Trabajo Especial de Licenciatura en Física

REDES NEURONALES Y MÉTODOS BASADOS EN ÁRBOLES PARA PREDICCIÓN DE VALORES DE SUELO EN CÓRDOBA

Autor: Stanic Najarro, Alvaro Mateo

Directores: Dr. Francisco Antonio Tamarit y Mgter. Juan Pablo Carranza

Universidad Nacional de Córdoba

Facultad de Matemática, Astronomía, Física y Computación

Diciembre 2023

Agradecimientos

Es con un profundo sentimiento de gratitud que presento este trabajo final, culminación de una etapa fundamental en mi desarrollo profesional y personal. Aprovecho esta oportunidad para expresar mi sincero agradecimiento a todos aquellos que han contribuido a mi viaje académico.

En primer lugar, deseo expresar mi más sincera gratitud a mi familia. Su apoyo incondicional, amor y aliento han sido el pilar fundamental en mi camino. Gracias por creer en mí y por brindarme la fortaleza necesaria para perseguir todo lo que quisiera proponerme.

A mis amigos, los viejos y los nuevos, gracias por estar siempre allí, por las palabras de ánimo en los momentos difíciles, y por compartir conmigo las alegrías de cada logro. Su presencia ha sido un regalo invaluable.

A mis profesores, gracias por su guía, su paciencia y su compromiso con la excelencia educativa. Su conocimiento y sabiduría no solo han moldeado mi aprendizaje académico, sino que también han dejado una huella indeleble en mi desarrollo como persona.

Finalmente, quiero agradecer a todos los que, de una forma u otra, han contribuido a este proyecto. Cada conversación, cada palabra de aliento y cada gesto de apoyo han sido partes integrales de esta travesía.

Con este trabajo, cierro un capítulo enorme de mi vida y me embarco en una nueva etapa, llevando conmigo las lecciones aprendidas, las amistades forjadas y los recuerdos atesorados. Gracias a todos por ser parte de este viaje.

Resumen

En la intersección de la tecnología de aprendizaje automático y la tasación inmobiliaria, este estudio proporciona una evaluación comparativa de los modelos de redes neuronales frente a métodos basados en árboles para la predicción de valores de suelo en la provincia de Córdoba. Abordando una cuestión de considerable importancia económica y social, este trabajo examina la viabilidad de implementar modelos de inteligencia artificial para automatizar y refinar el proceso de tasación de terrenos urbanos y rurales. A través de un análisis exhaustivo de casi 8.000 datos inmobiliarios y variables significativas como la cobertura leñosa y el contexto urbano, demostramos que las redes neuronales no solo pueden competir sino, en ciertos casos, superar a los métodos tradicionales en precisión y relevancia. Contrariamente a las sugerencias de investigaciones recientes sobre datos tabulares, encontramos que las redes neuronales ofrecen un desempeño superior en uno de nuestros conjuntos de datos, el conjunto de datos rurales, y están a la par con XGBoost en el conjunto urbano, con una diferencia de desempeño menor al 1% en favor de las redes. Además, destacamos la eficacia del Quantile Random Forest en segmentos específicos del mercado. Este estudio no solo arroja luz sobre la influencia crítica de factores tanto históricos como actuales en la valoración de propiedades sino que también establece un precedente para la aplicación práctica de algoritmos avanzados en el dominio de la tasación de bienes raíces, sugiriendo un futuro en nuestro país en el que las máquinas potencian la toma de decisiones basada en datos en el sector inmobiliario.

Palabras Clave: Aprendizaje Automático, Random Forest, Quantile Random Forest, XGBoost, Redes Neuronales, Catastro, Aprendizaje Supervisado, Precio De La Tierra

Abstract

At the nexus of machine learning technology and real estate appraisal, this study provides a comparative assessment of neural network models against tree-based methods for predicting land values in the province of Córdoba. Addressing a matter of substantial economic and social importance, this work examines the feasibility of implementing artificial intelligence models to automate and refine the land valuation process for urban and rural properties. Through a comprehensive analysis of over 8.000 real estate data points and significant variables such as woody coverage affected by wildfires and urban context, we demonstrate that neural networks can not only compete but, in certain cases, surpass traditional methods in accuracy and relevance. Contrary to suggestions from recent research, we found that neural networks offer superior performance on the rural dataset and are on par with XGBoost on urban datasets, with a performance difference of less than 1% in favour of the neural networks. Additionally, we highlight the effectiveness of Quantile Random Forest in specific market segments. This study not only sheds light on the critical influence of both historical and current factors in property valuation but also sets a precedent for the practical application of advanced algorithms in the realm of real estate appraisal, suggesting a future in our country where machines enhance data-driven decision-making in the real estate sector.

Keywords: Machine Learning, Random Forest, Quantile Random Forest, XGBoost, Neural Networks, Cadastre, Supervised Learning, Land Pricing

Contenidos

Introducción

En los últimos años hemos sido testigos de una revolución en el campo de la inteligencia artificial, impulsada en gran medida por avances significativos en el aprendizaje automático basado redes neuronales. Estos desarrollos han transformado súbitamente nuestras vidas, alterando las formas de relacionarnos, enseñar y aprender, producir y comercializar, para señalar apenas algunos aspectos relevantes del devenir de las sociedades contemporáneas. En este trabajo final se puede enmarcar en el campo de la aplicación de técnicas de inteligencia artificial neuronal a la definición de políticas públicas, y en especial a la tasación de bienes raíces y su impacto en la recaudación fiscal.

La física teórica, y en particular la física estadística, ha jugado un rol crucial en estos avances. La conexión entre el estado actual de la inteligencia artificial y la mecánica estadística de materiales complejos es muy profunda, ya que esta última, con sus raíces en el estudio de sistemas termodinámicos y fenómenos colectivos, ofrece un marco teórico robusto para comprender sistemas complejos como los son los cerebros naturales y artificiales. En particular, la conexión de la física y el aprendizaje automático puede pensarse desde tres aspectos fundamentales: en primer lugar, desde la teoría, puesto que se aplican métodos originados en la física matemática y teórica, especialmente en la física estadística, para analizar el desempeño de diversas técnicas de aprendizaje automático, y este análisis puede conducir a mejoras en los algoritmos existentes o a una comprensión más profunda de las condiciones necesarias para un rendimiento óptimo[1]. En segundo lugar, desde la inspiración, puesto que algunos de los enfoques metodológicos desarrollados en la física estadística han servido como base a los algoritmos de aprendizaje automático. Un ejemplo de ello es el uso de la teoría de campo medio y sus variantes[2]. Por último, en la aplicación, ya que las técnicas de aprendizaje automático han ganado relevancia recientemente en la resolución de difíciles problemas en la física teórica. Esto incluye desde la detección automática de fases de la materia hasta el aprendizaje de representaciones eficientes de funciones de onda cuánticas[3].

Es innegable que la física tuvo, tiene, y seguramente tendrá mucho que ver con la evolución de las técnicas de inteligencia artificial, sobre todo con redes neuronales. Desde un punto de vista más informal, por ejemplo, puede pensarse a las redes neuronales como sistemas dinámicos capaces de almacenar información por asociación en los cuales numerosos agentes (neuronas artificiales) interactúan para minimizar una función de energía, un concepto directamente tomado de la física estadística. Estos principios han

permitido diseñar algoritmos que imitan la forma en que el cerebro humano procesa la información, lo que resulta en modelos capaces de aprender de grandes volúmenes de datos con una eficiencia y precisión sin precedentes. En este nexo entre la física y el aprendizaje automático, se abren nuevas perspectivas para abordar problemas complejos en áreas muy variadas, como por ejemplo la economía y las ciencias del ambiente. Desde la simulación de sistemas físicos hasta la predicción de tendencias económicas, esta fusión de disciplinas ofrece un terreno fértil para la innovación, lo cual nos lleva al punto central de este trabajo: la aplicación de modelos de aprendizaje de máquinas para la predicción de valores de la tierra en Córdoba.

En la actualidad, el mundo de los bienes raíces enfrenta múltiples desafíos y uno de los más significativos es la tasación precisa y objetiva de terrenos y propiedades. La determinación del precio de un terreno, ya sea urbano o rural, edificado o no edificado, es un proceso que implica considerar múltiples variables, desde su ubicación geográfica hasta características socioeconómicas del entorno. En la provincia de Córdoba, como en casi toda las regiones del mundo, el proceso de tasación ha sido tradicionalmente llevado a cabo por profesionales tasadores que, basándose en su experiencia y conocimiento del mercado, establecen el valor de los bienes raíces.

Este proceso, a pesar de su trascendencia en términos de políticas públicas, ha sido el protagonista de muy pocas innovaciones a través de los años, y a lo ancho de nuestro país se realiza a través de tablas de catastro (conocidas como tablas de Fitte y Cervini [4]), lo cual suele involucrar tiempos prolongados de análisis y esfuerzos superlativos por parte de los expertos.

Sin embargo, en la era de los grandes volúmenes de datos y la inteligencia artificial se plantea una pregunta esencial: ¿es posible automatizar y mejorar la precisión de este proceso utilizando técnicas avanzadas de aprendizaje automático? Esta tesis aborda precisamente esta pregunta y busca contribuir a la literatura existente en el campo de la tasación de bienes raíces mediante la aplicación y comparación de diferentes técnicas de aprendizaje automático, centrándose en las redes neuronales, y comparándolas con métodos ya consolidados, al menos en Córdoba, como son RandomForest, QuantileForest, y XGBoost.

Dado el impacto económico y social que tiene la tasación de terrenos en la provincia de Córdoba y el potencial de estas técnicas para transformar el proceso de tasación en algo masivo, más objetivo, y basado en datos[5], esta investigación se presenta como una herramienta valiosa para tasadores, inversionistas, y políticos.

Además, una motivación adicional para este trabajo se debe al impacto de una reciente publicación científica [6] que establece, a grandes rasgos, que los modelos basados en árboles todavía superan en rendimiento a las redes neuronales cuando tratan con datos tabulares como los que se usan en el problema de tasación de bienes raíces.

A lo largo de este trabajo estableceremos primero un marco teórico sólido que sentará las bases para nuestra investigación. Luego, detallaremos el desarrollo de los modelos de aprendizaje automático, presentando los datos utilizados, los diferentes modelos consider-

ados y los detalles propios del proceso de entrenamiento. Posteriormente, discutiremos los resultados obtenidos, comparando la eficacia de cada método en la predicción del precio por metro cuadrado, tanto para propiedades urbanas como rurales de la Provincia de Córdoba. Finalmente, presentaremos conclusiones y discusiones sobre las implicaciones de nuestros hallazgos y cómo estos pueden ser aplicados en el mundo real.

Sobre El Catastro

El concepto de catastro es central para entender el proceso de tasación oficial de bienes raíces, y por este motivo dedicamos esta sección a explicar qué es un catastro.

El catastro se define como un registro administrativo en el que se describen los bienes inmuebles ubicados en un determinado territorio, proporcionando información sobre sus características físicas, económicas y jurídicas. Estas características pueden incluir la ubicación, la extensión, el valor, el uso, el propietario, y la forma jurídica de tenencia, entre otros aspectos.

El catastro suele ser gestionado por autoridades locales o nacionales y tiene múltiples propósitos, incluyendo:

1. Fiscal: Ayuda a determinar el valor de los bienes inmuebles para calcular impuestos como el Impuesto sobre Bienes Inmuebles (IBI).

2. Jurídico: Aporta seguridad jurídica al proporcionar información detallada sobre los bienes inmuebles.

3. Urbanístico y Territorial: Contribuye a la planificación y gestión del territorio

4. Estadístico: Provee datos para estudios y estadísticas sobre el territorio.

El catastro se actualiza regularmente para reflejar cambios en las propiedades o en su valoración. La información catastral es pública y puede ser consultada por cualquier ciudadano, aunque el acceso a ciertos detalles puede estar restringido para proteger la privacidad.

La coordinación entre el catastro y el registro de la propiedad (donde se inscriben los derechos reales sobre los bienes inmuebles) es fundamental para garantizar la seguridad jurídica en las transacciones inmobiliarias.

En el caso de la provincia de Córdoba, la institución responsable en el área es la Dirección General de Catastro, la cual, junto con IDECOR (Infraestructura de Datos Espaciales de la Provincia de Córdoba, dependiente de la Secretaría de Ingresos Públicos del Ministerio de Finanzas de la Pcia. de Córdoba), desarrolló el Observatorio del Mercado Inmobiliario (OMI)[7] . El OMI es una herramienta que registra y sistematiza información sobre el mercado inmobiliario en una base georreferenciada, que permite conocer la evolución y la dinámica de los valores de inmuebles urbanos y rurales.

La plataforma cuenta con más de 62.000 datos urbanos y rurales de todo tipo sobre el mercado inmobiliario de la provincia (ofertas de ventas, tasaciones, alquileres, entre otros).

La información se actualiza de manera permanente gracias a la *Red OMI*, una comunidad de usuarios conformada por dependencias gubernamentales, instituciones académicas, municipios, colegios profesionales y un gran número de personas que contribuyen con dicha labor.

Teniendo en cuenta estas herramientas, es aquí donde reforzamos nuevamente la idea de que la motivación subyacente que impulsa este estudio radica en la comparación entre dos enfoques fundamentales en el aprendizaje automático: las redes neuronales y las técnicas basadas en árboles. La relevancia de esta comparación reside en la eficiencia y la efectividad de las soluciones propuestas para IDECOR, quienes utilizan técnicas basadas en árboles que, si bien han demostrado su valía, su implementación requiere un considerable esfuerzo por su parte en el preprocesamiento de datos[8]. En contraste, las redes neuronales tienen el potencial de simplificar significativamente este proceso al aprender automáticamente las características relevantes de los datos, eliminando así la necesidad de un procesamiento matemático exhaustivo. La comparación detallada de estos dos enfoques en el contexto específico de IDECOR se convierte en la motivación última de este trabajo, con el objetivo de proporcionar al gobierno provincial una ayuda académica en su búsqueda de métodos precisos de tasación y, más específicamente, una orientación para la elección de las técnicas más adecuada en la predicción del valor de la tierra en la provincia de Córdoba.

Parte I

Marco Teórico

Capítulo 1

Aprendizaje Automático

El aprendizaje automático, una rama esencial de la inteligencia artificial, ha cobrado una importancia sin precedentes en la era digital actual. Es el arte y la ciencia de permitir que las máquinas aprendan de los datos, y su adopción en diversas industrias ha revolucionado los métodos tradicionales de toma de decisiones, análisis de datos y automatización.

El núcleo del aprendizaje automático radica en construir algoritmos que permitan a las computadoras recibir información y utilizarla para realizar predicciones o decisiones sin estar específicamente programadas para ello. Este avance ha desbloqueado innumerables aplicaciones, desde sistemas de recomendación hasta vehículos autónomos y traducción automática de idiomas, por nombrar solo algunas.

Existen diversos paradigmas dentro del aprendizaje automático (Figura 1.1), cada uno con su propia filosofía y aplicabilidad. A grandes rasgos, estos paradigmas se pueden clasificar en cuatro categorías principales:

- Aprendizaje supervisado: este tipo de aprendizaje se basa en la idea de entrenar modelos utilizando conjuntos de datos etiquetados. El "supervisor" en este contexto es el conjunto de datos que proporciona ejemplos de entrada junto con la salida correcta [9].

- Aprendizaje no supervisado: a diferencia del aprendizaje supervisado, en el aprendizaje no supervisado los modelos se entrenan con datos no etiquetados. Aquí, el objetivo principal es descubrir patrones subyacentes o estructuras dentro de los datos, como puede ser agruparlos en diferentes categorías. Algunos algoritmos que son clasificados en esta categoría son K-–Means Clustering, C–Means Clustering y Apriori.

- Aprendizaje por refuerzos: en este paradigma, un "agente" toma decisiones interactuando con un entorno, recibiendo recompensas o penalizaciones basadas en las acciones que realiza, es decir, actúa de acuerdo a una regla o política con una meta final. El objetivo es aprender una estrategia que maximice la recompensa acumulada

a lo largo del tiempo. Algunos algoritmos conocidos en esta rama son Q–Learning o SARSA[10].

- Aprendizaje híbrido: consiste, como su nombre índica, en un enfoque que combina múltiples técnicas o modelos de aprendizaje automático para abordar problemas específicos o mejorar el rendimiento de un sistema. En lugar de depender únicamente de un solo tipo de abordaje, el aprendizaje automático híbrido aprovecha las fortalezas de diferentes enfoques para lograr resultados más sólidos y precisos, ofreciendo flexibilidad y versatilidad. Al combinar modelos, puede adaptarse mejor a diferentes tipos de datos, desde datos estructurados hasta datos no estructurados, imágenes o secuencias de tiempo.

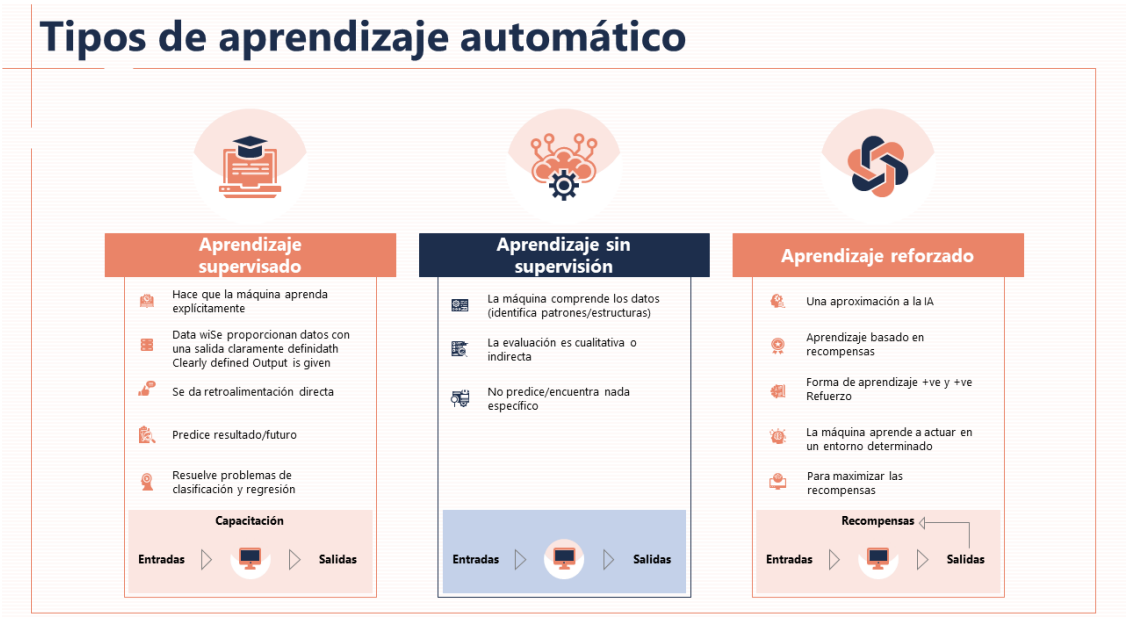

Figura 1.1: Esquema de comparación de paradigmas de aprendizaje automático. De izquierda a derecha: aprendizaje supervisado, aprendizaje no supervisado, y aprendizaje por refuerzos.

