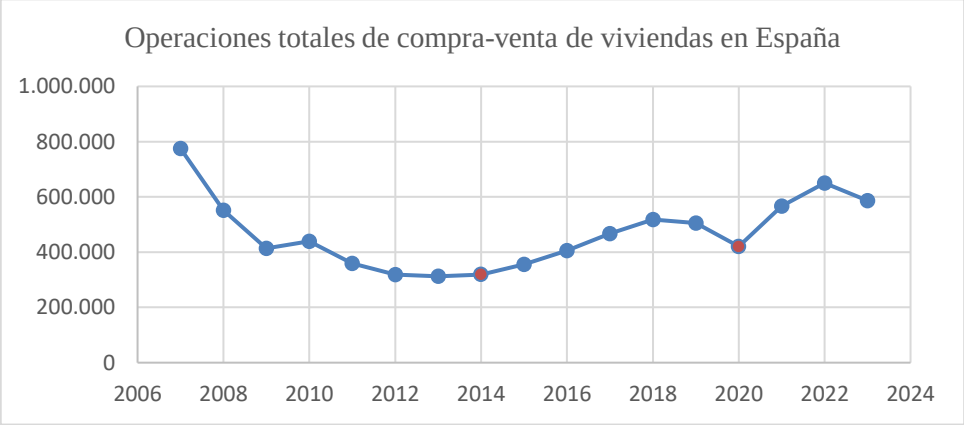


Ilustración 2-5 Operaciones de compra-venta de viviendas en España. Fuente: INE

En esta etapa, España emergió como un destino sumamente atractivo para inversores extranjeros, seducidos por sus precios competitivos y su clima favorable. Este interés se focalizó especialmente en el arco mediterráneo y los archipiélagos, donde las adquisiciones inmobiliarias extranjeras fueron más prominentes. Como se puede ver en las figuras, las adquisiciones realizadas por extranjeros constituyen una parte significativa del total.

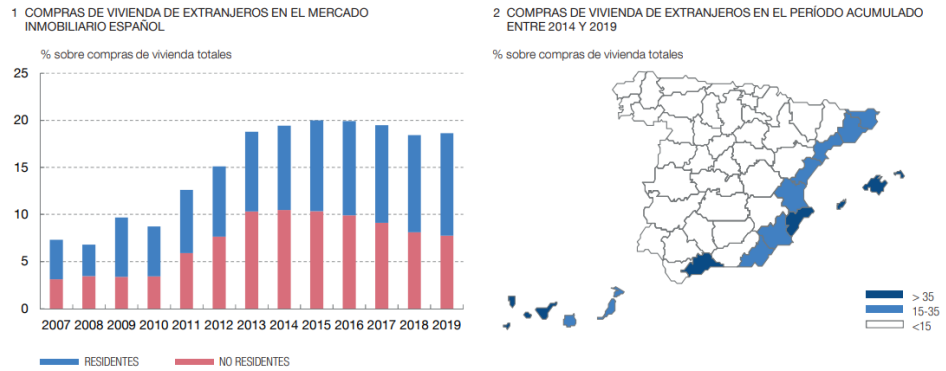


Ilustración 2-6 Participación de los extranjeros en el mercado inmobiliario residencial español. Fuente: BDE y Consejo General del Notariado

Por último, las entidades bancarias implementaron reformas para sanear sus balances y recuperar la confianza de los inversores. Además, el gobierno introdujo medidas para estimular el mercado.

En consecuencia, desde 2014 hasta la fecha, el mercado inmobiliario en España ha experimentado una recuperación gradual, aunque con algunos altibajos. Por ejemplo, es importante destacar que el mercado sufrió una caída en 2020 debido a la pandemia de COVID-19. Las medidas de confinamiento, la restricción de la movilidad y la incertidumbre económica afectaron negativamente a la demanda de vivienda.

En cuanto a las perspectivas de crecimiento para 2024-2025, son moderadamente positivas, según (Garriga, 2024) se prevé un incremento del precio de la vivienda del 2,7% y del 2,5% respectivamente, y que el número de compraventas alcance en torno a las 550.000 unidades anuales. Estas proyecciones respaldan la expectativa de que el BCE baje los tipos antes del verano y a que el desempeño de la economía español ha sido mejor del pronosticado en 2023, donde el PIB español creció un 2,5% cuando se pronosticaba un 1%.

2.2 MACHINE LEARNING

El *Machine Learning*, también conocido como aprendizaje automático, constituye una rama esencial dentro del campo de la inteligencia artificial, permitiendo que las máquinas aprendan y evolucionen sin una programación explícita (BBVA, 2019). Esta capacidad fundamental posibilita que los sistemas identifiquen patrones en los datos y realicen predicciones con precisión.

En términos generales, se basa en la utilización de datos y algoritmos para imitar el proceso de aprendizaje humano, mejorando progresivamente su precisión a medida que se le suministra más información. Esta disciplina se ha consolidado como un componente esencial dentro del ámbito de la ciencia de datos, donde se emplea para entrenar algoritmos y llevar a cabo diversas tareas como clasificaciones, predicciones y análisis de datos.

Hoy en día, el *Machine Learning* desempeña un papel crucial en la transformación digital y la toma de decisiones en una amplia gama de campos, desde la medicina y la ingeniería hasta las finanzas y el marketing. Su capacidad para extraer conocimiento significativo de conjuntos de datos complejos lo convierte en una herramienta de alto valor para entender el mundo que nos rodea y optimizar procesos en diversas industrias.

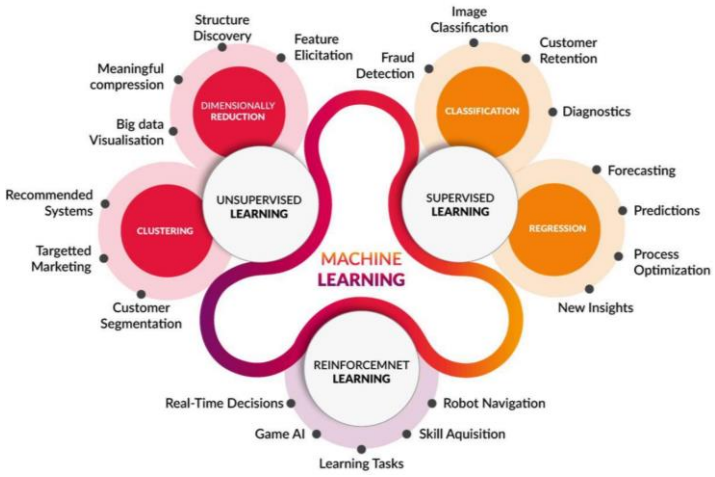


Ilustración 2-7 Ecosistema Machine Learning. Fuente: (Aizpun, 2020)

Como se ve en la figura, el ML puede ser dividido en tres enfoques fundamentales:

1. **Aprendizaje supervisado:** en este enfoque, los datos serán entrenados haciendo uso de un conjunto de datos etiquetado, donde cada ejemplo de entrada estará asociado con una etiqueta de salida correspondiente. El objetivo será enseñar una función que sea capaz de mapear las entradas a las salidas correctas.
    a. Suele ser utilizado en tareas de clasificación y regresión. Un ejemplo de aplicación podría ser la detección de fraudes bancarios o la predicción de la demanda de electricidad en un determinado momento.
2. **Aprendizaje no supervisado:** en este caso, a la hora de entrenar el modelo, los datos no estarán etiquetados, es decir, no se proporcionarán las salidas esperadas. En lugar de intentar mapear como en el caso supervisado, el objetivo será encontrar patrones o estructuras inherentes en los datos.
    a. Los algoritmos de aprendizaje no supervisado se utilizan en tareas como la segmentación de clientes o la personalización de contenido para el usuario.
3. **Aprendizaje por refuerzo:** por último, el *reinforcement learning* implica que un agente aprenda a tomar decisiones secuenciales para así maximizar una recompensa acumulada en un entorno particular.
    a. Es utilizado en problemas de toma de decisiones secuenciales, como el control de robots o la optimización de carteras de inversión.

En este proyecto, se emplearán técnicas de aprendizaje supervisado para solucionar un problema de regresión, el de la predicción de precios de las viviendas. Además, se integrará el aprendizaje profundo (*Deep Learning*), una subdisciplina del ML centrada en el entrenamiento de redes neuronales para lograr el aprendizaje de características complejas y abstractas de los datos.

2.2.1 MACHINE LEARNING ORIENTADO A PREDICCIÓN DE PRECIOS DE VIVIENDA

La predicción de precios de vivienda se ha convertido en un campo de investigación activo con aplicaciones en el mercado inmobiliario, finanzas y economía. En este ámbito, el *Machine Learning* ha demostrado ser una herramienta poderosa gracias a su capacidad para modelar relaciones complejas entre variables, permitiendo realizar predicciones más precisas y confiables del precio de las viviendas.

Los algoritmos de aprendizaje automático están siendo cada vez más utilizados para la evaluación masiva de bienes raíces y en modelos de valoración automatizados. Esta tendencia surge en respuesta a la necesidad de mejorar la precisión y eficiencia de las evaluaciones, especialmente frente a las limitaciones de las técnicas tradicionales (Grover, 2016).

Mientras que en las evaluaciones masivas usuales se emplean procedimientos estandarizados basados en reglas hedónicas[1], como se ha observado en proyectos como (Poeta, T.Gerhardt, & Gonzalez, 2019), donde se recopilan datos de ofertas de bienes raíces para estimar el valor de grandes grupos de propiedades, el ML ofrece la capacidad de modelar relaciones complejas entre variables, permitiendo así realizar predicciones más precisas y confiables del precio de las viviendas.

La ventaja de utilizar estos sistemas radica en su capacidad para realizar un gran número de valoraciones a un bajo coste por valoración y en un corto período de tiempo. Estas valoraciones son ampliamente empleadas en diversos contextos, tales como el cálculo de impuestos sobre bienes inmuebles, la evaluación del riesgo hipotecario o la gestión de carteras de inversión.

[1] Una regla hedónica descompone el precio de un bien en características individuales y específicas para poder estimar su valor. Cada característica contribuye de manera única y la regla hedónica cuantifica estas contribuciones.

En los últimos años, se han llevado a cabo numerosos proyectos de investigación y aplicaciones prácticas que han demostrado el potencial de estas técnicas para predecir los precios de vivienda con mayor precisión.

En ese sentido, Y. Wang (Wang Y. , 2022) en su estudio comparó seis modelos de predicción de precios de viviendas mediante técnicas de aprendizaje automático. Estos modelos incluyeron regresión lineal, árbol de decisiones, bosque aleatorio, aumento de gradiente, SVM y redes neuronales. Tras un análisis exhaustivo, Wang concluyó que el SVM [2]con función de *kernel* RBF se destacó como el modelo más eficaz para predecir los precios de las viviendas, particularmente basado en los datos de precios de viviendas en Boston.

Por otro lado, B. Sivasankar y colaboradores (Sivansankar, 2020) utilizaron datos relacionados con las ventas de viviendas para estimar los precios de estas basándose en un conjunto de datos del mundo real, IA. Se trata de un conjunto de datos público de una región específica en Estados Unidos. Integraron algoritmos de regresión Ridge, regresión Lasso y regresión XGBoost, logrando un error RMSE de 0.12, lo cual funcionó de manera óptima.

En el ámbito de los modelos de *ensemble*[3], S. Kulkarni (Julkarni, 2021) llevó a cabo una investigación centrada en su aplicación. Se encontró que la combinación ponderada de los algoritmos de regresión lineal, KNN y árbol de decisión presentaba mucho mejores resultados en términos de precisión (84%) y MAPE (16,09%) que los algoritmos utilizados por separado.

En el estudio realizado por F. Wang y colaboradores (Wang, Zou, Zhang, & Shi, 2019), analizan la complejidad y no linealidad de los factores influyentes en el precio de las viviendas, así como el crecimiento exponencial de los datos del mercado inmobiliario. Se propone un modelo de predicción de precios de vivienda basado en técnicas de *Deep*

[2] SVM es el acrónimo de Support Vector Machines, un algoritmo de ML que busca encontrar el hiperplano óptimo que se ajuste a la variable predictiva o que pueda dividir entre las clases de esta.
[3] Los modelos de ensemble consisten en la combinación de múltiples modelos sencillos para mejorar la robustez de las predicciones.

Learning y *forecasting* como ARIMA [4]para abordar estas dificultades. Los resultados muestran que su enfoque propuesto supera a los modelos tradicionales en términos de precisión y capacidad para predecir la tendencia del precio de las viviendas.

En (Das, Ali, Yuan-Fang Li, Kang, & Sellis, 2020) S. Das y colaboradores demuestran que el *embedding* de redes geoespaciales mejoró las predicciones de precios de viviendas. Representaron las relaciones espaciales de las propiedades utilizando datos inmobiliarios de Melbourne y *embedding* espacial. Su método superó al aprendizaje automático tradicional en la predicción de precios de viviendas.

[4] ARIMA es el acrónimo de AutoRegressive Integrated Moving Average, es un modelo estadístico utilizado para hacer predicciones en el análisis de series temporales.

Capítulo 3. DISEÑO

3.1 EXTRACCIÓN DE DATOS

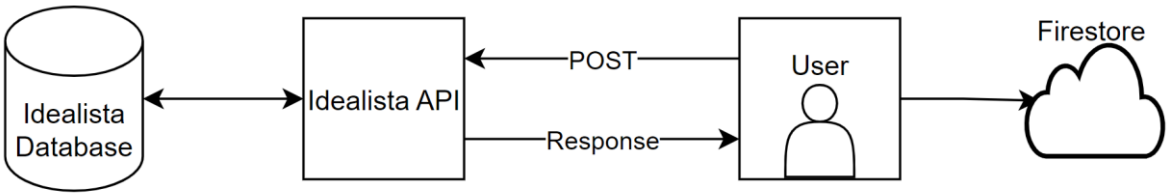


Ilustración 3-1 Proceso de obtención y almacenamiento de datos. Fuente: elaboración propia

El diagrama representa el flujo de obtención y almacenamiento de datos para su posterior uso. Para lograr este pipeline, se han seguido los siguientes pasos:

1) Solicitud de Acceso a la API

El proceso comienza con la solicitud de acceso a la API de Idealista, un servicio que proporciona información detallada sobre propiedades inmobiliarias, incluyendo precios, ubicaciones y características. Esta solicitud implica registrarse en Idealista y obtener credenciales de API, como una clave API.