A pesar de la diversidad de estos enfoques, el aprendizaje supervisado ha sido particularmente exitoso tanto en la algorítmica como en las aplicaciones prácticas, y será el foco principal de las siguientes secciones.

1.1 Conceptos fundamentales del aprendizaje supervisado

Existen un número de nociones esenciales que se necesitan para una tarea de aprendizaje supervisado:

- *Datos etiquetados:* para que un modelo de aprendizaje supervisado funcione, se requiere un conjunto de datos donde cada entrada tiene asociada una salida o etiqueta que indica el resultado esperado para esa entrada. Por ejemplo, si queremos enseñar

a un modelo a identificar imágenes de gatos, le proporcionaríamos un conjunto de imágenes donde cada figura que sea de un gato está etiquetada como "gato", y el resto como "no gato".

- *Función de pérdida:* en el aprendizaje supervisado, es crucial determinar qué tan bien (o mal) funciona un modelo. Para esto, se utiliza una "función de pérdida" que compara las predicciones del modelo con las etiquetas reales y devuelve un valor numérico que mide el error. El objetivo principal durante el entrenamiento es minimizar este error.
  En nuestro trabajo, utilizaremos la función pérdida "error de porcentaje medio absoluto", mejor conocida como MAPE, por su nombre en inglés (*mean absolute percentage error*). Se define como:

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|, \tag{1.1}$$

  donde A_t es el valor etiquetado del dato, F_t nuestras predicciones, y n la cantidad total de muestras[11].

- *Entrenamiento y prueba:* el conjunto de datos se suele dividir en dos partes: uno para entrenamiento y otro para prueba o testeo. El de entrenamiento se utiliza para ajustar el modelo, mientras que el conjunto de prueba se usa para evaluar su rendimiento en datos no vistos previamente por el mismo.
  En conjuntos de datos lo suficientemente grandes, sin embargo, suele utilizarse una división en tres partes, en lugar de dos: entrenamiento, validación, y prueba. El conjunto de entrenamiento se utiliza para lo mismo que lo antes mencionado; el grupo de validación proporciona una evaluación no sesgada del ajuste del modelo en el conjunto de datos de entrenamiento, mientras se ajustan los hiperparámetros del modelo; y el conjunto de prueba es utilizado para proporcionar una evaluación imparcial del ajuste final del modelo.

- *Validación cruzada:* una forma en la que podremos minimizar el contratiempo de la cantidad baja de datos de entrenamiento será la validación cruzada de "k"-iteraciones, donde los datos se dividen en k subconjuntos de igual tamaño, que también se denominan "iteraciones". Una de las k-iteraciones actuará como conjunto de prueba, y las iteraciones restantes entrenarán el modelo. Este proceso se repite sobre todos los bucles hasta que cada uno de las iteraciones ha actuado como una iteración de testeo. Después de cada evaluación, se retiene una puntuación, y cuando se han completado todas las iteraciones, las puntuaciones se promedian para evaluar el rendimiento del modelo general.

1.2 Aplicaciones del Aprendizaje Supervisado

Las aplicaciones del aprendizaje supervisado son vastas y se encuentran en múltiples campos, desde el reconocimiento de voz en asistentes virtuales[12] hasta la detección de fraudes en transacciones financieras[13], pasando por sistemas de recomendación en plataformas de streaming o comercio electrónico[14].

Existen dos grandes categorías en las cuales puede separarse el aprendizaje supervisado a la hora de realizar estas tareas: clasificación y regresión.

- Clasificación: utiliza un algoritmo para asignar de manera precisa los datos de prueba en categorías específicas. Reconoce entidades específicas dentro del conjunto de datos e intenta sacar algunas conclusiones sobre cómo esas entidades deben ser etiquetadas o definidas. Algunos de los algoritmos de clasificación más comunes son los clasificadores lineales, máquinas de vectores de soporte (SVM, por sus siglas en inglés), árboles de decisión, k-vecinos más cercanos (KNN) y bosque aleatorio (Random Forest). Un ejemplo que recae en esta categoría sería identificar si un correo electrónico es spam o no (en este caso sería una clasificación binaria).

- Regresión: aquí, el objetivo es predecir una cantidad continua en lugar de una categoría. Es comúnmente utilizada para hacer proyecciones, como por ejemplo cuáles serán las ventas de un negocio determinado[15]. La regresión lineal, la regresión logística y la regresión polinómica son algoritmos de regresión populares. Un ejemplo conocido es predecir la relación entre la disminución de la presión arterial y la inyección de dosis de prueba de ciertos medicamentos [16].

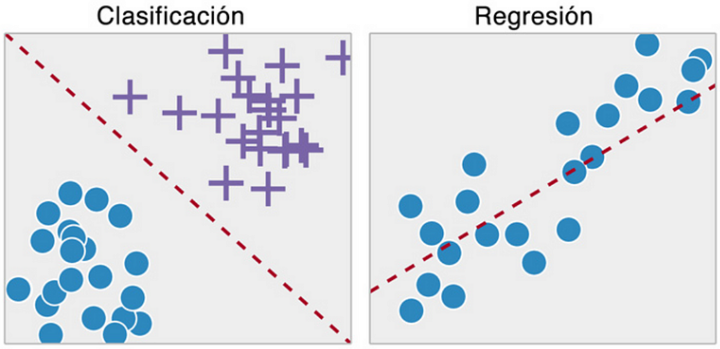

Figura 1.2: Esquematización de las tareas de regresión y clasificación.

1.3 Desafíos en el Aprendizaje Supervisado:

A pesar de su poder y aplicabilidad, el aprendizaje supervisado no está exento de desafíos. La calidad y cantidad de datos etiquetados es fundamental. Sin datos adecuados, incluso el mejor algoritmo puede fallar. Además, hay que tener cuidado con problemas como el *sobreajuste* (overfitting), que surje cuando el modelo se desempeña excepcionalmente bien en los datos de entrenamiento pero desarrollando un nivel de especificidad

extremo, lo que produce pérdida de generalidad, llevando a un desempeño mucho menor en datos nuevos. Otros problema puede ser el *subajuste* (underfitting), donde el modelo no puede capturar la relación entre las variables de entrada y salida con precisión y el desequilibrio de clases, donde algunas categorías están subrepresentadas en el conjunto de datos.

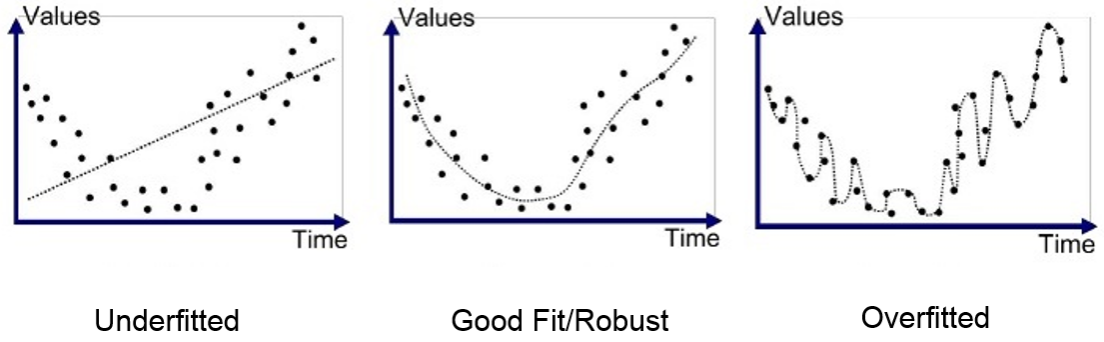

Figura 1.3: Problemas usuales en aprendizaje supervisado. De izquierda a derecha: subajuste, caso "aceptable", y sobreajuste.

Todos estos problemas han sido tenidos en cuenta en nuestro trabajo, y requirieron de un cuidadoso ajuste y selección de los modelos adecuados para el problema específico.

Capítulo 2

Sobre los Modelos

2.1 Redes Neuronales

Las redes neuronales son un pilar fundamental en el campo del aprendizaje profundo y la inteligencia artificial. Las redes neuronales, como concepto, tienen sus raíces en los esfuerzos por comprender y simular los procesos del cerebro humano. El interés en crear modelos computacionales inspirados en la biología comenzó a principios del siglo XX, pero no fue sino hasta la década de los ochenta de dicho siglo cuando la idea comenzó a tomar forma.

La exploración de estas redes como herramientas de aprendizaje automático se ha desarrollado a lo largo de varias décadas, marcada por avances significativos, retrocesos y un resurgimiento continuo en interés y aplicabilidad.

El Perceptrón Simple

En sus orígenes, las primeras conceptualizaciones de las redes neuronales fueron inspiradas por el funcionamiento del cerebro biológico. Pioneros como Warren McCulloch y Walter Pitts, en 1943, propusieron un modelo inicial llamado "perceptrón" que imitaba la operación de las neuronas biológicas. Este punto de partida sentó las bases para el desarrollo ulterior de las redes neuronales artificiales, y su primera implementación fue el 'Perceptrón Mark I', construido en 1957 en el laboratorio aeronáutico de Cornell, por Frank Rosenblatt[17].

Esta implementación estaba basada en una única "neurona" artificial que es utilizada para clasificaciones binarias. La idea fundamental detrás del perceptrón es imitar el comportamiento de una neurona biológica, donde múltiples señales de entrada a través de las dendritas se suman en el cuerpo celular de la neurona y, si la suma supera cierto umbral, la neurona se activa y emite una señal de salida (a través del axón).

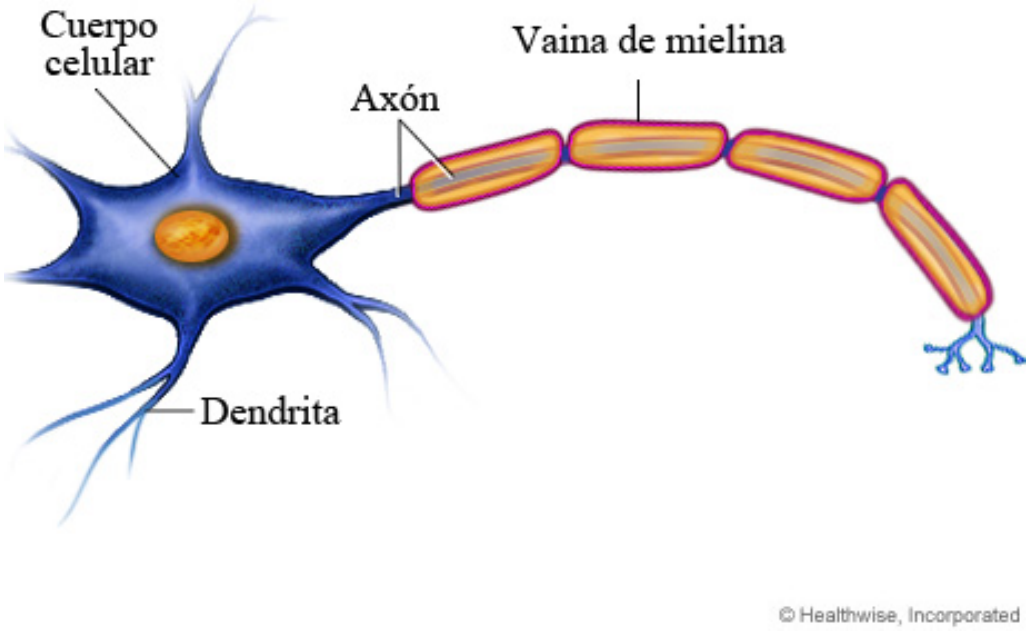

Figura 2.1: Esquema simplificado de una neurona.

El perceptrón realiza una suma ponderada por las sinapsis de sus entradas y luego aplica una función escalón a esta suma para producir una salida. La salida es típicamente binaria, representando clases como 1 o -1 (o 0 y 1). La función del perceptrón se puede representar como:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

donde y es la salida del perceptrón, f es la función de activación, típicamente una función escalón, w_i representa los pesos asociados a cada sinapsis, x_i son las entradas, y b es el sesgo o umbral de activación. La función escalón, que es comúnmente usada como la función de activación en el perceptrón simple, se define como:

$$f(z) = \begin{cases} +1 & \text{si } z \geq 0 \\ -1 & \text{de lo contrario} \end{cases} \tag{2.2}$$

La inicial euforia por las redes neuronales sufrió un revés significativo cuando se publicó el libro "Perceptrons" por Marvin Minsky y Seymour Papert en 1969 [18], donde se destacaban las limitaciones de los perceptrones, especialmente su incapacidad para resolver problemas no linealmente separables. Sin embargo, luego de un "invierno" en la década de los 60 y los 70, las redes neuronales resurgieron en los años 80 gracias, entre otras cosas, al surgimiento del modelo de Hopfield para memoria asociativa [19] y el desarrollo del perceptrón multicapa[20].

El Perceptrón Multicapa

El perceptrón multicapa (MLP, por sus siglas en inglés) es una extensión del perceptrón simple que permite modelar relaciones más complejas. A diferencia del perceptrón simple que consiste en una única neurona, el MLP tiene múltiples capas de neuronas, cada una conectada a las otras, lo que permite capturar patrones no lineales en los datos.

Un MLP típico consiste en tres tipos de capas:

1. Capa de Entrada: Recibe las señales de entrada (características).
2. Capas Ocultas: Una o más capas que transforman la entrada a través de pesos, sesgos y funciones de activación. La presencia de muchas capas ocultas es lo que hace "profunda" a una red neuronal.
3. Capa de Salida: Produce la salida del modelo. La función de activación en esta capa depende del tipo de problema (por ejemplo, regresión o clasificación).

Las operaciones en un MLP se pueden describir matemáticamente de la siguiente manera. Para cada capa l, la salida $h^{(l)}$ se calcula como:

$$h^{(l)} = f^{(l)}\left(W^{(l)}h^{(l-1)} + b^{(l)}\right) \tag{2.3}$$

donde $h^{(l-1)}$ es la salida de la capa anterior (o la entrada para $l = 1$), $W^{(l)}$ y $b^{(l)}$ son los pesos y sesgos de la capa l, respectivamente, y $f^{(l)}$ es la función activación de la capa l. Las funciones de activación comunes en las capas ocultas incluyen ReLU (Rectified Linear Unit, definida como $f(x) = \max(0, x)$), función logística, y $tanh$ (tangente hiperbólica), que introducen no linealidades permitiendo a la red aprender patrones complejos.

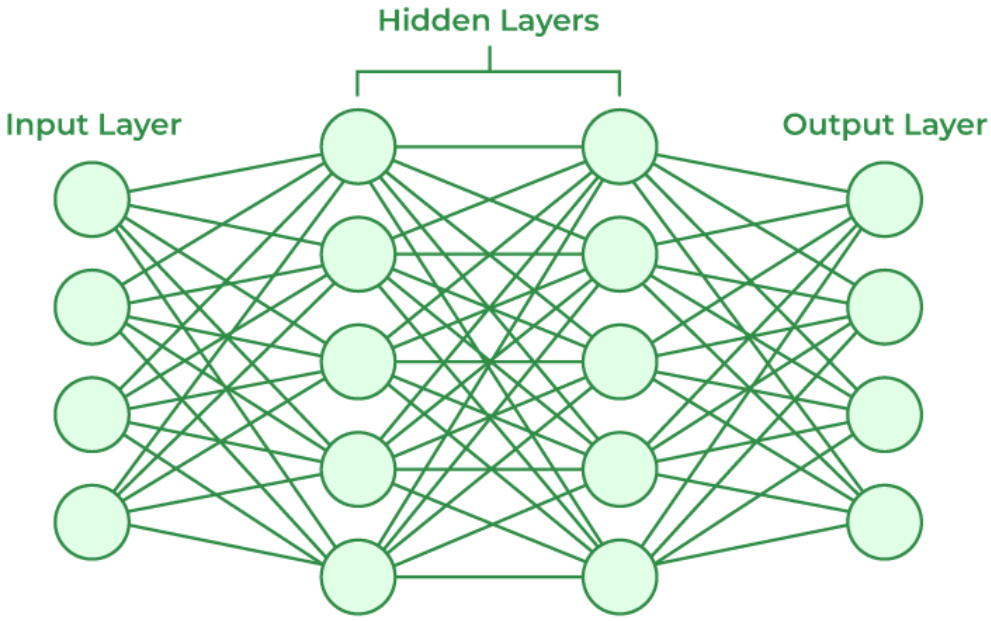

Figura 2.2: Esquema simple de un ejemplo de perceptrón multicapa.

Algoritmo de Retropropagación del error

El aprendizaje en un MLP se realiza a través de un proceso conocido como retropropagación del error y optimización mediante algoritmos como el descenso del gradiente y sus variedades. Durante la retropropagación del error, el algoritmo ajusta los pesos y sesgos de la red de manera que el error entre la salida predicha y la salida real (etiquetas) se minimice. Consiste de los siguientes pasos:

1. Propagar hacia adelante las entradas a través de la red para obtener la salida.
2. Calcular el error entre la salida obtenida y la salida deseada.
3. Propagar hacia atrás este error a través de la red, actualizando los pesos y sesgos en función de su contribución al error (esto se hace mediante el cálculo del gradiente del error con respecto a cada peso y sesgo).

Para iniciar, se define una función de pérdida (o costo) que mide qué tan bien la red neuronal está realizando su tarea. En los problemas de clasificación, en realidad se resuelve un problema de regresión en el cual se hace inferencia sobre la probabilidad de la entrada de pertenecer a cierta categoría. En este caso la función de pérdida más común es la entropía cruzada, mientras que para el problema de regresión se suele utilizar el error cuadrático medio. Luego, una vez realizada la propagación de los datos de entrada hacia adelante a través de la red, siguiendo la ecuación (2.3), la salida final de la red $\hat{y}$ se compara con la salida real y usando la función de pérdida.

El objetivo de la retropropagación es calcular el gradiente de la función de pérdida con respecto a cada uno de los pesos y sesgos en la red. Esto se realiza de la siguiente manera: primero, se calcula el gradiente de la función de pérdida con respecto a la salida de la red, $\frac{\partial L}{\partial \hat{y}}$ donde L es la función de pérdida. Después, este gradiente se propaga hacia atrás a través de la red, calculando el gradiente de la función de pérdida con respecto a cada peso y sesgo: Para cada capa l, empezando por la última y moviéndose hacia atrás, se calculan:

- El gradiente de la función de pérdida con respecto a los pesos $W^{(l)}$: $\frac{\partial L}{\partial W^{(l)}}$.
- El gradiente de la función de pérdida con respecto a los sesgos $b^{(l)}$: $\frac{\partial L}{\partial b^{(l)}}$.

Estos cálculos implican aplicar la regla de la cadena del cálculo diferencial, teniendo en cuenta las derivadas de las funciones de activación.

Finalmente, los pesos y sesgos se actualizan utilizando estos gradientes, generalmente a través del descenso del gradiente o alguna de sus variantes:

$$W^{(l)} = W^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(l)}}, \tag{2.4}$$

$$b^{(l)} = b^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial b^{(l)}}, \tag{2.5}$$

donde η es la tasa de aprendizaje.