2) Obtención de un Token Bearer

Una vez obtenida la clave API, se realiza una llamada a la API de Idealista para solicitar un token Bearer. Este token funciona como una credencial de acceso segura que se utiliza para autenticar y autorizar las solicitudes a la API. Es esencialmente una forma de identificación que permite a nuestro sistema interactuar con la API de Idealista de manera segura y controlada.

3) Realización de Solicitudes a la API de Idealista

Con el token Bearer obtenido, podemos realizar solicitudes a la API de Idealista [5]para obtener datos sobre propiedades inmobiliarias. Estas solicitudes pueden incluir una variedad de parámetros, como el tipo de operación (venta o alquiler), la ubicación y otros criterios de búsqueda específicos.

4) Manejo de la respuesta en Formato JSON

La API de Idealista responderá a nuestras solicitudes con datos estructurados en formato JSON (JavaScript Object Notation). Este formato es ampliamente utilizado para el intercambio de datos y es altamente legible tanto para humanos como para las máquinas. La respuesta generalmente incluirá una lista de elementos, cada uno representando una vivienda con detalles como el precio, la ubicación, el tamaño y otras características relevantes.

5) Almacenamiento de Datos en Firebase

Una vez que hemos recibido la respuesta de la API de Idealista en formato JSON, procedemos a procesar estos datos y almacenarlos en Firebase, una plataforma de desarrollo de aplicaciones móviles y web que ofrece una base de datos en la nube en tiempo real. El almacenamiento de los datos en este entorno facilita su acceso y asegura la unicidad, preservando la versión original de los datos.

[5] API que permite el acceso a los datos del portal inmobiliario. Se debe solicitar acceso para obtener las claves pertinentes para su uso aquí .La documentación correspondiente se encuentra en el siguiente repositorio de Idealista.

3.1.1 SOLICITUDES A LA API: PARÁMETROS Y ESPECIFICACIONES

Las llamadas a la API de Idealista se han realizado utilizando Python con la librería *requests*, aunque aquí se muestra el proceso en Postman [6] para mayor claridad.

- ✓ Los parámetros elegidos para las solicitudes incluyen propiedades ubicadas en el centro de Madrid, dentro de un radio de 3000 metros desde el punto central, publicadas en los últimos 2 años y ordenadas por distancia descendente.
- ✓ Dado el límite de 50 elementos por solicitud establecido por Idealista, se realizaron múltiples llamadas variando el número de página para obtener todos los resultados.

Este enfoque simula la experiencia de un usuario que busca propiedades en Idealista, pero en lugar de hacer *web scraping*[7], se utiliza la API oficial para acceder a los datos de manera estructurada y segura. La respuesta a esta llamada será una lista de viviendas que cumplen con estos requisitos, especificando para cada una todos sus atributos.

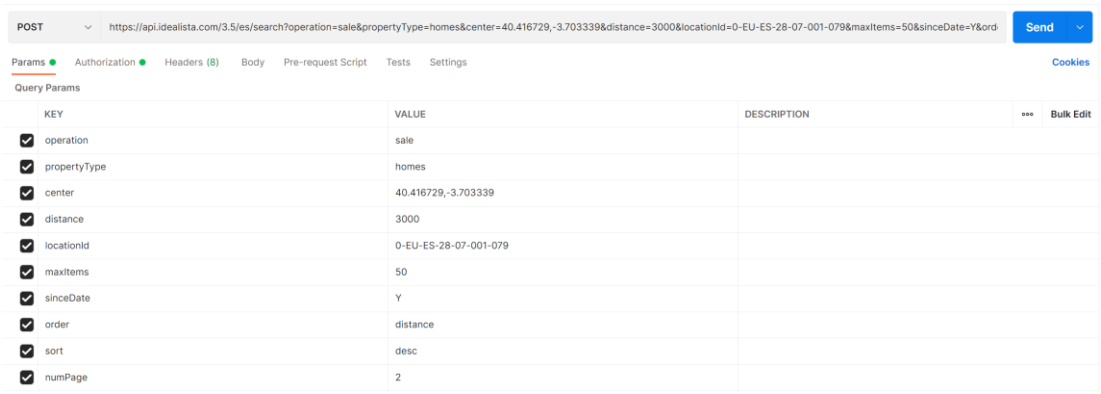


Ilustración 3-2 Ejemplo llamada API idealista

[6] Postman es una herramienta de desarrollo de API que permite realizar y probar solicitudes HTTP facilitando la interacción con la API y la visualización de las respuestas.

[7] El web scraping o raspado de datos consiste en la extracción de datos de páginas web de forma automatizada mediante el uso de scripts para posteriormente exportarla en un formato más útil para el usuario. Al aplicar esta técnica deben tenerse en cuenta ciertas consideraciones legales, ya que únicamente está permitido cuando los datos que se extraen están disponibles públicamente y no se recojan datos personales sin consentimiento previo.

3.1.2 ALMACENAMIENTO EN LA NUBE: FIRESTORE

La respuesta de la API de Idealista, estructurada en formato JSON, fue procesada y almacenada en Firebase Firestore. Para lograrlo, se siguieron los siguientes pasos:

- ✓ Creación de un proyecto en Firebase, estableciendo una base de datos Firestore y una colección para almacenar los datos.
- ✓ Inicialización de Firebase en Python utilizando las credenciales del SDK correspondiente (el SDK de Firebase es un conjunto de bibliotecas, APIs y herramientas que permiten interactuar con los servicios de Firebase desde una aplicación)
- ✓ Se desarrolló un script para cargar los datos en Firestore, agregando documentos a la colección creada en el paso anterior.

Finalmente, logramos obtener una colección como la representada en la figura, donde cada vivienda está representada por un documento único identificado por un ID único, y donde se encuentran almacenados los diferentes atributos correspondientes.

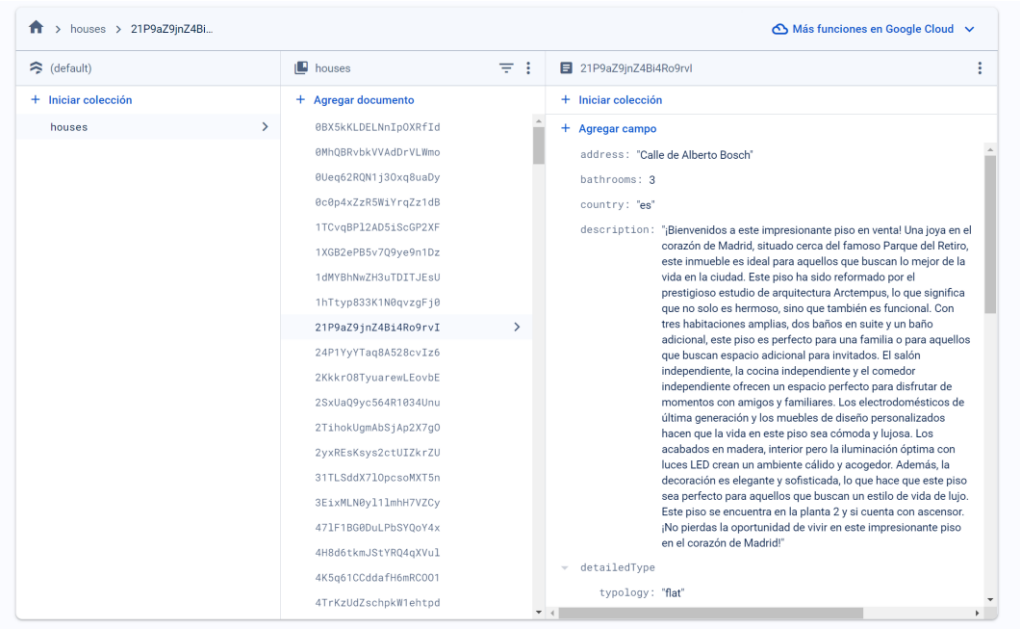


Ilustración 3-3 Ejemplo formato bbdd Firebase

3.2 *ANÁLISIS EXPLORATORIO DE LOS DATOS*

3.2.1 PREPROCESADO

Tras la recolección de los datos, se realizó una selección cuidadosa de las variables, eliminando aquellas que no aportaban valor significativo a la predicción del precio de la vivienda, como identificaciones específicas de la plataforma de Idealista (Disponibilidad de tour 360, url, número de fotos) y detalles específicos como la dirección. Además, se simplificó la información relacionada con el estacionamiento y la tipología de la vivienda mediante la creación de nuevas columnas.

Para garantizar la integridad de los datos, se eliminaron duplicados utilizando el property id como referencia. Este paso fue crucial para evitar la superposición de registros que podrían haber surgido de múltiples consultas a la API.

Finalmente, tras eliminar filas a las que les faltaban datos, se obtuvo un conjunto de datos con 815 entradas y las siguientes columnas:

Columna	Tipo de variable	Descripción
hasLift	Boolean	Disponibilidad de ascensor
Longitude	Numérica	Coordenadas de longitud
Latitude	Numérica	Coordenadas de latitud
Distance	Numérica	Distancia desde el punto marcado como centro
Status	Categórica	Estado de la vivienda (nueva construcción, bueno, para reformar)
Rooms	Numérica	Número de habitaciones
Bathrooms	Numérica	Número de baños
Size	Numérica	Tamaño de la vivienda en m2
Floor	Categórica	Planta en la que se encuentra la vivienda (valor numérico, bajo, entrada, sótano o semi-sótano)
newDevelopment	Boolean	Nueva construcción
Municipality	Categórica	Municipio
Price	Numérica	Precio de venta

Exterior	Binaria	Vivienda exterior
Neighborhood	Categórica	Vecindario
District	Categórica	Distrito
priceByArea	Numérica	Precio por m2
propertyCode	Categórica	Id de la vivienda
Country	Categórica	País
Province	Categórica	Provincia
Typology	Categoría	Tipo de vivienda (Piso, penthouse, dúplex, studio)
hasParking	Boolean	Parking incluido en el precio

Tabla 1 Variables del conjunto de datos

Nota: Es importante tener en cuenta que los datos recopilados a través de la API del portal Idealista pueden contener un cierto margen de error. Dado que estos datos son publicados por los usuarios, existe la posibilidad de que los precios registrados estén inflados debido a la inclusión de comisiones por parte del vendedor u otros factores.

3.2.2 VISUALIZACIONES

3.2.2.1 Análisis de la variable target

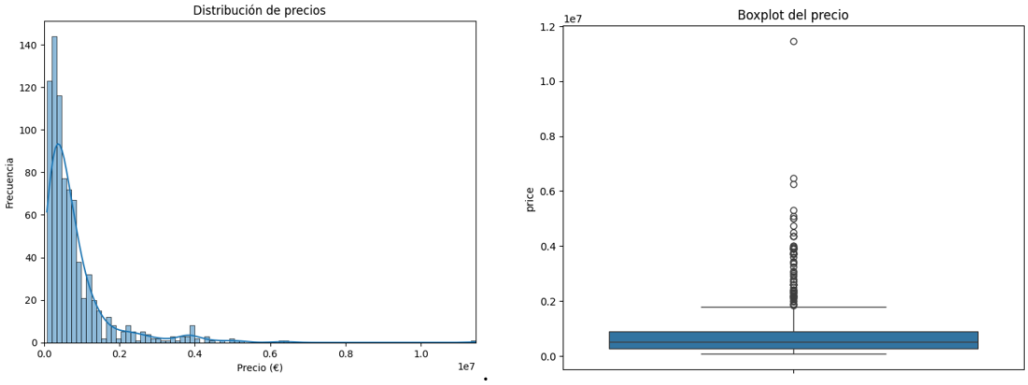


Ilustración 3-4 Distribución de la variable precio antes de eliminar outliers

Los resultados muestran una amplia variación en los precios de la vivienda, desde 78,000 hasta 11,450,000. La media es de aproximadamente 809,677 con una desviación estándar de alrededor de 962,201.

También, se decidió analizar las métricas de sesgo y curtosis para obtener más información sobre la distribución estudiada.

- ✓ El sesgo nos indicará cómo de desviada se encuentra nuestra distribución respecto a la distribución normal, por lo que un valor igual a 0 significaría una función idéntica a esta.
- ✓ La curtosis mide la concentración de datos en las colas de la distribución en comparación con una distribución normal.

Sesgo (Skewness)	3,84470
Curtosis	24,24486

El valor positivo obtenido para el sesgo, sugiere que la distribución está inclinada hacia precios más altos, mientras que una curtosis positiva y de alto valor 24.24 nos indica una

distribución con colas más pesadas y puntiagudas, sugiriendo una concentración notable de precios en los extremos.

También, se puede observar cómo estos datos crean lo que se conoce como una "long tail", donde algunos datos están considerablemente distantes del conjunto principal de valores, dando lugar a un efecto visual similar al de una cola.

Eliminación de outliers

Para obtener una representación más precisa de la distribución de los precios, se ha decidido eliminar los valores atípicos utilizando la regla de las tres sigmas. Esta regla implica eliminar aquellos valores que estén a más de tres desviaciones estándar de la media, ayudando así a eliminar valores extremadamente altos o bajos que puedan distorsionar la interpretación de los datos.

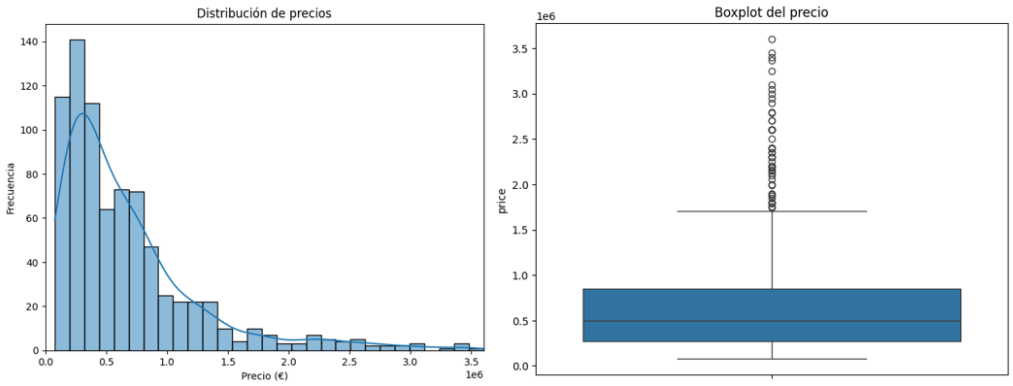


Ilustración 3-5 Distribución de la variable precio tras eliminar outliers

La eliminación de valores atípicos dio lugar a una distribución de datos menos sesgada y menos puntiaguda. Al estar más balanceada y menos influenciada por valores extremos será óptima para la creación de modelos de predicción más precisos.

Sesgo (Skewness)	2,0589803
Curtosis	4.8896854

3.2.2.2 Análisis de las variables explicativas

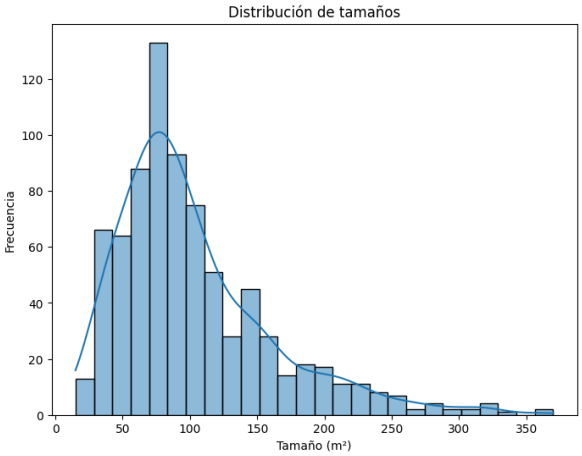


Ilustración 3-6 Distribución del tamaño de las viviendas

Como vemos en el histograma, la mayoría de las casas tienen un tamaño en torno a los 90-100 metros cuadrados, lo cual se alinea con el tamaño medio de la vivienda en el municipio de Madrid, que es de 94.4 metros cuadrados según datos del INE a fecha de 2021.

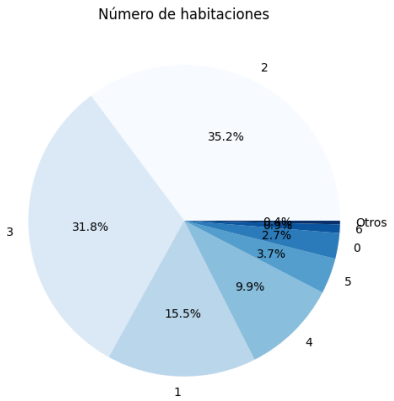


Ilustración 3-7 Distribución de la variable nº de habitaciones

El gráfico de pastel muestra la distribución del número de habitaciones en las viviendas. Mayormente, las viviendas de 90-100 metros cuadrados tienen 2 o 3 habitaciones, lo que se alinea con el tamaño medio de las casas en esa categoría. Esto sugiere una relación entre el tamaño de la vivienda y el número de habitaciones.

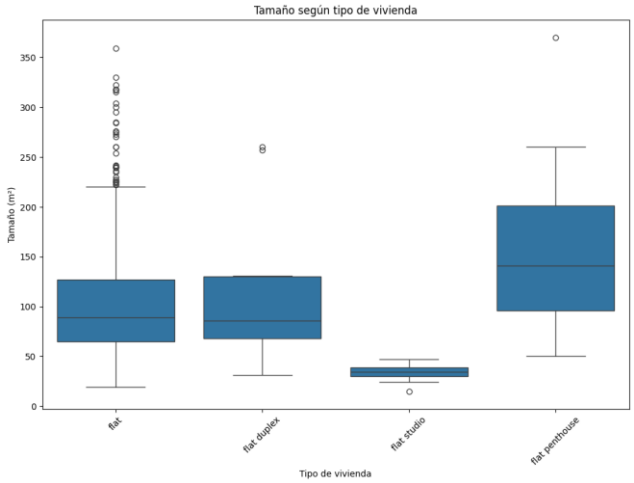


Ilustración 3-8 Tamaño de las viviendas según tipología

El gráfico muestra el tamaño de la vivienda según su tipo, ya sea piso, estudio, dúplex o ático. Podemos observar que el tamaño de cada tipo de vivienda se alinea de manera lógica con las expectativas comunes. Por ejemplo, los estudios tienden a ser más pequeños, mientras que los dúplex y los áticos suelen ser más espaciosos.

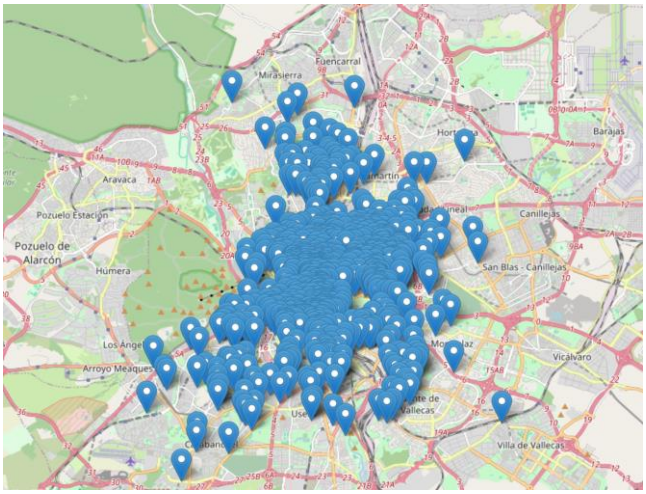


Ilustración 3-9 Ubicaciones de las viviendas analizadas

El estudio se centrará únicamente en la ciudad de Madrid, por lo que todas las viviendas analizadas estarán ubicadas dentro de sus límites como se ve en la representación anterior.

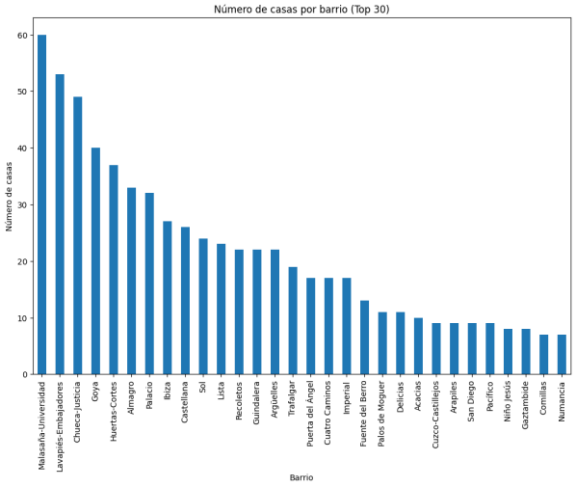


Ilustración 3-10 Número de casas por barrio

En cuanto al conjunto de datos recolectado, podemos ver en la ilustración anterior a qué vecindario pertenecen las observaciones.

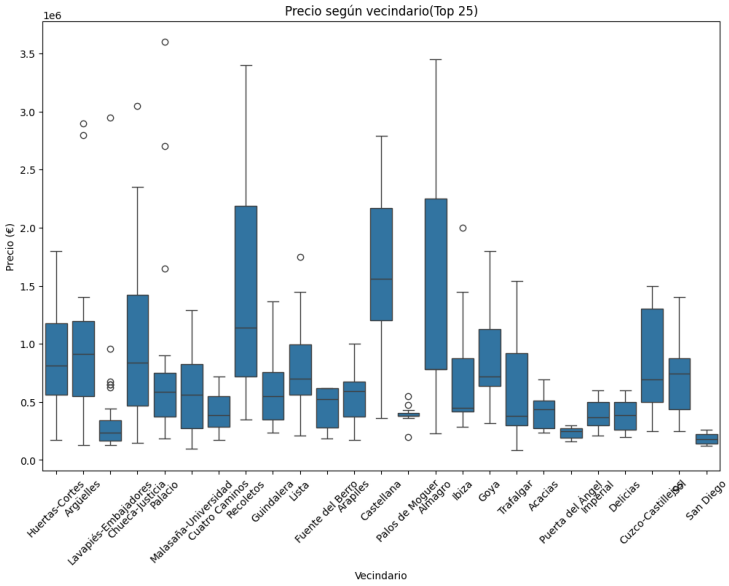


Ilustración 3-11 Variación de precio por vecindario

Como se puede observar, algunos vecindarios muestran un rango de precios más restringido que otros. En la mayoría de los casos, se encuentran viviendas cuyos precios se desvían notablemente de la tendencia del vecindario en cuestión, lo que podría considerarse como valores atípicos. No obstante, si podemos observar diferencias significativas en la

distribución de precios entre los distintos vecindarios. Por ejemplo, en vecindarios como Almagro, Castellana o Recoletos, los precios son generalmente más altos en comparación con otros, como Puerta del Ángel o Lavapiés, que se sitúan en el extremo inferior de la escala de precios. (Para elaborar la gráfica, se seleccionaron los 25 vecindarios con mayor cantidad de viviendas disponibles).

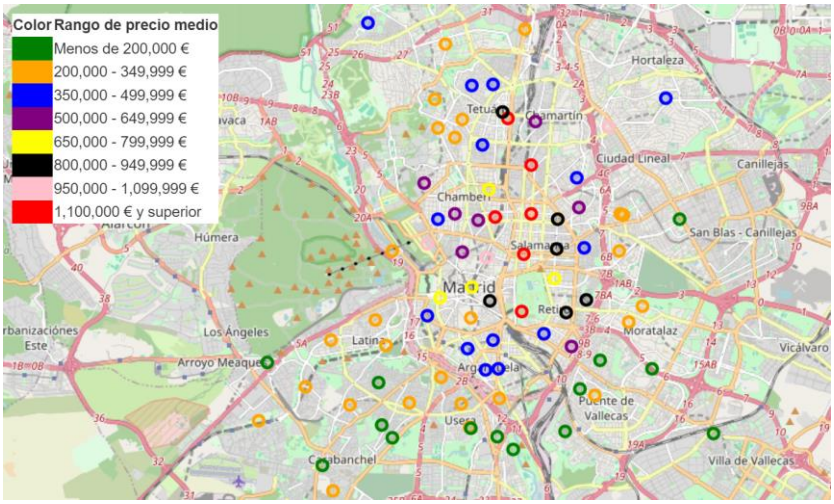


Ilustración 3-12 Mapa de precios por barrio

Por otro lado, se ha creado un mapa donde los barrios se clasifican según rangos de precio.

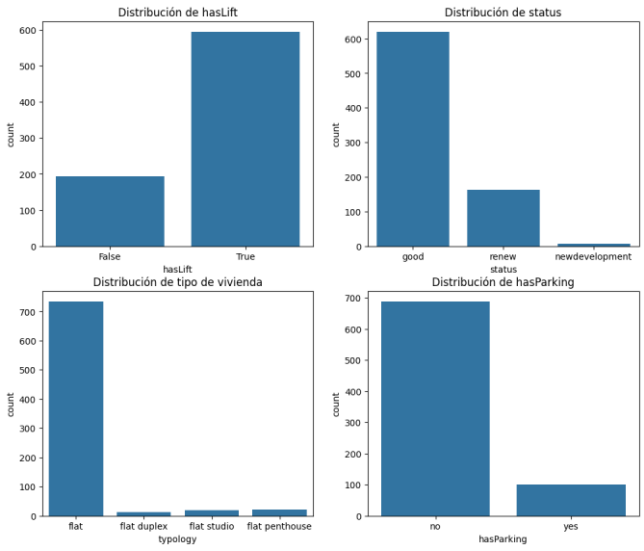


Ilustración 3-13 Distribución de variables categóricas

Se examinaron las variables categóricas y binarias en la distribución, revelando que la mayoría de las viviendas en venta pertenecen a la categoría de "piso" y que aproximadamente el 75% de las viviendas cuentan con ascensor. Además, se observó que el precio en la mayoría de las ocasiones no incluye plaza de garaje. También, es notaria la escasa presencia de nuevos desarrollos inmobiliarios en Madrid.

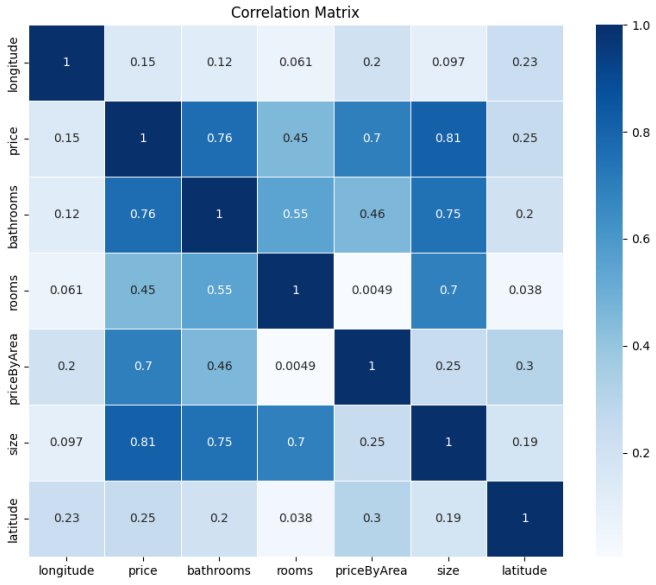


Ilustración 3-14 Matriz de correlación de variables numéricas

Por último, en la matriz de correlación podemos ver como las variables número de baños y habitaciones se encuentran altamente correlacionadas con el tamaño de la vivienda, como era de esperar. También, el tamaño y el precio están altamente correlacionados, lo que es coherente con la percepción común de que el tamaño de la vivienda es un factor importante a la hora de determinar su precio. Por otro lado, podemos apreciar la existencia de una correlación positiva más no tan fuerte, de la longitud y la latitud con el precio de la vivienda. Esto sugiere que ciertas ubicaciones geográficas podrán tener un impacto en el precio de la vivienda, lo cual puede estar vinculado a factores tales como la cercanía a servicios, centros comerciales o áreas con una alta demanda.

3.3 ESPECIFICACIONES DEL DISEÑO

3.3.1 PARTICIONAMIENTO DE LOS DATOS Y VALIDACIÓN CRUZADA

El conjunto de datos inicial se dividió en dos subconjuntos, entrenamiento y test, con 2/3 y 1/3 de los datos respectivamente. Esta división se hizo de forma aleatoria para garantizar la representatividad de ambos conjuntos.