Problemas de convergencia en redes neuronales

En el entrenamiento de redes neuronales, especialmente en arquitecturas profundas, a menudo se encuentran problemas de convergencia. Esto significa que el algoritmo de aprendizaje puede no encontrar una solución óptima o puede tardar demasiado en hacerlo. Estos problemas pueden ser causados por diversos factores, como la inicialización inadecuada de pesos, tasas de aprendizaje inapropiadas, o simplemente por la complejidad del espacio de búsqueda de soluciones. Para abordar estos desafíos, se han introducido diversas técnicas, como la regularización, minilotes y el dropout, entre otros.

La regularización es una técnica utilizada para ayudar a prevenir el sobreajuste, lo que puede ayudar a mejorar la convergencia del modelo. Algunos métodos comunes de regularización incluyen:

- Regularización L1 y L2: estos métodos agregan un término de penalización al costo de la red, que es proporcional a la suma de los valores absolutos (L1) o cuadrados (L2) de los pesos. Esto ayuda a mantener los pesos pequeños, lo que a menudo resulta en un modelo más generalizable.
- Early Stopping: consiste en detener el entrenamiento tan pronto como el rendimiento del modelo en un conjunto de validación comienza a degradarse, evitando así el sobreajuste.

Por otro lado, el dropout es una técnica poderosa y sorprendentemente simple para evitar el sobreajuste. Durante el entrenamiento, algunas neuronas se "apagan" aleatoriamente, es decir, su contribución a la activación en la siguiente capa se anula temporalmente. Esto evita que la red dependa demasiado de cualquier neurona individual y promueve una mejor generalización.

La técnica de minilotes consiste en dividir el conjunto de entrenamientos en varios subconjuntos y en cada iteración o época sortear esta división de forma tal de evitar que el método del descenso por el gradiente quede atrapado en mínimos locales o se frene en puntos de ensilladura de la función de pérdida.

Notamos que, a pesar de su poder y flexibilidad, los MLP tienen limitaciones, como la tendencia al sobreajuste especialmente en redes muy profundas o con pocos datos de entrenamiento, y la dificultad para procesar datos con estructuras secuenciales o espaciales, lo que llevó al desarrollo de arquitecturas más especializadas como las CNN (Redes Neuronales Convolucionales) y las RNN (Redes Neuronales Recurrentes), que se encuentran fuera del alcance de este trabajo. Sin embargo, destacamos que una de las grandes ventajas de estas arquitecturas más novedosas de redes neuronales es su capacidad para aprender representaciones de datos complejas y su flexibilidad para adaptarse a diversos tipos de datos y tareas. Esto las hace particularmente adecuadas para tareas más intrincadas como el reconocimiento de voz, la traducción de idiomas, la conducción autónoma de vehículos, el procesamiento de lenguaje natural, y el análisis de imágenes y videos.

En resumen, las redes neuronales representan una tecnología clave en el avance de la inteligencia artificial, con aplicaciones que van desde la mejora de los sistemas de recomendación[21] hasta el desarrollo de terapias médicas avanzadas[22] y la exploración de nuevas fronteras en la ciencia y la tecnología (como predecir, por ejemplo, la estructura del plegamiento de proteínas[23]). Además, no debemos olvidar que el entrenamiento eficaz de redes neuronales involucra no solo una comprensión detallada de la arquitectura de la red y su funcionamiento matemático, sino también una consideración cuidadosa de cómo guiar y regular el proceso de aprendizaje.

2.2 Modelos Basados en Árboles

Un árbol de decisión es un modelo jerárquico de soporte para la toma de decisiones que utiliza un modelo en forma de árbol para representar decisiones y sus posibles consecuencias, incluyendo resultados de eventos aleatorios, costos de recursos y utilidad. Es una forma de mostrar un algoritmo que solo contiene declaraciones de control condicional. Un árbol se construye empezando por la raíz y dividiendo los datos en ramas basadas en alguna medida estadística que evalúa la calidad de una división. Cada nodo interno del árbol corresponde a una característica, y cada hoja del árbol representa un valor de la variable objetivo, ofreciendo una visualización clara y comprensible de las decisiones tomadas por el modelo. Se trata, entonces, de un diagrama de flujo que empieza con una idea principal y luego se ramifica según las consecuencias de las decisiones tomadas.

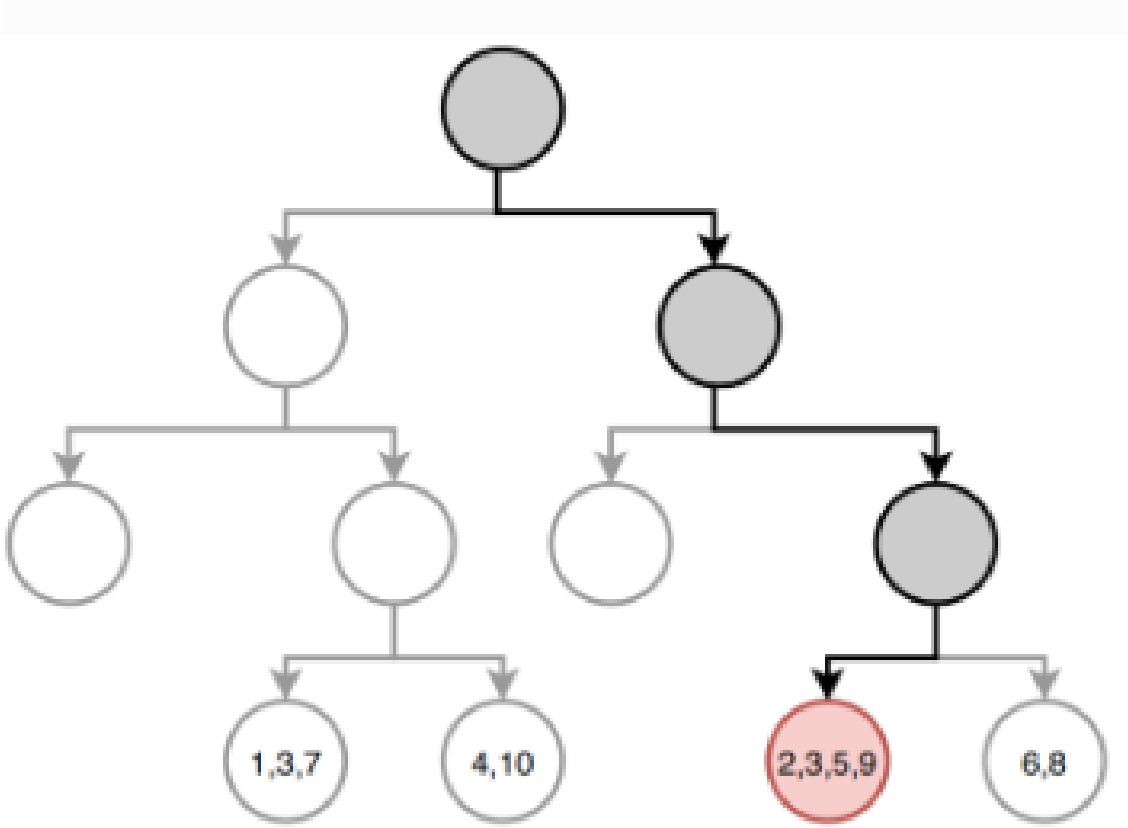

Figura 2.3: Esquema simple de un ejemplo de árbol de decisión.

Tres de los cuatro métodos que usaremos en este trabajo serán algoritmos derivados del concepto de árbol de decisión.

2.2.1 Bosque Aleatorio o Random Forest

Un bosque aleatorio es un clasificador que consiste en una colección de clasificadores estructurados en árboles $\{h(x, \theta_k), k = 1, ...\}$, donde los θ_k son vectores aleatorios independientes e idénticamente distribuidos, y cada árbol emite un voto unitario por la clase más popular en la entrada x[24]. En palabras más sencillas, el modelo Random Forest es una metodología de aprendizaje automático que opera construyendo una multitud de árboles de decisión durante el entrenamiento y entregando como resultado la clase que es la moda (el valor que aparece con mayor frecuencia) de las clasificaciones (en clasificación) o la media de las predicciones (en regresión) de los árboles individuales. Este método se destaca por su versatilidad, eficacia y facilidad de uso, convirtiéndose en uno de los algoritmos preferidos para problemas de clasificación y regresión.

Ventajas de Random Forest

1. Reducción de la varianza: al utilizar múltiples árboles, se reduce el riesgo de sobreajuste, que es común en los árboles de decisión simples, al promediar los resultados de árboles diversos.

2. Manejo de datos no lineales: Random Forest puede capturar relaciones no lineales y complejas entre las características, lo que le permite modelar interacciones que los métodos lineales no pueden.

3. Importancia de las características: ofrece métodos intuitivos para la selección de características, identificando aquellas más importantes para la predicción a través de múltiples árboles.

4. Flexibilidad: puede ser utilizado para datos categóricos y numéricos y no requiere transformaciones extensas de las variables.

5. Facilidad de Uso: Los hiperparámetros de Random Forest son fáciles de entender y no requieren ajustes tan meticulosos como otros modelos avanzados.

Limitaciones de Random Forest

A pesar de todas las ventajas mencionadas con anterioridad, este modelo, como todos, posee ciertas limitaciones. Por ejemplo, la interpretación de un Random Forest puede ser más compleja que la de un árbol de decisión individual debido a la cantidad de árboles involucrados.

Además, modelos con un gran número de árboles pueden ser computacionalmente intensivos y pueden requerir más tiempo para entrenar y realizar predicciones que otros modelos más simples.

En adición a esto, a pesar de sus ventajas en el manejo del sobreajuste, Random Forest puede caer en este problema cuando se enfrenta a conjuntos de datos con ruido extremo o con una cantidad desproporcionada de características irrelevantes.

Sin embargo, cabe destacar que, aunque otros modelos pueden superar a Random Forest en términos de precisión en ciertos escenarios específicos, su capacidad para ofrecer resultados sólidos sin una configuración extensiva lo convierte en una herramienta valiosa.

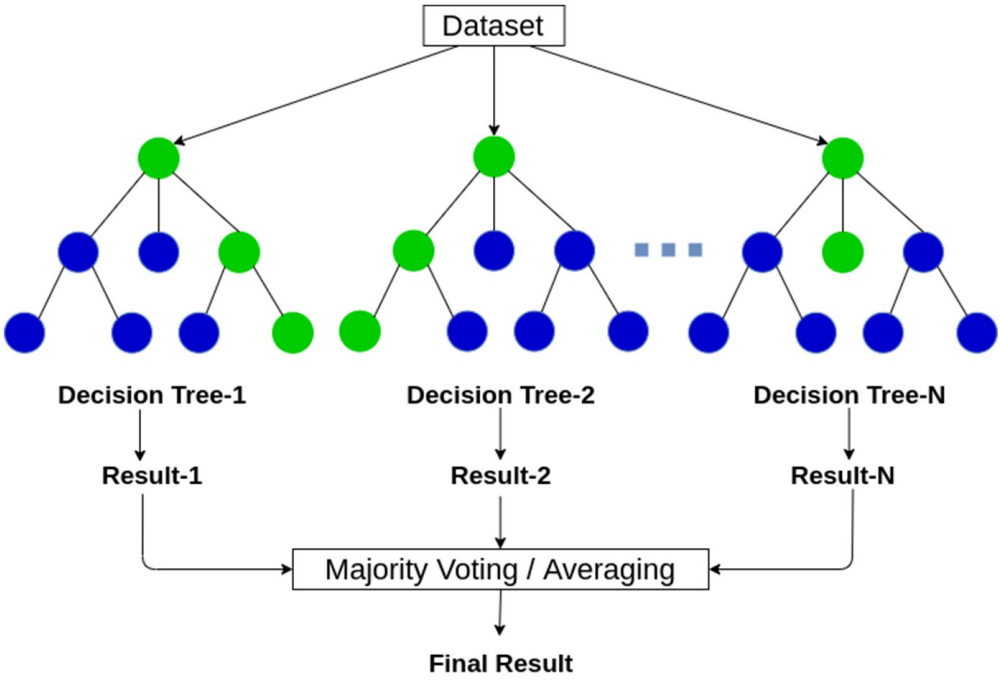

Figura 2.4: Esquema simple de un ejemplo de Random Forest.

2.2.2 Quantile Random Forest

El Quantile Random Forest es una extensión del algoritmo Random Forest, que es conocido por su robustez y eficacia en una amplia gama de tareas de predicción. Mientras que el Random Forest estándar se enfoca en predecir la media o el modo de la distribución de respuesta para la regresión y la clasificación respectivamente, el Quantile Random Forest permite estimar diferentes cuantiles de la distribución de la respuesta, proporcionando así una visión más completa de la posible variabilidad de las predicciones.

En estadística y probabilidad, los $q - cuantiles$ son valores que dividen un conjunto finito de valores en q subconjuntos de tamaños (aproximadamente) iguales. Hay $q - 1$ particiones de los q-cuantiles, una para cada entero k que cumple con $0 < k < q$. El cuantil de orden p de una distribución (con $0 < p < 1$) es el valor de la variable x_p que marca un corte, de modo que una proporción p de valores de la población es menor o igual que x_p. Por ejemplo, el cuantil de orden $0,36$ dejaría un 36% de valores por debajo y el cuantil de orden $0,50$ se corresponde con la mediana de la distribución. En algunos casos, el valor de un cuantil puede no estar determinado de manera única, como puede ser el caso de la mediana (2-cuantil) de una distribución de probabilidad uniforme en un conjunto de tamaño par. Los cuantiles también pueden aplicarse a distribuciones continuas, proporcionando una manera de generalizar las estadísticas de rango a variables continuas.

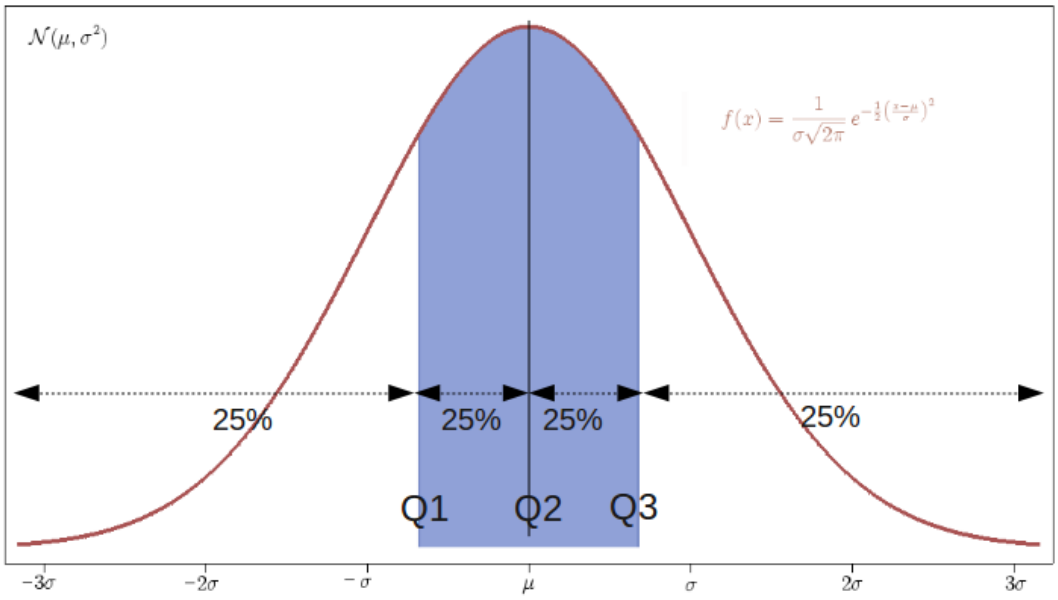

Figura 2.5: Densidad de probabilidad de una distribución normal, con cuartiles (4-cuantiles) mostrados. El área debajo de la curva roja es la misma para los intervalos $(\text{-}\infty, Q_1), (Q_1, Q_2), (Q_2, Q_3)$ y $(Q_3, +\infty)$.

El Quantile Random Forest funciona construyendo múltiples árboles de decisión durante el entrenamiento, con cada árbol aprendiendo de una muestra aleatoria del conjunto de datos. Sin embargo, en lugar de simplemente promediar las predicciones de los árboles (como en el Random Forest estándar), se recopila la distribución completa de los resultados de los nodos hoja de todos los árboles para un punto de datos dado, y de esta se pueden estimar los cuantiles.

Este modelo es particularmente útil en escenarios donde es importante obtener una estimación de la incertidumbre asociada a las predicciones, o cuando se está interesado en predicciones condicionales que no sean simplemente el valor esperado. Por ejemplo, en la gestión de riesgos financieros, uno podría estar interesado en el valor en riesgo (VaR)[25], que es esencialmente un cuantil de la distribución de pérdidas potenciales.

La habilidad para estimar los cuantiles permite que el modelo maneje de mejor manera las distribuciones asimétricas y los datos con valores atípicos o 'outliers', ya que proporciona un rango estimado de resultados posibles en lugar de un único punto de estimación. Además, esta característica hace que el Quantile Random Forest sea valioso para la planificación bajo incertidumbre, optimización de inventarios, y en áreas donde las predicciones de rango completo son más informativas que las predicciones puntuales.

2.2.3 XGBoost

XGBoost, que significa "eXtreme Gradient Boosting", es un algoritmo de aprendizaje supervisado que se ha hecho popular debido a su eficacia en competiciones de ciencia de datos [26] y su versatilidad en resolver problemas de clasificación y regresión. XGBoost es

una implementación del algoritmo de Gradient Boosting, pero con mejoras significativas en eficiencia, escalabilidad y precisión.

El 'boosting' es un meta-algoritmo de aprendizaje automático, basado en la hipótesis planteada por Kearns y Valiant [27] (1988, 1989): ¿Puede un conjunto de clasificadores débiles crear un clasificador robusto? Un clasificador débil está definido como un clasificador el cual está solo débilmente correlacionado con la clasificación correcta (el mismo clasifica mejor que un clasificador aleatorio). En contraste, un clasificador robusto es un clasificador que tiene un mejor desempeño que el de un clasificador débil, ya que sus clasificaciones se aproximan más a las verdaderas clases.

Aunque el boosting no está limitado algorítmicamente, la mayoría de los algoritmos de boosting consisten en aprender iterativamente clasificadores débiles con respecto a una distribución y añadirlos a un clasificador fuerte final. Cuando se añaden, se ponderan de una manera que está relacionada con la precisión de los aprendices débiles. Después de que se añade un clasificador débil, los pesos de los datos se reajustan, lo que se conoce como "re-ponderación". Los datos de entrada mal clasificados ganan un mayor peso y los ejemplos que se clasifican correctamente pierden peso. Así, los futuros clasificadores débiles se centran más en los ejemplos que los aprendices débiles anteriores clasificaron incorrectamente.

Una de las claves del éxito de XGBoost es su capacidad para manejar eficientemente grandes volúmenes de datos y su rapidez comparativa en el entrenamiento de modelos. Utiliza un método de aprendizaje en conjunto, donde nuevos modelos se añaden para corregir los errores cometidos por los modelos existentes. Este proceso se repite iterativamente, refinando continuamente el modelo final.