Además, en aquellos algoritmos en los que es necesario encontrar los hiperparámetros[8] óptimos, se emplearon técnicas de validación cruzada. Esta técnica se utiliza para evaluar un modelo y consiste en dividir el conjunto de datos en k subconjuntos. Luego, el modelo se entrena k veces, utilizando k-1 subconjuntos como datos de entrenamiento y uno como datos de evaluación en cada iteración. Al promediar las k estimaciones del rendimiento del modelo, se obtiene una medida más generalizada de su desempeño.

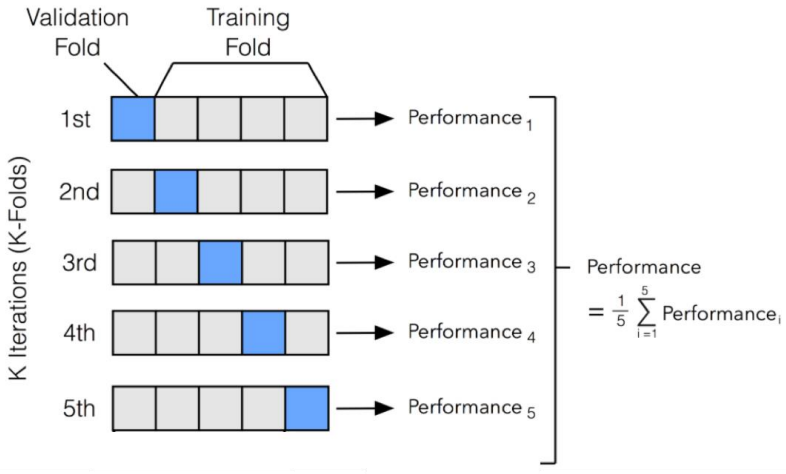


Ilustración 3-15 Funcionamiento k-folds Cross validation. Fuente: (Guerrero, 2021)

[8] Parámetros externos a los modelos de aprendizaje automático que influyen en su ajuste y rendimiento óptimo. No son aprendidos durante el entrenamiento del modelo, sino que deben ser definidos previamente.

3.3.2 LIBRERÍAS UTILIZADAS

El proyecto se desarrolló completamente en Python, utilizando para cada paso las librerías mostradas en la figura.

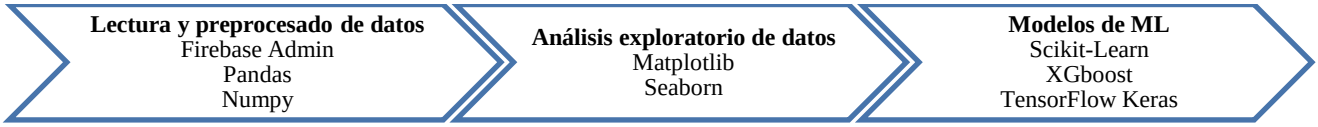


Ilustración 3-16 Librerías empleadas. Fuente: elaboración propia.

En la siguiente lista se proporciona una breve descripción de cada librería utilizada en el proyecto junto con enlaces a sus respectivas documentaciones.

- Pandas: manipulación y análisis de datos.
- Numpy: operaciones matemáticas y numéricas.
- Firebase Admin: gestión de datos en la nube.
- MatplotLib: visualización de datos.
- Seaborn: visualización de datos estadísticos.
- Scikit-Learn: implementación de algoritmos de ML y técnicas de cross-validation.
- XGBoost: implementación de algoritmos de gradient boosting.
- TensorFlow Keras: implementación de redes neuronales profundas.

En cuanto a las versiones específicas de las librerías, están especificadas en el archivo "requirements.txt" del código fuente del proyecto, donde se detallan todas las dependencias necesarias con sus versiones correspondientes.

Capítulo 4. ESTUDIO EMPÍRICO

4.1 ALGORITMOS DESARROLLADOS

En esta sección se explicarán los algoritmos desarrollados para la estimación de precios de las viviendas, que incluyen regresión lineal, algoritmos de boosting y bagging, así como redes neuronales. Cada algoritmo fue evaluado utilizando validación cruzada y métricas de evaluación apropiadas para la regresión explicadas en la sección 4.2 de este capítulo.

4.1.1 MODELO DE REGRESIÓN LINEAL

El modelo de regresión es uno de los métodos más simples y fundamentales en el análisis estadístico y el aprendizaje automático. Es empleado para describir una variable de respuesta continua como una combinación lineal de una o varias variables predictivas o explicativas (MathWorks, s.f.).

Cuando realizamos un modelo de regresión lineal, asumiremos que existe una relación lineal entre las variables predictoras y la variable objetivo, por lo que suponemos que los datos pueden ser aproximados por una recta en un espacio multidimensional.

La ecuación de un modelo de regresión lineal múltiple, con k variables explicativas, se presenta como:

$$Y = \beta_0 + \sum \beta_k X_K + \epsilon \qquad 4\text{-}1$$

Donde Y es la variable dependiente que se quiere predecir, β_0 es el término de intercepción, β_k son los parámetros o coeficientes lineales que se deben calcular para cada variable explicativa X_k y ϵ el término de error. Para definir dichos parámetros, se hace uso del método de mínimos cuadrados, el cual busca encontrar los valores de los coeficientes que

minimizan la suma de los cuadrados de los residuos. Es decir, la diferencia entre los valores observados de la variable dependiente y los valores predichos por el modelo.

Es importante recalcar que, al implementar un modelo de regresión lineal, estaremos haciendo varias suposiciones sobre los datos:

- Linealidad: asumiremos que la relación entre las variables predictoras y la variable dependiente es lineal.
- Independencia de los errores: el error asociado con una observación no estará correlacionado con el de otra observación.
- Homocedasticidad: asume que la varianza de los errores es constante, por lo que la distribución de los errores es considerada uniforme en todo el rango de predicción.
- Normalidad de los errores: los errores seguirán una distribución normal.
- Ausencia de multicolinealidad: no existe una relación lineal exacta entre las variables predictoras, si no que no están correlacionadas entre sí.

4.1.2 MODELOS DE ENSEMBLE

Los *ensemble methods*, son una técnica de aprendizaje automático que combina varios modelos base para producir un modelo predictivo óptimo (Lutis, 2017), y existen 3 métodos:

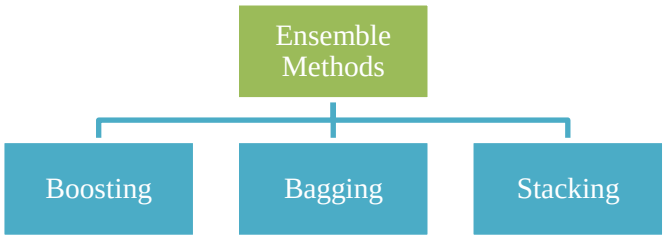


Ilustración 4-1 Clasificación de los métodos de ensamblaje. Fuente: elaboración propia

1. **Bagging:**

Consiste en construir múltiples modelos independientes utilizando muestras de datos obtenidas por remuestreo con remplazo (Bootstrap) del conjunto de datos de entrenamiento.

Cada modelo se entrenará en una submuestra diferente de los datos para luego contribuir al resultado total del modelo. En caso de regresión, se promediarán las estimaciones de cada uno de los árboles, en caso de clasificación se otorgará a la observación la clase más votada.

Los modelos más comunes dentro de la técnica de bagging incluyen principalmente Random Forest, Bagged SVM (Support Vector Machines) o Bagged Decision Trees.

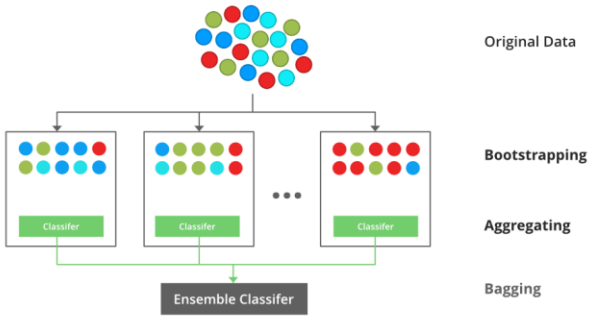


Ilustración 4-2 Funcionamiento de los métodos de Bagging. Fuente: (Geeks for Geeks, 2022)

2. **Boosting:**

Esta técnica de ensamblado consiste en entrenar secuencialmente modelos débiles[9], donde cada modelo se encargará de mejorar los errores que cometieron los modelos anteriores. Para ello, en cada iteración se intentará mejorar el rendimiento global del conjunto ajustando los pesos a las instancias, para dar más importancia a aquellas que estén mal clasificadas.

[9] Los modelos débiles son modelos simples o poco precisos que tienen un rendimiento ligeramente mejor que el modelo aleatorio, pero no son lo suficientemente complejos como para capturar la relación entre la entrada y la salida por si solos.

Los modelos más comunes dentro de la técnica de boosting incluyen AdaBoost (Adaptive Boosting), Gradient Boosting Machines (GBM), XGBoost (Extreme Gradient Boosting), LightGBM y CatBoost.

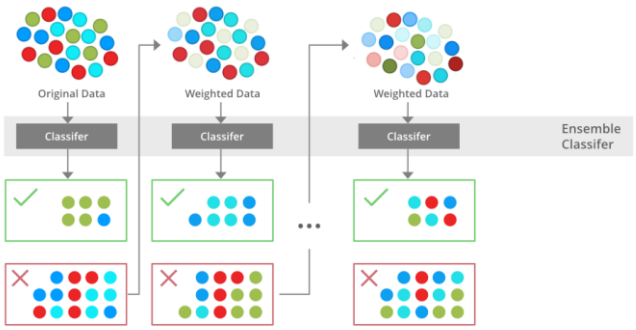


Ilustración 4-3 Funcionamiento de los métodos de Boosting. Fuente: (Geeks for Geeks, 2022)

3. Stacking:

Esta técnica combina las predicciones de muchos modelos base, conocidos como modelos de nivel 0, utilizando otro modelo de nivel 1(meta-modelo) .En lugar de simplemente promediar o votar las predicciones, el meta-modelo se entrena en las predicciones de los modelos base para aprender a combinarlas de manera óptima. Esta técnica es la más avanzada de todas, pero también la más costosa computacionalmente.

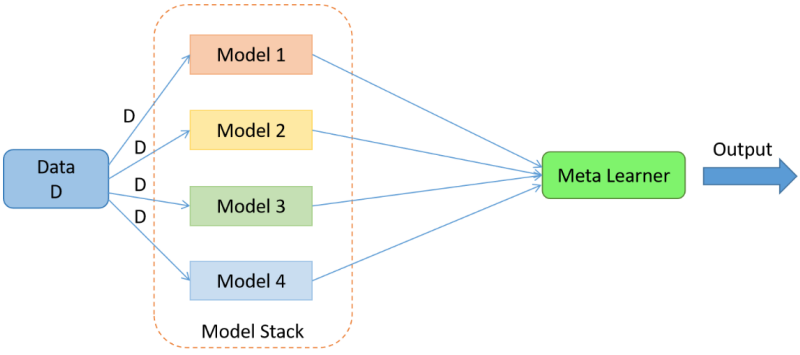


Ilustración 4-4 Funcionamiento de los métodos de stacking. Fuente: (Khandelwal, 2021)

4.1.2.1 RANDOM FOREST

Es una técnica de aprendizaje automático perteneciente a la familia de algoritmos de bagging, ya que está basada en el ensamblaje de árboles de decisión.

En este método, se construyen múltiples árboles de decisión independientes, cada uno entrenado en una submuestra aleatoria del conjunto de datos. Después de que cada árbol en el bosque ha realizado una predicción para un determinado punto de datos, se cuenta el voto de cada árbol. En el caso de la clasificación, la clase con más votos se considera la predicción final. En el caso de la regresión, se promedian las predicciones de todos los árboles para obtener un valor final. Esta combinación de predicciones de múltiples árboles ayuda a reducir el sesgo y la varianza del modelo, resultando en una predicción más precisa y robusta.

Sus ventajas incluyen la capacidad de manejar conjuntos de datos grandes y complejos, la reducción del riesgo de sobreajuste gracias a su naturaleza de ensamblaje, y la capacidad de proporcionar estimaciones de la importancia de las características para la predicción.

Descenso de Entropía o Ganancia de Información

Durante la construcción de cada árbol, se busca la división que maximice la reducción de entropía o la ganancia de información. La entropía es una medida de incertidumbre o caos en el conjunto de datos, y la ganancia de información representará la diferencia en entropía antes y después de una división. La fórmula de la entropía se expresa como:

$$Entropía(S) = -\sum_{i=1}^{c} p_i \log_2(p_i) \qquad 4\text{-}2$$

Donde:

- S representa el conjunto de datos sobre el que se calcula la entropía.
- c representa las clases en el conjunto S
- pi representa la proporción de puntos de datos que pertenecen a la clase i con respecto al número total de puntos de datos existentes en S.

Criterio de Gini

Una alternativa común al cálculo de la ganancia de información es el criterio de Gini. El criterio de Gini se basa en el índice de Gini, que mide la impureza de un conjunto de datos considerando la probabilidad de que un punto de datos seleccionado aleatoriamente sea clasificado incorrectamente si se clasifica al azar de acuerdo con la distribución de clases en el nodo.

En lugar de maximizar la ganancia de información, el criterio de Gini busca minimizar el índice de Gini en cada división. La fórmula de dicho índice se expresa como:

$$Gini(S) = 1 - \sum_{i=1}^{c} p_i^2 \qquad 4\text{-}3$$

Donde:

- S representa el conjunto de datos en un nodo dado.
- c es el número de clases en el conjunto de datos
- pi es la proporción de puntos de datos que pertenecen a la clase i, con respecto al número total de puntos de datos en el conjunto S.

La impureza de Gini y la entropía son fundamentales en la construcción de árboles de decisión para clasificación. Por otro lado, en el caso de problemas de regresión utilizando Random Forest, los árboles se construirán basándonos en métricas de error como el Error Cuadrático Medio (MSE) y el Error Medio Absoluto (MAE) para evaluar la calidad de las divisiones de los nodos en los árboles.

Hiperparámetros

El modelo ofrece varios hiperparámetros para ajustar su rendimiento. Tales como:

- n_estimators: Controla el número de árboles en el bosque.
- max_features: Determina el número máximo de características a considerar en cada división.
- max_depth: Limita la profundidad máxima de cada árbol en el bosque.
- min_samples_split: Especifica el número mínimo de puntos de datos requeridos para dividir un nodo interno.
- min_samples_leaf: Establece el número mínimo de muestras requeridas en un nodo hoja.
- bootstrap: Controla si se utiliza muestreo de arranque para entrenar cada árbol.
- criterion: Especifica la función para medir la calidad de una división, como "gini" o "entropy".