XGBoost utiliza un algoritmo de potenciación de gradiente, que minimiza una función de pérdida al ajustar los parámetros del modelo. A diferencia de otros métodos de boosting [28], XGBoost introduce un término de regularización en su función de pérdida, lo que ayuda a prevenir el sobreajuste, un problema común en los modelos de aprendizaje automático.

Otra característica importante de XGBoost es su capacidad para manejar automáticamente los valores faltantes y soportar diversas funciones de pérdida, lo que lo hace adaptable a diferentes tipos de problemas de predicción. Además, XGBoost ofrece varias formas de ajustar el modelo, incluyendo la personalización del número de árboles de decisión utilizados, la profundidad de cada árbol, y la tasa de aprendizaje, entre otros parámetros.

En el ámbito del análisis de datos, XGBoost ha demostrado ser extremadamente eficaz en una amplia gama de aplicaciones, desde la predicción de riesgos crediticios hasta la clasificación de imágenes [29] y texto. Su capacidad para manejar conjuntos de datos grandes y complejos, junto con su flexibilidad y precisión, lo convierte en una herramienta de elección para muchos profesionales en el campo del aprendizaje automático.

Capítulo 3

Sobre los Datos

Los datasets que se utilizaron en este trabajo fueron dos: datos urbanos, pertenecientes a las ciudades de San Francisco (y aledaños: Estación Luxardo), Río Cuarto (y aledaños: Santa Catalina, Las Higueras), Villa María, y Villa Nueva; y datos rurales, pertenecientes a parcelas de toda la provincia de Córdoba (Calamuchita, Capital, Colón, Cruz Del Eje, Gral. Roca, Gral. San Martin, Ischilín, Juárez Celman, Marcos Juárez, Minas, Pte. Roque Sáenz Peña, Pocho, Punilla, Río Cuarto, Río Primero, Río Seco, Río Segundo, San Alberto, San Javier, San Justo, Santa María, Sobremonte, Tercero Arriba, Totoral, Tulumba, Unión).

3.1 Datos Urbanos

Este conjunto de datos constaba de 2187 parcelas de lotes ofrecidos o vendidos entre enero de 2017 y junio de 2021, con 47 'features' o columnas cada una. Cada una de estas variables fue obtenida de distintas fuentes: del mercado inmobiliario, de la estructura urbana, de la base de datos catastral, y del procesamiento y clasificación de imágenes satelitales.

Las columnas eran las siguientes:

- forma: forma de la parcela; 0 regular, 1 irregular
- frente: largo de frente del lote en metros
- d_ruta: distancia a rutas
- d_viasprin: distancia a vías principales
- d_viassec: distancia a vías secundarias
- d_alta: distancia a zona de alto perfil inmobiliario
- d_baja: distancia a zona de bajo perfil inmobiliario
- d_lineadiv: distancia a ejes de alto valor inmobiliario

- d_depre: distancia a ejes de bajo valor inmobiliario
- d_rio: distancia a principales ríos y cuerpos de agua
- prom_edif: promedio de superficie edificada en un radio de 500 m
- prom_lote: promedio de superficie de lote en un radio de 500 m
- perc_edif: porcentaje de m2 edificados en un radio de 500 m
- perc_baldm: porcentaje de la superficie de baldío en un radio de 500 m
- perc_bald: porcentaje de m2 baldíos en un radio de 500 m
- perc_ph_cuenta: porcentaje de cuentas en ph, en un radio de 500 m
- perc_val_urb: porcentaje de parcelas con valuación urbana en un radio de 500 m
- inc_edif: valuación de terreno dividido m2 edificados, en un radio de 500 m
- porc_uec: porcentaje del tipo urbano edificado compacto en un radio de 500 m
- porc_ued: porcentaje del tipo urbano edificado disperso en un radio de 500 m
- porc_re: porcentaje del tipo rural edificado en un radio de 500 m
- porc_eau: porcentaje del tipo urbano edificado compacto en un radio de 500 m
- porc_bu: porcentaje del tipo borde urbano en un radio de 500 m
- porc_ear: porcentaje del tipo espacio abierto rural en un radio de 500 m
- porc_agua: porcentaje del tipo agua en un radio de 500 m
- ind_con: porcentaje de píxeles construidos en un entorno de 500m
- bci: promedio de *Biophysical Composition Index* en un entorno de 500m
- rndsi: promedio de *Ratio Normalized Difference Soil Index* en un entorno de 500m
- ui: promedio de *Urban Index* en un entorno de 500m
- ndbi: promedio de *Normal Difference Building Index* en un entorno de 500m
- ndvi: promedio de *Normal Difference Vegetation Index* en un entorno de 500m
- dens_osm: promedio de densidad de calles ponderadas de OpenStreetMaps
- heat_iibb: mapa de calor por ubicación de ingresos brutos

- osm_iibb: producto de valor dens_osm y un mapa de calor por ubicación de ingresos brutos
- fragment: posición en el nivel de consolidación. 0 sin categoría, 1 rural edificado, 2 espacio urbano disperso, 3 espacio urbano compacto.
- ubicación: variable categórica que indica la ubicación en la cuadra: 1 medial, 2 esquina, 3 interno, 4 salida a dos o mas calles, 5 pasillo
- superficie_geom: superficie geométrica de la parcela
- localidad: localidad en función del radio urbano
- oferta_inm: oferta inmobiliaria en el entorno
- fot: factor de ocupación total
- p_anio: año recolección dato
- p_tipodevalor: Tipo de valor relevado, se divide en 11 valores o subcategorías: 1 oferta publicada, 2 Oferta de corredor o prop. 3 venta, 4 valor escrituración, 5 tasación crédito hipotecario, 6 tasación sinceramiento fiscal, 7 remate, 8 tasación, 9 alquiler, 10 tasación auxiliar, 11 valor unitario de referencia.
- p_valor: valor total de la tierra (en dolares o pesos en función de p_moneda)
- p_fechavalor: fecha de relevamiento del valor de la tierra
- p_moneda: moneda a la cual esta expresada el valor, 0 pesos, 1 dólares.
- p_sj: situación jurídica, dividida en 4 clases: 0 default, 1 con escritura, 2 preventa, 3 sin título/posesión
- geometry: representación en coordenadas de la ubicación geográfica de cada dato. La proyección utilizada se llama *proyección POSGAR 98 faja 4*, establecida por el Instituto Geográfico Nacional de la República Argentina[30].

Podemos representar gráficamente la proyección geométrica de los datos como referencia:

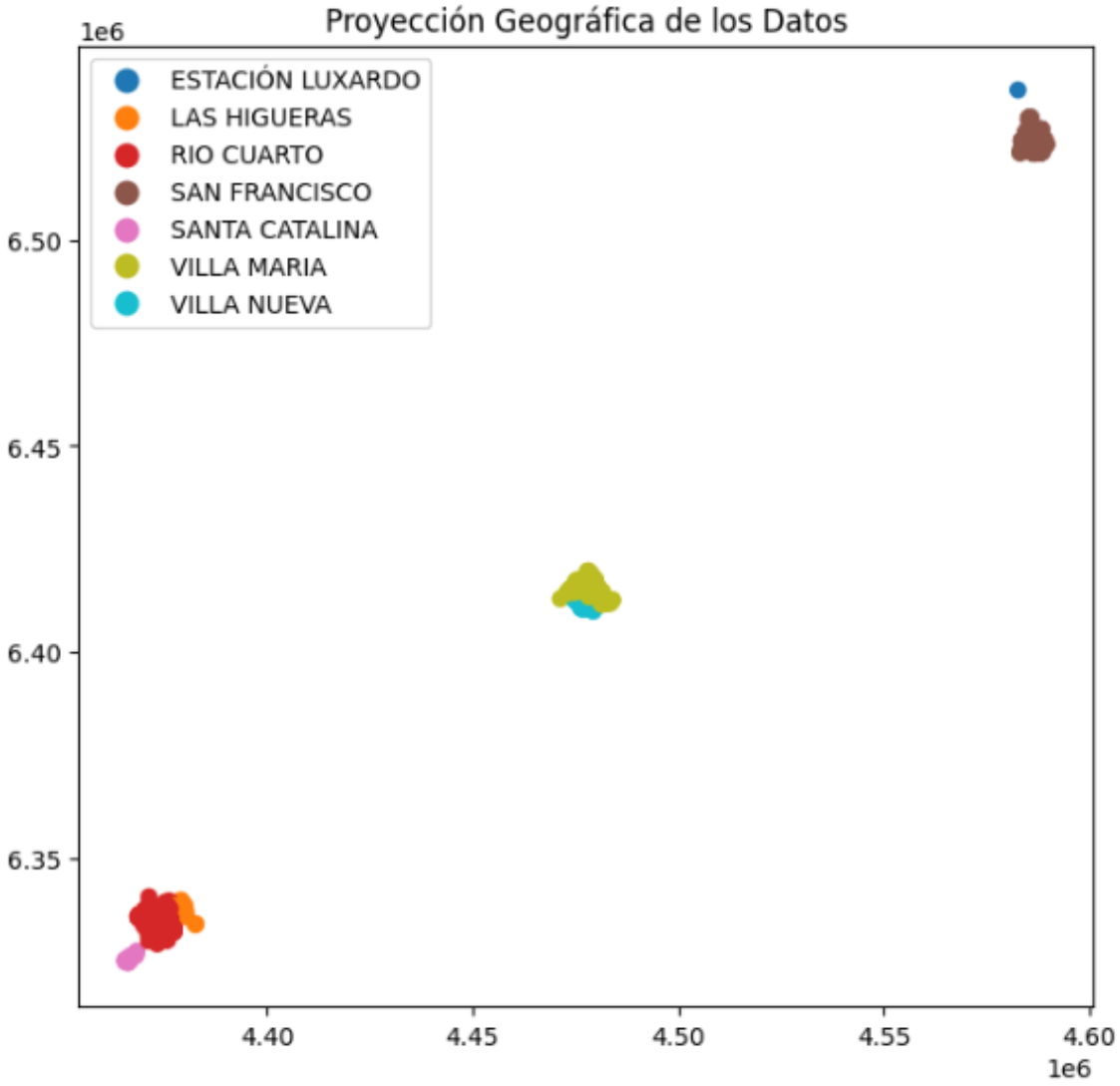

Figura 3.1: Proyección bidimensional de la ubicación geográfica de los datos urbanos. Proyección POSGAR 98 faja 4. Notamos que los ejes representan las coordenadas en esta representación.

Representamos gráficamente los datos pertenecientes a una sola ciudad, para más claridad. En este caso, las parcelas de la ciudad de San Francisco (Figura 3.2):

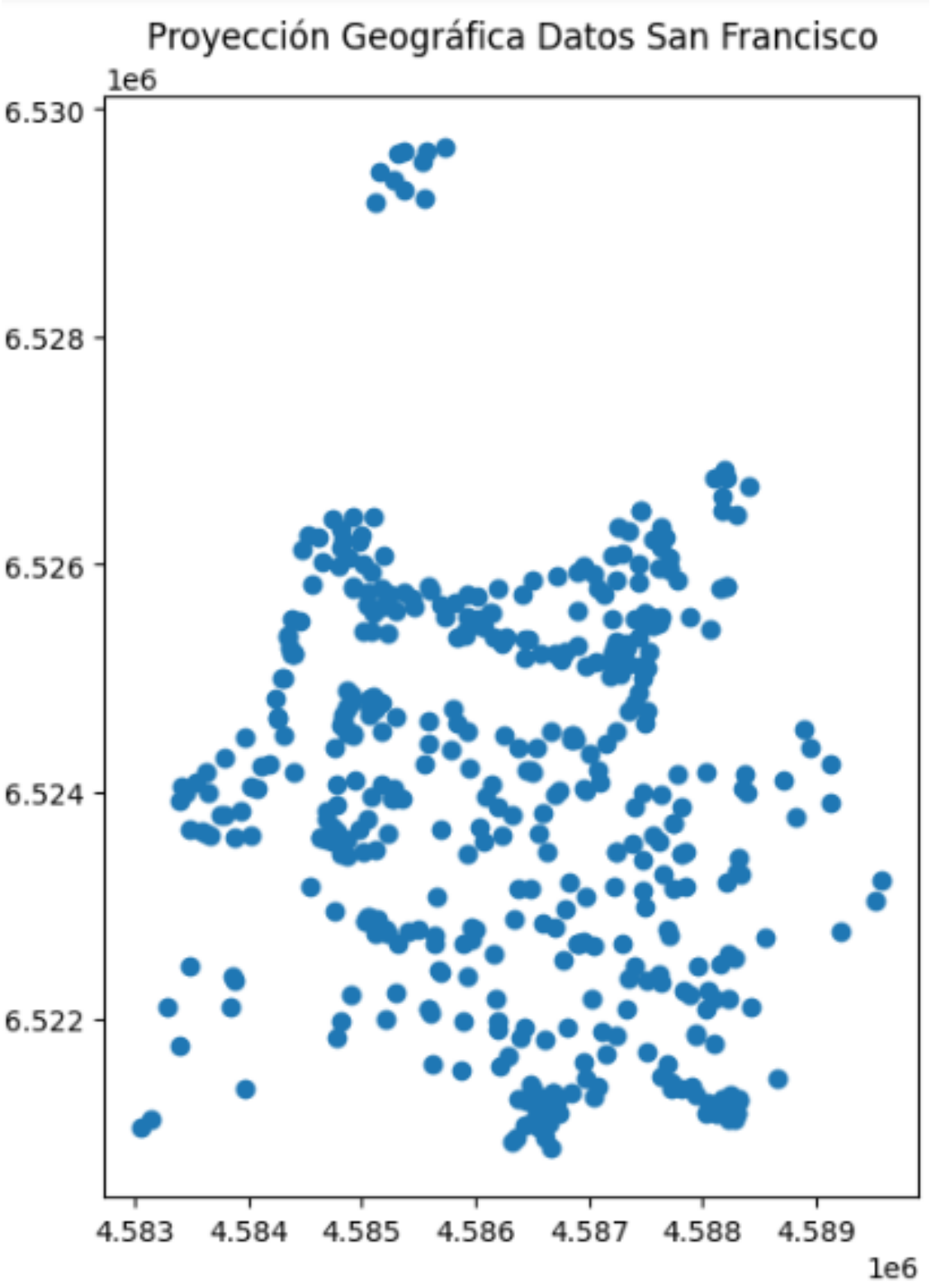

Figura 3.2: Proyección bidimensional de la ubicación geográfica de los datos urbanos de la ciudad de San Francisco.

3.2 Datos Rurales

Este conjunto de datos constaba de 5897 parcelas ofrecidas o vendidas entre febrero de 2014 y agosto de 2022, con 134 'features' o columnas cada una. Destacamos que, a grandes rasgos, podíamos clasificar todas las features en los siguientes grupos (separados por tema): catastrales, localización, vegetación/suelo, hidrológicas, topográficas, climáticas, infraestructura, y económicas.

Cada uno de estos grupos de variables fue obtenido de distintas fuentes: Las variables catastrales fueron calculadas a partir de la base de datos de la Dirección General de Catastro. Las de localización (analizan las características de cada celda según su ubicación respecto a normativas vigentes) fueron obtenidas por IDECOR. Las variables de suelo y vegetación provienen de cartas de suelo de INTA (Instituto Nacional de Tecnología Agropecuaria), y otras fuentes satelitales. Las variables topográficas se calcularon a partir de modelos digitales de elevación, el modelo utilizado es el MERIT DEM.[31] Las variables hidrológicas surgieron a partir de datos de la Administración Provincial de Recursos Hídricos (APRHI)[32]. Las variables climáticas se desarrollaron a partir de productos derivados de la base de datos 'WorldClim', la cual contiene datos sobre temperatura, precipitaciones y radiación solar para el período entre 1970 y 2000. Las variables de infraestructura fueron calculadas a partir del relevamiento a campos realizados por agentes calificados, y se utilizaron fuentes complementarias (publicaciones de la Dirección Provincial de Vialidad, de la Secretaría de Energía de la Nación, etc.). Las variables económicas se desarrollaron a partir de dos fuentes: en primera instancia se utilizó información publicada por la 'Bolsa de Cereales de Córdoba', y las variables referidas a rendimientos agrícolas se derivaron de datos obtenidos de un estudio piloto realizado por IDECOR junto con la Secretaría de Agricultura de la Provincia de Córdoba.

Las columnas eran las siguientes:

- resg_agrop: pertenencia área resguardo ambiental (Ley 9.164)
- cat_otbn1: porcentaje sup. sin presencia de bosque nativo (OTBN Ley 9.814)
- cat_otbn2: porcentaje sup. en categoría VERDE (OTBN Ley 9.814)
- cat_otbn3: porcentaje sup. en categoría AMARILLA (OTBN Ley 9.814)
- cat_otbn4: porcentaje sup. en categoría ROJA (OTBN Ley 9.814)
- nat_prot: pertenencia a área natural protegida
- altura_median: mediana de la altura (msnm)
- altura_stdev: desvío estándar de la altura (msnm)
- pend_median: mediana de la pendiente (porcentaje)
- pend_stdev: desvío estándar de la pendiente (porcentaje)

- tipo_suelo: posición- orden de suelo
- drenaje: posición - limitante edafológica de drenaje
- alcalinidad: posición- limitante edafológica de alcalinidad sódica
- prof_efect: posición- limitante edafológica de profundidad efectiva
- salinidad: posición- limitante edafológica de salinidad
- textura: posición- limitante edafológica de textura
- nitrogeno18: contenido de nitrógeno en suelo
- potasio18: contenido de potasio en suelo
- cic18: capacidad de intercambio catiónico (antes cec)
- d_rios: distancia a ríos principales (metros)
- salinid_agua: categorías de peligrosidad de salinidad de agua
- nfreatico: profundidad del nivel freático (metros)
- acc_riego: pertenencia a área servida de riego por gravedad
- t_med_anual: temperatura media anual (1970-2000) World Clim
- rad_solar: radiación solar media acumulada (1970-2000)
- d_urbaniz: distancia a centros urbanos con más de 2000 hab (metros)
- d_urb_agen: distancia a localidad de importancia zonal (metros)
- d_redelect: distancia a red eléctrica (metros)
- d_cacopio: distancia a localidad con centro de acopio (metros)
- d_puerto: distancia a puerto (San Lorenzo, Rosario en metros)
- arrenda_hist: arrendamiento agrícola zonal - promedio campañas BCCBA
- rto_sj_hist: rendimiento zonal de soja - promedio campañas de 2015 a 16/17 BCCBA
- rto_mz_hist: rendimiento zonal de maíz - promedio campañas de 2015 a 16/17 BCCBA
- arrenda_2021: arrendamiento agrícola zonal - primera estimación 20/21 BCCBA