Para ajustar los hiperparámetros del modelo Random Forest, se emplean técnicas como la búsqueda aleatoria o en cuadrícula, donde se exploran diversas combinaciones de valores para cada parámetro. Estas estrategias buscan maximizar una métrica de evaluación, como la precisión o el área bajo la curva ROC, para determinar la configuración óptima del modelo.

4.1.2.2 XGBOOST

Es una técnica de aprendizaje automático perteneciente a la familia de algoritmos de boosting, ya que está basada en el ensamblaje de árboles de decisión

En los algoritmos de Gradient Boosting crea un conjunto secuencial de árboles de decisión débiles, donde cada árbol se construye de manera iterativa para corregir los errores cometidos por los árboles anteriores. La predicción global del modelo se calcula sumando las predicciones de cada árbol, pero cada una de estas contribuciones está ponderada por un factor que determina cuánto influye cada árbol en el resultado final, el learning rrate.

En concreto, XGBoost, o Extreme Gradient Boosting, es una implementación optimizada y eficiente del algoritmo de Gradient Boosting. Algunas de sus ventajas son:

- Utiliza técnicas avanzadas de regularización durante el entrenamiento para prevenir el sobreajuste y mejorar la generalización del modelo.
- Es más eficiente computacionalmente y es capaz de manejar conjuntos de datos de alta dimensionalidad.

Descenso de gradiente

Al crear nuevos modelos que corrijan los errores que cometieron los predecesores, lo que estamos haciendo es disminuir la función de pérdida. Como se puede ver en la figura, se irá en dirección opuesta del gradiente, que es la dirección en la que dicha función incrementa.

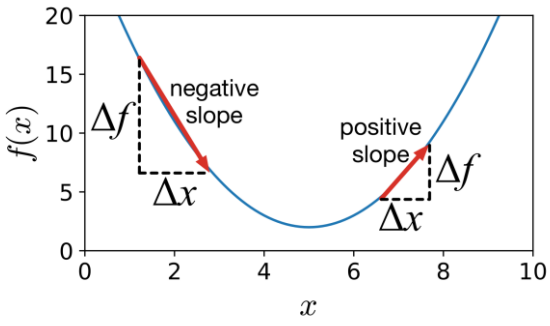


Ilustración 4-5 Descenso de gradiente. Fuente: (Parr & Howard, s.f.)

Técnicas de regularización

Las técnicas de regularización sirven para prevenir el sobreajuste y mejorar la generalización de los modelos. Dentro de los algoritmos de Boosting, son especialmente útiles, ya que al construir árboles que se enfocan en corregir los errores de los modelos anteriores, existe una tendencia inherente al sobreajuste de los datos.

Regularización L1, Lasso

$$Función\ de\ coste\ L1 = \sum_{i=0}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{M} |W_j| \qquad 4\text{-}4$$

La función de coste con regularización L1 se compone de dos términos que queremos minimizar:

- El primer término representa la función de pérdida, que mide la discrepancia entre los valores reales y_i y los predichos $\hat{y}_i$. Esta diferencia se eleva al cuadrado y se suma para todas las N observaciones, dando como resultado la suma de los errores cuadráticos.
- El segundo término es la regularización L1, que penaliza la magnitud absoluta de los pesos del modelo W_j. Esta penalización se controla mediante el parámetro λ que actúa como un factor de ponderación. La regularización L1 tiene el efecto de reducir los pesos de las características menos importantes, lo que promueve la simplicidad del modelo y previene el sobreajuste.

Regularización L2, Ridge

$$Función\ de\ coste\ L2 = \sum_{i=0}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^{M} {W_j}^2 \qquad 4\text{-}5$$

La función de coste con regularización L2 se compone de dos términos que queremos minimizar:

- Al igual que en la regularización L1, el primer término representa la función de pérdida.
- El segundo término es la regularización L2, que penaliza la magnitud al cuadrado de los pesos del modelo W_j. Esta penalización se controla mediante el parámetro λ que actúa como un factor de ponderación. La regularización L2 tiene el efecto de suavizar los pesos de las características, lo que promueve la generalización del modelo al evitar que los coeficientes se vuelvan excesivamente grandes.

La principal diferencia entre ambos métodos de regularización radica en cómo penalizan sus coeficientes.

- L1, penaliza la magnitud absoluta de los coeficientes, lo que resulta en una función de coste con esquinas en los ejes (Ver figura). Por ello, algunas características menos importantes se pueden reducir a cero, realizando así una selección de variables y dando lugar a modelos más dispersos.
- L2, penaliza la magnitud al cuadrado, con una función de coste suave y convexa en la que no hay puntos donde la solución óptima alcance exactamente cero para los coeficientes. Es utilizada para reducir la multicolinealidad entre variables y para aquellos casos en los que no queremos perder la contribución de ninguna variable en el modelo.

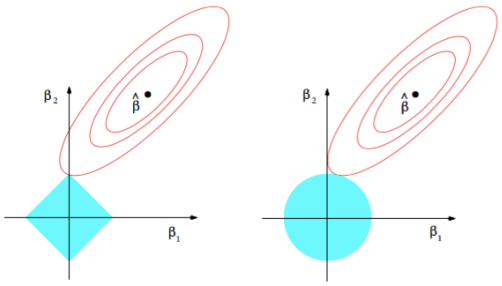


Ilustración 4-6 Problema de optimización (L1 derecha, L2 izquierda). Fuente: (C., 2018)

Hiperparámetros

El modelo ofrece varios hiperparámetros para ajustar su rendimiento. Tales como:

- n_estimators: Número de árboles a construir.
- max_depth: Profundidad máxima de cada árbol.
- learning_rate: Tasa de aprendizaje que controla la contribución de cada árbol.
- subsample: Proporción de muestras utilizadas para entrenar cada árbol.
- colsample_bytree: Proporción de características utilizadas para entrenar cada árbol.
- gamma: Mínima reducción de pérdida requerida para dividir un nodo.
- reg_alpha: Término de regularización L1 en pesos de hojas.
- reg_lambda: Término de regularización L2 en pesos de hojas.
- min_child_weight: Peso mínimo requerido en cada hoja del árbol.
- objective: Función de pérdida a optimizar durante el entrenamiento.
- eval_metric: Métrica de evaluación utilizada para monitorear el rendimiento del modelo durante el entrenamiento.

4.1.3 DEEP LEARNING

El aprendizaje profundo o *Deep Learning*, es una rama de la inteligencia artificial basada en modelos computacionales cuya estructura se asemeja a la del funcionamiento del cerebro humano, con redes neuronales. Estas redes cuentan con múltiples capas que a su vez tienen múltiples neuronas que procesan información de forma compleja, permitiendo a las redes neuronales aprender a partir de grandes cantidades de datos.

Algunos ejemplos de arquitecturas de redes neuronales incluyen las Perceptrón Multicapa (MLP), ideales para tareas de clasificación y regresión; las Redes Neuronales Convolucionales (CNN), especializadas en el procesamiento de imágenes; y los modelos basados en Transformers, fundamentales en el procesamiento del lenguaje natural.

4.1.3.1 Estructura básica de una red neuronal

- Capa de entrada: donde se ingresan los datos
- Bias: parámetro adicional añadido en cada neurona, representa el sesgo o desviación y permite que la red se adapte y aprenda cuando todas las entradas son 0.
- Capas ocultas: capas intermedias de neuronas que procesan la información
- Capa de salida: produce la salida de la red neuronal (regresión o clasificación)
- Conexiones entre neuronas: son las señales transmitidas que interconectan las capas.

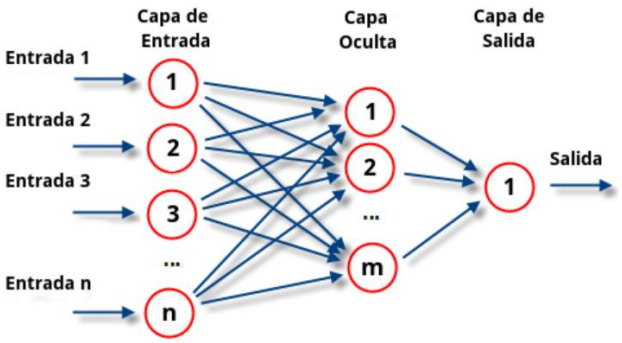


Ilustración 4-7 Estructura de una red neuronal. Fuente: (Atria Innovation, s.f.)

4.1.3.2 Funciones de pérdida

Estas funciones se utilizan para medir cómo de bien está funcionando un modelo, comparando sus predicciones con las etiquetas reales de los datos de entrenamiento. Al estar en un problema de regresión, utilizaremos el error cuadrático medio (MSE).

4.1.3.3 Optimizador

Se debe escoger un optimizador para ajustar los parámetros del modelo de aprendizaje automático con el objetivo de minimizar la función de pérdida. La elección del optimizador puede tener impacto significativo en la velocidad y eficiencia del entrenamiento del modelo, así como en su rendimiento y capacidad de generalización.

Gradient Descent:

Es un algoritmo de optimización que ajusta iterativamente los parámetros del modelo en la dirección opuesta al gradiente de la función de pérdida. Para ello, calcula el error promedio de todos los errores de la entrada y después desciende por la función objetivo hasta encontrar su mínimo.

Stochastic Gradient Descent:

Es una variante del descenso de gradiente que, a diferencia de este, calculará el gradiente utilizando una observación aleatoria. Esta técnica, proporciona una mayor velocidad de convergencia en conjuntos de datos grandes, pero puede causar cierta inestabilidad ya que al adaptar un único gradiente cada vez, puede llevar a oscilaciones en la convergencia.

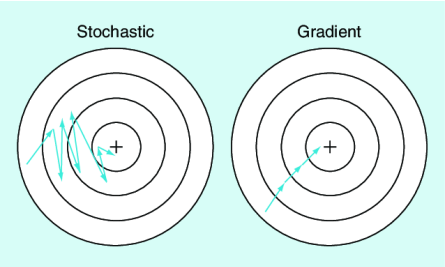


Ilustración 4-8 Descenso de gradiente. Fuente: (Balasubramanian, 2021)

4.1.3.4 Funciones de activación

Son las funciones matemáticas que se aplican a la salida de cada neurona de la red, introducen la no linealidad a la red, permitiendo aprender patrones complejos.

En la capa de salida de la red, los problemas de clasificación requieren de una función de activación para que la salida pueda interpretarse como probabilidades de pertenencia a una clase, como la función sigmoide para clasificación binaria o *Softmax* para clasificación multiclase.

Por el contrario, en problemas de regresión como el tratado en este trabajo, no se aplica una función de activación en la capa de salida, sino que la red produce una salida continua que representa directamente el valor numérico esperado.

ReLU

Rectified Linear Activation Function, es una de las funciones más utilizadas debido a su simplicidad y buen funcionamiento, permite aprender al modelo sin saturar su gradiente, lo que ayuda a mitigar problemas como el del desvanecimiento del gradiente.

$$ReLU(x) = \max(0, x) \qquad 4\text{-}6$$

Sigmoide

La función sigmoide o logística, se utiliza principalmente en clasificación binaria para mapear valores de entrada a un rango de salida entre 0 y 1, representando la probabilidad de que la entrada pertenezca a la clase positiva. Uno de los problemas de esta función es que a medida que la entrada se aleja de 0, la salida tiende a 0 o a 1, lo cual puede ocasionar problemas de desvanecimiento del gradiente.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \qquad 4\text{-}7$$

4.2 MÉTRICAS

Para evaluar la precisión de la predicción realizada, es esencial establecer un conjunto de métricas que pueden variar según el tipo de modelo en desarrollo (Raj, s.f.).

4.2.1 MEAN SQUARED ERROR (MSE)

Esta métrica mide la diferencia promedio entre los valores predichos y los valores reales, al elevar al cuadrado, se penalizará más a los errores grandes que a los pequeños.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad 4\text{-}8$$

4.2.2 ROOT MEAN SQUARED ERROR (RMSE)

Es la raíz cuadrada del MSE, su utilidad recae en que se expresa en las mismas unidades que la variable que se quiere predecir, lo que facilita su interpretabilidad.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \qquad 4\text{-}9$$

4.2.3 MEAN ABSOLUTE ERROR (MAE)

Mide la diferencia promedio entre los valores predichos y los reales, sin tener en cuenta el signo. Debido a que los errores se penalizan de igual forma (no eleva al cuadrado), es más sensible a los valores atípicos que el MSE.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \qquad 4\text{-}10$$

4.2.4 R-CUADRADO (R^2)

El coeficiente de determinación R cuadrado nos indica la proporción de la variabilidad de la variable dependiente que es capaz de explicar nuestro modelo. Un valor cercano a 1 indicará que el modelo se ajusta bien a los datos.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2} \quad \textit{4-11}$$

4.2.5 MEAN AVERAGE PERCENTAGE ERROR (MAPE)

El MAPE es una métrica relativa que nos indica, en promedio, qué tan lejos están las predicciones hechas por el modelo de los valores reales, expresado como un porcentaje.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad \textit{4-12}$$

Al analizar las métricas de rendimiento del modelo de regresión lineal, nos enfocaremos especialmente en métricas que están en la misma unidad que la variable objetivo, como el error absoluto medio (MAE) y la raíz del error cuadrático medio (RMSE). Estas métricas proporcionan una comprensión directa de la discrepancia entre los valores predichos y reales, lo que permite una evaluación más precisa del desempeño del modelo en términos de su capacidad para hacer predicciones precisas y útiles.

4.3 RESULTADOS

4.3.1 REGRESIÓN LINEAL MÚLTIPLE

Para este algoritmo, decidimos utilizar únicamente las variables numéricas del *dataset*, dejando fuera las categóricas y booleanas. Esto se debe a que la regresión lineal es más adecuada para variables numéricas continuas, ya que busca establecer una relación lineal entre las características de entrada y la variable de salida.

- Variables explicativas: longitud, latitud, número de baños, número de habitaciones, precio m2 por área y tamaño.
- Variable predictiva: precio de la vivienda.

4.3.1.1 Multicolinealidad

Antes de entrenar el modelo, se decidió analizar los coeficientes VIF (*Variance Inflation Factors*), ya que son una medida crucial en análisis de regresión. Indican la presencia de multicolinealidad entre variables predictoras que puede distorsionar los resultados.

Variable	Coeficiente
Constante	6.347466e06
Longitud	1.086207
Latitud	1.138794
Nº baños	3.513668
Nº habitaciones	2.537789
Precio por área	1.680564
Tamaño	3.534612

Tabla 2 Coeficientes VIF del modelo de regresión lineal

Como se puede observar en la tabla, todos los coeficientes VIF de nuestras variables predictoras se sitúan por debajo de 5, lo que indica que no existe una preocupante multicolinealidad entre ellas. Es importante destacar que el coeficiente para la constante no influye en esta evaluación, ya que no representa una variable predictora en el modelo de regresión.

4.3.1.2 Métricas y análisis de residuos

MSE	RMSE	MAE	R2	MAPE
1.4566996e+11	381706.109244	197125.187042	0.894891	0.398321

Tabla 3 Métricas del modelo de regresión lineal

Para el MSE y RMSE se obtuvieron valores muy altos, esto está relacionado con el cálculo de dichas métricas que, al elevar al cuadrado los errores, hacen que los valores altos influyan desproporcionalmente al valor final.

En el análisis de la variable target "precio" se obtuvo una varianza alta, además de ser una distribución con bastantes puntos dispersos, por lo que el MSE y consecuentemente el RMSE, pueden ser indicadores menos fiables a la hora de analizar el rendimiento real del modelo.

Por otro lado, vemos un MAE más bajo y un MAPE del 0.39%, que muestra un bajo porcentaje de error relativo. Además, el coeficiente de determinación de nuestro modelo es alto, lo que indica que una gran proporción de la variabilidad en la variable objetivo es explicada por las variables predictoras incluidas en el modelo.

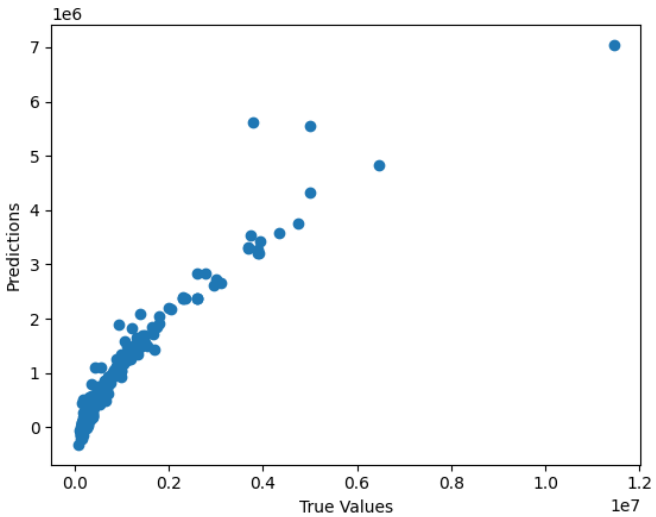


Tabla 4 Valores reales vs predicciones

El diagrama de dispersión ilustra la relación entre los valores reales y predichos. Cada punto representa un dato del conjunto de prueba. Al comparar estos valores, podemos evaluar visualmente el desempeño del modelo.

La proximidad de los puntos a una línea diagonal sugiere que las predicciones del modelo se aproximan a los valores reales, indicando una adecuada precisión en las predicciones del modelo.

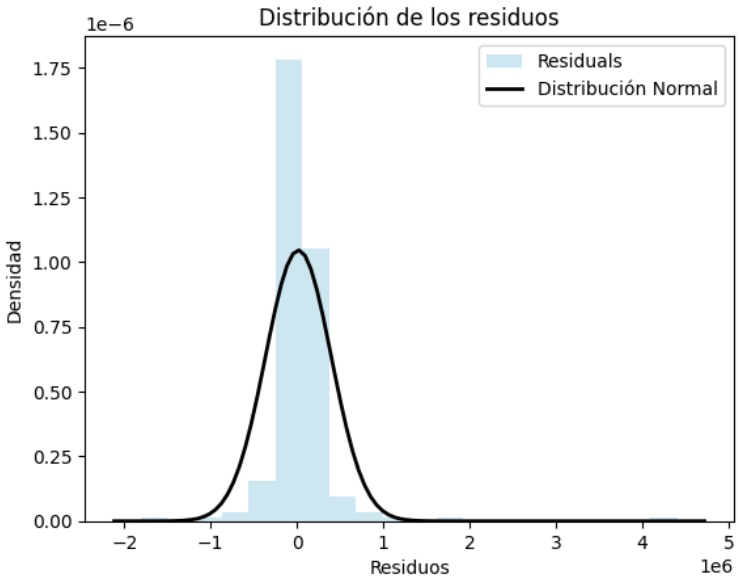


Ilustración 4-9 Distribución de los residuos

El histograma de la distribución de residuos muestra una forma que se asemeja a una distribución normal, esta similitud sugiere que los residuos están distribuidos de manera relativamente uniforme alrededor de cero, lo que indica que el modelo está capturando en gran medida la estructura de los datos.

4.3.2 RANDOM FOREST REGRESSOR

En este estudio, se emplearon todas las variables disponibles, incluyendo las categorías, binarias y numéricas, con el fin de capturar toda la información relevante para el modelo.

Las variables categóricas fueron preprocesadas utilizando la técnica de *one-hot encoding*, que convierte cada categoría en una nueva columna binaria, representando la presencia o ausencia de esa categoría en cada observación. Este enfoque permite al algoritmo de Random Forest Regressor trabajar de manera efectiva con datos categóricos, evitando sesgos inducidos por la codificación numérica.

4.3.2.1 Selección de parámetros

Se aplicó un proceso de *grid search* o búsqueda en cuadrícula con validación cruzada de 5 hojas para encontrar la combinación óptima de parámetros que minimizara el criterio de puntuación negativo de error absoluto medio. En la figura podemos ver que la combinación óptima de parámetros fue 40 observaciones de máxima profundidad y 250 árboles de decisión.

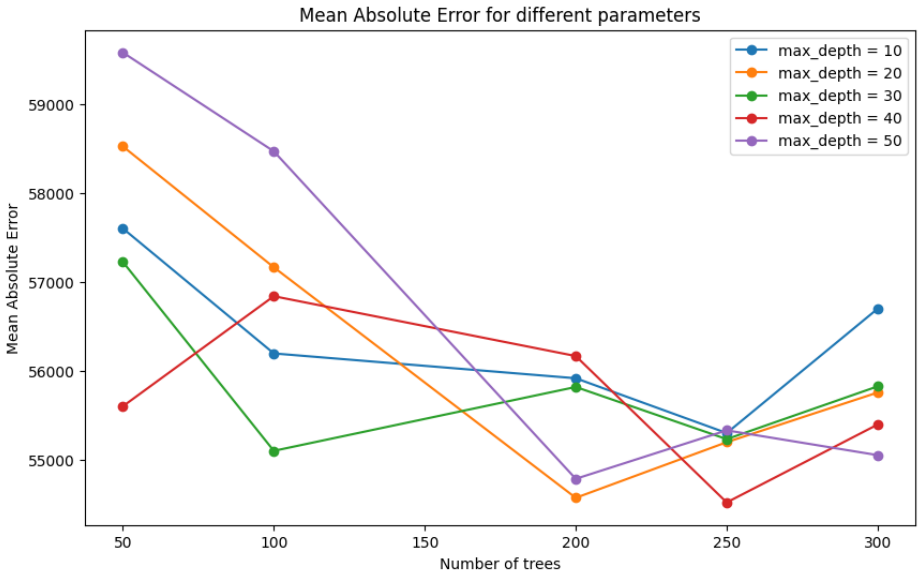


Ilustración 4-10 MAE para diferentes parámetros

4.3.2.2 Métricas y análisis de resultados

MSE	RMSE	MAE	R2	MAPE
228247515050.01	477752.56	94762.41	0.835	0.0629

Tabla 5 Métricas del modelo Random Forest Regressor

Al igual que en el modelo de regresión lineal, obtuvimos valores muy altos para los errores cuadráticos. Por el contrario, el MAE se redujo más de la mitad con respecto a la regresión lineal, esto se debe a la complejidad introducida por el modelo, que es capaz de capturar relaciones no lineales en los datos, y al número mayor de variables que se ha utilizado.

También, obtuvimos un MAPE muy bajo y un coeficiente de correlación alto, que nos indica que el modelo explicaba aproximadamente el 84% de los datos.

Variable	**Importancia**
size	0.695484
priceByArea	0.260990
bathrooms	0.026020
longitude	0.003849
latitude	0.002643
rooms	0.001617
neighborhood_Recoletos	0.001446
floor_2	0.000767
floor_4	0.000585
neighborhood_Castellana	0.000453

Tabla 6 Importancia de las variables en Random Forest Regressor

El análisis de importancia de características muestra que las variables "size" y "priceByArea" tienen la mayor influencia en la predicción de precios de vivienda, las variables número de habitaciones y de baños también son significativas. Por otro lado, la localización y la presencia en ciertos vecindarios tienen una contribución relativamente menor en la predicción, pero sigue siendo significativa.

4.3.3 XGBOOST

Al igual que en el algoritmo RF, en este estudio, se emplearon todas las variables disponibles, incluyendo las categorías, binarias y numéricas. Se procesaron y codificaron de la misma forma que en el caso anterior.

4.3.3.1 Selección de parámetros

En este caso, se empleó de nuevo la técnica de *grid search*, utilizando la siguiente matriz de búsqueda:

```
param_grid = {
    'n_estimators': [100, 200, 300, 350, 400, 500, 600],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.1, 0.01, 0.001]
}
```

Los parámetros óptimos encontrados fueron:

- Learning Rate: 0.1
- Profundidad Máxima: 3
- Número de Estimadores: 600

Pese a ser el número de árboles óptimo igual al máximo de la matriz de búsqueda, se tomó la decisión de no seguir aumentando el número de árboles a probar, ya que se observó que la mejora en el rendimiento en términos de MAE era mínima, evitando así un aumento significativo en el costo computacional y posibles problemas de sobreajuste u *overfitting*.

4.3.3.2 Métricas y análisis de resultados

MSE	RMSE	MAE	R2	MAPE
167806755172.53	409642.228	85621.27	0.87894	0.06122

Tabla 7 Métricas del modelo XGBoost

El MAE se redujo nuevamente en comparación con el modelo Random Forest. El coeficiente R2 se acerca aún más a 1, y el MAPE disminuyó hasta un 6,122%, indicando una mejora significativa en la precisión del modelo propuesto.

Variable	**Importancia**
size	0.601631
priceByArea	0.282377
bathrooms	0.073350
floor_7	0.020064
latitude	0.003502
neighborhood_Chueca-Justicia	0.003123
neighborhood_Recoletos	0.001198
district_Barrio de Salamanca	0.001091
typology_flat	0.000932
rooms	0.000721

Tabla 8 Importancia de las variables en XGBoost

Al igual que en el algoritmo RF, “size” y “priceByArea” siguen siendo las variables más influyentes en la predicción de precios de vivienda, seguidas por el número de baños, lo que destaca la importancia del tamaño de la vivienda para determinar su valor. Por otro lado, la influencia de la ubicación y los vecindarios es relativamente menor pero aún significativa en la predicción.

4.3.4 MODELO HÍBRIDO

Se desarrolló un modelo híbrido que combina los tres modelos de regresión desarrollados. Se entrenaron los modelos por separado y se evaluaron estos modelos utilizando el error medio absoluto (MAE) en un conjunto de datos de validación. Utilizando los errores como ponderaciones inversas, se combinaron las predicciones de los modelos en una sola predicción ponderada.

Este método de combinación ponderada permite aprovechar las fortalezas de cada modelo mientras se mitigan sus debilidades, resultando en una predicción final más precisa y robusta.

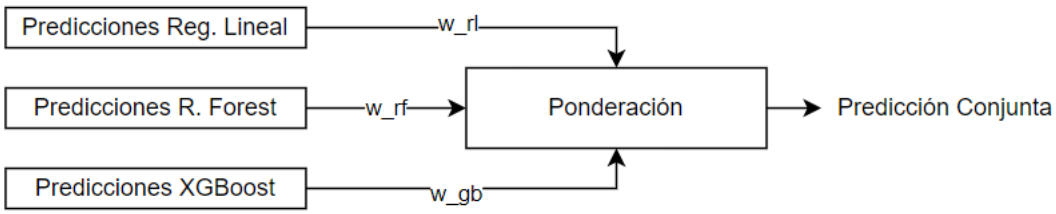


Ilustración 4-11 Estructura algoritmo híbrido. Fuente: elaboración propia

4.3.4.1 Métricas y análisis de resultados

MSE	RMSE	MAE	R2	MAPE
165503578078.78	406821.3097	82848.05	0.8806	0.0830

Tabla 9 Métricas del modelo híbrido

Como se puede ver en la tabla, obtuvimos un MAPE del 8,3% y un coeficiente de autocorrelación de 0,8806. Considerando esto y el MAE obtenido, podemos afirmar que el algoritmo híbrido ha demostrado un rendimiento superior en comparación con los modelos utilizados individualmente.

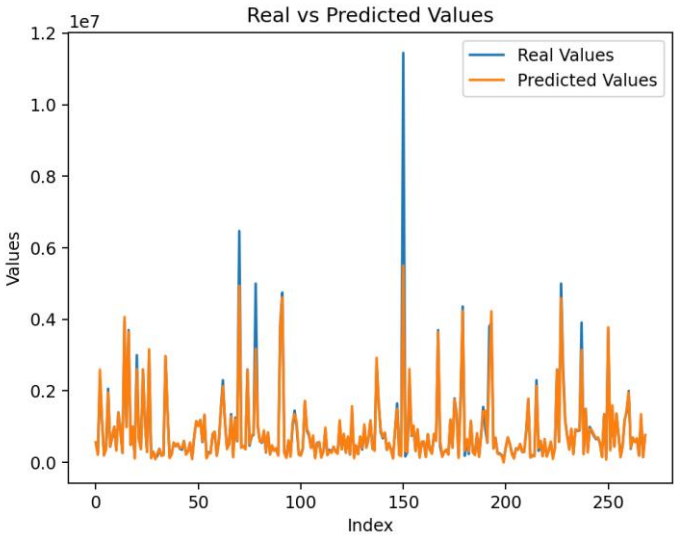


Ilustración 4-12 Gráfico valores reales vs predichos

En la figura, podemos ver como los valores predichos se ajustan casi a la perfección a los valores reales en la mayoría de los casos. Sin embargo, notamos un par de casos donde hay desviaciones significativas, por lo que se decidió examinar los residuos.