- rto_mz1718: rendimiento zonal de maíz. Cálculos finales campaña 17/18 BCCBA
- rto_sj1718: rendimiento zonal de soja. Cálculos finales campaña 17/18 BCCBA
- ndvi_mediana: mediana de NDVI (promedio histórico 2000-2020)
- ndvi_stdev: desvío estándar de NDVI (promedio histórico 2000-2020)
- evapo_medi_an: evapotranspiración, media mensual acumulada de la serie (2001-2020)
- pp_med_an: precipitación media acumulada anual histórica (1958-2019)
- t_min_med: temperatura máxima anual media (1958-2019)
- t_max_med: temperatura mínima anual media (1958-2019)
- def_hidric: déficit hídrico medio histórico (1958-2019)
- pdsi: índice de severidad de sequía media histórica (1958-2019)
- rec_1median: mediana dentro de la celda
- rec_1stdev: desvío estándar dentro de la celda
- rec_2median: mediana dentro de la celda
- rec_2stdev: desvío estándar dentro de la celda
- rec_3median: mediana dentro de la celda
- rec_3stdev: desvío estándar dentro de la celda
- perc_agua_perm: agua en la celda (año hidrológico 2020/06/01 al - 2021/04/20)
- perc_agua_aneg: agua en la celda (año hidrológico 2020/06/01 al - 2021/04/20)
- n2_cob0: porcentaje sup. de cobertura sin clasificar
- n2_cob1: porcentaje sup. monte
- n2_cob2: porcentaje sup. arbustales y matorrales
- n2_cob3: porcentaje sup. pastizal natural
- n2_cob4: porcentaje sup. pastizal con rocas o suelo desnudo
- n2_cob5: porcentaje sup. rocas
- n2_cob6: porcentaje sup. suelo desnudo

- n2_cob7: porcentaje sup. salina
- n2_cob8: porcentaje sup. cuerpos de agua
- n2_cob9: porcentaje sup. zonas anegables
- n2_cob10: porcentaje sup. cursos de agua
- n2_cob11: porcentaje sup. urbano compacidad alta
- n2_cob12: porcentaje sup. urbano compacidad media
- n2_cob13: porcentaje sup. urbano compacidad baja
- n2_cob14: porcentaje sup. urbano compacidad muy baja o Abierto
- n2_cob15: porcentaje sup. infraestructura vial
- n2_cob16: porcentaje sup. cultivos anuales de secano
- n2_cob17: porcentaje sup. cultivos irrigados
- n2_cob18: porcentaje sup. pasturas implantadas
- n2_cob19: porcentaje sup. pasturas naturales manejadas
- n2_cob20: porcentaje sup. cultivos hortícolas
- n2_cob21: porcentaje sup. plantaciones forestales maderables
- n2_cob22: porcentaje sup. cobertura leñosa afectada por incendio
- frag_0: porcentaje sup. mapeo fragmentación sin clasificar
- frag_uec: porcentaje sup. en categoría urbano edificado compacto
- frag_ued: porcentaje sup. en categoría urbano edificado disperso
- frag_re: porcentaje sup. en categoría rural edificado
- frag_eau: porcentaje sup. en categoría urbanizado abierto
- frag_bu: porcentaje sup. en categoría borde urbano
- frag_ear: porcentaje sup. en categoría espacio abierto rural
- frag_agua: porcentaje sup. en categoría agua
- sup_constr: porcentaje sup. construida
- parce_cant: cantidad de parcelas en entorno (5 km)

- parce_medi: superficie media de parcela en entorno (5 km)
- cu_moda: moda de CU de la celda
- ip_median: mediana del índice de productividad
- ip_stdev: desvío estándar del índice de productividad
- long_rvial: longitud de red vial y primaria en la grilla
- long_res_osm: longitud de red tipo residencial OSM en la grilla
- cu_clase0: porcentaje capacidad de uso sin clasificar
- cu_clase1: porcentaje capacidad de uso clase I
- cu_clase2: porcentaje capacidad de uso clase II
- cu_clase3: porcentaje capacidad de uso clase III
- cu_clase4: porcentaje capacidad de uso clase VI
- cu_clase5: porcentaje capacidad de uso clase V
- cu_clase6: porcentaje capacidad de uso clase VI
- cu_clase7: porcentaje capacidad de uso clase VII
- cu_clase8: porcentaje capacidad de uso clase VIII
- perc_rural: porcentaje de superficie de parcelas de tipo valuación rural en la celda
- perc_urb: porcentaje de superficie de parcelas de tipo valuación rural en la celda
- sup_med_parc: superficie promedio de las parcelas en la celda (en hectáreas)
- cant_3ha_t: cantidad de parcelas menores a 3 has en un entorno de 3x3 (las celdas aledañas)
- cant_3ha_rur: cantidad de parcelas rurales menores a 3 has en un entorno de 3x3 (las celdas aledañas)
- cant_osm_turi: cantidad de puntos turísticos de OSM en un entorno de 3x3 (las celdas aledañas)
- cant_oferta_inm: cantidad de ofertas y ventas (OMI + sellos) en un entorno de 3x3 (las celdas aledañas)
- cant_turi_omi: cantidad de ofertas de tipo turística en un entorno de 3x3 (las celdas aledañas)

- d_vialpav: distancia a red vial pavimentada (metros)
- min_2020: zonificación valor de agentes (mínimo valor ha)
- max_2020: zonificación valor de agentes (máximo valor ha)
- prom_2020: zonificación valor de agentes (promedio valor ha)
- dens_pivot: mediana de la densidad calculada con la herramienta mapas de calor en un entorno de 5 km
- perc_pivot: porcentaje de superficie con pivotes en la grilla
- ev: equivalente vaca
- vur_2018: valor unitario tierra rural 2018
- vur_2019: valor unitario tierra rural 2019
- vur_2020: valor unitario tierra rural 2020
- vur_2021: valor unitario tierra rural 2021
- qq_maiz: rinde maíz en quintales (campaña 2020/2021)
- qq_soja: rinde soja en quintales (campaña 2020/2021)
- depto: nombre de departamento
- media_mo: media de contenido de materia orgánica en suelo utilizando una grilla de 3x3
- std_mo: desvío estándar de contenido de materia orgánica en suelo utilizando una grilla de 3x3
- media_p: media de contenido de fósforo en suelo utilizando una grilla de 3x3
- std_p: desvío estándar de contenido de fósforo en suelo utilizando una grilla de 3x3
- media_ph: media de pH del suelo utilizando una grilla de 3x3
- std_ph: desvío estándar de pH del suelo utilizando una grilla de 3x3
- media_arcilla: media de valores de contenido de arcilla- SH utilizando una grilla de 3x3
- std_arcilla: desvío estándar de valores de contenido de arcilla- SH utilizando una grilla de 3x3
- media_limo: media de valores de contenido de limo utilizando una grilla de 3x3

- std_limo: desvío estándar de valores de contenido de limo utilizando una grilla de 3x3
- media_arena: media de valores de contenido de arena utilizando una grilla de 3x3
- std_arena: desvío estándar de valores de contenido de arena utilizando una grilla de 3x3

Nuevamente, podemos representar gráficamente la proyección geométrica de los datos como referencia (Figura 3.3):

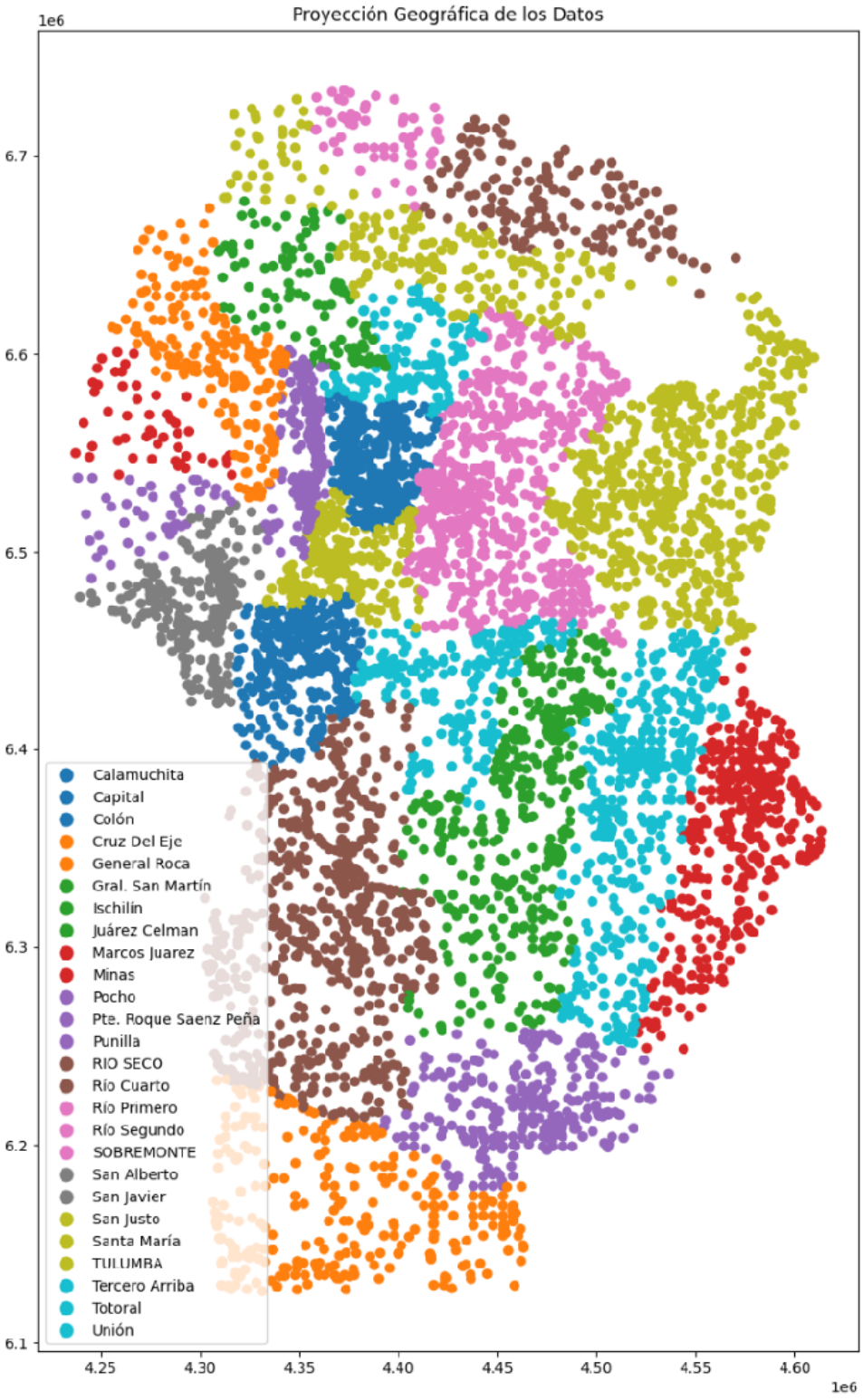

Figura 3.3: Proyección bidimensional de la ubicación geográfica de los datos rurales. Proyección POSGAR 98 faja 4. Notamos que los ejes representan las coordenadas en esta representación.

Parte II

Implementación y Resultados

Capítulo 4

Sobre el Preprocesamiento de los Datos e Implementación de los Modelos

El preprocesamiento es una etapa crucial en el análisis de datos y el modelado predictivo. Esta fase involucra una serie de procedimientos destinados a transformar los datos crudos en un formato más adecuado y eficiente para su análisis. Las razones principales para realizar el preprocesamiento de datos incluyen:

1) Mejora de la calidad de los datos, puesto que en su forma original pueden tener problemas como valores faltantes, errores, inconsistencias o formatos no estandarizados. El preprocesamiento ayuda a corregir estos problemas, asegurando que los datos sean precisos y coherentes, lo cual es fundamental para obtener resultados confiables en cualquier análisis o modelado posterior.

2) Muchos métodos estadísticos y algoritmos de aprendizaje automático requieren que los datos estén en un formato específico. Por ejemplo, las variables categóricas a menudo necesitan ser convertidas a un formato numérico. El preprocesamiento adapta los datos a estos requisitos, facilitando su manipulación y análisis.

3) Los datos bien preparados y limpios pueden mejorar significativamente el desempeño de los modelos de aprendizaje automático. Esto se debe a que la calidad y la relevancia de los datos de entrada tienen un impacto directo en la capacidad del modelo para aprender y hacer predicciones precisas.

4.1 Preprocesamiento Datos Urbanos

A continuación, se describen los pasos más importantes del proceso a realizar para el tratamiento de los datos para su posterior análisis por los modelos creados:

1. **Normalización de fechas:** este proceso involucra la conversión de la columna `p_fechavalor` al formato de fecha y hora estándar para procesamiento conocido

como 'datetime'. Esta transformación es crucial para facilitar el análisis, permitiendo un manejo más eficiente de los datos a lo largo del tiempo, puesto que, de no hacerlas, el algoritmo podría tomar las fechas sólo como números ordinales, y por lo tanto tendería a asignar un mayor peso a los números que determinara como mayores. También es importante notar que, dado que las fechas del dataset original solo contenían información mensual, es decir, información sobre el mes en el que las parcelas habían sido tasadas, al convertirlas a 'datetime', se le asignó automáticamente el primer día de cada mes a cada registro. Este detalle es relevante para comprender cómo se trataron los datos en pasos posteriores, especialmente en relación con el ajuste inflacionario.

2. **Integración del índice inflacionario:** los datos originales, al tener precios o valores fijados en las fechas anteriormente mencionadas, no se encontraban estandarizados a un valor en común de la moneda en el mercado. Por lo tanto, era necesario "ajustarlos" a una misma fecha, de manera que no tuvieran variaciones debido a la fluctuación del peso argentino. Entonces, aquí, se añadió una dimensión económica a los datos importando y fusionando un conjunto de datos adicional, que consiste en el Índice de Precios al Consumidor (IPC) provisto por INDEC. Este índice fue previamente estandarizado para reflejar los cambios en los precios, usando enero de 2017 como base y ajustándolos a junio de 2021, fecha de nuestro último dato relevado. Este paso es fundamental para comprender las variaciones de precios en el tiempo y su impacto en el valor del suelo, y de no realizarlo, precios más antiguos serían mucho menores (de estar en pesos argentinos) que tasaciones más nuevas.

3. **Ajuste de precios por inflación y tipo de moneda:** se realizó una actualización de los precios inmobiliarios teniendo en cuenta la inflación y la moneda en que se expresan. Para los precios en pesos argentinos (ARS), es decir, los precios con `p_moneda` igual a 0, se ajustaron multiplicándolos por el índice inflacionario inverso. Para los precios en dólares estadounidenses (USD), es decir, aquellos con `p_moneda` igual a 1, se convirtieron a pesos utilizando la cotización del dólar 'blue' de junio de 2021. Para el valor del dólar blue, y haciendo alusión al problema de las fechas que habíamos mencionado en el primer paso (únicamente información mensual de los valores), se optó por el promedio del valor en el mes de junio de 2021 (162.5 pesos) debido a la volatilidad observada (de 155 ARS el primer día del mes, a 170 ARS entre los últimos días del mismo mes). La variable resultante, `p_valor_ajustado`, refleja los precios ajustados y actualizados. Este paso es crucial para homogeneizar los valores y permitir una comparación justa entre propiedades.

4. **Limpieza de datos post-ajuste:** Tras los ajustes anteriores, se eliminaron del dataset columnas que se habían vuelto redundantes, tales como `p_anio`, `p_valor`, `p_moneda`, `p_fechavalor`, y las columna derivadas del archivo con el índice inflacionario que habíamos añadido en pasos anteriores. Esta limpieza es esencial

para simplificar el análisis y evitar confusiones o errores derivados de tener datos duplicados o innecesarios.

5. **Codificación de variables categóricas:** para poder procesar las variables categóricas, tales como `localidad`, `ubicacion`, `p_tipodevalor`, `p_sj`, y `fragment` en los modelos estadísticos y de aprendizaje automático, se empleó una técnica llamada One-Hot Encoding. El One-Hot Encoding es una técnica utilizada en el preprocesamiento de datos para convertir variables categóricas en un formato que puede ser más fácilmente procesado por algoritmos de aprendizaje automático y mo-delos estadísticos. Las variables categóricas son aquellas que contienen etiquetas o categorías, como nombres de ciudades, tipos de propiedades, o cualquier otro tipo de clasificación no numérica. En el One-Hot Encoding, cada categoría única de una variable se convierte en una nueva columna binaria (es decir, que solo contiene 0s y 1s) en el conjunto de datos. Estas columnas representan la presencia o ausencia de cada categoría en el registro original. Aquí hay un ejemplo para ilustrar cómo funciona: supongamos que se tiene una variable categórica llamada 'Color' con tres categorías: 'Rojo', 'Verde' y 'Azul'. Con el One-Hot Encoding, esta variable se transformaría en tres nuevas columnas: 'Color_Rojo', 'Color_Verde', y 'Color_Azul'. Si un registro original tenía el valor 'Rojo' para la variable 'Color', en las nuevas columnas tendría un '1' en 'Color_Rojo' y un '0' en 'Color_Verde' y 'Color_Azul'. Entonces, resumiendo, este paso convirtió las categorías en una serie de variables binarias, mejorando la capacidad del modelo para procesar y utilizar esta información.

6. **Descomposición de la geometría en coordenadas:** la variable 'geometry' se dividió en dos nuevas variables, `latitud` y `longitud`, lo que proporcionó una representación más clara y fácilmente utilizable de la ubicación geográfica de cada propiedad. Sin embargo, debemos recordar que las nuevas variables no representan las latitudes y longitudes que usamos en el día a día real, puesto que en la geometría original se hacía uso de una proyección bidimensional llamada POSGAR 98 faja 4, mientras que las latitudes y longitudes más universales son coordenadas en un espacio tridimensional. Los nombres son únicamente una cuestión de conveniencia.

7. **Creación de valor por metro cuadrado:** finalmente, se calculó una nueva métrica, el valor por metro cuadrado, dividiendo el `p_valor_ajustado` por la columna que representaba la superficie de los terrenos, `superficie_geom`. Este indicador es más representativo del valor real del terreno y reemplazó a `p_valor_ajustado` como nueva variable objetivo para los análisis subsiguientes.

8. **Escalado y normalización:** como un paso crucial adicional, todos los datos fueron escalados y normalizados. Este proceso implica ajustar el rango de los distintos atributos numéricos para asegurar que tengan una escala común. El escalado es especialmente importante cuando los atributos varían en magnitud, unidades o

rango, ya que las diferencias en estas escalas pueden sesgar o distorsionar la importancia relativa de ciertas características en los modelos estadísticos y de aprendizaje automático. Por otro lado, la normalización es un proceso que ajusta los valores en los datos para que sigan una distribución normal, lo cual es a menudo un requisito o una recomendación para muchos algoritmos de aprendizaje automático, ya que ayuda a mejorar la eficiencia y la precisión del modelado. Juntos, estos procesos de escalado y normalización aseguran que los datos estén en un formato óptimo para el procesamiento y análisis, contribuyendo a la confiabilidad y validez de los resultados obtenidos en estudios posteriores.

9. **Refinamiento y preparación para análisis**: tras estos pasos, se realizaron revisiones generales sobre el conjunto de datos para asegurarnos de que todas las transformaciones se hubieran aplicado correctamente. Este proceso de verificación fue esencial para garantizar la calidad y coherencia de los datos antes de proceder a cualquier análisis estadístico o modelado predictivo.

Cada paso de este proceso ha sido diseñado para asegurar que nuestros datos sean lo más precisos y representativos posible, sentando las bases para un análisis robusto y confiable en las secciones siguientes.

4.2 Preprocesamiento Datos Rurales

En esta parte nos dedicaremos a describir con precisión los pasos más relevantes en el protocolo de preprocesamiento llevado a cabo en el segundo conjunto de datos, enfocado en propiedades rurales.