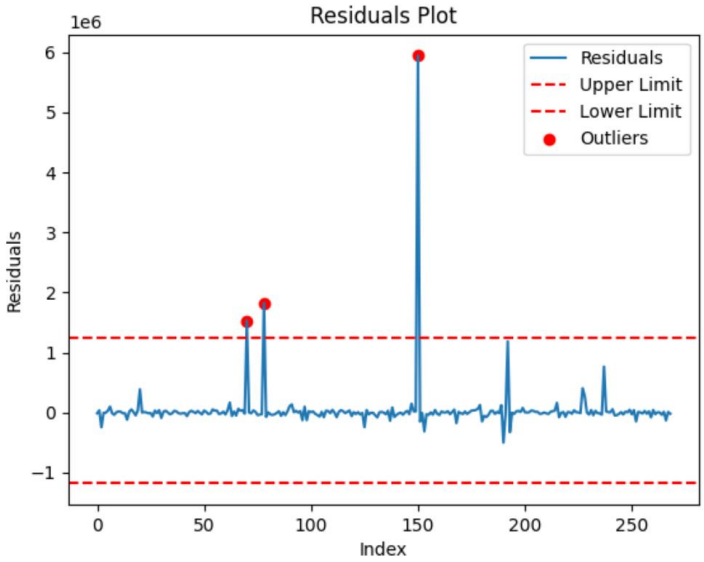


Ilustración 4-13 Gráfico de residuos

Se identificaron los residuos que estaban tres desviaciones estándar por encima de la media y se consideraron *outliers* o datos erróneos. Luego, se seleccionaron los índices correspondientes de estos *outliers* y los eliminamos del conjunto de datos. Posteriormente, evaluamos el modelo con este conjunto de datos modificado y obtuvimos las siguientes métricas de rendimiento.

MSE	RMSE	MAE	R2	MAPE
13246502917.850376	115093.4529756162	48823.21259198546	0.98342	0.079

Tabla 10 Métricas del modelo híbrido sin outliers

Como se puede ver en la tabla, al eliminar estos datos, hemos logrado una mejora notable en la capacidad predictiva del modelo, especialmente en términos de R2 y MAE.

4.3.5 MULTI LAYER PERCEPTRON (MLP)

Se desarrolló una red neuronal de tipo MLP utilizando la biblioteca TensorFlow [10]con la interfaz de alto nivel Keras.

4.3.5.1 Arquitectura de las capas

El modelo cuenta con las siguientes capas organizadas de forma secuencial:

1. Capa de entrada para variables numéricas.
2. Capa de entrada para variables categóricas.
3. Capa de *embedding* [11] para cada variable categóricas: para convertir las variables categóricas en vectores numéricos.
4. Capa de concatenación: unifica las salidas de (1) y (3) haciendo un promedio entre ambas.
5. Capas ocultas densas: utilizan la función de activación ReLU.
6. Capas de salida: capa sin función de activación que produce la salida del modelo.

4.3.5.2 Especificaciones del entrenamiento

- ✓ Como algoritmo de optimización se ha utilizado Adam (*Adaptative Moment Estimation*), que es un algoritmo que combina las ideas de dos variantes de de Gradient Descent, RMSprop y Momentum.

- ✓ Se limitó el número de épocas a 200 (Cantidad de veces que el conjunto de entrenamiento se pasa a la red para el ajuste de pesos) y se estableció un tamaño de lote de 128 (Número de muestras de datos procesadas simultáneamente durante cada iteración del entrenamiento).

[10] Biblioteca desarrollada por Google utilizada principalmente para construir modelos de redes neuronales. Su documentación se puede encontrar aquí .

[11] Las capas de embedding consisten en una representación densa y de menor dimensión de datos dispersos como pueden ser palabras o categorías.

- ✓ Se implementó la técnica de *Early Stopping*., la cual detiene el proceso de entrenamiento cuando la pérdida en el conjunto de validación deja de mejorar, evitando así el sobreajuste del modelo.

4.3.5.3 Selección del learning rate

Se iteró el entrenamiento del modelo utilizando diferentes tasas de aprendizaje para determinar la configuración óptima del modelo. Como se observa en la figura, el modelo con una tasa de aprendizaje de 0.01 fue el mejor, ya que en sólo 150 épocas obtuvo el menor error.

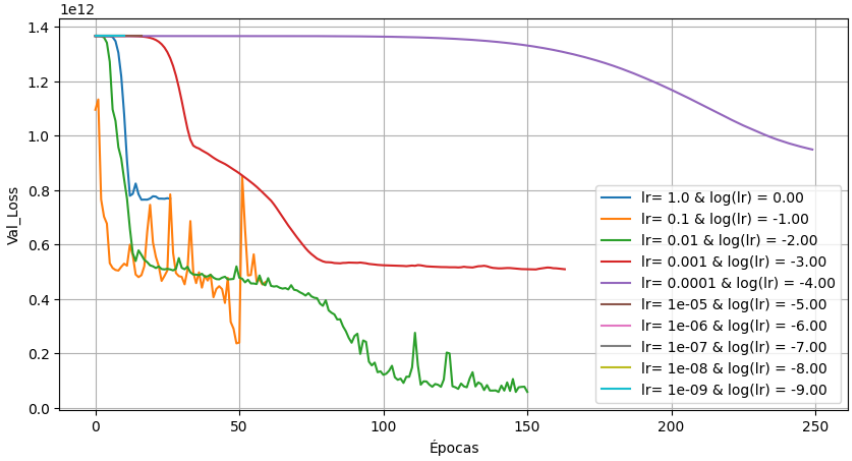


Ilustración 4-14 Selección del learning rate

4.3.5.4 Métricas y análisis de resultados

MSE	RMSE	MAE	R2	MAPE
5821686448	241281.5937	152201.5156	0.923837	0.34233587

Tabla 11 Métricas del modelo MLP

La implementación del modelo MLP resultó en un rendimiento inferior en comparación con los modelos de anteriores. A pesar de la flexibilidad para modelar relaciones complejas y no lineales en los datos, en este caso no logró superar a los modelos de ML tradicionales.

A pesar de rendir peor en todas las métricas, si notamos una mejora en el coeficiente de determinación R2, lo que sugiere que el modelo puede estar capturando ciertos patrones en los datos de manera más efectiva que los modelos tradicionales.

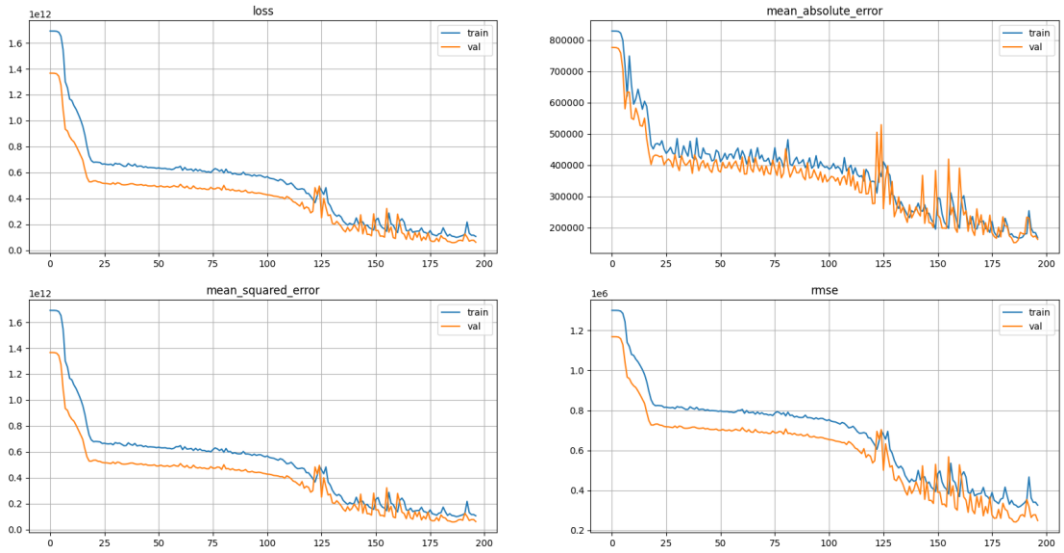


Ilustración 4-15 Evolución métricas vs épocas

En la figura, podemos observar la evolución de las distintas métricas a lo largo de las épocas. En todo momento, podemos ver como los errores en el conjunto de entrenamiento (azul) son mayores que en el conjunto de validación. Esto se debe a que durante la validación se realiza únicamente la inferencia, partiendo del modelo ya entrenado, lo que resulta en un menor error en comparación con el proceso de entrenamiento.

Capítulo 5. CONCLUSIONES

Los modelos de aprendizaje automático ofrecen una infinidad de aplicaciones, gracias a su habilidad de detección de patrones y análisis de datos pueden aplicarse en una gran variedad de campos. Estos modelos son fundamentales en la transición tecnológica actual que viven las empresas, ya que ofrecen una capacidad sin precedentes de análisis de grandes volúmenes de datos y permiten simplificar y automatizar tareas complejas.

En este trabajo nos hemos centramos en la aplicación de estas herramientas en el sector inmobiliario, en concreto, en resolver y simplificar la tarea de tasación de las viviendas. En concreto, el trabajo se ha completado utilizando datos de la ciudad de Madrid, ya que la ciudad se está convirtiendo en un destino cada vez más atractivo para la inversión y compra de viviendas. En este contexto tan dinámico, contar con un sistema de predicción de precios preciso y rápido se vuelve aún más crucial.

Los datos utilizados en este estudio fueron recopilados mediante la API del reconocido portal inmobiliario Idealista. Sin embargo, es importante tener en cuenta que estos datos provienen de anuncios publicados por particulares o agencias, lo que podría introducir un margen de error en la consideración del precio como el precio real de las viviendas. Este margen de error puede atribuirse a posibles inflaciones en los precios, dado que estos pueden estar sujetos a la comisión del vendedor u otros factores que podrían distorsionar la percepción del valor real de las propiedades.

Para asegurar una evaluación sólida y coherente, se eligieron métricas adaptadas al problema que se quería resolver. Es fundamental destacar que todas las métricas calculadas que se han presentado en este trabajo son sobre el conjunto de prueba, lo que garantiza una evaluación precisa del rendimiento de los modelos en la fase de inferencia, es decir, con datos que no había visto antes. Asegurando así el poder de generalización y la usabilidad de los modelos que se han desarrollado.

En la figura se presentan las principales métricas para cada uno de los cuatro modelos de ML desarrollados. Cabe mencionar que el modelo de MLP, no se incluyó en la figura debido a su rendimiento muy inferior que entorpecía el análisis comparativo conjunto.

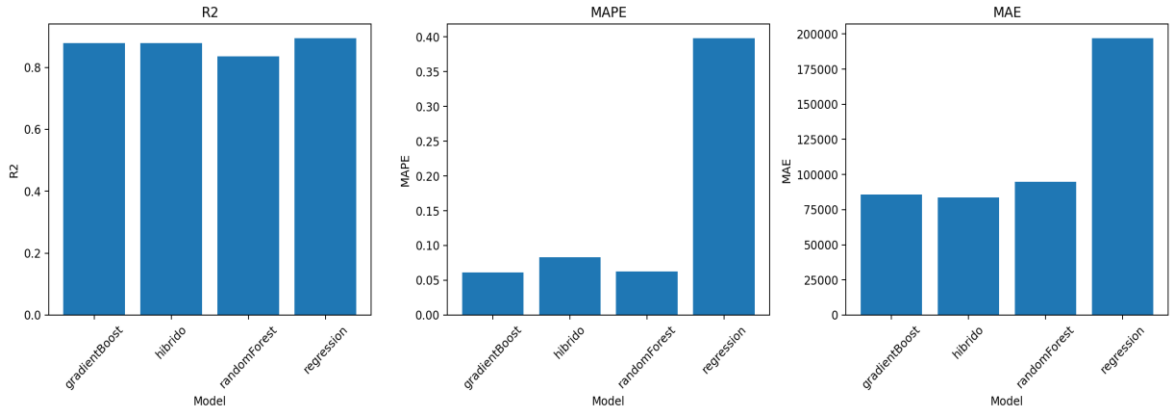


Ilustración 5-1 Comparación métricas entre modelos tradicionales de ML

Comenzando con la regresión lineal, observamos que, si bien es un modelo simple y fácil de interpretar, su rendimiento en este caso se ve limitado por el uso exclusivo de las variables numéricas dejando de lado las categóricas, lo que resulta en un mayor error en comparación con otros modelos más complejos.

Por otro lado, el modelo de Gradient Boosting destaca como una buena opción ya que muestra muy ben rendimiento en todas las métricas analizadas. Sin embargo, una consideración a tener en cuenta es que este modelo requiere un mayor tiempo de entrenamiento debido a su complejidad añadida.

El modelo Random Forest también ha demostrado ser una opción viable, aunque peor comparada con el otro método de ensemble utilizado. Una de las razones que hacen atractivo a este modelo pese a tener métricas ligeramente peores es su capacidad de manejar conjuntos de datos grandes y la posibilidad de ser paralelizado, lo que lleva a tiempos de entrenamiento más rápidos en comparación con los otros modelos.

Por último, el modelo híbrido ha demostrado los mejores rendimientos en términos del coeficiente de autocorrelación y el MAE, es un enfoque interesante ya que busca aprovechar las fortalezas individuales de cada componente para lograr un rendimiento optimizado. No obstante, requiere más recursos computacionales en comparación con modelos individuales.

En conclusión, en este trabajo se han conseguido desarrollar modelos de *Machine Learning* con un rendimiento óptimo, cumpliendo con el objetivo de demostrar la utilidad que estas técnicas tienen en un mercado tan importante como es el inmobiliario. Además, ha realizado un repaso exhaustivo del Estado del Arte del aprendizaje automático en este ámbito, lo que nos ha permitido comprender mejor las tendencias, desafíos y oportunidades en la aplicación de estas tecnologías a este sector.