1. **Conversión de Fecha y Hora**: de manera similar al proceso realizado en los datos urbanos, iniciamos transformando la columna `p_fechavalor` al formato estándar 'datetime'. Esta conversión es crucial en el manejo de los datos, ya que permite un análisis cronológico más efectivo y detallado. Facilita la comparación y el seguimiento de los cambios en los valores del terreno a lo largo del tiempo, un aspecto fundamental en el análisis del mercado inmobiliario rural.

2. **Unificación de Medidas de Valor**: observamos que los valores de los terrenos se presentaban de dos maneras: por superficie total y por hectárea (`valorha`). Para homogeneizar estos datos, se ejecutó un cálculo donde dividimos el valor total por la superficie correspondiente, obteniendo así una métrica uniforme similar a `valorha` para todos los registros. Esta uniformidad es esencial para comparaciones justas y precisas entre diferentes propiedades, y la eliminación de la columna original `valor` evita redundancias y confusiones futuras en el análisis.

3. **Selección de datos en dólares americanos**: dado que aproximadamente el 98% de las transacciones en las parcelas rurales estaban denominadas en dólares

estadounidenses (USD), se tomó la decisión de centrarse exclusivamente en estos datos. Esta decisión se basó en la premisa de que la devaluación de esta divisa es despreciable en el contexto del período analizado. Esta estabilidad relativa del dólar estadounidense nos permitió excluir del análisis los datos denominados en monedas locales, evitando así la complejidad de ajustarlos por inflación, un paso crucial en el dataset urbano pero que aquí resultó innecesario.

4. **Codificación de variables categóricas**: a continuación, y de manera similar al dataset urbanom, se aplicó la técnica de One-Hot Encoding a una serie de variables categóricas, incluyendo `base`, `origen`, `act_grupo`, `tv`, `tipo_suelo`, `drenaje`, `alcalinidad`, `prof_efect`, `salinidad`, `textura`, `parce_cant`, y `depto`. Si recordamos un poco, esta transformación convierte las categorías en variables binarias, facilitando su uso en análisis estadísticos y modelos predictivos. Es importante notar que este procedimiento resultó en un incremento significativo del número de variables binarias, lo que a su vez aumentó la dimensionalidad del dataset, que requiere un manejo cuidadoso en el modelado para evitar la complejidad excesiva y asegurar la precisión.

5. **Descomposición de datos geométricos**: nuevamente, en un paso elemental, desglosamos la variable `geometry` en dos componentes separadas: `latitud` y `longitud`. Esta división es crucial para análisis geoespaciales detallados y permite la integración con otros conjuntos de datos geográficos, facilitando estudios comparativos y ampliando las posibilidades de investigación. Además, recordamos no confundir los nombres de estas nuevas variables con las coordenadas tridimensionales usadas en el día a día para referirnos a ubicaciones geográficas en nuestro planeta.

6. **Escalado y normalización:** nuevamente, en un paso crucial adicional, todos los datos fueron escalados y normalizados. Si recordamos un poco, el proceso implica ajustar el rango de los distintos atributos numéricos para asegurar que tengan una escala común, contribuyendo a la estabilidad de los resultados obtenidos.

7. **Refinamiento y preparación para análisis**: finalmente, se realizaron revisiones exhaustivas sobre el conjunto de datos para asegurar que todas las transformaciones se hubieran aplicado correctamente. Este proceso de verificación fue esencial para garantizar la calidad y coherencia de los datos antes de proceder a cualquier análisis estadístico o modelado predictivo.

Con la finalización de estos pasos de preprocesamiento, tanto nuestros datos rurales como urbanos quedaron completamente preparados para su uso en los modelos de análisis y predicción. Cada una de estas etapas fue meticulosamente diseñada para asegurar que los datos fueran no solo precisos y coherentes, sino también óptimos para los complejos procesos de modelado estadístico y aprendizaje automático.

Como ya mencionamos antes, en este trabajo fueron utilizados cuatro modelos difer-

entes al problema de tasación de inmuebles urbanos y rurales: Redes Neuronales, Random Forest, Quantile Random Forest, y XGBoost. Se hizo especial hincapié en las redes neuronales y en su optimización, pero cabe destacar que aún así se realizaron búsquedas de algunos hiperparámetros óptimos para los métodos restantes.

Para llevar a cabo estas optimizaciones, se utilizaran métodos de validación cruzada junto con búsquedas exhaustivas dentro de ciertos rangos de valores. Esta metodología nos permitió no solo identificar los hiperparámetros más adecuados sino también entender cómo cada uno de ellos influyó en el desempeño del modelo.

4.3 Modelos y Optimización de Hiperparámetros

4.3.1 Redes Neuronales

Las redes neuronales, dada su complejidad y flexibilidad, ofrecieron un amplio espectro de hiperparámetros para optimizar. Se prestó particular atención a:

- Número de capas ocultas de la red: se varió la profundidad de la red para evaluar su impacto en el aprendizaje y la generalización.

- Número de neuronas por capas o num_units: se experimentó con diferentes cantidades de neuronas en cada para identificar la densidad óptima de cada caso, lo que influye directamente en la capacidad del modelo de capturar relaciones complejas en los datos.

- Taza de aprendizaje o learning_rate: se probó un rango de valores para determinar el tamaño del paso óptimo en cada iteración, buscando un equilibrio entre la rapidez de convergencia y la precisión del mínimo local encontrado.

- dropout_rate: se evaluaron varias tasas de dropout para mitigar el sobreajuste, probando valores que variaron entre desde 0.0 hasta 0.4.

- Tamaño de los lotes o batch_size: se experimentó con tamaños de lote que variaron entre 32 y 256, buscando así evitar que el descenso por el gradiente quedase atrapado en mínimos locales o demorado en puntos de ensilladura.

- Funciones de activación: se probó con diversas funciones de activación, como 'relu', 'sigmoid', 'selu' y 'elu', entre otras, para determinar cuál permitía captar mejor los detalles del problema.

- Optimizador: se exploraron distintos optimizadores, incluyendo 'nadam', 'adam', 'rmsprop', entre otros, para evaluar cómo afectaban la velocidad y calidad de la convergencia del modelo.

4.3.2 Random Forest

Para el modelo de Random Forest, nos centramos en dos hiperparámetros fundamentales:

- n_estimators: se experimentó con una variedad de cantidades de árboles para determinar cuántos eran necesarios para un rendimiento óptimo.
- criterion: se analizaron diferentes funciones de medida de calidad (como 'gini' o 'entropy') para las divisiones, evaluando su impacto en la precisión y generalización del modelo.

4.3.3 Quantile Random Forest

Para el modelo de Quantile Random Forest, el enfoque fue más específico: únicamente podíamos trabajar sobre los cuantiles que queríamos obtener. Nuestra atención se centró en obtener resultados para los cuantiles 0.25, 0.50 y 0.85, lo que nos permitió examinar la distribución de los datos en varios puntos clave.

4.3.4 XGBoost

En el modelo de XGBoost, se priorizaron dos hiperparámetros críticos:

- n_estimators: similar a Random Forest, se probó con distintas cantidades de árboles para encontrar el balance adecuado entre rendimiento y eficiencia computacional, aunque siempre priorizando el rendimiento.
- learning_rate: se ajustó este parámetro para optimizar la velocidad a la que el modelo aprende, sin sacrificar la precisión y evitando caer en mínimos locales subóptimos.

Cada uno de estos modelos requirió un enfoque único en la optimización de sus hiperparámetros, reflejando la diversidad y complejidad de las técnicas de modelado en el análisis de datos. Los resultados obtenidos de estas optimizaciones no solo mejoraron significativamente el rendimiento de los modelos sino que también proporcionaron una comprensión más profunda de cómo las diferentes configuraciones interactúan con los datos específicos de este estudio.

4.4 Resultados

4.4.1 Datos Urbanos

En la fase de optimización de hiperparámetros del modelo de redes neuronales, hemos adoptado un enfoque estratégico que persigue un equilibrio entre la exhaustividad de la búsqueda por un lado, y la eficiencia computacional por otro. En esta sección, presentamos

los resultados obtenidos de una serie de procesos de optimización de hiperparámetros, enfocándonos en aquellos que influyen significativamente en el rendimiento del modelo.

Para ello, hemos realizado una validación cruzada, como se explicó en el capítulo 1, para evaluar cómo diferentes valores de hiperparámetros como la tasa de aprendizaje (learning rate), el número de neuronas por capa, y la tasa de dropout, impactan en la eficacia del modelo. Además, se han incluido análisis comparativos para diferentes optimizadores y funciones de activación, así como para la evolución de la pérdida durante el entrenamiento y validación, tanto para un modelo inicial previo a la optimización, como también para el modelo final.

Es importante destacar que, aunque la búsqueda exhaustiva de parámetros (grid search) es un método comúnmente utilizado para la optimización de hiperparámetros, en este estudio hemos optado por no aplicar una búsqueda en grilla que incluya todos los parámetros simultáneamente. Esta decisión se debe a que el número de combinaciones posibles aumenta exponencialmente con cada parámetro adicional, lo que resulta en un tiempo de cómputo prohibitivo para la validación cruzada. Por tanto, hemos priorizado un enfoque más focalizado, seleccionando cuidadosamente los parámetros y sus rangos para la optimización.

La elección del número de capas y de neuronas en cada una de estas capas se basó en un equilibrio entre la complejidad del modelo y la prevención del sobreajuste. Optamos por una arquitectura con 7 capas, incluyendo las de entrada y salida, es decir, una red profunda de 5 capas ocultas, y se les aplicó dropout a las dos primeras capas ocultas (es decir, las dos capas siguientes a la de entrada), basándonos en la necesidad de capturar la complejidad de los datos sin incurrir en una carga computacional excesiva. Además, por simplicidad, se decidió que todas las capas profundas tuvieran el mismo número de neuronas entre sí. Esta decisión se respaldó con pruebas empíricas, donde se observó que un aumento adicional en el número de capas y mayor variabilidad en el número de neuronas de cada una no proporcionaba mejoras significativas en el rendimiento del modelo, y en algunos casos, conducía a un sobreajuste.

Los resultados que se muestran en esta sección, referidos al uso de redes neuornales, fueron obtenidos utilizando las librerías TensorFlow y Keras para entrenar los modelos, y Matplotlib, entre otras, para la representación gráfica de los resultados.

A continuación, presentamos una serie de gráficos que ilustran los resultados de estos procesos de optimización, proporcionando una visión detallada y cuantitativa de cómo cada ajuste de hiperparámetro contribuye a la mejora del modelo.

Para la combinación de los hiperparámetros `learnig_rate`, `num_units`, y `dropout_rate`:

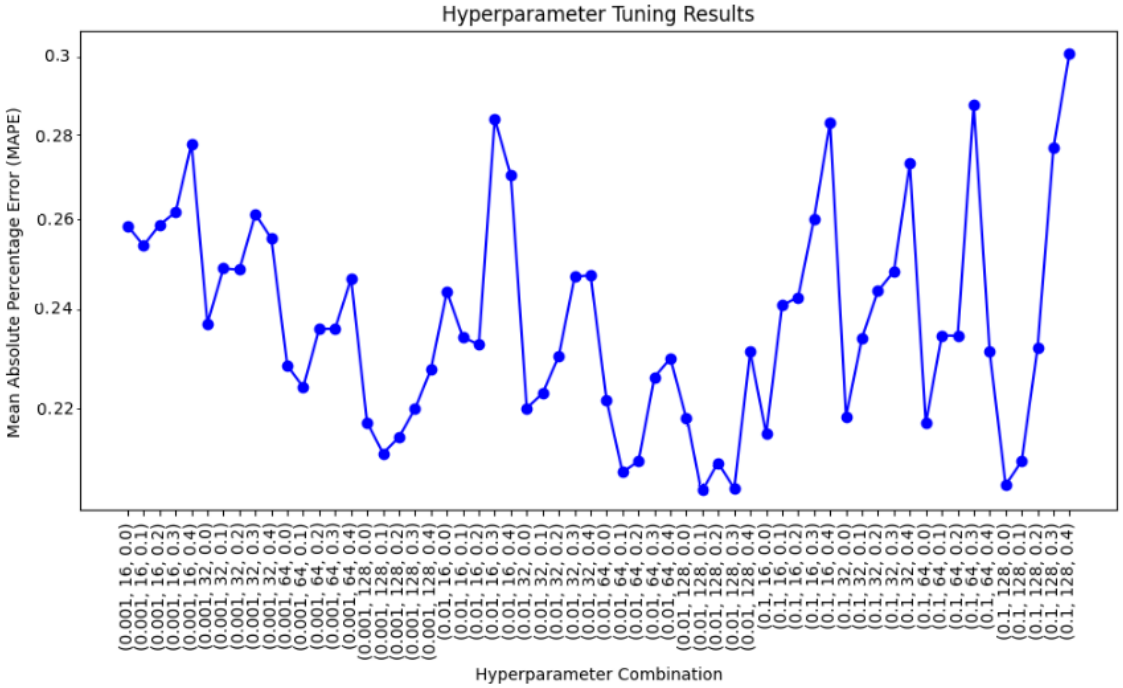

Figura 4.1: Resultados para la optimización de la combinación de los hiperparámetros `learnig_rate`, `num_units`, y `dropout_rate` para el modelo de redes neuronales, aplicado al dataset urbano.

Como puede verse en el gráfico, el mejor resultado se obtuvo para la combinación de `learning_rate` $\eta = 0.01$, `num_units` $= 128$ neuronas por capa oculta y `dropout` $p = 0.1$ para las capas que lo incluían.

A continuación, y usando los valores recién determinados, se exploraron diferentes optimizadores y se obtuvieron los resultados que se presentan en la figura 4.2. Es evidente que el mejor algoritmo de descenso por el gradiente para este problema y esta arquitectura fue 'adam' y por lo tanto es el que utilizaremos de aquí en adelante. Aclaramos aquí que los hiperparámetros del método 'adam', a excepción del `learning_rate`, tomaron los valores utilizados por las librerías por defecto.

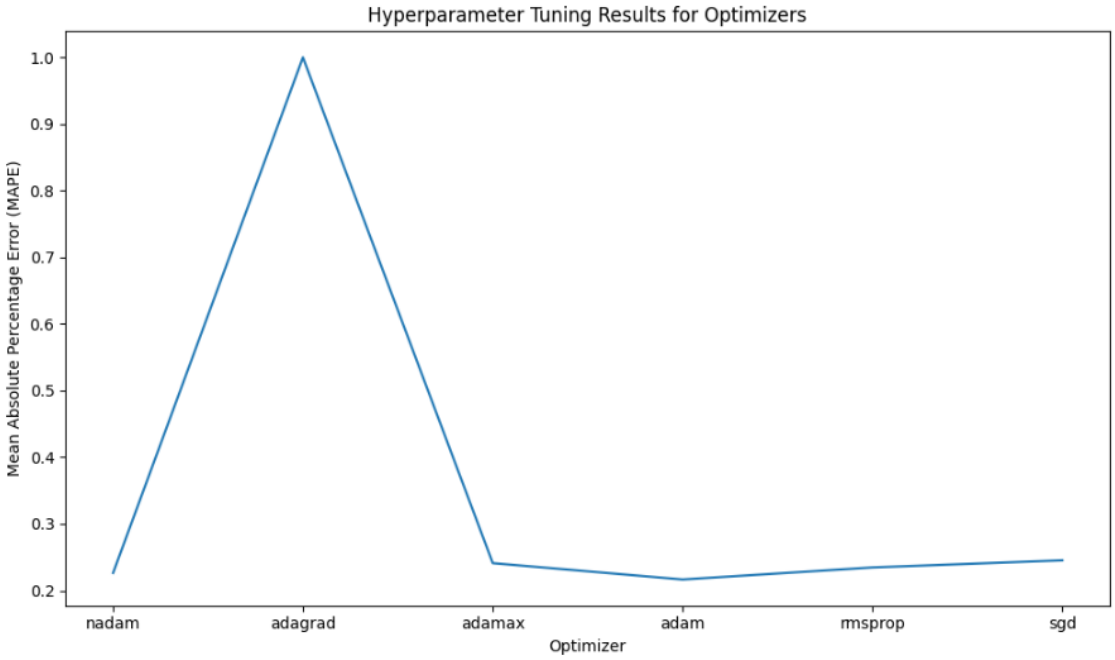

Figura 4.2: Resultados para la optimización del hiperparámetro `optimizer` en el modelo de redes del dataset urbano.

Por último, los resultados de la optimización para las funciones de activación fueron los siguientes:

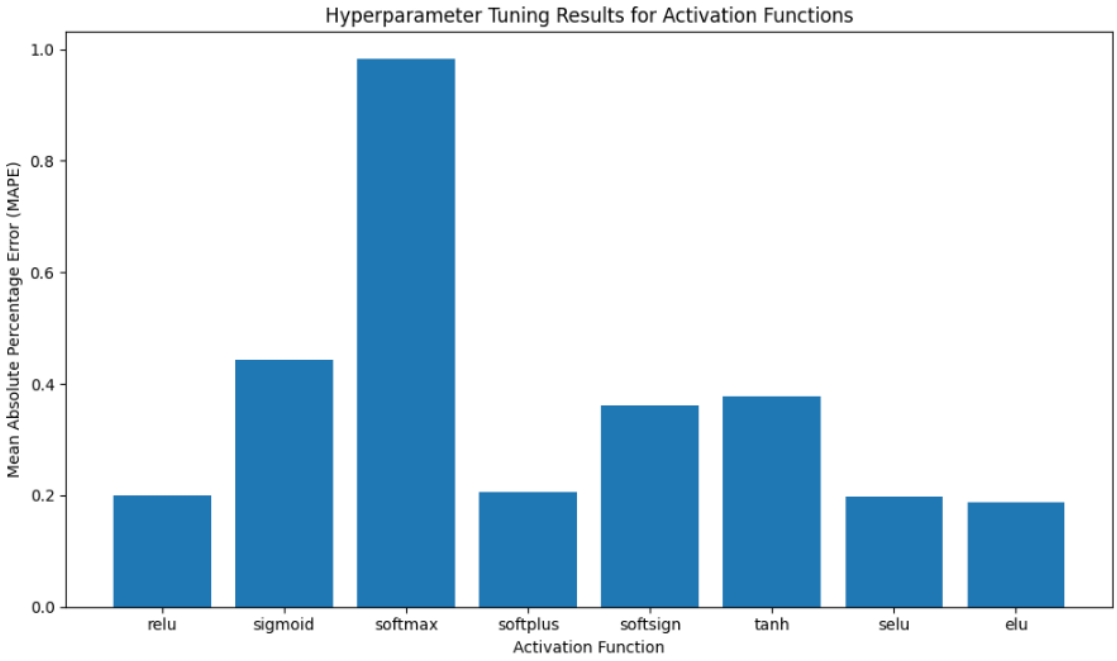

Figura 4.3: Resultados de la optimización para las funciones de activación en el modelo de redes del dataset urbano.

Por lo tanto, en el modelo final utilizamos la función de activación ELU, o "Exponential Linear Unit", que se desenvolvió ligeramente mejor que otras como RELU (Rectified

Exponential Linear Unit, por sus siglas en inglés) y SELU (Scaled Exponential Linear Unit)

Una vez realizadas las optimizaciones recientemente descriptas se procedió a entrenar nuestro mejor modelo de red neuronal feed-forward profunda. A continuación recordamos la arquitectura final y el conjunto de hiperparámetros:

Parámetro	**Valor**
Número de capas ocultas de la red	5
Número de neuronas por capa (num_units)	128 en c/u de las 5 capas ocultas
Tasa de aprendizaje (learning_rate) η	0.01
Dropout rate p	0.1
Tamaño de los lotes (batch_size)	256
Funciones de activación en las capas ocultas	*ELU*
Optimizador	'adam'
Función de pérdida (loss)	MAPE

Tabla 4.1: Arquitectura y configuración de la red neuronal final para los datos urbanos.

En la figura 4.4 presentamos la curva de la función de pérdida (loss) a lo largo de 396 épocas. Es importante destacar que si bien la curva de entrenamiento continua decreciendo, la curva de validación declina muy lentamente a partir de las $50 - 100$ épocas. Sin embargo, mencionamos que los modelos están definidos de manera tal que el entrenamiento se detiene luego de una cierta cantidad de épocas sin mejoras, guardando los mejores pesos del modelo de manera que se evita evaluarlo con pesos posteriores no tan eficientes (se aplica una función conocida como early stopping). Para este caso particular, notamos que los mejores resultados se obtuvieron alrededor de la época 280, obteniendo un MAPE del $19,23\%$. En definitiva, este es nuestro mejor resultado obtenido utilizando redes neuronales para el problema de tasación de bienes inmuebles urbanos en las ciudades consideradas.

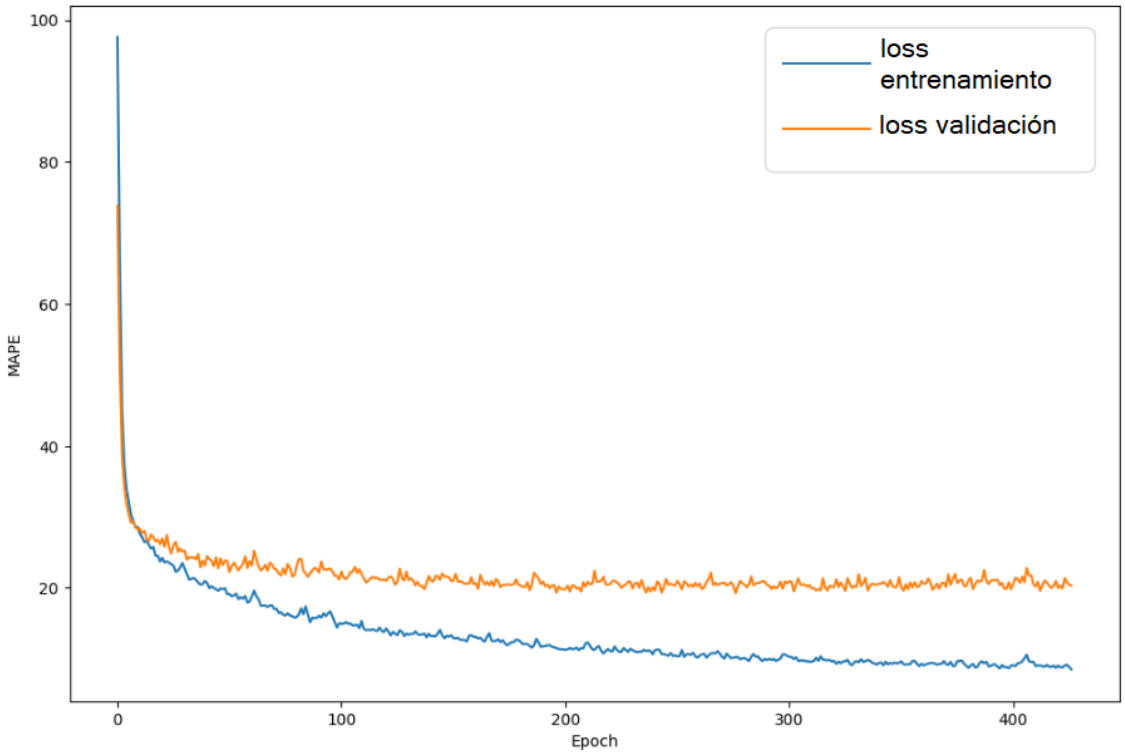

Figura 4.4: Gráfico del historial de entrenamiento y validación del modelo óptimo de redes neuronales para el dataset urbano.

A modo de comparación y antes de pasar a las otras técnicas, mostramos un gráfico obtenido con el mismo método pero para un modelo no optimizado, en la figura 4.5. Si bien en este caso también se usaron cinco capas ocultas y una única neurona lineal de salida, se usaron 64 neuronas por capa oculta, un `learning rate` de $\eta = 0.001$ y funciones de activación *ReLU*. El valor del `dropout` es el mismo utilizado en el modelo optimizado. En este caso el mejor resultado obtenido para el error MAPE fue del $25,79\%$.

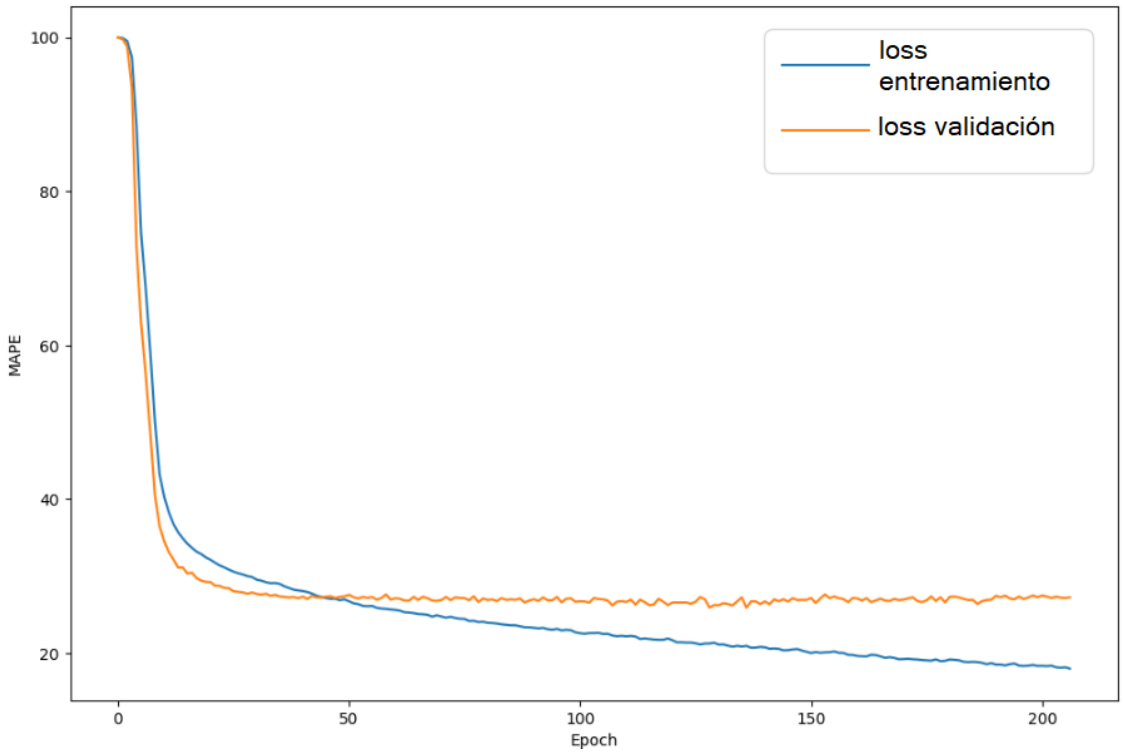

Figura 4.5: Gráfico del historial de entrenamiento y validación de un modelo previo a la optimización de redes neuronales para el dataset urbano.

Además de las redes neuronales, como ya vimos, este estudio también incorporó métodos basados en árboles de decisión, como parte de un enfoque integral. Si bien la atención principal se centró en el desarrollo y optimización de las redes neuronales, los métodos basados en árboles jugaron un papel complementario significativo en nuestra investigación. Para estos métodos, se adoptó un proceso de optimización más selectivo, enfocado en alcanzar un equilibrio entre la eficiencia computacional y la efectividad del modelo.

En el caso de XGBoost, se realizó una optimización un poco más exhaustiva debido a su robustez y capacidad para manejar grandes conjuntos de datos de manera eficiente. Se ajustaron parámetros como la tasa de aprendizaje, el número de árboles y la profundidad máxima de estos, para maximizar el rendimiento del modelo. Para esta optimización se obtuvieron los siguientes resultados:

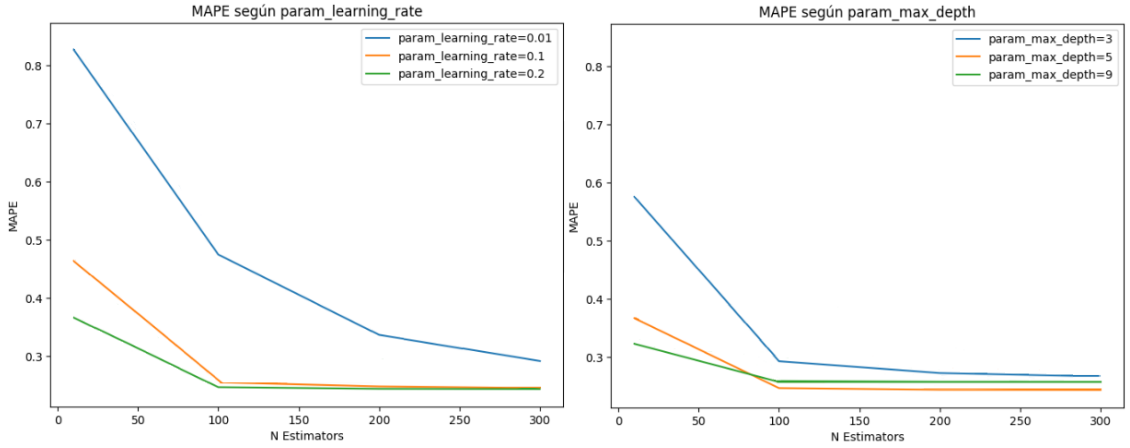

Figura 4.6: Gráfico del resultados para la optimización de los hiperparámetros para XGBoost en el dataset urbano.

Para los otros métodos basados en árboles, aunque la optimización no fue tan profunda como en XGBoost, se aseguró que los parámetros seleccionados fueran adecuados y eficientes, conduciendo a resultados satisfactorios y competitivos. Esta estrategia permitió aprovechar las ventajas inherentes de estos métodos, como su facilidad de interpretación y su capacidad para manejar diferentes tipos de características de datos, proporcionando así un enfoque complementario valioso a las redes neuronales.

Entonces, con todos los modelos optimizados, se obtuvieron los siguientes resultados para el error MAPE de nuestras predicciones en el dataset urbano:

Tabla 4.2: Resultados de MAPE para los modelos en datos urbanos (RN: Redes Neuronales, RF: Random Forest, QRF: Quantile Random Forest, XGB: XGBoost

	RN	**RF**	**QRF**	**XGB**
MAPE	19.23%	23.86%	Cuantil 0.25: 23.68% Cuantil 0.50: 21.92% Cuantil 0.85: 41.78%	20.07%

Destacamos que, para el modelo de Random Forest, aplicamos feature importance (o importancia de variables) para establecer cuáles eran las variables que más afectaban a la toma de decisiones del modelo, obteniendo las cinco principales variables:

Tabla 4.3: Importancia de las cinco características más relevantes del modelo Random Forest en dataset urbano

Feature	**Importancia**
`prom_lote`	0.4443
`perc_val_urb`	0.1049
`dens_osm`	0.0809
`fot`	0.0491
`oferta_inm`	0.0398

A modo de recordatorio rápido, mencionamos qué representa cada una de estas cinco

características: `prom_lote` es el promedio de superficie de lote en un radio de 500m, `perc_val_urb` es el porcentaje de parcelas con valuación urbana en un radio de 500 m, `dens_osm` representa el promedio de densidad de calles ponderadas de OpenStreetMaps, `fot` es el factor de ocupación total, y `oferta_inm` es la oferta inmobiliaria en el entorno.

4.4.2 Datos Rurales

Tras haber examinado los resultados de optimización de hiperparámetros para el modelo aplicado a datos urbanos, procedemos ahora a explorar las particularidades y hallazgos relevantes obtenidos del análisis de los datos rurales. En esta sección, presentamos los gráficos y análisis derivados de la optimización de nuestro modelo de redes neuronales, adaptado específicamente al conjunto de datos rurales.

Para la combinación de los hiperparámetros `learnig_rate`, `num_units`, y `dropout_rate`, obtuvimos:

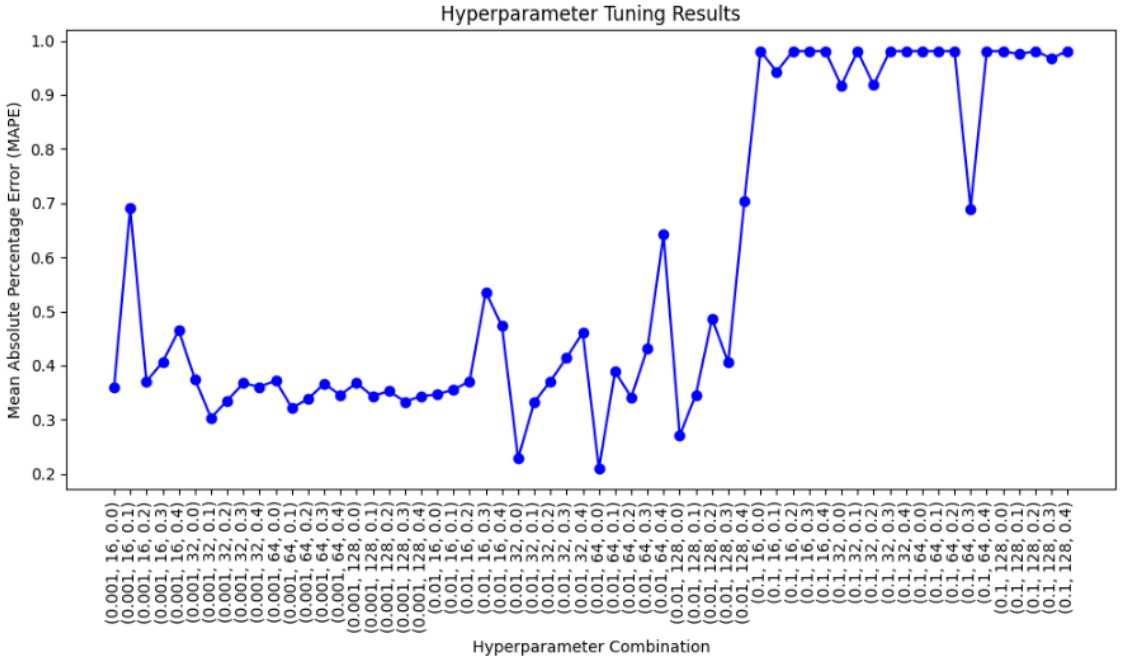

Figura 4.7: Resultados para la optimización de la combinación de los hiperparámetros `learnig_rate`, `num_units`, y `dropout_rate` para el modelo de redes neuronales, aplicado al dataset rural.

Como puede verse en el gráfico, el mejor resultado se obtuvo para la combinación de `learning_rate` $\eta = 0.01$, `num_units` $= 64$ neuronas por capa oculta y `dropout` $p = 0.0$.

Para el resto de los hiperparámetros evitamos presentar los gráficos, puesto que son muy similares a los del dataset urbano. En cambio, queremos hacer notar la diferencia entre un modelo no tan optimizado respecto al MAPE, y su diferencia con el modelo final, con todos los hiperparámetros optimizados, en el que se obtiene mucha mayor estabilidad a lo largo de las épocas y además, valores menores de MAPE.

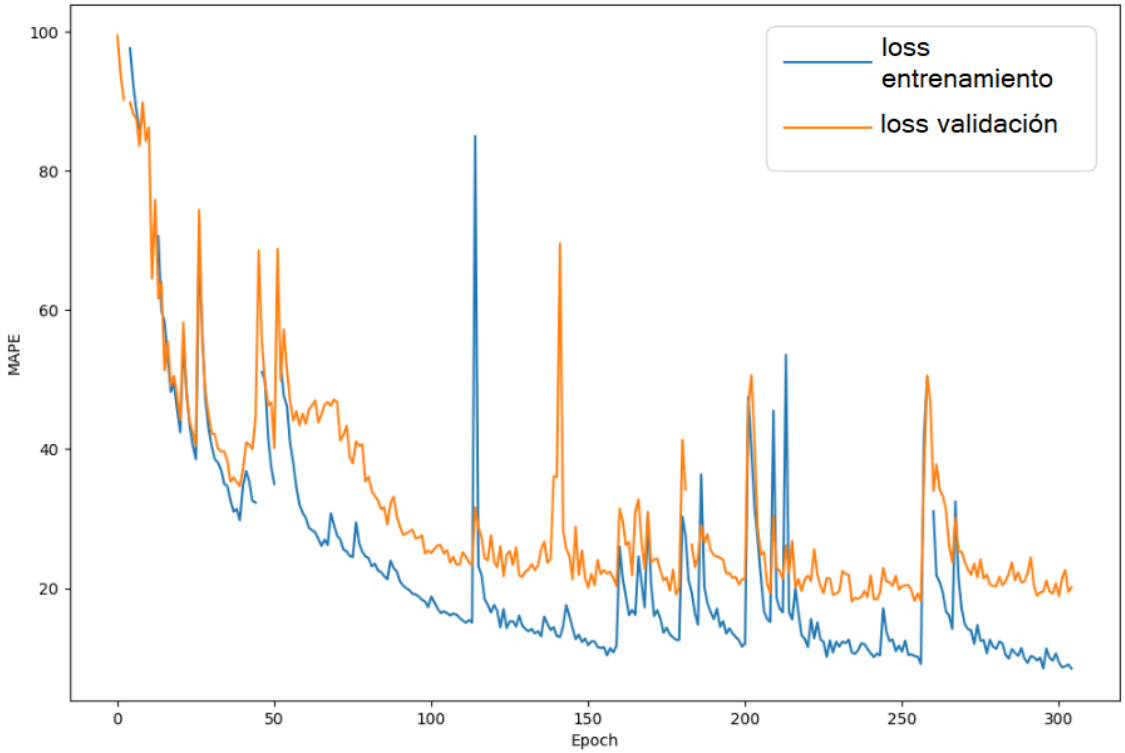

Figura 4.8: Gráfico del historial de entrenamiento y validación de un modelo de redes neuronales en un punto medio de optimización para el dataset rural.