Por último, mediante el análisis inicial de datos, hemos logrado una comprensión profunda del estado actual del mercado inmobiliario en la capital española. Este proceso de análisis de datos, junto con el desarrollo de los modelos, no solo es relevante para Madrid en particular, sino que puede extrapolarse a otras ciudades utilizando datos específicos para cada una, lo que subraya la versatilidad y aplicabilidad del trabajo desarrollado.

En el siguiente capítulo se detallan posibilidades de aplicación y trabajo futuro para continuar con la línea de investigación.

Capítulo 6. APLICACIONES Y TRABAJO FUTURO

Los modelos de predicción de precios de viviendas desarrollados tienen muchas aplicaciones potenciales dentro del sector inmobiliario. Como se mencionó anteriormente, son útiles para todas las partes involucradas en dicho mercado, incluidos agentes inmobiliarios, tasadores de propiedades, vendedores y compradores.

En esta sección se explorarán dos aplicaciones específicas para estos modelos. La primera estará centrada en la identificación de oportunidades de inversión en vivienda y la segunda se enfocará en el desarrollo de un sistema de tasación rápido y automatizado.

6.1 PROPUESTA DE APLICACIÓN: IDENTIFICACIÓN DE OPORTUNIDADES DE INVERSIÓN EN VIVIENDA

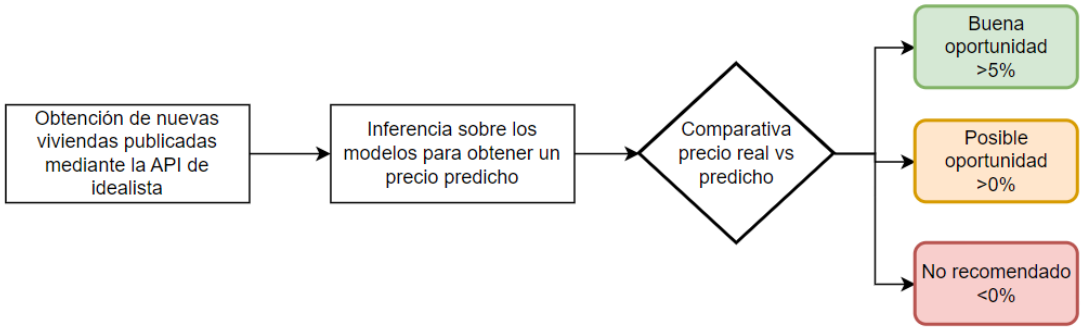


Ilustración 6-1 Diagrama de flujo : Evaluación de oportunidades de inversión.

Esta propuesta implica utilizar los modelos para identificar oportunidades de inversión. Como se puede ver en el diagrama de la figura, el proceso comienza con la extracción de las últimas viviendas publicadas a través de la API de idealista empleada en este trabajo. Luego, se realiza inferencia sobre los modelos desarrollados para obtener los precios estimados.

Una vez obtenidos los precios, se hará una comparación con los precios ofertados por los vendedores y se establecerá el siguiente sistema de clasificación:

1. Buena oportunidad de inversión: se incluirán las viviendas cuyo precio predicho sea al menos un 5% mayor que el precio anunciado.
2. Posible oportunidad de inversión: se incluirán las viviendas cuyo precio predicho sea superior a un 0% respecto al anunciado, pero inferior al 5%.
3. No recomendado para inversión: se incluirán las viviendas cuyo precio predicho es igual o menor al precio anunciado.

En la siguiente figura se plantea un ejemplo de la posible interfaz de usuario para la implementación de esta propuesta.

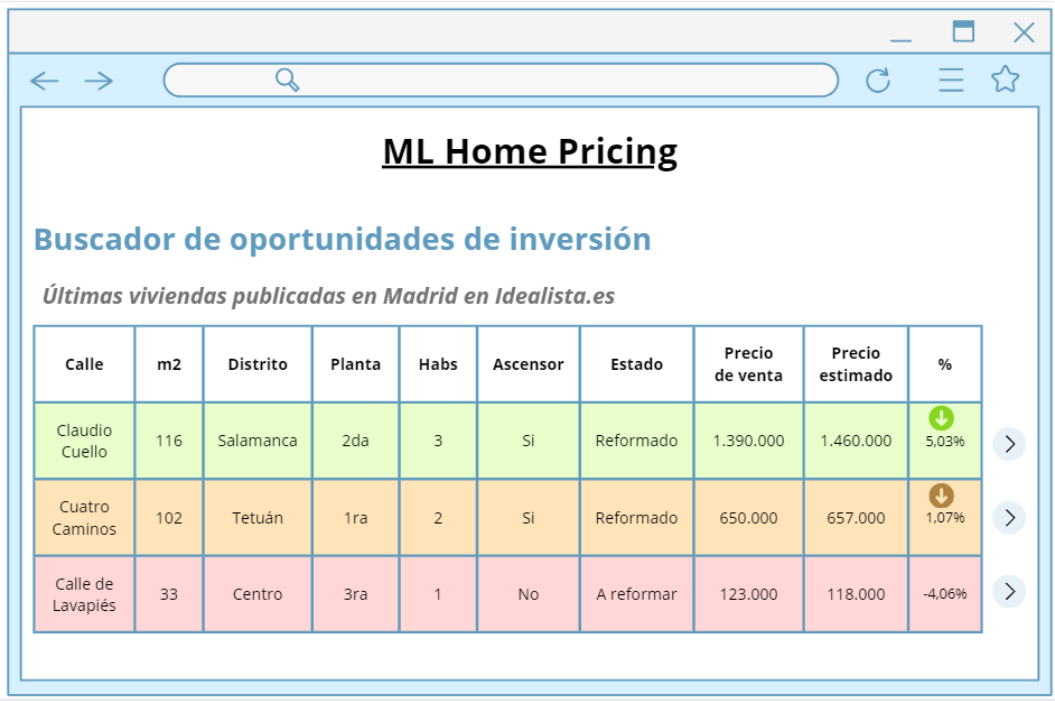


Ilustración 6-2 Ejemplo de interfaz. Buscador de oportunidades de inversión,

6.2 PROPUESTA DE APLICACIÓN: SISTEMA DE TASACIÓN DE VIVIENDAS

El enfoque se centra en desarrollar un sistema de tasación que permita al usuario introducir los datos relevantes sobre su vivienda, los cuales serán consumidos por los modelos para realizar inferencias sobre el precio de esta. De esta manera, el sistema ofrecerá al usuario un precio estimado para su vivienda de manera rápida y automática.

En la figura se presenta una posible interfaz de usuario para esta aplicación.

Ilustración 6-3 Ejemplo de interfaz. Sistema de tasación de precios de vivienda.

6.3 TRABAJO FUTURO

Como trabajo futuro, se plantea desarrollar las aplicaciones antes mencionadas. Esto implicaría desarrollar tanto las interfaces de usuario como un pipeline para que los modelos se reentrenasen periódicamente de forma automática.

Otra mejora posible sería incluir nuevos modelos que permitieran mejorar aún más la precisión de las predicciones proporcionadas.

Una propuesta sería incluir modelos de series temporales como ARIMA y SARIMA, para los que se requerirían datos temporales sobre el precio de la vivienda, segmentados por distritos y recopilados a lo largo de los últimos años.

Por otro lado, dada la creciente popularidad y eficacia de los modelos de lenguaje (LLMs), podría ser de utilidad explorar su aplicación en el sector inmobiliario. Por ejemplo, estos modelos podrían utilizarse para generar o analizar las descripciones de las propiedades que encontramos en los portales en línea como Idealista. También, incluir técnicas de procesamiento de lenguaje natural (NLP) sobre estas descripciones nos permitiría incorporar características relevantes a los modelos al considerar la información textual adicional.

ANEXO I: INFORMACIÓN SOBRE EL CÓDIGO

Se incluye el archivo README.md del repositorio de Github para clarificar la organización y estructura del código.

TFG Housing Prediction
AUTOR: Beatriz Sicilia Gómez

GRADO: 5ºGITT + BA

Descripción del proyecto: Este proyecto se centra en el desarrollo de modelos de Machine Learning para predecir el precio de venta de viviendas en Madrid. Utilizando datos obtenidos a través de la API del portal inmobiliario Idealista, se ha almacenado la información en la nube mediante Firebase. Posteriormente, se llevó a cabo un exhaustivo análisis exploratorio y limpieza de los datos. Los modelos se han construido haciendo uso de las bibliotecas SciKit-Learn, XGBoost y TensorFlow-Keras.

Estructura del Proyecto

Carpeta dataCollection:
llamada_api.py: Esta función permite llamar a la API de Idealista utilizando las claves obtenidas y los parámetros deseados, almacenando los resultados en archivos .json.
loading_data.py: Utilizada para cargar los datos en una colección inicial de Firestore Firebase.

Carpeta dataPreprocessing&EDA:
dataCleansing.py: Carga los datos en crudo desde Firebase y genera una nueva colección eliminando los valores nulos (NAs), realizando selección de variables y creando nuevas características (feature engineering).
EDA.ipynb: Contiene código con visualizaciones para el análisis exploratorio de los datos.

Carpeta para Cada Modelo:
Cada modelo tiene la siguiente estructura:
modelo.py: Contiene el código del modelo, incluyendo las fases de entrenamiento, validación y prueba.
infomodelo.txt: Archivo de texto donde se almacenan características del modelo como hiperparámetros y coeficientes.
metrics.csv: Archivo que recopila las métricas obtenidas para el conjunto de prueba.
visualizaciones: Incluye visualizaciones de los resultados del modelo.

Carpeta comparison:
comparison.py: Contiene el código que lee los archivos de métricas de todos los modelos, los une en un único dataframe y genera visualizaciones comparativas.
final_metrics.csv: Archivo que recopila todas las métricas de los modelos.
final_metrics.png: Imagen comparativa de las métricas.

Anexo:
Nota: Se han excluido del repositorio los archivos con las claves de la API de Idealista y las credenciales de acceso a Firebase para garantizar la seguridad de los datos.

Dependencias: En el archivo requirements.txt se encuentran todas las bibliotecas y versiones necesarias para ejecutar este proyecto de manera adecuada. Para instalarlo, ejecutar el comando:

pip install -r requirements.txt

LINK AL REPOSITORIO

Capítulo 7. BIBLIOGRAFÍA

Aizpun, G. (12 de julio de 2020). *Medium.* Recuperado el 4 de febrero de 2024, de https://medium.com/steplix/machine-learning-applied-to-diimeanalytics-6ed34841ffba

Atria Innovation. (s.f.). Recuperado el 26 de marzo de 2024, de https://atriainnovation.com/blog/que-son-las-redes-neuronales-y-sus-funciones/

Balasubramanian, R. (1 de febrero de 2021). *Medium*. Recuperado el 26 de marzo de 2024, de https://medium.com/analytics-vidhya/stochastic-gradient-descent-sgd-881d7a0ea137

BBVA. (19 de noviembre de 2019). *'Machine learning': ¿qué es y cómo funciona?* Recuperado el 28 de febrero de 2024, de https://www.bbva.com/es/innovacion/machine-learning-que-es-y-como-funciona/

C., X. (28 de mayo de 2018). *LinkedIn*. Recuperado el 3 de marzo de 2024, de https://www.linkedin.com/pulse/intuitive-visual-explanation-differences-between-l1-l2-xiaoli-chen/

Das, S. S., Ali, M. E., Yuan-Fang Li, Kang, Y.-B., & Sellis, T. (2020). *Boosting House Price Predictions using Geo-Spatial Network Embedding.* arXiv.

Fortuño, M. (12 de julio de 2017). *Así han sido los últimos 10 años del sector inmobiliario en España*. Recuperado el 7 de marzo de 2024, de https://www.elblogsalmon.com/sectores/asi-han-sido-los-ultimos-diez-anos-del-sector-inmobiliario-en-espana

Garriga, J. M. (22 de febrero de 2024). *Caixa Bank Research*. Recuperado el 7 de marzo de 2024, de https://www.caixabankresearch.com/es/analisis-sectorial/inmobiliario/mejoran-perspectivas-del-sector-inmobiliario-

espanol#:~:text=Desde%20CaixaBank%20Research%2C%20hemos%20mejorado,las%20550.000%20unidades%20por%20a%C3%B1o.

Geeks for Geeks. (1 de junio de 2022). Recuperado el 13 de febrero de 2024, de https://www.geeksforgeeks.org/bagging-vs-boosting-in-machine-learning/

Grover, R. (2016). Mass Valuations. *Journal of Property Investing & Finance, 34*(2), 191-204.

Guerrero, G. A. (23 de junio de 2021). *Medium*. Recuperado el 12 de marzo de 2024, de https://gladysandrea-rodriguez.medium.com/cross-validation-11e9ea688506

Julkarni, S. (2021). House Price Prediction Using Ensemble Learning. *International journal of creative research thoughts, 9*(5).

Khandelwal, Y. (13 de agosto de 2021). *Analytics Vidhya*. Recuperado el 13 de febrero de 2024, de https://www.analyticsvidhya.com/blog/2021/08/ensemble-stacking-for-machine-learning-and-deep-learning/

Lutis, E. (2 de agosto de 2017). *Medium*. Recuperado el 11 de febrero de 2024, de https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f

MathWorks. (s.f.). *MathWorks*. Recuperado el 10 de febrero de 2024, de https://es.mathworks.com/discovery/linear-regression.html

Parr, T., & Howard, J. (s.f.). *Explained.ai*. Recuperado el 19 de marzo de 2024, de https://explained.ai/gradient-boosting/descent.html

Poeta, S., T.Gerhardt, & Gonzalez, M. S. (2019). Análisis de precios hedónicos de viviendas. *Revista Ingeniería de Construcción, 34*(2).

Raj, R. (s.f.). *Enjoy Algorithms*. Recuperado el 06 de marzo de 2024, de https://www.enjoyalgorithms.com/blog/evaluation-metrics-regression-models

Sivansankar, B. (2020). House Price Prediction. *International Journal of Computer Sciences and Engineering*, *8*(7).

Wang, F., Zou, Y., Zhang, H., & Shi, H. (2019). House Price Prediction Approach based on Deep Learning and ARIMA Model. *IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT).* Zhengzhou, China.

Wang, Y. (2022). The Comparison of Six Prediction Models in Machine Learning: Based on the House prices Prediction. *International Conference on Machine Learning and Intelligent Systems Engineering (MLISE).* Guangzhou, China.