En el modelo de la figura 4.8 se utilizaron también cinco capas ocultas y una única neurona lineal de salida, y se usaron 32 neuronas por capa oculta, un `learning rate` de $\eta = 0.01$ y funciones de activación $ReLU$. El valor del `dropout` fue de 0.0, el mismo que en el modelo final. En este caso el mejor resultado obtenido para el error MAPE fue del $17,89\%$. Para compararlo con el modelo final, veremos la figura 4.9, donde presentamos la curva de la función de pérdida (loss) a lo largo de aproximadamente 300 épocas. Para este caso particular, y recordando que el entrenamiento se detiene luego de una cierta cantidad de épocas sin mejoras, notamos que los mejores resultados se obtuvieron alrededor de la época 270, obteniendo un MAPE del $15,56\%$. En definitiva, este es nuestro mejor resultado obtenido utilizando redes neuronales para el problema de tasación de bienes inmuebles rurales en la provincia de Córdoba:

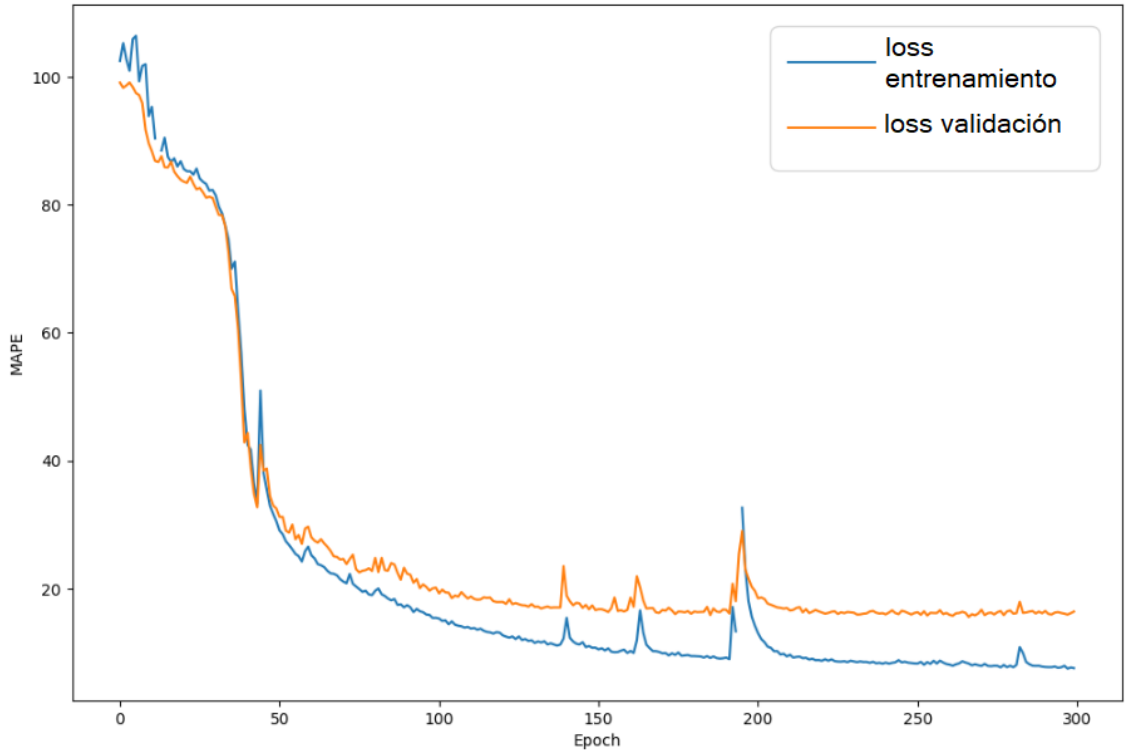

Figura 4.9: Gráfico del historial de entrenamiento y validación de un modelo óptimo de redes neuronales para el dataset rural.

Además, aquí puede verse esa diferencia de estabilidad anteriormente mencionada. Sin embargo, puesto que los modelos están definidos para guardar los mejores pesos a lo largo del entrenamiento, esto tampoco hubiera resultado un mayor problema.

Antes de pasar a los otros modelos, describimos a continuación la arquitectura y el conjunto de hiperparámetros de nuestro modelo final optimizado:

Parámetro	**Valor**
Número de capas ocultas de la red	5
Número de neuronas por capa (num_units)	64 en c/u de las 5 capas ocultas
Tasa de aprendizaje (learning_rate) η	0.01
Dropout rate p	0.0
Tamaño de los lotes (batch_size)	128
Funciones de activación en las capas ocultas	ELU
Optimizador	'adam'
Función de pérdida (loss)	MAPE

Tabla 4.4: Arquitectura y configuración de la red neuronal final para los datos rurales.

Además, para los modelos basados en árboles, se adoptó un proceso de optimización más selectivo, enfocado en alcanzar un equilibrio entre la eficiencia computacional y la efectividad del modelo. Sin embargo, notamos que, a diferencia de la sección sobre los datos urbanos, XGBoost nos dio resultados mucho peores que con cualquiera de los restantes métodos, y por lo tanto no explicitaremos el gráfico de su optimización, puesto que no lo

consideramos necesario.

Finalmente, tras la cuidadosa selección y optimización de los mejores hiperparámetros para nuestros modelos, procedimos a evaluar su desempeño en el conjunto de datos rurales. Los resultados obtenidos se centran en el Error Porcentual Absoluto Medio (MAPE) de nuestras predicciones, proporcionando una medida cuantitativa de la precisión del modelo. Los resultados fueron los siguientes:

Tabla 4.5: Resultados de MAPE para los modelos aplicados a los datos rurales (RN: Redes Neuronales, RF: Random Forest, QRF: Quantile Random Forest, XGB: XGBoost

	RN	**RF**	**QRF**	**XGB**
MAPE	15.56%	19.18%	Cuantil 0.25: 8.87% Cuantil 0.50: 8.63% Cuantil 0.85: 32.22%	490%

Notamos que, en general, los modelos se desempeñaron un poco mejor que en el dataset urbano, a excepción de XGBoost, que fue extremadamente malo. Este mejor desempeño (en general) quizá se debe al hecho de que no tuvimos que realizar el análisis inflacionario, dándole mayor estabilidad a los datos.

Destacamos que, nuevamente, para el modelo de Random Forest aplicamos feature importance (o importancia de variables) para establecer cuáles eran las variables que más afectaban a la toma de decisiones del modelo, obteniendo las cinco principales variables:

Tabla 4.6: Importancia de las cinco características más relevantes del modelo Random Forest en dataset rural

Feature	**Importancia**
`vur_2020`	0.2780
`n2_cob22`	0.2012
`vur_2019`	0.0815
`rec_1median`	0.0534
`rto_sj1718`	0.0519

Nuevamente, a modo de recordatorio, mencionamos qué representa cada una de estas cinco características: `vur_2019` y `vur_2020` son los valores unitarios de la tierra rural en 2019 y 2020, `n2_cob22` es el porcentaje de superficie de cobertura leñosa afectada por incendio, `rec_1median` es la mediana dentro de la celda territorial, y `rto_sj1718` representa los cálculos finales de la campaña 17/18 de la Bolsa de Cereales de Córdoba para el rendimiento zonal de soja.

Parte III

Discusión

Discusión y Conclusiones

En el contexto actual, marcado por la vorágine de datos y la necesidad de decisiones basadas en información precisa y actualizada, la tasación de terrenos emerge como un dominio crítico en el sector inmobiliario y fiscal. El estudio presente se sumergió en este desafío, explorando la confluencia entre el conocimiento tradicional y las capacidades emergentes de la inteligencia artificial. A lo largo de esta investigación, hemos desplegado un arsenal de técnicas de aprendizaje automático para dilucidar y predecir el valor del suelo en la provincia de Córdoba, enfrentándonos al reto de transformar un proceso históricamente artesanal en una práctica objetiva y automatizada.

La valiosa contribución de esta tesis reside en su enfoque completo hacia la tasación automatizada, sustentada en la aplicación y comparación meticulosa de redes neuronales frente a modelos consolidados como Random Forest, Quantile Random Forest y XGBoost. A pesar de que estudios recientes, incluyendo el trabajo referenciado en la introducción de esta tesis [6], sugieren una supremacía de los modelos basados en árboles para el manejo de datos tabulares, nuestros hallazgos narran una historia distinta. En los datos trabajados en este trabajo, las redes neuronales no solo han igualado, sino que en varios aspectos, han superado el rendimiento de sus contrapartes basadas en árboles, una revelación que desafía la noción preconcebida de su inferioridad en este ámbito.

En el conjunto de datos urbanos, la brecha entre las redes neuronales y XGBoost fue marginal, menos del 1% en términos de precisión. Esta mínima diferencia, aunada al significativo ahorro de tiempo computacional ofrecido por XGBoost, plantea un argumento robusto a favor de este último cuando se requiere eficiencia además de precisión. Sin embargo, en el ámbito rural, la red neuronal no solo mantuvo su superioridad al mejorar el desempeño de Random Forest por más del 3,5%, sino que también se destacó frente al Quantile Random Forest, especialmente en los cuantiles superiores. Esto indica que, si bien los métodos basados en cuantiles pueden ofrecer resultados prometedores en ciertos segmentos del mercado, las redes neuronales presentan una ventaja en la captura de la complejidad inherente a la valoración de terrenos rurales.

La importancia asignada a las variables en los modelos de Random Forest arroja luz sobre los factores que más influyen en la valoración de terrenos. En el caso rural, los precios de años anteriores y la cobertura leñosa afectada por incendios emergieron como determinantes significativos, reflejando la realidad de una provincia frecuentemente azotada por incendios y la relevancia histórica de los valores de mercado. En los datos

urbanos, el promedio de superficie de lote y el porcentaje de valor urbano, entre otros, fueron preponderantes, destacando la importancia de la ubicación y el contexto urbano en la valoración de propiedades.

Estos resultados no solo validan la capacidad de las redes neuronales para modelar la complejidad y la heterogeneidad de los datos inmobiliarios, sino que también abren la puerta a una nueva era en la tasación de terrenos, una donde la rapidez y la objetividad de la inteligencia artificial pueden coexistir armoniosamente con la sabiduría y la intuición humana. Este trabajo, por tanto, no solo proporciona una base para futuras investigaciones, sino que también ofrece una herramienta práctica y poderosa para tasadores, inversionistas y responsables de políticas públicas, quienes podrán tomar decisiones más informadas y eficientes.

La discusión no estaría completa sin reconocer las limitaciones y los caminos futuros. La adaptabilidad y el ajuste de los modelos de aprendizaje automático dependen en gran medida de la calidad y la granularidad de los datos disponibles. Asimismo, la interpretación de los modelos complejos como las redes neuronales sigue siendo un área que requiere un desarrollo considerable, especialmente cuando se busca un equilibrio entre la precisión y la explicabilidad. Además, la colaboración entre los registros catastrales y la infraestructura de datos espaciales que se está llevando a cabo en la provincia de Córdoba, es fundamental para proporcionar un entorno fértil donde estos modelos puedan ser entrenados y aplicados con eficacia.

Mirando hacia el futuro, se vislumbra un campo de posibilidades en expansión. La investigación futura podría explorar la integración de modelos más sofisticados, por ejemplo algoritmos convolucionales que consideren la distancia geográfica entre los datos, así como la incorporación de nuevas fuentes de datos. Además, la expansión del alcance geográfico de los modelos y la exploración de la dinámica entre los mercados urbanos y rurales ofrecerían perspectivas más amplias y generalizables.

En resumen, este trabajo ha demostrado que, en el terreno de la tasación de bienes raíces en Córdoba, las redes neuronales son no solo viables sino, en muchos casos, preferibles a los métodos tradicionales. A medida que avanzamos en la era de la información, estos hallazgos fortalecen la convicción de que la inteligencia artificial es un aliado invaluable en la evolución de prácticas ancestrales hacia horizontes modernos y eficientes. Con cada algoritmo entrenado y cada predicción realizada, nos acercamos un paso más a un mundo donde el valor de cada metro cuadrado de terreno no es solo una cifra calculada, sino un testimonio de la confluencia entre el conocimiento humano y la precisión de la máquina.

Bibliografía

[1] E. Agliari, A. Barra, P. Sollich, and L. Zdeborová, "Machine learning and statistical physics: Preface," *Journal of Physics A: Mathematical and Theoretical*, vol. 53, no. 50, p. 500 401, Nov. 2020. DOI: 10.1088/1751-8121/abca75. [Online]. Available: https://dx.doi.org/10.1088/1751-8121/abca75.

[2] M. Gabrié, "Mean-field inference methods for neural networks," *Journal of Physics A: Mathematical and Theoretical*, vol. 53, no. 22, p. 223 002, May 2020. DOI: 10.1088/1751-8121/ab7f65. [Online]. Available: https://dx.doi.org/10.1088/1751-8121/ab7f65.

[3] L.-M. Gao Xun; Duan, "Efficient representation of quantum many-body states with deep neural networks," *Nature Communications*, vol. 8, no. 1, pp. 2041–1723, Sep. 2017. DOI: 10.1038/s41467-017-00705-2. [Online]. Available: https://doi.org/10.1038/s41467-017-00705-2.

[4] R. E. F. y Ángel C. Cervini, *Antecedentes para el estudio de normas para tasaciones urbanas en capital federal*, 1939.

[5] J. P. Carranza, M. A. Piumetto, C. M. Lucca, and E. Da Silva, "Mass appraisal as affordable public policy: Open data and machine learning for mapping urban land values," *Land Use Policy*, vol. 119, p. 106 211, 2022, ISSN: 0264-8377. DOI: https://doi.org/10.1016/j.landusepol.2022.106211.

[6] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. [Online]. Available: https://openreview.net/forum?id=Fp7__phQszn.

[7] Infraestructura de Datos Espaciales de la Provincia de Córdoba y Dirección General de Catastro de la Provincia de Córdoba, *Observatorio del mercado inmobiliario*. [Online]. Available: https://omi.mapascordoba.gob.ar/spa/#/indicadores.

[8] M. E. Bullano, J. P. Carranza, M. A. Piumetto, R. M. Cerino, F. Monzani, and M. A. Cordoba, *El impacto de las variaciones del tipo de cambio sobre el valor de la tierra urbana. ¿el mercado inmobiliario est´a totalmente dolarizado?* 2020.

[9] B. W. Y. Michael W. Berry Azlinah Mohamed, *Supervised and Unsupervised Learning for Data Science*. Springer Cham, 2020. DOI: 10.1007/978-3-030-22475-2.

[10] A. G. Barto, "Chapter 2 - reinforcement learning," in *Neural Systems for Control*, O. Omidvar and D. L. Elliott, Eds., San Diego: Academic Press, 1997, pp. 7–30, ISBN: 978-0-12-526430-3. DOI: https://doi.org/10.1016/B978-012526430-3/50003-9.

[11] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, *Mathematics for Machine Learning*. Cambridge University Press, 2020. DOI: 10.1017/9781108679930.

[12] S. P. Yadav, A. Gupta, C. Dos Santos Nascimento, V. Hugo C. de Albuquerque, M. S. Naruka, and S. Singh Chauhan, "Voice-based virtual-controlled intelligent personal assistants," in *2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, 2023, pp. 563–568. DOI: 10.1109/CICTN57981.2023.10141447.

[13] K. G. Al-Hashedi and P. Magalingam, "Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019," *Computer Science Review*, vol. 40, p. 100 402, 2021, ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2021.100402.

[14] P. K. Roy, S. S. Chowdhary, and R. Bhatia, "A machine learning approach for automation of resume recommendation system," *Procedia Computer Science*, vol. 167, pp. 2318–2327, 2020, International Conference on Computational Intelligence and Data Science, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.03.284.

[15] J. Frost, *When should i use regression analysis?* 2017. [Online]. Available: https://statisticsbyjim.com/regression/when-use-regression-analysis/.

[16] J. Sato, S. Saito, H. Jonokoshi, K. Nishikawa, and F. Goto, "Correlation and linear regression between blood pressure decreases after a test dose injection of propofol and that following anaesthesia induction," *Anaesthesia and Intensive Care*, vol. 31, no. 5, pp. 523–528, 2003, PMID: 14601275. DOI: 10.1177/0310057X0303100506. eprint: https://doi.org/10.1177/0310057X0303100506. [Online]. Available: https://doi.org/10.1177/0310057X0303100506.

[17] F. Rosenblatt, "The perceptron — a perceiving and recognizing automaton," *Report 85-460-1, Cornell Aeronautical Laboratory*, 1957.

[18] S. A. P. Marvin Minsky, *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969.

[19] H. John, "Neural networks and physical systems with emergent collective computational abilities," *PNAS 79 (8) 2554-2558*, 1982. DOI: 10.1073/pnas.79.8.2554.

[20] R. Rumelhart D.E. Hinton G.E.and Williams, "Learning internal representations by error propagation," *Nature, 323, 533–536*, 1986.

[21] C. Djellali and M. adda, "A new hybrid deep learning model based-recommender system using artificial neural network and hidden markov model," *Procedia Computer Science*, vol. 175, pp. 214–220, 2020, The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC),The 15th International Conference on Future Networks and Communications (FNC),The 10th International Conference on Sustainable Energy Information Technology, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.07.032. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920317129.

[22] B. W. Shahid N Rappon T, "Applications of artificial neural networks in health care organizational decision-making: A scoping review.," *PLoS One*, 2019. DOI: 10.1371/journal.pone.0212356.

[23] P. A. e. a. Jumper J. Evans R., "Highly accurate protein structure prediction with alphafold," *Nature*, pp. 583–589, 2021. DOI: https://doi.org/10.1038/s41586-021-03819-2.

[24] L. Breiman, *Random forests*, 2001. [Online]. Available: https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf.

[25] W. Y. James Robert; Leung, "Forecasting financial risk using quantile random forests," 2023, SSRN. DOI: http://dx.doi.org/10.2139/ssrn.4324603. [Online]. Available: https://ssrn.com/abstract=4324603.

[26] D. Nielsen, "Tree boosting with xgboost - why does xgboost win "every" machine learning competition?," 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:114191144.

[27] M. Kearns, "Thoughts on hypothesis boosting," *University of Pennsylvania*, 1988.

[28] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, *A comparative analysis of xgboost*, Nov. 2019.

[29] C. Kwenda, M. V. Gwetu, and J. V. Fonou-Dombeu, "Forest image classification based on deep learning and xgboost algorithm," in *Computational Science – ICCS 2023*, J. Mikyška, C. de Mulatier, M. Paszynski, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. Sloot, Eds., Cham: Springer Nature Switzerland, 2023, pp. 217–229, ISBN: 978-3-031-36027-5.

[30] Instituto Geográfico Nacional de la República Argentina, *Definición de Sistemas de Coordenadas y Proyecciones Oficiales (EPSG)*, 2017. [Online]. Available: https://ramsac.ign.gob.ar/posgar07_pg_web/documentos/Informe_sobre_codigos_oficiales_EPSG.pdf.

[31] Y. D. et al, "A high accuracy map of global terrain elevations," *Geophysical Research Letters*, pp. 5844–5853, 2017. DOI: 10.1002/2017GL072874.

[32] Administración Provincial de Recursos Hídricos, *Portal de información hídrica de córdoba*. [Online]. Available: https://portal-aprhi.opendata.arcgis.com/.

Los abajo firmantes, miembros del Tribunal de evaluación de tesis,
damos fe que el presente ejemplar impreso se corresponde con el aprobado por este Tribunal.